University of Bath

PHD

The Crossroads of Categorical Algebra and Game Semantics

An investigation into the application of Kleisli categories and related constructions to the study of Full Abstraction for nondeterministic effects in Algol-like languages

Gowers, William John

Award date:
2020

Awarding institution:
University of Bath

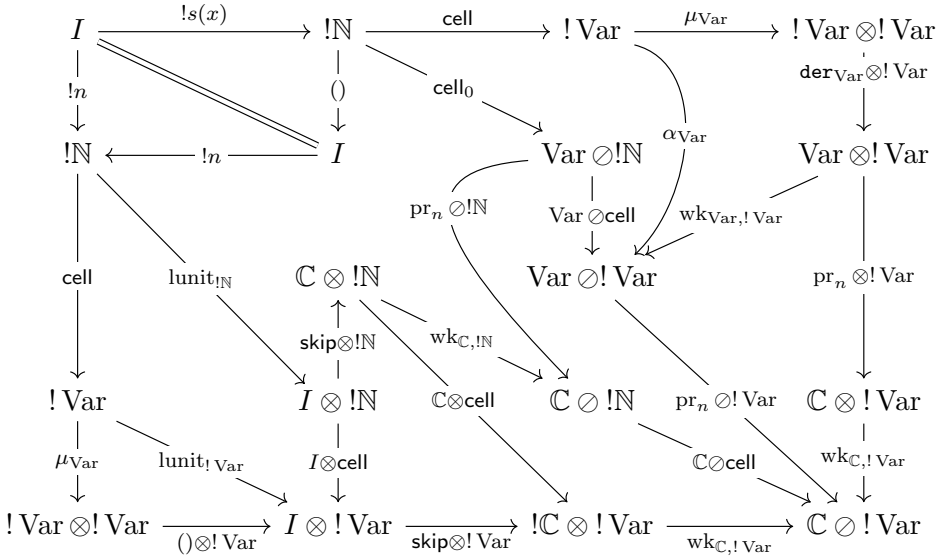
The Crossroads of Categorical Algebra and Game Semantics

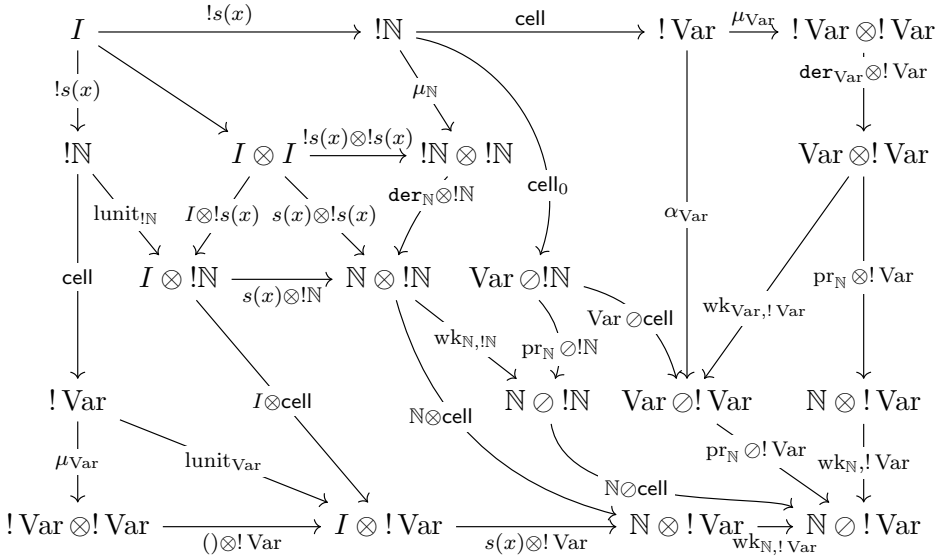
An investigation into the application of Kleisli categories and related constructions to the study of Full Abstraction for nondeterministic effects in Algol-like languages

William John Gowers

A thesis submitted for the degree of
Doctor of Philosophy

Department of Computer Science
University of Bath
United Kingdom
August 2019


I
$!s(x)$
$!\mathbb{N}$
cell
$!\,\mathrm{Var}$
μ_{Var}
$!\,\mathrm{Var} \otimes !\,\mathrm{Var}$
$!n$
$()$
cell_0
$\mathtt{der}_{\mathrm{Var}} \otimes !\,\mathrm{Var}$
α_{Var}
$\mathrm{Var} \oslash !\mathbb{N}$
$\mathrm{Var} \otimes !\,\mathrm{Var}$
$\mathrm{pr}_n \oslash !\mathbb{N}$
$\mathrm{Var} \oslash \mathsf{cell}$
$\mathrm{wk}_{\mathrm{Var},!\,\mathrm{Var}}$
$\mathrm{lunit}_{!\mathbb{N}}$
$\mathbb{C} \otimes !\mathbb{N}$
$\mathrm{Var} \oslash !\,\mathrm{Var}$
$\mathrm{pr}_n \otimes !\,\mathrm{Var}$
$\mathsf{skip} \otimes !\mathbb{N}$
$\mathrm{wk}_{\mathbb{C},!\mathbb{N}}$
$I \otimes !\mathbb{N}$
$\mathbb{C} \otimes \mathsf{cell}$
$\mathbb{C} \oslash !\mathbb{N}$
$\mathrm{pr}_n \oslash !\,\mathrm{Var}$
$\mathbb{C} \otimes !\,\mathrm{Var}$
$\mathrm{lunit}_{!\,\mathrm{Var}}$
$I \otimes \mathsf{cell}$
$\mathbb{C} \oslash \mathsf{cell}$
$\mathrm{wk}_{\mathbb{C},!\,\mathrm{Var}}$
$!\,\mathrm{Var} \otimes !\,\mathrm{Var}$
$I \otimes !\,\mathrm{Var}$
$!\mathbb{C} \otimes !\,\mathrm{Var}$
$\mathbb{C} \oslash !\,\mathrm{Var}$
$()\otimes !\,\mathrm{Var}$
$\mathsf{skip} \otimes !\,\mathrm{Var}$


I
$!s(x)$
$!\mathbb{N}$
cell
$!\,\mathrm{Var}$
μ_{Var}
$!\,\mathrm{Var} \otimes !\,\mathrm{Var}$
$\mu_{\mathbb{N}}$
$\mathtt{der}_{\mathrm{Var}} \otimes !\,\mathrm{Var}$
$I \otimes I$
$!s(x) \otimes !s(x)$
$!\mathbb{N} \otimes !\mathbb{N}$
$\mathrm{Var} \otimes !\,\mathrm{Var}$
$\mathrm{lunit}_{!\mathbb{N}}$
$I \otimes !s(x)$
$s(x) \otimes !s(x)$
$\mathtt{der}_{\mathbb{N}} \otimes !\mathbb{N}$
cell_0
α_{Var}
$I \otimes !\mathbb{N}$
$\mathbb{N} \otimes !\mathbb{N}$
$\mathrm{Var} \oslash !\mathbb{N}$
$\mathrm{pr}_{\mathbb{N}} \otimes !\,\mathrm{Var}$
$s(x) \otimes !\mathbb{N}$
$\mathrm{Var} \oslash \mathsf{cell}$
$\mathrm{wk}_{\mathrm{Var},!\,\mathrm{Var}}$
$\mathrm{wk}_{\mathbb{N},!\mathbb{N}}$
$\mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N}$
$!\,\mathrm{Var}$
$I \otimes \mathsf{cell}$
$\mathbb{N} \otimes \mathsf{cell}$
$\mathbb{N} \oslash !\mathbb{N}$
$\mathrm{Var} \oslash !\,\mathrm{Var}$
$\mathbb{N} \otimes !\,\mathrm{Var}$
$\mathrm{pr}_{\mathbb{N}} \oslash !\,\mathrm{Var}$
μ_{Var}
$\mathrm{lunit}_{\mathrm{Var}}$
$\mathrm{wk}_{\mathbb{N},!\,\mathrm{Var}}$
$\mathbb{N} \oslash \mathsf{cell}$
$!\,\mathrm{Var} \otimes !\,\mathrm{Var}$
$I \otimes !\,\mathrm{Var}$
$\mathbb{N} \otimes !\,\mathrm{Var}$
$\mathbb{N} \oslash !\,\mathrm{Var}$
$()\otimes !\,\mathrm{Var}$
$s(x) \otimes !\,\mathrm{Var}$
$\mathrm{wk}_{\mathbb{N},!\,\mathrm{Var}}$

Contents

List of Figures

Acknowledgements

There are many people and organizations to whom I owe a debt of gratitude, and without whom this thesis would not have been written. My first thanks go to EPSRC research grant EP/K018868/1 for funding my PhD, keeping me fed and housed for three and a half years, and sponsoring several trips to conferences and schools at home and abroad. I am also indebted to the organizers of those conferences and schools, including but not limited to

- Paul Blain Levy, Juha Kontinen, Marina Lenisa, Ugo dal Lago and Gabriel Sandu, chairs of various iterations of GaLoP (and, in Paul's case, organizer of the Midlands Graduate School 2016);
- Filippo Bonchi and Barbara König, organizers of CALCO 2017;
- Matthias Felleisen, Robby Findler, Matthew Flatt, Shriram Krishnamurthi, Jay McCarthy, and Justin Pombrio, teachers at the Racket Summer School 2017;
- Moshe Vardi, Daniel Kroening and Marta Kwiatkowska, co-chairs of FloC 2018; and
- Dan Ghica and Achim Jung, co-chairs of CSL 2018.

I would also like to express gratitude to those people whom I have had productive discussions with at conferences and during departmental visits: in particular, Martin Hyland, Paul-André Melliès, Paul Levy and Pierre Clairambault.

Thank you to the anonymous reviewers from CALCO 2017, LICS 2018, CSL 2018 and FoSSaCS 2019 for your careful comments, all of which have shaped this thesis in some way.

Thank you to my PhD examiners, Paul Blain Levy and Guy McCusker, for undertaking to read the first draft of this thesis (no small feat!), and for your many helpful comments and corrections.

Thank you to the whole Computer Science Department at the University of Bath, particularly the Mathematical Foundations team, for all your help over the years. Congratulations to John Power on the conclusion of a momentous career, and thanks for helping me with my category theory. Thank you too to Ben, David and Alessio for organizing the Departmental Seminar at Bath and for being such good colleagues.

Thank you to my family for all your support during my PhD. Thank you to Ellie for your patience and love, and for enduring a three-year long-distance relationship while I was in Bath. This thesis is dedicated to you.

Lastly, there would be no thesis at all without my PhD supervisor, Jim

Laird, who took me on when I had little to no knowledge of the subject and has carefully guided me throughout. That I am able to submit this now is testament to Jim's compendious knowledge of the subject and his patience in imparting some of it to me. At all stages of my PhD, I have been able to rely on our weekly meetings for insight when I was stuck or unsure about the correct direction to investigate. In addition, Jim often encouraged me to submit papers to workshops and conferences, even in the early stages of my PhD, and the experience of doing so has shaped the direction of my research, and helped me form valuable relationships with other researchers.

Abstract

Starting with a Fully Abstract denotational semantics for an Algol-like programming language, we investigate how we can use techniques of categorical algebra to adapt the semantics when the original language is extended with various nondeterministic effects. Our running example for the base denotational semantics is Abramsky and McCusker's game semantics for Idealized Algol [AM96]. We give a full presentation of this game semantics, including an alternative proof of Computational Adequacy that uses Laird's concept of a sequoidal category [Lai02] rather than the combinatorial proof from Abramsky and McCusker's original paper.

We introduce the familiar concepts of monads and Kleisli categories, and prove a Full Abstraction result that shows how the process of passing to certain Kleisli categories corresponds to particular language extensions. We link these language extensions to may- and must-testing for finite and countable nondeterminism, showing how we can construct Fully Abstract models of these effects using Kleisli categories.

We introduce a generalization of monads: lax actions, also known as parametric monads. We investigate various corresponding generalizations of Kleisli categories, and prove a Full Abstraction result for one such generalization, showing how it too can be used to construct a model of an extended version of our original Algol-like language. We show how a special case of this construction can be adapted to model a probabilistic language.

Chapter 1

Introduction

1.1 Denotational Semantics and Program Equivalence

Given two pieces of computer code, in what circumstances can we say that they are interchangeable? I.e., when can we be absolutely sure that we can replace one with the other and know that the behaviour of the program will not change? This is a simple question with a complicated answer.

A starting point is to require that the two pieces of code should return the same output for any choice of input values. But that is not necessarily enough. For example, the following two Haskell functions appear to do the same thing: indeed, both return 0, whatever input is passed in.

```
f :: Int -> Int
f n = if (n == 0) then 0 else 0

g :: Int -> Int
g n = 0
```

However, if we introduce a non-terminating function

```
diverge :: Int -> Int
diverge x = diverge x
```

then it becomes clear that f and g are not interchangeable: indeed, since Haskell evaluates inputs to functions lazily, evaluating g (diverge 0) will not evaluate the non-terminating term (diverge 0) and will terminate at the value 0, while f (diverge 0) *will* evaluate (diverge 0) and will itself fail to converge.

This is not itself a difficult problem to get around: all we have to do is treat non-termination as a separate input value in itself. Thus, we add to each datatype an extra distinguished value $\perp$ representing non-termination (so, for example, the type of integers is represented by the set $\mathbb{Z}_\perp = \mathbb{Z} + \{\perp\}$), and then function types are interpreted as functions between these sets – so a term of type **Int** −> **Int** is represented by a function $\mathbb{Z}_\perp \to \mathbb{Z}_\perp$. We can then tell that our functions f and g are different, since g corresponds to the constant 0 function $\mathbb{Z}_\perp \to \mathbb{Z}_\perp$, while f corresponds to the function that sends $n \in \mathbb{Z}$ to 0 but sends $\perp$ to $\perp$.

Every program gives rise to a function in this way, but not every such function arises from a program. For example, we cannot write a program corresponding to the function $\chi\colon \mathbb{Z}_\perp \to \mathbb{Z}_\perp$ that sends $\perp$ to 0 and all other values to 1: since χ is not constant, such a program would have to evaluate its argument, and would consequently fail to terminate if that argument did not terminate, so it would have to send $\perp$ to $\perp$.

If our language admits higher types, then it becomes especially important to exclude such 'impossible' functions from our model. For example, if F and G are two programs of type (**Int** −> **Int**) −> **Int** – i.e., functions that take in a function from integers to integers and return an integer – then we do not want to declare that F and G are different on the basis that $F(\chi) \neq G(\chi)$.

In order to rule out these 'impossible' functions, we define a partial order on the sets $T_\perp$ corresponding to types, defined by setting $x \leq x$ and $\perp \leq x$ for all x. We then require that functions should be monotonic and continuous with respect to this order. For example, a monotonic function $\mathbb{Z}_\perp \to \mathbb{Z}_\perp$ is either constant (corresponding to a function that does not evaluate its argument at all) or sends $\perp$ to $\perp$ and is otherwise unconstrained (corresponding to a function that evaluates its argument). It turns out that if we order functions pointwise, then we get the correct constraints at higher types as well.

Even if we get round the problems with divergence, there are other language features that we may need to consider if we want to determine whether two pieces of code are equivalent. If our functions have access to global variables, then we need to check that these variables end up taking the same final value, whatever their initial values were. If we have IO calls in our language, then we need to check that the functions print out the same text, whatever the user input was. If we have a random number generator, then we need to check that our functions return the same *set* of values, whatever the input.

What we have been doing in all these examples is *denotational semantics*: the art of using mathematical objects to study logic and programming languages. In the first case, our denotational semantics was expressed through the mathematics of sets and functions, where we captured the behaviour of a (programming language) function via a (mathematical) function.

Then, following Scott [Sco76], we refined this model to one based on partially ordered sets – more specifically, *Scott domains* – in which we modelled the behaviour or a program via an associated Scott-continuous functions between domains.

In our other examples, we need to come up with further refinements to our model in order to incorporate the new computational effects. For example, to handle nondeterminism, we might want to switch to using nondeterministic functions, or *relations*, instead of ordinary functions.

The advantages in all of these cases is that the mathematical objects we use are often fairly simple, whereas computer programs, even in simple 'toy' languages, are very complicated to study. A program is, at its heart, a string of symbols governed by a collection of operational rules that govern how such strings should behave. Such an object is very fiddly to reason with directly; indeed, the only way to think about it is as some kind of 'function' from input to outputs. Denotational takes this basic intuition further, and aims to capture features of programming languages through a diverse collection of different mathematical models.

A word of warning: the principal mode of denotational semantics which we shall be studying in this thesis is game semantics, which is much more complicated than the semantics of sets and functions.

1.2 Computational Adequacy & Full Abstraction

In order for a denotational semantics to tell us anything, we first need to prove some results that relate it to the language we are studying. For example, if we are hoping to model a programming language using sets and functions, then we need to define a mapping $[\![-]\!]$ (the *denotation*) that takes program types to sets and program functions to functions between those sets, and we also need to prove that this denotation respects the operational rules of the language. For example, we might want to prove that if $f\colon \mathrm{int} \to \mathrm{int}$ is a function and $M\colon \mathrm{int}$ is a term that evaluates to the integer n, then the term fM will evaluate to the integer $[\![f]\!](n)$.

This type of result is called *Computational Adequacy*, and relates to a program's *observable behaviour*. Some programs have an automatic notion of observable behaviour: if we run them, then they will return a concrete value or fail to terminate. In a typed setting, these are the programs of ground type. Others – for example, programs of function type – require input before they can return a value. These programs have no observable behaviour of their own, but if we insert them into a hole in a larger context, then they may influence the observable behaviour of the complete program.

Briefly speaking, a Computational Adequacy result tells us that the observ-

able behaviour of a program of ground type may be deduced exactly from its denotation. For example, in a domain-theoretic semantics, we might want to say that a program M evaluates to a value v if and only if $[\![M]\!] = v$ and that M fails to terminate if and only if $[\![M]\!] = \bot$.

Such a computational adequacy result extends readily to terms not of ground type. Given two programs M and N of the same type, we say that M and N are *observationally equivalent* if $C[M]$ and $C[N]$ have the same behaviour for any one-holed context $C[-]$ of ground type. If our semantics is *compositional* – so that the denotation of $C[M]$ is obtained by 'applying' the denotation of $C[-]$ to the denotation of M – and computationally adequate, then it follows that the semantics is *equationally sound*: if two terms M and N have the same denotation, then they are observationally equivalent.

If we have an effective way of computing denotations, then this can give us an easy way to prove observational equivalence of terms. However, it gives no guarantee that this technique is always going to work: the terms M and N might be observationally equivalent despite having distinct denotations. The gold standard of denotational semantics – *Full Abstraction* – asserts in addition that the converse of equational soundness holds, so that the denotational semantics completely captures the observational equivalence relation. This means that if two terms are observationally equivalent, then we can always prove that they are observationally equivalent by showing that they have the same denotation.

An important early success in this direction came with Plotkin's introduction of the stateless sequential programming language PCF [Plo77]. Plotkin was unable to provide a fully abstract denotational semantics for PCF itself, but he showed that if we add a simple parallel construct[1] to PCF, then a denotational semantics based on Scott domains is fully abstract. This result presents us with a world in which we can practically and systematically check observational equivalence (for terms of this extended version of PCF) by computing denotations. If two term displays infinitary behaviour, then it is not necessarily possible to check whether their denotations are equal, but if they are finitely presentable in some sense, so that their denotations as functions between Scott domains can be computed and compared, then the Full Abstraction result gives us, at least in principle, a systematic recipe for checking observational equivalence.

Unfortunately, this stops working once we remove the extra parallel construct. PCF is a sub-language of the parallelized version, and therefore can also be given a denotational semantics via Scott domains, but the absence of

[1] Specifically, 'parallel or', which evaluates its two boolean arguments in parallel, and is thus able to return true if either the left or the right argument returns true, even if the other fails to terminate.

parallelism also means that the observational equivalence relation is coarser: there may be terms that can be distinguished by a context including the parallel construct that cannot be distinguished inside any purely sequential context. This means that the Full Abstraction result does not automatically pass over to sequential PCF.

Any hope of solving the problem with a tweaked model was brought down to earth by Ralph Loader's 2001 theorem that observational equivalence for PCF is undecidable, even if we restrict ourselves to a finitary version of the language with no infinite datatypes or recursion beyond a simple non-termination primitive $\bot$. This in particular tells us that no concretely presentable denotational semantics for PCF can possibly be fully abstract, or it would in principle give us an algorithm for deciding observational equivalence in this finite version.

Despite this fact, there were, roughly contemporaneous with Loader's result, several fully abstract models of PCF published, in a watershed moment for the subject. The model published by O'Hearn and Riecke [OR95] was more or less along domain-theoretic lines, while those of Abramsky, Jagadeesan and Malacaria [AJM00], Hyland and Ong [HO00], and Nickau [Nic94] used the relatively new Game Semantics.

These models took a slightly oblique approach to Full Abstraction, which is how they reconcile themselves with Loader's Theorem. First, they defined the notion of *intrinsic equivalence* of terms of the same type T in a denotational model, where two elements σ and τ of the denotation of T are intrinsically equivalent if $\alpha(\sigma) = \alpha(\tau)$ for all functions $\alpha\colon [\![T]\!] \to [\![o]\!]$ from the denotation of T to the denotation of some fixed ground type o. This definition is very closely linked to that of observational equivalence; indeed, if two terms M and N are observationally equivalent, and we are working in a computationally adequate and compositional denotational semantics, then the denotations of M and N will be intrinsically equivalent, since for any ground-type context $C[-]$, we can take α to be the denotation of $C[-]$ in the above definition.

Proving the converse – that intrinsic equivalence of denotations implies observational equivalence – entails going in the opposite direction; i.e., starting with some element α in the model and coming up with a context $C[-]$ in the language whose denotation is α. Thus, proving this direction normally reduces to some kind of *definability* result. Typically, we do not actually need to prove that every α is definable within the language – and, indeed, this is often not the case. Instead, we try to show that each element α in the language can be written as the least upper bound of elements α' living in some more restricted class whose members can all be defined in the language. This class usually consists of those elements which are *compact* or finitary in some sense, though McCusker's Fully Abstract relational se-

mantics for SCI [McC02] is an important exception. Under mild continuity assumptions, this suffices to prove Full Abstraction, for we can then deduce that if $\alpha(\sigma) \neq \alpha(\tau)$ for some α, then there is some definable α' such that $\alpha'(\sigma) \neq \alpha'(\tau)$. This is the approach taken by the fully abstract semantics that have been given for PCF; there is no contradiction of Loader's theorem, because the intrinsic equivalence relation is itself undecidable, even for finitary terms.

If we can prove, for some denotational model of a language, that observational equivalence of terms is equivalent to intrinsic equivalence of their denotations, then we can form a fully abstract model by passing to equivalence classes under the intrinsic equivalence relation. In this thesis, we shall skip the final step of passing to equivalence classes and declare a denotational semantics to be fully abstract for a language if we can prove that observational equivalence of terms is equivalent to intrinsic equivalence of their denotations. Thus, the Full Abstraction results that we prove will have three main ingredients: compositionality, computational adequacy and definability.

Lastly, we note that the conclusion of Loader's theorem does not necessarily hold for other languages. For example, in Section 3.8, we shall demonstrate a denotational characterization of observational equivalence in Idealized Algol, due to Abramsky and McCusker [AM96], which can be adapted to give us an algorithm for deciding observational equivalence for a finitary version of Idealized Algol.

1.3 Categorical Semantics

There is a close link between (typed) programming languages and categories. Most programming languages have things called *types*, and many have *functions* that go from one type to another. Typically, it will be possible to compose two functions together in the language in an associative way, giving us a category. It should come as no surprise, then, that a very important branch of denotational semantics is *categorical semantics*, in which we take some existing category from mathematics, and use its objects and morphisms to represent the types and terms of a programming language.

Typically, each type T of the language will correspond to some object $[\![T]\!]$ of the category, while a term of type T will correspond to a morphism $1 \to [\![T]\!]$, where 1 is some fixed object in the category (usually a terminal object).

Particularly important [Lam68] are the Cartesian closed categories, which have a number of properties making them suitable for denotational semantics:

Product and function spaces Given types S and T, we can define the

denotations of the product type $S \times T$ and the function type $S \to T$ to be given by $[\![S]\!] \times [\![T]\!]$ and $[\![T]\!]^{[\![S]\!]}$.

Compositionality Given types S and T, and corresponding objects $[\![S]\!]$ and $[\![T]\!]$ of the category, we can define the denotation of the function type $S \to T$ to be given by the exponentiation $[\![T]\!]^{[\![S]\!]}$ as above. Then we automatically have a recipe for substituting a term of type S into a function of type $S \to T$ via the canonical morphism

$$[\![T]\!]^{[\![S]\!]} \times [\![S]\!] \to [\![T]\!] \,.$$

Abstraction Given a morphism $\sigma \colon A \times B \to C$, we may form a morphism $\Lambda(\sigma) \colon A \to C^B$. This gives us the semantics for λ-abstraction, whereby we pass from a term-in-context

$$\Gamma, x \colon S \vdash M \colon T$$

to the term-in-context

$$\Gamma \vdash \lambda x.M \colon S \to T \,.$$

These rules allow us to build up a model of the simply-typed λ-calculus within any Cartesian closed category, which means we get a large part of the denotation (and the subsequent proof of Computational Adequacy) for free.

This alone would be a good justification for using category theory in denotational semantics, but the benefits go further. The development of programming languages such as Haskell has been strongly influenced by category-theoretic concepts. For example, Moggi's 1991 observation [Mog91] that monads on categories provide us with a way of modelling computational effects influenced work that led directly to the introduction of support for monads in Haskell [Jon95], where they have become the primary tool for abstracting out effectful computation.

Monads will be particularly important in this thesis, so it is worth dwelling on them a little further. A *monad* on a category $\mathcal{C}$ is given by a functor $M \colon \mathcal{C} \to \mathcal{C}$, together with natural transformations

$$e \colon \mathrm{id}_{\mathcal{C}} \Rightarrow M \qquad\qquad m \colon M \circ M \Rightarrow M$$

that endow M with an algebraic structure. One example is the non-empty powerset functor on the category of sets, together with the natural transformations given by

$$\begin{array}{rl} e \colon A \to \mathcal{P}(A) & \qquad m \colon \mathcal{P}(\mathcal{P}(A)) \to \mathcal{P}(A) \\ a \mapsto \{a\} & \qquad \mathcal{A} \mapsto \bigcup_{A \in \mathcal{A}} A \,. \end{array}$$

This powerset monad indicates some kind of nondeterministic choice between elements of A, particularly if we modify the construction to the *non-empty powerset* functor $\mathcal{P}_+$.

Another example in the category of sets is the functor $A \mapsto A + \{\bot\}$, that appends an additional element on to a set. We have natural functions $A \to A + \{\bot\}$ and $A + \{\bot\} + \{\bot\} \to A + \{\bot\}$ that make this into a monad as well. In the study of programming languages, this is often called the *maybe monad*, because $A + \{\bot\}$ indicates an element of A that may or may not be present (with the distinguished value $\bot$ indicating no value).

Given a monad M on a category $\mathcal{C}$, we can form a new category $\mathrm{Kl}_M\,\mathcal{C}$ – the *Kleisli category* of M – whose objects are the objects of $\mathcal{C}$ and where a morphism from A to B is given by a morphism $A \to MB$ in $\mathcal{C}$. The monadic coherence gives us the correct notion of composition: given Kleisli morphisms $\sigma\colon A \to MB$ and $\tau\colon B \to MC$, we may compose them to give a morphism $A \to MC$ via the following formula.

$$A \xrightarrow{\sigma} MB \xrightarrow{M\tau} MMC \xrightarrow{m} MC$$

The Kleisli category of the powerset monad is the category of sets and relations, while the Kleisli category of the maybe monad is the category of sets and partial functions.

There are numerous other monads that can be used to model computational effects, such as the state monad and the exception monad. Work by Plotkin and Power [PP02] makes this more precise, by studying monads that can be built up via algebraic operations and equations. For example, we might want to model nondeterministic choice on a set A via an operation $\sum$ that takes in infinitely many elements of A – so $\sum a_i$ gives us a choice between the a_i. We then impose some axioms on this operation.

Idempotence If $a_i = a$ for all i, then $\sum a_i = a$;

Commutativity $\sum a_i = \sum_{a_{\pi(i)}}$ for any permutation π; and

Associativity $\sum_i (\sum_j a_{ij}) = \sum_{i,j} a_{ij}$.

This is an algebraic theory akin to the theory of groups, and its category of free algebras is isomorphic to the Kleisli category of the powerset monad.

1.4 Game Semantics

Game Semantics gives us a particularly fruitful categorical semantics for programming languages. The underlying idea is that a computer program behaves like a strategy for a two-player game, in that it needs to respond to arbitrary inputs (opponent moves) with its own behaviours (proponent

moves). Thus, we represent a programming language type by an idealized game between two players O and P[2], and represent a term of that type by a strategy for that game.

The power of game semantics comes from the fact, first noted by Blass in [Bla92], that certain natural operations on games correspond to some of the connectives of linear logic. For example, if A and B are two-player games, then we may form their *tensor product* $A \otimes B$, which is played by running the games A and B together in parallel, with the opposing player O allowed to switch between games when it is his turn.

A closely related construction takes games A and B and forms their *linear implication* $A \multimap B$, in which B is played in parallel with the *dual* of A, in which the roles of players P and O are swapped round. This time, player P can choose to switch game when it is her turn.

The remarkable thing about the $\multimap$ construction, first pointed out by Joyal in [Joy77], is that if A, B and C are games, then we may compose a P-strategy for $A \multimap B$ with a P-strategy for $B \multimap C$ to get a P-strategy for $A \multimap C$.

In order to form her strategy, player P sets up a 'scratchpad' consisting of the games $A \multimap B$ and $B \multimap C$ side by side. Now suppose that player O makes a move in $A \multimap C$ that originally came from the game C. Then player P treats this move as a move in $B \multimap C$, and uses her strategy for that game to come up with a reply. If that reply is a move in C, then she plays it as her response. Otherwise, if it is a move in B, she treats that move as an O-move in $A \multimap B$, and therefore has a reply in $A \multimap B$ according to her strategy for that game. Eventually, if she plays a move in A or C, then that will be her reply in the composite strategy. See Figure 1.1 for an illustration.

It is possible instead that player P flips between her two strategies for ever, always playing moves in B and never coming out into A or C. Computationally, this represents 'livelock': a computation that does not terminate because two subroutines are continually deferring to each other without returning values of their own.

This composition of strategies gives us a category $\mathcal{G}$ in which the objects are games and a morphism from a game A to a game B is a strategy for the game $A \multimap B$. In this category, the identity morphism on a game A is the *copycat strategy* for $A \multimap A$, which responds to an O-move in either copy of A with the identical move in the other copy.

What is more, the connectives $\otimes$ and $\multimap$ make $\mathcal{G}$ into a symmetric monoidal closed category. We typically apply some category-theoretic construction

[2]Another convention is to refer to player O as $\forall$belard and player P as $\exists$loïse, so I will refer to player O as 'he' and player P as 'she' throughout.

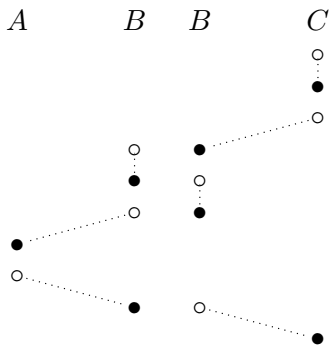


Figure 1.1: Illustration of the composition of strategies in Game Semantics. Working top-to-bottom, the symbol $\circ$ denotes a move by O and the symbol $\bullet$ a move by P. A dotted line indicates that a move is determined by one of player P's strategies for the games $A \multimap B$ and $B \multimap C$. Note that the moves in B are duplicated, so that they may always be considered as an O-move in either $A \multimap B$ or $B \multimap C$. Moves from B are hidden in the composite strategy: in this case, player P's first 'real' move is in C, her second in A and her third in C.

to $\mathcal{G}$ to get a Cartesian closed category, by passing either to the Kleisli category for a linear exponential comonad on $\mathcal{G}$ (as in [AJM00]; see [Sch04]), or to a subcategory of the category of cocommutative comonoids in $\mathcal{G}$ (as in [HO00, AM96]; see [Har99, §3.5.2]). Once we have a Cartesian closed category, we automatically have a way to interpret the simply-typed lambda calculus.

1.5 Game Semantics for Programming Languages

The first triumph of Game Semantics was to solve the Full Abstraction problem for PCF, but a more lasting application of the discipline has been to model more general programming languages with effects such as state. By making changes to the definitions of game and strategy, Game Semantics has proved to be applicable to a wide variety of programming language effects, including exceptions [Lai01], coroutines and continuations [Lai16], nondeterminism [HM99], probability [DH00], and general references [AHM98].

The precise definition of a strategy that we use depends on the language that we are trying to model – a language with less expressive power can realize fewer strategies. For example, the denotations of terms in a stateless language such as PCF are *history-free* or *innocent*, in which the proponent's moves can only depend on a particular subsequence of the current sequence of moves – the P-view – rather than on the whole sequence. So for a lot

of computational effects, particularly ones that have something to do with state, adding that effect corresponds to a *relaxing* of conditions on strategies.

We model other types of effects by extending the definition of a strategy. For example, if we want to provide a semantics for a language with nondeterminism, then we modify the definition of a strategy so that the proponent can have multiple replies to each opponent move, as in the work of Harmer and McCusker [HM99]. If we want to model a probabilistic language, then we decorate these different moves with probabilities, as in the work of Danos and Harmer [DH00].

When choosing a definition of a strategy, the aim is to prove a definability result, so that we can prove Full Abstraction. The original proofs of definability of compact innocent strategies in PCF from [AJM00] and [HO00] were intricate and technical. Subsequent work on languages that extend PCF tends to try to prove definability via a *factorization result*, in which we show that every strategy in an extended category of games may be written as the composition of a strategy in an original category of games with some fixed strategy in the new category. Then, if we have a definability result for the original semantics, we can extend it to a definability result in the new category.

For example, the language Idealized Algol is an extended version of PCF that adds some stateful primitives. Abramsky and McCusker's proof of compact definability for Idealized Algol in [AM96] first proves that each compact strategy in their model may be written as the composite of a compact innocent strategy with the (non-innocent) denotation of one of the new stateful constants. Thus, they can deduce compact definability for Idealized Algol from Hyland and Ong's result that every compact innocent strategy is the denotation of a term of PCF.

Similarly, Harmer and McCusker develop in [HM99] a model of game semantics in which strategies can be nondeterministic. They show that every such strategy can be written as the composite of a deterministic strategy with some particular fixed nondeterministic strategy. Then, to prove compact definability for nondeterministic Idealized Algol, it suffices for them to exhibit a nondeterministic term whose denotation is that fixed strategy.

We can now summarize the typical process of proving a Full Abstraction language for a language $\mathcal{L}'$ that extends a language $\mathcal{L}$ as follows.

- Starting with an existing computationally adequate model $\mathcal{C}$ of $\mathcal{L}$ that satisfies compact definability, define a categorical model $\mathcal{C}'$ that admits an identity-on-objects functor $J\colon \mathcal{C} \to \mathcal{C}'$. For example, if $\mathcal{C}$ is a category of games and strategies, $\mathcal{C}'$ might be a category whose objects are the same games as $\mathcal{C}$, but where the strategies are less rigidly constrained.

- Prove that the model $\mathcal{C}'$ is a Cartesian closed category and is computationally adequate for $\mathcal{L}'$.
- Prove a factorization result that exhibits every morphism g in $\mathcal{C}'$ as the composition of some morphism Jf in the image of J with one of some fixed collection of morphisms that are known to be definable in $\mathcal{L}'$ (for example, single terms from $\mathcal{L}' \setminus \mathcal{L}$). In addition, if g is compact, then f should be compact in $\mathcal{C}$.

The third bullet point allows us to deduce a compact definability result for $\mathcal{L}'$ in $\mathcal{C}'$ from the compact definability result of $\mathcal{L}$ in $\mathcal{C}$, and then Full Abstraction follows as we have outlined in Section 1.2.

The proofs of Full Abstraction for stateful and nondeterministic languages that we have mentioned [AM96, HM99] prove Full Abstraction in this way. Other important Full Abstraction results in Game Semantics that follow this pattern include Danos and Harmer's result for a probabilistic variant of Idealized Algol [DH00], the Full Abstraction result for general references of Abramsky, Honda and McCusker [AHM98], Laird's result for local exceptions [Lai01], Murawski and Tzevelekos's Nominal Game Semantics [MT16] and the result for countable nondeterminism by Laird and the present author [GL18].

Lastly, it is worth mentioning several papers that prove Full Abstraction without going through a factorization result. This is usually because they depart more radically from traditional game semantics. Such papers include Tsukada and Ong's sheaf-based models for nondeterministic PCF [TO15, TO14], Laird's categorical semantics for coroutines [Lai16] and the results for probabilistic PCF by Castellan, Clairambault, Paquet and Winskel using concurrent games [CCPW18].

1.6 Full Abstraction for Kleisli Categories

One thing which these Full Abstraction results have in common is that they rely on some degree of human intuition to build the original model. This process can often be very difficult. For instance, it seems obvious that if we want to model a nondeterministic programming language, then we need to relax the determinism constraint on strategies. However, relaxing the determinism constraint turns out not to be enough on its own: recall that we model a nonterminating computation in game semantics by a strategy that has no P-reply to a particular O-move. How then do we model a term which chooses between terminating at a value v and not terminating, and how do we distinguish it from a term which always terminates at v? In both cases, player P has the reply v to player O's initial move, but now there is nothing to indicate the possibility of non-termination.

Harmer and McCusker are able to solve this problem, in the finite nondeterminism case, by separately keeping track of divergences in the strategy, but further problems arise when we start to consider countable nondeterminism. Since Game Semantics usually keeps track of finite sequences of moves, nondeterministic strategies are unable to distinguish between a program that nondeterministically chooses a number n and prints "Hello, world" n times, and a program that either does the same thing, or prints out the message infinitely many times. So now we have to add extra information in about infinite sequences of moves (see Levy's work on infinite trace equivalence [Lev08] and the work of Laird and the present author on nondeterministic Idealized Algol [GL18]).

Things get even more difficult when we consider nondeterministic versions of stateless languages such as PCF. Naively relaxing the determinism constraint on the usual definition of an innocent strategy does not give the right notion of a nondeterministic innocent strategy [TO15]. There are definitions of nondeterministic innocence due to Levy [Lev14], to Tsukada and Ong [TO15], and, via concurrent games, to Castellan, Clairambault, Hayman and Winskel [CCHW18], that use the concept of *morphisms between plays*, but these are already some distance removed conceptually from Hyland and Ong's original paper that introduced deterministic innocent strategies. Tsukada and Ong's paper involves a complete recasting of strategies as sheaves in order to understand what happens when we try to mix nondeterminism and innocence.

This is just one example that shows that the process of adding different computational effects into a game semantics can be very hard and can require some ingenuity. Meanwhile, there is plenty of well-known theory that deals with computational effects in a purely systematic way. We have already met two such techniques in section 1.3: monads (and Kleisli categories) and Lawvere theories. Put simply, the purpose of this thesis is to investigate what happens when we try to use these systematic techniques to prove Full Abstraction results for effectful languages.

Let us start by looking at Kleisli categories for monads. Recall that a monad M on a category $\mathcal{C}$ is a functor $M\colon \mathcal{C} \to \mathcal{C}$ that satisfies certain conditions, and that the Kleisli category $\mathrm{Kl}_M\,\mathcal{C}$ of M has the objects of $\mathcal{C}$ as its objects, and that a morphism $a \to b$ in $\mathrm{Kl}_M\,\mathcal{C}$ is given by a morphism $a \to Mb$ in $\mathcal{C}$. There is a natural identity-on-objects functor $J\colon \mathcal{C} \to \mathrm{Kl}_M\,\mathcal{C}$.

In particular, if a is any object of $\mathcal{C}$, then we have a distinguished Kleisli morphism $\phi_A\colon Ma \to a$ in $\mathrm{Kl}_M\,\mathcal{C}$ given by the identity morphism $Ma \to Ma$ in $\mathcal{C}$. Now let $f\colon a \to b$ be an arbitrary morphism in $\mathrm{Kl}_M\,\mathcal{C}$, given by a morphism $\hat{f}\colon a \to Mb$ in $\mathcal{C}$. Then we can write f as the following composite

in $\mathrm{Kl}_M\,\mathcal{C}$.

$$a \xrightarrow{J\hat{f}} Mb \xrightarrow{\phi_b} b$$

In other words, the Kleisli category $\mathrm{Kl}_M\,\mathcal{C}$ automatically satisfies a factorization result of the type we have been talking about. So if $\mathcal{C}$ is a model of a language $\mathcal{L}$ that satisfies compact definability, then $\mathrm{Kl}_M\,\mathcal{C}$ will automatically satisfy compact definability for any denotational semantics for a language language $\mathcal{L}'$ that includes (via the functor J) the existing denotational semantics for $\mathcal{L}$ and in which the morphisms ϕ_a are all definable.

Proving computational adequacy is not so automatic. One approach is to treat $\mathrm{Kl}_M\,\mathcal{C}$ like any other model and to apply the usual arguments for proving adequacy. Most computational adequacy arguments rely on order-enriched properties of the underlying categories, particularly for dealing with recursion, and the Kleisli category $\mathrm{Kl}_M\,\mathcal{C}$ automatically inherits order-enriched structure from $\mathcal{C}$, by saying that $f \leq g \colon a \to b$ in $\mathrm{Kl}_M\,\mathcal{C}$ if $f \leq g$ when considered as morphisms $a \to Mb$ in $\mathcal{C}$.

An alternative approach, which we shall use in this thesis, is to try and deduce computational adequacy for $\mathrm{Kl}_M\,\mathcal{C}$ from a computational adequacy result for the original language $\mathcal{L}$ in $\mathcal{C}$. In this approach, we take a term N of the extended language $\mathcal{L}'$ and note that its denotation $[\![N]\!] : a \to b$ in $\mathrm{Kl}_M\,\mathcal{C}$ is given by some morphism $f \colon a \to Mb$ in $\mathcal{C}$. Typically, it is easy to construct a term P of $\mathcal{L}$ whose denotation in $\mathcal{C}$ is f. We then prove results to peg the operational behaviour of N in $\mathcal{L}'$ to that of P in $\mathcal{L}$, allowing us to deduce a computational adequacy result for our model of $\mathcal{L}'$ from the corresponding result for $\mathcal{L}$.

The last ingredient that we need is to prove that $\mathrm{Kl}_M\,\mathcal{C}$ is a Cartesian closed category. This is not immediate either: $\mathrm{Kl}_M\,\mathcal{C}$ need not be Cartesian, even if $\mathcal{C}$ is. For the purposes of this thesis, we will be considering only monads of a very special form – the *reader monads* R_z on Cartesian closed categories, given by $R_z a = z \to a$. The Kleisli category for a reader monad on a Cartesian closed category is always Cartesian closed [Lam74].

1.7 Difficulties with Countable Nondeterminism

This thesis will focus on nondeterministic effects. Of these, perhaps the trickiest to work with is countable nondeterminism, in which a program can nondeterministically choose between a possibly infinite number of different options, without the possibility of non-termination[3].

[3] If we have a source of finite nondeterminism, then we can get infinite branching if we accept the possibility that our program might never terminate – for example, count the number of coin tosses it takes before we get the first head.

There are three main difficulties when dealing with countable nondeterminism in game semantics. We have covered the first in the previous section: we cannot tell everything about a program's behaviour by looking at its possible finite traces, even when the traces are allowed to be arbitrarily long, as in the example where a program nondeterministically chooses a number n and prints out a message n times vs the program which may print out the message infinitely many times.

The fact that finite nondeterminism avoids this behaviour comes down to König's Lemma, which asserts that any finitely branching tree without an infinite branch has bounded height.

Related is second problem, which is the failure of continuity of composition. This was noticed by Dijkstra in [Dij97, Ch. 9] and later studied by Plotkin and Apt in [AP81].

In order to explain what we mean by continuity, we define the *observational preorder* on terms of a language. If $M, N : T$, we write $M \lesssim N$ if for all ground-type contexts $C[-]$ with a hole of type T we have

$$C[M] \text{ always terminates} \Rightarrow C[N] \text{ always terminates.}$$

In particular, M and N are observationally equivalent if and only if $M \lesssim N$ and $N \lesssim M$.

Continuity of composition means that least upper bounds with respect to this order are preserved by function application. Consider, for example, the infinite sequence of terms $\leq_m : \mathbb{N} \to \mathbb{N}$ for $m = 0, 1, \ldots$ that return 0 if their argument is less than or equal to m and go into an infinite loop otherwise. It is fairly clear (with a straightforward rigorous proof once we have introduced the denotational semantics) that a least upper bound for the $\leq_m$ is the term $\leq_\infty$ that evaluates its argument and then returns 0 regardless of what value it finds[4].

But now notice what happens if we have a countable nondeterminism primitive ? in our language which when evaluated returns an arbitrarily large natural number. Then the terms $\leq_m ?$ are all observationally equivalent: they either return 0 (when ? returns a number less than or equal to m), or fail to terminate (when ? returns a number greater than m). We might write

$$\leq_m ? = 0 + \Omega\,.$$

Therefore, the sequence $\leq_m ?$ is a constant sequence and its least upper bound is that constant value $0 + \Omega$.

[4] $\leq_\infty$ differs slightly from $\lambda n.0$ – also an upper bound for the $\leq_m$, but not the least upper bound – in that it evaluates its argument before returning 0. This means that it will fail to terminate if its argument fails to terminate.

Meanwhile, $\leq_\infty ?$ always returns 0. Therefore, application to ? does not preserve least upper bounds.

The reason that this is a problem is that a typical strategy for proving computational adequacy is via an order-enriched category in which the observational preorder at a type matches up with the ordering of morphisms into the denotation of that type. The standard adequacy proof uses continuity of composition in the model in an essential way – see Lemma 3.4.6 for an example of such an argument being used to prove computational adequacy for a deterministic language. But if the ordering of morphisms in a denotational model of a language with countable nondeterminism faithfully respects the observational ordering of terms, then composition of morphisms is necessarily not continuous – since it is not continuous in the language itself.

Our strategy for getting around this problem goes back to Levy's paper *Infinite Trace Equivalence* [Lev08], and essentially involves going via an auxiliary language, in which the nondeterministic oracle ?, when it chooses a value, must print that value to a log. Given a term P of ground type, we write $P \Downarrow_u$ if P terminates whenever it prints the sequence u to the log. We then replace our earlier definition of the observational preorder with a stronger one: we write that $M \lesssim N$ if for all suitable contexts $C[-]$ we have

$$C[M] \Downarrow_u \Rightarrow C[N] \Downarrow_u \ .$$

This ordering on terms is continuous. For example, the extra requirement wrecks our previous example of failure of continuity of composition. Indeed, the sequence $\leq_m ?$ is no longer constant, since, for example, $\leq_5 ?$ does not terminate if it prints 6 to the log, whereas $\leq_{1000} ?$ does. The least upper bound of the $\leq_m ?$ is then the term that terminates whenever it prints a single number to the log; i.e., $\leq_\infty ?$.

We can prove continuity of composition in general for the new language, which allows us to prove computational adequacy in the usual way. Having done this, we then use operational methods to find a way to identify terms that should be observationally equivalent in the original sense. This allows us to quotient out our model for the new language to get one that is computationally adequate for the ordinary nondeterministic version.

The last problem is to do with definability. First note that there are two different reasons why an element of a denotational model (e.g., a strategy) might not be definable in a particular language. One reason is structural: for example, a stateless language such as PCF cannot define a strategy whose behaviour depends on its entire history, since that would indicate stateful behaviour. The other reason is purely computational. Given a non-computable function $f : \mathbb{N} \to \mathbb{N}$ (for example, the function that returns 0 if

its argument is source code for a terminating program and 1 otherwise), we have a perfectly well-behaved history free strategy that represents f, but f is nevertheless still not definable in most programming languages. Historically, the study of Full Abstraction has not been too concerned with non-definable elements of the second kind: the reason is that they do not usually play a role in determining the intrinsic equivalence relation. However, in the presence of countable nondeterminism, there exist definable terms that can be distinguished in the model that cannot be distinguished by definable elements, but can be distinguished by non-computable functions. An example of such a function is found in Section 4.14; the general idea is due to Kleene and can be found in [Bau06].

What this means is that if we want to model countable nondeterminism then structural constraints are not enough: we need to impose computability constraints on strategies as well. Happily, such notions are well studied already, even appearing in the original Hyland-Ong and AJM papers [HO00, AJM00], in which the authors give a complete characterization of those strategies definable in PCF – a *universality* result for the model. Similar results can also be found in earlier work on Full Abstraction, such as Plotkin's original PCF paper [Plo77].

The difficulties with countable nondeterminism that we have outlined are essentially domain-theoretic, and can be summed up by saying that composition of countably nondeterministic functions is not continuous – i.e., does not preserve least upper bounds – with respect to the natural orderings on terms.

1.8 Plan for this Thesis

This thesis will develop the theory of Full Abstraction from the point of view of techniques of categorical algebra such as Kleisli categories.

Chapters 2 and 3 give a fairly traditional presentation of a Fully Abstract game semantics model for Idealized Algol. All subsequent results will be for extended versions of Idealized Algol, so this model can serve as the foundation for the rest of our work. Although these chapters form by some distance the longest part of the thesis, they also contain the smallest amount of new material, and essentially cover the same material as Abramsky and McCusker's Fully Abstract game semantics for Idealized Algol [AM96]. The main difference, which ties in with the general theme of the thesis, is that we prove Computational Adequacy using techniques of categorical algebra rather than via the ad hoc combinatorial arguments found in [AM96]. Specifically, we use the concept of a *sequoidal category* and of the exponential as a final coalgebra, introduced by Laird in [Lai02] to prove Full Abstraction for

a language with general references. We present the first application of this technique to a language with purely local state of ground type. The remainder of the Full Abstraction result is largely as in [AM96].

Chapter 4 presents the general theory of monads and Kleisli categories. It then presents a technique which can be used to prove Full Abstraction for several nondeterministic effects, along the lines we outlined in the previous section.

For the remainder of the thesis, we deal with a generalization of monads – *parametric monads*, in which the action of the monad is parameterized by an object of some monoidal category $\mathcal{X}$ (so that we deal with a functor $\mathcal{X} \times \mathcal{C} \to \mathcal{C}$ rather than a functor $\mathcal{C} \to \mathcal{C}$).

Chapter 5 introduces and defines parametric monads (also known as *lax actions*), and proves a number of technical results which we need. It also introduces the *Melliès category* and the oplax 2-limit $\mathcal{C}/\mathcal{X}$, which give us two related analogues of the Kleisli category in the parametric case. The Melliès category is an enriched category, while $\mathcal{C}/\mathcal{X}$ is an ordinary category obtained from it by change of base. As with monads, we need to specialize to a small class of parametric monads – the *reader actions* – in order to ensure that $\mathcal{C}/\mathcal{X}$ is a Cartesian closed category.

Chapter 6 uncovers a large source of these reader actions; in this chapter, we prove that reader actions on the category **Set** of sets are equivalent to lax monoidal functors $\mathbf{Set} \to \mathbf{Set}$.

Chapter 7 takes a small detour away from the main narrative to study the Melliès category from the point of view of profunctors.

Chapter 8 completes our study of parametric monads by proving a Full Abstraction result for categories derived from Melliès categories, using similar techniques to Chapter 4. As an example, we give a semantics for a probabilistic language for a category derived from a particular parametric monad, and show that it is Fully Abstract.

Lastly, our conclusion in **Chapter 9** provides a glimpse into several further directions that are left unexplored by the rest of the thesis.

Chapter 2

A Sequoidal Game Semantics for Idealized Algol

To introduce our material, we will go back over some old ground, namely the fully abstract game semantics for Idealized Algol developed by Abramsky and McCusker in [AM96]. In keeping with the spirit of this thesis, we will aim to use category theoretic methods, and so our proofs of soundness and adequacy will depart from those given by Abramsky and McCusker, and will instead involve coalgebraic ideas developed by Laird in [Lai02] and [Lai16].

2.1 Idealized Algol

The ground types of Idealized Algol are called `com`, `bool`, `nat` and `Var`. The first three are data types corresponding to the sets $\mathbb{C} = \{a\}$, $\mathbb{B} = \{\mathtt{t}, \mathtt{f}\}$ and $\mathbb{N} = \{0, 1, 2, \cdots\}$. `com` takes the role of a command or void type; typically, although the return value of a function $T \to \mathtt{com}$ does not convey any information, the function will have side effects that *do* make a difference.

The type `Var` is the type of a variable holding elements of $\mathbb{N}$. It is best understood as corresponding to the following pseudo-Java 'interface'.

```
public interface Var
{
  nat read();
  com write(nat value);
}
```

In other words, any term M of type `Var` supports two operations: one 'read' operation that returns a natural number and one 'write' operation that accepts a natural number. The most natural case to consider is when M

actually corresponds to a storage cell, with 'write' updating the value and 'read' returning it, but our language will permit any other implementation of the `Var` interface, including, for example, a term of type `Var` that ignores any values 'written' to it and always returns 0 when read: such terms, which fulfil the type-theoretic contract of a storage cell without replicating its behaviour, are sometimes called 'bad variables' (e.g., in [AM99, §2]).

The types of Idealized Algol are then defined by the following inductive grammar.

$$T ::= \mathtt{com} \mid \mathtt{bool} \mid \mathtt{nat} \mid \mathtt{Var} \mid T \to T\,.$$

Next, we present the typing rules for the language. Here, Γ will stand for a *context*; i.e., a list $x_1 \colon T_1, \cdots, x_n \colon T_n$ of variable names together with their types.

First, we have the usual rules for the simply typed lambda calculus.

$$\frac{}{\Gamma, x \colon T \vdash x \colon T} \qquad \frac{\Gamma \vdash M \colon S \to T \qquad \Gamma \vdash N \colon S}{\Gamma \vdash MN \colon T} \qquad \frac{\Gamma, x \colon S \vdash M \colon T}{\Gamma \vdash \lambda x^S.M \colon S \to T}$$

We then have rules for each of the base types. At type `com` we have:

$$\frac{}{\Gamma \vdash \mathsf{skip} \colon \mathtt{com}} \qquad \frac{\Gamma \vdash M \colon \mathtt{com} \qquad \Gamma \vdash N \colon T}{\Gamma \vdash M; N \colon T}\ T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}\,.$$

Here, skip is a generic command with no side-effects that returns the unique element a of the singleton set $\mathbb{C}$. $M; N$ represents the sequential composition of M with N; i.e., the term that first evaluates M, performing any of its side-effects, and then evaluates N and returns the result.

At type `bool` we have true/false values and conditionals.

$$\frac{}{\Gamma \vdash \mathbb{t} \colon \mathtt{bool}} \qquad \frac{}{\Gamma \vdash \mathbb{f} \colon \mathtt{bool}}$$

$$\frac{\Gamma \vdash M \colon \mathtt{bool} \qquad \Gamma \vdash N \colon T \qquad \Gamma \vdash P \colon T}{\Gamma \vdash \mathsf{If}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \colon T}\ T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}$$

At type `nat` we have numerals, arithmetic operators and a conditional that tests whether a number is equal to 0 or not.

$$\frac{}{\Gamma \vdash n \colon \mathtt{nat}} \qquad \frac{\Gamma \vdash M \colon \mathtt{nat}}{\Gamma \vdash \mathsf{succ}\, M \colon \mathtt{nat}} \qquad \frac{\Gamma \vdash M \colon \mathtt{nat}}{\Gamma \vdash \mathsf{pred}\, M \colon \mathtt{nat}}$$

$$\frac{\Gamma \vdash M \colon \mathtt{nat} \qquad \Gamma \vdash N \colon T \qquad \Gamma \vdash P \colon T}{\Gamma \vdash \mathsf{If0}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P \colon T}\ T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}$$

At type $\mathtt{Var}$, we have terms that call the read and write 'methods' to assign a new value to the variable or to get its current value.

$$\frac{\Gamma \vdash V : \mathtt{Var} \qquad \Gamma \vdash E : \mathtt{nat}}{\Gamma \vdash V \leftarrow E : \mathtt{com}} \qquad\qquad \frac{\Gamma \vdash V : \mathtt{Var}}{\Gamma \vdash\, !V : \mathtt{nat}}$$

We have the let keyword that allows us to evaluate an expression and bind the result to a variable name.

$$\frac{\Gamma \vdash M : S \qquad \Gamma, x : S \vdash N : T}{\Gamma \vdash \mathsf{let}_{S,T}\, x = M \;\mathsf{in}\; N : T}\; S \in \{\mathtt{bool}, \mathtt{nat}\};\; T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}$$

We have two ways to create terms of type $\mathtt{Var}$. The first gives us the storage-cell behaviour.

$$\frac{\Gamma, x : \mathtt{Var} \vdash M : T}{\Gamma \vdash \mathsf{new}_T\, \lambda x.M : T}\; T \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}$$

The idea here is that if M is a term that refers to some free variable x of type $\mathtt{Var}$; then $\mathsf{new}\, \lambda x.M$ makes x behave like an actual (zero-initialized) storage cell (so, for instance, the result of the computation $\mathsf{new}_{\mathtt{nat}}\, \lambda x.(x \leftarrow 5); !x$ will be 5).

The second way of creating terms of type $\mathtt{Var}$ is using the mkvar keyword. mkvar can be used to create 'bad variables': arbitrary pairs of read and write methods that do not necessarily actually behave like actual storage cells. If we think back to our earlier illustration of the $\mathtt{Var}$ type as an interface, this becomes clearer. mkvar creates a new anonymous instance of the $\mathtt{Var}$ interface, using the 'methods' supplied through its arguments.

$$\frac{\Gamma \vdash M : \mathtt{nat} \qquad \Gamma \vdash N : \mathtt{nat} \to \mathtt{com}}{\Gamma \vdash \mathsf{mkvar}\, MN : \mathtt{Var}}$$

Lastly, we have fixpoint combinators at all types that we use to implement recursion.

$$\frac{\Gamma \vdash M : T \to T}{\Gamma \vdash \mathbf{Y}_T M : T}$$

2.2 Games and Strategies

We adopt the game semantics from [AM96]. For the most part, these are very similar to the *arenas* defined by Hyland and Ong in [HO00]; however, they are slightly modified (through the prescription of a set of *legal plays*) in order to give us access to the substructural logic (in particular, linear logic) that we will use as the basis of the model of local state.

Definition 2.2.1. An *arena* is a tuple $A = (M_A, \lambda_A, \vdash_A)$, where

- M_A is a set of *moves*,
- $\lambda_A \colon M_A \to \{O, P\} \times \{Q, A\}$ is a function that identifies each move as either an *O-move* or a *P-move*, and as either a *question* or an *answer*, and
- $\vdash_A$ is a relation between $M_A + \{*\}$ and M_A such that
  - if $* \vdash_A a$, then $\lambda_A(a) = (O, Q)$, and if $b \vdash_A a$ then $b = *$,
  - if $a \vdash_A b$ and a is an answer, then b is a question, and
  - if $a \vdash_A b$ and $a \neq *$, then either a is an O-move and b a P-move, or the other way round.

If $* \vdash_A a$, then we say that a is an *initial move* in A. If $a \vdash_A b$, the we say that a *enables* b.

As a shorthand, we write $\lambda_A^{OP} \colon M_A \to \{O, P\}$ for $\mathrm{pr}_1 \circ \lambda_A$ and $\lambda_A^{QA} \colon M_A \to \{Q, A\}$ for $\mathrm{pr}_2 \circ \lambda_A$.

Definition 2.2.2. A *justified sequence* in an arena A is a finite sequence s of moves together with, for each non-initial move a occurring in s, a pointer from a back to some move b occurring earlier in s such that $b \vdash_A a$. We say that b *justifies* a or that b is the *justifier* of a.

Given such a justified sequence, we define the *P-view* $\ulcorner s \urcorner$ and *O-view* $\llcorner s \lrcorner$ of s inductively as follows.

$$
\begin{aligned}
\ulcorner \epsilon \urcorner &= \epsilon \\
\ulcorner sa \urcorner &= \ulcorner s \urcorner a && \text{if } a \text{ is a } P\text{-move} \\
\ulcorner sa \urcorner &= a && \text{if } a \text{ is initial} \\
\ulcorner sbta \urcorner &= \ulcorner s \urcorner ba && \text{if } a \text{ is an } O\text{-move justified by } b
\end{aligned}
$$

$$
\begin{aligned}
\llcorner \epsilon \lrcorner &= \epsilon \\
\llcorner sa \lrcorner &= \llcorner s \lrcorner a && \text{if } a \text{ is an } O\text{-move} \\
\llcorner sbta \lrcorner &= \llcorner s \lrcorner ba && \text{if } a \text{ is a } P\text{-move justified by } b
\end{aligned}
$$

A justified sequence s is *well-bracketed* if whenever a question q justifies some answer a, then any question q' occurring after q and before a must justify some answer a' occurring between q' and a, and if a is, moreover, the

only answer justified by q:

We say that a justified sequence s is *alternating* if it alternates between O-moves and P-moves, and that it is *well-formed* if it is both well-bracketed and alternating.

We say that a well-formed justified sequence is *visible* if whenever $ta \sqsubseteq s$, and a is a P-move, then the justifier of a occurs in the P-view of t, and if whenever $tb \sqsubseteq s$, and b is a non-initial O-move, then the justifier of b occurs in the O-view of t.

We say that a justified sequence s is *legal* if it is well-formed and visible, and write $\mathcal{L}_A$ for the set of legal sequences occurring in A.

Note that since every non-initial move in a justified sequence s must be justified by some previous move, then the first move in the sequence must be initial and therefore an O-question. If s is alternating, this means that s ends with an O-move if it has odd length and with a P-move if it has even length.

We shall call an odd-length sequence $s \in P_A$ an *O-position* and an even-length sequence a *P-position*.

Definition 2.2.3. A *game* is a tuple

$$A = (M_A, \lambda_A, \vdash_A, P_A)$$

where $(M_A, \lambda_A, \vdash_A)$ is an arena and P_A is a non-empty prefix-closed subset of $\mathcal{L}_A$.

Example 2.2.4 (Empty game). The *empty game* I is given by the tuple

$$(\varnothing, \varnothing, \varnothing, \{\epsilon\}),$$

where ϵ is the empty sequence.

Example 2.2.5 (Data-type games). Let X be some set. Then we have a game, which we shall also call X, given by:

- $M_X = \{q\} + X$;
- $\lambda_X(q) = (O, Q)$ and $\lambda_X(x) = (P, A)$ for all $x \in X$;
- $q \vdash_X x$ for each $x \in X$; and
- $P_X = \{\epsilon, q\} \cup \{qx : x \in X\}$, where the x in qx is justified by q.

In particular, we have games $\mathbb{C}$, $\mathbb{B}$ and $\mathbb{N}$, which we shall use to model the datatypes `com`, `bool` and `nat` of Idealized Algol.

Given a game A, there are several ways to capture the natural notion of a strategy for A for player P. The most convenient way is to identify a strategy with the set of P-positions that may occur if P plays according to that strategy.

Definition 2.2.6. Let A be a game. Then a *strategy* for A is a non-empty even-prefix-closed set $\sigma \subseteq P_A$ of P-positions in A such that if $sab, sac \in \sigma$ then $b = c$ and the justifier of b is the justifier of c.

Here, we have identified a strategy for a game with the set of P-positions that can occur when player P plays according to that strategy. So the condition we have given is one of *determinism*: in any O-position sa that can occur in the strategy, player P must have at most one reply.

Note that there may be O-positions for which player P has no reply at all; we use these to model non-terminating computations.

We write $\sigma : A$ to denote that σ is a strategy for the game A.

Definition 2.2.7. A strategy σ for a game A is called *innocent* if player P's moves only depend on the current P-view; i.e., if whenever $sab \in \sigma$, $t \in \sigma$ and $ta \in P_A$ such that $\ulcorner sa \urcorner = \ulcorner ta \urcorner$, then we have $tab \in \sigma$.

2.3 Connectives on Games

In the *product* $A \times B$ of games A and B, player O chooses either A or B on the first move and subsequent play is that game.

Definition 2.3.1. Given games A, B, define the *product* of A and B to be the game $A \times B$ given by

- $M_{A \times B} = M_A + M_B$;
- $\lambda_{A \times B} = [\lambda_A, \lambda_B]$;
- $* \vdash_{A \times B} a$ if and only if $* \vdash_A a$ or $* \vdash_B a$ and $a \vdash_{A \times B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$; and

- $P_{A\times B} = \{s \in \mathcal{L}_{A\times B} : \ s|_A \in P_A \ \&\ s|_B = \epsilon \text{ or } s|_A = \epsilon \ \&\ s|_B \in P_B\}$.

Here, we have written $s|_A$ for the subsequence of s consisting of all moves from M_A and $s|_B$ for the subsequence consisting of all moves from M_B.

We extend this to arbitrary products $\prod_i A_i$ in the obvious way. In particular, the product 1 of the empty collection is the same as the empty game I defined in Example 2.2.4.

Our interpretation of the type `Var` will be given by a particular infinite product; namely, the product of countably many copies of the game $\mathbb{C}$, each one corresponding to the action of writing a particular natural number into the variable, together with a copy of the game $\mathbb{N}$, corresponding to requesting the current value of the variable.

In the *tensor product* $A \otimes B$ of games A and B, the games A and B are played in parallel, and player O may switch between games when it is his turn.

Definition 2.3.2. Given games A,B, define a game $A \otimes B$ by

- $M_{A\otimes B} = M_A + M_B$;
- $\lambda_{A\otimes B} = [\lambda_A, \lambda_B]$;
- $* \vdash_{A\otimes B} a$ if and only if $* \vdash_A a$ or $* \vdash_B a$ and $a \vdash_{A\otimes B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$; and
- $P_{A\otimes B} = \{s \in \mathcal{L}_{A\otimes B} : \ s|_A \in P_A \text{ and } s|_B \in P_B\}$.

In the *linear implication* $A \multimap B$, the game B is played in parallel with a version of A in which the two players' roles have been switched around, and player P may switch between the two games when it is her turn.

Definition 2.3.3. Given games A,B, define a game $A \multimap B$ by

- $M_{A\multimap B} = M_A + M_B$;
- $\lambda_{A\multimap B} = [\neg \circ \lambda_A, \lambda_B]$;
- $* \vdash_{A\multimap B} a$ if and only if $* \vdash_B a$, and $a \vdash_{A\multimap B} b$ if and only if $a \vdash_A b$ or $a \vdash_B b$, or if a is initial in B and b is initial in a; and
- $P_{A\multimap B} = \{s \in \mathcal{L}_{A\multimap B} : \ s|_A \in P_A \text{ and } s|_B \in P_B\}$.

Here, $\neg\colon \{O, P\} \times \{Q, A\} \to \{O, P\} \times \{Q, A\}$ is the function that reverses O and P, while leaving $\{Q, A\}$ unchanged.

In the *exponential* of a game A, infinitely many copies of A are played in parallel, and player O may switch between copies whenever it is his move.

Definition 2.3.4. Given a game A, define a game $!A$ by

- $M_{!A} = M_A$;
- $\lambda_{!A} = \lambda_A$;
- $\vdash_{!A} = \vdash_A$; and
- $P_{!A} = \{s \in \mathcal{L}_{!A} : s|_b \in P_A \text{ for each initial move } b \text{ occurring in } s\}$.

Lastly, the *sequoid* $A \oslash B$ of two games A and B behaves like the tensor product $A \otimes B$, except that the opening move must take place in A.

Definition 2.3.5. Given games A, B, define a game $A \oslash B$ by

- $M_{A \oslash B} = M_{A \otimes B}$;
- $\lambda_{A \oslash B} = \lambda_{A \otimes B}$;
- $\vdash_{A \oslash B} = \vdash_{A \otimes B}$; and
- $P_{A \oslash B} = \{s \in P_{A \otimes B} : s = \epsilon \text{ or } s \text{ begins with a move from } A\}$.

2.4 Composition of strategies

Definition 2.4.1. Let A, B, C be arenas. An *interaction sequence* between A, B, C is a justified sequence $\mathfrak{s}$ of moves drawn from M_A, M_B and M_C such that $\mathfrak{s}|_{A,B} \in \mathcal{L}_{A \multimap B}$ and $\mathfrak{s}|_{B,C} \in \mathcal{L}_{B \multimap C}$. Here, $\mathfrak{s}|_{A,B}$ is the subsequence of $\mathfrak{s}$ consisting of those moves from $\mathfrak{s}$ that occur in A or B, together with all justification pointers between moves in A and B, and $\mathfrak{s}|_{B,C}$ is defined similarly.

We write $\mathrm{int}(A, B, C)$ for the set of all interaction sequences between A, B, C.

Given $\mathfrak{s} \in \mathrm{int}(A, B, C)$, we write $\mathfrak{s}|_{A,C}$ for the subsequence of $\mathfrak{s}$ consisting of those moves from $\mathfrak{s}$ that occur in A or C. A move b in $\mathfrak{s}|_{A,C}$ justifies a move a either if b justifies a in either the A or the C components, or if b justifies in $\mathfrak{s}$ some initial move c in B, which itself justifies a.

Definition 2.4.2. Let A, B, C be games, let σ be a strategy for $A \multimap B$ and let τ be a strategy for $B \multimap C$. We define $\sigma \| \tau$ to be given by the set

$$\{\mathfrak{s} \in \mathrm{int}(A, B, C) : \mathfrak{s}|_{A,B} \in \sigma \text{ and } \mathfrak{s}|_{B,C} \in \tau\}.$$

Then we define the *composition* $\sigma; \tau$ of σ and τ to be given by the set

$$\{\mathfrak{s}|_{A,C} : \mathfrak{s} \in \sigma \| \tau\}.$$

We have not yet shown that this is indeed a well-formed strategy; the proof of this fact will take us a few pages. We start by proving some lemmata and making some definitions.

Lemma 2.4.3. *We extend the function λ_A^{OP} to alternating sequences of moves by*

- $\lambda_A^{OP}(\epsilon) = P$ *and*
- $\lambda_A^{OP}(sa) = \lambda_A(a)$.

If $s \in P_{A \multimap B}$, then $\lambda_{A \multimap B}^{OP}(s) = (\lambda_A^{OP}(s|_A) \Rightarrow \lambda_B^{OP}(s|_B))$, where $\Rightarrow$ is the binary operation on $\{O, P\}$ defined by

P	Q	$P \Rightarrow Q$
P	P	P
O	P	P
P	O	O
O	O	P

.

Moreover, if $\lambda_A^{OP}(s|_A) = O$ then $\lambda_A^{OP}(s|_B) = O$. In other words, the second row of the table above never occurs.

Proof. Induction on the length of s. If $s = \epsilon$, then $s|_A = s|_B = \epsilon$, and so

$$(\lambda_A^{OP}(s|_A) \Rightarrow \lambda_B^{OP}(s|_B)) = (P \Rightarrow P) = P = \lambda_{A \multimap B}^{OP}(s)\,.$$

Suppose then that $s = ta$, and that $\lambda_{A \multimap B}^{OP}(t) = O$. This means that

$$\lambda_A^{OP}(t|_A) = P \qquad\qquad \lambda_B^{OP}(t|_B) = O\,.$$

Then, whether a is a move in A or a move in B, adding it will flip exactly one of these components – so

$$\lambda_{A \multimap B}(s|_A) = O \qquad\qquad \lambda_{A \multimap B}^{OP}(s|_B) = O$$

if a is a move in A, and

$$\lambda_{A \multimap B}(s|_A) = P \qquad\qquad \lambda_{A \multimap B}^{OP}(s|_B) = P$$

if a is a move in C.

Suppose instead that $\lambda_{A \multimap B}^{OP}(t) = P$. By induction, this means that either

$$\lambda_A^{OP}(t|_A) = P \qquad\qquad \lambda_B^{OP}(t|_B) = P$$

or that

$$\lambda_A^{OP}(t|_A) = O \qquad\qquad \lambda_B^{OP}(t|_B) = O\,.$$

In the first case, this means that either $t|_A$ is empty or its last move is a P-move in A (and therefore an O-move in $A \multimap B$), and so the move a must take place in C, meaning that $\lambda_A^{OP}(s|_A) = P$ and $\lambda_B^{OP}(s|_B) = O$.

Similarly, in the second case, the last move in $t|_C$ must be an O-move in B (and therefore an O-move in $A \multimap B$, and so the move a must take place in A, meaning that $\lambda_A^{OP}(s|_A) = P$ and $\lambda_B^{OP}(s|_B) = O$. □

It follows that

Corollary 2.4.4 (Switching condition)**.** *Only player P may switch between games in $A \multimap B$; i.e., if $tab \in P_{A \multimap B}$, and a occurs in A and b in B, or if a occurs in B and b in A, then b is a P-move.*

Proof. Otherwise, $\lambda_{A \multimap B}(t) = O$, so $\lambda_A(t|_A) = P$ and $\lambda_B(t|_B) = O$. But we must also have $\lambda_{A \multimap B}(tab) = O$, so $\lambda_A(tab|_A) = P$ and $\lambda_B(tab|_B) = O$. But this is a contradiction, since $tab|_A$ and $tab|_B$ are both one move longer than the plays $t|_A$ and $t|_B$, so $\lambda_A(tab|_A) \neq \lambda_A(t|_A)$ and $\lambda_B(tab|_B) \neq \lambda_B(t|_B)$. □

Definition 2.4.5 ([Har06, §3.1])**.** Given $\mathfrak{s} \in \mathrm{int}(A, B, C)$, we define the *$P$-view* $\ulcorner \mathfrak{s} \urcorner$ of $\mathfrak{s}$ inductively as follows.

$$
\begin{array}{rll}
\ulcorner \epsilon \urcorner = \epsilon & & \\
\ulcorner \mathfrak{s} a \urcorner = \ulcorner \mathfrak{s} \urcorner a & \text{if } a \text{ is a move in } B, \text{ an } O\text{-move in } A \text{ or a } P\text{-move in } C \\
\ulcorner \mathfrak{s} c \urcorner = c & \text{if } c \text{ is an initial move of } C \\
\ulcorner \mathfrak{s} b \mathfrak{t} a \urcorner = \ulcorner \mathfrak{s} \urcorner b a & \text{if } a \text{ is a } P\text{-move of } A \text{ or an } O\text{-move of } C \\
 & \text{and is justified by } b
\end{array}
$$

Lemma 2.4.6. *If $\mathfrak{s} \in \mathrm{int}(A, B, C)$, then $\ulcorner \mathfrak{s} \urcorner|_{A,C} = \ulcorner \mathfrak{s}|_{A,C} \urcorner$.*

Proof. Induction on the length of $\mathfrak{s}$. This is clear if $\mathfrak{s} = \epsilon$.

If a is an O-move in A or a P-move in C, then a is a P-move in $A \multimap C$. We have $\ulcorner \mathfrak{s} a \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C} a$, which by the inductive hypothesis is equal to $\ulcorner \mathfrak{s}|_{A,C} \urcorner a$, which is the same as $\ulcorner \mathfrak{s} a|_{A,C} \urcorner$. If b is a move in B, then

$$\ulcorner \mathfrak{s} b \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner b|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C} = \ulcorner \mathfrak{s}|_{A,C} \urcorner = \ulcorner \mathfrak{s} b|_{A,C} \urcorner,$$

by the inductive hypothesis.

If c is initial in C, then $\ulcorner \mathfrak{s} c \urcorner|_{A,C} = c = \ulcorner \mathfrak{s} c|_{A,C} \urcorner$.

Suppose a is a P-move of A or an O-move of C – so a is an O-move in $A \multimap C$ – and suppose that a is justified by b in the sequence $\mathfrak{s} b \mathfrak{t} a$. Since a

cannot be an initial move in A, b must occur in the same game as a, and in particular must not occur in B. Then we have

$$\ulcorner \mathfrak{s} b \mathfrak{t} a \urcorner|_{A,C} = \ulcorner \mathfrak{s} \urcorner b a|_{A,C} = \ulcorner \mathfrak{s} \urcorner|_{A,C} b a\,,$$

which by the inductive hypothesis is equal to $\ulcorner \mathfrak{s}|_{A,C} \urcorner b a = \ulcorner \mathfrak{s} b a|_{A,C} \urcorner$. □

Lemma 2.4.7 ([Har06, §3.1])**.** *Let $\mathfrak{s}a \in \mathrm{int}(A, B, C)$ (so, in particular, $\mathfrak{s}|_{A,B}$ and $\mathfrak{s}|_{B,C}$ satisfy the visibility condition). If a is a move in B, an O-move in A or a P-move in C, then $\ulcorner \mathfrak{s} a \urcorner \in \mathrm{int}(A, B, C)$.*

Proof. Induction on the length of $\mathfrak{s}$. If $\mathfrak{s} = \epsilon$, then this is clear. Otherwise, suppose that $\mathfrak{s}$ is non-empty.

First, we claim that $\ulcorner \mathfrak{s} \urcorner \in \mathrm{int}(A, B, C)$. If $\mathfrak{s}$ ends with a move in B, an O-move in A or a P-move in C, then this follows immediately from the inductive hypothesis. Otherwise, suppose that $\mathfrak{s}$ ends with a P-move in A or an O-move in C. If this last move is initial, then $\ulcorner \mathfrak{s} \urcorner$ is a single move, so the claim is trivial. Otherwise, write $\mathfrak{s} = \mathfrak{t} p \mathfrak{u} r$, where p justifies r. By the inductive hypothesis, we have $\ulcorner \mathfrak{t} p \urcorner \in \mathrm{int}(A, B, C)$, and then $\ulcorner s \urcorner = \ulcorner \mathfrak{t} p \mathfrak{u} r \urcorner = \ulcorner \mathfrak{t} \urcorner p r = \ulcorner \mathfrak{t} p \urcorner r \in \mathrm{int}(A, B, C)$.

Now, since a is a P-move in $A \multimap B$ or in $B \multimap C$, its predecessor b is an O-move and has some justifier c contained in $\ulcorner \mathfrak{s}|_X \urcorner$, where $X \in \{A \multimap B, B \multimap C\}$ is that component in which a is a P-move. Then this c is preceded by some other O-move b', which is necessarily also contained in $\ulcorner \mathfrak{s} \urcorner$, and so has some justifier c', contained in $\ulcorner \mathfrak{s}|_X \urcorner$ by visibility. Continuing in this way until we reach an initial move, we build up the whole of the sequence $\ulcorner \mathfrak{s}|_X \urcorner$ as a subsequence of $\ulcorner \mathfrak{s} \urcorner$. Therefore, the justifier of a must be contained in $\ulcorner \mathfrak{s} \urcorner$, and so $\ulcorner \mathfrak{s} a \urcorner = \ulcorner \mathfrak{s} \urcorner a \in \mathrm{int}(A, B, C)$. □

Lemma 2.4.8 (O-views in the linear implication, [HO00, 4.2,4.3])**.** *Let A, B be games, and let bs be a non-empty play in $A \multimap B$ beginning with an initial move b in B.*

i) If bs ends with a P-move in B, then $\llcorner bs \lrcorner_{A \multimap B} = \llcorner bs|_B \lrcorner_B$.

ii) If bs ends with a P-move in A, then $\llcorner bs \lrcorner_{A \multimap B} = b \ulcorner s|_A \urcorner^A$.

Proof. Induction on the length of s. If $s = \epsilon$, then bs ends with an O-move in B, and we have $\llcorner b \lrcorner^{A \multimap B} = b = \llcorner b \lrcorner^B$.

Otherwise, suppose that bs ends with a P-move c in B. Let d be the justifier of c. Then d must be an O-move in B. Write $bs = tduc$, where t, u are sequences. Then $\llcorner tduc \lrcorner_{A \multimap B} = \llcorner t \lrcorner_{A \multimap B} dc$ and $\llcorner tduc|_B \lrcorner_B = \llcorner t|_B \lrcorner_B dc$. By Corollary 2.4.4, t must end with a P-move in B, or be empty, so by the

inductive hypothesis we have $\llcorner t \lrcorner_{A \multimap B} = \llcorner t|_B \lrcorner_B$. Therefore, $\llcorner bs \lrcorner_{A \multimap B} = \llcorner tduc \lrcorner_{A \multimap B} = \llcorner t \lrcorner_B dc = \llcorner tduc|_B \lrcorner_B = \llcorner bs|_B \lrcorner_B$.

Next, suppose that bs ends with a P-move a in A. Let c be the justifier of a. Then c must be an O-move in A. Write $s = tcua$, where t, u are sequences. Then $\llcorner btcua \lrcorner_{A \multimap B} = \llcorner bt \lrcorner_{A \multimap B} ca$ and $\ulcorner tcua|_A \urcorner^A = \ulcorner t|_A \urcorner^A ca$, since the roles are reversed in A. By Corollary 2.4.4, t must end in a P-move in A, or be empty, so by the inductive hypothesis we have $\llcorner bt \lrcorner_{A \multimap B} = b \ulcorner t|_A \urcorner^A$. Therefore, $\llcorner bs \lrcorner_{A \multimap B} = \llcorner btcua \lrcorner_{A \multimap B} = \llcorner bt \lrcorner_{A \multimap B} ca = b \ulcorner t|_A \urcorner^A ca = b \ulcorner tcua|_A \urcorner^A = b \ulcorner s|_A \urcorner^A$. □

Proposition 2.4.9. *$\sigma; \tau$ is a strategy for $A \multimap C$.*

Proof. First, we claim that $\mathfrak{s}|_{A,C} \in P_{A \multimap B}$ for any $\mathfrak{s} \in \sigma \| \tau$. Since we certainly have $\mathfrak{s}|_{A,C}|_A = \mathfrak{s}|_{A,B}|_A \in P_A$ and $\mathfrak{s}|_{A,C}|_C = \mathfrak{s}|_C = \mathfrak{s}|_{B,C}|_C \in P_C$, it suffices to show that $\mathfrak{s}|_{A,C} \in \mathcal{L}_{A \multimap C}$.

Suppose that $ta \sqsubseteq \mathfrak{s}|_{A,C}$. We claim that $\lambda_{A \multimap C}(t) = \neg \lambda_{A \multimap C}(a)$. By Lemma 2.4.3, we are in one of the following configurations.

$\lambda_A^{OP}(t\|_A)$	$\lambda_B^{OP}(t\|_B)$	$\lambda_C^{OP}(t\|_C)$	$\lambda_{A \multimap B}^{OP}(t\|_{A,B})$	$\lambda_{B \multimap C}^{OP}(t\|_{B,C})$	$\lambda_{A \multimap C}^{OP}(t\|_{A,C})$
P	P	P	P	P	P
P	P	O	P	O	O
P	O	O	O	P	O
O	O	O	P	P	P

In the configuration PPP, the move a cannot be a move in A, since that would leave $ta|_{A \multimap B}$ in the configuration OP, which is impossible by Lemma 2.4.3. Therefore, it must be a move in C, and must therefore be an O-move in C and hence an O-move in $A \multimap C$.

In the configuration PPO, once again the move a cannot take place in A, since this would leave $ta|_{A \multimap B}$ in the illegal configuration OP. Therefore, it must occur in C, and must be a P-move in C and hence a P-move in $A \multimap C$.

In the configuration POO, the move a cannot take place in C, or it would leave $ta|_{B,C}$ in the illegal configuration OP, so the move a takes place in A. Therefore, it must be an O-move in A and hence a P-move in $A \multimap C$.

Lastly, in the configuration OOO, the move a cannot occur in C, or it would leave $ta|_{B,C}$ in the configuration OP, and so it must take place in A. Therefore, it must be a P-move in A, and hence an O-move in $A \multimap C$.

Having established that $\mathfrak{s}|_{A,C}$ is alternating, we now show that it is well-bracketed. Suppose that a question move q in $\mathfrak{s}|_{A,C}$ justifies some answer move a. q and a must occur in the same component, since the only case in which a move from one of A and C can justify a move in the other is when

both moves are initial, and hence questions. Suppose first that q and a both occur in the game C. Suppose that some other question move q' occurs between q and a in $\mathfrak{s}|_{A,C}$. If q' occurs in C, then it must be answered by some a' occurring between q' and a, since $\mathfrak{s}|_C$ is a well-bracketed sequence. Otherwise, suppose that q' occurs in A.

By examining the table above, we see that there must be some move in B occurring between q and q' in $\mathfrak{s}$, since moves in A move between configurations OOO and POO, while moves in C move us between configurations PPP and PPO. Let b be the earliest such move. Then b must be a question; indeed, if it is an answer, then it is non-initial and so can only be justified by questions in B. But such a question must occur earlier in $\mathfrak{s}|_{B,C}$ than q, which would mean that q was an unanswered question when the move b was played, contradicting well-bracketedness of $\mathfrak{s}|_{B,C}$. Since b is a question, it must be answered by some a'' occurring between b and a. Therefore, since $\mathfrak{s}|_{A,B}$ is well-bracketed, the move q' must be answered by some a' occurring between a' and a'' in $\mathfrak{s}|_{A,B}$, and therefore between a' and a in $\mathfrak{s}|_{A,C}$.

The case when q and a both occur in A is similar.

Lastly, we need to show that $\mathfrak{s}|_{A,C}$ satisfies the visibility condition. Let $ta \sqsubseteq \mathfrak{s}|_{A,C}$. Choose some $\mathfrak{t}a \sqsubseteq \mathfrak{s}$ such that $\mathfrak{t}a|_{A,C} = ta$.

Suppose a is a P-move. Then by Lemma 2.4.7, $\ulcorner \mathfrak{t}a \urcorner \in \mathrm{int}(A, B, C)$. By Lemma 2.4.6, we know that $\ulcorner t \urcorner a = \ulcorner ta \urcorner = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$, and therefore that the justifier of a must be inside $\ulcorner t \urcorner$.

Secondly, suppose that a is an O-move. If a is an O-move in C, then either it is initial or $\mathfrak{t}$ ends with some P-move in C, and therefore $\llcorner t \lrcorner_{A \multimap C} = \llcorner t|_C \lrcorner_C = \llcorner \mathfrak{t}|_{B,C} \lrcorner_C$ by Lemma 2.4.8. Therefore, since $\mathfrak{t}|_{B,C}$ satisfies visibility, the justifier of a must lie in $\llcorner t \lrcorner_{A \multimap C}$. If a is an O-move in A, then write $t = cu$ and $\mathfrak{t} = c\mathfrak{u}$, where c is the starting move in C. We have $\llcorner cua \lrcorner_{A \multimap C} = c^{\ulcorner} u|_A a^{\urcorner_A} = \llcorner c\mathfrak{u}|_{A,B} \lrcorner_{A \multimap B}$. Therefore, the justifier of a must lie in $\llcorner t \lrcorner_{A \multimap C}$.

Therefore, $\mathfrak{s}|_{A,C} \in \mathcal{L}_{A \multimap C}$, so $\mathfrak{s}|_{A,C} \in P_{A \multimap C}$.

It is fairly clear that $\sigma; \tau$ is even-prefix closed, since σ and τ are. Indeed, if $\mathfrak{s}|_{A,C} \in \sigma; \tau$ and $t \sqsubseteq \mathfrak{s}|_{A,C}$, then we may choose some prefix $\mathfrak{t}$ of $\mathfrak{s}$ such that $t = \mathfrak{t}|_{A,C}$. Then $\mathfrak{t}|_{A,B} \sqsubseteq \mathfrak{s}|_{A,B} \in \sigma$ and $\mathfrak{t}|_{B,C} \sqsubseteq \mathfrak{s}|_{B,C} \in \tau$, so $\mathfrak{t} \in \sigma \| \tau$.

We claim that every sequence in $\sigma; \tau$ has even length. Indeed, if $\mathfrak{s}|_{A,B} \in \sigma$ and $\mathfrak{s}|_{B,C} \in \tau$, then both $\mathfrak{s}|_{A,B}$ and $\mathfrak{s}|_{B,C}$ must have even length, so must be in configuration OO or PP. This means that $\mathfrak{s}$ as a whole must be in configuration OOO or PPP, and so $\mathfrak{s}|_{A,C}$ must be in configuration OO or PP, so must have even length.

Lastly, we need to show that $\sigma; \tau$ is deterministic. Suppose that $sab, sac \in \sigma; \tau$, and suppose that $b \neq c$. Suppose that $\mathfrak{s}|_{A,C} = sab$ and $\mathfrak{t}|_{A,C} = sac$,

for $\mathfrak{s}, \mathfrak{t} \in \sigma \| \tau$, and let $\mathfrak{u}$ be the longest common prefix of $\mathfrak{s}, \mathfrak{t}$. $\mathfrak{s}$ and $\mathfrak{t}$ are certainly incomparable under the prefix ordering, since $\mathfrak{s}|_{A,C}$ and $\mathfrak{t}|_{A,C}$ are, so we have $\mathfrak{u}p \sqsubseteq \mathfrak{s}$ and $\mathfrak{u}q \sqsubseteq \mathfrak{t}$, where $p \neq q$. Now p and q cannot be O-moves in A, P-moves in C or moves in B, or they would have to be equal by determinism of σ and τ. Therefore, they are P-moves in A or O-moves in C, but this contradicts $\mathfrak{s}|_{A,C} = sab$ and $\mathfrak{t}|_{A,C} = sac$.

Therefore, the composition $\sigma; \tau$ is a strategy. □

We also want to show that the composition of innocent strategies is innocent. We follow the proof given in [Har06]. First, we use a lemma.

Lemma 2.4.10 ([Har06, 3.3.3])**.** *Let $\mathfrak{s}a \in \mathrm{int}(A, B, C)$.*

i) If a is a P-move of A or an O-move of B, then $\ulcorner \mathfrak{s}a|_{A,B} \urcorner = \ulcorner \ulcorner \mathfrak{s}a \urcorner|_{A,B} \urcorner$.

ii) If a is a P-move of B or an O-move of C, then $\ulcorner \mathfrak{s}a|_{B,C} \urcorner = \ulcorner \ulcorner \mathfrak{s}a \urcorner|_{B,C} \urcorner$.

Proof. Induction on the length of $\mathfrak{s}$. We prove (i); the proof of (ii) is exactly the same.

If a is a P-move of A or an O-move of B, then it is an O-move of $A \multimap B$. If a is an initial move of $A \multimap B$, then we have $\ulcorner \mathfrak{s}|_{A,B} a \urcorner = a = \ulcorner a \urcorner|_{A,B} = \ulcorner \ulcorner \mathfrak{s}a \urcorner \urcorner|_{A,B}$. Otherwise, write $\mathfrak{s} = \mathfrak{t}b\mathfrak{u}$, where b justifies a. Then $\ulcorner \mathfrak{s}a|_{A,B} \urcorner = \ulcorner \mathfrak{t}|_{A,B} b\mathfrak{u}|_{A,B} a \urcorner = \ulcorner t|_{A,B} \urcorner ba$, which by the inductive hypothesis is equal to $\ulcorner \ulcorner \mathfrak{t} \urcorner|_{A,B} \urcorner ba$, which is equal to $\ulcorner \ulcorner \mathfrak{t}b\mathfrak{u}a \urcorner|_{A,B} \urcorner = \ulcorner \ulcorner \mathfrak{s}a \urcorner|_{A,B} \urcorner$. □

Proposition 2.4.11. *If $\sigma \colon A \multimap B$ and $\tau \colon B \multimap C$ are innocent strategies, then $\sigma; \tau \colon A \multimap C$ is innocent.*

Proof. Suppose there are $sab, t \in \sigma; \tau$ such that $ta \in P_{A \multimap C}$, $\ulcorner sa \urcorner = \ulcorner ta \urcorner$. Let $\mathfrak{s}'b$ be such that $\mathfrak{s}'b|_{A,C} = sab$ and choose the minimal prefix $\mathfrak{s} \sqsubseteq \mathfrak{s}'$ such that $\mathfrak{s}a|_{A,C} = sa$.

Let $\mathfrak{t}a$ be such that $\mathfrak{t}a|_{A,C} = ta$. Since $\ulcorner sa \urcorner = \ulcorner ta \urcorner$, we have $\ulcorner \mathfrak{s}a \urcorner|_{A,C} = \ulcorner \mathfrak{s}a|_{A,C} \urcorner = \ulcorner sa \urcorner = \ulcorner ta \urcorner = \ulcorner \mathfrak{t}a|_{A,C} \urcorner = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$ by Lemma 2.4.6. Let $\mathfrak{u}$ be the longest common prefix of $\ulcorner \mathfrak{s}a \urcorner$ and $\ulcorner \mathfrak{t}a \urcorner$. If $\mathfrak{s}a$ and $\mathfrak{t}a$ are not equal, then without loss of generality there is some $\mathfrak{u}p \sqsubseteq \mathfrak{s}$, where $\mathfrak{u}p \not\sqsubseteq \mathfrak{t}$. Then, by determinism of σ and τ, this p cannot be a P-move in either $A \multimap B$ or $B \multimap C$, so it must be a P-move in A or an O-move in C, and is therefore preceded by another move in A or C, which contradicts $\ulcorner \mathfrak{s}a \urcorner|_{A,C} = \ulcorner \mathfrak{t}a \urcorner|_{A,C}$. Therefore, $\ulcorner \mathfrak{s}a \urcorner = \ulcorner \mathfrak{t}a \urcorner$.

Now write $\mathfrak{s}' = \mathfrak{s}ab_1 \cdots b_n b$, where each b_i is a move in B. We show by induction that $\mathfrak{t}ab_1 \cdots b_j \in \sigma \| \tau$. Indeed, if $\mathfrak{t}ab_1 \cdots b_{j-1} \in \sigma \| \tau$, then b_j (or b) is a P-move in either $A \multimap B$ or $B \multimap C$, and b_{j-1} is an O-move in that same component. Write X for the component ($A \multimap B$ or $B \multimap C$) in which

b_j is a P-move. Repeating the argument above, we see that $\ulcorner \mathfrak{t} a b_1 \cdots b_{j-1} \urcorner = \ulcorner \mathfrak{s} a b_1 \cdots b_{j-1} \urcorner$, and so we have that $\ulcorner \mathfrak{t} a b_1 \cdots b_{j-1}|_X \urcorner = \ulcorner \mathfrak{s} a b_1 \cdots b_{j-1}|_X \urcorner$ by Lemma 2.4.10. Therefore, by innocence of σ (if $X = A \multimap B$) or τ (if $X = B \multimap C$), we see that $\mathfrak{t} a b_1 \cdots b_j \in \sigma \| \tau$. It follows that $\mathfrak{t} a b_1 \cdots b_n b \in \sigma \| \tau$, and therefore that $tab \in \sigma; \tau$. □

2.5 Associativity of composition

In this section, we shall prove that composition of strategies is associative; i.e., that if $\sigma \colon A \multimap B$, $\tau \colon B \multimap C$ and $\upsilon \colon C \multimap D$ are strategies, then $(\sigma; \tau); \upsilon = \sigma; (\tau; \upsilon)$. To do this, if A, B, C, D are arenas, we define the set $\mathrm{int}(A, B, C, D)$ to be the set of all sequences $\mathfrak{u}$ of moves such that $\mathfrak{u}|_{A,B} \in \mathcal{L}_{A \multimap B}$, $\mathfrak{u}|_{B,C} \in \mathcal{L}_{B \multimap C}$ and $\mathfrak{u}|_{C,D} \in \mathcal{L}_{C \multimap D}$. Given such a sequence $\mathfrak{u}$, we define $\mathfrak{u}|_{A,D}$ as before; i.e., we take all moves from $\mathfrak{u}$ occurring in A and D, together with justification pointers within these games, and if an initial move in A is justified by an initial move in B, which is justified by an initial move in C, which is justified by an initial move in D, then we add a justification pointer from that move in A to that move in D.

Given strategies σ, τ, υ as above, we define $\sigma \| \tau \| \upsilon$ to be the set of all $\mathfrak{u} \in \mathrm{int}(A, B, C, D)$ such that $\mathfrak{u}|_{A,B} \in \sigma$, $\mathfrak{u}|_{B,C} \in \tau$ and $\mathfrak{u}|_{C,D} \in \upsilon$. We then claim that:

Proposition 2.5.1.

$$(\sigma; \tau); \upsilon = \{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma \| \tau \| \upsilon\} = \sigma; (\tau; \upsilon)\,.$$

Proof. Firstly, if $\mathfrak{u} \in \sigma \| \tau \| \upsilon$, then it is clear to see that $\mathfrak{u}|_{A,B,C} \in \sigma \| \tau$ and that $\mathfrak{u}|_{B,C,D} \in \tau \| \upsilon$, and therefore that $\{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma \| \tau \| \upsilon\} \subseteq (\sigma; \tau); \upsilon$ and $\{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma \| \tau \| \upsilon\} \subseteq \sigma; (\tau; \upsilon)$.

Conversely, suppose that $\mathfrak{t} \in (\sigma; \tau) \| \upsilon$, so that $\mathfrak{t}|_{A,C} \in \sigma; \tau$ and $\mathfrak{t}|_{C,D} \in \upsilon$, and choose some $\mathfrak{s} \in \sigma \| \tau$ such that $\mathfrak{s}|_{A,C} = \mathfrak{t}|_{A,C}$. We may write

$$\mathfrak{s} = \mathbf{c}_1 \mathbf{b}_1 \mathbf{a}_1 \cdots \mathbf{c}_n \mathbf{b}_n \mathbf{a}_n$$

for some (possibly empty) sequences of moves $\mathbf{a}_i$ from A, $\mathbf{b}_i$ from B and $\mathbf{c}_i$ from C. We may then write

$$\mathfrak{t} = \mathbf{d}_1 \mathbf{c}_1 \mathbf{a}_1 \cdots \mathbf{d}_n \mathbf{c}_n \mathbf{a}_n$$

(for the same $\mathbf{a}_i$, $\mathbf{c}_i$), and we can therefore interleave these sequences into the sequence

$$\mathfrak{u} = \mathbf{d}_1 \mathbf{c}_1 \mathbf{b}_1 \mathbf{a}_1 \cdots \mathbf{d}_n \mathbf{c}_n \mathbf{b}_n \mathbf{a}_n\,,$$

which is in $\sigma \| \tau \| \upsilon$. Then we have $\mathfrak{u}|_{A,D} = \mathfrak{t}|_{A,D}$, and it follows that $(\sigma; \tau); \upsilon \subseteq \{\mathfrak{u}|_{A,C} \,:\, \mathfrak{u} \in \sigma \| \tau \| \upsilon\}$, and the case for $\sigma; (\tau; \upsilon)$ is identical. □

2.6 Copycat strategies

Definition 2.6.1. Let A, B be games. Then a *subset inclusion* of A into B is a partial injection $i\colon M_A \hookrightarrow M_B$ such that

- if i is defined at a and b then $* \vdash_A a$ if and only if $* \vdash_B i(a)$, and $a \vdash_A b$ if and only if $i(a) \vdash_B i(b)$;
- $i(a)$ is defined for every move a occurring in a play in P_A; and
- $i_*(s) \in P_B$ for every $s \in P_A$.

Here, $i_*(s)$ means the function i applied pointwise to the elements of the string s, leaving justification indices as they are in s.

If i is a subset inclusion of A into B, then we get an innocent strategy $\mathrm{subs}_i\colon B \multimap A$ defined by

$$\mathrm{subs}_i = \{s \in P_{B \multimap A} : \text{ for all even-length } t \sqsubseteq s,\ t|_B = i_*(t|_A)\}\,.$$

If $P_B = \{i_*(s) : s \in P_A\}$, then we call i a *structural isomorphism*, and we write cc_i ('copycat') instead of subs_i.

Example 2.6.2. If, in particular, $A = B$ and i is the identity, then the resulting strategy is sometimes called *the* copycat strategy on $A \multimap A$. It will take the role of the identity in our categorical semantics.

Example 2.6.3. If A and B are games, then the set of plays of $A \oslash B$ is a subset of the set of plays of $A \otimes B$. This will give us a subset inclusion morphism $A \otimes B \multimap A \oslash B$.

Example 2.6.4. If (A_i) is a collection of games, then there is a natural injection from the set of plays for A_j into the set of plays for $\prod_i A_i$. This gives us projection morphisms $\prod_i A_i \multimap A_j$ for each j.

Proposition 2.6.5. subs_i *is an innocent strategy.*

Moreover, if $\sigma\colon C \multimap B$ *is a strategy, then*

$$\sigma; \mathrm{subs}_i = \{[\mathrm{id}_{M_C}, i^{-1}]_*(s) : s \in \sigma,\ s|_B \in i_*(P_A)\}\,,$$

where $i^{-1}\colon M_B \rightharpoonup M_A$ *is the canonical partial right-inverse to* i*, and if* $\tau\colon A \multimap D$ *is a strategy, then*

$$\mathrm{subs}_i; \tau = \{[i, \mathrm{id}_{M_D}]_*(s) : s \in \tau\}\,.$$

Proof. subs_i is clearly prefix-closed by definition. Suppose that $sab, sac \in \mathrm{subs}_i$; then $s|_A = i_*(s|_B)$ and $sab|_A = i_*(sab|_B)$. It follows that $ab|_A =$

$i_*(ab|_B)$, so either a is a move in A and $b = i(a)$ or a is a move in B and $a = i(b)$. Since the same applies to c, and since i is injective, we have $b = c$.

This argument also shows that subs_i is *history-free* – i.e., that its reply to an O-position is entirely determined by the last O-move – and therefore that it is certainly innocent.

Now let $\sigma : C \multimap B$ be a strategy. Suppose that $\mathfrak{s} \in \sigma \| \mathrm{subs}_i$. Then $\mathfrak{s}|_{C,B} \in \sigma$ and $\mathfrak{s}|_B = i_*(\mathfrak{s}|_A)$; i.e.,

$$\mathfrak{s}|_{C,B} = [\mathrm{id}_{M_C}, i]_*(\mathfrak{s}|_{C,A}) \,,$$

and therefore

$$\mathfrak{s}|_{C,A} = [\mathrm{id}_{M_C}, i]_*(\mathfrak{s}|_{C,B}) \,,$$

where $\mathfrak{s}|_B \in i_*(P_A)$.

Conversely, given $s \in \sigma$, where $s|_B \in i_*(P_A)$, for each P-move $b = i(a)$ in s occurring in the component B, insert the move a immediately after it, and for each O-move $b' = i(a')$ in s occurring in the component B, insert the move a' immediately before it. Let these extra moves in B be justified according to the original moves in A, and let all initial moves in B be justified by the initial moves in A that occur immediately before them. Then the resulting sequence $\mathfrak{s}$ is contained in $\sigma \| \mathrm{subs}_i$, and $\mathfrak{s}|_{A,C} = [\mathrm{id}_{M_C}, i^{-1}]_*(s)$.

The case for composition in the other direction is similar. □

An easy corollary of this fact is that composition of copycat strategies respects composition of the underlying subset inclusions.

Corollary 2.6.6. *Let i be a subset inclusion from A to B and let j be a subset inclusion from B to C. Then $j \circ i$ is a subset inclusion from A to C and* $\mathrm{subs}_{j \circ i} = \mathrm{subs}_j ; \mathrm{subs}_i : C \multimap A$.

It is also easy to see from Proposition 2.6.5 that the identity function $\mathrm{id} : M_A \to M_A$ is a structural isomorphism from A to itself, and that the resulting copycat strategy $\mathrm{cc}_{\mathrm{id}}$ is an identity for composition. Combining this with our result for associativity in the previous section (Proposition 2.5.1), we get that

Theorem 2.6.7. *The collection of games forms a category $\mathcal{G}$, where the morphisms $A \to B$ are strategies for $A \multimap B$, composition is as above and the identity morphisms are the copycat strategies induced from the identity functions on moves.*

In this setting, Proposition 2.6.5 tells us that a structural isomorphism gives rise to an isomorphism in $\mathcal{G}$.

Proposition 2.6.8. *Let f be a structural isomorphism from a game A to a game B. Then cc_f is an isomorphism in $\mathcal{G}$ from A to B.*

Proof. The underlying partial injection $f : M_A \hookrightarrow M_B$ has an inverse partial injection $f^{-1} : M_B \to M_A$, inducing a structural isomorphism from B to A. Then Proposition 2.6.5 tells us that cc_f and $\mathrm{cc}_{f^{-1}}$ are inverses in $\mathcal{G}$. □

General subset inclusions are not, of course, isomorphisms, but we can still say something category-theoretic about them.

Proposition 2.6.9. *Let i be a subset inclusion from a game A to a game B. Then the strategy subs_i is an epimorphism from B to A.*

Proof. In fact, it is a split epimorphism: we can define a retract

$$\mathrm{ret}_i = \{s \in P_{A \multimap B} \ : \ \text{for all even-length } t \sqsubseteq s,\ t|_A = i_*(t|_B)\}\,.$$

The same argument as in Proposition 2.6.5 tells us that this is indeed a strategy for $A \multimap B$. Note that although subs_i is always a total strategy (i.e., if $s \in \mathrm{subs}_i$ and $sa \in P_{B \multimap A}$, then there is always $sab \in \mathrm{subs}_i$ for some b), the same is not in general true about ret_i.

In any case, if $\mathfrak{s} \in \mathrm{ret}_i \,\|\, \mathrm{subs}_i$, then $\mathfrak{s}|_{A^L} = i_*(\mathfrak{s}|_B) = \mathfrak{s}|_{A^R}$, and the same is true of any even-length substring of $\mathfrak{s}$, and so $\mathfrak{s}|_{A,A} \in \mathrm{id}_A$. Conversely, given any $s \in \mathrm{id}_A$, we can form some $\mathfrak{s} \in \mathrm{ret}_i \,\|\, \mathrm{subs}_i$ such that $\mathfrak{s}|_{A,A} = s$ as in Proposition 2.6.5.

We can also prove that subs_i is an epimorphism directly, which might be useful, for example, in a setting in which non-total strategies such as ret_i are disallowed. In this setting, let $\sigma, \tau : A \multimap C$ be strategies such that $\mathrm{subs}_i ; \sigma = \mathrm{subs}_i ; \tau$. Then, by Proposition 2.6.5, we know that

$$\{[i, \mathrm{id}_{M_C}]_*(s) \ : \ s \in \sigma\} = \{[i, \mathrm{id}_{M_C}]_*(s) \ : \ s \in \tau\}\,.$$

Then, since the function $[i, \mathrm{id}_{M_D}]_* : P_{A \multimap C} \to P_{B \multimap C}$ is an injection, we deduce that $\sigma = \tau$. □

2.7 $\mathcal{G}$ as a Symmetric Monoidal Category

We now claim that the tensor product connective $\otimes$ makes $\mathcal{G}$ into a symmetric monoidal closed category, with internal hom given by $\multimap$. We shall prove this result over the next two sections.

Definition 2.7.1. Let $\sigma : A \multimap B$ and $\tau : C \multimap D$ be strategies. We define a strategy $\sigma \otimes \tau : (A \otimes C) \multimap (B \otimes D)$ by

$$\sigma \otimes \tau = \{s \in P_{(A \otimes C) \multimap (B \otimes D)} \ : \ s|_{A,B} \in \sigma \text{ and } s|_{C,D} \in \tau\}\,.$$

To prove that this is a strategy, we prove a lemma analogous to our Lemma 2.4.3.

Lemma 2.7.2. *Let $s \in P_{A\otimes B}$. Then $\lambda^{OP}_{A\otimes B}(s) = \lambda^{OP}_A(s|_A) \wedge \lambda^{OP}_B(s|_B)$, where $\wedge$ is the binary operator on $\{O, P\}$ given by*

p	q	$p \wedge q$
P	P	P
O	P	O
P	O	O
O	O	O

.

Moreover, either $\lambda^{OP}_A(s|_A) = P$ or $\lambda^{OP}_B(s|_B) = P$; i.e., the bottom row of the table above does not occur.

Proof. Mutual induction on the length of s. This is obvious if s is empty. Suppose that $sa \in P_{A\otimes B}$, where a is an O-move. By induction, since $\lambda_{A\otimes B}(s) = P$, we must have $\lambda_{A\otimes B}(s|_A) = P$ and $\lambda_{A\otimes B}(s|_B) = P$. Therefore, depending on which game a is played in, either $\lambda_A(sa|_A) = O$ and $\lambda_B(sa|_B) = P$ or $\lambda_A(sa|_A) = P$ and $\lambda_B(sa|_B) = O$.

If $sb \in P_{A\otimes B}$, where b is a P-move, then by induction either $\lambda_A(s|_A) = O$ and $\lambda_B(s|_B) = P$ or $\lambda_A(s|_A) = P$ and $\lambda_B(s|_B) = O$. In either case, player P must play in whichever game is currently in an O-position, returning us to configuration PP. □

The above proof gives us the following result, which is analogous to Corollary 2.4.4.

Corollary 2.7.3 (Switching condition for $\otimes$)**.** *Player O switches games in $A \otimes B$; i.e., if $sab \in P_{A\otimes B}$, where a and b take place in different games (i.e., a in A and b in B or a in B and b in A), then b is an O-move.*

Proposition 2.7.4. *$\sigma \otimes \tau$ is a strategy for $(A \otimes C) \multimap (B \otimes D)$.*

Proof. $\sigma \otimes \tau$ is certainly an even-prefix-closed subset of $P^{even}_{(A\otimes C)\multimap(B\otimes D)}$.

Let s be a play of $P_{(A\otimes B)\multimap(C\otimes D)}$. We consider the possible configurations of s; i.e., the tuples $(\lambda_A(s|_A), \lambda_B(s|_B), \lambda_C(s|_C), \lambda_D(s|_D))$.

By Lemma 2.4.3 we must avoid the overall configuration OP for the linear implication, and by Lemma 2.7.2 we must avoid the configuration OO inside

either tensor product, so we end up with the following possibilities.

$\lambda_A(s\|_A)$	$\lambda_C(s\|_C)$	$\lambda_B(s\|_B)$	$\lambda_D(s\|_D)$	$\lambda_{A\otimes C}(s\|_{A,C})$	$\lambda_{B\otimes D}(s\|_{B,C})$	$\lambda_{(A\otimes C)\multimap(B\otimes D)}(s)$
P	P	P	P	P	P	P
P	P	P	O	P	O	O
P	P	O	P	P	O	O
P	O	P	O	O	O	P
O	P	O	P	O	O	P
P	O	O	P	O	O	P
O	P	P	O	O	O	P

If $s \in \sigma \otimes \tau$, or an odd-length sequence formed by adding an O-move to the end of a sequence in $\sigma \otimes \tau$, then we also know that $s|_{A,B} \in \sigma \subseteq P_{A\multimap B}$ and that $s|_{C,D} \in \tau \subseteq P_{C\multimap D}$. This means that we can discount the last two configurations in the table above, since one contains the illegal configuration OP in $C \multimap D$ and the other contains the illegal configuration OP in $A \multimap B$.

Now suppose that $sab, sac \in \sigma \otimes \tau$. Then sa is an O-position in $(A \otimes C) \multimap (B \otimes D)$, and is therefore in configuration $PPPO$ or $PPOP$. By inspecting the table above, we see that if sa is in configuration $PPPO$, then b and c must both occur either in C or in D, and that if sa is in configuration $PPOP$, then b and c must both occur either in A or in B. In either case, we must have $b = c$, by determinism of τ (in the first case) or of σ (in the second case). □

We need a lemma to prove that the tensor product of two innocent strategies is innocent.

Lemma 2.7.5. *Let $s \in \sigma \otimes \tau$.*

i) If s ends with a move in A or B, then $\ulcorner s \urcorner^{(A\otimes C)\multimap(B\otimes D)} = \ulcorner s|_{A,B} \urcorner^{A\multimap B}$.

ii) If s ends with a move in C or D, then $\ulcorner s \urcorner^{(A\otimes C)\multimap(B\otimes D)} = \ulcorner s|_{C,D} \urcorner^{C\multimap D}$.

Proof. Induction on the length of s. We prove (i); (ii) is exactly the same.

If a is a P-move, then we have $\ulcorner sa \urcorner = \ulcorner s \urcorner a$. By our analysis in the proof of Proposition 2.7.4, player P only switches moves between A and B, and between C and D, so s must end with a move from A or B. Therefore, by the inductive hypothesis, $\ulcorner s \urcorner = \ulcorner s|_{A,B} \urcorner$. Then $\ulcorner sa \urcorner = \ulcorner s \urcorner a = \ulcorner s|_{A,B} \urcorner a = \ulcorner sa|_{A,B} \urcorner$.

If a is an initial move, then $\ulcorner sa \urcorner = a = \ulcorner sa|_{A,B} \urcorner$.

If a is an O-move justified by b in $sbta$, then $\ulcorner sbta \urcorner = \ulcorner s \urcorner ba$. Then b is a P-move, so s must end with a move in A or B, as before. Therefore, by the

inductive hypothesis, $\ulcorner s \urcorner = \ulcorner s|_{A,B} \urcorner$. Then $\ulcorner sbta \urcorner = \ulcorner s \urcorner ba = \ulcorner s|_{A,B} \urcorner ba = \ulcorner sbta|_{A,B} \urcorner$. □

Proposition 2.7.6. *Let $\sigma : A \to B$, $\tau : C \to D$ be innocent strategies. Then $\sigma \otimes \tau$ is innocent.*

Proof. Suppose $sab, t \in \sigma \otimes \tau$ such that $ta \in P_{(A \otimes C) \multimap (B \otimes D)}$ and $\ulcorner sa \urcorner = \ulcorner ta \urcorner$. Suppose without loss of generality that a is a move in A or B. Then $\ulcorner sa \urcorner = \ulcorner sa|_{A,B} \urcorner$ and $\ulcorner ta \urcorner = \ulcorner ta|_{A,B} \urcorner$ by Lemma 2.7.5, and therefore $tab|_{A,B} \in \sigma$ by innocence of σ, and so $tab \in \sigma \otimes \tau$. □

The most important thing we need to prove is that $\otimes$ is a functor.

Proposition 2.7.7. *Let $\sigma' : A'' \multimap A'$, $\sigma : A' \multimap A$, $\tau' : B'' \multimap B'$ and $\tau : B' \multimap B$ be strategies. Then $(\sigma' \otimes \tau'); (\sigma \otimes \tau) = (\sigma'; \sigma) \otimes (\tau'; \tau)$.*

Moreover, if A', A, B', B are games, i is a subset inclusion from A to A' and j is a structural isomorphism from B to B', then $\mathrm{subs}_i \otimes \mathrm{subs}_j = \mathrm{subs}_{[i,j]}$. In particular, if A and B are games, then $\mathrm{id}_A \otimes \mathrm{id}_B = \mathrm{id}_{A \otimes B}$.

Proof. First suppose that $s \in (\sigma' \otimes \tau'); (\sigma \otimes \tau)$; so $s = \mathfrak{s}|_{A'',B'',A,B}$, where $\mathfrak{s} \in (\sigma' \otimes \tau') \| (\sigma \otimes \tau)$. Then $\mathfrak{s}|_{A'',A'} \in \sigma'$ and $\mathfrak{s}|_{A',A} \in \sigma$, so $\mathfrak{s}|_{A'',A',A} \in \sigma' \| \sigma$ and therefore $s|_{A'',A} = \mathfrak{s}|_{A'',A} \in \sigma'; \sigma$. Similarly, $s|_{B'',B} \in \tau'; \tau$, and therefore $s \in (\sigma'; \sigma) \otimes (\tau'; \tau)$.

Conversely, suppose that $s \in (\sigma'; \sigma) \otimes (\tau'; \tau)$. Choose some $\mathfrak{s} \in \sigma' \| \sigma$, $\mathfrak{t} \in \tau' \| \tau$ such that $s|_{A'',A} = \mathfrak{s}|_{A'',A}$ and $s|_{B'',B} = \mathfrak{s}|_{B'',B}$. By our analysis, the only time we switch from the A'', A-component to the B'', B component in s, or *vice versa*, is when player O switches between the games A and B. Thus, we may divide s up into blocks, each starting and ending with a move in the outer component $A \otimes B$. This then gives us a way to divide up $\mathfrak{s}$ and $\mathfrak{t}$ into blocks, such that each block of $\mathfrak{s}$ or $\mathfrak{t}$ projects on to a block of s. Lastly, we can string these blocks together to give us some $\mathfrak{u} \in (\sigma' \otimes \tau') \| (\sigma \otimes \tau)$ such that $\mathfrak{u}|_{A'',B'',A,B} = s$.

For the second part, let A', A, B', B be games, let i be a structural isomorphism from A to A' and let j be a structural isomorphism from B to B'. Suppose that $s \in \mathrm{subs}_i \otimes \mathrm{subs}_j$. Then $s|_{A',A} \in \mathrm{subs}_i$ and $s|_{B',B} \in \mathrm{subs}_j$ – so if $u \sqsubseteq s|_{A,A}$ has even length, then $u|_{A'} = i_*(u|_A)$, and if $v \sqsubseteq s|_{B,B}$ has even length, then $v|_{B'} = i_*(v|_B)$. Suppose that $t \sqsubseteq s$ is of even length. Then, since only player O switches between the A', A-component and the B', B-component, both $t|_{A',A}$ and $t|_{B',B}$ are of even length, it follows that $t|_{A',B'} = [i,j]_*(t|_{A,B})$. Since t was arbitrary, this means that $s \in \mathrm{subs}_{[i,j]}$.

Conversely, suppose that $s \in \mathrm{subs}_{[i,j]}$. Then for all even-length $t \sqsubseteq s$, $t|_{A'} = i_*(t|_A)$ and $t|_{B'} = j_*(t|_B)$. Since any play in σ or in τ is itself a

play of $\sigma \otimes \tau$, then if $u \sqsubseteq s|_{A',A}$ has even length, then $u|_{A'} = i_*(u|_A)$, and if $v \sqsubseteq s|_{B',B}$, then $v|_{B'} = j_*(v|_B)$. It follows that $s|_{A',A} \in \mathrm{subs}_i$ and $s|_{B',B} \in \mathrm{subs}_j$, and therefore that $s \in \mathrm{subs}_i \otimes \mathrm{subs}_j$. □

Now it is fairly clear that if A, B, C are games, then we have structural isomorphisms

$$(A \otimes B) \otimes C \cong A \otimes (B \otimes C)$$

$$A \cong A \otimes I \qquad\qquad A \cong I \otimes A$$

$$A \otimes B \cong B \otimes A\,,$$

induced by the associators, unitors and symmetry of the category of sets with coproduct. We claim that these are natural transformations.

Proposition 2.7.8. *The families of morphisms*

$$\mathrm{cc}_{\mathrm{assoc}_{M_A,M_B,M_C}} : (A \otimes B) \otimes C \to A \otimes (B \otimes C)$$

$$\mathrm{cc}_{\mathrm{lunit}_{M_A}} : A \to I \otimes A \qquad\qquad \mathrm{cc}_{\mathrm{runit}_{M_A}} : A \to A \otimes I$$

$$\mathrm{cc}_{\mathrm{sym}\, M_A,M_B} : A \otimes B \to B \otimes A$$

are natural transformations in $\mathcal{G}$.

Proof. We prove this for the associator; the other cases are similar.

Let $\sigma : A' \multimap A$, $\tau : B' \multimap B$, $\upsilon : C' \multimap C$ be strategies. By Proposition 2.6.5, we have

$$\begin{aligned}
&((\sigma \otimes \tau) \otimes \upsilon); \mathrm{cc}_{\mathrm{assoc}_{M_A,M_B,M_C}} \\
&= \{[\mathrm{id}_{M_{(A'\otimes B')\otimes C'}}, \mathrm{assoc}_{M_A,M_B,M_C}]_*(s) : s \in (\sigma \otimes \tau) \otimes \upsilon\} \\
&= \left\{ [\mathrm{id}_{M_{(A'\otimes B')\otimes C'}}, \mathrm{assoc}_{M_A,M_B,M_C}]_*(s) \;\middle|\; \begin{array}{l} s \in P_{((A'\otimes B')\otimes C') \multimap ((A\otimes B)\otimes C)} \\ s|_{A',A} \in \sigma,\ s|_{B',B} \in \tau,\ s|_{C',C} \in \upsilon \end{array} \right\} \\
&= \{s \in P_{((A'\otimes B')\otimes C') \multimap (A\otimes(B\otimes C))} : s|_{A',A} \in \sigma, s|_{B',B} \in \tau, s|_{C',C} \in \upsilon\} \\
&= \left\{ [\mathrm{assoc}_{M_{A'},M_{B'},M_{C'}}, \mathrm{id}_{M_{A\otimes(B\otimes C)}}]_*(s) \;\middle|\; \begin{array}{l} s \in P_{(A'\otimes(B'\otimes C')) \multimap (A\otimes(B\otimes C))} \\ s|_{A',A} \in \sigma,\ s|_{B',B} \in \tau,\ s|_{C',C} \in \upsilon \end{array} \right\} \\
&= \{[\mathrm{assoc}_{M_{A'},M_{B'},M_{C'}}, \mathrm{id}_{M_{A\otimes(B\otimes C)}}]_*(s) : s \in \sigma \otimes (\tau \otimes \upsilon)\} \\
&= \mathrm{cc}_{\mathrm{assoc}_{M_{A'},M_{B'},M_{C'}}}; (\sigma \otimes (\tau \otimes \upsilon))\,.
\end{aligned}$$

□

This proves that the coherences we have defined are natural transformations. By Proposition 2.6.5 again, these natural transformations satisfy the same coherence diagrams (pentagon, triangles, hexagon etc.) satisfied by the original associators, unitors and symmetry in $(\mathbf{Set}, +)$.

It follows that $\otimes$ makes $\mathcal{G}$ into a symmetric monoidal category.

2.8 $\mathcal{G}$ as a Symmetric Monoidal Closed Category

Definition 2.8.1. Let A, B, C, D be games, let σ be a strategy for $A \multimap B$ and let τ be a strategy for $C \multimap D$. Then we define a strategy

$$\sigma \multimap \tau \colon (B \multimap C) \multimap (A \multimap D)$$

by

$$\sigma \multimap \tau = \{s \in P_{(B \multimap C) \multimap (A \multimap D)} \ : \ s|_{A,B} \in \sigma,\, s|_{C,D} \in \tau\}\,.$$

Proposition 2.8.2. *$\sigma \multimap \tau$ is a strategy for $(B \multimap C) \multimap (A \multimap D)$.*

Proof. $\sigma \multimap \tau$ is certainly a prefix-closed subset of $P^{even}_{(B \multimap C) \multimap (A \multimap D)}$.

We examine the sign configuration of a play in $(B \multimap C) \multimap (A \multimap D)$, using Lemma 2.4.3. Since we must avoid the configuration OP in either $B \multimap C$, $A \multimap D$ or in $(B \multimap C) \multimap (A \multimap D)$, we arrive at the following list of possibilities.

$\lambda^{OP}_B(s\|_B)$	$\lambda^{OP}_C(s\|_C)$	$\lambda^{OP}_A(s\|_A)$	$\lambda^{OP}_D(s\|_D)$	$\lambda^{OP}_{B \multimap C}(s\|_{B,C})$	$\lambda^{OP}_{A \multimap D}(s\|_{A,D})$	$\lambda^{OP}_{(B \multimap C) \multimap (A \multimap D)}(s)$
P	O	P	O	O	O	P
P	P	P	O	P	O	O
O	O	P	O	P	O	O
P	P	P	P	P	P	P
O	O	O	O	P	P	P
P	P	O	O	P	P	P
O	O	P	P	P	P	P

If $s \in \sigma \multimap \tau$, then we can immediately discount the last two of these possibilities, since one includes the illegal configuration OP in $A \multimap B$, and the other includes the illegal configuration OP in $B \multimap D$.

By examining the remaining possibilities, we arrive at the conclusion that any O-position in configuration $PPPO$ constrains player P to play in C (to reach configuration $POPO$) or to play in D (to reach configuration $PPPP$), and that any O-position in configuration $OOPO$ constrains player P to play in A (to reach configuration $OOOO$) or to play in C (to reach configuration $POPO$).

Now suppose that $sab, sac \in \sigma \multimap \tau$. Then, by our above analysis, b and c must either both take place in the B, A-component, in which case $b = c$ by determinism of σ, or both in the C, D-component, in which case $b = c$ by determinism of τ. □

To prove that $\sigma \multimap \tau$ is innocent if σ and τ are, we need a lemma analogous to Lemma 2.7.5.

Lemma 2.8.3. *Let $s \in \sigma \multimap \tau$.*

i) If s ends with a move in A or B, then $\ulcorner s \urcorner^{(B \multimap C) \multimap (A \multimap D)} = \ulcorner s|_{A,B} \urcorner^{A \multimap B}$.

ii) If s ends with a move in C or D, then $\ulcorner s \urcorner^{(B \multimap C) \multimap (A \multimap D)} = \ulcorner s|_{C,D} \urcorner^{C \multimap D}$.

Proof. Exactly the same as in Lemma 2.7.5, using the analysis from the proof of Proposition 2.8.2 to show that player P only switches moves between A and B, and between C and D, in $\sigma \multimap \tau$. □

Proposition 2.8.4. *Let $\sigma \colon A \multimap B$, $\tau \colon C \multimap D$ be innocent strategies. Then $\sigma \multimap \tau$ is innocent.*

Proof. Suppose $sab, t \in \sigma \multimap \tau$ such that $ta \in P_{(B \multimap C) \multimap (A \multimap D)}$ and $\ulcorner sa \urcorner = \ulcorner ta \urcorner$. Suppose without loss of generality that a is a move in A or B. Then $\ulcorner sa \urcorner = \ulcorner sa|_{A,B} \urcorner$ and $\ulcorner ta \urcorner = \ulcorner ta|_{A,B} \urcorner$ by Lemma 2.8.3, and therefore $tab|_{A,B} \in \sigma$ by innocence of σ, and so $tab \in \sigma \multimap \tau$. □

We now need to prove that $\multimap$ is a functor $\mathcal{G}^{\mathrm{op}} \times \mathcal{G} \to \mathcal{G}$.

Proposition 2.8.5. *Let $\sigma' \colon A'' \multimap A'$, $\sigma \colon A' \multimap A$, $\tau' \colon B'' \multimap B'$ and $\tau \colon B' \multimap B$ be strategies. Then $(\sigma \multimap \tau'); (\sigma' \multimap \tau) = (\sigma'; \sigma) \multimap (\tau'; \tau)$.*

Moreover, if A', A, B', B are games, f is a structural isomorphism from A' to A and g is a structural isomorphism from B' to B, then $\mathrm{cc}_f \multimap \mathrm{cc}_g = \mathrm{cc}_{[f^{-1}, g]}$. In particular, if A, B are games than $\mathrm{id}_A \multimap \mathrm{id}_B = \mathrm{id}_{A \multimap B}$.

Proof. As in Proposition 2.7.7. □

Now it is easy to see that the associator $\mathrm{assoc}_{M_A, M_B, M_C}$ is a structural isomorphism from $(A \otimes B) \multimap C$ to $A \multimap (B \multimap C)$, so it induces a copycat isomorphism $\Lambda_{A,B,C} = \mathrm{cc}_{\mathrm{assoc}_{M_A, M_B, M_C}} : (A \otimes B) \multimap C \to A \multimap (B \multimap C)$.

Proposition 2.8.6. *$\Lambda_{A,B,C}$ is natural in A, B, C.*

Proof. The same argument as in Proposition 2.7.8. □

We have proved the following.

Theorem 2.8.7. *$\mathcal{G}$ is a symmetric monoidal closed category, with tensor product given by $\otimes$ and internal hom given by $\multimap$.*

2.9 Tree Immersions and Zigzag Strategies

So far, the strategies we have been considering have all been innocent. For our model of state, we are going to need to start using some non-innocent strategies as well. We begin with a generalization of the idea of a subset inclusion to that of a *tree immersion*, which gives us a similar recipe for constructing strategies. Tree immersions are similar to subset inclusions, but generated by a function between plays, rather than between moves. A consequence of this is that while tree immersions do give rise to strategies, these strategies are not in general innocent.

Definition 2.9.1. Let A, B be games. A *tree immersion* from A to B is a function $\phi \colon P_A \hookrightarrow P_B$ such that

- ϕ preserves length and justification indices;
- for all sequences $s, t \in P_A$, if $t \sqsubseteq s$ then $\phi(t) \sqsubseteq \phi(s)$; and
- if $\phi(sb) = \phi(sc)$, where b, c are P-moves in A, then $b = c$.

Given a tree immersion ϕ from A to B, we define a strategy $\mathrm{zz}_\phi \colon B \multimap A$ by

$$\mathrm{zz}_\phi = \{s \in P_{B \multimap A} : \text{ for all even-length } t \sqsubseteq s,\ t|_B = \phi(t|_A)\}\,.$$

We shall refer to strategies of the form zz_ϕ as *zigzag strategies*.

Example 2.9.2. If i is a subset inclusion from A to B, then i_* is a tree immersion from A to B and $\mathrm{zz}_{i_*} = \mathrm{subs}_i$.

Example 2.9.3. If A, B, C are games, then we have a natural morphism

$$(A \oslash C) \times (B \oslash C) \multimap (A \times B) \oslash C\,,$$

given by a tree immersion from the set of plays of $(A \times B) \oslash C$ to the set of plays of $(A \oslash C) \times (B \oslash C)$. Note that this is not a subset inclusion: if s is a play of $(A \times B) \oslash C$ ending in a move in C, then the copy of C that that move ends up in is determined by whether s begins with a move in A or in B.

Proposition 2.9.4. *zz_ϕ is a strategy.*

Proof. zz_ϕ is a prefix-closed subset of $P_{B \multimap A}$ by definition. If $sab, sac \in \mathrm{zz}_\phi$, then we have $s|_B = \phi(s|_A)$ and $sab|_B = \phi(sab|_A)$. Since ϕ is length-preserving, $s|_A$ and $s|_B$ must have the same length, and the same is true of $sab|_A$ and $sab|_B$. Therefore, either a is a move in A and $s|_B b = \phi(s|_A a)$ or a is a move in B and $s|_B a = \phi(s|_A b)$. The same applies to c: so either $s|_B b = \phi(s|_A a) = s|_B c$ or $\phi(s|_A b) = s|_B a = \phi(s|_A c)$. In either case, we have $b = c$. □

We want an analogue of Proposition 2.6.5.

Definition 2.9.5. Given a tree immersion from A to B, and a play $s \in P_{C \multimap A}$ for some C, we write s^ϕ for the play obtained by replacing the moves of $s|_A$ wholesale with the moves of $\phi(s|_A)$. Since ϕ preserves length and justification indices, the resulting play is a legal play of $C \multimap A$.

Proposition 2.9.6. *If $\sigma : C \multimap B$ is a strategy, then $\sigma ; \mathrm{zz}_\phi$ is given by*

$$\{s \in P_{C \multimap A} : \ s^\phi \in \sigma\}.$$

In particular, if ϕ is a tree immersion from A to B and ψ is a tree immersion from B to C, then $\psi \circ \phi$ is a tree immersion from A to C and $\mathrm{zz}_{\psi\circ\phi} = \mathrm{zz}_\psi ; \mathrm{zz}_\phi$.

Proof. Suppose that $\mathfrak{s} \in \sigma \| \mathrm{zz}_\phi$. Then $\mathfrak{s}|_{C,B} \in \sigma$ and $t|_B = \phi(t|_A)$ for all even-length $t \sqsubseteq \mathfrak{s}|_{B,A}$. Then it is clear that $(\mathfrak{s}|_{C,A})^\phi = \mathfrak{s}|_{C,B}$.

Conversely, suppose that $s \in P_{C \multimap A}$ and that $s^\phi \in \sigma$. We construct a sequence $\mathfrak{s} \in \sigma \| \mathrm{zz}_\phi$ such that $\mathfrak{s}|_{C,B} = s^\phi$ and $\mathfrak{s}|_{C,A} = s$ by taking the sequence s and inserting, in order, the elements of the sequence $\phi(s|_A)$ immediately after each O-move in $s|_C$ and immediately before each P-move in C, leaving the rest of s intact. Then $\mathfrak{s}|_{C,B} = s^\phi \in \sigma$ and $\mathfrak{s}|_{B,A} \in \mathrm{zz}_\phi$, by construction. So $s = \mathfrak{s}|_{C,A} \in \sigma ; \mathrm{zz}_\phi$. □

Definition 2.9.7. We say that a tree immersion ϕ is a *tree isomorphism* if it is a bijection.

Proposition 2.9.8. *If ϕ is a tree isomorphism from a game A to a game B, then zz_ϕ is an isomorphism in $\mathcal{G}$.*

Proof. If ϕ is a tree isomorphism, then its inverse ϕ^{-1} is also a tree isomorphism, and Proposition 2.9.6 tells us that zz_ϕ and $\mathrm{zz}_{\phi^{-1}}$ are inverses in $\mathcal{G}$. □

More generally:

Proposition 2.9.9. *If ϕ is a surjection, then zz_ϕ is a monomorphism.*

Proof. Let $\sigma, \tau \colon C \multimap B$ be strategies. Then, by Proposition 2.9.6, we have

$$\{s \in P_{C \multimap A} : s^\phi \in \sigma\} = \{s \in P_{C \multimap A} : s^\phi \in \tau\}.$$

Let $t \in \sigma$. Then, since ϕ is surjective, there is some $u \in P_A$ such that $\phi(u) = t|_B$. As before, we may construct some sequence t' such that $t'|_A = u$ and $t = (t')^\phi$. Then, since $(t')^\phi = t \in \sigma$, we must have $(t')^\phi \in \tau$; i.e., that $t \in \tau$. So $\sigma \subseteq \tau$.

Similarly, $\tau \subseteq \sigma$, and so σ and τ are equal. □

Applying this to subset inclusions using Proposition 2.6.9, we see that a surjective subset inclusion is both a monomorphism and a split epimorphism, and hence an isomorphism.

2.10 Products in $\mathcal{G}$

Proposition 2.10.1. *Given some family A_i of games, the game $\prod_i A_i$, as defined in Definition 2.3.1, is the category-theoretic product of the A_i.*

Proof. We have natural injections $\mathrm{in}_j \colon M_{A_j} \hookrightarrow M_{\prod_i A_i}$ giving rise to subset inclusions. Then our projections are given by the morphisms

$$\mathrm{pr}_j \coloneqq \mathrm{subs}_{\mathrm{in}_j} \colon \prod_i A_i \to A_j .$$

Now suppose we have some game B, and strategies $\sigma_i \colon B \multimap A_i$ for each i. Define a strategy

$$\langle \sigma_i \rangle = \bigcup_i [\mathrm{id}_{M_B}, \mathrm{in}_i]_*(\sigma_i) .$$

We claim that this is indeed a strategy for $B \multimap \prod_i A_i$. Indeed, it is certainly a prefix-closed subset of $P_{C \multimap \prod_i A_i}$.

Moreover, if $sab, sac \in \langle \sigma_i \rangle$, then there is some unique j such that a comes from a move in A_j, and therefore sab, sac are both plays in σ_j, so $b = c$.

Next, we claim that $\langle \sigma_i \rangle ; \mathrm{pr}_j = \sigma_j$. Indeed, we have

$$\begin{aligned} \langle \sigma_i \rangle ; \mathrm{pr}_j &= \langle \sigma_i \rangle ; \mathrm{subs}_{\mathrm{in}_j} \\ &= \{[\mathrm{id}_{M_B}, \mathrm{in}_j^{-1}]_*(s) : s \in \langle \sigma_i \rangle,\ s|_{\prod_i Ai} \in (\mathrm{in}_j)_*(P_{A_j})\} \ \textit{Prop. 2.6.5} \\ &= \sigma_j . \end{aligned}$$

Lastly, suppose $\tau \colon B \multimap \prod_i A_i$ is a strategy such that $\tau ; \mathrm{pr}_j = \sigma_j$ for each j. We claim that $\tau = \langle \sigma_i \rangle$. Indeed, by the argument above, we must have

$$\{[\mathrm{id}_{M_B}, \mathrm{in}_j^{-1}]_*(s) : s \in \tau,\ s|_{\prod_i A_i} \in (\mathrm{in}_j)_*(P_{A_j})\} = \sigma_j$$

for each j. Suppose that $s \in \tau$. Then $s|_{\prod_i A_i} \in (\mathrm{in}_j)_*(P_{A_j})$ for some j, by the definition of $\prod_i A_i$. Therefore, $s \in [\mathrm{id}_{M_B}, \mathrm{in}_j]_*(\sigma_j)$.

Conversely, let $t \in \sigma_j$. By the above equation, we know that there is some $s \in \tau$ such that $s|_{\prod_i A_i} \in (\mathrm{in}_j)_*(P_{A_j})$ and $[\mathrm{id}_{M_B}, \mathrm{in}_j^{-1}]_*(s) = t$. It follows that $[\mathrm{id}_{M_B}, \mathrm{in}_j]_*(t) = s \in \tau$. □

An examination of the definitions tells us that

Proposition 2.10.2. *Let A_i, B be games and let ϕ_i be tree immersions from A_i to B. Then $\langle \mathrm{zz}_{\phi_i} \rangle = \mathrm{zz}_\phi$, where ϕ is the tree immersion from $\prod_i A_i$ to B given by*

$$\phi(s) = \begin{cases} \epsilon & \text{if } s = \epsilon \\ \phi_i(s|_{A_i}) & \text{if } s \text{ starts with a move from } A_i \end{cases}$$

Proof. The only thing we really need to check is that this is indeed a tree immersion. Let sb, sc be positions in $\prod_i A_i$, where b, c are P-moves. Then sb, sc must start with the same move, so if $\phi(sb) = \phi(sc)$ then we have $\phi_i(sb) = \phi_i(sc)$ for some i and therefore $b = c$. □

Note that $\langle \sigma_i \rangle$ is not in general innocent, even if all the σ_i are, and Proposition 2.10.2 cannot be adapted to use subset inclusions rather than tree immersions. Of course, since a subset inclusion is a special case of a tree immersion, then $\langle \mathrm{subs}_i : i \in \mathcal{I} \rangle$ is always a tree immersion strategy for any set $\mathcal{I}$ of subset inclusions.

2.11 Sequoidal categories

We have given the category-theoretic properties of all the connectives from section 2.3, with the exception of the sequoid $\oslash$ and the exponential !. In this section, we will introduce the categorical semantics of the sequoid operator $\_ \oslash \_$. The sequoid is a non-standard operator and does not fall into any of the established patterns of categorical algebra; rather, it gives rise to the new notion of a *sequoidal category* introduced by Laird in [Lai02].

To start with, we would like to be able to say that $\_ \oslash \_$ is a functor from $\mathcal{G} \times \mathcal{G} \to \mathcal{G}$, as is the case with the tensor product $\_ \otimes \_$. However, this does not quite work: given strategies $\sigma \colon A \multimap B$ and $\tau \colon C \multimap D$, we would *like* to say that we get a strategy $\sigma \oslash \tau \colon A \oslash C \multimap B \oslash D$. However, this does not quite work as we might expect: suppose player O begins (as he must) with some move b in B, and suppose that σ's reply to b is another move b' in B. Suppose then that player O switches and plays a move d in D, and

suppose that τ's reply to d is a move c in C. Then player P is unable to play according to τ, because d is not yet a legal move in $(A \oslash C) \multimap (B \oslash D)$ since no move has yet been played in A.

We can fix this problem by imposing some constraints on the strategies σ and τ. The problem occurs when player O's initial move in B is not reflected by an initial move by player P in A; therefore, if σ is such that player P always responds to the initial move in B with a move in A, then we can form a strategy $\sigma \oslash \tau$ for $(A \oslash C) \multimap (B \oslash D)$. Moreover, this strategy $\sigma \oslash \tau$ will inherit this property that the first move on the right is always replied to by a move on the left.

Definition 2.11.1. Let A, B be games. A *strict morphism* from A to B is a strategy σ for $A \multimap B$ such that any player P response to an opening move in B is a move in A; i.e., such that if b is an initial O-move in B and $ba \in \sigma$, then a is a move in A.

We will call such a σ a *strict strategy* for $A \multimap B$, although this is a slight abuse of language, since the definition depends on the constituent games A and B, which may not be recoverable from $A \multimap B$.

It is clear that the composition of strict morphisms is again a strict morphism, and that any morphism of the form subs_i is a strict morphism, and so we get a wide subcategory $\mathcal{G}_s$ of $\mathcal{G}$ whose objects are games and where the morphisms are the strict strategies. We then have a natural inclusion functor $J \colon \mathcal{G}_s \to \mathcal{G}$.

Definition 2.11.2. Given games A, B, C, D, a strict morphism $\sigma \colon A \multimap B$ and a strategy $\tau \colon C \multimap D$, we define a strict morphism $\sigma \oslash \tau \colon (A \oslash C) \multimap (B \oslash D)$ by

$$\sigma \oslash \tau = \{ s \in P_{(A \oslash C) \multimap (B \oslash D)} \ : \ s|_{A,B} \in \sigma,\ s|_{C,D} \in \tau \}\,.$$

Proposition 2.11.3. *$\sigma \oslash \tau$ is a strategy.*

Proof. $\sigma \oslash \tau$ is certainly a prefix-closed subset of $P_{(A \oslash C) \multimap (B \oslash D)}$. Moreover, if $sab, sac \in \sigma \oslash \tau$, then $sab, sac \in \sigma \otimes \tau$, so $b = c$. □

Of course, $P_{A \oslash B}$ is a subset of $P_{A \otimes B}$, which means that the identity function $M_A + M_B \to M_A + M_B$ gives us a subset inclusion from $A \oslash B$ to $A \otimes B$, and hence a strategy $\mathrm{subs}_{\mathrm{id}_{M_A + M_B}}$ for $A \otimes B \multimap A \oslash B$, which we shall refer to as $\mathrm{wk}_{A,B}$.

Proposition 2.11.4. *Let A, B, C, D be games, let $\sigma\colon A \multimap B$ be a strict strategy and let $\tau\colon C \multimap D$ be a strategy. Then the following diagram commutes.*

$$\begin{array}{ccc} A \otimes C & \xrightarrow{\sigma\otimes\tau} & B \otimes D \\ {\scriptstyle \mathrm{wk}_{A,C}}\downarrow & & \downarrow{\scriptstyle \mathrm{wk}_{B,D}} \\ A \oslash C & \xrightarrow{\sigma\oslash\tau} & B \oslash D \end{array}$$

Proof. By Proposition 2.6.5 and the definition of wk, we know that

$$\begin{aligned} \sigma \otimes \tau; \mathrm{wk}_{B,D} &= \{s \in \sigma \otimes \tau,\, s|_{B,D} \in P_{B\oslash D}\} \\ \mathrm{wk}_{A,C}; \sigma \oslash \tau &= \sigma \oslash \tau\,, \end{aligned}$$

as sets of plays.

Now we know that $\sigma \oslash \tau = \{s \in \sigma \otimes \tau :\ s \in P_{(A\oslash C)\multimap(B\oslash D)}\}$, so it suffices to show that if $s \in \sigma \otimes \tau$ is such that $s|_{B,D} \in P_{B\oslash D}$ then $s \in P_{(A\oslash C)\multimap(B\oslash D)}$.

Indeed, if $s|_{B,D} \in P_{B\oslash D}$ then s begins with an initial O-move in B. Then, *since σ is strict*, the next move in s must be a move in A, and therefore $s|_{A,C}$ begins with a move in A. Since we also have $s|_{A,C} \in P_{A\otimes C}$, we must have that $s|_{A,C} \in P_{A\oslash C}$. □

Remark 2.11.5. If σ is not a strict strategy, then the set

$$\{s \in \sigma \otimes \tau :\ s \in P_{(A\oslash C)\multimap(B\oslash D)}\}$$

is still a valid strategy for $P_{(A\oslash C)\multimap(B\oslash D)}$, but now the conclusion of Proposition 2.11.4 no longer holds.

Remark 2.11.6. Of course, we would *like* to restate Proposition 2.11.4 by saying that wk is a natural transformation, but it doesn't make sense to do so, because we don't yet know that $\_ \oslash \_$ is a functor.

Proposition 2.11.7. *If we have strict strategies $\sigma'\colon A'' \multimap A'$ and $\sigma\colon A' \multimap A$, and strategies $\tau'\colon B'' \multimap B'$ and $\tau\colon B' \multimap B$, then we have*

$$(\sigma' \oslash \tau'); (\sigma \oslash \tau) = (\sigma'; \sigma) \oslash (\tau'; \tau)\,.$$

If A', A, B', B are games, i is a subset inclusion from A into A' and j is a subset inclusion from B into B', then

$$\mathrm{subs}_i \oslash \mathrm{subs}_j = \mathrm{subs}_{[i,j]}\colon A' \oslash B' \multimap A \oslash B\,.$$

In particular, if A, B are games, then $\mathrm{id}_A \oslash \mathrm{id}_B = \mathrm{id}_{A\oslash B}$.

Proof. Let A'', A', A, B'', B', B and $\sigma', \sigma, \tau', \tau$ be as above.

We have

$$\begin{aligned}
\mathrm{wk}_{A'',B''}; (\sigma' \oslash \tau'); (\sigma \oslash \tau) &= (\sigma' \otimes \tau'); \mathrm{wk}_{A',B'}; (\sigma \oslash \tau) && \textit{Prop. 2.11.4} \\
&= (\sigma' \otimes \tau'); (\sigma \otimes \tau); \mathrm{wk}_{A,B} && \textit{Prop. 2.11.4} \\
&= ((\sigma'; \sigma) \otimes (\tau'; \tau)); \mathrm{wk}_{A,B} && \textit{Prop. 2.7.7} \\
&= \mathrm{wk}_{A'',B''}; ((\sigma'; \sigma) \oslash (\tau'; \tau)) \,. && \textit{Prop. 2.11.4}
\end{aligned}$$

By Proposition 2.6.9, $\mathrm{wk}_{A'',B''}$ is an epimorphism, and therefore we have that

$$(\sigma' \oslash \tau'); (\sigma \oslash \tau) = (\sigma'; \sigma) \oslash (\tau'; \tau) \,.$$

Now let A', A, B', B be games, let i be a subset inclusion from A into A' and let j be a subset inclusion from B into B'. Then, since subset inclusion strategies are automatically strict, we have

$$\begin{aligned}
\mathrm{wk}_{A',B'}; (\mathrm{subs}_i \oslash \mathrm{subs}_j) &= (\mathrm{subs}_i \otimes \mathrm{subs}_j); \mathrm{wk}_{A,B} && \textit{Prop. 2.11.4} \\
&= \mathrm{subs}_{[i,j]}; \mathrm{wk}_{A,B} && \textit{Prop. 2.7.7} \\
&= \mathrm{subs}_{[i,j]} && \textit{Prop. 2.6.5} \\
&= \mathrm{wk}_{A',B'}; \mathrm{subs}_{[i,j]} \,. && \textit{Prop. 2.6.5}
\end{aligned}$$

As before, we know from Proposition 2.6.9 that $\mathrm{wk}_{A',B'}$ is an epimorphism, and so

$$\mathrm{subs}_i \oslash \mathrm{subs}_j = \mathrm{subs}_{[i,j]} \,. \qquad \square$$

Proposition 2.11.7 tells us that $\_ \oslash \_$ is a functor $\mathcal{G}_s \times \mathcal{G} \to \mathcal{G}$. As before, write J for the inclusion functor $\mathcal{G}_s \hookrightarrow \mathcal{G}$. Then we can restate Proposition 2.11.4 in a more *natural* way.

Proposition 2.11.8. $\mathrm{wk}_{A,B}$ *is a natural transformation*

$$JA \otimes B \to J(A \oslash B) \,.$$

We have some additional structure on the $\otimes$ and $\oslash$ operators. By inspecting the definitions, we can see that if A, X, Y are games then the associator $\mathrm{assoc}_{M_A,M_X,M_Y}$ and unitor runit_{M_A} in $(\mathbf{Set}, +)$ give rise to structural isomorphisms

$$(A \oslash X) \oslash Y \cong A \oslash (X \otimes Y) \qquad\qquad A \cong A \oslash I \,.$$

Indeed, in the first case, both games are the game in which A, X and Y are played in parallel, but where the first move must take place in A. In the second case, we have $A \oslash I = A \otimes I$, because there are no moves in I anyway, and the copycat morphism induced from the right unitor in $(\mathbf{Set}, +)$ is the same strategy as the right unitor $A \xrightarrow{\cong} A \otimes I$.

We formalize the structure we have uncovered so far in the concept of a *sequoidal category*.

Definition 2.11.9 ([Lai02])**.** A *sequoidal category* $\mathcal{C}$ is given by

- a symmetric monoidal category $(\mathcal{C}, \otimes, I)$ (with coherences assoc, lunit, runit, sym);
- a (strong) right action of $\mathcal{C}$ on a category $\mathcal{C}_s$; i.e., a functor $\_ \oslash \_ : \mathcal{C}_s \times \mathcal{C} \to \mathcal{C}_s$ together with natural isomorphisms

$$\mathsf{passoc}_{a,x,y} \colon (a \oslash x) \oslash y \xrightarrow{\cong} a \oslash (x \otimes y) \qquad \mathsf{r}_a \colon a \xrightarrow{\cong} a \oslash I$$

that make the diagrams

$$\begin{array}{ccccc} ((a \oslash x) \oslash y) \oslash z & \xrightarrow{\mathsf{passoc}_{a,x,y} \oslash z} & (a \oslash (x \otimes y)) \oslash z & \xrightarrow{\mathsf{passoc}_{a,x\otimes y,z}} & a \oslash ((x \otimes y) \otimes z) \\ \downarrow{\scriptstyle \mathsf{passoc}_{a\oslash x,y,z}} & & & \swarrow{\scriptstyle a \oslash \mathrm{assoc}_{x,y,z}} & \\ (a \oslash x) \oslash (y \otimes z) & & \xrightarrow{\mathsf{passoc}_{a,x,(y\otimes z)}} & & a \oslash (x \otimes (y \otimes z)) \end{array}$$

$$\begin{array}{ccc} a \oslash x & \xrightarrow{a \oslash \mathrm{lunit}_x} & a \oslash (I \otimes x) \\ \downarrow{\scriptstyle \mathsf{r}_a \oslash x} & \nearrow{\scriptstyle \mathsf{passoc}_{a,I,x}} & \\ (a \oslash I) \oslash x & & \end{array} \qquad \begin{array}{ccc} a \oslash x & \xrightarrow{a \oslash \mathrm{runit}_x} & a \oslash (x \otimes I) \\ \downarrow{\scriptstyle \mathsf{r}_{a\oslash x}} & \nearrow{\scriptstyle \mathsf{passoc}_{a,x,I}} & \\ (a \oslash x) \oslash I & & \end{array}$$

commute; and

- a lax morphism of actions from $\_ \oslash \_$ to the right tensor multiplication action $\_ \otimes \_$ of $\mathcal{C}$ on itself; i.e., a functor $J \colon \mathcal{C}_s \to \mathcal{C}$ and a natural transformation $\mathrm{wk}_{a,x} \colon Ja \otimes x \to J(a \oslash x)$ that makes the following diagrams commute.

$$\begin{array}{ccccc} (Ja \otimes x) \otimes y & \xrightarrow{\mathrm{wk}_{a,x} \otimes y} & J(a \oslash x) \otimes y & \xrightarrow{\mathrm{wk}_{a\oslash x,y}} & J((a \oslash x) \oslash y) \\ \downarrow{\scriptstyle \mathrm{assoc}_{Ja,x,y}} & & & \swarrow{\scriptstyle J\,\mathsf{passoc}_{a,x,y}} & \\ Ja \otimes (x \otimes y) & & \xrightarrow{\mathrm{wk}_{a,x\otimes y}} & & J(a \oslash (x \otimes y)) \end{array}$$

$$\begin{array}{ccc} Ja & \xrightarrow{J\,\mathsf{r}_a} & J(a \oslash I) \\ \downarrow{\scriptstyle \mathrm{runit}_{Ja}} & \nearrow{\scriptstyle \mathrm{wk}_{a,I}} & \\ Ja \otimes I & & \end{array}$$

Remark 2.11.10. The definitions of *lax action* can be found at the start of Chapter 5, while that of an *oplax morphism of actions* is found at Definition 5.0.2. The definitions we have used here are not quite the same as the ones from Chapter 5, but they are similar: a *strong action* is a lax action in which the coherences (called m and e in Chapter 5 and passoc and r here) are isomorphisms. A *lax morphism of actions* is defined in the same way as an oplax morphism, except that the coherence (called μ in Definition 5.0.2 and wk here) goes in the opposite direction.

Proposition 2.11.11. *The monoidal category $\mathcal{G}$, together with the category $\mathcal{G}_s$, the natural transformations*

$$\mathsf{passoc}_{A,X,Y} = \mathrm{cc}_{\mathrm{assoc}_{M_A,M_X,M_Y}} : (A \oslash X) \oslash Y \xrightarrow{\cong} A \oslash (X \otimes Y)$$

$$\mathsf{r}_A = \mathrm{cc}_{\mathrm{runit}_{M_A}} : A \xrightarrow{\cong} A \oslash I\,,$$

the inclusion functor $J\colon \mathcal{G}_s \to G$ and the natural transformation

$$\mathrm{wk}_{A,X} \colon JA \otimes X = A \otimes X \to A \oslash X = J(A \oslash X)$$

form a sequoidal category.

Proof. We have shown most of this already; all that remains is to show that passoc and r are natural transformations and that the five diagrams in Definition 2.11.9 commute.

Let us start with the diagrams. By Proposition 2.6.5, commutativity of these diagrams follows from commutativity of the diagrams formed from the corresponding subset inclusion functions in **Set**. For example, to show that the first diagram commutes in $\mathcal{G}$, we must show that the following diagram commutes in **Set**.

$$\begin{array}{ccc}
((M_A + M_X) + M_Y) + M_Z & \xrightarrow{[\mathrm{assoc}_{M_A,M_X,M_Y},\mathrm{id}_{M_Z}]} & (M_A + (M_X + M_Y)) + M_Z \\
\Big\downarrow{\scriptstyle \mathrm{assoc}_{M_A+M_X,M_Y,Z}} & & \Big\downarrow{\scriptstyle \mathrm{assoc}_{M_A,M_X+M_Y,M_Z}} \\
(M_A + M_X) + (M_Y + M_Z) & & M_A + ((M_X + M_Y) + M_Z) \\
& \searrow^{\mathrm{assoc}_{M_A,M_X,M_Y+M_Z}} & \Big\downarrow{\scriptstyle [\mathrm{id}_{M_A},\mathrm{assoc}_{M_X,M_Y,M_Z}]} \\
& & M_A + (M_X + (M_Y + M_Z))
\end{array}$$

This diagram is, of course, none other than the pentagon diagram for the coproduct $+$ in **Set**. Similarly, the second and third diagrams in Definition 2.11.9 reduce in this case to the triangle diagrams for the coproduct $+$ in **Set**.

For the fourth diagram in Definition 2.11.9, since wk is a subset inclusion strategy induced from an identity map, Proposition 2.6.5 tells us that both arms of the diagram are the strategy induced by the subset inclusion

$$\mathrm{assoc}_{M_A,M_X,M_Y} : (M_A + M_X) + M_Y \to M_A + (M_X + M_Y)$$

from $(A \otimes X) \otimes Y$ to $A \oslash (X \otimes Y)$. Similarly, both arms of the last diagram in Definition 2.11.9 are the strategies induced by the subset inclusion $\mathrm{runit}_{M_A} : M_A \to M_A + \varnothing$ from A to $A \oslash I$.

It now remains only to show that passoc and r are natural transformations. For passoc, suppose that A', X', Y', A, X, Y are games, that $\sigma : A' \multimap A$ is a strict strategy and that $\tau : B' \multimap B, \upsilon : C' \multimap C$ are strategies. Then we need to show that the following diagram commutes.

$$\begin{array}{ccc} (A' \oslash X') \oslash Y' & \xrightarrow{\mathsf{passoc}_{A',X',Y'}} & A' \oslash (X' \otimes Y') \\ {\scriptstyle (\sigma\oslash\tau)\oslash\upsilon}\downarrow & & \downarrow{\scriptstyle \sigma\oslash(\tau\otimes\upsilon)} \\ (A \oslash X) \oslash Y & \xrightarrow{\mathsf{passoc}_{A,X,Y}} & A \oslash (X \otimes Y) \end{array}$$

We have

$$\begin{aligned} & (\mathrm{wk}_{A',X'} \otimes Y'); \mathrm{wk}_{A' \oslash X',Y'}; \mathsf{passoc}_{A',X',Y'}; (\sigma \oslash (\tau \otimes \upsilon)) \\ &= \mathrm{assoc}_{A',X',Y'}; \mathrm{wk}_{A',X' \otimes Y'}; (\sigma \oslash (\tau \otimes \upsilon)) && \text{(see above)} \\ &= \mathrm{assoc}_{A',X',Y'}; (\sigma \otimes (\tau \otimes \upsilon)); \mathrm{wk}_{A,X \otimes Y} && \textit{Prop. 2.11.8} \\ &= ((\sigma \otimes \tau) \otimes \upsilon); \mathrm{assoc}_{A,X,Y}; \mathrm{wk}_{A,X \otimes Y} && \textit{Prop. 2.7.8} \\ &= ((\sigma \otimes \tau) \otimes \upsilon); (\mathrm{wk}_{A,X} \otimes Y); \mathrm{wk}_{A \oslash X,Y}; \mathsf{passoc}_{A,X,Y} && \text{(see above)} \\ &= (\mathrm{wk}_{A',X'} \otimes Y'); ((\sigma \oslash \tau) \otimes \upsilon); \mathrm{wk}_{A \oslash X,Y}; \mathsf{passoc}_{A,X,Y} && \textit{Prop. 2.11.8} \\ &= (\mathrm{wk}_{A',X'} \otimes Y'); \mathrm{wk}_{A' \oslash X',Y'}; ((\sigma \oslash \tau) \oslash \upsilon); \mathsf{passoc}_{A,X,Y} \ . && \textit{Prop. 2.11.8} \end{aligned}$$

Now observe that $\mathrm{wk}_{A',X'} \otimes Y = \mathrm{subs}_{[\mathrm{id}_{M_{A'}+M_{X'}}, \mathrm{id}_{M_{Y'}}]}$ by Proposition 2.7.7, so it is an epimorphism by Proposition 2.6.9. Proposition 2.6.9 also tells us that $\mathrm{wk}_{A' \oslash X',Y'}$ is an epimorphism. Therefore, we have

$$\mathsf{passoc}_{A',X',Y'}; (\sigma \oslash (\tau \otimes \upsilon)) = ((\sigma \oslash \tau) \oslash \upsilon); \mathsf{passoc}_{A,X,Y}$$

for any $A', X', Y', A, X, Y, \sigma, \tau, \upsilon$ as above. It follows that passoc is a natural transformation.

The proof that r is a natural transformation is similar. Let A', A be games and let $\sigma : A' \multimap A$ be a strict strategy. We need to show that the following diagram commutes.

$$\begin{array}{ccc} A' & \xrightarrow{\mathsf{r}_{A'}} & A' \oslash I \\ {\scriptstyle \sigma}\downarrow & & \downarrow{\scriptstyle \sigma\oslash I} \\ A & \xrightarrow{\mathsf{r}_{A}} & A \oslash I \end{array}$$

Indeed, we have

$$\begin{aligned}
\mathsf{r}_{A'};(\sigma \oslash I) &= \mathrm{runit}_{A'};\mathrm{wk}_{A',I};(\sigma \otimes I) && \text{(see above)}\\
&= \mathrm{runit}_{A'};(\sigma \otimes I);\mathrm{wk}_{A,I} && \textit{Prop. 2.11.8}\\
&= \sigma;\mathrm{runit}_{A};\mathrm{wk}_{A,I} && \textit{Prop. 2.7.8}\\
&= \sigma;\mathsf{r}_{A}\ . && \text{(see above)}
\end{aligned}$$

Therefore, r is a natural transformation, which completes our check of the criteria required by Definition 2.11.9. □

2.12 Sequoidally decomposable categories

The definition of a sequoidal category captures some of the important properties of the sequoid operator $\oslash$ in the category of games, but not all of them. In this section, we will consider some further category-theoretic properties of the sequoid operator.

Definition 2.12.1 ([CLM11])**.** Let $\mathcal{C}$ be a sequoidal category such that C_s has arbitrary products (including a terminal object 1). We say that $\mathcal{C}$ is *distributive* if whenever a_i is a collection of objects of $\mathcal{C}'$ and x is an object of $\mathcal{C}$, the morphism

$$\mathsf{dist}_{(a_i),x} = \langle \mathrm{pr}_i \oslash x\rangle \colon \left(\prod_i a_i\right) \oslash x \to \prod_i (a_i \oslash x)$$

is an isomorphism.

Remark 2.12.2. In particular, taking (a_i) to be the empty collection, the morphism $\mathsf{l}_x = () \colon 1 \oslash x \to 1$ is an isomorphism.

Proposition 2.12.3. *$\mathcal{G}$ is a distributive sequoidal category.*

Proof. Let $(A_i), X$ be games. By Proposition 2.10.2, the morphism $\langle \mathrm{pr}_i \oslash X\rangle$ is given by the tree immersion $\phi \colon P_{\prod_i (A_i \oslash X)} \to P_{\prod_i A_i \oslash X}$ defined as follows.

$$\phi(s) = \begin{cases} \epsilon & \text{if } s = \epsilon \\ [\mathrm{in}_{A_j}, \mathrm{in}_X]_*(s) & \text{if } s \text{ begins with a move in the } j\text{-th component} \end{cases}$$

When we say $[\mathrm{in}_{A_j}, \mathrm{in}_{X^j}]_*(s)$, we have considered s as a sequence in $(M_{A_j} + M_{X^j})^*$.

We claim that ϕ is a bijection. Indeed, it is certainly injective, since if $\phi(s) = \phi(t)$, then the first move of $\phi(s) = \phi(t)$ occurs in one of the A_j, which means that s, t must both come from the j-th component. Then,

if we have a non-empty sequence $s \in P_{\prod_i A_i \oslash X)}$, then s must start with a move in some A_j, and must thereafter take place in the games A_j and X. Then $s = \phi([\mathrm{in}_{A_j}, \mathrm{in}_{X^j}]_*(s))$, where we have considered s as a sequence in $(M_{A_j} + M_{X^j})^*$.

Therefore, ϕ is a tree isomorphism, so $\mathsf{dist}_{(A_i),X} = \mathrm{zz}_\phi$ is an isomorphism by Proposition 2.9.6. □

We can get a distributivity result in the other direction, but this one is not as strong, since the morphism we get is only a monomorphism, not an isomorphism.

Definition 2.12.4. Let $\mathcal{C}$ be a distributive sequoidal category. We say that $\mathcal{C}$ is *strongly distributive* if whenever $(A_i), (B_i), C$ are objects of $\mathcal{C}_s$, where (B_i) is a non-empty collection, then the morphism

$$\langle JC \otimes J(A_1 \oslash (\cdots \oslash (A_n \oslash J(\mathrm{pr}_i)) \cdots))\rangle$$

is a monomorphism

$$JC \otimes J\left(A_1 \oslash \left(\cdots \oslash \left(A_n \oslash J\left(\prod_i B_i\right)\right)\cdots\right)\right) \to$$

$$\prod_i (JC \otimes J(A_1 \oslash (\cdots \oslash (A_n \oslash J(B_i)) \cdots)))\,.$$

Proposition 2.12.5. *$\mathcal{G}$ is a strongly distributive sequoidal category.*

Proof. We first ignore the $JC \otimes \_$ part.

By Proposition 2.10.2, the morphism $\langle (A_1 \oslash (\cdots \oslash (A_n \oslash \mathrm{pr}_i) \cdots))\rangle$ is given by the tree immersion

$$\phi \colon P_{\prod_i (A_1 \oslash (\cdots \oslash (A_n \oslash B_i) \cdots))} \to P_{A_1 \oslash (\cdots \oslash (A_n \oslash \prod_i B_i))}$$

defined as follows.

$$\phi(s) = \begin{cases} \epsilon & \text{if } s = \epsilon \\ [\mathrm{in}_{A_1,\cdots,A_n}, \mathrm{in}_{B_j}]_*(s) & \text{if } s \text{ begins with a move in the } j\text{-th component} \end{cases}$$

Note that ϕ is not in general injective, since if s occurs entirely inside one of the copies of the A_i, then $\phi(s) = \phi(s')$ for any identical sequence s' occurring inside one of the other copies of the A_i.

We claim that ϕ is surjective. Indeed, let $t \in P_{A_1 \oslash (\cdots \oslash (\prod_i B_i))}$ be a non-empty sequence. If t contains moves in one of the B_j, then we have $t =$

$\phi([\mathrm{in}_{A_i^j}, \mathrm{in}_{B_j}]_*(t))$, where A_i^j is the copy of A_i in the j-th component of the product and we have considered t as a sequence in $(M_{A_1} + \cdots + M_{A_n} + M_{B_j})^*$. If t only contains moves in the A_i, then pick some fixed index 0; then we have $t = \phi([\mathrm{in}_{A_i^0}]_*(t))$, where we have considered t as a sequence in $(M_{A_1} + \cdots + M_{A_n})^*$.

Therefore, ϕ is a monomorphism by Proposition 2.9.9.

Now, if we take the tensor product on the left by JC, we still get a surjective tree immersion and hence a monomorphism. □

Definition 2.12.6. A sequoidal category $\mathcal{C}$ is *inclusive* if $\mathcal{C}_s$ is a full-on-objects subcategory of $\mathcal{C}$ containing wk and all isomorphisms of $\mathcal{G}$, and the functor J is the inclusion functor.

In such a situation, we will sometimes drop the mention of the functor J.

Proposition 2.12.7. *$\mathcal{G}$ is an inclusive sequoidal category.*

Proof. The only thing we really need to check is that isomorphisms in $\mathcal{G}$ are always strict strategies. Indeed, if σ is a strategy for $A \multimap B$ and τ a strategy for $B \multimap A$ such that $\sigma;\tau = \mathrm{id}_A$, then for any opening move a in A on the right of τ there is some $\mathfrak{s} \in \mathrm{int}(A, B, A)$ such that $\mathfrak{s}|_{A,A} = aa$, and therefore the reply to a in τ must take place in B. □

An important fact about the sequoid operator for games is that it gives us a way to decompose the tensor product as

$$A \otimes B \cong (A \oslash B) \times (B \oslash A)\,.$$

Informally, this is because both sides allow player O to start either in A or in B on the first move, and thereafter switch between A and B as he chooses.

Definition 2.12.8. Let $\mathcal{C}$ be a distributive inclusive sequoidal category, where $\mathcal{C}$ is a symmetric monoidal category. We say that $\mathcal{C}$ is *decomposable* if the morphisms

$$\mathsf{dec}_{a,b} = \langle \mathrm{wk}_{a,Jb}, \mathrm{sym}_{Ja,Jb}; \mathrm{wk}_{b,Ja} \rangle \colon Ja \otimes Jb \to (a \oslash Jb) \times (b \oslash Ja)$$

$$() \colon I \to 1$$

are isomorphisms in $\mathcal{C}_s$.

Proposition 2.12.9. *Let $\mathcal{C}$ be a decomposable sequoidal category and suppose that $a_1, \cdots, a_n$ is a list of objects of $\mathcal{C}_s$. Then we have an isomorphism*

$$a_1 \otimes \cdots \otimes a_n \cong \prod_{i=1}^{n} (a_i \oslash (a_1 \otimes \cdots \otimes a_{i-1} \otimes a_{i+1} \otimes \cdots \otimes a_n))\,.$$

Proof. Induction on n. If $n = 0$, then we have the isomorphism $(): I \to 1$. More generally, we have

$$\begin{aligned}
& a_1 \otimes \cdots \otimes a_{n+1} \\
\cong \quad & (a_1 \otimes \cdots \otimes a_n) \otimes a_{n+1} \\
\xrightarrow{\mathsf{dec}} \quad & (a_1 \otimes \cdots \otimes a_n) \oslash a_{n+1} \times a_{n+1} \oslash (a_1 \otimes \cdots \otimes a_n) \\
\cong \quad & \left(\prod_{i=1}^{n} \left(a_i \oslash \bigotimes_{j \neq i}^{j \leq n} a_j \right) \right) \oslash a_{n+1} \times a_{n+1} \oslash (a_1 \otimes \cdots \otimes a_n) \\
\xrightarrow{\mathsf{dist} \times \mathrm{id}} \quad & \prod_{i=1}^{n} \left(\left(a_i \oslash \bigotimes_{j \neq i}^{j \leq n} a_j \right) \oslash a_{n+1} \right) \times a_{n+1} \oslash (a_1 \otimes \cdots \otimes a_n) \\
\xrightarrow{\langle \mathsf{passoc} \rangle \times \mathrm{id}} \quad & \prod_{i=1}^{n} \left(a_i \oslash \left(\bigotimes_{j \neq i}^{j \leq n} a_j \otimes a_{n+1} \right) \right) \times a_{n+1} \oslash (a_1 \otimes \cdots \otimes a_n) \\
\cong \quad & \prod_{i=1}^{n+1} \left(a_i \times \bigotimes_{j \neq i}^{j \leq n+1} a_j \right) \times a_{n+1} \oslash (a_1 \otimes \cdots \otimes a_n) \\
\cong \quad & \prod_{i=1}^{n+1} (a_i \oslash (a_1 \otimes \cdots \otimes a_{i-1} \otimes a_{i+1} \otimes \cdots \otimes a_{n+1})) \,,
\end{aligned}$$

where each of the arrows is an isomorphism. □

The formula for the isomorphism given in the proof of Proposition 2.12.9 is rather complicated. Our next task will be to give an equivalent formulation that will be simpler to work with later.

Definition 2.12.10. Given objects $a_1, \cdots, a_n$ of a monoidal category, we write sym_i^n for the unique symmetric coherence isomorphism

$$a_1 \otimes \cdots \otimes a_n \cong a_i \otimes a_1 \cdots \otimes a_{i-1} \otimes a_{i+1} \otimes a_n \,.$$

Proposition 2.12.11. *The isomorphism in the proof of Proposition 2.12.9 is given by*

$$\mathsf{dec}^n_{(a_i)} = \langle \mathrm{sym}_i^n ; \mathrm{wk}_{a_i, a_1 \otimes \cdots \otimes a_{i-1} \otimes a_{i+1} \otimes \cdots \otimes a_n} \rangle \,.$$

Proof. Induction on n. We will make use of the coherence theorem for symmetric monoidal categories [Mac71, §11] to allow us to elide associators. The base case is obviously true, because $(): I \to 1$ is the unique

morphism between these objects. Otherwise, we observe that the morphism into $\prod_{i=1}^{n+1}\left(a_i \oslash \bigotimes_{j\neq i}^{j\leq n+1} a_j\right)$ is given component-wise by morphisms $a_1 \otimes \cdots \otimes a_{n+1} \to a_i \oslash \bigotimes_{j\neq i}^{j\leq n+1}$ for each $i = 1, \cdots, n+1$; we need to check that each of these components is equal to $\mathrm{sym}_i^{n+1}; \mathrm{wk}_{a_i, \bigotimes_{j\neq i} a_j}$.

If $i \leq n$, then the i-th component of the morphism in the proof of Proposition 2.12.9 is given by the composite thick dashed arrows in Figure 2.1, and is therefore equal to the composite of the solid arrows, which is equal to $\mathrm{sym}_i^{n+1}; \mathrm{wk}_{a_i, \bigotimes_{j\neq i} a_j}$ as desired. The $n+1$-th component of the morphism in the proof of Proposition 2.12.9 is given by the composite

$$\bigotimes_{j=1}^{n+1} a_j \to \bigotimes_{j=1}^{n} a_j \otimes a_{n+1} \xrightarrow{\mathrm{sym}_{\bigotimes_{j=1}^n a_j, a_{n+1}}} a_{n+1} \otimes \bigotimes_{j=1}^{n} a_j \xrightarrow{\mathrm{wk}_{a_{n+1}, \bigotimes_{j=1}^n a_j}} a_{n+1} \oslash \bigotimes_{j=1}^{n} a_j \,,$$

and then we use the fact that the leftmost two morphisms in this composite compose to give us sym_{n+1}^{n+1}. □

Proposition 2.12.12. *$\mathcal{G}$ is a decomposable sequoidal category.*

Proof. Let A, B be games. By Proposition 2.10.2, the strategy

$$\langle \mathrm{wk}_{A,B}, \mathrm{sym}_{A,B}; \mathrm{wk}_{A,B} \rangle$$

is given by the tree immersion ϕ from $(A \oslash B) \times (B \oslash A)$ to $A \otimes B)$ given by

$$\phi(s) = \begin{cases} \epsilon & \text{if } s = \epsilon \\ s|_{A \oslash B} & \text{if } s \text{ takes place entirely within } A \oslash B \\ s|_{B \oslash A} & \text{if } s \text{ takes place entirely within } B \oslash A \end{cases} .$$

We claim that this tree immersion is a bijection. Indeed, it is certainly injective. Now let $s \in P_{A \otimes B}$ be a non-empty play. Then, if s begins with a move in A, we have $s = \phi((\mathrm{in}_{A \oslash B})_*(s))$, and if s begins with a move in B, we have $s = \phi((\mathrm{in}_{B \oslash A})_*(s)$. Therefore, ϕ is a tree isomorphism, so $\mathsf{dec}_{A,B} = \mathrm{zz}_\phi$ is an isomorphism in $\mathcal{G}$.

Lastly, we have $I = 1$ in $\mathcal{G}$, and the unique morphism $I \to 1$ is the identity. □

Definition 2.12.13 ([Lai02])**.** A *sequoidal closed category* is an inclusive sequoidal category $\mathcal{C}$ such that $\mathcal{C}$ is a monoidal closed category (with inner hom $\multimap$) and such that the map $f \mapsto \Lambda(\mathrm{wk}_{A,B}; f)$ defines an isomorphism

$$\Lambda_s \colon \mathcal{C}_s(A \oslash B, C) \xrightarrow{\cong} \mathcal{C}_s(A, B \multimap C) \,.$$

Proposition 2.12.14. *$\mathcal{G}$ is a sequoidal closed category.*

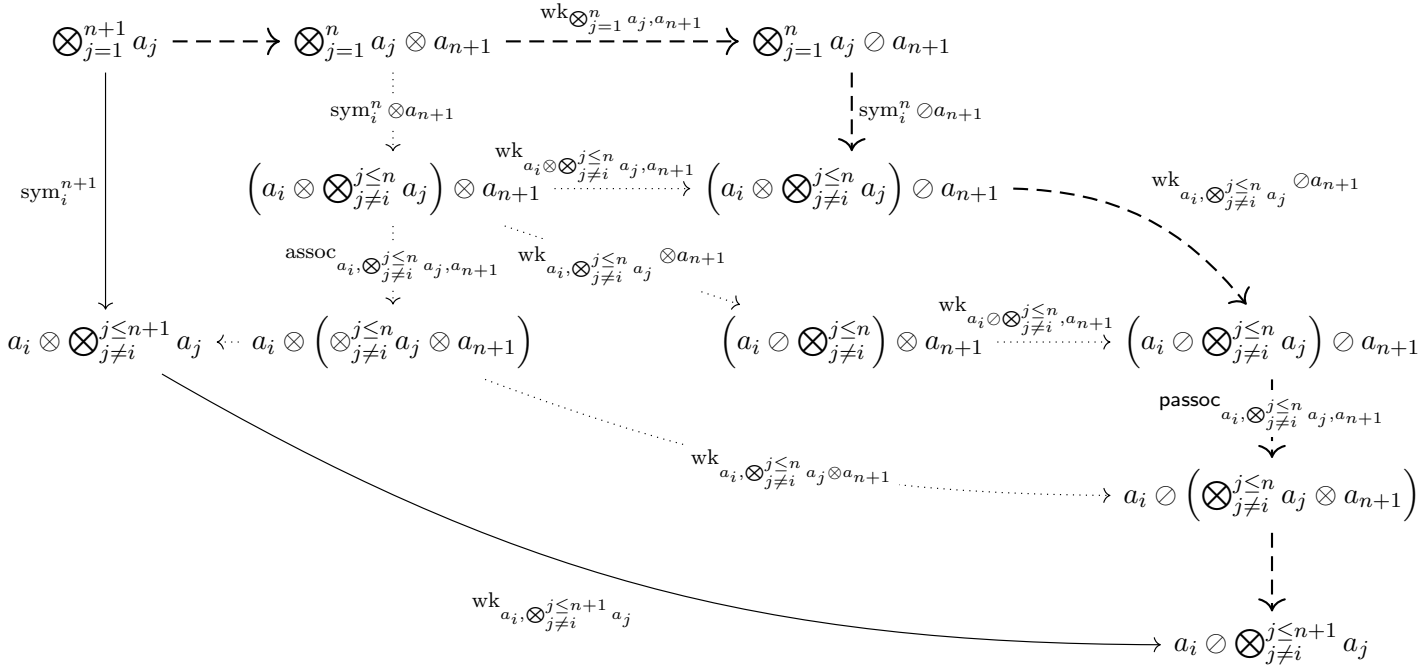


Figure 2.1: Diagram used in the proof of Proposition 2.12.11. The pentagon at the heart of the diagram is the coherence diagram for passoc and wk from Definition 2.11.9.

Proof. Since $\mathrm{wk}_{A,B}$ is an epimorphism and Λ is a bijection, the map is certainly injective. Showing that it is surjective comes down to proving that uncurrying of a strict strategy for $A \multimap (B \multimap C)$ is a strict strategy for $(A \oslash B) \multimap C$. Indeed, after the opening move in C, in both cases player P must play the next move in A. □

2.13 A Formula for the Exponential

We now move on to the categorical semantics of the exponential operator $!\_$. As the name suggests, this operator gives us a way to model the exponential connective from linear logic. More specifically, we shall show that, for a certain class of games A (the *well-opened games*), $!A$ is the carrier for a *cofree commutative comonoid* over A. We use a result of Melliès, Tabareau and Tasson to show why this is the case.

Definition 2.13.1. Let $\mathcal{C}$ be a symmetric monoidal category. Given objects $A_1, \cdots, A_n$ of $\mathcal{C}$ and a permutation $\pi \in \mathfrak{S}_n$, there is a unique canonical symmetry isomorphism

$$\mathrm{sym}_\pi \colon A_1 \otimes \cdots \otimes A_n \xrightarrow{\cong} A_{\pi(1)} \otimes \cdots \otimes A_{\pi(n)} \,.$$

Given an object A of $\mathcal{C}$, an *n-th symmetrized tensor power* of A is an equalizer (A^n, eq^n) for the diagram given by all morphisms of the form

$$\mathrm{sym}_\pi \colon A^{\otimes n} \to A^{\otimes n} \,.$$

We say that the symmetrized tensor power A^n *commutes with the tensor product* if $(B \otimes A^n, B \otimes \mathsf{eq}_n)$ is an equalizer for the diagram given by morphisms of the form

$$B \otimes \mathrm{sym}_\pi \colon B \otimes A^{\otimes n} \to B \otimes A^{\otimes n} \,.$$

Proposition 2.13.2 ([GL17])**.** *Let $\mathcal{C}$ be an inclusive, strongly distributive, decomposable sequoidal category. Then $\mathcal{C}$ has all symmetrized tensor powers and they commute with the tensor product.*

Proof. Let A be an object of $\mathcal{C}$ (equivalently, an object of $\mathcal{C}_s$). We inductively define objects $A^{\oslash n}$ by

- $A^{\oslash 0} = I$; and
- $A^{\oslash(n+1)} = J(A \oslash A^{\oslash n})$.

We claim that $A^{\oslash n}$ is a symmetrized tensor power of A.

Given n, we inductively define a morphism $\mathrm{wk}^n \colon A^{\otimes n} \to A^{\oslash n}$, where $\mathrm{wk}^0 = \mathrm{id}_I$, and wk^{n+1} is given by the composite

$$A^{\otimes(n+1)} \to A \otimes A^{\otimes n} \xrightarrow{A \otimes \mathrm{wk}^n} A \otimes A^{\oslash n} \xrightarrow{\mathrm{wk}_{A, A^{\oslash n}}} A \oslash A^{\oslash n} .$$

We show by induction on n that if B is an object of $\mathcal{C}$ and $k \geq 0$ then the composite

$$B \otimes (A \oslash \_)^k A^{\otimes n} \xrightarrow{\langle B \otimes (A \oslash \_)^k \, \mathrm{sym}_\pi \rangle} (B \otimes (A \oslash \_)^k A^{\otimes n})^{n!} \xrightarrow{(B \otimes (A \oslash \_)^k \, \mathrm{wk}^n)^{n!}} (B \otimes A^{\oslash(k+n)})^{n!}$$

(i.e., the morphism $\langle B \otimes (A \oslash \_)^k (\mathrm{sym}_\pi ; \mathrm{wk}^n) \rangle$) is a monomorphism. In particular, taking $k = 0$, we will have shown that $\langle B \otimes (\mathrm{sym}_\pi ; \mathrm{wk}^n) \rangle$ is a monomorphism.

The hypothesis is clearly true for $n = 0$; in the general case, we have a composite

$$\begin{aligned} & B \otimes (A \oslash \_)^k A^{\otimes(n+1)} \\ \xrightarrow{B \otimes (A \oslash \_)^k \langle \mathrm{sym}_i^{n+1} ; \mathrm{wk}_{A, A^{\otimes n}} \rangle} & B \otimes (A \oslash \_)^k (A \oslash A^{\otimes n})^{n+1} \\ \xrightarrow{\langle B \otimes (A \oslash \_)^k ; \mathrm{pr}_i \rangle} & (B \otimes (A \oslash \_)^{k+1} A^{\otimes n})^{n+1} \\ \longrightarrow & (B \otimes A^{\oslash(k+n+1)})^{(n+1)!} , \end{aligned}$$

where the last arrow is the tensor product of B with the $(n+1)$-th power of the composite given by

$$(A \oslash \_)^{k+1} A^{\otimes n} \xrightarrow{\langle (A \oslash \_)^{k+1} \, \mathrm{sym}_\sigma \rangle} ((A \oslash \_)^{k+1} A^{\otimes n})^{n!} \xrightarrow{((A \oslash \_)^{k+1} \, \mathrm{wk}^n)^{n!}} (A^{\oslash(k+n+1)})^{n!} ,$$

which is a monomorphism by the induction hypothesis. Then the previous composite is the composite of monomorphisms (by our assumptions on $\mathcal{C}$), and is therefore itself a monomorphism. Now this composite may be written as

$$\langle B \otimes (A \oslash \_)^k (\mathrm{sym}_i^{n+1} ; \mathrm{wk}_{A, A^{\otimes n}} ; (A \oslash \mathrm{sym}_\sigma) ; (A \oslash \mathrm{wk}^n)) \rangle ,$$

which, since wk is a natural transformation, is equal to

$$\langle B \otimes (A \oslash \_)^k (\mathrm{sym}_i^{n+1} ; (A \otimes \mathrm{sym}_\sigma) ; (A \otimes \mathrm{wk}^n) ; \mathrm{wk}_{A, A^{\oslash n}}) \rangle ,$$

where σ ranges over the permutations in $\mathfrak{S}_n$. Moreover, by the definition of wk^n, this is equal to

$$\langle B \otimes (A \oslash \_)^k (\mathrm{sym}_i^{n+1} ; (A \otimes \mathrm{sym}_\sigma) ; \mathrm{wk}^{n+1} \rangle .$$

Now, given $i \in \{1, \cdots n+1\}$ and $\sigma \in \mathfrak{S}_n$, there is a unique permutation $\pi \in \mathfrak{S}_{n+1}$ such that $\mathrm{sym}_i^{n+1} ; (A \otimes \mathrm{sym}_\sigma = \mathrm{sym}_\pi)$; moreover, the map $(i, \sigma) \mapsto \pi$ defines a bijection from $\{1, \cdots, n+1\} \times \mathfrak{S}_n \to \mathfrak{S}_{n+1}$. Therefore (after

choosing an appropriate enumeration of our permutations), we see that this composite is in fact equal to

$$\langle B \otimes (A \oslash \_)^k(\mathrm{sym}_\pi; \mathrm{wk}^{n+1})\rangle\,.$$

Therefore, $\langle B \otimes (A \oslash \_)^k(\mathrm{sym}_\pi; \mathrm{wk}^{n+1})\rangle$ is a monomorphism as desired, completing the induction.

Next, we define morphisms $\mathsf{eq}_n \colon A^{\oslash n} \to A^{\otimes n}$ inductively, where $\mathsf{eq}_0 = \mathrm{id}$ and eq_{n+1} is defined by the following composite

$$A^{\oslash(n+1)} = A \oslash A^{\oslash n} \xrightarrow{\langle (A \oslash \mathsf{eq}_n)^n_1 \rangle} (A \oslash A^{\otimes n})^n \cong A^{\otimes(n+1)}\,,$$

where the final isomorphism is as in Propositions 2.12.9 and 2.12.11.

First, we show inductively that $\mathsf{eq}_n; \mathrm{sym}_\pi; \mathrm{wk}^n = \mathrm{id}_{A^{\oslash n}}$ for all permutations π of S_n. This is certainly true for $n = 0$; in the general case, let $\pi \in \mathfrak{S}_{n+1}$ be a permutation. Let $j = \pi^{-1}(1)$ be the element sent to 1 by π and let σ be the permutation of $1, \cdots, n$ such that applying σ to the elements $2, \cdots, n+1$ and composing with π gives us the j-cycle $(1 \ldots j)$. Then we have

$$\mathrm{sym}^{n+1}_j; (A \otimes \mathrm{sym}_\sigma) = \mathrm{sym}_\pi\ .$$

Now we get

$$\begin{aligned}
&\mathsf{eq}_{n+1}; \mathrm{sym}_\pi; \mathrm{wk}^{n+1} \\
&= \langle (A \oslash \mathsf{eq}_n)^n_1 \rangle; (\mathsf{dec}^{n+1}_{\vec{A}})^{-1}; \mathrm{sym}_\pi; (A \otimes \mathrm{wk}^n); \mathrm{wk}_{A, A^{\oslash n}} \\
&= \langle (A \oslash \mathsf{eq}_n)^n_1 \rangle; (\mathsf{dec}^{n+1}_{\vec{A}})^{-1}; \mathrm{sym}^{n+1}_j; (A \otimes \mathrm{sym}_\sigma); (A \otimes \mathrm{wk}^n); \mathrm{wk}_{A, A^{\oslash n}} \\
&= \langle (A \oslash \mathsf{eq}_n)^n_1 \rangle; (\mathsf{dec}^{n+1}_{\vec{A}})^{-1}; \mathrm{sym}^{n+1}_j; \mathrm{wk}_{A, A^{\otimes n}}; (A \oslash \mathrm{sym}_\sigma); (A \oslash \mathrm{wk}^n) \\
&= \langle (A \oslash \mathsf{eq}_n)^n_1 \rangle; (\mathsf{dec}^{n+1}_{\vec{A}})^{-1}; \langle \mathrm{sym}^{n+1}_i; \mathrm{wk}_{A, A^{\otimes n}} \rangle; \mathrm{pr}_j; (A \oslash \mathrm{sym}_\sigma); (A \oslash \mathrm{wk}^n) \\
&= A \oslash (\mathsf{eq}_n; \mathrm{sym}_\sigma; \mathrm{wk}^n)\,,
\end{aligned}$$

which is equal to the identity on $A^{\oslash(n+1)}$ by the induction hypothesis.

Now let ρ be a permutation in $\mathfrak{S}_n$. We claim that $\mathsf{eq}_n = \mathsf{eq}_n; \mathrm{sym}_\rho$.

Indeed, we have

$$\begin{aligned}
\mathsf{eq}_n; \langle \mathrm{sym}_\pi; \mathrm{wk}^n \rangle &= \langle \mathsf{eq}_n; \mathrm{sym}_\pi; \mathrm{wk}^n \rangle \\
&= \langle \mathrm{id} \rangle \\
&= \langle \mathsf{eq}_n; \mathrm{sym}_{\rho\pi}; \mathrm{wk}^n \rangle \\
&= \mathsf{eq}_n; \mathrm{sym}_\rho; \langle \mathrm{sym}_\pi; \mathrm{wk}^n \rangle\,.
\end{aligned}$$

Since $\langle \mathrm{sym}_\pi; \mathrm{wk}^n \rangle$ is a monomorphism, this means that $\mathsf{eq}_n = \mathsf{eq}_n; \mathrm{sym}_\rho$, as desired. Therefore, eq_n equalizes the morphisms eq_n. We claim that it is an equalizer, and that this equalizer is preserved by the tensor product.

Indeed, let B, C be objects of $\mathcal{C}$, and let $f\colon C \to B \otimes A^{\otimes n}$ be a morphism such that $f = f; (B \otimes \mathrm{sym}_\pi)$ for all $\pi \in \mathfrak{S}_n$.

Let $\tilde{f} = f; (B \otimes \mathrm{wk}^n)\colon C \to B \otimes A^{\oslash n}$. We claim that $\tilde{f}; (B \otimes \mathsf{eq}_n) = f$; indeed, we have

$$\begin{aligned}\tilde{f}; (B \otimes \mathsf{eq}_n); \langle B \otimes (\mathrm{sym}_\pi; \mathrm{wk}^n)\rangle &= \langle f; (B \otimes (\mathrm{wk}^n; \mathsf{eq}_n; \mathrm{sym}_\pi; \mathrm{wk}^n))\rangle \\ &= \langle f; (B \otimes \mathrm{wk}^n)\rangle \\ &= f; \langle B \otimes (\mathrm{sym}_\pi; \mathrm{wk}^n)\rangle\,.\end{aligned}$$

Therefore, since $\langle B \otimes (\mathrm{sym}_\pi; \mathrm{wk}^n)\rangle$ is a monomorphism, we know that $\tilde{f}; (B \otimes \mathsf{eq}_n) = f$.

Now suppose that $h\colon C \to B \otimes A^{\oslash n}$ is such that $h; (B \otimes eq_n) = f$. We claim that $h = \tilde{f}$. Indeed, we have

$$\tilde{f} = f; (B \otimes \mathrm{wk}^n) = h; (B \otimes \mathsf{eq}_n); (B \otimes \mathrm{wk}^n) = h\,.$$

Therefore, $(B \otimes A^{\oslash n}, B \otimes \mathsf{eq}_n)$ is an equalizer of the arrows $B \otimes \mathrm{sym}_\pi\colon B \otimes A^{\otimes n} \to B \otimes A^{\otimes n}$, as desired. □

We are interested in symmetrized tensor powers because of an important result of Melliès, Tabareau and Tasson. Suppose $\mathcal{C}$ is a monoidal category with a terminal unit object, and that $\mathcal{C}$ has symmetrized tensor powers that commute with the tensor product. Given n, we have a morphism

$$A^{\otimes n} \otimes ()\colon A^{\otimes(n+1)} \to A^{\otimes n}\,,$$

where $()$ is the unique morphism into the terminal object. Then, if A^n and A^{n+1} are the n-th and $n{+}1$-th symmetrized tensor powers of A, and $\mathsf{eq}_{n+1}, \mathsf{eq}$ the corresponding equalization, for any $\pi \in \mathfrak{S}_n$ we have a commutative diagram

$$\begin{array}{ccccc} B \otimes A^{n+1} & \xrightarrow{B \otimes \mathsf{eq}_{n+1}} & B \otimes A^{\otimes(n+1)} & \xrightarrow{B \otimes A^{\otimes n} \otimes ()} & B \otimes A^{\otimes n} \\ & & \downarrow{\scriptstyle \mathrm{sym}_{\pi'}} & & \downarrow{\scriptstyle \mathrm{sym}_\pi} \\ & & B \otimes A^{\otimes(n+1)} & \xrightarrow{B \otimes A^{\otimes n} \otimes ()} & B \otimes A^{\otimes n} \end{array}\,,$$

where π' is the permutation of $1, \cdots, n+1$ that fixes 1 and applies π to the remaining elements $2, \cdots, n+1$.

This means that for each $\pi \in \mathfrak{S}_n$ we have

$$\begin{aligned}(B \otimes \mathsf{eq}_{n+1}); (B \otimes A^{\otimes n} \otimes ()) &= (B \otimes \mathsf{eq}_{n+1}); (B \otimes \mathrm{sym}_{\pi'}); (B \otimes A^{\otimes n} \otimes ()) \\ &= (B \otimes \mathsf{eq}_{n+1}); (B \otimes A^{\otimes n} \otimes ()); (B \otimes \mathrm{sym}_\pi)\,,\end{aligned}$$

and that there is therefore an induced morphism

$$B \otimes A^{n+1} \to B \otimes A^n \,,$$

by the universal property of the equalizer.

Note also that if m, n are integers, then any permutations σ of $1, \cdots, m$ and π of $1, \cdots, n$ induce a permutation $[\sigma, \pi]$ of $1, \cdots, m+n$. Then we get morphisms

$$A^{m+n} \to A^{\otimes(m+n)} \to A^{\otimes m} \otimes A^{\otimes n} \,,$$

which are equalized by all symmetries on $A^{\otimes m}$ and $A^{\otimes n}$ individually. Since the equalizers A^m and A^n are preserved by the tensor product, then we get an induced morphisms

$$A^{m+n} \to A^m \otimes A^n \,.$$

Theorem 2.13.3 ([MTT09])**.** *Let $\mathcal{C}$ be a monoidal category such that the monoidal unit for $\mathcal{C}$ is a terminal object. Suppose that $\mathcal{C}$ has symmetrized tensor powers that commute with the tensor product.*

Then, for any objects A, B of $\mathcal{C}$, there is a natural sequence

$$B \leftarrow B \otimes A \leftarrow B \otimes A^2 \leftarrow B \otimes A^3 \leftarrow \cdots \,.$$

In particular, there is a sequence

$$I \leftarrow A \leftarrow A^2 \leftarrow A^3 \leftarrow \cdots \,.$$

Suppose that this sequence has a limit $!A$, and suppose moreover that tensoring this limiting cone with B exhibits $B \otimes !A$ as the limit of

$$B \leftarrow B \otimes A \leftarrow B \otimes A^2 \leftarrow B \otimes A^3 \leftarrow \cdots \,.$$

for any B.

For each m, we can define morphisms

$$!A \to A^{m+n} \to A^m \otimes A^n$$

for each n, which commute with the morphisms $A^{n+1} \to A^n$ and hence induce a morphism $!A \to A^m \otimes !A$. Then these morphisms themselves commute with the morphisms $A^{m+1} \otimes !A \to A^m \otimes !A$, and so we get a morphism $\mu_A \colon !A \to !A \otimes !A$.

The morphisms $\mu_A \colon !A \to !A \otimes !A$ and $() \colon A \to I$ give $!A$ the structure of a commutative comonoid in $\mathcal{C}$. In fact, this is the cofree commutative comonoid *over A in $\mathcal{C}$. The corresponding counit $\mathtt{der}_A \colon !A \to A$ is the morphism in the limiting cone.*

We want to show that this theorem applies in $\mathcal{G}$. First, we find an explicit formula for the morphisms $B \otimes A^{\oslash(n+1)} \to B \otimes A^{\oslash n}$.

Proposition 2.13.4. *Let $\mathcal{C}$ be an inclusive, strongly distributive, decomposable sequoidal category – so $\mathcal{G}$ has symmetrized tensor powers preserved by the tensor product as in Proposition 2.13.2.*

Then the canonical morphisms $B \otimes A^{\oslash(n+1)} \to B \otimes A^{\oslash n}$ are given by

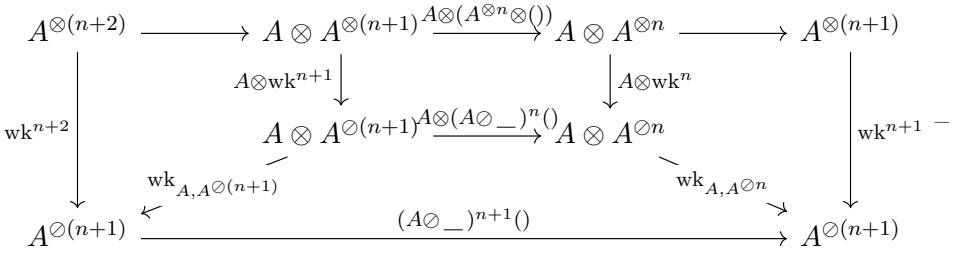
$$B \otimes (A \oslash \_)^n() \,,$$

where $(): A \to I$ is the unique morphism into the terminal object.

Proof. First, we show by induction on n that the following diagram commutes.

$$\begin{array}{ccc} A^{\otimes(n+1)} & \xrightarrow{A^{\otimes n}\otimes()} & A^{\otimes n} \\ {\scriptstyle A\otimes \mathrm{wk}^{n+1}}\downarrow & & \downarrow{\scriptstyle \mathrm{wk}^n} \\ A^{\oslash(n+1)} & \xrightarrow{(A\oslash\_)^n()} & A^{\oslash n} \end{array}$$

This is clearly true for $n = 0$; in the general case, we have the following commutative diagram –



where the middle square is the inductive hypothesis (tensored by A), the outer trapezia are the definitions of wk^{n+2} and wk^{n+1}, and the bottom trapezium commutes because wk is a natural transformation.

Now, by the proof of Proposition 2.13.2, the canonical morphism $B \otimes A^{\oslash(n+1)} \to B \otimes A^{\oslash n}$ must be constructed as the composite

$$B \otimes (\mathsf{eq}_{n+1}; (A^{\otimes n} \otimes ()); \mathrm{wk}^n) \,,$$

which we have shown is equal to

$$B \otimes (\mathsf{eq}_{n+1}; \mathrm{wk}^{n+1}; (A \oslash \_)^n()) = B \otimes (A \oslash \_)^n() \,. \qquad \square$$

Definition 2.13.5. Let A be a game. We say that A is *well-opened* if initial moves of A can only occur as the very first move in a play.

It is immediate from the definitions that:

- the empty game I and our data-type games $\mathbb{C}, \mathbb{B}, \mathbb{N}$ are well opened;
- if A_i are well-opened games, then so is $\prod_i A_i$; and
- if B is a well-opened game, then so is $A \multimap B$;

but that $A \otimes B$, $A \oslash B$ and $!A$ are not in general well-opened, even if both A and B are.

Proposition 2.13.6. *Let A be a well-opened game. Then we have natural morphisms*

$$!A \to A^{\oslash n}$$

for each n, and these commute with the natural morphisms $A^{\oslash(n+1)} \to A^{\oslash n}$ and make $!A$ the limit of the sequence

$$I \leftarrow A \leftarrow A^{\oslash 2} \leftarrow A^{\oslash 3} \leftarrow \cdots .$$

Moreover, if B is any game, then $B \otimes !A$ is the limit of the sequence

$$B \leftarrow B \otimes A \leftarrow B \otimes A^{\oslash 2} \leftarrow B \otimes A^{\oslash 3} \leftarrow \cdots .$$

Proof. For the sake of notational simplicity, we will only prove the first part of the Proposition, but the second part (the $B \otimes \cdots$ version) goes through in exactly the same way.

The morphism in question is the (non-innocent) zigzag strategy given by the tree immersion $\phi_n \colon P_{A^{\oslash n}} \to P_{!A}$ defined by

$$\phi_n(s) = \nabla_*(s)\,,$$

where $\nabla \colon M_A + \cdots + M_A$ is the co-diagonal.

We have seen already that the natural morphism $A^{\oslash(n+1)} \to A^{\oslash n}$ is the copycat morphism generated by the inclusion $n.(M_A) \to (n+1).(M_A)$, and so it is clear that these commute with zz_ϕ by Proposition 2.9.6.

Now let C be a game and suppose that there are strategies $\sigma_n \colon C \multimap A^{\oslash n}$ that commute with the natural morphisms $A^{\oslash(n+1)} \to A^{\oslash n}$. Then we define a morphism $\sigma \colon C \to !A$ by

$$\sigma = \left\{ s \in P_{C \multimap !A} \;\middle|\; \begin{array}{l} \text{for some } n,\ s|_{!A} \text{ contains} \\ \text{at most } n \text{ initial moves, and} \\ [\mathrm{in}, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(s) \in \sigma_n. \end{array} \right\} .$$

Here we have used the fact that A is well-opened to tell us that $s|_{!A}$ is indeed a valid play in $A^{\oslash n}$.

We claim that σ is indeed a strategy. First we show that σ is prefix closed. If $s \in \sigma$ and $t \sqsubseteq s$, write n for the number of initial moves in s. Then t has at most n initial moves; if $[\mathrm{id}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(s) \in \sigma_n$, then $[\mathrm{id}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(t) \in \sigma_n$, and therefore $t \in \sigma$.

Now note that if $s \in C_{C \multimap !A}$ is such that $s|_{!A}$ contains k initial moves, and if $m, n \geq k$, then

$$[\mathrm{in}, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_m}]_*(s) = [\mathrm{in}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(s)\,,$$

since the σ_n commute with the natural morphisms $A^{\oslash(n+1)} \to A^{\oslash n}$. So, if $sab, sac \in \sigma$, then we can assume that

$$[\mathrm{in}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_k}]_*(sab), [\mathrm{in}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_k}]_*(sac) \in \sigma_k$$

for some common k, and therefore that $b = c$.

Now we have $\sigma; \mathrm{zz}_{\phi_n} = \sigma_n$ for each n by Proposition 2.9.6.

Suppose that τ is some other strategy for $C \multimap !A$ such that $\tau; \mathrm{zz}_{\phi_n} = \sigma_n$ for each n. By Proposition 2.9.6, we have

$$\sigma_n = \tau; \mathrm{zz}_{\phi_n} = \{s \in P_{C \multimap A^{\oslash n}} \,:\, s^{\phi_n} \in \tau\}\,.$$

Suppose $s \in \sigma$ and that s contains n initial moves. Then

$$[\mathrm{in}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(s) \in \sigma_n$$

for some n. Therefore, $s = ([\mathrm{in}_C, \mathrm{in}_{A_1}, \cdots, \mathrm{in}_{A_n}]_*(s))^{\phi_n} \in \tau$. So $\sigma \subseteq \tau$.

Conversely, suppose that $t \in \tau$. Suppose that $t|_{!A}$ contains n initial moves. Then $t = s^{\phi_n}$ for some sequence $s \in P_{C \multimap A^{\oslash n}}$, and we must have $s \in \sigma_n$. Therefore, $t \in \sigma$. So $\tau \subseteq \sigma$. □

Therefore, by Theorem 2.13.3, if A is a well-opened game, then $!A$ inherits the structure of a cofree commutative comonoid on A.

Let $\mathcal{G}_{wo}$ denote the category of well-opened games and strategies. Let $\mathrm{CCom}(\mathcal{G})$ denote the category of commutative comonoids with respect to the symmetric monoidal structure on $\mathcal{G}$.

In general, given two commutative comonoids M, N in a symmetric monoidal category $\mathcal{C}$, we can form the *tensor product*

$$M \otimes N \to (M \otimes M) \otimes (N \otimes N) \to (M \otimes N) \otimes (M \otimes N)$$

$$M \otimes N \to I \otimes I \to I\,,$$

and this makes $\mathrm{CCom}(\mathcal{C})$ into a *Cartesian* category.

Now note that we have defined a functor

$$\mathcal{G}_{wo} \to \mathrm{CCom}(\mathcal{G})$$

We define a category $\mathcal{G}^!_{wo}$ to be the span of this functor inside $\mathrm{CCom}(\mathcal{G})$. Then the corresponding functor

$$\mathcal{G}_{wo} \to \mathcal{G}^!_{wo}$$

is a right adjoint, and therefore preserves products. So if A, B are well-opened games, then we get a natural isomorphism of comonoids between the tensor product of the comonoids on $!A$ and $!B$ and the comonoid on $!(A \times B)$. In particular, we have a natural isomorphism

$$!A \otimes !B \cong !(A \times B)\,.$$

$\mathcal{G}^!_{wo}$ therefore inherits the structure of a Cartesian category.

This gives us a category in which the objects are commutative comonoids over well-opened games. A more convenient description of this category is that it is the category where the objects are well-opened games and where the morphisms

$$A \to B$$

are morphisms $!A \to B$ in the original category $\mathcal{G}$. We compose two such morphisms $\sigma\colon !A \multimap B$ and $\tau\colon !B \multimap C$ as

$$!A \xrightarrow{\sigma^\dagger} !B \xrightarrow{\tau} C\,,$$

where the *promotion* operator $\_^\dagger$ from strategies $\sigma\colon !A \to B$ to strategies $\sigma^\dagger\colon !A \to !B$ is the right half of the adjunction between $\mathcal{G}_{wo}$ and $\mathcal{G}^!_{wo}$.

Since the functor $\mathcal{G}_{wo} \to \mathrm{CCom}(\mathcal{G})$ preserves products, $\mathcal{G}^!_{wo}$ obtains a Cartesian structure given by the category-theoretic product $\times$. We claim that it is Cartesian closed, with the function object from A to B given by $!A \multimap B$. Indeed, we have

$$\begin{aligned}
\mathcal{G}^!_{wo}(A, !B \multimap C) &\cong \mathcal{G}(!A, !B \multimap C) \\
&\cong \mathcal{G}(!A \otimes !B, C) \\
&\cong \mathcal{G}(!(A \times B), C) \\
&\cong \mathcal{G}^!_{wo}(A \times B, C)\,.
\end{aligned}$$

We have one thing left to prove.

Proposition 2.13.7. *Let $\sigma\colon !A \multimap B$, $\tau\colon !B \multimap C$ be innocent strategies, where A, B, C are well-opened games. Then the composite of σ and τ in $\mathcal{G}^!_{wo}$ is an innocent strategy.*

Proof. It suffices to show that $\sigma^\dagger$ is innocent, which follows by an argument similar to that in Proposition 2.7.6. □

2.14 The Exponential as a Final Coalgebra

Our proof in the previous section that $!A$ is the cofree commutative comonoid over A (if A is well-opened) leads us towards another category-theoretic property enjoyed by $!A$, one that is more closely tied to the sequoid operator. This property will be key to our treatment of the semantics of state.

Definition 2.14.1. Let $F\colon \mathcal{C} \to \mathcal{C}$ be a functor. A *coalgebra for* F or *F-coalgebra* is an object a of $\mathcal{C}$, together with a morphism $f\colon a \to Fa$.

A *coalgebra homomorphism* from (a, f) to (b, g) is a morphism $h\colon a \to b$ such that the following diagram commutes.

$$\begin{array}{ccc} a & \xrightarrow{f} & Fa \\ {\scriptstyle h}\downarrow & & \downarrow{\scriptstyle Fh} \\ b & \xrightarrow{g} & Fb \end{array}$$

Clearly, the coalgebras for a given functor F form a category. A *final coalgebra* for F is a terminal object for this category; i.e., an F-coalgebra (t, α) such that for all F-coalgebras (a, f) there is a unique morphism $(\!|\, f \,|\!)\colon a \to t$ such that the following diagram commutes.

$$\begin{array}{ccc} a & \xrightarrow{f} & Fa \\ {\scriptstyle (\!|\, f \,|\!)}\downarrow & & \downarrow{\scriptstyle F(\!|\, f \,|\!)} \\ t & \xrightarrow{\alpha} & Ft \end{array}$$

We call $(\!|\, f \,|\!)$ the *anamorphism* of f.

We use two standard pieces of theory about coalgebras.

Theorem 2.14.2 (Lambek's Theorem, [Lam68])**.** *If (t, α) is a final coalgebra for a functor F, then $\alpha\colon t \to Ft$ is an isomorphism with inverse given by $(\!|\, F\alpha \,|\!)$.*

Theorem 2.14.3 (Adámek's Theorem, [Adá03])**.** *Suppose $\mathcal{C}$ has a terminal object 1. By repeatedly applying F to the morphism $F1 \to 1$, we build up a sequence*

$$1 \leftarrow F1 \leftarrow F^2 1 \leftarrow F^3 1 \leftarrow \cdots .$$

If this sequence has a limit $F^\omega 1$, and if the morphism $\beta\colon F(F^\omega 1) \to F^\omega 1$ induced from the universal property of the limit is an isomorphism, then $(F^\omega 1, \beta^{-1})$ is a final coalgebra for F.

Now we have already shown that if A is well-opened, then $!A$ is the limit of the sequence

$$I \leftarrow A \leftarrow A^{\oslash 2} \leftarrow A^{\oslash 3} \leftarrow \cdots ,$$

and this sequence is precisely the sequence from Adámek's Theorem, when $F = J(A \oslash \_) \colon \mathcal{G} \to \mathcal{G}$. Moreover, this limit is preserved when taking the sequoid with A on the left, and so we get that

Corollary 2.14.4. *If A is a well-opened game, then $!A$ is a final coalgebra for the functor $J(A \oslash \_) \colon \mathcal{G} \to \mathcal{G}$.*

In this case, the morphism $!A \to A \oslash !A$ is the zigzag strategy that plays copycat between the different copies of A; i.e., zz_ϕ, where $\phi \colon P_{A \oslash !A} \to P_{!A}$ is the tree isomorphism given by

$$\phi(s) = [\mathrm{in}_{M_A}, \mathrm{id}]_*(s) \, .$$

One small thing we need to do is to relate the two structures on the exponential.

Proposition 2.14.5. *The final coalgebra*

$$\alpha \colon !A \to A \oslash !A$$

is given by the composite

$$!A \xrightarrow{\mu_A} !A \otimes !A \xrightarrow{\mathtt{der}_A \otimes !A} A \otimes !A \xrightarrow{\mathrm{wk}_{A,!A}} A \oslash !A \, .$$

Proof. By Theorem 2.13.3, we can tell that this composite is a copycat strategy between $!A$ and $A \oslash !A$, as is α. Since A is well-opened, there is a unique such strategy. □

2.15 Denotational Semantics of Idealized Algol

At last we are in a position to interpret Idealized Algol within our category $\mathcal{G}^!_{wo}$. The base types `com`, `bool` and `nat` are interpreted by the games $\mathbb{C}$, $\mathbb{B}$ and $\mathbb{N}$, while the type `Var` is interpreted by the game

$$\mathrm{Var} = \mathbb{C}^{\mathbb{N}} \times \mathbb{N} \, ,$$

where $\mathbb{C}^{\mathbb{N}}$ is the product of countably many copies of $\mathbb{C}$ (so we have used the symbol $\mathbb{N}$ to denote both the set $\mathbb{N}$ and the game $\mathbb{N}$ in the same formula).

Given types S, T, the denotation $[\![S \to T]\!]$ of the type of functions from S to T is given by

$$[\![S]\!] \to [\![T]\!] \coloneqq ![\![S]\!] \multimap [\![T]\!] \,.$$

This gives us the denotation of the types of Idealized Algol.

We inductively define a denotation of terms-in-context $\Gamma \vdash M$ of IA, where $[\![x_1 \colon T_1, \cdots, x_n \colon T_n \vdash M \colon T]\!]$ is a strategy

$$[\![T_1]\!] \times \cdots \times [\![T_n]\!] \to [\![T]\!] \,.$$

First note that we have natural innocent strategies $a \colon \mathbb{C}$, $\mathtt{t}, \mathtt{f} \colon \mathbb{B}$ and $n \colon \mathbb{N}$, which give us the denotations of $\Gamma \vdash \mathsf{skip}$, $\Gamma \vdash \mathtt{t}$, $\Gamma \vdash \mathtt{f}$ and $\Gamma \vdash n$.

Moreover, if we have a strategy

$$[\![\Gamma, x \colon S \vdash M \colon T]\!] \colon [\![\Gamma]\!] \times [\![S]\!] \to [\![T]\!] \,,$$

then, since $\mathcal{G}^!_{wo}$ is Cartesian closed, we get a strategy

$$[\![\Gamma \vdash \lambda x^s.M]\!] \colon [\![\Gamma]\!] \to [\![S]\!] \to [\![T]\!] \,.$$

In addition, we have natural morphisms

$$[\![\Gamma, x \colon T \vdash x \colon T]\!] = [\![\Gamma]\!] \times [\![T]\!] \xrightarrow{\mathrm{pr}_{[\![T]\!]}} [\![T]\!] \,.$$

Lastly, if we have strategies

$$[\![\Gamma \vdash M \colon S \to T]\!] \colon [\![\Gamma]\!] \to [\![S]\!] \to [\![T]\!] \qquad [\![\Gamma \vdash N \colon S]\!] \colon [\![\Gamma]\!] \to [\![S]\!] \,,$$

then we get a strategy

$$[\![\Gamma \vdash MN \colon T]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{[\![\Gamma \vdash M]\!] \times [\![\Gamma \vdash N]\!]} ([\![S]\!] \to [\![T]\!]) \times [\![S]\!] \xrightarrow{\mathrm{ev}} [\![S]\!] \,.$$

In order to form the denotation of the next lot of terms, we need a new definition.

Definition 2.15.1. Let X be a set, and let $(\sigma_x \,:\, x \in X)$ be a collection of strategies for a game A. Write X for the datatype game corresponding to X. Then we define a strict strategy $(\sigma_x) \colon X \to A$ by

$$(\sigma_x) = \{\epsilon\} \cup \{*q \,:\, *q \in P_A\} \cup \{*qys \,:\, *s \in \sigma_y\} \,.$$

In other words, after the initial move in A, (σ_x) requests some element $y \in X$, and thereafter plays according to σ_y in A.

Proposition 2.15.2. *If the σ_x are innocent strategies, then (σ_x) is an innocent strategy.*

Proof. If $*s \in \sigma_y$, then $\ulcorner *qysa \urcorner = *qy\ulcorner sa\urcorner$. Then, if $t \in (\sigma_x)$ and $\ulcorner ta\urcorner = *qy\ulcorner sa\urcorner$, we have $t = *qyt'$ for $t' \in \sigma_y$ and $\ulcorner *t'a\urcorner = \ulcorner *sa\urcorner$. So if $*sab \in \sigma_y$, then $*t'ab \in \sigma_y$ and therefore $tab \in (\sigma_x)$. □

The most important feature of strategies of the form (σ_x) is the most obvious one: given X, and $y \in X$, we have a strategy y for the game X with maximal play qy. Then $y;(\sigma_x) = \sigma_y$.

Now we define morphisms

- $\mathsf{seq}_X = (\mathrm{id}_X) \colon \mathbb{C} \multimap (X \to X)$;
- $\mathsf{If}_X = (\lambda x.\lambda y.x, \lambda x.\lambda y.y) \colon \mathbb{B} \multimap (X \to X \to X)$;
- $\mathsf{succ} = (1, 2, 3, 4, \cdots) \colon \mathbb{N} \multimap \mathbb{N}$;
- $\mathsf{pred} = (0, 0, 1, 2, \cdots) \colon \mathbb{N} \multimap \mathbb{N}$;
- $\mathsf{If0}_X = (\lambda x.\lambda y.x, \lambda x.\lambda y.y, \lambda x.\lambda y.y, \cdots) \colon \mathbb{N} \multimap (X \to X \to X)$;
- $\mathsf{assign} = (\mathrm{pr}_0, \mathrm{pr}_1, \cdots) \colon \mathbb{N} \multimap (\mathrm{Var} \to \mathbb{C})$;
- $\mathsf{deref} = \mathrm{pr}_{\mathbb{N}} \colon \mathrm{Var} \multimap \mathbb{N}$;
- $\mathsf{let}_{\mathbb{B},X} = (\lambda f.f\mathbb{t}, \lambda f.f\mathbb{f}) \colon \mathbb{B} \multimap (\mathbb{B} \to X) \to X$;
- $\mathsf{let}_{\mathbb{N},X} = (\lambda f.f0, \lambda f.f1, \cdots) \colon \mathbb{N} \multimap (\mathbb{N} \to X) \to X$; and
- $\mathsf{mkvar} = \lambda w.\lambda r.\langle (w\, n)_{n\in\mathbb{N}}, r\rangle \colon (\mathbb{N} \to \mathbb{C}) \to \mathbb{N} \multimap \mathrm{Var}$.

Here, pr_n and $\mathrm{pr}_{\mathbb{N}}$ are the natural projections on to $\mathbb{C}$ and $\mathbb{N}$ from $\mathtt{Var} = (\mathbb{C})^n \times \mathbb{N}$.

These morphisms give us an obvious way to interpret most of the rest of the terms of Idealized Algol. For example, if we have strategies

$$[\![\Gamma \vdash V \colon \mathrm{Var}]\!] \qquad\qquad [\![\Gamma \vdash E \colon \mathbb{N}]\!] \ ,$$

then we get a strategy

$$[\![\Gamma \vdash V \leftarrow E \colon \mathbb{C}]\!] = [\![\Gamma]\!] \xrightarrow{\Delta} [\![\Gamma]\!] \times [\![\Gamma]\!] \xrightarrow{[\![\Gamma\vdash E]\!]\times[\![\Gamma\vdash V]\!]} \mathbb{N} \times \mathrm{Var} \xrightarrow{\mathsf{assign}} \mathbb{C}$$

that we can use as the denotation of the term $\Gamma \vdash V \leftarrow E$.

We use seq_X for the denotation of $M;N$ and deref for the denotation of $!V$. The role played by the remaining morphisms can easily be deduced from their names.

2.16 Order-Enrichment of $\mathcal{G}$

The remaining parts of Idealized Algol that we have yet to define are the fixpoint combinator $\mathbf{Y}_T$ and the new variable constructor new.

To define $\mathbf{Y}_T$, we use order-enriched properties of $\mathcal{G}$.

Note that if A is a game, then we can order the strategies for A by subset inclusion. This order is clearly preserved by composition.

Proposition 2.16.1. *The partial order of strategies for A, ordered by inclusion, is directed-complete. So is the partial order of innocent strategies for A.*

Proof. Let Σ be a directed set of strategies for A; so if $\sigma, \tau \in \Sigma$ then there is some $\upsilon \in \Sigma$ such that $\sigma \subseteq \upsilon$ and $\tau \subseteq \upsilon$. We claim that $\bigcup \Sigma$ is a strategy for A. Indeed, it is certainly even-prefix-closed, and if $sab, sac \in \bigcup \Sigma$, then $sab \in \sigma$ and $sac \in \tau$ for $\sigma, \tau \in \Sigma$, and therefore $sab, sac \in \upsilon$ for some $\upsilon \in \Sigma$ and so $b = c$.

Now suppose that all $\sigma \in \Sigma$ are innocent. Let $sab \in \bigcup \Sigma$ and suppose that $t \in \bigcup \Sigma$ is such that $\ulcorner ta \urcorner = \ulcorner sa \urcorner$. Then, as before, we have $sab, t \in \upsilon$ for some innocent $\upsilon \in \Sigma$, and therefore $tab \in \upsilon \subseteq \bigcup \Sigma$. □

It is clear then that composition of strategies is Scott-continuous with respect to this ordering.

Writing $\bot = \{\epsilon\}$ for the bottom strategy for a game A, if we have a strategy $\sigma \colon A \to A$, then the Kleene fixed point theorem tells us that we may construct a fixed point for σ as the union of the chain

$$\bot \subseteq \bot; \sigma \subseteq \bot; \sigma; \sigma \subseteq \cdots .$$

Given a game A, we define a strategy $\mathbf{Y}_A \colon (A \to A) \to A$ as the fixed point of the strategy

$$\lambda F. \lambda f. f(Ff) \colon ((A \to A) \to A) \to (A \to A) \to A \,.$$

We can then use $\mathbf{Y}_A$ to interpret the term $\Gamma \vdash \mathbf{Y}_T M \colon T$ for any term $\Gamma \vdash M \colon T \to T$.

We will later require other order-theoretic properties of the set of strategies for a game A. Recall that an element σ of a directed-complete partially ordered set is called *compact* if whenever we have $\sigma = \bigcup \Sigma$ for some directed set Σ, then $\sigma \in \Sigma$.

A little thought convinces us that a strategy $\sigma \colon A$ is compact if and only if it is finite as a set of plays; indeed, suppose σ is a finite set and $\sigma = \bigcup \Sigma$.

For each $s \in \sigma$, we have $s \in \tau_s$ for some $\tau_s \in \Sigma$; since Σ is directed, then there is some $\upsilon \in \Sigma$ such that $\tau_s \in \Sigma$ for each s, and therefore $\sigma \subseteq \upsilon \subseteq \sigma$. Conversely, if σ is infinite, then by König's lemma, it either has an infinite branching point (i.e., $s \in \sigma$ such that there are infinitely many plays $sab \in \sigma$) or an infinite branch (i.e., an infinite increasing sequence $s_1 \sqsubseteq s_2 \sqsubseteq \cdots$ in σ). In either case, it is easy to construct some directed set Σ such that $\sigma = \bigcup \Sigma$ but $\sigma \notin \Sigma$.

Recall that a directed-complete partial order P is said to be *algebraic* if whenever $p \in P$, the set of compact elements of P lying below p is directed and its supremum is p.

Proposition 2.16.2. *The set of strategies for a game A is an algebraic directed-complete partial order.*

Proof. Let σ be a strategy for a game A and let τ_1, τ_2 be two finite substrategies such that $\tau_1, \tau_2 \subseteq \sigma$. Then $\tau_1 \cup \tau_2 \subseteq \sigma$ and is finite; moreover, if $sab, sac \in \tau_1 \cup \tau_2$, then $sab, sac \in \sigma$, so $b = c$.

Lastly, given any $s \in \sigma$, there is a compact strategy σ_s containing s; namely

$$\sigma_s = \{t : t \sqsubseteq s \text{ has even length}\}. \qquad \square$$

It is possible to prove an innocent version of this (see [HO00]), but we will not need this to prove Full Abstraction for Idealized Algol.

2.17 The Strategy cell

Now we come to the denotation of new. For this, we shall define a strategy $\mathsf{cell} \colon !\mathbb{N} \multimap !\,\mathrm{Var}$ by using the universal property of $!\,\mathrm{Var}$ that arises from the fact that it is a final coalgebra for $\mathtt{Var} \oslash \_$.

Given $n \in \mathbb{N}$, we define a strategy $\mathrm{write}_n \colon !\mathbb{N} \multimap \mathbb{C} \oslash !\mathbb{N}$ by

$$\mathrm{write}_n = !\mathbb{N} \xrightarrow{()} I \xrightarrow{!n} !\mathbb{N} \xrightarrow{\mathrm{lunit}_{!\mathbb{N}}} I \otimes !\mathbb{N} \xrightarrow{\mathsf{skip}\otimes !\mathbb{N}} \mathbb{C} \otimes !\mathbb{N} \xrightarrow{\mathrm{wk}} \mathbb{C} \oslash !\mathbb{N}.$$

Let $\mathrm{read} \colon !\mathbb{N} \multimap \mathbb{N} \oslash !\mathbb{N}$ be the morphism part α of the limiting coalgebra. In other words, by Proposition 2.14.5, read is the composite

$$\mathrm{read} = !\mathbb{N} \xrightarrow{\mu_{\mathbb{N}}} !\mathbb{N} \otimes !\mathbb{N} \xrightarrow{\mathsf{der}_{\mathbb{N}}\otimes !\mathbb{N}} \mathbb{N} \otimes !\mathbb{N} \xrightarrow{\mathrm{wk}_{\mathbb{N},!\mathbb{N}}} \mathbb{N} \oslash !\mathbb{N}.$$

Then we get a coalgebra $\mathsf{cell}_0 \colon !\mathbb{N} \multimap \mathrm{Var} \oslash !\mathbb{N}$ given by

$$!\mathbb{N} \xrightarrow{\langle(\mathrm{write}_n)_{n\in\mathbb{N}},\mathrm{read}\rangle} (\mathbb{C}\oslash !\mathbb{N})^{\mathbb{N}} \times (\mathbb{N}\oslash !\mathbb{N}) \xrightarrow{\mathsf{dist}^{-1}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}} (\mathbb{C}^{\mathbb{N}} \times \mathbb{N}) \oslash !\mathbb{N} = \mathrm{Var} \oslash !\mathbb{N}.$$

We then define the strategy cell to be the anamorphism $(\!|\, \mathsf{cell}_0 \,|\!)\colon !\mathbb{N} \multimap !\,\mathrm{Var}$; i.e., cell is the unique morphism making the following diagram commute.

$$\begin{array}{ccc} !\mathbb{N} & \xrightarrow{\mathsf{cell}_0} & \mathrm{Var} \oslash !\mathbb{N} \\ {\scriptstyle \mathsf{cell}}\downarrow & & \downarrow{\scriptstyle \mathrm{Var} \oslash \mathsf{cell}} \\ !\,\mathrm{Var} & \xrightarrow{\alpha_{\mathrm{Var}}} & \mathrm{Var} \oslash !\,\mathrm{Var} \end{array}$$

Concretely, the strategy cell behaves as follows, as we shall prove in Proposition 2.17.2. When player O plays in $!\mathtt{Var}$, he chooses to play either in one of the copies of $\mathbb{C}$ or in $\mathbb{N}$. If he plays the initial move q_n in the n-th copy of $\mathbb{C}$, player P updates the value she has stored in her head to n. If he plays the initial move q in $\mathbb{N}$, then player P replies with this stored value. Lastly, if he plays this initial move q without having played in any of the copies of $\mathbb{C}$, then player P interrogates the argument in order to find out which value to play.

This strategy cell now gives us a morphism $\mathsf{new}_A\colon !(!\,\mathrm{Var} \multimap A) \multimap A$ (for A well-opened), given by

$$\begin{aligned} !(!\,\mathrm{Var} \multimap A) &\xrightarrow{\mathtt{der}} (!\,\mathrm{Var} \multimap A) \\ &\xrightarrow{\mathrm{lunit}} I \otimes (!\,\mathrm{Var} \multimap A) \\ &\xrightarrow{!0 \otimes (!\,\mathrm{Var} \multimap A)} !\mathbb{N} \otimes (!\,\mathrm{Var} \multimap A) \\ &\xrightarrow{\mathsf{cell} \otimes (!\,\mathrm{Var} \multimap A)} !\,\mathrm{Var} \otimes (!\,\mathrm{Var} \multimap A) \\ &\xrightarrow{\mathrm{ev}} A\,. \end{aligned}$$

We use this to provide the denotation of the term new.

Lemma 2.17.1. $\mathsf{cell}_0; (\mathrm{pr}_n \oslash !\mathbb{N}) = \mathrm{write}_n$ *for each* n *and* $\mathsf{cell}_0; (\mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N}) = \mathrm{read}$*, where* $\mathrm{pr}_n\colon \mathrm{Var} = \mathbb{C}^{\mathbb{N}} \times \mathbb{N} \multimap \mathbb{C}$ *is the projection on to the* n*-th copy of* $\mathbb{C}$ *and* $\mathrm{pr}_{\mathbb{N}}\colon \mathrm{Var} = \mathbb{C}^{\mathbb{N}} \times \mathbb{N} \multimap \mathbb{N}$ *is the projection on to the copy of* $\mathbb{N}$*.*

Proof. We have

$$\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; \mathrm{pr}_n = \langle (\mathrm{pr}_n \oslash !\mathbb{N})_{n\in\mathbb{N}}, \mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N} \rangle; \mathrm{pr}_n = \mathrm{pr}_n \oslash !\mathbb{N}$$

and

$$\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; \mathrm{pr}_{\mathbb{N}} = \langle (\mathrm{pr}_n \oslash !\mathbb{N})_{n\in\mathbb{N}}, \mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N} \rangle; \mathrm{pr}_{\mathbb{N}} = \mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N}\,,$$

so

$$\begin{aligned} &\mathsf{cell}_0; (\mathrm{pr}_n \oslash !\mathbb{N}) \\ &= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read} \rangle; \mathsf{dist}^{-1}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; (\mathrm{pr}_n \oslash !\mathbb{N}) \\ &= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read} \rangle; \mathsf{dist}^{-1}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}\, \mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; \mathrm{pr}_n \\ &= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read} \rangle; \mathrm{pr}_n \\ &= \mathrm{write}_n \end{aligned}$$

and

$$\begin{aligned}
&\mathsf{cell}_0; (\mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N}) \\
&= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read}\rangle; \mathsf{dist}^{-1}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; (\mathrm{pr}_{\mathbb{N}} \oslash !\mathbb{N}) \\
&= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read}\rangle; \mathsf{dist}^{-1}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}\, \mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N},!\mathbb{N}}; \mathrm{pr}_{\mathbb{N}} \\
&= \langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read}\rangle; \mathrm{pr}_{\mathbb{N}} \\
&= \mathrm{read}\ . \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \square
\end{aligned}$$

While this coalgebraic definition of the cell strategy will be sufficient for most of our purposes, it will also be convenient to have a more direct definition.

Proposition 2.17.2. *The strategy* cell *is the strategy on* $!\mathbb{N} \multimap !\,\mathrm{Var}$ *that behaves as follows. It replies to a move* q_n *(i.e., the initial move in the* n*-th component of* $\mathbb{C}$ *in* Var*) with the unique answer* a*, and replies to the move* q *in the* $\mathbb{N}$ *component of* Var *with that number* n *such that* q_n *was the most recently played O-move in the* $\mathbb{C}^{\mathbb{N}}$ *component, or, if no such move has been played, it interrogates* $!\mathbb{N}$ *on the left and copies the value back on to the right.*

Proof. It will suffice to show that the strategy cell as described in the statement makes the diagram

$$\begin{array}{ccc}
!\mathbb{N} & \xrightarrow{\mathsf{cell}_0} & \mathrm{Var} \oslash !\mathbb{N} \\
{\scriptstyle \mathsf{cell}}\downarrow & & \downarrow{\scriptstyle \mathrm{Var}\oslash\mathsf{cell}} \\
!\,\mathrm{Var} & \xrightarrow{\alpha_{\mathrm{Var}}} & \mathrm{Var} \oslash !\,\mathrm{Var}
\end{array}$$

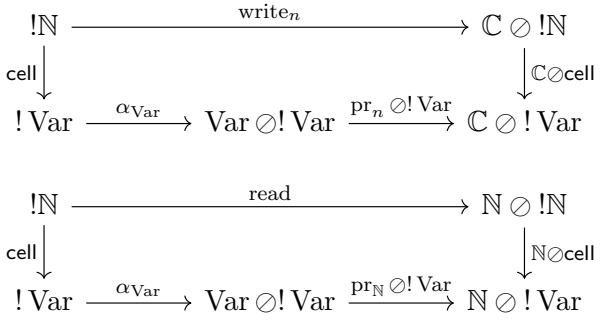
commute. Since $\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N}}$ is a natural isomorphism, it will suffice to show commutativity of the diagram formed by composing by $\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N}}$ on the right to give us the diagram

$$\begin{array}{ccccc}
!\mathbb{N} & \xrightarrow{\mathsf{cell}_0} & \mathrm{Var} \oslash !\mathbb{N} & \xrightarrow{\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N}}} & (\mathbb{C} \oslash !\mathbb{N})^{\mathbb{N}} \times (!\mathbb{N} \oslash !\mathbb{N}) \\
{\scriptstyle \mathsf{cell}}\downarrow & & & & \downarrow{\scriptstyle (\mathbb{C}\oslash\mathsf{cell})^{\mathbb{N}}\times(\mathbb{N}\oslash\mathsf{cell})} \\
!\,\mathrm{Var} & \xrightarrow{\alpha_{\mathrm{Var}}} & \mathrm{Var} \oslash !\,\mathrm{Var} & \xrightarrow{\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N}}} & (\mathbb{C} \oslash !\,\mathrm{Var})^{\mathbb{N}} \times (\mathbb{N} \oslash !\,\mathrm{Var})\,,
\end{array}$$

which, by the definition of cell_0, is the same as the diagram

$$\begin{array}{ccccc}
!\mathbb{N} & \xrightarrow{\langle (\mathrm{write}_n)_{n\in\mathbb{N}}, \mathrm{read}\rangle} & & & (\mathbb{C} \oslash !\mathbb{N})^{\mathbb{N}} \times (!\mathbb{N} \oslash !\mathbb{N}) \\
{\scriptstyle \mathsf{cell}}\downarrow & & & & \downarrow{\scriptstyle (\mathbb{C}\oslash\mathsf{cell})^{\mathbb{N}}\times(\mathbb{N}\oslash\mathsf{cell})} \\
!\,\mathrm{Var} & \xrightarrow{\alpha_{\mathrm{Var}}} & \mathrm{Var} \oslash !\,\mathrm{Var} & \xrightarrow{\mathsf{dist}_{(\mathbb{C})_{\mathbb{N}},\mathbb{N}}} & (\mathbb{C} \oslash !\,\mathrm{Var})^{\mathbb{N}} \times (\mathbb{N} \oslash !\,\mathrm{Var})\,.
\end{array}$$

That is, we need to show that the following diagrams commute, where n in the first diagram ranges over the natural numbers.



Consider the first diagram. We may identify plays of $\mathsf{cell}; \alpha_{\mathrm{Var}}; (\mathrm{pr}_n \oslash ! \mathrm{Var})$ with those plays in cell that begin with the initial move q_n in Var (i.e., the initial move q in the n-th copy of $\mathbb{C}$). We need to show that plays in $\mathrm{write}_n; (\mathbb{C} \oslash \mathsf{cell})$ take the same form. Indeed, the response to the initial move in $\mathbb{C}$ in $\mathrm{write}_n; (\mathbb{C} \oslash \mathsf{cell})$ is the unique response a, by the definition of write_n; thereafter, play continues in $! \mathrm{Var}$ according to cell – so the response to a move q_m will be a, while the response to q, if some q_m has been played already in $! \mathrm{Var}$, will be the most recently occurring value of m.

If no q_m has been played in $! \mathrm{Var}$, then the behaviour of cell tells us that we must interrogate the $!\mathbb{N}$ in $\mathbb{C} \oslash !\mathbb{N}$ in order to get the value. By the definition of write_n, the value returned will be n, mirroring the behaviour in $\mathsf{cell}; \alpha_{\mathrm{Var}}; (\mathrm{pr}_n \oslash ! \mathrm{Var})$.

Now consider the second diagram. We may identify plays of the strategy $\mathsf{cell}; \alpha_{\mathrm{Var}}; (\mathrm{pr}_n \oslash ! \mathrm{Var})$ with those plays in cell that begin with the initial move q in Var (i.e., the initial move q in the copy of $\mathbb{N}$). We need to show that plays in $\mathrm{read}; (\mathbb{N} \oslash \mathsf{cell})$ take the same form. Indeed, the response to the initial move will be to interrogate $!\mathbb{N}$ on the left to get the value. Thereafter, play will continue between $!\mathbb{N}$ and $! \mathrm{Var}$ according to the cell strategy – so once again the response to a move q_m will be a, while the response to q, if some q_m has been played, will be the most recently occurring value m.

If no q_m has been played in $! \mathrm{Var}$, then, by the definition of read, the strategy will interrogate $!\mathbb{N}$ in order to get its response, just as in the strategy $\mathsf{cell}; \alpha_{\mathrm{Var}}; (\mathrm{pr}_{\mathbb{N}} \oslash ! \mathrm{Var})$. □

Chapter 3

Operational Semantics, Computational Adequacy and Full Abstraction

3.1 Big-Step Operational Semantics

We now introduce the operational semantics of Idealized Algol. When we have done so, we will link it to our denotational semantics by proving Computational Adequacy and Full Abstraction.

We first define a *canonical form* of the language to be

- at type `com`, the term skip;
- at type `bool`, the terms $\mathbb{t}$ and $\mathbb{f}$;
- at type `nat`, the numerals n;
- at type `Var`, variable names x: `Var` and expressions taking the form $\mathsf{mkvar}\, W\, R$; and
- at type $S \to T$, expressions of the form $\lambda x^S.M$.

These are the terms that cannot be evaluated any further.

We define a `Var`-*context* to be a context Γ of the form x_1: `Var`, $\cdots$, x_n: `Var`. Given a `Var`-context Γ, we define a Γ-*store* to be a function s from the set of variable names occurring in Γ (i.e., the x_i) to the set of natural numbers. Given such a store s, we write $(s|x \mapsto n)$ for the store given by

$$(s|x \mapsto n)(y) = \begin{cases} n & \text{if } y = x \\ s(y) & \text{otherwise} \end{cases}.$$

We now inductively define a relation $\Gamma, s \vdash M \Downarrow c, s'$, where

- Γ is a `Var`-context;
- s and s' are Γ-stores; and
- $\Gamma \vdash M$, $\Gamma \vdash c$ are Idealized Algol terms-in-context, where c is a canonical form.

The definition of this relation is shown in Figure 3.1. The idea behind it is that the predicate $\Gamma, s \vdash M \Downarrow c, s'$ should represent the assertion that, given initial state s, the term M will evaluate to the canonical form c, leaving the state s'.

3.2 Small-Step Operational Semantics

We also give an equivalent small-step operational semantics for Idealized Algol. For a lot of our purposes, the small-step semantics will be easier to work with.

This time, instead of defining a relation $\Gamma, s \vdash M \Downarrow c, s'$, we define a relation $\Gamma, s \vdash M \longrightarrow \Gamma, \Delta, s' \vdash M'$, where

- Γ, Δ are disjoint `Var`-contexts;
- s is a Γ-store and s' a Γ, Δ-store; and
- $\Gamma \vdash M$, $\Gamma, \Delta \vdash M'$ are Idealized Algol terms-in-context.

As an auxiliary definition, we need the notion of an *evaluation context*, as introduced by Felleisen in [FH92]. An evaluation context is a single-holed context defined inductively by the following BNF formula, where M ranges over IA terms (subject to typing rules).

$$\mathbf{E} ::= - \mid \mathbf{E}\, M \mid \mathsf{succ}\, \mathbf{E} \mid \mathsf{pred}\, \mathbf{E} \mid \mathbf{E}; M \mid \mathsf{If}\ \mathbf{E}\ \mathsf{then}\ M\ \mathsf{else}\ M \mid$$

$$\mathsf{If0}\ \mathbf{E}\ \mathsf{then}\ M\ \mathsf{else}\ M \mid !\mathbf{E} \mid M \leftarrow \mathbf{E} \mid \mathbf{E} \leftarrow n \mid \mathsf{let}\, x = \mathbf{E}\ \mathsf{in}\ M$$

The idea behind the evaluation contexts is that they tell us about the order of evaluation of a term: if we want to evaluate the term $\mathbf{E}[M]$ for one step, then we first evaluate M for one step, yielding another term M', and then replace $\mathbf{E}[M]$ with $\mathbf{E}[M']$.

Next, we define a relation $\Gamma, s, M \dashrightarrow \Gamma, \Delta, s', M'$. The definition of this relation is in Figure 3.2; we interpret the judgement $\Gamma, s, M \dashrightarrow \Gamma, \Delta, s', M'$ as saying that if we start in state s and evaluate the term M for a single step, then we get the term M' in the state s', possibly adding some new free variables through the `Var`-context Δ.

$$\frac{}{\Gamma, s \vdash c \Downarrow c, s} \qquad \frac{\Gamma, s \vdash M \Downarrow \lambda x.M', s' \qquad \Gamma, s' \vdash M'[N/x] \Downarrow c, s''}{\Gamma, s \vdash MN \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M(\mathbf{Y}M) \Downarrow c, s'}{\Gamma, s \vdash \mathbf{Y}M \Downarrow c, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow n, s'}{\Gamma, s \vdash \mathsf{succ}\, M \Downarrow n+1, s'}$$

$$\frac{\Gamma, s \vdash M \Downarrow n+1, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow n, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow 0, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow 0, s'}$$

$$\frac{\Gamma, s \vdash M \Downarrow \mathsf{skip}, s' \qquad \Gamma, s' \vdash N \Downarrow c, s''}{\Gamma, s \vdash M; N \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow \mathtt{t}, s' \qquad \Gamma, s' \vdash N \Downarrow c, s''}{\Gamma, s \vdash \mathsf{If}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow \mathtt{f}, s' \qquad \Gamma, s' \vdash P \Downarrow c, s''}{\Gamma, s \vdash \mathsf{If}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow 0, s' \qquad \Gamma, s' \vdash N \Downarrow c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow n+1, s' \qquad \Gamma, s' \vdash P \Downarrow c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow c, s' \qquad \Gamma, s' \vdash N[c/x] \Downarrow c', s''}{\Gamma, s \vdash \mathsf{let}\, x = M\ \mathsf{in}\ N \Downarrow c', s''}$$

$$\frac{\Gamma, s \vdash E \Downarrow n, s' \qquad \Gamma, s' \vdash V \Downarrow x, s''}{\Gamma, s \vdash V \leftarrow E \Downarrow \mathsf{skip}, (s''|x \mapsto n)}\ x \in \Gamma \qquad \frac{\Gamma, s \vdash V \Downarrow x, s'}{\Gamma, s \vdash !V \Downarrow n, s'}\ s'(x) = n$$

$$\frac{\Gamma, x\colon \mathtt{Var}, (s|x \mapsto 0) \vdash M \Downarrow c, (s'|x \mapsto n)}{\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow c, s'}$$

$$\frac{\Gamma, s \vdash E \Downarrow n, s' \qquad \Gamma, s' \vdash V \Downarrow \mathsf{mkvar}\, WR, s'' \qquad \Gamma, s'' \vdash Wn \Downarrow \mathsf{skip}, s'''}{\Gamma, s \vdash V \leftarrow E \Downarrow \mathsf{skip}, s'''}$$

$$\frac{\Gamma, s \vdash V \Downarrow \mathsf{mkvar}\, WR, s' \qquad \Gamma, s' \vdash R \Downarrow n, s''}{\Gamma, s \vdash !V \Downarrow n, s''}$$

Figure 3.1: Operational semantics for Idealized Algol. See [Har99] and [AM96].

$$\Gamma, s \vdash (\lambda x.M)N \dashrightarrow \Gamma, s \vdash M[N/x] \qquad \Gamma, s \vdash \mathbf{Y}M \dashrightarrow \Gamma, s \vdash M(\mathbf{Y}M)$$

$$\Gamma, s \vdash \mathsf{succ}\, n \dashrightarrow \Gamma, s \vdash n + 1$$

$$\Gamma, s \vdash \mathsf{pred}(n+1) \dashrightarrow \Gamma, s \vdash n \qquad \Gamma, s \vdash \mathsf{pred}\, 0 \dashrightarrow \Gamma, s \vdash 0$$

$$\Gamma, s \vdash \mathsf{skip}; M \dashrightarrow \Gamma, s \vdash M$$

$$\Gamma, s \vdash \mathsf{If}\ \mathbb{t}\ \mathsf{then}\ N\ \mathsf{else}\ P \dashrightarrow \Gamma, s \vdash N \qquad \Gamma, s \vdash \mathsf{If}\ \mathbb{f}\ \mathsf{then}\ N\ \mathsf{else}\ P \dashrightarrow \Gamma, s \vdash P$$

$$\Gamma, s \vdash \mathsf{If0}\, 0\ \mathsf{then}\ N\ \mathsf{else}\ P \dashrightarrow \Gamma, s \vdash N$$

$$\Gamma, s \vdash \mathsf{If0}(n+1)\ \mathsf{then}\ N\ \mathsf{else}\ P \dashrightarrow \Gamma, s \vdash P$$

$$x, \Gamma, s \vdash x \leftarrow n \dashrightarrow x, \Gamma, (s|x \mapsto n) \vdash \mathsf{skip} \qquad x, \Gamma, s \vdash !x \dashrightarrow x, \Gamma, s \vdash s(x)$$

$$\Gamma, s \vdash \mathsf{new}\, \lambda x.M \dashrightarrow \Gamma, x, (s|x \mapsto 0) \vdash M$$

$$\Gamma, s \vdash (\mathsf{mkvar}\, W\, R) \leftarrow n \dashrightarrow \Gamma, s \vdash Wn \qquad \Gamma, s \vdash !(\mathsf{mkvar}\, W\, R) \dashrightarrow \Gamma, s \vdash R$$

$$\Gamma, s \vdash \mathsf{let}\, x = \mathbb{t}\ \mathsf{in}\ M \dashrightarrow \Gamma, s \vdash M[\mathbb{t}/x]$$

$$\Gamma, s \vdash \mathsf{let}\, x = \mathbb{f}\ \mathsf{in}\ M \dashrightarrow \Gamma, s \vdash M[\mathbb{f}/x]$$

$$\Gamma, s \vdash \mathsf{let}\, x = n\ \mathsf{in}\ M \dashrightarrow \Gamma, s \vdash M[n/x]$$

Figure 3.2: Felleisen-style small-step operational semantics for Idealized Algol.

Lastly, we define the relation $\longrightarrow$ as

$$\frac{\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash M'}{\Gamma, s \vdash \mathbf{E}[M] \longrightarrow \Gamma, \Delta, s' \vdash \mathbf{E}[M']}$$

for each evaluation context $\mathbf{E}$ whose free variables lie in Γ.

We need to prove that this is equivalent to our original semantics.

Given a Γ, Δ-store s, write $s|_\Gamma$ for the restriction of s to Γ.

Lemma 3.2.1. *Suppose that* $\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash M'$ *and that* $\Gamma, \Delta, s' \vdash \mathbf{E}[M'] \Downarrow c, s''$. *Then* $\Gamma, s \vdash \mathbf{E}[M] \Downarrow c, s''|_\Gamma$.

Proof. Structural induction on $\mathbf{E}$. The base case, when $\mathbf{E} = -$, covers the interesting cases, so we shall leave it to last. The remaining cases are quite similar, so we will show the proof for the case where $\mathbf{E} = \mathbf{E}'\, N$ for illustration.

If $\mathbf{E} = \mathbf{E}'\,N$, for some term N, then we have $\Gamma, \Delta, s' \vdash \mathbf{E}'[M']\,N \Downarrow c, s''$. By inspection of the rules in Figure 3.1, the derivation of this must end with a rule of the form

$$\frac{\Gamma, \Delta, s' \vdash \mathbf{E}'[M'] \Downarrow \lambda x.M'', t \qquad \Gamma, \Delta, t \vdash M''[N/x] \Downarrow c, s''}{\Gamma, \Delta, s' \vdash \mathbf{E}'[M']\,N \Downarrow c, s''} \quad .$$

Thus, $\Gamma, \Delta, s' \vdash \mathbf{E}'[M'] \Downarrow \lambda x.M'', t$ and $\Gamma, \Delta \vdash M''[N/x] \Downarrow c, s''$ must be provable for some M'', t. By the induction hypothesis, this means that $\Gamma, s \vdash \mathbf{E}'[M] \Downarrow \lambda x.M'', t$ is provable. Then we have a derivation

$$\frac{\Gamma, \Delta, s \vdash \mathbf{E}'[M] \Downarrow \lambda x.M'', t \qquad \Gamma, \Delta, t \vdash M''[N/x] \Downarrow c, s''}{\Gamma, \Delta, s \vdash \mathbf{E}'[M]\,N \Downarrow c, s''|_{\Gamma,\Delta}} \quad .$$

Then, because any variables in Δ are not mentioned in $\mathbf{E}'[M]\,N$ or in c, we have

$$\Gamma, s \vdash \mathbf{E}'[M]\,N \Downarrow c, s''|_{\Gamma}\,.$$

The remaining cases when $\mathbf{E} \neq -$ are quite similar. Now, let us suppose that $\mathbf{E} = -$, so that $\mathbf{E}[M'] = M'$.

Then there are a number of cases, depending on the particular $\dashrightarrow$ rule we are using. Many of these cases are similar, so we will cover a few of them for the purposes of illustration.

- For function application, suppose that $\Gamma, s \vdash M[N/x] \Downarrow c, s'$. Then we have a derivation

$$\frac{\overline{\Gamma, s \vdash \lambda x.M \Downarrow \lambda x.M, s} \qquad \Gamma, s \vdash M[N/x] \Downarrow c, s'}{\Gamma, s \vdash (\lambda x.M)N \Downarrow c, s'} \quad .$$

- For variable assignment, we have a derivation

$$\frac{\overline{\Gamma, s \vdash x \Downarrow x, s} \qquad \overline{\Gamma, s \vdash n \Downarrow n, s}}{\Gamma, s \vdash x \leftarrow n \Downarrow \mathsf{skip}, (s|x \mapsto n)} \quad .$$

- For variable dereference, we have a derivation

$$\frac{\overline{\Gamma, s \vdash x \Downarrow x, s'}}{\Gamma, s \vdash\, !x \Downarrow s(x), s} \quad .$$

- For new, suppose that $\Gamma, x, (s|x \mapsto 0) \vdash M \Downarrow c, s'$. Then we have a derivation

$$\frac{\Gamma, x, (s|x \mapsto 0) \vdash M \Downarrow c, s'}{\Gamma, s \vdash \mathsf{new}\ \lambda x.M \Downarrow c, s'|_{\Gamma}} \quad ,$$

since $s' = (s'|_{\Gamma}|x \mapsto s'(x))$. □

We have proved:

Proposition 3.2.2. *Suppose that we have a sequence*

$$\Gamma_1, s^{(1)}, M_1 \longrightarrow \cdots \longrightarrow \Gamma_n, s^{(n)}, M_n\,,$$

where M_n is a canonical form. Then $\Gamma_1, s^{(1)} \vdash M_1 \Downarrow M_n, s^{(n)}|_{\Gamma_1}$.

Proof. Induction on n. The inductive step is Lemma 3.2.1, while the base case ($n = 1$) is given by the derivation

$$\frac{}{\Gamma, s \vdash c \Downarrow c, s}\ .$$ □

We can also prove the converse.

Proposition 3.2.3. *Suppose that $\Gamma, s \vdash M \Downarrow c, s'$. Then there are sequences $\Gamma = \Gamma_1, \cdots, \Gamma_n = \Gamma, \Delta$ of $\mathtt{Var}$-contexts, $s = s^{(1)}, \cdots, s^{(n)}$ of Γ_n-stores and $M = M_1, \cdots, M_n = c$ of terms such that*

$$\Gamma_1, s^{(1)} \vdash M_1 \longrightarrow \cdots \longrightarrow \Gamma_n, s^{(n)} \vdash M_n\,,$$

and $s^{(n)}|_\Gamma = s'$.

Proof. Induction on the derivation of $\Gamma, s \vdash M \Downarrow c, s'$. Since most of the cases are similar, we cover a selection for illustration.

- Suppose that the last step in the derivation is

  $$\frac{}{\Gamma, s \vdash c \Downarrow c, s}\ .$$

  Then we have the one-element sequence $\Gamma, s \vdash c$.

- Suppose that the last step in the derivation is

  $$\frac{\Gamma, s \vdash M \Downarrow \lambda x.M', s' \qquad \Gamma, s' \vdash M'[N/x] \Downarrow c, s''}{\Gamma, s \vdash MN \Downarrow c, s''}\ .$$

  Then, by the inductive hypothesis, we have small-step derivations

  $$\Gamma, s \vdash M \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t', \lambda x.M'$$

  $$\Gamma, s' \vdash M'[N/x] \longrightarrow \cdots \longrightarrow \Gamma, \Delta', t'', c\,,$$

  where $t'|_\Gamma = s'$ and $t''|_\Gamma = s''$. Note that we can easily enlarge Γ to make sure that it contains all the free variables in N.

  If we apply the evaluation context $-N$ pointwise to the first small-step derivation, then we have another valid small-step derivation

  $$\Gamma, s \vdash M\,N \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t', \lambda x.M'\,N\,.$$

  Then we can join the two together to get the derivation

$$\Gamma, s \vdash MN \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t', (\lambda x.M')N \longrightarrow$$

$$\Gamma, \Delta, t', M'[N/x] \longrightarrow \cdots \longrightarrow \Gamma, \Delta \cup \Delta', t'' \setminus t', c\,,$$

where $t'' \setminus t'$ is the $\Gamma, \Delta \cup \Delta'$-store that agrees with t'' on Γ, Δ' and with t' on $\Delta \setminus \Delta'$. Then $(t'' \setminus t')|_\Gamma = s''$.

- Suppose that the last step in the derivation is

$$\frac{\Gamma, s \vdash E \Downarrow n, s' \qquad \Gamma, s' \vdash V \Downarrow x, s''}{\Gamma, s \vdash V \leftarrow E \Downarrow \mathsf{skip}, (s''|x \mapsto n)}\quad .$$

By the inductive hypothesis, we have small-step derivations

$$\Gamma, s \vdash E \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t', n \qquad \Gamma, s' \vdash V \longrightarrow \cdots \longrightarrow \Gamma, \Delta', t'', x\,,$$

where $t'|_\Gamma = s'$ and $t''|_\Gamma = s''$.

We may apply the evaluation context $V \leftarrow -$ pointwise to the first derivation and the evaluation context $- \leftarrow n$ pointwise to the second, and then string the two together to get

$$\Gamma, s \vdash V \leftarrow E \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t', V \leftarrow n \longrightarrow \cdots \longrightarrow$$

$$\Gamma, \Delta \cup \Delta', t'' \setminus t', x \leftarrow n \longrightarrow \Gamma, \Delta \cup \Delta', (t'' \setminus t'|x \mapsto n)\,,$$

where we have $(t'' \setminus t'|x \mapsto n)|_\Gamma = (s''|x \mapsto n)$.

- Suppose that the last step in the derivation is

$$\frac{\Gamma, s \vdash V \Downarrow x, s'}{\Gamma, s \vdash !V \Downarrow s'(x), s'}\quad .$$

Then, by the induction hypothesis, we have a small-step derivation

$$\Gamma, s \vdash V \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t' \vdash x\,,$$

where $t'|_\Gamma = s'$. Then we may compose this derivation pointwise with the evaluation context $!-$, and add an extra term on the end, to arrive at the derivation

$$\Gamma, s \vdash !V \longrightarrow \cdots \longrightarrow \Gamma, \Delta, t' \vdash !x \longrightarrow \Gamma, \Delta, t' \vdash t'(x)\,,$$

where $t'(x) = s'(x)$.

- Suppose that the last step in the derivation is

$$\frac{\Gamma, x, (s|x \mapsto 0) \vdash M \Downarrow c, (s'|x \mapsto n)}{\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow c, s'} \quad .$$

  By the induction hypothesis, we have a small-step derivation

$$\Gamma, x, (s|x \mapsto 0) \vdash M \longrightarrow \cdots \longrightarrow \Gamma, \Delta, x, (t'|x \mapsto n), c\,,$$

  where $t'|_\Gamma = s'$. Then we may add a term at the beginning to give us

$$\Gamma, s \vdash \mathsf{new}\, \lambda x.M \longrightarrow \Gamma, x, (s|x \mapsto 0) \vdash M \longrightarrow$$

$$\cdots \longrightarrow \Gamma, \Delta, x, (t'|x \mapsto n), c\,,$$

  where $(t'|x \mapsto n)|_\Gamma = s'$. □

3.3 Soundness

To prove soundness of our model, we shall use the small-step formulation. Our reason for this is that the most difficult part of the denotational semantics we are using is the part to do with state. In the big-step formulation, nearly every rule involves the state changing in some way, whereas in the small-step formulation, only the rules that specifically pertain to the stateful components of the language do.

Given a `Var`-context Γ, we will write S_Γ for $\mathbb{N} \times \cdots \times \mathbb{N}$.

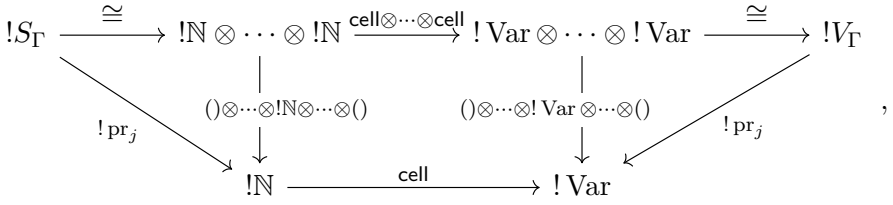
Given Γ, we have a morphism $\mathsf{cell}^\Gamma \colon !S_\Gamma \multimap ! [\![\Gamma]\!]$, given by

$$!S_\Gamma \cong !\mathbb{N} \otimes \cdots \otimes !\mathbb{N} \xrightarrow{\mathsf{cell}\otimes\cdots\otimes\mathsf{cell}} ! \,\mathrm{Var} \otimes \cdots \otimes ! \,\mathrm{Var} \cong ! [\![\Gamma]\!] \,.$$

Lemma 3.3.1. *For $j = 1, \cdots, |\Gamma|$, the following diagram commutes.*

$$\begin{array}{ccc} !S_\Gamma & \xrightarrow{\mathsf{cell}^\Gamma} & !V_\Gamma \\ {\scriptstyle !\,\mathrm{pr}_j}\downarrow & & \downarrow{\scriptstyle !\,\mathrm{pr}_j} \\ !\mathbb{N} & \xrightarrow{\mathsf{cell}} & ! \,\mathrm{Var} \end{array}$$

Proof. We have a commutative diagram



,

where the outer triangles commute because the vertical arrows are the projections in the tensor product of comonoids. □

Given a Γ-store s, we will write $[\![s]\!]$ for the corresponding morphism $I \to S_\Gamma$.

We start our proof of soundness with a structural result about evaluation contexts. Recall that the most important property of an evaluation context $\mathbf{E}$ is that if a term M fits into the hole in $\mathbf{E}$, then M is the first part of $\mathbf{E}[M]$ to be evaluated. Using the sequoid operator, we can capture in category-theoretic terms an analogous property for the denotation of $\mathbf{E}$.

Lemma 3.3.2. *Let $\Gamma \vdash M : T$ be an Idealized Algol term-in-context (where Γ is an arbitrary context), and let $\mathbf{E}$ be an evaluation context with a hole of type T, where $\mathbf{E}[M] : U$. Then there are (independent of M) a game A, a strategy $\sigma : !\llbracket\Gamma\rrbracket \multimap A$ and a strict strategy $\tau : \llbracket T\rrbracket \oslash A \multimap \llbracket S\rrbracket$, such that the denotation $\llbracket\Gamma \vdash \mathbf{E}[M]\rrbracket$ factors as*

$$!\llbracket\Gamma\rrbracket \xrightarrow{\mu_{\llbracket\Gamma\rrbracket}} !\llbracket\Gamma\rrbracket \otimes !\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma\vdash M\rrbracket\otimes\sigma} \llbracket T\rrbracket \otimes A \xrightarrow{\mathrm{wk}_{\llbracket T\rrbracket,A}} \llbracket T\rrbracket \oslash A \xrightarrow{\tau} \llbracket U\rrbracket \ .$$

Proof. Structural induction on $\mathbf{E}$.

- If $\mathbf{E} = -$ is a hole, then we may take $A = I$, $\sigma = ()$ and $\tau = \mathsf{r}_{\llbracket T\rrbracket}$.
- If $\mathbf{E} = \mathbf{E}'N$ for some term N, where $\mathbf{E}'$ has type $S \to T$, N has type S, and $M : S'$ fits into the hole, then the denotation $\llbracket\Gamma \vdash \mathbf{E}[M]\rrbracket = \llbracket\Gamma \vdash \mathbf{E}'[M]\, N\rrbracket$ is given by the composite

  $$!\llbracket\Gamma\rrbracket \xrightarrow{\mu_{\llbracket\Gamma\rrbracket}} !\llbracket\Gamma\rrbracket\otimes!\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma\vdash\mathbf{E}'[M]\rrbracket\otimes\llbracket\Gamma\vdash N\rrbracket} (!\llbracket S\rrbracket \multimap \llbracket T\rrbracket)\otimes!\llbracket S\rrbracket \xrightarrow{\mathrm{ev}_{!\llbracket S\rrbracket,\llbracket T\rrbracket}} \llbracket T\rrbracket\,.$$

  By the inductive hypothesis, $\llbracket\Gamma \vdash \mathbf{E}'[M]\rrbracket$ factors as

  $$!\llbracket\Gamma\rrbracket \xrightarrow{\mu_{\llbracket\Gamma\rrbracket}} !\llbracket\Gamma\rrbracket\otimes!\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma\vdash M\rrbracket\otimes\sigma'} \llbracket S'\rrbracket\otimes A' \xrightarrow{\mathrm{wk}_{\llbracket S'\rrbracket,A'}} \llbracket S'\rrbracket\oslash A' \xrightarrow{\tau'} (!\llbracket S\rrbracket \multimap \llbracket T\rrbracket),$$

  for appropriate A', σ', τ'. Then $\llbracket\Gamma \vdash \mathbf{E}[M]\rrbracket = \llbracket\Gamma \vdash \mathbf{E}'[M]\, N\rrbracket$ is given by the thick dashed arrows in the diagram in Figure 3.3. But this composite is equal to that given by the thin solid arrows in the diagram, which is of the required form, with

  $$A = A' \otimes !\llbracket S\rrbracket \qquad \sigma = \mu_{\llbracket\Gamma\rrbracket}; (\sigma' \otimes \llbracket\Gamma \vdash N\rrbracket^\dagger)$$

  $$\tau = \mathsf{passoc}^{-1}_{\llbracket S'\rrbracket,A',!\llbracket S\rrbracket}; (\tau' \oslash !\llbracket S\rrbracket); \mathrm{ev}_{s\,!\llbracket S\rrbracket,\llbracket T\rrbracket}\ .$$

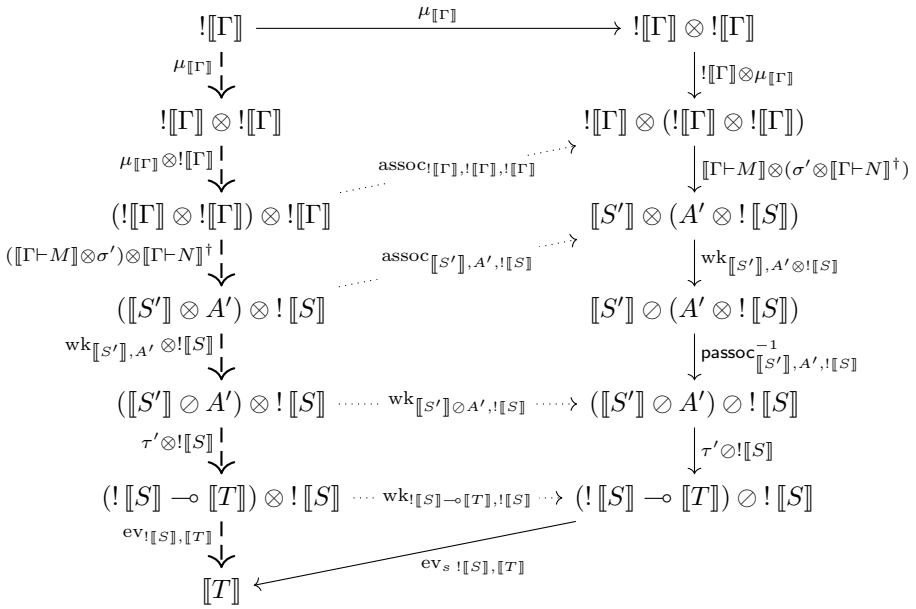


Figure 3.3: The property in Lemma 3.3.2 is preserved by function application. Here, $\mathrm{ev}_{s\,!\llbracket S\rrbracket,\llbracket T\rrbracket} = \Lambda_s^{-1}(\mathrm{id}_{!\llbracket S\rrbracket \multimap \llbracket T\rrbracket})$.

- If $\mathbf{E} = \mathsf{succ}\,\mathbf{E}'$ or $\mathsf{pred}\,\mathbf{E}'$, where $\mathbf{E}'$ is a context of type $\mathtt{nat}$, and $M : T$ is a term that fits into the hole, then the denotation $\llbracket\Gamma \vdash \mathbf{E}[M]\rrbracket$ is given by the composite

$$!\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma \vdash \mathbf{E}'[M]\rrbracket} \mathbb{N} \xrightarrow{\theta} \mathbb{N}\,,$$

  where θ is either pred or succ. By the inductive hypothesis, $\llbracket\Gamma \vdash \mathbf{E}'[M]\rrbracket$ factors as

$$!\llbracket\Gamma\rrbracket \xrightarrow{\mu_{\llbracket\Gamma\rrbracket}} !\llbracket\Gamma\rrbracket \otimes !\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma \vdash M\rrbracket \otimes \sigma'} \llbracket T\rrbracket \otimes A' \xrightarrow{\mathrm{wk}_{\llbracket T\rrbracket,A'}} \llbracket T\rrbracket \oslash A' \xrightarrow{\tau'} \mathbb{N}\,,$$

  for appropriate A', σ', τ'. When we compose on the right by θ, we are already in the required form, for

$$A = A' \qquad \sigma = \sigma' \qquad \tau = \tau'; \theta\,.$$

- Similarly, if $\mathbf{E} = !\mathbf{E}'$, where $\mathbf{E}'$ is a context of type $\mathtt{Var}$, and $M : T$ is a term that fits into the hole, then the denotation $\llbracket\Gamma \vdash \mathbf{E}[M]\rrbracket$ is given by the composite

$$!\llbracket\Gamma\rrbracket \xrightarrow{\llbracket\Gamma \vdash \mathbf{E}'[M]\rrbracket} \mathrm{Var} \xrightarrow{\mathsf{deref}} \mathbb{N}\,.$$

By the inductive hypothesis, $[\![\Gamma \vdash \mathbf{E}'[M]]\!]$ factors as

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma'} [\![T]\!] \otimes A' \xrightarrow{\mathrm{wk}_{[\![T]\!], A'}} [\![T]\!] \oslash A' \xrightarrow{\tau'} \mathrm{Var}\,,$$

for appropriate A', σ', τ'. When we compose on the right by deref, we are already in the required form, for

$$A = A' \qquad \sigma = \sigma' \qquad \tau = \tau'; \mathsf{deref}\ .$$

- If $\mathbf{E} = \mathbf{E}'; N$ for some term N of type $X \in \{\mathtt{com}, \mathtt{bool}, \mathtt{nat}\}$, where $\mathbf{E}'$ is an evaluation context of type $\mathtt{com}$, and if $M\colon T$ fits into the hole in $\mathbf{E}'$, then the denotation $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by the composite

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash \mathbf{E}'[M]]\!] \otimes [\![\Gamma \otimes N]\!]} \mathbb{C} \otimes X \xrightarrow{\Lambda^{-1}(\mathsf{seq}_X)} X\,.$$

  If $\mathbf{E} = N \leftarrow \mathbf{E}'$, for some term N of type $\mathtt{Var}$, where $\mathbf{E}'$ is an evaluation context of type $\mathtt{com}$, and if $M\colon T$ fits into the hole in $\mathbf{E}'$, then the denotation $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by the composite

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash \mathbf{E}'[M]]\!] \otimes [\![\Gamma \otimes N]\!]} \mathbb{N} \otimes \mathtt{Var} \xrightarrow{\Lambda^{-1}(\mathsf{assign})} \mathbb{C}\,.$$

  Write $Y = \mathbb{C}$, $X' = X$, $Z = X$ and $\upsilon = \mathsf{seq}_X$ in the sequencing case, and $Y = \mathbb{N}$, $X' = \mathtt{Var}$, $Z = \mathbb{C}$ and $\upsilon = \mathsf{assign}$ in the variable assignment case. By the inductive hypothesis, $[\![\Gamma \vdash \mathbf{E}'[M]]\!]$ factors as

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma'} [\![T]\!] \otimes A' \xrightarrow{\mathrm{wk}_{[\![T]\!], A'}} [\![T]\!] \oslash A' \xrightarrow{\tau'} Y\,,$$

  for suitable A', σ', τ'. Then $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by the thick dashed arrows in the diagram in Figure 3.4. But this composite is equal to that given by the thin solid arrows in the diagram, which is of the required form, with

$$A = A' \otimes X' \qquad \sigma = \mu_{[\![\Gamma]\!]}; (\sigma' \otimes [\![\Gamma \vdash N]\!])$$

$$\tau = \mathsf{passoc}^{-1}_{[\![T]\!], A', X'}; (\tau' \oslash X'); \Lambda_s^{-1}(\upsilon)\,.$$

- If $\mathbf{E} = \mathsf{If}\ \mathbf{E}'\ \mathsf{then}\ N\ \mathsf{else}\ P$ for a context $\mathbf{E}'$ of type $\mathtt{bool}$, where N and P are terms of type $X \in \{\mathtt{bool}, \mathtt{com}, \mathtt{nat}\}$, then write $Y = \mathtt{bool}$ and $\eta = \mathsf{If}_X$. If $\mathbf{E} = \mathsf{If0}\,\mathbf{E}'\ \mathsf{then}\ N\ \mathsf{else}\ P$ for a context $\mathbf{E}'$ of type $\mathtt{nat}$, where N and P are terms of type $X \in \{\mathtt{bool}, \mathtt{com}, \mathtt{nat}\}$, then write $Y = \mathtt{nat}$ and $\eta = \mathsf{If0}_X$. In either case, if $M\colon T$ is a term that fits into the hole, then the denotation $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{V_\Gamma}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{\mu_\Gamma \otimes [\![\Gamma]\!]} (!{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]}) \otimes !{[\![\Gamma]\!]}$$

$$\xrightarrow{([\![\Gamma \vdash \mathbf{E}'[M]]\!] \otimes [\![\Gamma \vdash N]\!]) \otimes [\![\Gamma \vdash P]\!]} (Y \otimes X) \otimes X \xrightarrow{\Lambda^{-1}(\Lambda^{-1}(\eta))} X\,.$$

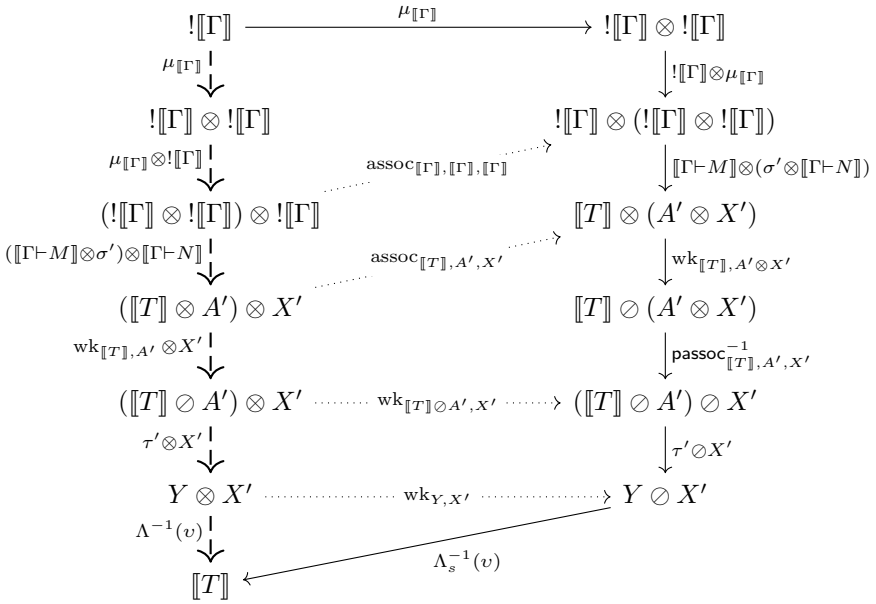


Figure 3.4: The property in Lemma 3.3.2 is preserved by sequencing and variable assignment. We use the fact that $v \in \{\mathsf{seq}_X, \mathsf{assign}\}$ is a strict strategy, so that $\Lambda_s^{-1}(v)$ is well-defined.

By the inductive hypothesis, $[\![\Gamma \vdash \mathbf{E}'[M]]\!]$ factors as

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma'} [\![T]\!] \otimes A' \xrightarrow{\mathrm{wk}_{[\![T]\!], A'}} [\![T]\!] \oslash A' \xrightarrow{\tau'} Y\,,$$

for appropriate A', σ', τ'. Then $[\![\Gamma \vdash \mathbf{E}[M]]\!] = [\![\Gamma \vdash \mathbf{E}'[M]\, N]\!]$ is given by the thick dashed arrows in the diagram in Figure 3.5. But this composite is equal to that given by the thin solid arrows in the diagram, which is of the required form, with

$$A = (A' \otimes X) \otimes X \qquad \sigma = \mu_{[\![\Gamma]\!]}; (\mu_{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]}); ((\sigma' \otimes [\![\Gamma \vdash N]\!]) \otimes [\![\Gamma \vdash P]\!])$$

$$\tau =$$

$$\mathsf{passoc}^{-1}_{[\![T]\!], A' \otimes X, X}; (\mathsf{passoc}^{-1}_{[\![T]\!], A', X} \oslash X); ((\tau' \oslash X) \oslash X); \Lambda_s^{-1}(\Lambda_s^{-1}(\eta))\,.$$

- If $\mathbf{E} = \mathsf{let}_{Y,X}\, x = \mathbf{E}'$ in N, for $X \in \{\mathbb{C}, \mathbb{B}, \mathbb{N}\}$ and $Y \in \{\mathbb{B}, \mathbb{N}\}$, then the denotation $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{V_\Gamma}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{[\![\Gamma \vdash \mathbf{E}'[M]]\!] \otimes [\![\Gamma \vdash \lambda x.N]\!]} Y \otimes (!Y \multimap X) \xrightarrow{\Lambda^{-1}(\mathsf{let})} X\,.$$

  By the inductive hypothesis, $[\![\Gamma \vdash \mathbf{E}'[M]]\!]$ can be written as

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma'} [\![T]\!] \otimes A' \xrightarrow{\mathrm{wk}_{[\![T]\!], A'}} [\![T]\!] \oslash A' \xrightarrow{\tau'} Y\,,$$

  for suitable A, σ, τ. This means that $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by the thick dashed arrows in the diagram in Figure 3.6. But this composite is equal to that given by the thin solid arrows in the diagram, which is of the required form with

$$A = A' \otimes (!Y \multimap X) \qquad \sigma = \mu_{[\![\Gamma]\!]}; (\sigma' \otimes [\![\Gamma \vdash \lambda x.N]\!])$$

$$\tau = \mathsf{passoc}^{-1}_{[\![T]\!], A', (!Y \multimap X)}; (\tau' \oslash (!Y \multimap [\![T]\!])); \Lambda_s^{-1}(\mathsf{let})\,.$$

- Lastly, suppose that $\mathbf{E} = \mathbf{E}' \leftarrow n$ for some numeral n, where $\mathbf{E}'$ is a context of type $\mathtt{Var}$, and suppose that $M\colon T$ fits into the hole. The denotation $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ is given by

$$!{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash \mathbf{E}'[M]]\!]} \mathrm{Var} \xrightarrow{\mathrm{lunit}_{\mathrm{Var}}} I \otimes \mathrm{Var} \xrightarrow{n \otimes \mathrm{Var}} \mathbb{N} \otimes \mathrm{Var} \xrightarrow{\mathsf{assign}} \mathbb{C}\,.$$

  By the induction hypothesis, $[\![\Gamma \vdash \mathbf{E}'[M]]\!]$ takes the form

$$!{[\![\Gamma]\!]} \xrightarrow{\mu_{[\![\Gamma]\!]}} !{[\![\Gamma]\!]} \otimes !{[\![\Gamma]\!]} \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma'} [\![T]\!] \otimes A' \xrightarrow{\mathrm{wk}_{[\![T]\!], A'}} [\![T]\!] \oslash A' \xrightarrow{\tau'} \mathrm{Var}\,,$$

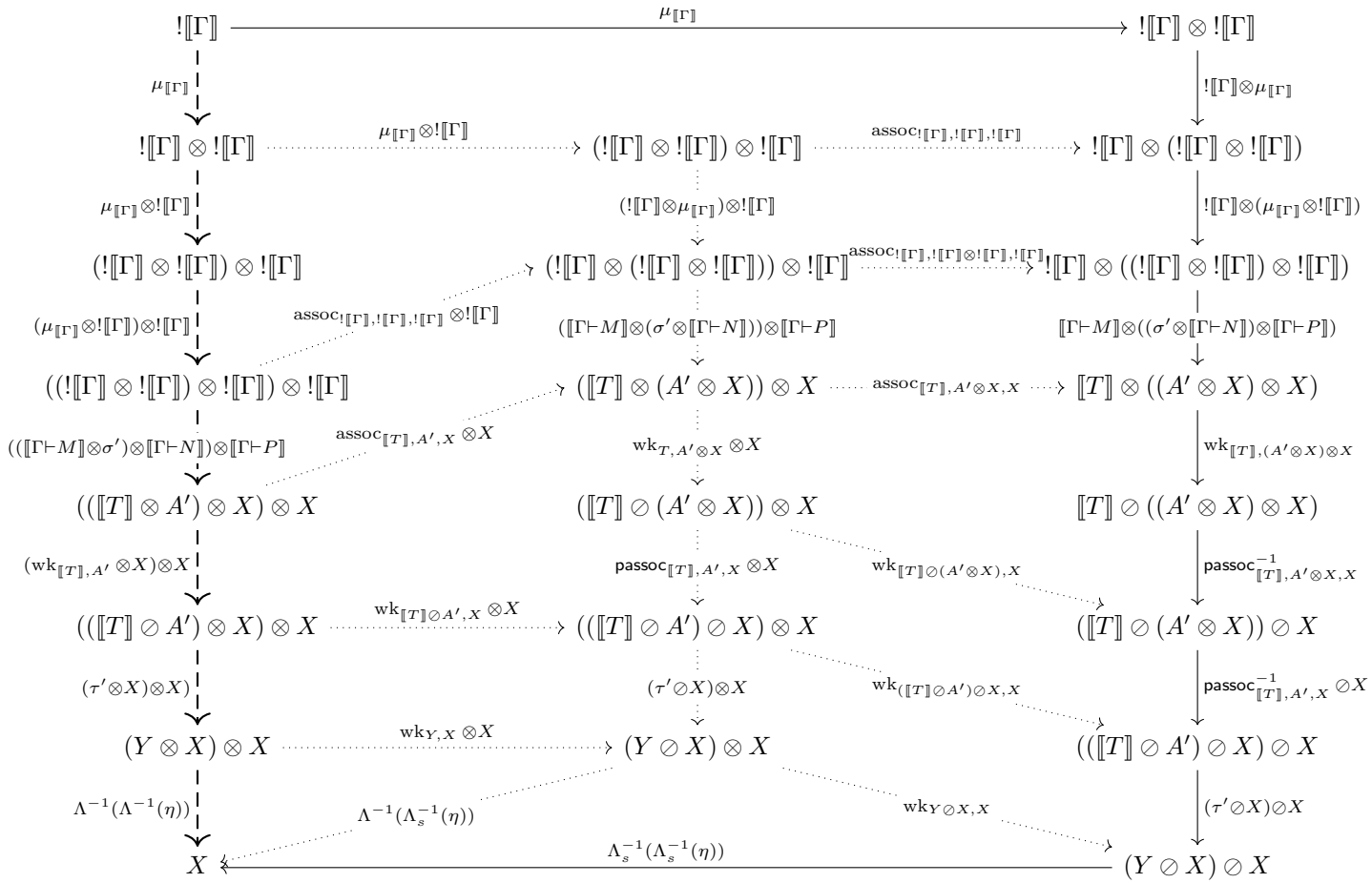


Figure 3.5: The property in Lemma 3.3.2 is preserved by conditionals. We use the fact that $\eta \in \{\mathsf{If}_X, \mathsf{If0}_X\}$ is a strict strategy, and that the Λ_s^{-1} is a function from strict strategies to strict strategies, so that $\Lambda_s^{-1}(\Lambda_s^{-1}(\eta))$ is well-defined and strict.

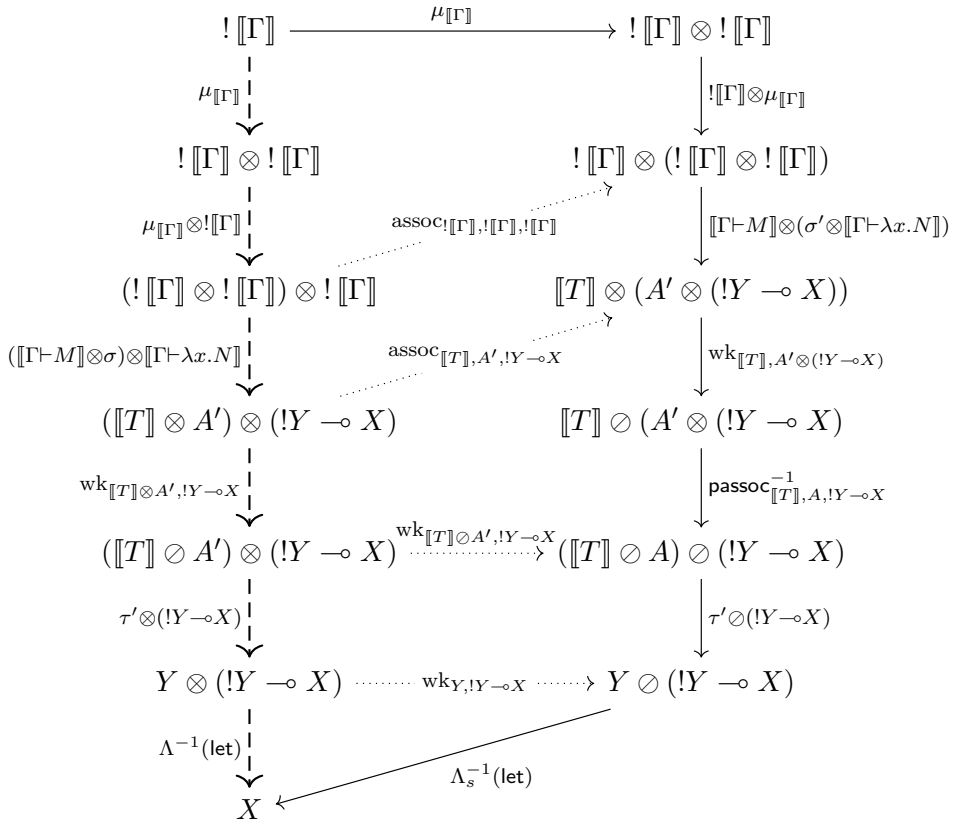


Figure 3.6: The property in Lemma 3.3.2 is preserved by the let keyword. We use the fact that $\mathsf{let}_{Y,X}$ is a strict strategy, and that the Λ_s^{-1} is a function from strict strategies to strict strategies.

for suitable A', σ', τ'. When we compose on the right by the morphism $\mathrm{lunit}_{\mathrm{Var}}; (n \otimes \mathrm{Var}); \mathsf{assign}$, in order to give us the denotation $[\![\Gamma \vdash \mathbf{E}'[M] \leftarrow n]\!] = [\![\Gamma \vdash \mathbf{E}[M]]\!]$, then we are already in the required form, with

$$A = A' \qquad \sigma = \sigma' \qquad \tau = \tau'; \mathrm{lunit}_{\mathrm{Var}}; (n \otimes \mathrm{Var}); \mathsf{assign}\ . \qquad \square$$

Having proved this result about evaluation contexts, we now give an outline of the general soundness proof. We would like to represent a term-with-store $\Gamma, s \vdash M : T$ by its denotation

$$I \xrightarrow{![\![s]\!]} !S_\Gamma \xrightarrow{\mathsf{cell}^\Gamma} ![\![\Gamma]\!] \xrightarrow{[\![\Gamma \vdash M]\!]} [\![T]\!]\ .$$

This composite tells us what the behaviour of M is when we evaluate it with state s. However, it tells us nothing about the state that M leaves after it has evaluated, and is therefore inadequate for inductive proofs along a derivation.

Instead, we want to capture information about the state that M leaves, which we do using the following derivation.

- We use the comonoid structure of $![\![\Gamma]\!]$ to create two separate references to the storage cells – one for the use of M and one for the use of any code that will run after M has finished.
- We then use the sequoid operator to indicate the temporal relationship – namely, that M runs first and all other accesses to the storage cells will receive the state that M left behind.

Definition 3.3.3. Let $\Gamma, s \vdash M : T$ be a term with store. Then we define the *sequoidal denotation* $[\!|\Gamma, s \vdash M|\!]$ to be the composite

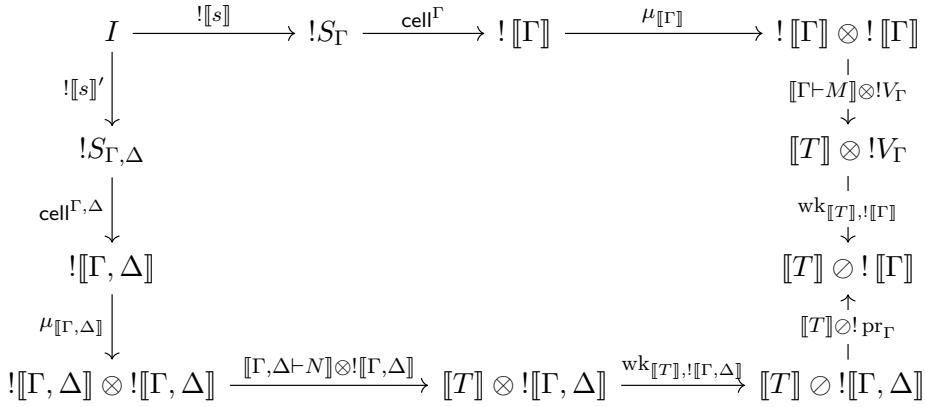
$$I \xrightarrow{![\![s]\!]} !S_\Gamma \xrightarrow{\mathsf{cell}^\Gamma} ![\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} ![\![\Gamma]\!] \otimes ![\![\Gamma]\!] \xrightarrow{[\![\Gamma \vdash M]\!] \otimes ![\![\Gamma]\!]} [\![T]\!] \otimes ![\![\Gamma]\!] \xrightarrow{\mathrm{wk}_{[\![T]\!], ![\![\Gamma]\!]}} [\![T]\!] \oslash ![\![\Gamma]\!]\ .$$

Our next lemma will help us deal with the base $\dashrightarrow$ rules. We will then use Lemma 3.3.2 to extend this to the $\longrightarrow$ relation.

Lemma 3.3.4. *The relation $\dashrightarrow$ between triples $\Gamma, s \vdash M$ preserves the sequoidal denotation $[\!|\Gamma, s \vdash M|\!]$.*

In other words, if $\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash N$, where M, N have type T, then

the following diagram commutes.



I.e., if $\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash N$*, then* $[\![\Gamma, s \vdash M]\!] = [\![\Gamma, \Delta, s' \vdash N]\!]; !\,\mathrm{pr}_\Gamma$.

Proof. We prove this on a case-by-case basis.

- For most of the rules, $\Delta = \_$ and $s = s'$; i.e., the rule is of the form

  $$\Gamma, s \vdash M \dashrightarrow \Gamma, s \vdash N$$

  for some M, N. In such a case, it suffices to show that $\llbracket \Gamma \vdash M \rrbracket = \llbracket \Gamma \vdash N \rrbracket$. Indeed, we have

  - $\llbracket \Gamma \vdash (\lambda x.M)N \rrbracket = \llbracket \Gamma \vdash M[N/x] \rrbracket$ (by a usual substitution-lemma argument);
  - $\llbracket \Gamma \vdash \mathbf{Y}M \rrbracket = \llbracket \Gamma \vdash (\lambda F.\lambda f.f(Ff))\mathbf{Y}M \rrbracket = \llbracket \Gamma \vdash M(\mathbf{Y}M) \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{succ}\, n \rrbracket = \llbracket \Gamma \vdash n + 1 \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{pred}(n+1) \rrbracket = \llbracket \Gamma \vdash n \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{pred}\, 0 \rrbracket = \llbracket \Gamma \vdash 0 \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{skip}; M \rrbracket = \llbracket \Gamma \vdash M \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{If}\ \mathtt{t}\ \mathsf{then}\ N\ \mathsf{else}\ P \rrbracket = \llbracket \Gamma \vdash (\lambda x.\lambda y.x)NP \rrbracket = \llbracket \Gamma \vdash N \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{If}\ \mathtt{f}\ \mathsf{then}\ N\ \mathsf{else}\ P \rrbracket = \llbracket \Gamma \vdash (\lambda x.\lambda y.y)NP \rrbracket = \llbracket \Gamma \vdash P \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{If0}\, 0\ \mathsf{then}\ N\ \mathsf{else}\ P \rrbracket = \llbracket \Gamma \vdash (\lambda x.\lambda y.x)NP \rrbracket = \llbracket \Gamma \vdash N \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{If0}(n+1)\ \mathsf{then}\ N\ \mathsf{else}\ P \rrbracket = \llbracket \Gamma \vdash (\lambda x.\lambda y.y)NP \rrbracket = \llbracket \Gamma \vdash P \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{let}\, x = \mathtt{t}\ \mathsf{in}\ M \rrbracket = \llbracket (\lambda x.M)\mathtt{t} \rrbracket = \llbracket M[\mathtt{t}/x] \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{let}\, x = \mathtt{f}\ \mathsf{in}\ M \rrbracket = \llbracket (\lambda x.M)\mathtt{f} \rrbracket = \llbracket M[\mathtt{f}/x] \rrbracket$;
  - $\llbracket \Gamma \vdash \mathsf{let}\, x = n\ \mathsf{in}\ M \rrbracket = \llbracket (\lambda x.M)n \rrbracket = \llbracket M[n/x] \rrbracket$;

– $[\![\Gamma \vdash (\mathsf{mkvar}\, W\, R) \leftarrow n]\!] \;=\; \langle [\![\Gamma \vdash Wn]\!]_{n\in\mathbb{N}}, [\![\Gamma \vdash R]\!]\rangle; \mathrm{pr}_n \;=\; [\![\Gamma \vdash Wn]\!]$; and

– $[\![\Gamma \vdash !(\mathsf{mkvar}\, W\, R)]\!] = \langle [\![\Gamma \vdash Wn]\!]_{n\in\mathbb{N}}, [\![\Gamma \vdash R]\!]\rangle; \mathrm{pr}_{\mathbb{N}} = [\![\Gamma \vdash R]\!]$.

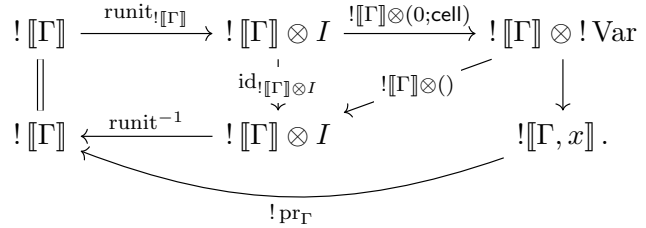
- Now consider the rule

$$\Gamma, s \vdash \mathsf{new}\, \lambda x.M \dashrightarrow \Gamma, x, (s|x \mapsto 0) \vdash M\,,$$

where $\Gamma \vdash M : T$. The first observation to make is that the denotation $[\![\Gamma \vdash \mathsf{new}\, \lambda x.M]\!]$ may be written as

$$!\,[\![\Gamma]\!] \xrightarrow{\mathrm{runit}_{![\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes I \xrightarrow{![\![\Gamma]\!]\otimes(0;\mathsf{cell})} !\,[\![\Gamma]\!] \otimes !\,\mathrm{Var} \to ![\![\Gamma, x]\!] \xrightarrow{\Lambda^{-1}([\![\Gamma \vdash \lambda x.M]\!])} [\![T]\!]\,,$$

where, of course, $\Lambda^{-1}([\![\Gamma \vdash \lambda x.M]\!]) = [\![\Gamma, x \vdash M]\!]$, and that the projection $\mathrm{pr}_\Gamma : ![\![\Gamma, x]\!]$ is a right inverse for the composite $\mathrm{runit}_{![\![\Gamma]\!]}; (![\![\Gamma]\!] \otimes (0; \mathsf{cell})); \cong$, since we have a commutative diagram



Then we may prove this case using the commutative diagram in Figure 3.7.

- Now consider the rules

$$x, \Gamma, s \vdash x \leftarrow n \dashrightarrow x, \Gamma, (s|x \mapsto n) \vdash \mathsf{skip} \qquad x, \Gamma, s \vdash !x \dashrightarrow x, \Gamma, s \vdash s(x)\,.$$

In each of these cases, the outer context Γ is unchanged by the rule; moreover, each term ignores the variables in Γ. So the denotations of the terms-in-context $x, \Gamma \vdash M : T$ on either side of each rule are of the form

$$![\![x, \Gamma]\!] \xrightarrow{!\,\mathrm{pr}_1} !\,\mathrm{Var} \xrightarrow{[\![x \vdash M]\!]} [\![T]\!]\,.$$

Now the diagram in Figure 3.8 tells us that each $[\![\Gamma, x, s \vdash M]\!]$ may be written as the composite

$$!\,[\![s|_\Gamma]\!]\,; \mathsf{cell}^\Gamma; \mathrm{lunit}_{![\![\Gamma]\!]};$$

$$((!\,[\![s|_x]\!]\,; \mathsf{cell}; \mu_{\mathrm{Var}}; ([\![x \vdash M]\!] \otimes !\,\mathrm{Var}); \mathrm{wk}_{[\![T]\!],!\,\mathrm{Var}}) \otimes !\,[\![\Gamma]\!]);$$

$$\mathrm{wk}_{[\![T]\!]\oslash !\,\mathrm{Var},![\![\Gamma]\!]}; \mathsf{passoc}_{[\![T]\!],!\,\mathrm{Var},![\![\Gamma]\!]}; \cong$$

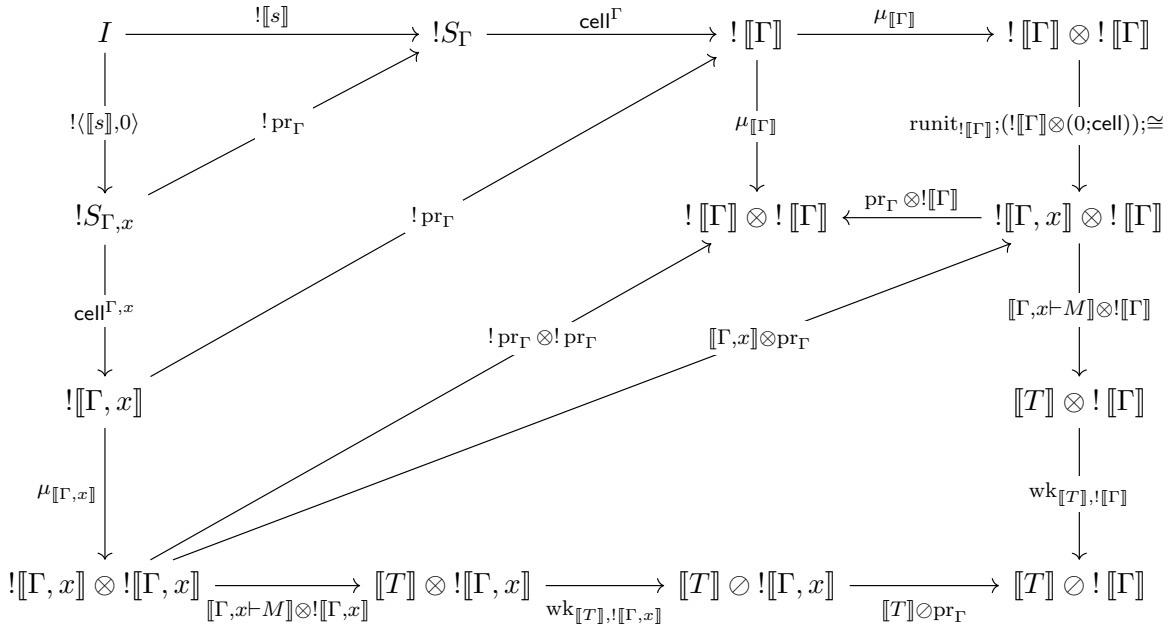


Figure 3.7: The conclusion of Lemma 3.3.4 holds for the new rule.

(where $\cong$ represents the natural isomorphism between the games $[\![T]\!] \oslash (!\,\mathrm{Var} \otimes\, ![\![\Gamma]\!])$ and $[\![T]\!] \oslash\, ![\![x, \Gamma]\!]$), and so is completely determined by the value of the composite

$$!\,[\![s|_x]\!]\,; \mathsf{cell}; \mu_{\mathrm{Var}}; ([\![x \vdash M]\!] \otimes !\,\mathrm{Var}); \mathrm{wk}_{[\![T]\!],!\,\mathrm{Var}}\,;$$

i.e., the sequoidal denotation $[\!\![x, s|_x \vdash M]\!\!]$.

This tells us that we can assume that Γ is the empty context, so that the rules take on the form

$$x, s \vdash x \leftarrow n \dashrightarrow x, (x \mapsto n) \vdash \mathsf{skip} \qquad x, s \vdash !x \dashrightarrow x, s \vdash s(x)\,.$$

Now the commutative diagrams in Figure 3.9 prove that

$$[\!\![x, s|_x \vdash x \leftarrow n]\!\!] = [\!\![x, (x \mapsto n) \vdash \mathsf{skip}]\!\!]$$

and that

$$[\!\![x, s|_x \vdash !x]\!\!] = [\!\![x, s|_x \vdash s(x)]\!\!]\,,$$

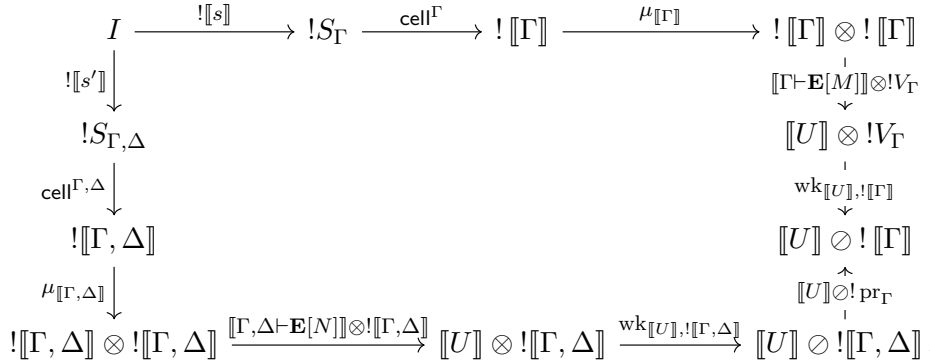
completing the proof. □

Now that we have dealt with the base rules, we can move on to the full relation $\longrightarrow$.

Lemma 3.3.5. *The relation $\longrightarrow$ between triples $\Gamma, s \vdash M$ preserves the sequoidal denotation*

$$[\!\![\Gamma, s \vdash M]\!\!] = !\,[\![s]\!]\,; \mathsf{cell}^{\Gamma}; \mu_{[\![\Gamma]\!]}; ([\![\Gamma \vdash M]\!] \otimes !\,[\![\Gamma]\!]); \mathrm{wk}_{[\![T]\!],![\![\Gamma]\!]}\,.$$

I.e., if $\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash N$, where M, N have type T, and $\mathbf{E}$ is a context of type U with a hole of type T, then the following diagram commutes.



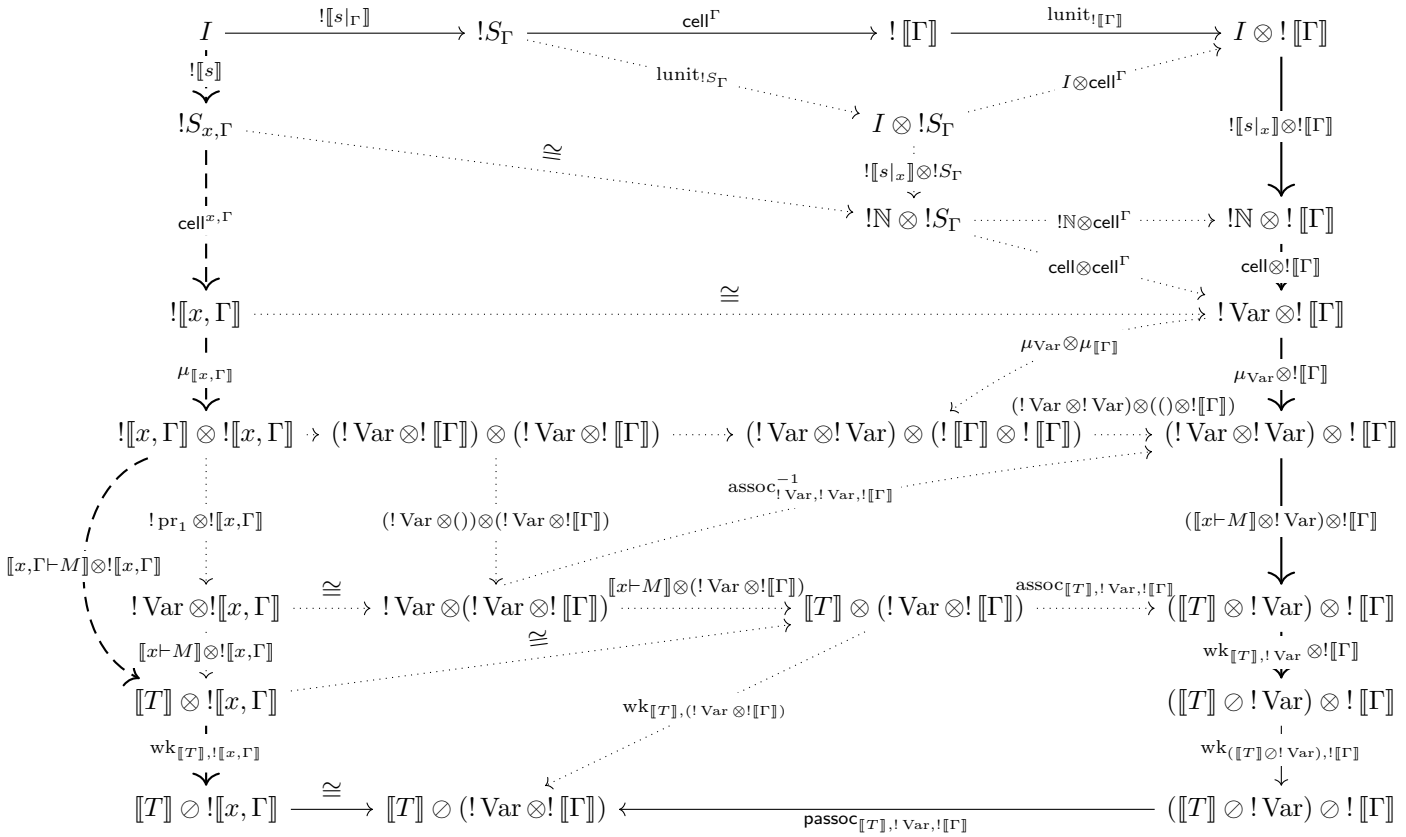


Figure 3.8: Diagram proving that if we want to prove the conclusion of Lemma 3.3.4 for a small-step rule that does not change the context and only mentions one variable, then it suffices to assume that that variable is the only variable in the context.

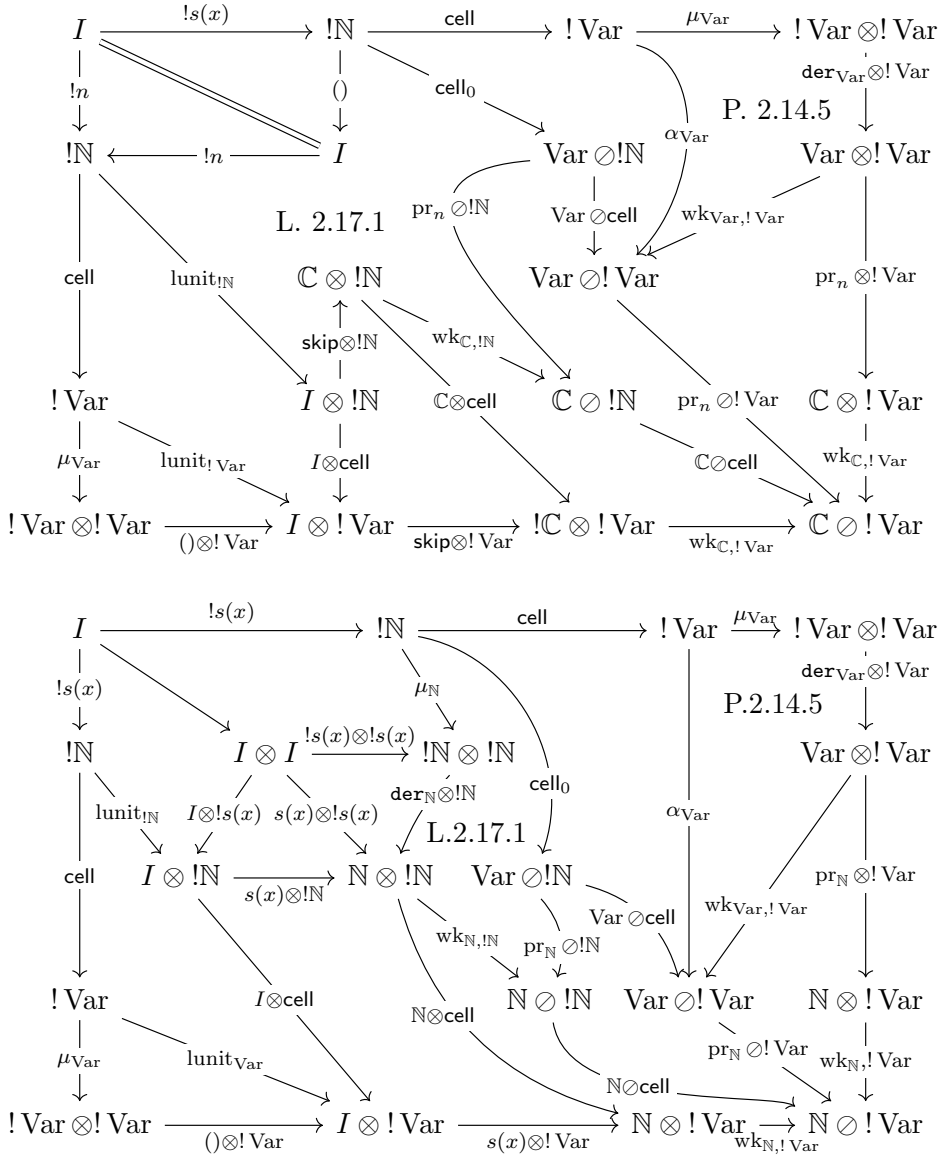


Figure 3.9: Diagrams to prove that the conclusion of Lemma 3.3.4 holds for the storage cell rules. References in the middle of a shape refer to Lemma 2.17.1 and Proposition 2.14.5 above. Note the prominent role played in both diagrams by the anamorphism square for cell given in Section 2.17.

Proof. We use Lemma 3.3.2 and Lemma 3.3.5. Lemma 3.3.2 tells us that for any $\Gamma, s \vdash M$, $[\![\Gamma \vdash \mathbf{E}[M]]\!]$ may be written as

$$!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{[\![\Gamma \vdash M]\!] \otimes \sigma} [\![T]\!] \otimes A \xrightarrow{\mathrm{wk}_{[\![T]\!], A}} [\![T]\!] \oslash A \xrightarrow{\tau} [\![U]\!]$$

for suitably chosen A, σ, τ.

Let us write $[\![\Gamma, s \vdash M]\!]$ for the composite

$$!\,[\![s]\!] \,;\, \mathsf{cell}^{\Gamma} ; \mu_{[\![\Gamma]\!]} ; ([\![\Gamma \vdash M]\!] \otimes !\,[\![\Gamma]\!]) ; \mathrm{wk}_{[\![T]\!], ![\![\Gamma]\!]} ; ([\![T]\!] \oslash (\Gamma + 0_{\Delta}))\,.$$

So we are trying to show that $[\![\Gamma, s \vdash \mathbf{E}[M]]\!] = [\![\Gamma, \Delta, s' \vdash \mathbf{E}[N]]\!] ; ([\![U]\!] \oslash !\,\mathrm{pr}_{\Gamma})$.

Now the diagram in Figure 3.10 shows us that for any $\Gamma, s \vdash M$ we may write

$$[\![\Gamma, s \vdash \mathbf{E}[M]]\!] = [\![\Gamma, s \vdash M]\!] ; ([\![T]\!] \oslash (\mu_{[\![\Gamma]\!]} ; (\sigma \otimes !\,[\![\Gamma]\!])) ; \mathsf{passoc}^{-1}_{[\![T]\!], A, ![\![\Gamma]\!]} ; (\tau \oslash !\,[\![\Gamma]\!])\,.$$

Therefore, Lemma 3.3.4 tells us that if $\Gamma, s \vdash M \dashrightarrow \Gamma, \Delta, s' \vdash N$, then we have

$$\begin{aligned}
&[\![\Gamma, s \vdash \mathbf{E}[M]]\!] \\
&= [\![\Gamma, s \vdash M]\!] ; ([\![T]\!] \oslash (\mu_{[\![\Gamma]\!]} ; (\sigma \otimes !\,[\![\Gamma]\!])) ; \mathsf{passoc}^{-1} ; (\tau \oslash !\,[\![\Gamma]\!]) \\
&= [\![\Gamma, s \vdash N]\!] ; ([\![T]\!] \oslash !\,\mathrm{pr}_{\Gamma}) ; ([\![T]\!] \oslash (\mu_{[\![\Gamma]\!]} ; (\sigma \otimes !\,[\![\Gamma]\!])) ; \mathsf{passoc}^{-1} ; (\tau \oslash !\,[\![\Gamma]\!]) \\
&= [\![\Gamma, s \vdash N]\!] ; ([\![T]\!] \oslash (\mu_{[\![\Gamma, \Delta]\!]} ; ((\mathrm{pr}_{\Gamma} ; \sigma) \otimes ![\![\Gamma, \Delta]\!]))) ; \mathsf{passoc}^{-1} ; \\
&\qquad (\tau \oslash !\,[\![\Gamma]\!]) ; ([\![U]\!] \oslash !\,\mathrm{pr}_{\Gamma}) \\
&= [\![\Gamma, s \vdash \mathbf{E}[N]]\!] ; ([\![U]\!] \oslash !\,\mathrm{pr}_{\Gamma})\,,
\end{aligned}$$

as desired. □

It is now a simple induction to lift this result from the small-step $\longrightarrow$ relation to the big-step $\Downarrow$ relation.

Lemma 3.3.6. *Suppose that $\Gamma, s \vdash M \Downarrow c, s'$, where $M, c \colon T$. Then the following diagram commutes.*

$$\begin{array}{ccccccc}
I & \xrightarrow{![\![s]\!]} & !S_{\Gamma} & \xrightarrow{\mathsf{cell}^{\Gamma}} & !\,[\![\Gamma]\!] & \xrightarrow{\mu_{[\![\Gamma]\!]}} & !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \\
{\scriptstyle ![\![s']\!]}\downarrow & & & & & & \downarrow{\scriptstyle [\![\Gamma \vdash M]\!] \otimes ![\![\Gamma]\!]} \\
!S_{\Gamma} & & & & & & [\![T]\!] \otimes !\,[\![\Gamma]\!] \\
{\scriptstyle \mathsf{cell}^{\Gamma}}\downarrow & & & & & & \downarrow{\scriptstyle \mathrm{wk}_{[\![T]\!], ![\![\Gamma]\!]}} \\
!\,[\![\Gamma]\!] & \xrightarrow{\mu_{[\![\Gamma]\!]}} & !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] & \xrightarrow{[\![\Gamma \vdash c]\!] \otimes ![\![\Gamma]\!]} & [\![T]\!] \otimes !\,[\![\Gamma]\!] & \xrightarrow{\mathrm{wk}_{[\![T]\!], ![\![\Gamma]\!]}} & [\![T]\!] \oslash !\,[\![\Gamma]\!]
\end{array}$$

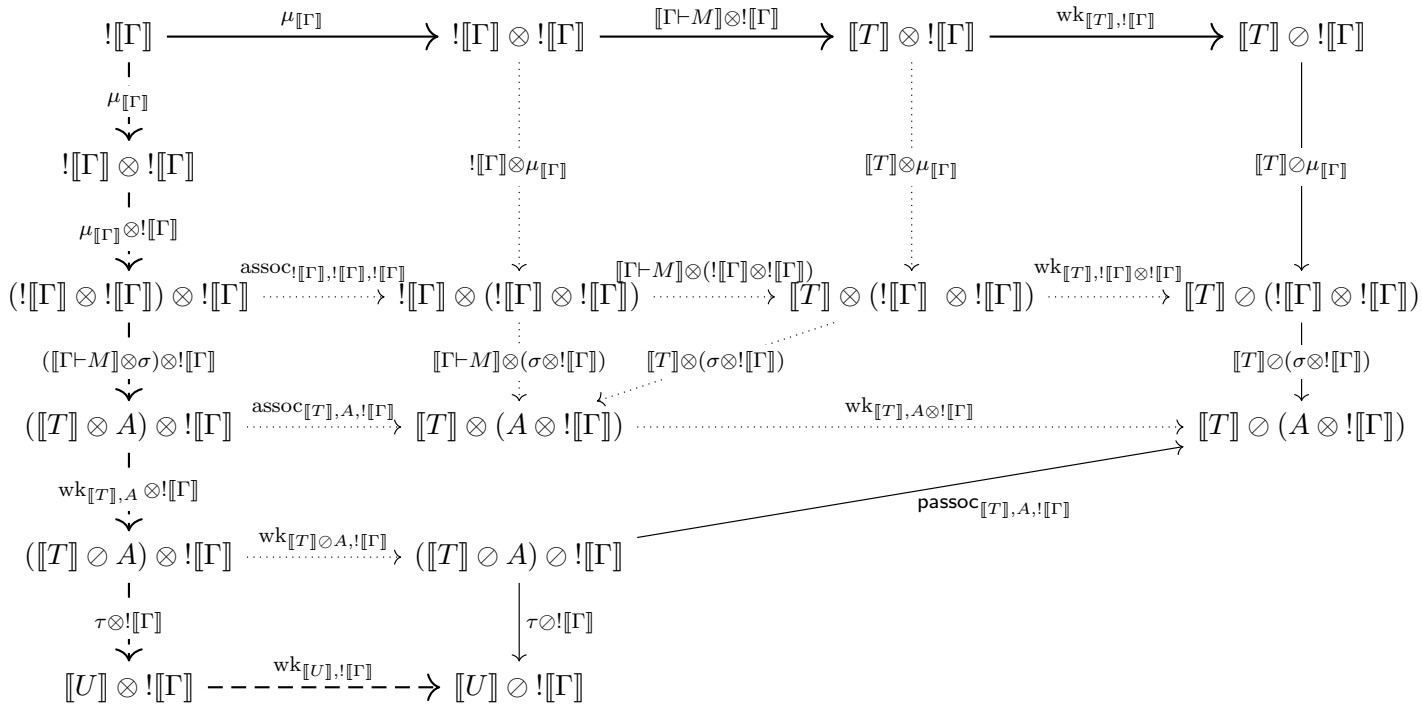


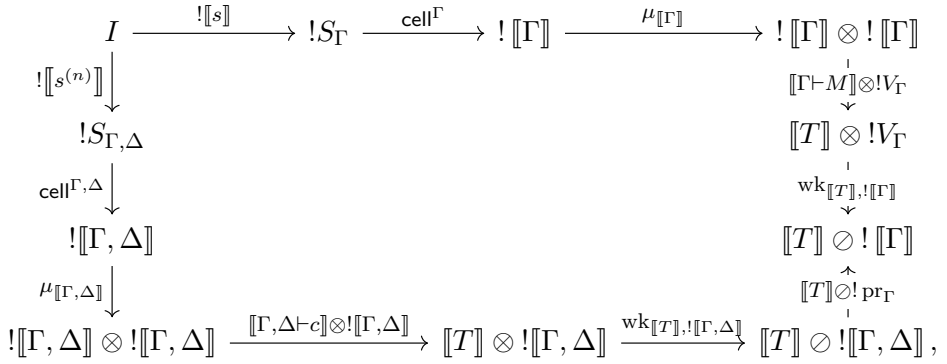
Figure 3.10: Diagram proving that the conclusion of Lemma 3.3.4 can be lifted to the $\longrightarrow$ relation.

Proof. By Proposition 3.2.3, there are sequences $\Gamma = \Gamma_1, \cdots, \Gamma_n = \Gamma, \Delta$, $s = s^{(1)}, \cdots, s^{(n)}$, $M = M_1, \cdots, M_n = c$ such that

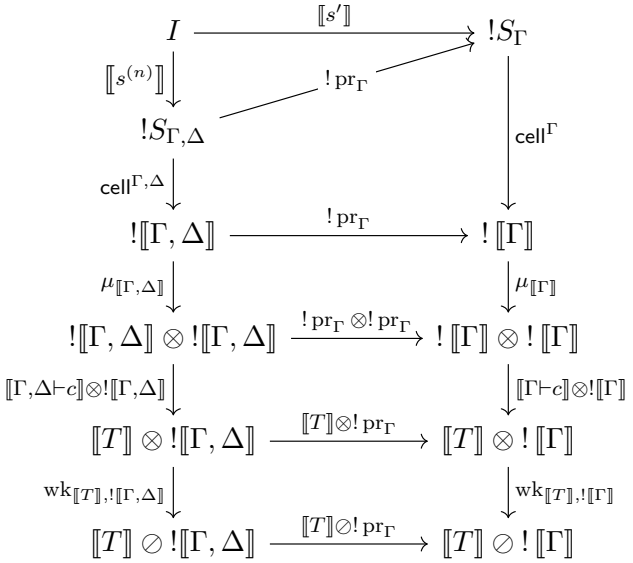
$$\Gamma_1, s^{(1)} \vdash M_1 \longrightarrow \cdots \longrightarrow \Gamma_n, s^{(n)} \vdash M_n \,,$$

and $s^{(n)}|_\Gamma = s'$.

By inductively applying Lemma 3.3.5, we see that we have a commutative diagram



Now, since $s^{(n)}|_\Gamma = s'$, we have a commutative diagram



which, together with the diagram above, gives us the commutative diagram in the statement. □

So we have now proved that if $\Gamma, s \vdash M \Downarrow c, s'$, then the sequoidal denotations

$$\llbracket \Gamma, s \vdash M \rrbracket \qquad\qquad \llbracket \Gamma, s' \vdash c \rrbracket$$

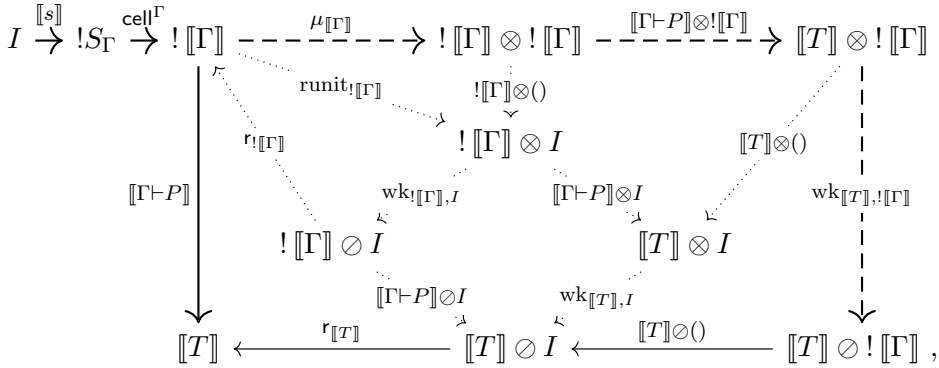
are equal.

It is then an easy corollary to show the sense in which our semantics is sound.

Proposition 3.3.7 ([AM96])**.** *Suppose that* $\Gamma, s \vdash M \Downarrow c, s'$*. Then*

$$[\![s]\!] \,;\, \mathsf{cell}^{\Gamma} \,;\, [\![\Gamma \vdash M]\!] = [\![s']\!] \,;\, \mathsf{cell}^{\Gamma} \,;\, [\![\Gamma \vdash c]\!] \,.$$

Proof. Lemma 3.3.6, plus the fact that if $\Gamma, s \vdash P \colon T$, then we have a commutative diagram



allowing us to recover $[\![s]\!] \,;\, \mathsf{cell}^{\Gamma} \,;\, [\![\Gamma \vdash P]\!]$ from

$$/\!/\Gamma, s \vdash P/\!/ = [\![s]\!] \,;\, \mathsf{cell}^{\Gamma} \,;\, \mu_{[\![\Gamma]\!]} \,;\, ([\![\Gamma \vdash P]\!] \otimes ![\![\Gamma]\!]) \,;\, \mathrm{wk}_{[\![T]\!], ![\![\Gamma]\!]} \,,$$

for $P = M, c$. □

3.4 Computational Adequacy

Our proof of computational adequacy is based on that from [AM96], but modified to make use of the coalgebraic definition of the cell strategy. As is usual in proofs of computational adequacy, our proof relies on logical relations.

First, we note some additional order-theoretic properties of our model. For any game A, we have a strategy $\bot_A \colon A$, given by $\bot_A = \{\epsilon\}$; i.e., the strategy that has no reply even for the very first move. It is clear that $\bot$ is the bottom element of the set of strategies for A, ordered by inclusion.

It is then easy to see the following.

Proposition 3.4.1.

- *Given* $\sigma\colon A \multimap B$, $\sigma; \bot_{B \multimap C} = \bot_{A \multimap C}$.
- *Given a strict strategy* $\tau\colon B \multimap C$, $\bot_{A \multimap B}; \tau = \bot_{A \multimap C}$.
- $\bot_{A \multimap B}$ *is a strict strategy and* $\bot_{A \multimap B} \oslash \sigma = \bot_{(A \oslash C) \multimap (B \oslash D)}$ *for any* $\sigma\colon C \multimap D$.
- $\Lambda(\bot_{A \otimes B \multimap C}) = \bot_{A \multimap (B \multimap C)}$.
- *Given a zigzag (copycat) strategy* $\mathrm{zz}_\phi\colon B \multimap C$ *and a strategy* $\sigma\colon A \multimap B$, *if* $\sigma; \mathrm{zz}_\phi = \bot_{A \multimap C}$ *then* $\sigma = \bot_{A \multimap B}$.

Definition 3.4.2. Given a `Var`-store Γ, a strategy $\sigma\colon !\llbracket \Gamma \rrbracket \multimap A$ and a Γ-store s, we write

$$\llparenthesis s, \sigma \rrparenthesis$$

for the composite

$$I \xrightarrow{!\llbracket s \rrbracket} !S_\Gamma \xrightarrow{\mathsf{cell}^\Gamma} !\llbracket \Gamma \rrbracket \xrightarrow{\mu_{\llbracket \Gamma \rrbracket}} !\llbracket \Gamma \rrbracket \otimes !\llbracket \Gamma \rrbracket \xrightarrow{\sigma \otimes !\llbracket \Gamma \rrbracket} A \otimes !\llbracket \Gamma \rrbracket \xrightarrow{\mathrm{wk}_{A, !\llbracket \Gamma \rrbracket}} A \oslash !\llbracket \Gamma \rrbracket \ .$$

In particular, if $\Gamma \vdash M\colon T$ is a term in context, then $\llparenthesis s, \llbracket \Gamma \vdash M \rrbracket \rrparenthesis$ is equal to the sequoidal denotation $\llparenthesis \Gamma, s \vdash M \rrparenthesis$.

Definition 3.4.3. We inductively define a relation $\lhd^\Gamma_T$, where Γ is a `Var`-store and T a type, between strategies $\sigma\colon !\llbracket \Gamma \rrbracket \multimap \llbracket T \rrbracket$ and terms $\Gamma \vdash M\colon T$ in context as follows.

- If $X \in \{\mathbb{C}, \mathbb{B}, \mathbb{N}\}$ is a datatype, $M\colon X$, $u \in X$ and $\sigma\colon !\llbracket \Gamma \rrbracket \multimap X$, then we say that $\sigma \lhd^\Gamma_X u$ if for all Γ-stores s, either $\llparenthesis s, \sigma \rrparenthesis = \bot_{A \oslash !\llbracket \Gamma \rrbracket}$ or $\llparenthesis s, \sigma \rrparenthesis = \llparenthesis \Gamma, s' \vdash u \rrparenthesis$ for some Γ-store s' such that $\Gamma, s \vdash M \Downarrow u, s'$.
- If $\sigma\colon !\llbracket \Gamma \rrbracket \multimap \mathrm{Var}$ and $\Gamma \vdash M\colon \mathtt{Var}$, we say that $\sigma \lhd^\Gamma_{\mathtt{Var}} M$ if

  $$\sigma; \mathrm{pr}_n \lhd^\Gamma_{\mathtt{com}} M \leftarrow n$$

  for all n, and if

  $$\sigma; \mathrm{pr}_{\mathbb{N}} \lhd^\Gamma_{\mathtt{nat}} !M \, .$$

- If $\sigma\colon !\llbracket \Gamma \rrbracket \multimap (!\llbracket S \rrbracket \multimap \llbracket T \rrbracket)$ and $M\colon S \to T$, we say that $\sigma \lhd^\Gamma_{S \to T} M$ if whenever $\tau\colon !\llbracket \Gamma \rrbracket \multimap \llbracket S \rrbracket$ is a strategy and $N\colon S$ is a term such that $\tau \lhd^\Gamma_S N$, then

  $$\left(!\llbracket \Gamma \rrbracket \xrightarrow{\mu_{\llbracket \Gamma \rrbracket}} !\llbracket \Gamma \rrbracket \otimes !\llbracket \Gamma \rrbracket \xrightarrow{\sigma \otimes \tau^\dagger} (!\llbracket S \rrbracket \multimap \llbracket T \rrbracket) \otimes !\llbracket S \rrbracket \xrightarrow{\mathrm{ev}} \llbracket T \rrbracket \right) \lhd^\Gamma_T M\, N.$$

We now prove some lemmata about this new relation.

Lemma 3.4.4. *Let $\Gamma \vdash M, N : T$ be terms in context of Idealized Algol such that*

$$\Gamma, s \vdash M \longrightarrow \Gamma, s \vdash N$$

for all Γ-stores s. Suppose $\sigma : !\llbracket\Gamma\rrbracket \multimap \llbracket T\rrbracket$ is a strategy such that $\sigma \lhd^{\Gamma}_{T} N$. Then $\sigma \lhd^{\Gamma}_{T} M$.

Proof. Induction on T.

Suppose that $\Gamma, s \vdash M, N \vdash X$, where X is some datatype, and that $\Gamma, s \vdash M \longrightarrow \Gamma, s \vdash N$ for all Γ-stores s. Fix some Γ-store s and some strategy $\sigma : !\llbracket\Gamma\rrbracket \multimap \llbracket T\rrbracket$, and suppose that $\sigma \lhd^{\Gamma}_{T} N$.

If $\llparenthesis s, \sigma \rrparenthesis \neq \bot_{!\llbracket\Gamma\rrbracket \multimap X}$, then by hypothesis it is equal to $\llparenthesis s', u \rrparenthesis$ for some u such that $\Gamma, s \vdash N \Downarrow u, s'$. Then, by Lemma 3.2.1, $\Gamma, s \vdash M \Downarrow u, s'$.

If $\Gamma, s \vdash M, N : \mathtt{Var}$ and $\sigma \lhd^{\Gamma}_{\mathtt{Var}} N$, then we have $\sigma ; \mathrm{pr}_n \lhd^{\Gamma}_{\mathtt{com}} N \leftarrow n$ for each n and $\sigma ; \mathrm{pr}_{\mathbb{N}} \lhd^{\Gamma}_{\mathtt{nat}} !N$. If $\Gamma, s \vdash M \longrightarrow \Gamma, s \vdash N$, then $\Gamma, s \vdash M \leftarrow n \longrightarrow \Gamma, s \vdash N \leftarrow n$ for each n and $\Gamma, s \vdash !M \longrightarrow \Gamma, s \vdash !N$. Then, by the previous paragraph, we must have $\sigma ; \mathrm{pr}_n \lhd^{\Gamma}_{\mathtt{com}} M \leftarrow n$ for each n and $\sigma ; \mathrm{pr}_{\mathbb{N}} \lhd^{\Gamma}_{\mathtt{nat}} !M$, and therefore $\sigma \lhd^{\Gamma}_{\mathtt{Var}} M$.

Lastly, suppose that $\Gamma, s \vdash M, N : S \to T$ and suppose that $\Gamma, s \vdash M \longrightarrow \Gamma, s \vdash N$. Let $\sigma : !\llbracket\Gamma\rrbracket \multimap (!\llbracket S\rrbracket \multimap \llbracket T\rrbracket)$ be a strategy and suppose that $\sigma \lhd_{S \to T} N$. We claim that $\sigma \lhd_{S \to T} M$. Indeed, let $\tau : !\llbracket\Gamma\rrbracket \multimap \llbracket S\rrbracket$ be a strategy and let $\Gamma \vdash P : S$ be a term in context such that $\tau \lhd^{\Gamma}_{S} P$. Then, since $\sigma \lhd_{S \to T} N$, we must have that $\mu_{\llbracket\Gamma\rrbracket} ; (\sigma \otimes \tau^{\dagger}) ; \mathrm{ev} \lhd^{\Gamma}_{T} N\, P$. Since $\Gamma, s \vdash M \longrightarrow \Gamma, s \vdash N$, we must have $\Gamma, s \vdash M\, P \longrightarrow \Gamma, s \vdash N\, P$, and therefore $\mu_{\llbracket\Gamma\rrbracket} ; (\sigma \otimes \tau^{\dagger}) ; \mathrm{ev} \lhd^{\Gamma}_{T} M\, P$ by induction. □

Lemma 3.4.5. *Let T be a type of Idealized Algol, and let Γ be a* `Var`*-context. Then $\bot_{!\llbracket\Gamma\rrbracket \multimap \llbracket T\rrbracket} \lhd^{\Gamma}_{T} M$ for any term-in-context $\Gamma \vdash M : T$.*

Proof. Induction on T. If T is a datatype and s a store, then for any $\sigma : !\llbracket\Gamma\rrbracket \multimap \llbracket T\rrbracket$ we have

$$!\llbracket s\rrbracket ; \mathsf{cell}^{\Gamma} ; \sigma = \llparenthesis s, \sigma \rrparenthesis ; (\llbracket T\rrbracket \oslash ()) ; \mathsf{r}_{\llbracket T\rrbracket},$$

as in the proof of Proposition 3.3.7; since $(\llbracket T\rrbracket \oslash ()) ; \mathsf{r}_{\llbracket T\rrbracket}$ is a zigzag strategy, if $\sigma = \bot_{!\llbracket\Gamma\rrbracket \multimap \llbracket T\rrbracket}$, then $!\llbracket s\rrbracket ; \mathsf{cell}^{\Gamma} ; \sigma = \bot_{\llbracket T\rrbracket}$, and therefore $\llparenthesis s, \sigma \rrparenthesis = \bot_{T \oslash !\llbracket\Gamma\rrbracket}$.

Suppose $\Gamma \vdash M : \mathtt{Var}$. Since the projections are strict strategies, we have $\bot_{!\llbracket\Gamma\rrbracket \multimap \mathrm{Var}} ; \mathrm{pr}_n = \bot_{!\llbracket\Gamma\rrbracket \multimap \mathbb{C}} \lhd^{\Gamma}_{\mathtt{Var}} M$ and $\bot_{!\llbracket\Gamma\rrbracket \multimap \mathrm{Var}} ; \mathrm{pr}_{\mathbb{C}} = \bot_{!\llbracket\Gamma\rrbracket \multimap \mathbb{N}} \lhd^{\Gamma}_{\mathbb{N}} M$, and therefore $\bot_{!\llbracket\Gamma\rrbracket \multimap \mathrm{Var}} \lhd^{\Gamma}_{\mathtt{Var}} M$.

Suppose that $\Gamma \vdash M : S \to T$. Fix some $N : S$ and $\tau : !\llbracket\Gamma\rrbracket \multimap \llbracket S\rrbracket$ such that $\tau \lhd^{\Gamma}_{S} N$.

We are required to show that

$$\left(!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{\bot\otimes\tau^\dagger} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\right) \lhd^\Gamma_T M\,N\,,$$

which, by induction, we can do by showing that it is equal to $\bot_{!\,[\![\Gamma]\!]\multimap[\![T]\!]}$. Indeed, we may write this composite as

$$!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{!\,[\![\Gamma]\!]\otimes\tau^\dagger} !\,[\![\Gamma]\!] \otimes !\,[\![S]\!] \xrightarrow{\Lambda^{-1}(\bot_{!\,[\![\Gamma]\!]\multimap(!\,[\![S]\!]\multimap[\![T]\!])})} [\![T]\!]\,,$$

which is equal to $\bot_{!\,[\![\Gamma]\!]\multimap[\![T]\!]}$. □

Lemma 3.4.6. *Let T be an idealized Algol type, and let $\Gamma \vdash M\colon T$ be a typing judgement, where Γ is a* `Var`*-context. Suppose that $\sigma_1 \subseteq \sigma_2 \subseteq \cdots$ is a nested sequence of strategies for $!\,[\![\Gamma]\!] \multimap [\![T]\!]$ such that $\sigma_i \lhd^\Gamma_T M$ for all i. Let $\sigma = \bigcup_i \sigma_i$. Then $\sigma \lhd^\Gamma_T M$.*

Proof. If X is a datatype, Γ is a `Var`-context, s is a Γ-store and $\Gamma \vdash M\colon X$ is a typing judgement, then if $[\![s, \sigma_i]\!] = \bot_{X\oslash!\,[\![\Gamma]\!]}$ for all i, then we must have $[\![s, \sigma_i]\!] = \bot_{X\oslash!\,[\![\Gamma]\!]} = \bot_{X\oslash!\,[\![\Gamma]\!]}$ by continuity of composition. Otherwise, $[\![s, \sigma_i]\!] = [\![\Gamma, s' \vdash u]\!]$ for some i, some Γ-store s' and some $u \in X$ such that $\Gamma, s \vdash M \Downarrow u, s'$. In that case, since $[\![\Gamma, s' \vdash u]\!]$ is a maximal strategy for $X \oslash !\,[\![\Gamma]\!]$, we must have $[\![s, \sigma_j]\!] = [\![\Gamma, s' \vdash u]\!]$ for all $j \geq i$, and therefore that $[\![s, \sigma]\!] = [\![\Gamma, s' \vdash u]\!]$.

By induction on T, this extends to higher types by continuity of composition. For example, suppose that we have $\Gamma \vdash M\colon S \to T$, and that the σ_i are strategies $!\,[\![\Gamma]\!] \multimap (!\,[\![S]\!] \multimap [\![T]\!])$ such that $\sigma_i \lhd^\Gamma_{S\to T} M$ for all i. We claim that $\bigcup_i \sigma_i \lhd^\Gamma_{S\to T} M$.

Indeed, suppose that $\tau\colon !\,[\![\Gamma]\!] \multimap [\![S]\!]$ is a strategy and $N\colon S$ a term such that $\tau \lhd^\Gamma_S N$. Then, by hypothesis, we have

$$\left(!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{\sigma_i\otimes\tau^\dagger} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\right) \lhd^\Gamma_T M\,N$$

for each i, and so, by the inductive hypothesis applied to T, we know that we have

$$\bigcup_i \left(!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{\sigma_i\otimes\tau^\dagger} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\right) \lhd^\Gamma_T M\,N\,.$$

But continuity of composition tells us that this composite is equal to

$$\left(!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{\bigcup_i \sigma_i\otimes\tau^\dagger} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\right) \lhd^\Gamma_T M\,N\,,$$

and so the result follows. □

Now we are ready to prove our main adequacy Lemma.

Lemma 3.4.7. *Let Γ be a* `Var`*-context, let Δ be an arbitrary context and let T be an Idealized Algol type. Write $\Delta = x_1 \colon T_1, \cdots, x_n \colon T_n$. Suppose that $\sigma_i \colon !\,[\![\Gamma]\!] \multimap [\![T_i]\!]$ are strategies and $\Gamma \vdash N_i \colon T_i$ are terms-in-context such that $\sigma_i \lhd^{\Gamma}_{T_i} N_i$ for each i.*

Given a strategy $\sigma \colon !\,[\![\Gamma, \Delta]\!] \multimap [\![T]\!]$, we write

$$(\sigma_i) \fatsemi \sigma$$

for the composite

$$!\,[\![\Gamma]\!] \xrightarrow{\langle \mathtt{der}, \sigma_1, \cdots, \sigma_n^{\dagger} \rangle^{\dagger}} !\,[\![\Gamma, \Delta]\!] \xrightarrow{\sigma} [\![T]\!] \ .$$

Then for any term-in-context $\Gamma, \Delta \vdash M \colon T$, we have

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!] \lhd^{\Gamma}_{T} M[N_i/x_i] \, .$$

Proof. Induction on the typing derivation $\Gamma, \Delta \vdash M \colon T$.

- Let $x_j \colon T_j$ be a variable occurring in Δ. Then $x_j[N_i/x_i] = N_j$. Moreover, it is clear that $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash x_j]\!] = \sigma_j$. And we have $\sigma_j \lhd^{\Gamma}_{T_j} N_j$ by hypothesis.

- Next, suppose that $x \colon \mathtt{Var}$ is a variable occurring in Γ. Then we have $x[N_i/x_i] = x$, and it is easy to see that we have

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash x]\!] = [\![\Gamma \vdash x]\!] \ .$$

  Now we have $[\![\Gamma \vdash x]\!] \, ; \mathrm{pr}_n = [\![\Gamma \vdash x \leftarrow n]\!]$ and $[\![\Gamma \vdash x]\!] \, ; \mathrm{pr}_{\mathbb{N}} = [\![\Gamma \vdash !\mathbb{N}]\!]$. From the proof of Lemma 3.3.5 (in particular, Figures 3.8 and 3.9), we know that if s is a Γ-store, then

  $$\langle\!\langle s, [\![\Gamma \vdash x \leftarrow n]\!] \rangle\!\rangle = \langle\!\langle \Gamma, s \vdash x \leftarrow n \rangle\!\rangle = \langle\!\langle \Gamma, (s | x \mapsto n) \vdash \mathsf{skip} \rangle\!\rangle \, ,$$

  where $\Gamma, s \vdash x \leftarrow n \Downarrow \mathsf{skip}, (s | x \mapsto n)$, and that

  $$\langle\!\langle s, [\![\Gamma \vdash !x]\!] \rangle\!\rangle = \langle\!\langle \Gamma, s \vdash !x \rangle\!\rangle = \langle\!\langle \Gamma, s \vdash s(x) \rangle\!\rangle \, ,$$

  where $\Gamma, s \vdash !x \Downarrow s(x), s$.

- Next, suppose that $\Gamma, \Delta \vdash M \colon S \to T$ and $\Gamma, \Delta \vdash N \colon S$, where we already know that

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!] \lhd^{\Gamma}_{S \to T} M[N_i/x_i]$$

and that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash N]\!] \lhd^{\Gamma}_{S} N[N_i/x_i]\,.$$

Then it easy to see that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M\,N]\!]$$

is given by the composite

$$!\,[\![\Gamma]\!] \xrightarrow{\mu_{[\![\Gamma]\!]}} !\,[\![\Gamma]\!] \otimes !\,[\![\Gamma]\!] \xrightarrow{((\sigma_i)\fatsemi[\![\Gamma,\Delta\vdash M]\!])\otimes((\sigma_i)\fatsemi[\![\Gamma,\Delta\vdash N]\!])} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\,.$$

Therefore, by the definition of $\lhd^{\Gamma}_{S\to T}$, it must be the case that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M\,N]\!] \lhd^{\Gamma}_{T} M[N_i/x_i]\,N[N_i/x_i] = (M\,N)[N_i/x_i]\,.$$

- Next, suppose that $\Gamma, \Delta, x\colon S \vdash M\colon T$. We claim that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta, \lambda x.M]\!] \lhd^{\Gamma}_{S\to T} (\lambda x.M)[N_i/x_i]\,.$$

Let $\sigma'\colon !\,[\![\Gamma]\!] \multimap [\![S]\!]$ be a strategy and let $\Gamma \vdash N\colon S$ be a term in context such that $\sigma \lhd^{\Gamma}_{S} N$. Then, by the inductive hypothesis (with x lying in the 'Δ-part', we know that

$$(\sigma_i, \sigma') \fatsemi [\![\Gamma, \Delta, x\colon S \vdash M\colon T]\!] \lhd^{\Gamma}_{T} M[N_i/x_i, N'/x]\,.$$

Then, since $\Gamma, s \vdash (\lambda x.M[N_i/x_i])\,N' \longrightarrow \Gamma, s \vdash M[N_i/x_i, N'/x]$ for any Γ-store s, by Lemma 3.4.4 we know that

$$(\sigma_i, \sigma') \fatsemi [\![\Gamma, \Delta, x\colon S \vdash M]\!] \lhd^{\Gamma}_{T} ((\lambda x.M)[N_i/x_i])\,N'\,.$$

Finally, observe that $(\sigma_i, \sigma') \fatsemi [\![\Gamma, \Delta, x\colon S \vdash M\colon T]\!]$ is the composite

$$!\,[\![\Gamma]\!] \xrightarrow{\langle \mathtt{der}, \sigma_1, \cdots, \sigma_n, \sigma' \rangle^{\dagger}} ![\![\Gamma, \Delta, x]\!] \xrightarrow{[\![\Gamma, \Delta, x\colon S \vdash M]\!]} [\![T]\!]\ ,$$

which can alternatively be written as

$$!\,[\![\Gamma]\!] \xrightarrow{\langle \mathtt{der}, \sigma_1, \cdots, \sigma_n \rangle^{\dagger}} ![\![\Gamma, \Delta]\!] \xrightarrow{\mu_{[\![\Gamma,\Delta]\!]}} ![\![\Gamma, \Delta]\!] \otimes ![\![\Gamma, \Delta]\!]$$

$$\xrightarrow{[\![\Gamma,\Delta\vdash\lambda x.M]\!]\otimes(\sigma')^{\dagger}} (!\,[\![S]\!] \multimap [\![T]\!]) \otimes !\,[\![S]\!] \xrightarrow{\mathrm{ev}} [\![T]\!]\ .$$

Now we move on to the structural constants for the datatypes. We deal with the case of the type `nat` (i.e., the terms n, $\mathsf{succ}\,M$, $\mathsf{pred}\,M$ and $\mathsf{If0}\,M\ \mathsf{then}\ N\ \mathsf{else}\ P$), and it should be clear how our ideas translate to the corresponding rules at the types `com` and `bool` (i.e., skip, $\mathtt{t}/\mathtt{f}$, sequencing and the boolean conditional).

Before we cover these cases, we make an important observation. Suppose that $\Gamma, \Delta \vdash M, N \colon T$, and that

$$\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]) \mathbb{]\!]} = \mathbb{[\![} s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash N]\!] \mathbb{]\!]} \,.$$

Let $\mathbf{E}$ be an evaluation context of type U with a hole of type T. Then

$$\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathbf{E}[M]]\!]) \mathbb{]\!]} = \mathbb{[\![} s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathbf{E}[N]]\!] \mathbb{]\!]} \,.$$

We can prove this by precomposing both directions of the diagram in Figure 3.10 (with Γ in that diagram standing for Γ, Δ) with the composite

$$I \xrightarrow{!\llbracket s \rrbracket} !S_\Gamma \xrightarrow{\mathsf{cell}^\Gamma} ! \, [\![\Gamma]\!] \xrightarrow{\langle \mathtt{der}, \sigma_1, \cdots, \sigma_n \rangle^\dagger} ! \, [\![\Gamma, \Delta]\!] \ ,$$

in order to prove that the value of $\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathbf{E}[M]]\!] \mathbb{]\!]}$ depends only on the value of $\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]) \mathbb{]\!]}$ and $\mathbf{E}$.

- Given a numeral n and a Γ-store s, we have $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash n]\!] = [\![\Gamma \vdash n]\!]$ and $\mathbb{[\![} s, [\![\Gamma \vdash n]\!] \mathbb{]\!]} = \mathbb{[\![} \Gamma, s \vdash n \mathbb{]\!]}$, where $\Gamma, s \vdash n \Downarrow n, s$.
- Suppose $\Gamma, \Delta \vdash M : \mathtt{nat}$, and that we already know that

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!] \lhd^\Gamma_{\mathtt{nat}} M[N_i/x_i] \,.$$

  So for all Γ-stores s, either $\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]) \mathbb{]\!]} = \bot_{\mathbb{N} \oslash ! [\![\Gamma]\!]}$ or it is equal to $\mathbb{[\![} \Gamma, s' \vdash n \mathbb{]\!]}$ for some numeral n.

  Now we have

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{succ}\, M]\!] = (\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!] \,; (\mathsf{succ} \otimes ! \, [\![\Gamma]\!]) \,;$$

  since $\mathsf{succ} \colon \mathbb{N} \multimap \mathbb{N}$ is a strict strategy, we know that either

  $$\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{succ}\, M]\!]) \mathbb{]\!]} = \bot_{\mathbb{N} \oslash ! [\![\Gamma]\!]}$$

  or

  $$\mathbb{[\![} s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{succ}\, M]\!]) \mathbb{]\!]} = \mathbb{[\![} \Gamma, s' \vdash n \mathbb{]\!]} ; (\mathsf{succ} \otimes ! \, [\![\Gamma]\!]) = \mathbb{[\![} \Gamma, s' \vdash n+1 \mathbb{]\!]} \,,$$

  and we have $\Gamma, s \vdash \mathsf{succ}\, M[N_i/x_i] \Downarrow n+1$. Therefore,

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{succ}\, M]\!] \lhd^\Gamma_{\mathtt{nat}} \mathsf{succ}\, M[N_o/x_i] \,.$$

  A similar argument proves that

  $$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{pred}\, M]\!] \lhd^\Gamma_{\mathtt{nat}} \mathsf{pred}\, M[N_i/x_i] \,.$$

- Now we move on to the conditional. Suppose that we have $\Gamma, \Delta \vdash M : \mathtt{nat}$ and $\Gamma, \Delta \vdash N, P : X$ for some datatype X. Let s be a Γ-store. Since $\mathsf{If0}$ is a strict strategy, if $[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!])]\!] = \bot_{\mathbb{N} \oslash ![\![\Gamma]\!]}$, then

$$[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{If0}\, M \;\mathsf{then}\; N \;\mathsf{else}\; P]\!]\,]\!] = \bot_{X \oslash ![\![\Gamma]\!]}\,.$$

  Otherwise, by induction we know that $[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]\,]\!] = [\![\Gamma, s' \vdash n]\!]$ for some numeral n such that $\Gamma, s \vdash M[N_i/x_i] \Downarrow n, s'$. Without loss of generality, suppose that $n = 0$. Then we have

$$\begin{aligned} &[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{If0}\, M \;\mathsf{then}\; N \;\mathsf{else}\; P]\!])]\!] \\ &= [\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{If0}\, 0 \;\mathsf{then}\; N \;\mathsf{else}\; P]\!])]\!] \\ &= [\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash N]\!])]\!]\,, \end{aligned}$$

  where the first equality is by our observation above (since $\mathsf{If0} - \mathsf{then}\ N\ \mathsf{else}\ P$ is an evaluation context), and the second is by the definition of $\mathsf{If0}$. Then, by induction, this last term is either equal to $\bot_{X \oslash [\![\Gamma]\!]}$,or is equal to $[\![\Gamma, s'' \vdash u]\!]$ for some $u \in X$ such that $\Gamma, s' \vdash N[N_i/x_i] \Downarrow u, s''$. In this second case, we have $\Gamma, s \vdash \mathsf{If0}\, M[N_i/x_i] \;\mathsf{then}\; N[N_i/x_i] \;\mathsf{else}\; P[N_i/x_i] \Downarrow u, s''$.

The argument for let is similar.

- Suppose that $\Gamma, \Delta \vdash M : Y$ and $\Gamma, \Delta, x : Y \vdash N : X$, where Y is either $\mathtt{bool}$ or $\mathtt{nat}$ and X is either $\mathtt{bool}$, $\mathtt{nat}$ or $\mathtt{com}$. Let s be a Γ-store. Since $\mathsf{let} : Y \multimap ((Y \to X) \to X)$ is a strict strategy, if $[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!])]\!] = \bot_{Y \oslash ![\![\Gamma]\!]}$, then

$$[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{let}\, x = M \;\mathsf{in}\; N]\!]\,]\!] = \bot_{X \oslash ![\![\Gamma]\!]}\,.$$

  Otherwise, by induction we must have $[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]\,]\!] = [\![\Gamma, s' \vdash y]\!]$ for some $y \in Y$ such that $\Gamma, s \vdash M[N_i/x_i] \vdash y, s'$. Then we have

$$\begin{aligned} &[\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{let}\, x = M \;\mathsf{in}\; N]\!])]\!] \\ &= [\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{let}\, x = y \;\mathsf{in}\; N]\!]\,]\!] \\ &= [\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash (\lambda x.N)y]\!]\,]\!]\,, \\ &= [\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash N[y/x]]\!]\,]\!]\,, \end{aligned}$$

  where the first inequality holds because $\mathsf{let}\, x = -\ \mathsf{in}\ N$ is an evaluation context, and the second holds by the definition of let. Then, by induction, this last term is either equal to $\bot_{X \oslash ![\![\Gamma]\!]}$ or it is equal to $[\![\Gamma, s'' \vdash u]\!]$ for some $u \in X$ such that

$$\Gamma, s' \vdash N[y/x, N_i/x_i] \vdash u, s''\,.$$

In this second case, we have

$$\Gamma, s \vdash \mathsf{let}\, x = M[N_i/x_i]\ \mathsf{in}\ N[N_i/x_i] \Downarrow u, s'' .$$

Next, we deal with the rules for variables.

- Suppose $\Gamma, \Delta \vdash V \colon \mathtt{Var}$, and suppose we already know that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V]\!] \lhd^{\Gamma}_{\mathtt{Var}} V[N_i/x_i] .$$

Then by definition we have

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash\, !V]\!] = (\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V]\!] ; \mathrm{pr}_{\mathbb{N}} \lhd^{\Gamma}_{\mathbb{N}} !\mathcal{V}[N_i/x_i] .$$

Suppose $\Gamma, \Delta \vdash E \colon \mathtt{nat}$, and suppose we know already that $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash E]\!] \lhd_{\mathbb{N}} E[N_i/x_i]$. Then, for all s, either $[\!\![s, (\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash E]\!]]\!] = \bot_{\mathbb{N}\oslash !\llbracket\Gamma\rrbracket}$ – in which case

$$[\!\![s, (\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V \leftarrow E]\!]]\!] = \bot_{\mathbb{C}\oslash !\llbracket\Gamma\rrbracket}$$

(since assign is a strict strategy) – or it is equal to $[\!\![s', n]\!]$ for some $n \in \mathbb{N}$ such that $\Gamma, s \vdash E[N_i/x_i] \Downarrow n, s'$. In the second case, we have

$$\begin{aligned} & [\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V \leftarrow E]\!])]\!] \\ = {} & [\!\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V \leftarrow n]\!]]\!] \\ = {} & [\!\![s', ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V]\!] ; \mathrm{pr}_n)]\!] . \end{aligned}$$

Now we have $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash V]\!] ; \mathrm{pr}_n \lhd^{\Gamma}_{\mathtt{com}} V[N_i/x_i] \leftarrow n$ by induction, so this composite is either equal to $\bot_{\mathbb{C}\oslash !\llbracket\Gamma\rrbracket}$ or it is equal to $[\!\![\Gamma, s'' \vdash \mathsf{skip}]\!]$ for s'' such that $\Gamma, s' \vdash V[N_i/x_i] \leftarrow n \Downarrow \mathsf{skip}, s''$. In this second case, we have $\Gamma, s \vdash V[N_i/x_i] \leftarrow E[N_i/x_i] \Downarrow \mathsf{skip}, s''$.

- Suppose that $\Gamma, \Delta \vdash W \colon \mathtt{nat} \to \mathtt{com}$ and that $\Gamma, \Delta \vdash R \colon \mathtt{nat}$. We claim that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{mkvar}\ W\ R]\!] ; \mathrm{pr}_n \lhd^{\Gamma}_{\mathtt{com}} (\mathsf{mkvar}\ W[N_i/x_i]\, R[N_i/x_i]) \leftarrow n$$

for each n and that

$$(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{mkvar}\ W\ R]\!] ; \mathrm{pr}_{\mathbb{N}} \lhd^{\Gamma}_{\mathtt{nat}} !(\mathsf{mkvar}\ W[N_i/x_i]\, R[N_i/x_i]) .$$

For the first of these, let s be a Γ-store. We have

$$\begin{aligned} & [\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{mkvar}\ W\ R]\!] ; \mathrm{pr}_n)]\!] \\ = {} & [\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash W\ n]\!])]\!] . \end{aligned}$$

By induction, we have $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash W\, n]\!] \lhd^{\Gamma}_{\mathtt{com}} W[N_i/x_i]\, n$. Therefore, this last composite is either equal to $\bot_{\mathbb{C}\oslash !\llbracket\Gamma\rrbracket}$, or it is equal to $[\!\![\Gamma, s', \mathsf{skip}]\!\!]$ for some s' such that $\Gamma, s \vdash W\, n \Downarrow \mathsf{skip}, s'$. In this second case, we have $\Gamma, s \vdash (\mathsf{mkvar}\, W\, R) \leftarrow n \Downarrow \mathsf{skip}, s'$.

For the second, we have

$$\begin{aligned} &[\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{mkvar}\ W\, R]\!]\, ; \mathrm{pr}_{\mathbb{N}})]\!\!] \\ &= [\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash R]\!])]\!\!]\,. \end{aligned}$$

By induction, we have $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash R]\!] \lhd^{\Gamma}_{\mathtt{nat}} R[N_i/x_i]$. Therefore, this last composite is either equal to $\bot_{\mathbb{C}\oslash !\llbracket\Gamma\rrbracket}$, or it is equal to $[\!\![\Gamma, s', n]\!\!]$ for some s', n such that $\Gamma, s, R \Downarrow n, s'$. In this second case, we have $\Gamma, s \vdash !(\mathsf{mkvar}\, W\, R) \Downarrow n, s'$.

- Suppose that $\Gamma, x\colon \mathtt{Var}, \Delta \vdash M\colon X$, where X is a datatype, and suppose we already know that

$$(\sigma_i) \fatsemi [\![\Gamma, x\colon \mathtt{Var}, \Delta \vdash M]\!] \lhd^{\Gamma, x}_{X} M\,,$$

where x is considered as belonging to the 'Γ-component'. Let s be a Γ-store – then we get a Γ, x-store $(s|x \mapsto 0)$. So either $[\!\![(s|x \mapsto 0), ((\sigma_i) \fatsemi [\![\Gamma, x, \Delta \vdash M]\!])]\!\!] = \bot_{X\oslash !\llbracket\Gamma\rrbracket}$, or it is equal to $[\!\![\Gamma, x, (s'|x \mapsto n), u]\!\!]$ for some s', n, u such that $\Gamma, (s|x \mapsto 0) \vdash M[N_i/x_i] \Downarrow u, (s'|x \mapsto n)$.

But by the definition of `new`, we have

$$\begin{aligned} &[\!\![(s|x \mapsto 0), ((\sigma_i) \fatsemi [\![\Gamma, x, \Delta \vdash M]\!])]\!\!]; (X \oslash !\,\mathrm{pr}_{\Gamma}) \\ &= [\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{new}\, \lambda x.M]\!])]\!\!]\,. \end{aligned}$$

It follows that either $[\!\![s, ((\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash \mathsf{new}\, \lambda x.M]\!])]\!\!] = \bot_{X\oslash !\llbracket\Gamma\rrbracket}$ or it is equal to $[\!\![\Gamma, x, (s'|x \mapsto n), u]\!\!]; (X \oslash !\,\mathrm{pr}_{\Gamma}) = [\!\![\Gamma, s', u]\!\!]$ for some u such that $\Gamma, (s|x \mapsto 0) \vdash M[N_i/x_i] \Downarrow u, (s'|x \mapsto n)$. In this case, $\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow u, s'$.

It remains to deal with the case of the recursion combinator $\mathbf{Y}_T$. First recall that if $h\colon ((\llbracket T\rrbracket \to \llbracket T\rrbracket) \to \llbracket T\rrbracket) \to (\llbracket T\rrbracket \to \llbracket T\rrbracket) \to T$ (where we write $A \to B$ for $!A \multimap B$) is the morphism corresponding to the λ-term

$$\lambda F^{(T\to T)\to T}.\lambda f^{T\to T}.f(F\, f)\,,$$

then $\mathbf{Y}_T$ is constructed as the limit of the chain

$$\bot_{(T\to T)\to T} \subseteq \bot_{(T\to T)\to T}; h \subseteq \bot_{(T\to T)\to T}; h; h \subseteq \cdots\,.$$

By Lemma 3.4.6, then, it suffices to show that if $\Gamma, \Delta \vdash M\colon T \to T$ then $(\sigma_i) \fatsemi [\![\Gamma, \Delta \vdash M]\!]\, ; \left(\bot_{(T\to T)\to T}\right); h^n \lhd^{\Gamma}_{T} \mathbf{Y}_T M[N_i/x_i]$ for each n.

We do this by induction on n. Lemma 3.4.5 takes care of the initial case, and it will therefore suffice to prove that if $\sigma \lhd^{\Gamma}_{T} \mathbf{Y}_T M$ then $\sigma; h \lhd^{\Gamma}_{T} \mathbf{Y}_T M$.

Suppose $T = T_1 \to \cdots \to T_n \to X$, where X is a datatype. Fix terms $M_i \colon T_i$ and strategies $\tau_i \colon [\![\Gamma]\!] \to [\![T_i]\!]$ such that $\tau_i \lhd^{\Gamma}_{T_i} M_i$ for each i.

Since $\sigma \lhd^{\Gamma}_{T} \mathbf{Y}_T M$, we know that $\langle \tau_1, \cdots, \tau_n \rangle; \sigma \lhd^{\Gamma}_{X} \mathbf{Y}_T M\, M_1 \cdots M_n$. It follows that $\langle \tau_1, \cdots, \tau_n \rangle; \sigma; h \lhd^{\Gamma}_{X} M(\mathbf{Y}_T M)\, M_1, \cdots M_n$. Then, since

$$\Gamma, s, \mathbf{Y}_T M\, M_1, \cdots M_n \longrightarrow \Gamma, s, M(\mathbf{Y}_T M)\, M_1 \cdots \mathcal{M}_n$$

for any Γ-store s, by Lemma 3.4.4 we have that

$$\langle \tau_1, \cdots, \tau_n \rangle; \sigma; h \lhd^{\Gamma}_{X} \mathbf{Y}_T M\, M_1, \cdots, M_n\,,$$

and therefore that $\sigma; h \lhd^{\Gamma}_{X} \mathbf{Y}_T M$, since the M_i, τ_i were arbitrary. □

This is now enough to prove a *computational adequacy result* for our denotational semantics.

Definition 3.4.8 (Computational Adequacy Result)**.** Suppose we have a programming language $\mathcal{L}$ and a denotational semantics $[\![-]\!]$ of $\mathcal{L}$ in some category $\mathcal{C}$. Let o be a ground type of $\mathcal{L}$ and let $\Downarrow$ be some operational predicate on closed terms of $\mathcal{L}$ of type o. Let $\downarrow$ be a predicate on morphisms $1 \to [\![L]\!]$ in $\mathcal{C}$. Then a *computational adequacy* result for the semantics is a result that says that for all terms $M \colon o$ in $\mathcal{L}$,

$$M \Downarrow \text{ if and only if } [\![M]\!] \downarrow\,.$$

We will prove a computational adequacy result for our semantics of IA, taking $o = \mathtt{com}$. If $M \colon \mathtt{com}$ is a term of IA, we say that $M \Downarrow$ if $\_, () \vdash M \Downarrow \mathsf{skip}, ()$. If $\sigma \colon 1 \multimap \mathbb{C}$ is a strategy, we say that $\sigma \downarrow$ if $\sigma \neq \bot_{\mathtt{com}}$.

Theorem 3.4.9 (Computational Adequacy)**.** *Let $M \colon \mathtt{com}$ be a closed term of Idealized Algol. Then $M \Downarrow$ if and only if $[\![M]\!] \neq \bot_{\mathtt{com}}$.*

Proof. First suppose that $M \Downarrow$. Then Proposition 3.3.7 tells us that $[\![M]\!] = [\![\mathsf{skip}]\!] \neq \bot_{\mathtt{com}}$.

Conversely, suppose that $[\![M]\!] \neq \bot_{\mathtt{com}}$. Lemma 3.4.7 tells us that $[\![M]\!] \lhd_{\mathtt{com}} M$. Since $[\![M]\!] \neq \bot_{\mathtt{com}}$, the only possibility is that $\_, () \vdash M \Downarrow \mathsf{skip}, ()$. □

We can now prove an *equational soundness* result. First, we recall the definition of *observational equivalence* of terms.

Definition 3.4.10 (Observational Equivalence)**.** Suppose we have a language $\mathcal{L}$, together with a distinguished ground type o and operational predicate $\Downarrow$ as above. Given two closed terms $M, N : T$ in $\mathcal{L}$, we say that M and N are *observationally equivalent* if for all contexts $C[-] : o$ in $\mathcal{L}$ with a hole of type T, $C[M] \Downarrow$ if and only if $C[N] \Downarrow$.

Next, we create a definition that mirrors this one within the denotational semantics.

Definition 3.4.11 (Intrinsic Equivalence)**.** Suppose we have a language $\mathcal{L}$, together with a denotational semantics $[\![-]\!]$ in a Cartesian closed category $\mathcal{C}$, a distinguished ground type o and a predicate $\downarrow$ on morphisms $1 \to [\![o]\!]$ as above.

Given objects A, B of $\mathcal{C}$, and morphisms $\sigma, \tau : A \to B$, we say that σ, τ are *intrinisically equivalent*, and write $\sigma \sim \tau$, if for all $\alpha : (A \to B) \to [\![o]\!]$, $\Lambda(\sigma); \alpha \downarrow$ if and only if $\Lambda(\tau); \alpha \downarrow$.

Definition 3.4.12 (Equational Soundness Result)**.** An *equational soundness result* for a semantics as above is a result that says that if $M, N : T$ are such that $[\![M]\!] \sim [\![N]\!]$, then M and N are observationally equivalent.

Proposition 3.4.13. *Any compositional semantics that satisfies Computational Adequacy satisfies Equational Soundness.*

Proof. Suppose that $[\![M]\!] \sim [\![N]\!]$. Let $C[-] : o$ be a context with a hole of type T. Then, since $\mathcal{C}$ is Cartesian closed, the β-rule is valid in $\mathcal{C}$, and therefore we have

$$[\![C[M]]\!] = 1 \xrightarrow{[\![M]\!]} [\![T]\!] \xrightarrow{[\![x : T \vdash C[x]]\!]} [\![o]\!] \ ,$$

and similarly for $C[N]$. Therefore, taking $\alpha = [\![x \vdash C[x]]\!]$ in the definition of $\sim$, we get that $[\![C[M]]\!] \downarrow$ if and only if $[\![C[N]]\!] \downarrow$. By Computational Adequacy, this means that $C[M] \Downarrow$ if and only if $C[N] \Downarrow$. Since C was arbitrary, it follows that M and N are observationally equivalent. □

This proof relies on the fact that the context C gives rise to a morphism $\alpha = [\![x \vdash C[x]]\!]$. To go in the other direction (full abstraction), we need a *definability* result, to allow us to transform a morphism in $\mathcal{G}$ into an Idealized Algol context. This will be the subject of the next section.

3.5 Innocent Factorization

The aim of this section will be to show that any strategy between the denotations of Idealized Algol types may be factorized as the composite of some

innocent strategy with the strategy cell. This will reduce the problem to showing that certain innocent strategies are definable, for which we can use known results.

Again, we follow the proof from [AM96] very closely.

Proposition 3.5.1 (Innocent Factorization, [AM96, 14])**.** *Let A be a game, and suppose A has the property that whenever $sb \in P_A$ is an O-position and $tbc \in P_A$ is a P-position such that $\ulcorner sb \urcorner = \ulcorner tb \urcorner$, then $sbc \in P_A$. Suppose also that M_A is a countable set. Let σ be a strategy for A. Then there is an innocent strategy $\hat{\sigma}$ for $!\,\mathrm{Var} \multimap A$ such that σ factorizes as the composite*

$$I \xrightarrow{!0} !\mathbb{N} \xrightarrow{\mathsf{cell}} !\,\mathrm{Var} \xrightarrow{\hat{\sigma}} A\,.$$

Proof. Fix some injection $\mathcal{L}_A \hookrightarrow \mathbb{N}$, thought of as giving a 'code' to each legal sequence, such that the code of the empty sequence is 0. $\hat{\sigma}$ proceeds as follows. At each position sb ending with an O-move a in A, player P first plays the move q in $!\,\mathrm{Var}$, after which player O returns some natural number n. If n is not the code of a sequence in P_A, or if it encodes some sequence t such that $\ulcorner tb \urcorner \neq \ulcorner s|_A b \urcorner$, then player P has no reply. Otherwise, let c be the reply to ta that player P would have made under σ. Player P computes the code k of the sequence tbc and plays q_k in $!\,\mathrm{Var}$. After the O-reply a, player P then plays the move b in $!\,\mathrm{Var}$.

Since $tbc \in P_A$ and $\ulcorner tb \urcorner = \ulcorner tc \urcorner$, we have $s|_A bc \in P_A$ by our hypothesis on A, which implies that this is a valid strategy. We claim that it is innocent. Moreover, since player P's moves depend only on the number n returned and on the current P-view within A, this strategy is innocent.

Lastly, if we compose with $!0; \mathsf{cell}$, then we ensure that this number n that is returned is always the code of the current sequence in A, and therefore that $!0; \mathsf{cell}; \hat{\sigma} = \sigma$ by the explicit description of the cell strategy given in Proposition 2.17.2. □

We want to apply this result in the case that A is the denotation of an IA type, so we need to show that for any IA type T, the game $[\![T]\!]$ satisfies the hypotheses of Proposition 3.5.1. In fact, we show something stronger.

Lemma 3.5.2 ([AM96, Lemma 15])**.** *Let T be a type of Idealized Algol. Then $P_{[\![T]\!]}$ is the set of all sequences in $\mathcal{L}_{[\![T]\!]}$ that contain at most a single initial move.*

Proof. This is obvious for the datatypes and for `Var`. Let S, T be types – so $[\![S \to T]\!] = !\,[\![S]\!] \multimap [\![T]\!]$ – and let $s \in P_{!\,[\![S]\!] \multimap [\![T]\!]}$. An initial move in

$!\llbracket S \rrbracket \multimap \llbracket T \rrbracket$ is an initial move in $\llbracket T \rrbracket$; by induction, $s|_{\llbracket T \rrbracket} \in P_{\llbracket T \rrbracket}$ contains at most one initial move, and so s contains at most one initial move.

Conversely, suppose that $s \in \mathcal{L}_{!\llbracket S \rrbracket \multimap \llbracket T \rrbracket}$ contains at most one initial move. We show by induction on the length of s that $s \in P_{!\llbracket S \rrbracket \multimap \llbracket T \rrbracket}$. This is obvious if $s = \epsilon$; if $s = ta$ for some t, then by induction we have $t \in P_{!\llbracket S \rrbracket \multimap \llbracket T \rrbracket}$, so $t|_{!\llbracket S \rrbracket} \in P_{!\llbracket S \rrbracket}$ and $t|_{\llbracket T \rrbracket} \in P_{\llbracket T \rrbracket}$. We need to show that $ta \in P_{!\llbracket S \rrbracket}$, for which it suffices by induction to show that $ta|_{!\llbracket S \rrbracket}$ and $ta|_{\llbracket T \rrbracket}$ are both legal positions, $ta|_{!\llbracket S \rrbracket}|_n$ has a unique initial move in $\llbracket S \rrbracket$ for each initial move n of $\llbracket S \rrbracket$, and $ta|_{\llbracket T \rrbracket}$ has a unique initial move in $\llbracket T \rrbracket$. Indeed, we can show that they are alternating using the same arguments we used in Proposition 2.4.9, and they are clearly well-bracketed sequences. Visibility then follows from Lemma 2.4.8, and from Proposition 4.3 from [HO00], which states that if s is a play in $A \multimap B$, then $\ulcorner s^{\urcorner A \multimap B}|_A$ is a subsequence of $\ulcorner s|_A{}^{\urcorner A}$ and $\ulcorner s^{\urcorner A \multimap B}|_B$ is a subsequence of $\ulcorner s|_B{}^{\urcorner B}$. This means that the justifier of the move a must occur within the appropriate O- or P-view, since we know that $t|_{!\llbracket S \rrbracket}$ and $t|_{\llbracket T \rrbracket}$ are both visible sequences. □

Proposition 3.5.3. *Let T be a type of Idealized Algol. Then $\llbracket T \rrbracket$ satisfies the hypotheses of Proposition 3.5.1.*

Proof. Suppose that $sb, tbc \in P_{\llbracket T \rrbracket}$, where c is a P-move, are such that $\ulcorner sb \urcorner = \ulcorner tb \urcorner$. It is clear that sbc is an alternating and visible sequence; moreover, it is well-bracketed, since the most recently unanswered question in sb is the same as in tb. Therefore, sbc is a legal sequence with a unique initial move, and so it is contained in $P_{\llbracket T \rrbracket}$ by Lemma 3.5.2. □

3.6 Arena-only Semantics

Before we complete our definability proof, and hence our proof of Full Abstraction, we take a detour to consider a variation of our game semantics in which we do away with the sets of plays P_A. We call this an *arena-only semantics*, since the objects of our category are plain arenas. Since the semantics given in [HO00] are of this form, and since we want to quote definability results from this work, it makes sense to cast our own work in this form.

Definition 3.6.1. Let A be a game. We say that A is *full* if $P_A = \mathcal{L}_A$.

Let A be a game. Since the underlying arena of $!A$ is the same as that of A, and since we have $P_A \subseteq P_{!A}$, we have the following result.

Proposition 3.6.2. *If A is full, then $A = !A$.*

Note that the definition of the ! connective, together with Lemma 3.5.2, implies the following.

Proposition 3.6.3. *Let T be an Idealized Algol type. Then $!\llbracket T \rrbracket$ is full.*

This does not immediately seem very useful: $!\llbracket T \rrbracket$ may be full, but $\llbracket T \rrbracket$ itself is not, and $\llbracket T \rrbracket$ is the game that we are using to model the type T in our semantics. Recall, however, that the 'co-Kleisli'-style morphisms in our category, which are strategies for $!A \multimap B$, may alternatively be regarded as comonoid homomorphisms from the comonoid on $!A$ to the comonoid on $!B$. Thus, if S, T are Idealized Algol types, then the set of morphisms $!\llbracket S \rrbracket \multimap \llbracket T \rrbracket$ in our original category is precisely the same as the set of morphisms $!\llbracket S \rrbracket \multimap !\llbracket T \rrbracket$ that are comonoid homomorphisms. We will take up the rest of this section giving a combinatorial characterization of these morphisms.

Definition 3.6.4. Let A be a game and let sa be a play in A. Then the *current thread* $\lceil sa \rceil$ of sa is $sa|_n$, where n is the unique initial move that hereditarily justifies a.

Let σ be a strategy for A. We say that σ is *single-threaded* if player P's moves only depend on the current thread; i.e., if whenever $sab \in \sigma$, $t \in \sigma$ and $ta \in P_A$ such that $\lceil sa \rceil = \lceil ta \rceil$, then we have $tab \in \sigma$.

Note that this definition is identical to the definition of an innocent strategy, except with the current thread $\lceil - \rceil$ taking the role of the P-view $\ulcorner - \urcorner$. Since $\ulcorner s \urcorner$ is a subsequence of $\lceil s \rceil$ for any sequence s, we get:

Proposition 3.6.5. *Any innocent strategy is single-threaded.*

We now make a new definition.

Definition 3.6.6. Given a legal sequence $s \in \mathcal{L}_A$, and a move b in s, we say that a move a in s is *hereditarily justified by* b if there is a chain of justification pointers going back from a to b.

We write $s|_b$ for the subsequence of s given by all moves in s that are hereditarily justified by b. Given a set I of initial moves, we write $s|_I$ for the subsequence of s given by all moves that are hereditarily justified by some $b \in I$.

We say that a game A is *thread-closed* if $s|_I \in P_A$ whenever $s \in P_A$ and I is some collection of occurrences of initial moves in s.

If A and B are non-empty games, then the sequoid $A \oslash B$ is never thread-closed: if we take some play s with moves in both A and B, then $s|_B$ is not a play of $A \oslash B$.

However, if A is a well-opened game, then $!A$ is always thread-closed.

Lemma 3.6.7 ([Har99, 3.5.1])**.** *Let A be a thread-closed game and let $\sigma : A$ be a single-threaded strategy. Suppose $s \in \sigma$, and let I be a set of occurrences of initial moves in s. Then $s|_I \in \sigma$.*

Proof. Induction on the length of s. Suppose that $sab \in \sigma$. Since sab is a visible sequence, we know that the justifier of b occurs in $\ulcorner sa \urcorner$, and in particular in $\lceil sa \rceil$. Now a must be hereditarily justified by that initial move that hereditarily justifies b; if this initial move is not contained in I, then we have $sab|_I = s|_I \in \sigma$ by induction. Otherwise, we have $sab|_I = s|_I ab$. Then, since we have $\lceil sa \rceil = \lceil s|_I a \rceil$, we must have $s|_I ab \in \sigma$ by single-threadedness of σ. □

We now link the concept of single-threadedness to the idea of a comonoid homomorphism.

Proposition 3.6.8. *Let A, B be well-opened games. Given a strategy*

$$\sigma : !A \multimap !B\,,$$

σ is single-threaded if and only if it is a comonoid homomorphism with respect to the natural comonoid structures on $!A$ and $!B$; i.e., it makes the following diagrams commute.

$$\begin{array}{ccc} !A & \xrightarrow{\sigma} & !B \\ {\scriptstyle \mu_A}\downarrow & & \downarrow{\scriptstyle \mu_B} \\ !A \otimes !A & \xrightarrow{\sigma \otimes \sigma} & !B \otimes !B \end{array} \qquad \begin{array}{ccc} !A & \xrightarrow{\sigma} & !B \\ {\scriptstyle ()}\downarrow & \swarrow_{()} & \\ I & & \end{array}$$

Proof. Of course, the second diagram is automatically satisfied by any strategy σ, since I is a terminal object.

Since A, B are well-opened, $!A$ and $!B$ are the cofree commutative comonoids on A, B, and therefore the comonoid homomorphisms $\sigma : !A \multimap !B$ are precisely the strategies of the form $\tau^\dagger$, where $\tau : !A \multimap B$ is an arbitrary morphism. It is easy to check that such a strategy is single-threaded; indeed, the plays of $\tau^\dagger$ are precisely those plays $s \in P_{!A \multimap !B}$ such that $s|_n \in \tau$ for any occurrence n of an initial move in s.

Conversely, let $\sigma : !A \multimap !B$ be single-threaded. Let $\tau = \sigma \cap P_{!A \multimap B}$ – so τ consists of all sequences in σ that have at most one initial move. Then τ is clearly a strategy, since it is a prefix-closed subset of σ. We claim that $\sigma = \tau^\dagger$. Indeed, suppose that $s \in \sigma$. Let n be an occurrence of an initial move in s. Then, by Lemma 3.6.7, $s|_n \in \sigma$ and has a unique initial move, and is therefore contained in τ. Therefore, $\sigma = \tau^\dagger$. □

Now consider the category whose objects are arenas and where the morphisms from A to B are single-threaded strategies for $A \multimap B$, where we identify A and B with their corresponding full games. Then, if T is any Idealized Algol type, there is some corresponding arena such that the full game on that arena is isomorphic to $!\llbracket T \rrbracket$. We shall write $\llbracket T \rrbracket$ for this arena, relying on context to distinguish the two notions of denotation. This means that our semantics of Idealized Algol lives within the new category, since a single-threaded strategy for $\llbracket S \rrbracket \to \llbracket T \rrbracket$ is the same thing as a strategy for $!\llbracket S \rrbracket \multimap \llbracket T \rrbracket$ in the original semantics.

We end up with a game semantics entirely based around arenas, without having to talk about distinguished sets of legal plays. This semantics is much closer to the game semantics of PCF developed by Hyland and Ong in [HO00]; the difference there is that they use innocent rather than single-threaded strategies.

We might ask why we did not try to work with this arena-only semantics from the start (as has been done in other work – see [Har99] and [Har06], for example). The reason is that this semantics is unable to accommodate the tensor and sequoid operators, which – although not involved in the denotation of any Idealized Algol type – have been essential to our proof of Computational Adequacy.

3.7 Full Abstraction

We now quote a definability result from [HO00].

Theorem 3.7.1 ([HO00, 7.1])**.** *Let T be a PCF type, and let σ be a compact innocent strategy for $\llbracket T \rrbracket$ (within the arena-only semantics). Then there is some closed PCF term $M : T$ such that $\llbracket M \rrbracket = \sigma$.*

Here, PCF is the sub-language of Idealized Algol generated by the types `nat` and `bool`, together with their structural constants. Notably, it does not contain `let` or any of the explicit stateful structure of Idealized Algol.

We will take it as read that this result may be extended to the types `com` and Var, so that we can say that if T is an Idealized Algol type, then any compact innocent strategy for $\llbracket T \rrbracket$ is the denotation of some IA term. Then our factorization result (Proposition 3.5.1) gives us the following definability result.

Proposition 3.7.2. *Let T be an Idealized Algol type, and let $\sigma : \llbracket T \rrbracket$ be a compact strategy. Then there is some closed IA term $M : T$ such that $\llbracket M \rrbracket = \sigma$.*

Proof. We work in the arena-only semantics so that we can apply Theorem 3.7.1. By Proposition 3.5.1, we know that $\sigma = !0; \mathsf{cell}; \hat{\sigma}$ for some innocent strategy $\hat{\sigma} \colon \mathrm{Var} \to [\![T]\!]$. By looking at the proof of Proposition 3.5.1, we can see that $\hat{\sigma}$ is compact if σ is. Then, by Theorem 3.7.1, we know that there is some IA term $\lambda x.M \colon \mathtt{Var} \to T$ such that $[\![\lambda x.M]\!] = \hat{\sigma}$. It follows that σ is the denotation of the term $\mathtt{new}(\lambda x.M) \colon T$. □

Definition 3.7.3 (Full Abstraction result)**.** A *full abstraction result* encompasses both an equational soundness result and its converse, asserting that two programs are observationally equivalent if and only if their denotations are intrinsically equivalent.

Theorem 3.7.4 (Full Abstraction)**.** *Let $M, N \colon T$ be closed terms of Idealized Algol. Then M and N are observationally equivalent if and only if $[\![M]\!] \sim [\![N]\!]$.*

Proof. We have already proved that our semantics is computationally adequate, and therefore that the reverse direction (equational soundness) holds. Now suppose instead that $[\![M]\!] \not\sim [\![N]\!]$. Without loss of generality, suppose there is $\alpha \colon [\![T]\!] \to \mathbb{C}$ such that $[\![M]\!]^\dagger ; \alpha \neq \bot_{\mathbb{C}}$ and $[\![N]\!]^\dagger ; \alpha = \bot_{\mathbb{C}}$.

Since α is the supremum of all its compact sub-strategies, if $[\![M]\!]^\dagger ; \alpha' = \bot_{\mathbb{C}}$ for all compact $\alpha' \subseteq \alpha$, then $[\![M]\!]^\dagger ; \alpha = \bot_{\mathbb{C}}$ by continuity of composition. Therefore, there is some compact $\alpha' \subseteq \alpha$ such that $[\![M]\!]^\dagger ; \alpha \neq \bot_{\mathbb{C}}$; moreover, we have that $[\![N]\!]^\dagger ; \alpha' = \bot_{\mathbb{C}}$ by monotonicity of composition.

Then α' is the denotation of some term $P \colon T \to \mathtt{com}$, and we have $P\,M \Downarrow$ and $P\,N \not\Downarrow$ by Computational Adequacy. Therefore, M and N are not observationally equivalent. □

3.8 The Intrinsic Equivalence Relation

We conclude by giving a more concrete description of the intrinsic equivalence relation on strategies. This will give us a concrete representation of a morphism in the quotiented category, and hence an algorithm for deciding observational equivalence of (finitary) terms of Idealized Algol.

Definition 3.8.1. We say that a justified sequence s is *complete* if every question move q occurring in s has some answer move a justified by q.

Proposition 3.8.2 ([AM96, 25])**.** *Let $\sigma, \tau \colon !A \multimap B$ be strategies, where A, B are well-opened games. Then $\sigma \sim \tau$ if and only if σ and τ contain the same complete plays.*

Proof. Suppose first that σ and τ do not contain the same complete plays. So, without loss of generality, there is some complete $s \in \sigma \setminus \tau$. Then we may define a strategy $\alpha \colon !(!A \multimap B) \multimap \mathbb{C}$ that behaves as follows. After the opening move q in $\mathbb{C}$, player P plays precisely those moves in $!A \multimap B$ that make up the sequence s. If player O deviates from the sequence s at any point, then player P has no response under α; if player P succeeds in building up the entire sequence s, then she finishes by playing a in $\mathbb{C}$. Note that α is not innocent in general. It is clear then that $\sigma^\dagger ; \alpha = \mathsf{skip}$ and $\tau^\dagger ; \alpha = \bot_{\mathbb{C}}$.

Conversely, suppose that $\sigma \not\simeq \tau$. Then, without loss of generality, there is some $\alpha \colon !(!A \multimap B) \multimap \mathbb{C}$ such that $\sigma^\dagger ; \alpha = \mathsf{skip}$ and $\tau^\dagger ; \alpha = \bot_{\mathbb{C}}$. Therefore we must have some $\mathfrak{s} \in \sigma^\dagger \| \alpha$ such that $\mathfrak{s}|_{\mathbb{C}} = qa$, and $\mathfrak{s}|_{!A \multimap B}$ decomposes into 'threads' which are plays of $!A \multimap B$, one for each initial move occurring in that sequence. By the well bracketing condition, every one of these threads is itself a complete play in $!A \multimap B$. Every one of these sequences is contained in σ; if they were all contained in τ, then we would have $\mathfrak{s} \in \tau^\dagger \| \alpha$, which would be a contradiction, so we end up with a complete sequence contained in $\sigma \setminus \tau$. $\square$

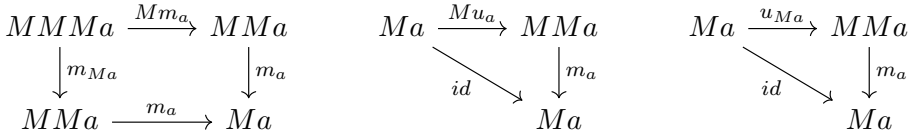
Chapter 4

Monads and Kleisli categories

4.1 Monads

Let $\mathcal{C}$ be a category. Then the category $[\mathcal{C},\mathcal{C}]$ of functors $\mathcal{C} \to \mathcal{C}$ and natural transformations has a (strict) monoidal structure given by composition. A *monad* [Mac71, §VI] in $\mathcal{C}$ is a monoid in $[\mathcal{C},\mathcal{C}]$.

In other words, a monad is a functor $M\colon \mathcal{C} \to \mathcal{C}$ together with natural transformations $m_a\colon MMa \to Ma$ and $u_a\colon a \to Ma$ such that the following diagrams commute for all objects a of $\mathcal{C}$.



Example 4.1.1. In the category of sets, the *nonempty powerset functor* $\mathcal{P}_+$ sends a set A to the set of nonempty subsets of A. This functor has the structure of a monad on **Set**, since we have a natural transformation (union) from $\mathcal{P}_+\mathcal{P}_+A$ to $\mathcal{P}_+A$ and a natural transformation (singleton) from A to$\mathcal{P}_+A$ that obey the diagrams given above.

Example 4.1.2. Let $\mathcal{M}$ be a monoidal category and let x be a monoid in $\mathcal{M}$. The *writer monad* W_x on $\mathcal{M}$ is defined by $W_x y = y \otimes x$, with natural transformations

$$m_y\colon y \otimes x \otimes x \to y \otimes x \qquad\qquad u_y\colon y \to y \otimes x$$

given by the monoid structure on x.

Going the other way, if $\mathcal{M}$ is monoidal closed with inner hom $\multimap$, and if z is a comonoid in $\mathcal{M}$, then the *reader monad* R_z is given by $R_z y = z \multimap y$. Then the monadic coherences

$$m_y \colon z \multimap z \multimap y \to z \multimap y \qquad\qquad u_y \colon y \to z \multimap y$$

are induced from the comonoid structure on z.

If the monoidal structure on $\mathcal{M}$ is Cartesian, then a comonoid in $\mathcal{M}$ is the same thing as an object a of $\mathcal{M}$ (with the diagonal $a \to a \times a$ and terminal $a \to 1$ morphisms). In such a case, every object a of $\mathcal{M}$ gives rise to a reader monad on $\mathcal{M}$.

Example 4.1.3. If $\mathcal{C} \underset{R}{\overset{L}{\underset{\longleftarrow}{\overset{\longrightarrow}{\perp}}}} \mathcal{D}$ is an adjunction with counit $\epsilon \colon LR \to 1$ and unit $\eta \colon 1 \to RL$, then the composite $RL \colon \mathcal{C} \to \mathcal{C}$ has the structure of a monoid on $\mathcal{C}$, where the multiplication and unit are given by

$$R\epsilon L \colon RLRL \to RL \qquad\qquad \eta \colon 1 \to RL\,.$$

We will see in the next chapter that every monad is induced by an adjunction in this way.

Let us look at an example of a monad that arises from an adjunction in this way. Recall that the definition of a monoidal closed category $\mathcal{M}$ is that it admits an adjunction between functors $\_ \otimes w$ and $w \multimap \_$ for any object w of M. If $\mathcal{M}$ is a monoidal closed category and w is an object of $\mathcal{M}$, then the functor S_w on $\mathcal{M}$ defined by

$$S_w x = w \multimap (x \otimes w)$$

is the composition of these adjoint functors and therefore inherits the structure of a monad, known as the *state monad*.

Example 4.1.4. Another example that arises from an adjunction is the *list monad* on **Set** that arises from the adjunction between the category **Set** of sets and the category of monoids. The underlying set of the free monoid on a set A is the set A^* of finite lists of elements of A, and the functor $A \mapsto A^*$ inherits a monoid structure where the multiplication $m_A \colon (A^*)^* \to A^*$ concatenates a list of lists into a single list and the unit $u_a \colon A \to A^*$ forms a list with a single element.

Example 4.1.5. A monad on $\mathcal{C}^{\mathrm{op}}$ is called a *comonad* on $\mathcal{C}$. The carrier of a comonad is still a functor $M \colon \mathcal{C} \to \mathcal{C}$, but now the multiplication and unit are natural transformations $M \Rightarrow MM$ and $M \Rightarrow 1$, rather than the other way round.

An adjunction $\mathcal{C} \underset{R}{\overset{L}{\rightleftarrows}} \mathcal{D}$ gives rise to a comonad structure on LR in much the same way as it gives rise to a monad structure on RL. So, for example, we have the *store comonad* S'_r for any object r of a monoidal closed category $\mathcal{M}$, given by

$$S'_r x = (r \multimap x) \otimes r\,.$$

4.2 Kleisli Categories

Let $\mathcal{C}$ be a category and let M be a monad on $\mathcal{C}$. Then [Kle65] there is a category Kl_M, called the *Kleisli category* of M, whose objects are the objects of $\mathcal{C}$ and where a morphism from an object a to an object b is a morphism $a \to Mb$ in $\mathcal{C}$.

Identity arrows are given by the morphisms $u_c \colon c \to Mc$ (considered as morphisms $c \to c$ in Kl_M) and the composition of arrows $f \colon a \to Mb$ and $g \colon b \to Mc$ is given by the following composite in $\mathcal{C}$.

$$a \xrightarrow{f} Mb \xrightarrow{Mg} MMc \xrightarrow{m_c} Mc$$

There is a natural identity-on-objects functor $J \colon \mathcal{C} \to \mathrm{Kl}_M$ that sends a morphism $f \colon a \to b$ in $\mathcal{C}$ to the composite

$$a \xrightarrow{f} b \xrightarrow{u_b} Mb\,,$$

considered as a morphism $a \to b$ in Kl_M.

In the other direction, we have a functor $S \colon \mathrm{Kl}_M \to \mathcal{C}$ that sends an object a of Kl_M to the object Ma of $\mathcal{C}$ and sends a morphism $f \colon a \to Mb$ from a to b in Kl_M to the composite

$$Ma \xrightarrow{Mf} MMb \xrightarrow{m_b} Mb$$

in $\mathcal{C}$. Note that $SJ = M$, by one of our coherence conditions on m and u. Meanwhile, JS is the functor $\mathrm{Kl}_M \to \mathrm{Kl}_M$ that sends an object a to Ma and sends a morphism $a \to b$ given by a morphism $f \colon a \to Mb$ in $\mathcal{C}$ to the morphism $Mf \colon Ma \to MMb$, considered as a morphism $Ma \to Mb$ in Kl_M.

Proposition 4.2.1 ([Kle65])**.** *S is a right adjoint to J. The unit of the adjunction is $u \colon \mathrm{id} \Rightarrow M$. The counit $e_a \colon J(Sa) \to a$ is given by the identity morphism $Ma \to Ma$ in $\mathcal{C}$, considered as a morphism $Ma \to a$ in Kl_M.*

4.3 Multiplicative natural transformations

Given a monad M on a category $\mathcal{C}$ and a functor $F\colon \mathcal{C} \to \mathcal{D}$, where $\mathcal{D}$ is another category, we say that a natural transformation $\psi_a\colon FMa \to Fa$ is *M-multiplicative* if it makes the following diagrams commute.

$$\begin{array}{ccc} FMMa & \xrightarrow{\psi_{Ma}} & FMa \\ {\scriptstyle Fm_a}\downarrow & & \downarrow{\scriptstyle \psi_a} \\ FMa & \xrightarrow{\psi_a} & Fa \end{array} \qquad\qquad \begin{array}{ccc} Fa & \xrightarrow{Fu_a} & FMa \\ & {\scriptstyle \mathrm{id}}\searrow & \downarrow{\scriptstyle \psi_a} \\ & & Fa \end{array}$$

Given two triples $(\mathcal{D}, F, \psi), (\mathcal{D}', F', \psi')$, where $F\colon \mathcal{C} \to \mathcal{D}, F'\colon \mathcal{C}' \to \mathcal{D}'$ are functors and $\psi\colon FM \Rightarrow F, \psi'\colon F'M \Rightarrow F'$ are functors, we define a *morphism* from $(\mathcal{D}', F', \psi')$ to $(\mathcal{D}, F, \psi)$ to be a functor $H\colon \mathcal{D}' \to \mathcal{D}$ such that $F = HF'$ and $\psi = H\psi'$. This gives us a category.

A defining property of the Kleisli category is that it is an initial object in the category of such triples $(\mathcal{D}, F, \psi)$:

Proposition 4.3.1 ([Str72a])**.** *i) Given an object a of $\mathcal{C}$, consider the identity morphism $Ma \to Ma$ as a morphism $\phi_a\colon JMa \to Ja$ in Kl_M. Then ϕ_a is an M-multiplicative natural transformation.*

ii) Let $\mathcal{D}$ be a category, let $F\colon \mathcal{C} \to \mathcal{D}$ be a functor and suppose that $\psi_a\colon FMa \to Ma$ is an M-multiplicative natural transformation. Then there is a unique functor $\hat{F}\colon \mathrm{Kl}_M \to \mathcal{D}$ such that $F = \hat{F}J$ and $\psi = \hat{F}\phi$.

Another way to characterize the Kleisli category Kl_M is to say that the adjunction we described above is initial among all adjunctions giving rise to the monad M. This can be deduced from Proposition 4.3.1 using the following result.

Lemma 4.3.2 ([Str72a])**.** *Let $\mathcal{C}$ be a category and let M be a monad on $\mathcal{C}$. If $\mathcal{C} \underset{R}{\overset{L}{\rightleftarrows}} \mathcal{D}$ is an adjunction (with counit ϵ and unit η), we say it* gives rise to *M if $M = RL$, $m = R\epsilon L$ and $u = \eta$.*

Any such adjunction gives rise to an M-multiplicative natural transformation $\psi\colon LM \Rightarrow L$. This gives us a fully faithful functor from the category of adjunctions giving rise to M to the category of triples $(\mathcal{D}, F, \psi)$ where ψ is M-multiplicative.

The proof of Proposition 4.3.1 essentially comes down to the following factorization result. If $f\colon a \to b$ is a morphism in Kl_M, then f may be factorized as

$$f = a \xrightarrow{Jf} Mb \xrightarrow{\phi_b} b\,,$$

where we use 'f' to refer both to the morphism $a \to b$ in Kl_M and to the underlying morphism $a \to Mb$ in $\mathcal{C}$. Indeed, if we compute this composite inside $\mathcal{C}$, we get

$$a \xrightarrow{f} Mb \xrightarrow{u_{Mb}} MMb \xrightarrow{M\,\mathrm{id}} MMb \xrightarrow{m_b} Mb\,,$$

which is equal to f by the coherence conditions on m and u. This means that the Kleisli category may be thought of as being freely generated from the original category $\mathcal{C}$ and an M-multiplicative natural transformation ϕ.

Example 4.3.3. The morphisms in the Kleisli category for the nonempty powerset monad $\mathcal{P}_+$ on **Set** are functions $A \to \mathcal{P}_+B$, which can be thought of as nondeterministic functional programs. Given a set A, the morphism $\phi_A \colon \mathcal{P}_+A \to A$ in $\mathrm{Kl}_{\mathcal{P}_+}$ can be interpreted as a 'nondeterministic choice' function that accepts a nonempty set of elements of A and nondeterministically chooses one of them. The factorization then means that the category is freely generated over $\mathcal{C}$ by these nondeterministic choice morphisms.

Example 4.3.4. Let $\mathcal{C}$ be a Cartesian closed category and let z be some fixed object of $\mathcal{C}$. Then the Kleisli category for the reader monad R_z on $\mathcal{C}$ is generated over $\mathcal{C}$ by a natural transformation $\phi_y \colon (z \to y) \to y$. As a Cartesian closed category, $\mathcal{C}$ may be regarded as being enriched over itself. By the enriched Yoneda lemma, then, such a natural transformation is always given by precomposition with some fixed morphism $\mathsf{ask}_z \colon 1 \to z$. This means that Kl_{R_z} is suitable for modelling any situation in which we are generally working in $\mathcal{C}$, but need the ability to request a value of type z (for example, a config file, a piece of user input, a random number or something else that isn't being passed into the function in question).

A particularly important fact about the reader monad in Cartesian closed categories is the following.

Theorem 4.3.5 ([Lam74])**.** *Let $\mathcal{C}$ be a Cartesian closed category and let z be an object of $\mathcal{C}$. Then the Kleisli category Kl_{R_z} for the reader monad over z on $\mathcal{C}$ is Cartesian closed.*

The *functional completeness* theorem [Lam74] can be thought of as a special case of our remarks above.

4.4 Denotational Semantics

Having given an overview of the general theory of monads and Kleisli categories, we now examine the relationship between Kleisli categories and Full Abstraction. From now till the end of the chapter, we fix an base Cartesian closed category $\mathcal{G}$ that admits a denotational semantics of Idealized Algol satisfying Computational Adequacy. In addition, we require that $\mathcal{G}$

can be interpreted as being enriched in algebraic partial orders, in such a way that any compact morphism between the denotation of IA types is the denotation of some IA term. The prototypical example, of course, will be the category of games and visible strategies that we met in Chapter 2, but we will not exploit any properties of this model beyond the ones we have already mentioned, mentioning it only in examples where appropriate.

In addition, we will specialize to reader monads. The rationale behind this is twofold: firstly, we do not wish to assume too much about the underlying category $\mathcal{G}$ beyond the fact that it gives us a good model of Idealized Algol. Reader monads can be constructed for any object inside any Cartesian closed category. Secondly, in order to get a compositional semantics for the λ-calculus, it is important that the Kleisli categories we consider are themselves Cartesian closed. Theorem 4.3.5 tells us that this is the case for the Kleisli categories of reader monads on categories that are themselves Cartesian closed, but this may not be the case for other monads we can build from the Cartesian closed structure on our base category, such as state monads.

Let $X \in \{\mathbb{B}, \mathbb{N}, \mathbb{C}\}$ be a set that has an interpretation as an Idealized Algol type X, and write X also for the corresponding object of $\mathcal{G}$. We shall write $\mathcal{G}_X$ as a shorthand for $\mathrm{Kl}_{R_X}\,\mathcal{G}$, the Kleisli category for the reader monad on $\mathcal{G}$ that corresponds to the object X. The purpose of the rest of this chapter will be to define a new language, give it a denotational semantics in $\mathcal{G}_X$, and prove a full abstraction result for this denotational semantics.

Definition 4.4.1 (The language IA_X)**.** The language IA_X is formed by taking Idealized Algol, and adding to it a new constant

$$\mathsf{ask}_X$$

with typing rule

$$\frac{}{\Gamma \vdash \mathsf{ask}_X : X}\,.$$

From Proposition 4.3.1, we know that there is a distinguished natural transformation $\phi_A \colon (X \to A) \to A$ in $\mathcal{G}_X$; in particular, we have a morphism

$$\phi = \Lambda(\mathrm{lunit}_X ; \mathrm{id}_X) ; \phi_X$$

(where $\Lambda(\mathrm{lunit}_X ; \mathrm{id}_X) \colon 1 \to (X \to X)$ is the interpretation of the λ-term $\lambda x.x$), which will be the denotation of the term ask_X. Together with the existing denotational semantics of Idealized Algol within $\mathcal{G}$, this gives us an inductively defined denotational semantics of IA_X within $\mathcal{G}_X$.

Clearly any term-in-context of IA_X is of the form

$$\Gamma \vdash M[\mathsf{ask}_X / x] \colon T\,,$$

where

$$\Gamma, x\colon X \vdash M\colon T$$

is a judgement of Idealized Algol. Given such a term-in-context, we know that the denotation of

$$\Gamma \vdash (\lambda x.M)\,\mathsf{ask}_X : T$$

is given by the composite

$$1 \xrightarrow{\phi} X \xrightarrow{[\![\Gamma, x \vdash M]\!]} [\![T]\!] \ .$$

Now this last term is β-equivalent to our original term-in-context $\Gamma \vdash M$. Since $\mathcal{G}_X$ is Cartesian closed (by Theorem 4.3.5), the β rule is valid in $\mathcal{G}_X$, and this means that the composite above is an alternative definition of the denotation of $\Gamma \vdash M$.

4.5 Operational Semantics

We now define the operational semantics of IA_X and prove a computational adequacy result for our denotational semantics.

Definition 4.5.1 (Operational semantics of IA_X)**.** Let X^* be the free monoid on the set X; i.e., the set of all finite strings of elements of X. Given $u, v \in X^*$ we shall write $u +\!\!+ v$ for their product in X^*; i.e., the concatenation of the two strings. We shall write ϵ for the unit in X^*; i.e., the empty string.

If $u \in X^*$,we write $|u|$ for the length of u. If $0 \leq n < |u|$, then we write $u^{(n)}$ for the corresponding element of u, numbering from 0.

We inductively define a relation $\Gamma, s \vdash M \Downarrow_u c, s'$, where Γ is a `Var`-context, M, c are terms of IA_X with all free variables in Γ, where c is an IA canonical form, s, s' are Γ-stores and $u \in X^*$. The definition of this relation is shown in Figure 4.1.

Closer examination of the rules in Figure 4.1 reveals an alternative, indirect definition of the operational semantics of IA_X. Note that each rule from ordinary Idealized Algol takes the form

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow c, s^{(n)}} \ ,$$

Here, we have interpreted each IA rule as an infinite scheme of rules ranging over the different terms M_i, M that the rule can apply to. We first extend

$$\frac{}{\Gamma, s \vdash c \Downarrow_\epsilon c, s} \qquad \frac{\Gamma, s \vdash M \Downarrow_u \lambda x.M', s' \qquad \Gamma, s' \vdash M'[N/x] \Downarrow_v c, s''}{\Gamma, s \vdash MN \Downarrow_{u\#v} c, s''}$$

$$\frac{\Gamma, s \vdash M(\mathbf{Y}M) \Downarrow_u c, s'}{\Gamma, s \vdash \mathbf{Y}M \Downarrow_u c, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow_u n, s'}{\Gamma, s \vdash \mathsf{succ}\, M \Downarrow_u n+1, s'}$$

$$\frac{\Gamma, s \vdash M \Downarrow_u n+1, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow_u n, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow_u 0, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow_u 0, s'}$$

$$\frac{\Gamma, s \vdash M \Downarrow_u \mathsf{skip}, s' \qquad \Gamma, s' \vdash N \Downarrow_v c, s''}{\Gamma, s \vdash M; N \Downarrow_{u\#v} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_u \mathbb{t}, s' \qquad \Gamma, s' \vdash N \Downarrow_v c, s''}{\Gamma, s \vdash \mathsf{If}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{u\#v} c, s''} \qquad \frac{\Gamma, s \vdash M \Downarrow_u \mathbb{f}, s' \qquad \Gamma, s' \vdash P \Downarrow_v c, s''}{\Gamma, s \vdash \mathsf{If}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{u\#v} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_u 0, s' \qquad \Gamma, s' \vdash N \Downarrow_v c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{u\#v} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_u n+1, s' \qquad \Gamma, s' \vdash P \Downarrow_v c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{u\#v} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow c', s' \qquad \Gamma, s' \vdash N[c'/x] \Downarrow c, s''}{\Gamma, s \vdash \mathsf{let}\, x = M\, \mathsf{in}\, N \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash E \Downarrow_u n, s' \qquad \Gamma, s' \vdash V \Downarrow_v x, s''}{\Gamma, s \vdash V \leftarrow E \Downarrow_{u\#v} \mathsf{skip}, (s''|x \mapsto n)}\ x \in \Gamma \qquad \frac{\Gamma, s \vdash V \Downarrow_u x, s'}{\Gamma, s \vdash !V \Downarrow_u n, s'}\ s'(x) = n$$

$$\frac{\Gamma, x\colon \mathtt{Var}, (s|x \mapsto 0) \vdash M \Downarrow_u c, (s'|x \mapsto n)}{\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow_u c, s'}$$

$$\frac{\Gamma, s \vdash E \Downarrow_u n, s' \qquad \Gamma, s' \vdash V \Downarrow_v \mathsf{mkvar}\, WR, s'' \qquad \Gamma, s'' \vdash Wn \Downarrow_w \mathsf{skip}, s'''}{\Gamma, s \vdash V \leftarrow E \Downarrow_{u\#v\#w} \mathsf{skip}, s'''}$$

$$\frac{\Gamma, s \vdash V \Downarrow_u \mathsf{mkvar}\, WR, s' \qquad \Gamma, s' \vdash R \Downarrow_v n, s''}{\Gamma, s \vdash !V \Downarrow_{u\#v} n, s''}$$

$$\frac{}{\Gamma, s \vdash \mathsf{ask}_X \Downarrow_x x, s}\ x \in X$$

Figure 4.1: Operational semantics for IA_X. All the rules except the last one are deterministic and may be obtained from the corresponding rules of Idealized Algol by suitably annotating the $\Downarrow$ relation with sequences from X^*.

this scheme of rules to a scheme of rules for IA_X, by allowing the M_i, M to range over terms of IA_X. We then replace the rule with the new rule

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow_{u_n} c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow_{u_1 \# \cdots \# u_n} c, s^{(n)}},$$

to give us an operational rule for IA_X (if $n = 0$, then we treat the empty string ϵ as the empty concatenation). Lastly, we add the rule for the new constant ask_X:

$$\frac{}{\Gamma, s \vdash \mathsf{ask}_X \Downarrow_x x, s}\ x \in X \quad .$$

This rule is the only nondeterministic one in our language, as well as being the only one in which the sequence annotating the $\Downarrow$ symbol at the bottom is not formed from concatenating together the sequences on the top.

Example 4.5.2. If $X = \mathbb{C}$, then, since X has a single element a, a sequence $u = \underbrace{a \cdots a}_{n}$ of elements of X may be identified with its length n. In this case, the language IA_X gives us a way to model time complexity, and the term ask_X may be considered as a subroutine `sleep : com` whose semantics is to wait for some fixed period of time before continuing. In this case,

$$\Gamma, s \vdash M \Downarrow_n c, s'$$

is interpreted to say that 'starting with the store s, M converges to c in time n, leaving store s''.

Example 4.5.3. If $X = \mathbb{N}$, we can interpret the language IA_X as giving us a way to model user input. When we call $\mathsf{ask}_{\mathbb{N}}$, we are asking the user to provide us as a string of text (which we interpret as a binary string and hence as a natural number). Then we interpret the judgement

$$\Gamma, s \vdash M \Downarrow_u c, s'$$

as saying that 'starting with the store s, and the sequence u of pieces of user input, M converges to c, leaving store s''.

Example 4.5.4. If $X \in \{\mathbb{B}, \mathbb{N}\}$, then the language IA_X gives us a way to model nondeterminism, where ask_X behaves as a *nondeterministic oracle*; i.e., a device that nondeterministically returns an element of X.

If $X = \mathbb{B}$ then we have a model of binary (i.e., finite) nondeterminism, whereas if $X = \mathbb{N}$ then we have a model of countable nondeterminism.

In these cases, we interpret the relation

$$\Gamma, s \vdash M \Downarrow_u c, s'$$

as saying that M converges to c in the case that the sequence of values returned by the nondeterministic oracle is given by the sequence u.

4.6 Soundness

To prove our adequacy result for the operational semantics of IA_X, we first give some definitions.

Definition 4.6.1. We inductively define terms in context $\mathrm{tr}_u \colon \mathtt{nat} \to X$ of *ordinary deterministic* Idealized Algol for each $u \in X^*$ as follows.

$$\mathrm{tr}_\epsilon = \lambda n.\Omega \qquad \mathrm{tr}_{xu} = \lambda n.\, \mathsf{let}\, y = n\, \mathsf{in}\, \mathsf{If0}\, y\, \mathsf{then}\, x\, \mathsf{else}\, \mathrm{tr}_u(\mathsf{pred}\, y)$$

In other words, tr_u is the function that will return the i-th term of the sequence u when given i as an input, or will fail to terminate if passed some index $k \geq |u|$.

Proposition 4.6.2. *Let $u \in X^*$ and let $n < |u|$. Then it is possible to deduce that*

$$\frac{\Gamma, s \vdash M \Downarrow n, s'}{\Gamma, s \vdash \mathrm{tr}_u\, M \Downarrow u^{(n)}, s'}$$

in Idealized Algol.

We start our soundness proof with a small lemma to help us deal with substitution.

Lemma 4.6.3. *Let*

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow c, s^{(n)}}$$

be an inference, where the M_i,M are terms of IA_X and the whole inference satisfies one of the patterns of the Idealized Algol rules. Let Q be a fixed term of type X. Then

$$\frac{\begin{array}{c}\Gamma, s^{(0)} \vdash M_1[Q/\,\mathsf{ask}_X] \Downarrow c_1, s^{(1)} \\ \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n[Q/\,\mathsf{ask}_X] \Downarrow c_n, s^{(n)}\end{array}}{\Gamma, s^{(0)} \vdash M[Q/\,\mathsf{ask}_X] \Downarrow c, s^{(n)}}$$

is a valid inference of Idealized Algol.

Proof. Informally, this is true because the term ask_X is not mentioned anywhere in the IA rules, so substitution of the term Q for ask_X could not possibly break the pattern. Formally, we can show this by inspection on each of the different rules. For instance, if the original rule is the one for sequencing:

$$\frac{\Gamma, s \vdash M \Downarrow \mathsf{skip}, s' \qquad \Gamma, s' \vdash N \Downarrow c, s''}{\Gamma, s \vdash M; N \Downarrow c, s''},$$

then we have $(M;N)[Q/\,\mathsf{ask}_X] = M[Q/\,\mathsf{ask}_X];N[Q/\,\mathsf{ask}_X]$ and the inference

$$\frac{\Gamma, s \vdash M[Q/\,\mathsf{ask}_X] \Downarrow \mathsf{skip}, s' \qquad \Gamma, s' \vdash N[Q/\,\mathsf{ask}_X] \Downarrow c, s''}{\Gamma, s \vdash M[Q/\,\mathsf{ask}_X];N[P/\,\mathsf{ask}_X] \Downarrow c, s''}$$

is still a valid instance of the sequencing rule. □

We can now state and prove our soundness lemma. As suggested by the substitution lemmas we have just proved, we will be taking a term M of nondeterministic IA and replacing all instances of ask_X in M with some particular term Q. The Q that we will use is

$$\Gamma, v\colon \mathtt{Var} \vdash v \leftarrow\!\!\succ\, !v; \mathrm{tr}_w\, !v\colon X\,,$$

where w is some sequence of elements of X. The idea is that this term will trace the effect of a particular evaluation path of ask_X, similarly to how we sometimes 'mock' a random number generator when writing unit tests.

Each time we call the term above, we increment the value of v, and successive invocations will end up spelling out a particular substring u of w – depending on the initial value of v and assuming that w is long enough. This will allow us to study one particular evaluation path of the original term M – namely, the one where the successive values returned by ask_X spell out the sequence u.

Lemma 4.6.4. *Suppose that*

$$\Gamma, s \vdash M \Downarrow_u c, s'$$

is derivable in IA_X. Fix $k \in \mathbb{N}$ and let $w \in X^*$ be a sequence such that u is the substring of w starting at position $k+1$ (i.e., $u^{(j)} = w^{(k+j+1)}$ for each $j = 0, \cdots, |u|-1$). Then

$$\Gamma, v\colon \mathtt{Var}, (s|v \mapsto k) \vdash M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v/\,\mathsf{ask}_v] \Downarrow c, (s'|v \mapsto k + |u|)$$

in Idealized Algol.

Proof. Structural induction on the derivation.

Suppose that the last rule we use comes from one of the Idealized Algol rules. That is, there is an inference

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow c, s^{(n)}}\,,$$

derived from one of the Idealized Algol schemas, and we have replaced it with the rule

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow_{u_n} c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow_{u_1 \# \cdots \# u_n} c, s^{(n)}}\,,$$

where each of the relations $\Gamma, s^{(i-1)} \vdash M_i \Downarrow_{u_i} c_i, s^{(i)}$ is derivable in IA_X.

Fix $k \in \mathbb{N}$ and a sequence w such that $u_1 \mathbin{+\!\!+} \cdots \mathbin{+\!\!+} u_n$ is a subsequence of w starting at position $k+1$. In particular, for each $i = 1, \cdots, n$, u_i is a subsequence of w starting at position $k + \sum_{j=1}^{i-1} |u_j| + 1$.

Then by the inductive hypothesis, we know that for each $i = 1, \cdots, n$, the relation

$$\Gamma, v, (s^{(i-1)} | v \mapsto k + \sum_{j=1}^{i-1} |u_j|) \vdash M_i[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v / \mathsf{ask}_v] \Downarrow c, (s^{(i)} | v \mapsto k + \sum_{j=1}^{i} |u_j|)$$

is derivable in Idealized Algol. Then we may apply the Idealized Algol inference and Lemma 4.6.3 to deduce that

$$\Gamma, v, (s^{(0)} | v \mapsto k) \vdash M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v / \mathsf{ask}_v] \Downarrow c, (s^{(n)} | v \mapsto k + \sum_{i=1}^{n} |u_n|)\,,$$

as desired.

Now suppose instead that the last rule was the new one for ask_X; i.e.,

$$\frac{}{\Gamma, s \vdash \mathsf{ask}_X \Downarrow_x x, s}\,,$$

where $x \in X$. Fix some $k \in \mathbb{N}$ and some w such that the single term x is a subsequence of w starting at position $k+1$; i.e., that $x = w^{(k+1)}$. Then we would like to derive that

$$\Gamma, v, (s | v \mapsto k) \vdash v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v \Downarrow x, (s | v \mapsto k+1)\,,$$

which we can do using the derivation in Figure 4.2a, where we have used Proposition 4.6.2 to deal with the tr_w term.

This completes the induction. □

In light of Lemma 4.6.4, we can state our soundness result.

First recall the statement of Computational Adequacy for $\mathcal{G}$ (Theorem 3.4.9 if $\mathcal{G}$ is the category of games):

Theorem 4.6.5. *Let M : $\mathtt{com}$ be a closed term of Idealized Algol. Then*

$$\_, () \vdash M \Downarrow \mathsf{skip}, ()\,.$$

if and only if $[\![M]\!] \neq \bot$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, v, (s|v \mapsto k) \vdash v \Downarrow v, (s|v \mapsto k)}}{\Gamma, v, (s|v \mapsto k) \vdash !v \Downarrow k, (s|v \mapsto k)}}{\Gamma, v, (s|v \mapsto k) \vdash \mathsf{succ}!v \Downarrow k+1, (s|v \mapsto k)} \qquad \dfrac{}{\Gamma, v, (s|v \mapsto k) \vdash v \Downarrow v, (s|v \mapsto k)}}{\Gamma, v, (s|v \mapsto k) \vdash v \leftarrow \mathsf{succ}!v \Downarrow \mathsf{skip}, (s|v \mapsto k+1)} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, v, (s|v \mapsto k+1) \vdash v \Downarrow v, (s|v \mapsto k+1)}}{\Gamma, v, (s|v \mapsto k+1) \vdash !v \Downarrow k+1, (s|v \mapsto k+1)}}{\Gamma, v, (s|v \mapsto k+1) \vdash \mathrm{tr}_w!v \Downarrow x, (s|v \mapsto k+1)}}{\Gamma, v, (s|v \mapsto k) \vdash v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v \Downarrow x, (s|v \mapsto k+1)}$$

(a) IA derivation that if $s(v) = k$, then $\Gamma, s \vdash v \leftarrow \mathsf{succ}!v; \mathrm{tr}_w!v$ converges to the $k+1$-th term of w, leaving state $v \mapsto k+1$.

$$\textsc{Lem.}\ 4.6.4\ \dfrac{\dfrac{\dfrac{}{v, (v \mapsto 0) \vdash M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_x] \Downarrow \mathsf{skip}, (v \mapsto |u|)} \qquad \dfrac{\dfrac{}{v, (v \mapsto |u|) \vdash v \Downarrow v, (v \mapsto |u|)}}{v, (v \mapsto |u|) \vdash !v \Downarrow |u|, (v \mapsto |u|)}}{v, (v \mapsto 0) \vdash M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_X]; v \Downarrow |u|, (v \mapsto |u|)}}{\_, () \vdash \mathtt{new}(\lambda v.M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_X]; !v) \Downarrow |u|, ()}$$

(b) IA derivation used in the proof of Proposition 4.6.9.

Figure 4.2: Some useful IA derivations

Definition 4.6.6. Let $u \in X^*$. Let $u^\top$ be the sequence formed by appending some fixed value $\top \in X$ to the start of u, so that u is the subsequence of $u^\top$ running from position 1 up to position $|u|$. Define a morphism

$$\eta_u = [\![f \colon X \to \mathtt{com} \vdash \lambda v. f(v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v); !v]\!] : (X \to \mathbb{C}) \to (\mathtt{Var} \to \mathbb{N})$$

in $\mathcal{G}$.

Definition 4.6.7. Let n be a natural number. We inductively define terms $\mathrm{test}_n \colon \mathtt{nat} \to \mathtt{com}$ by

$$\mathrm{test}_0 = \lambda m. \, \mathsf{If0}\, m \, \mathsf{then}\, \mathsf{skip} \, \mathsf{else}\, \Omega$$

$$\mathrm{test}_{n+1} = \lambda m. \, \mathsf{let}\, x = m \, \mathsf{in}\, \mathsf{If0}\, x \, \mathsf{then}\, \Omega \, \mathsf{else}\, \mathrm{test}_n(\mathsf{pred}\, x)$$

So test_n converges if its input evaluates to n and diverges otherwise.

We then define $t_n \colon \mathbb{N} \to \mathbb{C}$ to be the denotation of test_n in $\mathcal{G}$.

Definition 4.6.8. Let $\sigma \colon 1 \to \mathbb{C}$ be a morphism in $\mathcal{G}_X$, considered as a morphism $(X \to \mathbb{C})$ in $\mathcal{G}$. We say that $\sigma \downarrow_u$ if the composite

$$1 \xrightarrow{\sigma} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$.

Proposition 4.6.9. *Let $M \colon \mathtt{com}$ be a closed term of IA_X, let $u \in X^*$ be a sequence and suppose that*

$$\_, () \vdash M \Downarrow_u \mathsf{skip}, () \, .$$

Let the denotation $[\![M]\!] : 1 \to \mathtt{com}$ in $\mathcal{G}_X$ be considered as a morphism $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$. Then $[\![M]\!] \downarrow_u$; i.e., the composite

$$1 \xrightarrow{[\![M]\!]} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$.

Proof. Since the β rule is valid in $\mathcal{G}_X$, this composite is equal to the denotation of the term

$$\mathrm{test}_{|u|}(\mathtt{new}(\lambda v. M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v / \, \mathsf{ask}_X]; !v))$$

in IA. By the adequacy result for Idealized Algol, it suffices to show that this term converges to skip; i.e., that the term

$$\mathtt{new}(\lambda v. M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v / \, \mathsf{ask}_X]; !v)$$

converges to $|u|$ in IA.

We can prove this using the derivation tree in Figure 4.2b. □

Remark 4.6.10. Definition 4.6.8 looks a bit odd. This is a result of working at such a high level of generality: since we have not assumed much about $\mathcal{G}$ beyond the fact that it is a suitable model of Idealized Algol, then we have to define everything in terms of Idealized Algol denotations.

If $\mathcal{G}$ is the category of games and visible strategies, then the statements of Proposition 4.6.9 (and our later Adequacy and Full Abstraction results) become clearer. Observe that if $\sigma\colon 1 \to (X \to \mathbb{C})$ is a strategy in $\mathcal{G}$ (considered as a strategy for $!X \multimap \mathbb{C}$, then the maximal plays in the interaction

$$\sigma||(\eta_u; [\![\mathsf{new}]\!])$$

take the form

$$\begin{array}{ccc} X & \mathbb{C} & \mathbb{N} \\ & & q \\ & q & \\ q & & \\ u^{(0)} & & \\ \vdots & & \\ q & & \\ u^{(k-1)} & & \\ & a & \\ & & k\,, \end{array}$$

for $k \leq |u|$, where the component in $X, \mathbb{C}$ is a valid play of σ. Moreover, the strategy t_n is the one with maximal plays of the form

$$\begin{array}{cc} \mathbb{N} & \mathbb{C} \\ & q \\ q & \\ n & \\ & a \end{array}$$

or

$$\begin{array}{cc} \mathbb{N} & \mathbb{C} \\ & q \\ q & \\ m & \end{array}$$

for $m \neq n$.

This means that the composite

$$1 \xrightarrow{\sigma} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$ if and only if σ contains the sequence

$$\begin{array}{cc} X & \mathbb{C} \\ & \\ & q \\ q & \\ u^{(0)} & \\ \vdots & \cdot \\ q & \\ u^{(|u|-1)} & \\ & a \end{array}$$

Since complete plays in the game $!X \multimap \mathbb{C}$ are always of the form q, followed by some sequence of pairs of the form qx_i for $x_i \in X$, followed by a, it is very natural to consider conditions on those x_i when dealing with a strategy $\sigma : !X \multimap \mathbb{C}$.

4.7 Computational Adequacy

Now we want to prove Computational Adequacy; i.e., the converse to Proposition 4.6.9. To do this, we need to prove a converse to Lemma 4.6.4.

First of all, we need to prove a reverse result to Lemma 4.6.3 that deals with substitution in the opposite direction. That is, instead of telling us what happens when we substitute a term for ask_X, we will look at what happens when we substitute a term for $v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v$.

In most cases, this will not disrupt the structure of the IA rule. For instance, we always have

$$(!V)[Q/v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v] = !(V[Q/v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v]),$$

and so the derivation

$$\frac{\Gamma, s \vdash V[Q/v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v] \Downarrow v, s'}{\Gamma, s \vdash !V[Q/v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v] \Downarrow n, s'}\ s'(v) = n$$

still follows the pattern of the Idealized Algol rule for variable dereference.

There is only one case where this breaks down. Consider the following instance of the sequencing rule.

$$\frac{\Gamma, v, s \vdash v \leftarrow \mathsf{succ}!v \Downarrow \mathsf{skip}, s' \qquad \Gamma, v, s' \vdash \mathrm{tr}_u!v \Downarrow x, s''}{\Gamma, v, s \vdash v \leftarrow \mathsf{succ}!v; \mathrm{tr}_u!v \Downarrow x, s''}$$

In this case, substituting some term Q for $v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v$ in the top two terms will have no effect, whereas it will replace the whole of the bottom with Q, invalidating the whole inference.

We have proved the following.

Lemma 4.7.1. *Let*

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow c, s^{(n)}}$$

be an inference of Idealized Algol. Let $u \in X^*$ and let $Q\colon X$ be a term of IA_X. Fix an unused variable name v and suppose that $M \neq v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v$. Then

$$\frac{\begin{array}{c}\Gamma, s^{(0)} \vdash M_1[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow c_1, s^{(1)} \\ \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow c_n, s^{(n)}\end{array}}{\Gamma, s^{(0)} \vdash M[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow c, s^{(n)}}$$

conforms to the same Idealized Algol pattern. In particular, if $w_1, \cdots, w_n \in X^*$, then

$$\frac{\begin{array}{c}\Gamma, s^{(0)} \vdash M_1[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow_{w_1} c_1, s^{(1)} \\ \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow_{w_n} c_n, s^{(n)}\end{array}}{\Gamma, s^{(0)} \vdash M[Q/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v] \Downarrow_{w_1 \# \cdots \# w_n} c, s^{(n)}}$$

is a valid inference of IA_X.

We need one more lemma to help us deal with substitution.

Lemma 4.7.2. *Suppose that $\Gamma, y \vdash M : T$ is a typing judgement of Idealized Algol, where Γ is a* `Var`*-context and y is a free variable of type X. Fix $u \in X^*$. Suppose that $M \neq y$ and that we have some inference*

$$\frac{\Gamma, s^{(0)} \vdash N_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash N_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y] \Downarrow c, s^{(n)}}\,.$$

of Idealized Algol. Then each N_i may be written as $M_i[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y]$ for some $\Gamma, y \vdash M_i$.

Proof. This can be checked case-by-case. The most interesting is the case for sequencing: if $M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y] \neq v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v$, then we must have

$$M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y] = N[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y]; P[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y]\,,$$

which is deduced from $N[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y]$ and $P[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y]$. □

Now we can state and prove our adequacy lemma.

Lemma 4.7.3. *Suppose that $w \in X^*$ is a sequence of length greater than or equal to k, l and that*

$$\Gamma, v, (s|v \mapsto k) \vdash M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y] \Downarrow c, (s'|v \mapsto l)$$

is derivable in Idealized Algol, where v is not free in M and y is a variable name of type X. Then $l \geq k$ and

$$\Gamma, s \vdash M[\mathsf{ask}_X\,/y] \Downarrow_u c, s'$$

in IA_X, where u is the subsequence of w consisting of all terms from $k+1$ up to l.

Proof. Induction on the derivation.

Suppose that $M \neq y$. Then, by Lemma 4.7.2, the last step in the derivation of $M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y]$ must be of the form

$$\frac{\begin{array}{cc} & \Gamma, s^{(0)} \vdash M_1[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y] \Downarrow c_1, s^{(1)} \\ \cdots & \quad \Gamma, s^{(n-1)} \vdash M_n[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y] \Downarrow c_n, s^{(n)} \end{array}}{\Gamma, s^{(0)} \vdash M[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y] \Downarrow c, s^{(n)}}\,,$$

where each $M_i[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v/y]$ is derivable in Idealized Algol.

By the inductive hypothesis, $s^{(i-1)}(v) \leq s^{(i)}(v)$ for each i and so $s^{(0)}(v) \leq s^{(n)}(v)$, as desired (in the case that there are no premises – i.e., the case of the rule for canonical forms – we have $s^{(0)}(v) = s^{(0)}(v)$). Moreover, by the inductive hypothesis, it is derivable that

$$\Gamma, s^{(i-1)} \vdash M_i[\mathsf{ask}_X\,/y] \Downarrow_{u_i} c_i, s^{(i)}\,,$$

where u_i is the subsequence of w going from term $s^{(i-1)}(v)+1$ up to $s^{(i)}(v)$.

Now for any term $\Gamma, y \vdash P$, we have

$$P[\mathsf{ask}_X\,/y] = P[v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v/y][\mathsf{ask}_X\,/v \leftarrow \mathsf{succ!}v; \mathrm{tr}_u!v]\,,$$

and so by Lemma 4.7.1 we may derive

$$\Gamma, s^{(0)} \vdash M[\mathsf{ask}_X\,/y] \Downarrow_{u_1 + \cdots + u_n} c, s^{(n)}\,.$$

But $u_1 + \cdots + u_n$ is precisely the subsequence of w going from term $s^{(0)}(v)+1$ up to $s^{(n)}(v)$!

This completes the first case. The second case is where $M = y$. Suppose, then, that

$$\Gamma, v, (s|v \mapsto k) \vdash v \leftarrow \mathsf{succ!}v; \mathrm{tr}_w!v \Downarrow x, (s'|v \mapsto l)$$

is derivable in Idealized Algol.

Since IA is a deterministic language (so if $\Gamma, s \vdash M \Downarrow c, s'$ and $\Gamma, s \vdash M \Downarrow c', s''$ then $c = c'$ and $s' = s''$), then the derivation of this term must agree with the valid IA derivation given in Figure 4.2a. It follows that $l = k + 1$ and that x is the $(k + 1)$-th term of w, so the single-term sequence x is the subsequence of w going from $k + 1$ to l.

Then we have the derivation

$$\frac{}{\Gamma, s \vdash \mathsf{ask}_X \Downarrow_x x, s'}$$

in IA_X. This completes the induction. □

We can now prove computational adequacy for our model. We have proved one direction already in Proposition 4.6.9, so it suffices to prove the other direction.

Proposition 4.7.4 (Computational adequacy)**.** *Let $M \colon \mathtt{com}$ be a closed term of IA_X. Consider the denotation $[\![M]\!] : 1 \to \mathbb{C}$ in $\mathcal{G}_X$ as a morphism $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$. Let $u \in X^*$ be a sequence and suppose that $[\![M]\!] \downarrow_u$; i.e., that the composite*

$$1 \xrightarrow{[\![M]\!]} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$. Then

$$\_, () \vdash M \Downarrow_u \mathsf{skip}, () \,.$$

Proof. As before, the composite given in the statement is the denotation of the term

$$\mathrm{test}_{|u|}(\mathtt{new}(\lambda v.M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_X]; !v)) \,.$$

By the adequacy result for Idealized Algol, the fact that this denotation is not equal to $\bot$ means that the term converges to skip, from which we can deduce that

$$\mathtt{new}(\lambda v.M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_X]; !v$$

converges to $|u|$.

It is easy to see that this is equivalent to asking whether we can derive the following relation in Idealized Algol.

$$v, (v \mapsto 0) \vdash M[v \leftarrow \mathsf{succ}!v; \mathrm{tr}_{u^\top}!v/\,\mathsf{ask}_X] \Downarrow \mathsf{skip}, (v \mapsto |u|)$$

Now u is the subsequence of $u^\top$ going from position 1 to position $|u|$. So Lemma 4.7.3 tells us that we must have

$$\_, () \vdash M \Downarrow_u \mathsf{skip}, ()$$

in IA_X. □

Remark 4.7.5. Recall that if $\mathcal{G}$ is the category of games, then we have $\sigma \downarrow_u$ if and only if σ contains the play

$$\begin{array}{cc} X & \mathbb{C} \\ & \\ & q \\ q & \\ u^{(0)} & \\ \vdots & \cdot \\ q & \\ u^{(|u|-1)} & \\ & a \end{array}$$

Therefore, we have now proved that we have $\_, () \vdash M \Downarrow_u \mathsf{skip}, ()$ if and only if $[\![M]\!]$ contains that sequence.

4.8 Full Abstraction

To prove full abstraction of our semantics for IA_X, we introduce the usual intrinsic equivalence on terms.

Definition 4.8.1. Let $\sigma, \tau \colon A \to B$ be morphisms in $\mathcal{G}_X$. By currying, we may consider A and B as morphisms $1 \to (A \to B)$ in $\mathcal{G}_X$. We say that $\sigma \sim \tau$ if for all morphisms $\alpha \colon (A \to B) \to \mathbb{C}$ and for all sequences $u \in X^*$, if we regard $\sigma; \alpha, \tau; \alpha \colon 1 \to \mathbb{C}$ as morphisms $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$, then the composites

$$1 \xrightarrow{\sigma;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

$$1 \xrightarrow{\tau;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

are equal.

Theorem 4.8.2 (Full abstraction)**.** *Let $M, N \colon T$ be closed terms of IA_X. Then M, N are observationally equivalent – i.e., for all contexts $C[-] \colon \mathtt{com}$ of IA_X with a hole of type T and for all sequences $u \in X^*$,*

$$\_, () \vdash C[M] \Downarrow_u \mathsf{skip}, () \iff \_, () \vdash C[N] \Downarrow_u \mathsf{skip}, () \text{ –}$$

if and only if $[\![M]\!] \sim [\![N]\!]$.

Proof. First, suppose that $[\![M]\!] \sim [\![N]\!]$. Let $C[-]\colon \mathtt{com}$ be a context with a hole of type T. Then the denotation of $t \vdash C[t]$ is a morphism $\alpha\colon [\![T]\!] \to \mathbb{C}$. Moreover, the denotation of $C[M]$ is the composite $[\![M]\!]\,;\alpha$ and that of $C[N]$ is the composite $[\![N]\!]\,;\alpha$ since we are working in a Cartesian closed category.

Then the composites

$$1 \xrightarrow{\sigma;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

$$1 \xrightarrow{\tau;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

are equal, so $C[M] \Downarrow_u \mathsf{skip}$ if and only if $C[N] \Downarrow_u \mathsf{skip}$ by Propositions 4.6.9 and 4.7.4.

Conversely, suppose that $M \not\sim N$. So there is some $\alpha\colon [\![T]\!] \to \mathbb{C}$ in $\mathcal{G}_X$ and some sequence u such that (without loss of generality),

$$([\![M]\!]\,;\alpha);\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} \neq \bot \qquad ([\![N]\!]\,;\alpha);\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} = \bot\,.$$

Here, we have enclosed $[\![M]\!]\,;\alpha$ and $[\![N]\!]\,;\alpha$ in brackets to indicate that the composition is taken in the Kleisli category $\mathcal{G}_X$, and then the whole thing is considered as a morphism $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$.

More specifically, these composites are given by the composites

$$1 \xrightarrow{[\![M]\!]} (X \to [\![T]\!]) \xrightarrow{X\to\alpha} (X \to (X \to \mathbb{C})) \xrightarrow{\mu} (X \to \mathbb{C})$$

$$1 \xrightarrow{[\![N]\!]} (X \to [\![T]\!]) \xrightarrow{X\to\alpha} (X \to (X \to \mathbb{C})) \xrightarrow{\mu} (X \to \mathbb{C})$$

in $\mathcal{G}$, where μ indicates precomposition with the diagonal.

Now α is the least upper bound of its compact approximants, so it follows that there is some compact $\alpha' \subseteq \alpha$ such that

$$[\![M]\!]\,;(X \to \alpha');\mu;\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} \neq \bot$$

$$[\![N]\!]\,;(X \to \alpha');\mu;\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} = \bot\,.$$

Then, by compact definability in $\mathcal{G}$, α' is the denotation of some IA term $x\colon T \vdash C[x]\colon X \to \mathtt{com}$, which is therefore the denotation of the term $x\colon T \vdash C[x]\ \mathsf{ask}_X\colon \mathtt{com}$ in $\mathcal{G}_X$. So we get

$$[\![C[M]]\!]\,;\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} \neq \bot \qquad [\![C[N]]\!]\,;\eta_u;[\![\mathsf{new}]\!]\,;t_{|u|} = \bot\,,$$

and so $C[M] \Downarrow_u \mathsf{skip}$ by Proposition 4.7.4, while $C[N] \not\Downarrow_u \mathsf{skip}$ by Proposition 4.6.9. Therefore, M and N are observationally inequivalent in IA_X. □

4.9 Comparison with Ghica's slot games

For this section, let us suppose that $\mathcal{G}$ is the category of games and visible strategies, and that $X = \mathbb{C}$. As we remarked above, this means that IA_X can be interpreted as a language for modelling time complexity.

We compare our approach to a different one, due to Dan Ghica [Ghi05]. Given a game A, Ghica defines a *play with costs* in A to be a justified sequence $s \in (M_A + \{\circledS\})^*$ such that $s|_{M_A} \in P_A$. Here, $\circledS$ is a special symbol called a *slot* or *token-action*, which can be interleaved throughout the play $s|_{M_A}$ from A. We shall also impose the requirement (not found in [Ghi05]) that an occurrence of the special symbol $\circledS$ must take place either after an O-move in A or after another instance of $\circledS$. The token actions do not carry justification pointers.

Following Ghica, we define a *strategy with costs* to be a prefix-closed set σ of plays with costs such that the set $\sigma|_{M_A} = \{s|_{M_A} : s \in \sigma\}$ is a valid visible strategy for A and such that determinism applies to token actions as well as ordinary moves – i.e., if s is an O-play with costs in A such that $s\circledS a \in \sigma$ for some ordinary move a, then $sb \notin \sigma$ for all ordinary moves b.

The identity strategy with costs is the usual identity strategy, without any token actions. We can similarly define an *interaction sequence with costs* to be a sequence $\mathfrak{s} \in (M_A + M_B + M_C + \{\circledS\})^*$ such that $\mathfrak{s}|_{A,B,C}$ is an interaction sequence for A, B, C and such that each occurrence of the token action $*$ occurs after a move that could be considered as an O-move in either $A \multimap B$ or $B \multimap C$.

Given such an interaction sequence $\mathfrak{s} \in (M_A + M_B + M_C + \{\circledS\})^*$ between the games A, B and C, write $\mathfrak{s}|_{A,B}$ for the subsequence consisting of all those moves in A and B, together with all token actions such that the previous move was an O-move in $A \multimap B$. Define $\mathfrak{s}|_{B,C}$ similarly. Then if $\sigma\colon A \multimap B$ and $\tau\colon B \multimap C$ are strategies with costs, we define $\sigma\|\tau$ to be the set of all such sequences $\mathfrak{s}$ such that $\mathfrak{s}|_{A,B} \in \sigma$ and $\mathfrak{s}|_{B,C} \in \tau$. Lastly, we define $\sigma;\tau$ to be the set of all sequences obtained by taking a sequence $\mathfrak{s} \in \sigma\|\tau$ and removing all the moves in B (but retaining all the token actions, including those that arise between moves in B). The usual arguments apply to show that this is indeed a category.

This seems like a purely combinatorial construction, but it can actually be subsumed into our category-theoretic apparatus.

Proposition 4.9.1. *Let A be a game. Then there is a bijection*

$$c\colon \{\textit{normal strategies for } !\mathbb{C} \multimap A\} \leftrightarrow \{\textit{strategies with costs for } A\}$$

Moreover, this bijection respects composition and identity in the Kleisli cat-

egory: let $\sigma\colon !\mathbb{C} \multimap (A \to B)$ *and* $\tau\colon !\mathbb{C} \multimap (B \to C)$ *be strategies. Write* $\sigma;\tau$ *for the Kleisli composition of* σ *and* τ *in* $\mathcal{G}_{\mathbb{C}}$*; i.e., the composite*

$$!\mathbb{C} \xrightarrow{\mu} !\mathbb{C} \otimes !\mathbb{C} \xrightarrow{\sigma\otimes\tau} (A \multimap B) \otimes (B \multimap C) \xrightarrow{;} (A \multimap C)\,.$$

Then $c(\sigma;\tau) = c(\sigma);c(\tau)$*. Moreover,* $c(\mathrm{id}_A)$ *is the identity in the category of games and strategies with costs.*

Proof. The map c is the unique functor given by the functional completeness theorem (Proposition 4.3.1) that sends the canonical strategy $\phi\colon 1 \to \mathbb{C}$ to the strategy with costs for $\mathbb{C}$ with maximal play

$$q\textcircled{\$}a\,.$$

More synthetically, we get from a strategy for $\sigma\colon !\mathbb{C} \multimap A$ to a strategy with costs for A by replacing each occurrence of the pair qa occurring in the $\mathbb{C}$ component with the token action Ⓢ in each play of σ.

This functor is the identity on objects, and it is fully faithful, since it has an obvious inverse, given by taking a strategy with costs and replacing each occurrence of the token action with a pair of moves qa in $!\mathbb{C}$, where q justifies a and is itself justified by the most recently occurring O-move in A. Since each token action must always occur after an opponent move or after another token action, and since player O has no reply to the move q other than the move a, this always gives us a legal strategy. □

Therefore, we see that the category of games and strategies with costs is isomorphic to the Kleisli category $\mathcal{G}_{\mathbb{C}}$, which we have already shown to be fully abstract for a language with time complexity.

4.10 Alternative Reduction Rules - May Testing

We remarked above that if $X \in \{\mathbb{B}, \mathbb{N}\}$, then IA_X is a model of nondeterminism, finite in the case of $\mathbb{B}$ and countable in the case of $\mathbb{N}$. However, we have given a rather strange operational semantics for these languages (the relation $\Downarrow_u$) that does not accurately reflect how they are normally used.

For example, the terms

$$\mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ \mathtt{t}\ \mathsf{else}\ \mathtt{f} : \mathtt{bool} \qquad\qquad \mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ \mathtt{f}\ \mathsf{else}\ \mathtt{t} : \mathtt{bool}$$

are not observationally equivalent in our operational semantics; indeed, we have

$$\mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ \mathtt{t}\ \mathsf{else}\ \mathtt{f} \Downarrow_{\mathtt{t}} \mathtt{t} \qquad\qquad \mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ \mathtt{f}\ \mathsf{else}\ \mathtt{t} \Downarrow_{\mathtt{t}} \mathtt{f}\,.$$

However, these terms (which both nondeterministically choose either the true or the false value), *should* be observationally equivalent in any sensible nondeterministic semantics. The issue is the labelling on the reduction relations, which is saving too much information about the reduction of the term. Indeed, this is sort of the point of nondeterminism: we should be able to make nondeterministic choices without recording which value we used for that choice.

Definition 4.10.1 ([HM99])**.** If $X \in \{\mathbb{B}, \mathbb{N}\}$, we define an operational relation $\Downarrow$ ('may converge') on the language IA_X as follows. The rules for $\Downarrow$ are identical to the operational rules for Idealized Algol, with the addition of the following rule for the primitive ask_X.

$$\frac{}{\Gamma, s \vdash \mathsf{ask}_X \Downarrow x, s}\ x \in X$$

These rules are exactly the same as our original operational semantics for IA_X, but with the sequences u removed. Moreover, if we have a valid derivation of $\Gamma, s \vdash M \Downarrow u, s'$, then it is clear (by induction) that we may annotate all the occurrences of $\Downarrow$ with suitable sequences in order to obtain a derivation of $\Gamma, s \vdash M \Downarrow_u c, s'$ for some $u \in X^*$. So we get the following alternative definition of may convergence.

Definition 4.10.2. We say that $\Gamma, s \vdash M \Downarrow c, s'$ if there is some $u \in X^*$ such that $\Gamma, s \vdash M \Downarrow_u c, s'$.

We can reflect this operational relation in the semantics by modifying the definition of intrinsic equivalence.

Firstly, an obvious consequence of Propositions 4.6.9 and 4.7.4 is that

Corollary 4.10.3 (Computational Adequacy for May Testing)**.** *Let $M : \mathtt{com}$ be a closed term of IA_X. Consider the denotation $[\![M]\!] : 1 \to \mathbb{C}$ in $\mathcal{G}_X$ as a morphism $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$. Then there exists some sequence $u \in X^*$ such that the composite*

$$1 \xrightarrow{[\![M]\!]} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$, if and only if

$$\_, () \vdash M \Downarrow \mathsf{skip}, ()\,.$$

In light of this result, we can define a new intrinsic equivalence on morphisms in $\mathcal{G}_X$:

Definition 4.10.4. Let $\sigma, \tau \colon A \to B$ be morphisms in $\mathcal{G}_X$, considered as morphisms $1 \to (A \to B)$ in $\mathcal{G}_X$. We say that $\sigma \sim_{\text{may}} \tau$ if for all $u \in X^*$ there exists $v \in X^*$ such that the composites

$$1 \xrightarrow{\sigma;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

$$1 \xrightarrow{\tau;\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_v} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|v|}} \mathbb{C}$$

are equal, and vice versa.

Then exactly the same argument as before gives us a full abstraction result for may-equivalence.

Theorem 4.10.5. *Let $M, N \colon T$ be closed terms of IA_X. Then M, N are may-observationally equivalent – i.e., for all contexts $C[-] \colon \mathtt{com}$ of IA_X with a hole of type T,*

$$\_, () \vdash C[M] \Downarrow \mathsf{skip}, () \Leftrightarrow \_, () \vdash C[N] \Downarrow \mathsf{skip}, () \text{ –}$$

if and only if $[\![M]\!] \sim_{\text{may}} [\![N]\!]$.

Proof. Suppose that $[\![M]\!] \sim_{\text{may}} [\![N]\!]$. Let $C[-]$ be a context of IA_X, interpreted as a morphism $\alpha \colon [\![T]\!] \to \mathbb{C}$.

Suppose that $\_, () \vdash C[M] \Downarrow \mathsf{skip}, ()$. So there is some sequence $u \in X^*$ such that $\_, () \vdash C[M] \Downarrow_u \mathsf{skip}, ()$ and therefore the composite

$$1 \xrightarrow{[\![M]\!];\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$. This means that there exists some $v \in X^*$ such that the composite

$$1 \xrightarrow{[\![N]\!];\alpha} (X \to \mathbb{C}) \xrightarrow{\eta_v} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|v|}} \mathbb{C}$$

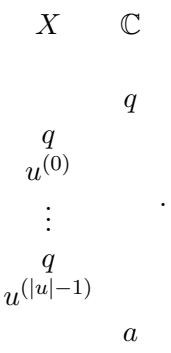
Therefore, $\_, () \vdash C[N] \Downarrow_v \mathsf{skip}, ()$, and so $\_, () \vdash C[N] \Downarrow \mathsf{skip}, ()$. The reverse direction is identical.

Conversely, suppose that M, N are may-observationally equivalent. Then, as before, we can take α to be compact, whence definable, in Definition 4.10.4, and the proof continues as in the first part, but in reverse. □

Let us examine what this means in the category of games. If $\sigma \colon !X \multimap \mathbb{C}$ is a strategy, then, by our discussion at the end of §4.6, we know that, for any sequence u, the composite

$$1 \xrightarrow{\sigma} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$ if and only if σ contains the play



Moreover, since any complete play in $!X \multimap \mathbb{C}$ must take this form, we can see there exists such a u making the composite above not equal to $\bot$ if and only if

$$qa \in \{s|_{\mathbb{C}} : s \in \sigma \text{ is complete}\}.$$

This suggests a general equivalence relation on Kleisli morphisms in $\mathcal{G}_X$: given a strategy $\sigma : !X \multimap A$, we write $\sigma|_A$ for the set

$$\{s|_A : s \in \sigma \text{ is complete}\}.$$

We say that two strategies $\sigma, \tau : !X \multimap A$ are may-equivalent, and write $\sigma \approx_{\text{may}} \tau$, if

$$\sigma|_A = \tau|_A .$$

In this case, Corollary 4.10.3 tells us that $M \Downarrow \mathsf{skip}$ if and only if $\sigma \approx_{\text{may}} \tau$.

We need to show that this respects composition, so that we get a category if we take the quotient by this equivalence relation.

Proposition 4.10.6. *Let $\sigma, \sigma' : A \to B$, $\tau, \tau' : B \to C$ be morphisms in $\mathcal{G}_X$. Suppose that $\sigma \approx_{\text{may}} \sigma'$ and $\tau \approx_{\text{may}} \tau'$. Then $\sigma;\tau \approx_{\text{may}} \sigma';\tau'$.*

Proof. A complete play in $\sigma;\tau$ is given by a sequence $\mathfrak{s}|_{X,A,C}$, where $\mathfrak{s} \in (M_X + M_A + M_B + M_C)^*$ is a legal interaction of a complete play in τ with a collection of complete plays in σ, B-components being identified. Then $\mathfrak{s}|_{A,C}$ can alternatively be characterized as $\mathfrak{t}|_{A,C}$, where $\mathfrak{t} \in (M_A + M_B + M_C)^*$ is a legal interaction of a sequence from $\tau|_{B,C}$ with a collection of sequences from $\sigma|_{A,B}$.

It follows that if $\sigma|_{A,B} = \sigma'|_{A,B}$ and $\tau|_{B,C} = \tau'|_{B,C}$, then $\sigma;\tau|_{A,C} = \sigma';\tau'|_{A,C}$. □

Now we can also see that this equivalence we have just defined is subsumed into the intrinsic equivalence.

Proposition 4.10.7. *Let $\sigma, \tau \colon !X \multimap A$ be strategies. If $\sigma \approx_{\text{may}} \tau$ then $\sigma \sim_{\text{may}} \tau$.*

Proof. Given strategies $\sigma, \tau \colon A$ in $\mathcal{G}_X$, we have $\sigma \sim_{\text{may}} \tau$ if and only if $\sigma; \alpha \approx_{\text{may}} \tau; \alpha$ for any morphism $\alpha \colon !A \multimap \mathbb{C}$ in $\mathcal{G}_X$. If $\sigma \approx_{\text{may}} \tau$, we have

$$\sigma; \alpha|_{\mathbb{C}} = (\sigma|_{\mathbb{C}}); \alpha = (\tau|_{\mathbb{C}}); \alpha = \tau; \alpha|_{\mathbb{C}} \,. \qquad \square$$

Note that it is also the case that if $\alpha \approx_{\text{may}} \alpha'$ and $\sigma \approx_{\text{may}} \tau$ then $\sigma; \alpha \approx_{\text{may}} \tau; \alpha'$. Therefore, the Full Abstraction result we have just proved applies to the quotiented category.

The definition of the relation $\approx_{\text{may}}$ suggests that we might forget about the $!X$ component of a strategy $\sigma \colon !X \multimap A$ altogether, and consider only the set $\sigma|_A$. This set is not a strategy, since it does not satisfy the determinism requirement, but it satisfies every other requirement.

Definition 4.10.8. Given a game A, a *nondeterministic strategy* is a prefix-closed set of even-length legal plays from A.

We can compose nondeterministic strategies using 'parallel composition plus hiding', just as for deterministic ones, and we get a Cartesian closed category in the same way. We interpret all the Idealized Algol terms in the usual way as deterministic strategies, interpreting the nondeterministic primitive ask_X as the nondeterministic strategy for X with maximal plays

$$qx$$

for every $x \in X$.

Now, if we have a Kleisli morphism $\sigma \colon !X \multimap A$, then $\sigma|_A$ is a nondeterministic strategy for A. The composition of nondeterministic strategies respects the composition and identity in the Kleisli category; moreover, any two Kleisli morphisms that give rise to the same nondeterministic strategy are intrinsically equivalent.

It is already known (see [HM99]) that this model is fully abstract for (finitely or countably) nondeterministic Idealized Algol with may-contextual equivalence. We have just provided an alternative proof of this fact.

4.11 Alternative Reduction Rules - Must Testing

A more interesting, and more complicated, reduction rule for nondeterministic IA is the *must-convergence* relation.

We shall define this indirectly via its negation.

Definition 4.11.1. We define a relation

$$\Gamma, s \vdash M \Uparrow$$

(M 'may diverge') between `Var`-contexts Γ, Γ-stores s and terms $\Gamma \vdash M : T$ of Idealized Algol.

The rules for this relation may be deduced from the rules for Idealized Algol. If

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow c, s^{(n)}},$$

is an IA rule, then for each $j = 1, \cdots, n$ we have a rule

$$\frac{\begin{array}{cc} \Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} & \cdots \\ \Gamma, s^{(j-2)} \vdash M_{j-1} \Downarrow c_{j-1}, s^{(j-1)} & \Gamma, s^{(j-1)} \vdash M_j \Uparrow \end{array}}{\Gamma, s^{(0)} \vdash M \Uparrow}.$$

In other words, if the evaluation of the term M diverges at any step, then the whole thing diverges. Note that since the rules for the canonical forms have no premises, there are no rules relating them to the $\Uparrow$ predicate.

Crucially, these rules are interpreted not inductively but coinductively; in other words, the derivation trees for $\Uparrow$ are allowed to have infinite height. Indeed, since every rule for $\Uparrow$ has at least one premise, any valid tree must be infinitely tall.

If there is no such derivation tree, then we say that M *must converge*, and write $\Gamma, s \vdash M \Downarrow^{\text{must}}$.

Example 4.11.2. If $\Gamma \vdash c$ is a canonical form, then $\Gamma \vdash c \Downarrow^{\text{must}}$, since there is no inference that we can apply that will yield the term $\Gamma \vdash c \Uparrow$ on the bottom.

Example 4.11.3. $\_, () \vdash \mathbf{Y}(\lambda x.x) \Uparrow$ because it has the following infinite derivation tree.

$$\dfrac{\dfrac{\dfrac{}{\_, () \vdash (\lambda x.x) \Downarrow (\lambda x.x), ()} \qquad \dfrac{\vdots}{\_, () \vdash \mathbf{Y}(\lambda x.x) \Uparrow}}{\_, () \vdash (\lambda x.x)(\mathbf{Y}(\lambda x.x)) \Uparrow}}{\_, () \vdash \mathbf{Y}(\lambda x.x) \Uparrow}$$

Remark 4.11.4. It is possible to characterize the $\Downarrow^{\text{must}}$ predicate directly via an inductive relation (see [Har99], for instance).

We will next give an alternative definition of $\Downarrow^{\text{must}}$ that will relate it to our existing rules $\Downarrow_u$.

Definition 4.11.5. We say that $\Gamma, s \vdash M \Downarrow^{\text{must}}$ if for every infinite sequence $w \in X^\omega$ there is some finite prefix $u \sqsubsetneq w$ such that $\Gamma, s \vdash M \Downarrow_u c, s'$ for some canonical form c and some Γ-store s'.

Before we show that the two definitions are equivalent, we shall examine why this one makes sense. Recall that we said that the statement $\Gamma, s \vdash M \Downarrow_u c, s'$ meant 'M will converge to c in the case that the values we obtain by querying the nondeterministic oracle form the sequence u'. Our initial thought, then, might be to declare that $\Gamma, s \vdash M \Downarrow^{\text{must}}$ if for all $u \in X^*$, $\Gamma, s \vdash M \Downarrow_u c, s'$ for some c, s'. However, this is not the case even for terms that clearly have to converge. For example, if we have

$$M = \mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ \mathbb{t}\ \mathsf{else}\ \mathbb{f}\,,$$

then $M \Downarrow_{\mathbb{t}} \mathbb{t}$ and $M \Downarrow_{\mathbb{f}} \mathbb{f}$, and it is certainly true that $M \Downarrow^{\text{must}}$. However, it is not the case that, for example, $M \Downarrow_\epsilon c$; nor is it the case that $M \Downarrow_{\mathbb{f}\mathbb{f}\mathbb{t}\mathbb{f}\mathbb{t}} c$ for any c. In the first case, the sequence is too short: we can extend it to the sequence $\mathbb{t}$ and then we get convergence. In the second case, the sequence is too long: we have already converged after the first term $\mathbb{f}$, so we never get the chance to read any other terms.

The idea is that if we fix beforehand an infinite sequence made up of all the values that the nondeterministic oracle could possibly give, then we know that the term will always end up reading some finite sequence of values from the beginning of the sequence and then terminating at a value.

Proposition 4.11.6. *The two definitions of the $\Downarrow^{must}$ predicate given in Definitions 4.11.1 and 4.11.5 are equivalent.*

Proof. Suppose that $\Gamma, s \vdash M \Uparrow$. We coinductively construct a sequence $w \in X^\omega$ such that $\Gamma, s \vdash M \not\Downarrow_u c, s'$ for any finite prefix $u \sqsubsetneq w$ and any c, s'. We shall use the notation $\Gamma, s \vdash M \Uparrow^w$ to indicate this w.

Given any rule

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow c_1, s^{(1)} \quad \cdots \quad \Gamma, s^{(j-2)} \vdash M_{j-1} \Downarrow c_{j-1} \qquad \Gamma, s^{(j-1)} \vdash M_j \Uparrow}{\Gamma, s^{(0)} \vdash M \Uparrow}\,,$$

by our definition of $\Downarrow$, there are $u_1, \cdots, u_{j-1} \in X^*$ such that $\Gamma, s^{(i-1)} \vdash M_i \Downarrow_{u_i} c_i, s^{(i)}$ for each $i = 1, \cdots, j-1$.

The corresponding coinductive rule is then

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \quad \cdots \quad \Gamma, s^{(j-2)} \vdash M_{j-1} \Downarrow_{u_{j-1}} c_{j-1} \quad \Gamma, s^{(j-1)} \vdash M_j \Uparrow^w}{\Gamma, s^{(0)} \vdash M \Uparrow^{u_1 \# \cdots \# u_{j-1} \# w}}\,.$$

In other words, the sequence w at the bottom of the tree may be characterized as the concatenation of all the sequences u constructed for the $\Downarrow$ relation in the tree, working bottom to top and left to right. If the w so formed is a finite sequence, then pad it with arbitrary values so that it becomes infinite.

Now suppose that $u \sqsubsetneq w$ is some finite prefix such that $\Gamma, s \vdash M \Downarrow_u c, s'$ for some c, s'. We claim that $\Gamma, s \vdash M \not\Uparrow$, giving us a contradiction.

The proof of the claim is via induction on the derivation of $\Gamma, s \vdash M \Downarrow_u c, s'$.

Suppose that the last rule takes the form

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow_{u_n} c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow_{u_1 + \cdots + u_n} c, s^{(n)}} \ .$$

Then inspection of the IA rules tells us that the last step in our derivation of $\Gamma, s^{(0)} \vdash M \Uparrow$ must look like

$$\frac{\begin{array}{c}\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \\ \cdots \quad \Gamma, s^{(j-2)} \vdash M_{j-1} \Downarrow_{u_{j-1}} c_{j-1} \quad \Gamma, s^{(j-1)} \vdash M_j \Uparrow^w\end{array}}{\Gamma, s^{(0)} \vdash M \Uparrow^{u_1 + \cdots + u_{j-1} + w}}$$

for the same M_i and c_i. But we had $\Gamma, s^{(j-1)} \vdash M_j \Downarrow_{u_j} c_j, s^{(j)}$. By hypothesis, $u_1 + \cdots + u_n$ is a finite prefix of $u_1 + \cdots + u_{j-1} + w$, and so u_j is a finite prefix of w. Therefore, by the inductive hypothesis applied to M_j, we cannot have $\Gamma, s^{(j-1)} \vdash M_j \Uparrow$, leading to the desired contradiction.

For the converse, define an *n-truncated* derivation tree to be a derivation tree of height n such that premises of the form $\Gamma, s \vdash M \Uparrow^w$ may occur at the top level without proof. We now define, for any infinite sequence $w \in X^\omega$, any triple $\Gamma, s \vdash M$ and any $i \geq 0$, either

- a proof of height $\leq i$ that $\Gamma, s \vdash M \Downarrow_u c, s'$ for some $u \sqsubsetneq w$ (and some c, s'); or
- an i-truncated proof $F_i(w, \Gamma, s \vdash M)$ of $\Gamma, s \vdash M \Uparrow^v$ for some (possibly infinite) $v \sqsubseteq w$.

Moreover, the $F_i(w, \Gamma, s \vdash M)$ will form an infinite sequence, so that if $i < j$ then $F_i(w, \Gamma, s \vdash M)$ is the result of truncating $F_j(w, \Gamma, s \vdash M)$ at the i-th level.

For $i = 0$, $F_i(w, \Gamma, s \vdash M)$ is the tree of the form

$$\Gamma, s \vdash M \Uparrow^w ,$$

where this statement is treated as an unproven (and in many cases untrue) hypothesis.

Now we define $F_{i+1}(w, \Gamma, s \vdash M)$ as follows. If M is a canonical form, then $F_{i+1}(w, \Gamma, s \vdash M)$ is the derivation.

$$\frac{}{\Gamma, s \vdash M \Downarrow_\epsilon M, s} \ .$$

If $M = \mathsf{ask}_X$, then $F_{i+1}(w, \Gamma, s \vdash M)$ is the derivation

$$\frac{}{\Gamma, s \vdash M \Downarrow_{w^{(0)}} w^{(0)}, s} \ .$$

Otherwise, based on the structure of M, we choose a first premise M_1 for M. In most cases, there is a unique IA_X rule that applies to M, but sometimes (i.e., if M is $\mathsf{pred}\, M'$, $\mathsf{If}\, M'\ \mathsf{then}\ N'\ \mathsf{else}\ P'$, $\mathsf{If0}\, M'\ \mathsf{then}\ N'\ \mathsf{else}\ P'$, $!V$ or $V \leftarrow E$) there may be more than one possible rule. However, in all of these cases, the first premise of each rule is the same. For instance, if $M = \mathsf{If}\, M'\ \mathsf{then}\ N'\ \mathsf{else}\ P'$ then the first premise of both the possible IA rules is M', and the value that this converges to then determines which instance of the If rule we use.

We evaluate $F_i(w, \Gamma, s \vdash M_1)$. If $F_i(w, \Gamma, s \vdash M_1)$ is a complete proof that $\Gamma, s \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)}$ for $u \sqsubsetneq w$, then write $w = u_1 w_2$ and move on to the next premise, computing $F_i(w_2, \Gamma, s \vdash M_2)$. Note that now the value of c_1 completely determines the IA_X rule we are using; for example, if $M = \mathsf{If}\, M'\ \mathsf{then}\ N'\ \mathsf{else}\ P'$, and $\Gamma, s \vdash M' \Downarrow_u c, s'$, then c is either $\mathbb{t}$ or $\mathbb{f}$, and there is a unique IA_X rule that applies in each case.

We keep going through the premises of M in order. If we satisfy them all, so that $F_i(w_i, \Gamma, s^{(i-1)} \vdash M_i)$ is a proof that $\Gamma, s^{(i-1)} \vdash M_i \Downarrow_{u_i} c_i, s^{(i)}$ for each M_i, then we can put these together to get a derivation of

$$\Gamma, s \vdash M \Downarrow_{u_1 \# \cdots \# u_n} c, s^{(n)} \,,$$

of height $\leq i + 1$, where $u_1 \# \cdots \# u_n$ is a finite prefix of w.

Otherwise, at some point $F_i(w, \Gamma, s^{(j-1)} \vdash M_j)$ must be an i-truncated proof that

$$\Gamma, s^{(j-1)} \vdash M_j \Uparrow^v$$

for some $v \sqsubseteq w$. In that case, since we have a proof of height $\leq i$ for $\Gamma, s^{(k-1)} \vdash M_k \Downarrow_{u_k} c_k, s^{(k)}$ for each $k < j$, we can apply the appropriate rule for $\Uparrow^v$ to get an $(i+1)$-truncated proof that

$$\Gamma, s \vdash M \Uparrow^{u_1 \# \cdots \# u_{j-1} \# v} \,,$$

where $u_1 \# \cdots \# u^{j-1} \# v$ is a prefix of w.

Now, given w and $\Gamma, s \vdash M$, either $F_m(\Gamma, s \vdash M)$ is a proof that $\Gamma, s \vdash M \Downarrow_u c, s'$ for $u \sqsubsetneq w$ and m sufficiently large, or the $F_i(w, \Gamma, s \vdash M)$ form an infinite

sequence of trees of increasing height that all extend one another. In the second case, we can stitch the trees together to form an infinite derivation tree for $\Gamma, s \vdash M \Uparrow$.

It follows that if there is some w such that $\Gamma, s \vdash M \not\Downarrow_u c, s'$ for any finite prefix $u \sqsubsetneq w$, then there is an infinite derivation tree for $\Gamma, s \vdash M \Uparrow$. □

We can now get an adequacy result for must testing.

Definition 4.11.7. Given a morphism $\sigma \colon 1 \to \mathbb{C}$ in $\mathcal{G}_X$, considered as a morphism $X \to \mathbb{C}$ in $\mathcal{G}$, we write $\sigma \downarrow_{\text{must}}$ if whenever $w \in X^\omega$ is an infinite sequence, then w has a finite prefix u such that the composite

$$1 \xrightarrow{\sigma} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$.

Corollary 4.11.8. *Let $M \colon \mathtt{com}$ be a closed term of IA_X and consider the denotation $[\![M]\!] : 1 \to \mathbb{C}$ in $\mathcal{G}_X$ as a morphism $1 \to (X \to \mathbb{C})$ in $\mathcal{G}$. Then $\_, () \vdash M \Downarrow^{must}$ if and only if $[\![M]\!] \downarrow_{\text{must}}$.*

Once again, this slightly abstruse Definition 4.11.7 becomes much more natural when $\mathcal{G}$ is the category of games, though in this case things are a bit more subtle. Let σ be a strategy for $!X \multimap \mathbb{C}$. We observed earlier that $\sigma; \eta_u; [\![\mathtt{new}]\!]; t_{|u|} \neq \bot$ if and only if the sequence

$$\begin{array}{ccc} !X & \multimap & \mathbb{C} \\ & & q \\ q & & \\ u^{(0)} & & \\ \vdots & & \\ q & & \\ u^{(|u|-1)} & & \\ & & a \end{array}$$

is contained in σ. Now fix some infinite sequence $w \in X^\omega$ and consider what it means if we say that such a sequence is not contained in σ for any $u \sqsubsetneq w$. By the definition of a strategy, there are two reasons why this might happen. Either there is a partiality in the strategy, so that for some finite

$v \sqsubsetneq w$, player P has no reply at all to the O-position

$$\begin{array}{ccc} !X & \multimap & \mathbb{C} \\ & & q \\ q & & \\ v^{(0)} & & \\ \vdots & & \\ q & & \\ v^{(|v|-1} & & \\ & & a\,. \end{array}$$

Or, σ contains an infinite increasing sequence of plays, whose limit is the infinite sequence

$$\begin{array}{ccc} !X & \multimap & \mathbb{C} \\ & & q \\ q & & \\ v^{(0)} & & \\ q & & \\ v^{(1)} & & \\ \vdots & & \end{array}$$

Definition 4.11.9. We say that a strategy $\sigma \colon !X \multimap \mathbb{C}$ is *winning* if σ is total and contains no infinite plays.

We have shown the following adequacy result.

Corollary 4.11.10. *Let $M \colon \mathtt{com}$ be a closed term of IA_X and consider the denotation $[\![M]\!] : 1 \to \mathbb{C}$ in $\mathcal{G}_X$ as a morphism $X \to \mathbb{C}$ in $\mathcal{G}$. Then*

$$\_, () \vdash M \Downarrow^{must}$$

if and only if this strategy is winning.

4.12 Full Abstraction for Finite Nondeterminism Under Must Testing

We now extend our earlier definitions using the new $\Downarrow^{\text{must}}$ rule.

Definition 4.12.1. Given closed terms $M, N \colon T$ of IA_X, we write

$$M \equiv_{\text{must}} N$$

if for all contexts $- \colon T \vdash C[-] \colon \mathtt{com}$, $C[M] \Downarrow^{\text{must}}$ if and only if $C[N] \Downarrow^{\text{must}}$.

We write $M \equiv_{m\&m} N$ if $M \equiv_{\text{may}} N$ and $M \equiv_{\text{must}} N$.

Definition 4.12.2. Given morphisms $\sigma, \tau \colon 1 \to \mathbb{C}$ in $\mathcal{G}_X$, we write $\sigma \approx_{\text{must}} \tau$ if $\sigma \downarrow_{\text{must}}$ and $\tau \downarrow_{\text{must}}$ or if $\sigma \not\downarrow_{\text{must}}$ and $\tau \not\downarrow_{\text{must}}$.

We write $\sigma \approx_{m\&m} \tau$ if $\sigma \approx_{\text{may}} \tau$ and $\sigma \approx_{\text{must}} \tau$.

Definition 4.12.3. Given morphisms $\sigma, \tau \colon A \to B$ in $\mathcal{G}_X$, considered as morphisms $1 \to (A \to B)$, we write $\sigma \sim_{\text{must}} \tau$ if whenever $\alpha \colon (A \to B) \to \mathbb{C}$ is a morphism in $\mathcal{G}_X$, then $\sigma; \alpha \approx_{\text{must}} \tau; \alpha$.

We write $\sigma \sim_{m\&m} \tau$ if $\sigma \sim_{\text{may}} \tau$ and $\sigma \sim_{\text{must}} \tau$.

Remark 4.12.4. In the category of games we can simplify the definition of $\sim_{m\&m}$ to saying that $\sigma \sim_{m\&m} \tau$ if for all $\alpha \colon (A \to B) \to \mathbb{C}$, $\sigma; \alpha \approx_{\text{may}} \tau; \alpha$ and either $\sigma; \alpha$ and $\tau; \alpha$ are both winning, or neither is.

Now we might expect to be able to prove a full abstraction result as before, namely that $[\![M]\!] \sim_{m\&m} [\![N]\!]$ if and only if $M \equiv_{m\&m} N$. In the case of finite nondeterminism, this is possible.

Theorem 4.12.5. *Let $M, N \colon T$ be closed terms of $IA_{\mathbb{B}}$. Then $M \equiv_{m\&m} N$ if and only if $[\![M]\!] \sim_{m\&m} [\![N]\!]$.*

Proof. We already know that $[\![M]\!] \sim_{\text{may}} [\![N]\!]$ if and only if M and N are may-contextually equivalent, so it is sufficient to prove that $[\![M]\!] \sim_{\text{must}} [\![N]\!]$ if and only if $M \equiv_{\text{must}} N$.

One direction is easy, as usual. Suppose that $[\![M]\!] \sim_{\text{must}} [\![N]\!]$, and let $C[-]$ be a context. Suppose that $C[M] \Downarrow^{\text{must}}$, and fix $w \in X^\omega$. Then $[\![C[M]]\!] = [\![C]\!]; [\![M]\!]$, so there is some finite prefix $u \sqsubsetneq w$ such that the composite

$$1 \xrightarrow{[\![C]\!];[\![M]\!]} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$. Since $M \sim_{\text{must}} N$, there is some finite prefix $v \sqsubsetneq w$ and some compact $\alpha' \colon [\![T]\!] \to \mathbb{C}$ such that the composite

$$1 \xrightarrow{[\![C]\!];[\![N]\!]} (X \to \mathbb{C}) \xrightarrow{\eta_v} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|v|}} \mathbb{C}$$

is not equal to $\bot$, and therefore $C[N] \Downarrow^{\text{must}}$. The other direction is completely symmetric.

The converse is harder. Let $\alpha \colon [\![T]\!] \to \mathbb{C}$ be an arbitrary morphism. Suppose that for all $w \in \mathbb{B}^\omega$ there is some finite $u \sqsubsetneq w$ such that the composite

$$[\![M]\!]; (X \to \alpha); \mu; \eta_u; [\![\mathsf{new}]\!]; t_{|u|}$$

is not equal to $\bot$. Let V be the set of all sequences $v \in \mathbb{B}^*$ such that v has no prefix u with this property. Then V is a finitely branching tree with no

infinite path so, by König's Lemma, it is finite. Therefore, there is some finite set of sequences $U \subseteq \mathbb{B}^*$ such that for every sequence $w \in \mathbb{B}^\omega$, w has some prefix $u \in U$ such that $[\![M]\!]; (X \to \alpha); \mu; \eta_u; [\![\mathsf{new}]\!]; t_{|u|} \neq \bot$.

Since $\mathcal{G}$ is enriched in algebraic cpos, for each $u \in U$ there is some compact $\alpha^u \subseteq \alpha$ such that

$$[\![M]\!]; (X \to \alpha^u); \mu; \eta_u; [\![\mathsf{new}]\!]; t_{|u|}$$

is not equal to $\bot$. Since the set of compact elements below α is directed, there is some compact $\alpha' \subseteq \alpha$ such that $\alpha^u \subseteq \alpha'$ for each $u \in U$.

So we now know that: for any infinite sequence $w \in \mathbb{B}^\omega$, there is some finite prefix $u \sqsubsetneq w$ such that the composite

$$[\![M]\!]; (X \to \alpha'); \mu; \eta_u; [\![\mathsf{new}]\!]; t_{|u|}$$

is not equal to $\bot$. Now if $[\![N]\!] \not\sim_{\text{must}} [\![M]\!]$, then there is some infinite $w \in \mathbb{B}^\omega$ is such that for every $v \sqsubsetneq w$ the composite

$$[\![N]\!]; (X \to \alpha); \mu; \eta_v; [\![\mathsf{new}]\!]; t_{|v|}$$

is equal to $\bot$; since $\alpha' \subseteq \alpha$, the same is true if we replace α with α'.

Now, since α' is compact, it is definable as a term in IA and hence as a term L in $\text{IA}_{\mathbb{B}}$. By Corollary 4.11.8, this means that $LM \Downarrow^{\text{must}}$, while $LN \Uparrow$, so $M \not\equiv_{\text{must}} N$. □

This last result makes crucial use of the finiteness of $\mathbb{B}$ in order to apply König's lemma. In the case of $\text{IA}_{\mathbb{N}}$, the argument we have just used does not apply, since we can certainly have infinite bounded-height trees if we allow infinite branching.

In fact, this point is essential: not only does this specific argument not work for our model of $\text{IA}_{\mathbb{N}}$, the model is not necessarily Fully Abstract at all! In particular, if $\mathcal{G}$ is the category of games, then the semantics of must testing in $\mathcal{G}_{\mathbb{N}}$ is not Fully Abstract. The next result will show that no model of countable nondeterminism can be Fully Abstract, subject to some rather mild assumptions. We will then explore how we can come up with a model for which these assumptions do not hold, and eventually prove a Full Abstraction result.

4.13 The Kleene Tree

In order to show that our model is not Fully Abstract, we start by introducing the Kleene tree.

We define a *binary tree* to be a prefix-closed subset of $\mathbb{B}^*$. Using the binary representation, we can identify $\mathbb{B}^*$ with $\mathbb{N}$, meaning that every such tree can be specified by its characteristic function $\chi\colon \mathbb{N} \to \mathbb{B}$.

We say that the tree is *computable* if there is some IA term $M\colon \mathtt{nat} \to \mathtt{bool}$ such that Mn converges to $\chi(n)$ for all $n \in \mathbb{N}$.

In this section, we will show the construction of the *Kleene tree* (see [Sti18, 4.5] and [Bau06]). This is a tree K that has the following seemingly paradoxical properties.

- It is computable.
- It has an infinite path; that is, there are $b_0, b_1, \cdots \in \mathbb{B}$ such that for all k, the string $b_0 \cdots b_{k-1}$ is contained in K.
- It has no computable infinite path. I.e., if $M\colon \mathtt{bool} \to \mathtt{nat}$ is a term of IA such that Mi converges to some $a_i \in \mathbb{B}$ for each $i \in \mathbb{N}$, then there is some k such that the string $a_0 \cdots a_{k-1}$ is not contained in K.

In order to construct K, we are first going to assume that we have fixed some sensible encoding of IA terms as binary strings and hence as natural numbers. We shall further assume the existence of an *interpreter* for IA terms of type `bool`; that is, a term $\mathrm{int}\colon \mathtt{nat} \to \mathtt{nat}$ such that:

- if n is the encoding of some $M\colon \mathtt{bool}$ such that $M \Downarrow \mathtt{t}$, then $\mathrm{int}\, n \Downarrow 0$;
- if n is the encoding of some $M\colon \mathtt{bool}$ such that $M \Downarrow \mathtt{f}$, then $\mathrm{int}\, n \Downarrow 1$;
- if n is the encoding of some $M\colon \mathtt{bool}$ such that M diverges, then $\mathrm{int}\, n$ diverges; and
- if n is not the encoding of a term of type `bool`, then $\mathrm{int}\, n \Downarrow 255$.

The construction of int is more a matter of software engineering than anything else. The idea is to use pattern matching and recursion to implement the small-step semantics of Idealized Algol within the encoding we have fixed.

In fact, we shall need a slightly modified version of int that accepts an extra *timeout* parameter. That is, we require some $\mathrm{int}'\colon \mathtt{nat} \to \mathtt{nat} \to \mathtt{nat}$ such that

- if n is the encoding of some $M\colon \mathtt{bool}$ such that $M \Downarrow \mathtt{t}$ in under k steps, then $\mathrm{int}'\, k\, n \Downarrow 0$;
- if n is the encoding of some $M\colon \mathtt{bool}$ such that $M \Downarrow \mathtt{f}$ in under k steps, then $\mathrm{int}'\, k\, n \Downarrow 1$; and
- if n is the encoding of some $M\colon \mathtt{bool}$ such that M does not converge after k steps, or if n is not the encoding of a term of type `bool`, then

$\mathrm{int}'\, k\, n \Downarrow 255$.

This modification to the original int is not too hard to make, but now we can be assured that int′ will always converge, even if fed the encoding of a divergent program. This will be important later.

Definition 4.13.1 (Kleene tree)**.** We construct the tree K as follows. Suppose $b = b_0, \cdots, b_{k-1}$ is a finite sequence of binary digits. Then b is contained in K if and only if for all $n = 0, \cdots, k-1$, if n is the encoding of a program $M : \mathtt{nat} \to \mathtt{bool}$ such that Mn converges in under k steps, then Mn converges to a value different from b_n.

Remark 4.13.2. We can use int′ to construct an IA term $\mathtt{nat} \to \mathtt{bool}$ that computes the characteristic function of K. Since the syntax of IA is not very transparent, we shall describe the operation of the term informally. Given an input n, treated as a sequence $b_0 \cdots \mathfrak{b}_{k-1}$, we iterate from 0 up to $k-1$. For each number p, if p is the encoding of a term $M : \mathtt{nat} \to \mathtt{bool}$, we pass the encoding of Mp to int′, giving k as the timeout parameter. If we get 0 (true) or 1 (false) back, then we compare against b_p, returning $\mathbb{f}$ if the values are equal and $\mathbb{t}$ if they are different. If we get the error value 255 back, then we return $\mathbb{t}$.

Lastly, if we have got $\mathbb{t}$ back for each p, we return $\mathbb{t}$; otherwise, we return $\mathbb{f}$.

Proposition 4.13.3. *i) K has an infinite path.*

ii) K has no computable infinite path.

Proof. (i): For each p such that p is the encoding of some $M : \mathtt{nat} \to \mathtt{bool}$ such that Mp converges, let a_p be $\mathbb{t}$ if $Mp \Downarrow \mathbb{f}$ and $\mathbb{f}$ if $Mp \Downarrow \mathbb{t}$. Otherwise – i.e., if Mp does not converge or if p is not the encoding of a term $\mathtt{nat} \to \mathtt{bool}$ – then return any value – $\mathbb{f}$, say. Then any sequence $a_0 \cdots a_{k-1}$ must be contained in K.

(ii): Let $M : \mathtt{nat} \to \mathtt{bool}$ be such that there are $b_i \in \mathbb{B}$ for which $Mi \Downarrow b_i$ for all i, let p be the encoding of M and suppose that Mp converges in k steps. Let $N = \max\{k-1, p\}$. We claim that $b_0 \cdots b_N$ is not contained in K. Indeed, p is the encoding of M and Mp converges to b_p in under N steps. □

4.14 Non-computable functions and Observational Equivalence

The presence of non-computable functions in the semantics for a language does not normally present a problem for Full Abstraction. In previous sec-

tions, we have been able to appeal to continuity results that tell us that the intrinsic equivalence relation in our model is determined entirely by some collection of *compact* (and hence computable) morphisms. In the presence of countable nondeterminism, these continuity results no longer hold, and the intrinsic equivalence becomes sensitive to noncomputable elements in the semantics.

To investigate this, we define a language IA+Functions by extending IA with, for each (set-theoretic) function $f\colon \mathbb{N} \to \mathbb{B}$, a primitive $\Phi_f\colon \mathtt{nat} \to \mathtt{bool}$ with the following operational semantics.

$$\frac{\Gamma, s \vdash N \Downarrow n, s'}{\Gamma, s \vdash \Phi_f\, N \Downarrow f(n), s'}$$

We make some new claims.

Proposition 4.14.1. *i) The Game Semantics model from Sections 2 and 3 is Computationally Adequate and Fully Abstract for IA+Functions.*

ii) Two terms of IA are observationally equivalent within IA if and only if they are observationally equivalent when we consider them as terms of IA+Functions.

Proof. (i): The denotation of Φ_f is the strategy with maximal plays of the form

$$\begin{array}{cc} \mathbb{N} & \mathbb{N} \\ & q \\ q & \\ n & \\ & f(n) \end{array},$$

for $n \in \mathbb{N}$. In the notation of Section 2.15, this is the strategy given by the formula

$$(f(0), f(1), f(2), \cdots)\,.$$

Computational adequacy for this function follows by exactly the same arguments as for the functions succ and pred; note that we used no special properties of the successor or predecessor functions in Chapter 3, so exactly the same arguments go through for any function $\mathbb{N} \to \mathbb{N}$. Full Abstraction also goes through without modification.

(ii): This follows from (i), since the two languages share a Fully Abstract model. □

In the presence of countable nondeterminism, however, we shall show that the equivalent of (ii) does not hold; i.e., that there are observationally equivalent terms of nondeterministic IA that are observationally distinguishable within IA+Functions+Countable nondeterminism.

This will then give the reason why our proposed semantics cannot be fully abstract: our starting model is fully abstract for both IA and IA+Functions and we have already proved that the nondeterministic version is Equationally Sound for nondeterministic IA (and, by a similar argument, for IA+Functions+Countable nondeterminism). Equational Soundness tells us that any observationally distinguishable terms of IA+Functions+Countable nondeterminism must therefore be distinguished by the model, proving that it cannot be fully abstract for IA+Countable nondeterminism.

Let us now examine the two terms that we shall be considering. First, we define $M\colon (\mathtt{nat} \to \mathtt{bool}) \to \mathtt{com}$ by the formula

$$\begin{aligned} M = &\lambda f.\, \mathsf{let}\, k =? \,\mathsf{in} \\ &\mathsf{If}\, K((f\,0)\cdots(f\,(k-1)))\, \mathsf{then}\, (\mathsf{skip}\,\mathsf{or}\,\mathsf{skip})\, \mathsf{else}\, (\mathsf{skip}\,\mathsf{or}\,\Omega)\,, \end{aligned}$$

where

$$((f\,0)\cdots(f\,(k-1)))$$

is shorthand for the natural number corresponding to the binary sequence given by $(f\,0)$ up to $(f\,(k-1))$, or is binary choice and K is the Kleene tree from the previous section. Now define $N\colon (\mathtt{nat} \to \mathtt{bool}) \to \mathtt{com}$ by the formula

$$\begin{aligned} N = &\lambda f.\, \mathsf{let}\, k =? \,\mathsf{in}\, (f\,0); \cdots ; (f\,(k-1)); \\ &\mathsf{skip}\,\mathsf{or}\,\Omega\,, \end{aligned}$$

where we have used the notation $P; Q$ (for $P\colon \mathtt{nat}$) as a shorthand for

$$\mathsf{If0}\, P\, \mathsf{then}\, Q\, \mathsf{else}\, Q\,;$$

i.e., the program that evaluates P, forgets the result and then evaluates Q. The reason for evaluating $(f\,0), \cdots, (f\,(k-1))$ before either returning or diverging is to make sure that any side effects of the evaluation of f are the same for both programs (we shall assume that K operates in such a way that it evaluates each of the $(f\,i)$ exactly once).

We now claim that M and N are observationally equivalent in nondeterministic IA. We will postpone the formal proof of this fact until we have a working Fully Abstract semantics for that language, but a rough argument is as follows. M and N can only be distinguished by an f such that

$$K((f\,0)\cdots(f\,(k-1)))$$

evaluates to $\mathtt{t}$ for all k. Otherwise, both terms may diverge. But we have shown in the previous section that there is no computable f with this property.

Meanwhile, M and N are not observationally equivalent in nondeterministic IA+Functions; indeed, if $\kappa\colon \mathbb{N} \to \mathbb{B}$ is an infinite path in the Kleene tree, then M and N are distinguished by Φ_κ.

4.15 Full Abstraction for Countable Nondeterminism under Must Testing

The previous two sections make it clear that the problem is the existence in the semantics of morphisms that behave like certain noncomputable functions. The presence of such morphisms is normally an orthogonal issue to full abstraction, since we can usually assume that the strategy α in the definition of intrinsic equivalence is compact. However, in our case, the presence of these noncomputable morphisms is a big problem.

In order to get around this problem, then, we will clearly need to disallow noncomputable functions $\mathbb{N} \to \mathbb{N}$, which quickly entails that we should disallow all morphisms that are not computable.

Such models were considered by Hyland and Ong [HO00] and by Abramsky, Jagadeesan and Malacaria [AJM00] in their original proofs of Full Abstraction for PCF. In these models, a game (or arena) A comes equipped with an encoding function $M_A \to \mathbb{N}$; a strategy is said to be *recursive* if it can be defined by a recursive function $\mathbb{N} \to \mathbb{N}$. The remarkable thing about these models is that they satisfy a property called *universality*: *every* morphism into the denotation of one of the types of the language is definable.

Suppose our model $\mathcal{G}$ of Idealized Algol satisfies universality. Then it is relatively easy to pass from Computational Adequacy to Full Abstraction.

Theorem 4.15.1. *Let $M, N : T$ be terms of IA_X and suppose that our base semantics of IA in $\mathcal{G}$ satisfies universality. Then $M \equiv_{m\&m} N$ if and only if $[\![M]\!] \sim_{m\&m} [\![N]\!]$.*

Proof. See the proof of Theorem 4.12.5 for the proof that if $[\![M]\!] \sim_{m\&m} [\![N]\!]$ then $M \equiv_{m\&m} N$.

Conversely, if $[\![M]\!] \not\sim_{m\&m} [\![N]\!]$, then without loss of generality there is some $\alpha : [\![T]\!] \to \mathbb{C}$ such that $[\![M]\!]; \alpha \downarrow_{\text{must}}$ and $[\![N]\!]; \alpha \not\downarrow_{\text{must}}$. But by universality $[\![M]\!]; \alpha = [\![LM]\!]$ and $[\![N]\!]; \alpha = [\![LN]\!]$ for some term $L : T \to \texttt{com}$ of IA. Therefore, the result follows from Corollary 4.11.8. □

The next step is to find an example of such a $\mathcal{G}$. As we said earlier, we will do this by cutting our game semantics down so that it only contains *computable* strategies. The first step is to fix an enumeration $e_A : M_A \to \mathbb{N}$ of the moves of each game A. We do this for the ground type games and

then extend to the connectives. For example:

$$e_{\mathbb{C}}(q) = 0 \qquad e_{\mathbb{C}}(a) = 1 \qquad e_{\mathbb{B}}(q) = 0 \qquad e_{\mathbb{B}}(\mathtt{t}) = 1 \qquad e_{\mathbb{B}}(\mathtt{f}) = 2$$

$$e_{\mathbb{N}}(q) = 0 \qquad e_{\mathbb{N}}(n) = n + 1 \qquad e_{\mathtt{Var}}(q) = 0 \qquad e_{\mathtt{Var}}(n) = 3n + 1$$

$$e_{\mathtt{Var}}(q_n) = 3n + 2 \qquad e_{\mathtt{Var}}(a_n) = 3n + 3$$

$$e_{A\otimes B}(\mathrm{in}_A(a)) = 2e_A(a) \qquad e_{A\otimes B}(\mathrm{in}_B(b)) = 2e_B(b) + 1$$

...and so on.

We can extend the enumeration of moves to an enumeration of (justified) plays in A. We then say that a strategy σ is *recursive* if it is a recursively enumerable subset of P_A. Recursive visible strategies give us a Cartesian closed subcategory of the original category of games and visible strategies.

We now appeal to some results about these recursive strategies. For example, the following was proved in [HO00].

Theorem 4.15.2 (Universality for PCF)**.** *Let A be the denotation of a PCF type T Then every recursive innocent strategy for A is innocently intrinsically equivalent to the denotation of some PCF term $M : T$.*

This result is not quite enough for us, since the innocent intrinsic equivalent is coarser than that for visible strategies. However, if we add the constant let to PCF, then we can get a stronger result that gives us definability *on the nose*.

Theorem 4.15.3 (Unpublished, but see [LN15, §7.1.5])**.** *Let A be the denotation of a PCF type T. Then every recursive innocent strategy for A is the denotation of a term $M : T$ of PCF+let.*

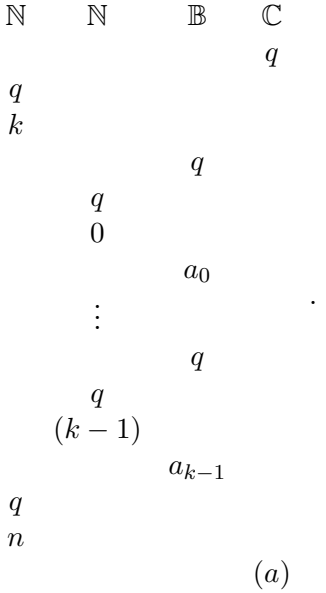
Moreover, these results are all easily extended to the IA ground types that are not present in PCF. So we see that any recursive innocent $\sigma : [\![T]\!]$ is the denotation of some Idealized Algol term $M : T$.

Lastly, we observe that our innocent factorization result (Proposition 3.5.1) will always produce a recursive innocent strategy if the original strategy is recursive. It follows, then, that any recursive *visible* $\sigma : [\![T]\!]$ is the denotation of some Idealized Algol term $M : T$. Thus, the category $\mathcal{G}^{rec}$ of games and recursive visible strategies is universal for Idealized Algol, and therefore its Kleisli category $\mathcal{G}^{rec}_X$ is fully abstract for IA_X.

As an application of this fully abstract semantics, we clear up a loose end from Section 4.14.

Lemma 4.15.4. *The terms $M, N\colon (\mathtt{nat} \to \mathtt{bool}) \to \mathtt{com}$, as defined in Section 4.14, are observationally equivalent in Idealized Algol with countable nondeterminism.*

Proof. The denotations for M and N both have maximal plays that look like this:



Here, the copy of $\mathbb{N}$ on the left denotes the nondeterministic oracle given by the Kleisli category. The final move a is in brackets to indicate that it does not always occur, depending on the moves that have gone before.

More specifically, in $[\![M]\!]$, the final move a always occurs if $a_0 \cdots a_{k-1}$ is a finite path in the Kleene tree; otherwise, it may or may not occur, depending on whether n is 0 or not.

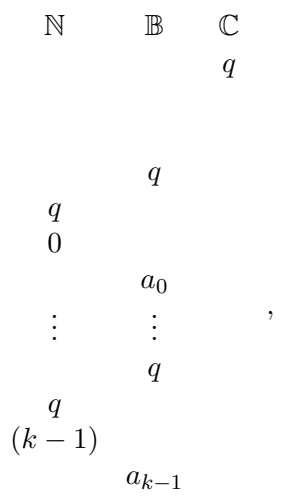
Meanwhile, the final move a is present in $[\![N]\!]$ if and only if the number n is equal to 0. This means that $[\![N]\!]$ is a subset of $[\![M]\!]$.

Fix some strategy $\alpha\colon ((\mathbb{N} \to \mathbb{C}) \to \mathbb{C}) \to \mathbb{C}$ in $\mathcal{G}_X$ and suppose that

$$[\![M]\!]\,;\alpha \not\approx_{m\&m} [\![N]\!]\,;\alpha\,.$$

First note that we always have $[\![M]\!]\,;\alpha \approx_{\text{may}} [\![N]\!]\,;\alpha$ for arbitrary α, which we can deduce by ignoring the copy of $\mathbb{N}$ on the left and considering $[\![M]\!]$ and $[\![N]\!]$ as nondeterministic strategies. In this setting, $[\![M]\!]$ and $[\![N]\!]$ are

equal, since both have maximal plays of the form



(a)

where k is arbitrary.

Therefore, we must have $[\![M]\!]\,;\alpha \not\approx_{\text{must}} [\![N]\!]\,;\alpha$.

Since $[\![N]\!] \subseteq [\![M]\!]$, it must be the case that $[\![M]\!]\,;\alpha \downarrow_{\text{must}}$ and $[\![N]\!]\,;\alpha \not\downarrow_{\text{must}}$; i.e., that $[\![M]\!]\,;\alpha$ is a winning strategy, when considered as a strategy $\mathbb{N} \to \mathbb{C}$ in $\mathcal{G}$, but $[\![N]\!]\,;\alpha$ is not.

Since $[\![M]\!]$ and $[\![N]\!]$ do not differ until the very last move, this means that α must respond to the initial move q in the rightmost copy of $\mathbb{C}$ with the move q in the copy of $\mathbb{C}$ on the left (otherwise, $[\![N]\!]\,;\alpha$ is winning), and that α must be total (otherwise, $[\![M]\!]\,;\alpha$ is not winning). α replies to each move i in $\mathbb{N}$ with some move in $\mathbb{B}$, and this defines a function $\mathbb{N} \to \mathbb{B}$. Since α is total, this function must trace out an infinite path in the Kleene tree.

But if α were recursive, then we would be able to recover this infinite path from α via a recursive algorithm, and we know that this is not possible Therefore, α is not recursive.

It follows that M and N are observationally equivalent in Idealized Algol with countable nondeterminism under may-and-must testing. □

Note that we certainly have $\sigma_\phi; [\![M]\!] \downarrow_{\text{must}}$, while $\sigma_\phi; [\![N]\!] \not\downarrow_{\text{must}}$, demonstrating that the recursivity requirement is essential to the full abstraction result in this case.

4.16 The Intrinsic Equivalence Relation

We conclude this chapter by examining the intrinsic equivalence relation in $\mathcal{G}_X$.

Proposition 4.16.1. *Let $\sigma, \tau \colon A \to B$ be two morphisms in $\mathcal{G}_X$, If $\sigma \sim \tau$ when considered as morphisms $X \to (A \to B)$ in $\mathcal{G}$, then $\sigma \sim \tau$ in $\mathcal{G}_X$.*

Proof. Suppose that $\alpha \colon (A \to B) \to \mathbb{C}$ separates σ and τ in $\mathcal{G}_X$, so without loss of generality, we have

$$1 \xrightarrow{(\sigma;\alpha)} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C} = \bot$$

$$1 \xrightarrow{(\tau;\alpha)} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathtt{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C} \neq \bot$$

for some u. Expanding the Kleisli compositions $\sigma; \alpha$ and $\tau; \alpha$, we get

$$\sigma; (X \to \alpha); \mu; \eta_u; [\![\mathsf{new}]\!]\,; t_{|u|} = \bot$$

$$\tau; (X \to \alpha); \mu; \eta_u; [\![\mathsf{new}]\!]\,; t_{|u|} \neq \bot\,,$$

and so $(X \to \alpha); \mu; \eta_u; [\![\mathsf{new}]\!]\,; t_{|u|}$ distinguishes σ and τ, when considered as morphisms $X \to (A \to B)$ in $\mathcal{G}$. □

As a special case, if $\sigma, \tau \colon A \to B$ are morphisms in $\mathcal{G}$, then $\sigma \sim \tau$ in $\mathcal{G}$ if and only if $J\sigma \sim J\tau$ in $\mathcal{G}_X$. Combined with our Full Abstraction results, this tells us that IA_X is a conservative extension of Idealized Algol.

What is more, we have

$$\sigma \sim \tau \Rightarrow J\sigma \sim J\tau \Rightarrow J\sigma \sim_{m\&m} J\tau \Rightarrow \sigma \sim \tau\,.$$

This proves that the finite and countable nondeterministic variants of Idealized Algol are conservative extensions of Idealized Algol.

Chapter 5

Parametric monads

Since a monoid in a monoidal category $\mathcal{X}$ is the same thing as a lax monoidal functor from the unit category into $\mathcal{X}$, the definition of a monad on a category $\mathcal{C}$ (i.e., a monoid in $[\mathcal{C},\mathcal{C}]$) may be generalized to that of a *lax action* [JK02] of a monoidal category $\mathcal{X}$ on $\mathcal{C}$; i.e., a lax monoidal functor $\mathcal{X} \to [\mathcal{C},\mathcal{C}]$ (so that a monad on $\mathcal{C}$ is a lax action of the unit category on $\mathcal{C}$).

Equivalently, a lax action is given by a functor $\_.\_ : \mathcal{X} \times \mathcal{C} \to \mathcal{C}$ together with natural transformations

$$m_{x,y,a} \colon x.y.a \to (x \otimes y).a \qquad\qquad e_a \colon a \to I.a$$

such that the following diagrams commute.

$$\begin{array}{ccccc} x.y.z.a & \xrightarrow{x.m_{y,z,a}} & x.(y \otimes z).a & \xrightarrow{m_{x,y\otimes z,a}} & (x \otimes (y \otimes z)).a \\ {\scriptstyle m_{x,y,z.a}}\downarrow & & & \nearrow_{\mathrm{assoc}_{x,y,z}\,.a} & \\ (x \otimes y).z.a & \xrightarrow{m_{x\otimes y,z,a}} & ((x \otimes y) \otimes z).a & & \end{array}$$

$$\begin{array}{ccc} x.a & \xrightarrow{e_{x.a}} & I.x.a \\ & {\scriptstyle \mathrm{lunit}_x\,.a}\searrow & \downarrow{\scriptstyle m_{I,x,a}} \\ & & (I \otimes x).a \end{array} \qquad \begin{array}{ccc} x.a & \xrightarrow{x.e_a} & x.I.a \\ & {\scriptstyle \mathrm{runit}_x\,.a}\searrow & \downarrow{\scriptstyle m_{x,I,a}} \\ & & (x \otimes I).a \end{array}$$

Since lax actions are a generalization of monads, we shall sometimes follow Melliès [Mel17] and refer to them as *parametric monads*. We shall not be considering strong actions (for which m, e are required to be isomorphisms), so we shall sometimes say 'action' to mean 'lax action'.

Example 5.0.1. If $\mathcal{X}$ is a monoidal category, $\mathcal{C}$ is a monoidal closed category and $j \colon \mathcal{X} \to \mathcal{C}$ is an oplax monoidal functor, then we have a lax action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{C}$ given by

$$x.a = jx \multimap a\,,$$

together with the natural coherences

$$jx \multimap jy \multimap a \longrightarrow (jx \otimes jy) \multimap a \xrightarrow{m^j_{x,y} \multimap a} j(x \otimes y) \multimap a$$

$$a \longrightarrow I \multimap a \xrightarrow{e^j \multimap a} jI \multimap a\,.$$

This generalizes the *reader monads* that we met earlier, so we shall call a parametric monad of this form a *parametric reader monad* or *lax reader action*.

Definition 5.0.2. Suppose that a monoidal category $\mathcal{X}$ acts on a category $\mathcal{C}$ and that it also acts on a category $\mathcal{D}$. An *oplax morphism of actions* from one action to the other is given by a functor $F\colon \mathcal{C} \to \mathcal{D}$ together with a natural transformation $\mu_{x,a}\colon F(x.a) \to x.Fa$ that makes the following diagrams commute for all objects x, y of $\mathcal{X}$ and a of $\mathcal{C}$.

$$\begin{array}{ccccc} F(x.y.a) & \xrightarrow{\mu_{x,y.a}} & x.F(y.a) & \xrightarrow{x.\mu_{y,a}} & x.y.Fa \\ {\scriptstyle Fm_{x,y,a}}\downarrow & & & & \downarrow{\scriptstyle m_{x,y,Fa}} \\ F((x \otimes y).a) & & \xrightarrow{\quad\mu_{x\otimes y,a}\quad} & & (x \otimes y).Fa \end{array} \qquad \begin{array}{ccc} Fa & & \\ {\scriptstyle Fe_a}\downarrow & \searrow^{e_{Fa}} & \\ F(I.a) & \xrightarrow{\mu_{I,a}} & I.Fa \end{array}$$

5.1 The Melliès category

The main thing we want to do with lax actions is to perform a construction analogous to that of the Kleisli category of a monad. Fujii, Katsumata and Melliès give a construction called a 'Kleisli resolution' in the paper [FKM16], but we shall prefer an alternative construction due to Melliès.

Suppose that a monoidal category $\mathcal{X}$ acts on a category $\mathcal{C}$.

Definition 5.1.1. A *Melliès morphism* from a to b is a morphism of the form

$$a \to x.b\,,$$

for some object x of $\mathcal{X}$.

Remark 5.1.2. In what follows, the object x will be part of the data of the Melliès morphism, so a Melliès morphism from a to b is a pair (x, f), where $f\colon a \to x.b$ is an ordinary morphism. We will omit the x when it is clear from the context.

Definition 5.1.3. Given Melliès morphisms f from a to b and g from b to c given by

$$\tilde{f}\colon a \to x.b \qquad\qquad \tilde{g}\colon b \to y.c\,,$$

their *Melliès composition* is given by the composite

$$a \xrightarrow{\tilde{f}} x.b \xrightarrow{x.\tilde{g}} x.y.c \xrightarrow{m_{x,y,c}} (x \otimes y).c\,,$$

which is a Melliès morphism from a to c.

Having defined our basic notion of composition, we can move on to our main constructions. We start with some definitions that will help us navigate size issues.

Definition 5.1.4. Let $\mathcal{C}$ be a category. An *ancestral set* (see [KK81]) for $\mathcal{C}$ is a set $\mathcal{A}$ of objects of $\mathcal{C}$ such that for any object a of $\mathcal{C}$, there is an object $a' \in \mathcal{A}$ and a morphism $f\colon a' \to a$. A *descendent*[1] *set* for $\mathcal{C}$ is an ancestral set for $\mathcal{C}^{\mathrm{op}}$; i.e., a set $\mathcal{D}$ of objects of $\mathcal{C}$ such that for any object a of $\mathcal{C}$, there is an object $a' \in \mathcal{D}$ and a morphism $f\colon a \to a'$.

We are interested in (small) ancestral and descendent sets, because they help guarantee the existence of certain large limits and colimits in **Set**.

Proposition 5.1.5 ([KK81])**.** *i) Let $J\colon \mathcal{X} \to$ **Set** be a diagram, where $\mathcal{X}$ has a small ancestral set $\mathcal{A}$. Then J has a limit in* **Set**.

*ii) Let $J\colon \mathcal{Y} \to$ **Set** be a diagram, where $\mathcal{Y}$ has a small descendent set $\mathcal{D}$. Then J has a colimit in* **Set**.

Proof. (i): The limit is given by the set

$$l = \left\{ (\xi_a) \in \prod_{a \in A} J(a) \;:\; \forall f\colon a \to x, g\colon b \to x,\, J(f)(\xi_a) = J(g)(\xi_b) \right\},$$

where f and g range over all morphisms from objects in $\mathcal{A}$ to general objects of $\mathcal{X}$.

(ii): The colimit is given by the set

$$c = \sum_{d \in \mathcal{D}} J(d)/\sim,$$

where $\sim$ is the equivalence relation on pairs (d, η_d) (for $d \in \mathcal{D}$ and $\eta_d \in J(d)$), generated by

$$(\exists y\colon \mathcal{Y}, f\colon d \to y, g\colon e \to y \;.\; J(f)(\eta_d) = J(g)(\eta_e)) \Rightarrow (d, \eta_d) \sim (e, \eta_e)\,.$$

The proofs that these are indeed a limit and a colimit for the corresponding diagrams are very similar to the proofs for ordinary small limits and colimits in **Set**. However, the constructions we have made above rely on more than

[1] This is not a typo – *descendent* is the adjectival form of the noun *descendant*.

the fact that **Set** is complete and cocomplete. We also need to be able to perform universal and existential quantification over large collections of objects; this corresponds to the category-theoretic fact that if we have any (possibly large) collection of monomorphisms

$$A_i \to B$$

in **Set** (for fixed B), then we can form their intersection (i.e., a category-theoretic pullback for the collection), and similarly that we can form the pushout of any (possibly large) collection of epimorphisms out of a set. See [KK81] for full details. □

Definition 5.1.6. Let $\mathcal{X}$ be a monoidal category. Given an object x of $\mathcal{X}$, a *tensor-descendent set for* x is an ancestral set for the category of elements[2] of the functor $\mathcal{X}(\_ \otimes \_, x)\colon \mathcal{X}^{\mathrm{op}} \times \mathcal{X}^{\mathrm{op}} \to \mathbf{Set}$; i.e., a set $\mathcal{D}_x$ of objects of $\mathcal{X}$ such that for any morphism

$$f\colon y \otimes z \to x$$

there exist $y', z' \in \mathcal{D}_x$ and morphisms $h\colon y \to y'$, $k\colon z \to z'$, $\tilde{f}\colon y' \otimes z' \to x$ such that the following diagram commutes.

$$\begin{array}{ccc} y \otimes z & \xrightarrow{h \otimes k} & y' \otimes z' \\ & \searrow^{f} & \downarrow{\scriptstyle \tilde{f}} \\ & & x \end{array}$$

We say that $\mathcal{X}$ is *well-tensored* if every object x of $\mathcal{X}$ has a small tensor-descendent set.

Definition 5.1.7 ([Day70])**.** Let $\mathcal{X}$ be a well-tensored monoidal category and let

$$F, G\colon \mathcal{X} \to \mathbf{Set}$$

be functors. Then the *Day convolution* of F and G is a functor

$$F \otimes_{\mathrm{Day}} G\colon \mathcal{X} \to \mathbf{Set}$$

given by the coend

$$(F \otimes_{\mathrm{Day}} G)(x) = \int^{y,z\colon \mathcal{X}} F(y) \times G(z) \times \mathcal{X}(y \otimes z, x)\,.$$

This makes $[\mathcal{X}, \mathbf{Set}]$ into a monoidal category (the monoidal unit is the functor $\mathcal{X}(I, \_)\colon \mathcal{X} \to \mathbf{Set}$).

[2] See the proof of Proposition 6.3.1.

Remark 5.1.8. The reason that we need well-tensoredness of $\mathcal{X}$ is to ensure that the coend above is a small set. Indeed, let

$$[y, z, a, b, f]$$

be an arbitrary element of the coend (for $y, z\colon \mathcal{X}$, $a \in F(y)$, $b \in G(z)$ and $f\colon y \otimes z \to x$). Given a small tensor-descendent set $\mathcal{D}_x$, we can find $y', z' \in \mathcal{D}_x$ and morphisms $h\colon y \to y'$, $k\colon z \to z'$ and $\tilde{f}\colon y' \otimes z' \to x$ as in Definition 5.1.6. Then we have

$$[y, z, a, b, f] = [y', z', F(h)(a), F(k)(b), \tilde{f}]\,.$$

It follows that the coend is a subset of the small set

$$\sum_{y', z' \in \mathcal{D}_x} F(y') \times G(z') \times \mathcal{X}(y' \otimes z', x)\,,$$

and is therefore itself small.

We could alternatively have deduced this from Proposition 5.1.5, using the fact that the coend above may be recast as a colimit indexed by the opposite of the category of elements of $\mathcal{X}(\_ \otimes \_, x)$.

Definition 5.1.9 ([Mel12])**.** Given a lax action of a well-tensored monoidal category $\mathcal{X}$ upon a category $\mathcal{C}$, the *Melliès category* of the action is an $[\mathcal{X}, \mathbf{Set}]$-enriched category $\mathrm{Mell}_{\mathcal{X}}\,\mathcal{C}$ whose objects are the objects of $\mathcal{C}$ and where the hom objects are given by

$$\mathrm{Mell}_{\mathcal{X}}\,\mathcal{C}(a, b)(x) = \mathcal{C}(a, x.b)\,.$$

The composition is given by the natural transformation

$$\begin{aligned}
&(\mathrm{Mell}_{\mathcal{X}}\,\mathcal{C}(a, b) \otimes_{\mathrm{Day}} \mathrm{Mell}_{\mathcal{X}}\,\mathcal{C}(b, c))(x) \\
&= \int^{y,z\colon \mathcal{X}} \mathcal{C}(a, y.b) \times \mathcal{C}(b, z.c) \times \mathcal{X}(y \otimes z, x) \\
&\to \int^{y,z\colon \mathcal{X}} \mathcal{C}(a, (y \otimes z).c) \times \mathcal{X}(y \otimes z, x) \\
&\cong \mathcal{C}(a, x.c) \\
&= \mathrm{Mell}_{\mathcal{X}}\,\mathcal{C}(a, c)(x)\,,
\end{aligned}$$

where the arrow on the third line is induced by the Melliès composition $\mathcal{C}(a, y.b) \times \mathcal{C}(b, z.c) \to \mathcal{C}(a, (y \otimes z).c)$.

The identity transformation

$$\mathcal{C}(I, x) \to \mathcal{C}(a, x.a)$$

is the one sending a morphism $f\colon I \to x$ to the composite

$$a \xrightarrow{e_a} I.a \xrightarrow{f.a} x.a\,.$$

5.2 The category $\mathcal{C}/\mathcal{X}$

We will mainly be concerned not with the Melliès category itself, but with a closely related ordinary category. The usual method for turning a $\mathcal{V}$-enriched category into an ordinary category is via base change along a monoidal functor $\mathcal{V} \to \mathbf{Set}$.

For example, any $\mathcal{V}$-enriched category has an *underlying ordinary category*, obtained via base change along the functor $\mathcal{V}(I, \_)$. If a and b are objects of the base category $\mathcal{C}$, then the set of morphisms in the underlying ordinary category of the Melliès category is the set of natural transformations

$$\mathcal{C}(I, x) \to \mathcal{C}(a, x.b)\,,$$

which by the Yoneda lemma is the same as the set $\mathcal{C}(a, I.b)$. In other words, this underlying ordinary category is precisely the Kleisli category for the monad on $\mathcal{C}$ given by the composite

$$1 \xrightarrow{I} \mathcal{X} \xrightarrow{\_\cdot\_} \mathrm{End}[\mathcal{C}]\,;$$

in other words, the monad on $\mathcal{C}$ given by $Ma = I.a$.

More generally, if $m \otimes m \to m$ is a monoid in $\mathcal{X}$, then we can construct a lax monoidal functor $[\mathcal{X}, \mathbf{Set}] \to \mathbf{Set}$ that sends F to $F(m)$. If we change base through this functor, then the morphisms $a \to b$ in the resulting ordinary category will be elements of $\mathcal{C}(a, m.b)$; i.e., morphisms $a \to m.b$ in $\mathcal{C}$, and the category turns out to be the Kleisli category corresponding to the monad on $\mathcal{C}$ given by $Ma = m.a$. So the Melliès category can be thought of as classifying all the Kleisli categories that arise from the action $\_\cdot\_$ through the monoids in $\mathcal{M}$.

For our purposes, however, we will want to use a single ordinary category that captures as much as possible of the structure of the Melliès category. Suppose that $\mathcal{X}$ has, in addition, a small descendent set. Then we have a functor

$$\begin{aligned} [\mathcal{X}, \mathbf{Set}] &\to \quad \mathbf{Set} \\ F \quad &\mapsto \varinjlim_{x\colon \mathcal{X}} F(x) \end{aligned}$$

given by the colimit in **Set**.

Moreover, this functor is lax monoidal, via the multiplicative

$$\begin{aligned}
&\left(\int^{y} F(y)\right) \times \left(\int^{z} G(z)\right) \\
&\to \int^{y,z} F(y) \times G(z) \\
&\to \int^{y,z} F(y) \times G(z) \times \mathcal{X}(y \otimes z, x) \\
&= \int^{x} (F \otimes_{\mathrm{Day}} G)(x)
\end{aligned}$$

(where the second morphism sends picks out the inhabitant $\mathrm{in}_{y\otimes z}(\mathrm{id}_{y\otimes z})$ of $\varinjlim_{x\colon \mathcal{X}} \mathcal{X}(y \otimes z, x)$), and via the unital

$$\begin{aligned}
1 &\to \varinjlim_{x\colon \mathcal{X}} \mathcal{X}(I, x) \\
() &\mapsto \quad \mathrm{in}_I(\mathrm{id}_I)\,.
\end{aligned}$$

We then conclude by changing base through this colimit functor. It turns out to be fairly easy to describe the resulting category directly, and we can drop the well-tensoredness condition.

Definition 5.2.1. Let a monoidal category $\mathcal{X}$ with a small descendent set act on a category $\mathcal{C}$ via a lax action. Then we define a new category $\mathcal{C}/\mathcal{X}$ where

- the objects are the objects of $\mathcal{C}$;
- given objects a, b of $\mathcal{C}$, the set of morphisms $a \to b$ is given by the colimit
$$\varinjlim_{x\colon \mathcal{X}} \mathcal{C}(a, x.b)\,;$$
i.e., a morphism in $\mathcal{C}/\mathcal{X}$ from a to b is an equivalence class of Melliès morphisms
$$a \to x.b$$
in $\mathcal{C}$, where x ranges over the objects of $\mathcal{X}$, and where the equivalence relation on morphisms is generated by identifying two morphisms $f\colon a \to x.b$, $g\colon a \to y.b$ if there is a morphism $h\colon x \to y$ in $\mathcal{X}$ such that the following diagram commutes (we say that h *mediates* between f and g);
$$\begin{array}{ccc} a & \xrightarrow{f} & x.b \\ & \searrow^{g} & \downarrow{\scriptstyle h.b} \\ & & y.b \end{array}$$

- composition of morphisms is via the Melliès composition of Definition 5.1.3; and
- the identity $a \to a$ is given by the equivalence class corresponding to the morphism
$$e_a \colon a \to I.a$$
in $\mathcal{C}$.

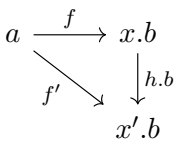
Remark 5.2.2. If we follow Proposition 5.1.5 exactly, then the object x in the Melliès morphism $a \to x.b$ has to be contained in the small descendent set of $\mathcal{X}$. Instead, we shall stick to the traditional formulation of the colimit, as for small diagrams, only using Proposition 5.1.5 so that we can be sure that the colimit is a small set.

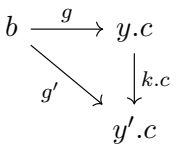
Proposition 5.2.3. *$\mathcal{C}/\mathcal{X}$ is a well defined category.*

Proof. Let us first check that Melliès composition is well defined with respect to our equivalence relation. Suppose that

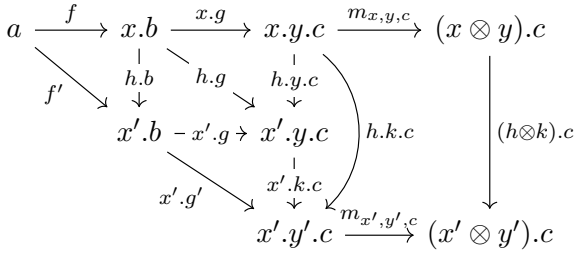
$$f \colon a \to x.b$$
$$f' \colon a \to x'.b$$
$$g \colon b \to y.c$$
$$g' \colon b \to y'.c$$

are Melliès morphisms, and that f is equivalent to f' and g is equivalent to g'. By induction, we can assume that f, f' and g, g' are related by the relation that generates the equivalence relation on Melliès morphisms, so that there are morphisms $h \colon x \to x'$, $k \colon y \to y'$ in $\mathcal{X}$ such that the following diagrams commute.



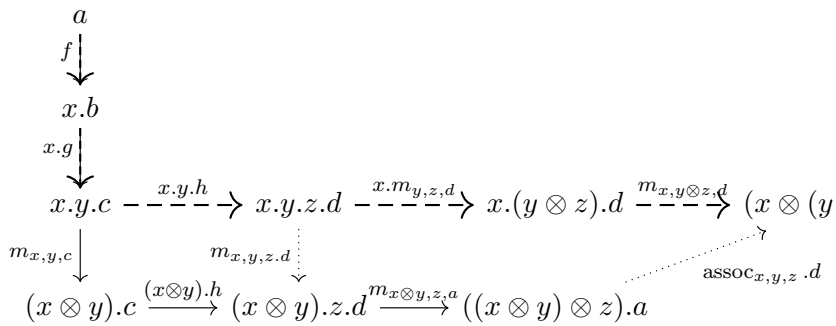


Then we have the following commutative diagram, since m is a natural transformation.



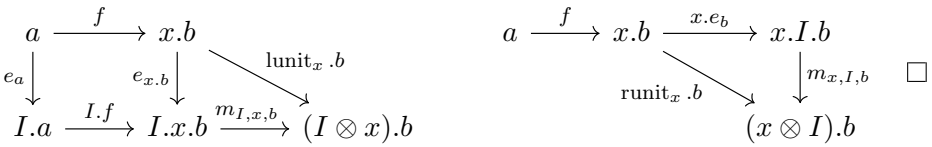
This proves that the Melliès composition of f and g is equivalent to the Melliès composition of f' and g'.

We should check that Melliès composition is associative with respect to our equivalence relation. Let $f\colon a \to x.b$, $g\colon b \to y.c$, $h\colon c \to z.d$ be Melliès morphisms. Then we have the following commutative diagram.



Here, the pentagon at the right is one of the coherence diagrams for a lax action, while the left-hand square commutes because m is a natural transformation. The composite given by the thick dashed lines is the Melliès composition $f;(g;h)$, while that given by thin lines is the Melliès composition $(f;g);h$. The arrow $\mathrm{assoc}_{x,y,z}$ in $\mathcal{X}$ then mediates between these morphisms, so they are equivalent, and therefore correspond to the same morphism in $\mathcal{C}/\mathcal{X}$.

Lastly, we need to check that the identity we have defined is indeed an identity for the composition. The following diagrams show us that the morphism lunit_x in $\mathcal{X}$ mediates between a Melliès morphism $f\colon a \to x.b$ and the Melliès composite $e_a; f$, and that the morphism runit_x in $\mathcal{X}$ mediates between f and the Melliès composite $f; e_b$.



□

Remark 5.2.4. This generalizes the Kleisli category of a monad. Indeed, if M is a monad on $\mathcal{C}$, then we may consider M as a lax action of the unit category 1 on $\mathcal{C}$. Then composition of morphisms $f\colon a \to I.b$, $g\colon b \to I.c$ in the category $\mathcal{C}/1$ is given by

$$a \xrightarrow{f} I.b \xrightarrow{I.g} I.I.c \xrightarrow{m_{I,I,c}} I.c\,.$$

Writing M for $I.\_$, we see that this is precisely the definition of composition in the Kleisli category for M (and the identity is the same thing too). Note

that since the unit category has only an identity morphism, the equivalence relation on morphisms in $\mathcal{C}/1$ is discrete, so $\mathcal{C}/1$ *is* the Kleisli category for M.

There is an identity-on-objects functor $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{X}$ given by sending a morphism $f\colon a \to b$ to the Melliès morphism

$$a \xrightarrow{f} b \xrightarrow{e_b} I.b\,.$$

It is easy to show that this is a functor, and we will prove that it is in a more general context in Proposition 5.5.4.

A special role will be played by the identity morphisms

$$\mathrm{id}_{x.a}\colon x.a \to x.a\,,$$

considered as Melliès morphisms $x.a \to a$. We will see later that these are the components of a natural transformation $\phi_{x,a}\colon x.a \to a$ in $\mathcal{C}/\mathcal{X}$, generalizing the natural transformation $\phi_a\colon Ma \to a$ in the Kleisli category for a monad M that we met in Proposition 4.3.1.

5.3 Lax 2-colimits

In the previous chapter, we approached the category $\mathcal{C}/\mathcal{X}$ via the Melliès category. In this chapter, we will show that $\mathcal{C}/\mathcal{X}$ has important properties in its own right: namely that it is a certain lax 2-colimit in **Cat**. We will briefly return to the Melliès category in chapter 7, when we will approach it from the point of view of profunctors.

Definition 5.3.1 ([Str72b])**.** Let $\mathcal{C}, \mathcal{D}$ be bicategories. A *lax functor* $F\colon \mathcal{C} \to \mathcal{D}$ is given by

- a map F from the objects of $\mathcal{C}$ to the objects of $\mathcal{D}$;
- for each pair a, b of objects of $\mathcal{C}$, a functor

  $$F\colon \mathcal{C}(a,b) \to \mathcal{D}(F(a), F(b))\,;$$

- for each triple a, b, c of objects of $\mathcal{C}$, a transformation

  $$m_{f,g}\colon F(f); F(g) \to F(f;g)$$

  natural in $f\colon a \to b$, $g\colon b \to c$; and
- for each object a of $\mathcal{C}$, a 2-cell

  $$e_a\colon \mathrm{id}_{Fa} \Rightarrow F(\mathrm{id}_a)\colon Fa \to Fa\,;$$

such that for all tuples a, b, c, d of objects and all morphisms $f\colon a \to b$, $g\colon b \to c$, $h\colon c \to d$, the following diagrams commute.

$$\begin{array}{ccc}
(F(f);F(g));F(h) & \xrightarrow{\mathrm{assoc}_{F(f),F(g),F(h)}} & F(f);(F(g);F(h)) \\
{\scriptstyle m_{f,g};F(h)}\downarrow & & \downarrow{\scriptstyle F(f);m_{g,h}} \\
F(f;g);F(h) & & F(f);F(g;h) \\
{\scriptstyle m_{f;g,h}}\downarrow & & \downarrow{\scriptstyle m_{f,g;h}} \\
F((f;g);h) & \xrightarrow{F(\mathrm{assoc}_{f,g,h})} & F(f;(g;h))
\end{array}$$

$$\begin{array}{ccc}
Ff & \xrightarrow{\mathrm{lunit}_{Ff}} & \mathrm{id}_{Fa};Ff \\
{\scriptstyle F\,\mathrm{lunit}_f}\downarrow & & \downarrow{\scriptstyle e_a;Ff} \\
F(\mathrm{id}_a;f) & \xleftarrow{m_{\mathrm{id}a,f}} & F\,\mathrm{id}_a;Ff
\end{array}
\qquad
\begin{array}{ccc}
Ff & \xrightarrow{\mathrm{runit}_{Ff}} & Ff;\mathrm{id}_{Fb} \\
{\scriptstyle F\,\mathrm{runit}_f}\downarrow & & \downarrow{\scriptstyle Ff;e_b} \\
F(f;\mathrm{id}_b) & \xleftarrow{m_{f,\mathrm{id}_b}} & Ff;F\,\mathrm{id}_b
\end{array}$$

Example 5.3.2. Given a monoidal category $\mathcal{X}$, write $\mathbf{B}\mathcal{X}$ for the bicategory having a single object $*$, where $\mathbf{B}\mathcal{X}(*,*)$ is the category $\mathcal{X}$ and composition of 1-cells $x, y\colon * \to *$ is given by

$$x;y = x \otimes y\,.$$

If $\mathcal{C} = \mathbf{B}\mathcal{X}$, $\mathcal{D} = \mathbf{B}\mathcal{Y}$ are the delooping bicategories of monoidal categories $\mathcal{X}, \mathcal{Y}$, then a lax functor $\mathcal{C} \to \mathcal{D}$ is the same thing as a lax monoidal functor $\mathcal{X} \to \mathcal{Y}$.

Example 5.3.3. More generally, a lax functor $F\colon \mathbf{B}\mathcal{X} \to \mathcal{D}$ is the same as a lax monoidal functor from $\mathcal{X}$ to the monoidal category of 1-cells $F(*) \to F(*)$. In particular, a lax functor $1 \to \mathbf{Cat}$ is the same thing as a monad and a lax functor $\mathbf{B}\mathcal{X} \to \mathbf{Cat}$ is the same thing as a parametric monad parameterized by $\mathcal{X}$.

Definition 5.3.4. Let $F\colon \mathcal{C} \to \mathcal{D}$ be a lax functor of bicategories. Then an *oplax cocone under* F is given by an object d of $\mathcal{D}$ (called the *tip* of the cocone), together with 1-cells

$$l_c\colon F(c) \to d$$

for each object c of $\mathcal{C}$ and 2-cells

$$\mu_h\colon F(h);l_{c'} \Rightarrow l_c\colon F(c) \to d$$

for each 1-cell $h\colon c \to c'$ in $\mathcal{C}$, such that for all 2-cells $\phi\colon h' \Rightarrow h\colon c \to c'$ the diagram

$$\begin{array}{ccc}
F(h');l_{c'} & \xRightarrow{\mu_{h'}} & l_c \\
{\scriptstyle F(\phi);l_{c'}}\Downarrow & \nearrow_{\mu_h} & \\
F(h);l_{c'} & &
\end{array}$$

commutes, and such that for all $h\colon c \to c'$, $h'\colon c' \to c''$, the diagrams

$$\begin{array}{ccccc}
(F(h);F(h'));l_{c''} & \xRightarrow{\mathrm{assoc}_{F(h),F(h'),l_{c''}}} & F(h);(F(h');l_{c''}) & \xRightarrow{F(h);\mu_{h'}} & F(h);l_{c'} \\
{\scriptstyle m_{h,h'};l_{c''}}\Downarrow & & & & \Downarrow{\scriptstyle \mu_h} \\
F(h;h');l_{c''} & & \xRightarrow{\qquad\qquad\mu_{h;h'}\qquad\qquad} & & l_c
\end{array}$$

$$\begin{array}{ccc}
l_c & \xRightarrow{\mathrm{lunit}_{l_c}} & \mathrm{id}_c;l_c \\
{\scriptstyle \mathrm{id}_{l_c}}\Downarrow & & \Downarrow{\scriptstyle e_c;l_c} \\
l_c & \xLeftarrow{\mu_{\mathrm{id}}} & F(\mathrm{id}_c);l_c
\end{array}$$

commute.

Definition 5.3.5 ([Lei98])**.** Let (d, l, μ), (d', l', μ') be oplax cocones for a lax functor $F\colon \mathcal{C} \to \mathcal{D}$. Then a *modification of oplax cocones* $l \to l'$ is given by a collection of 2-cells

$$\delta_c\colon l_c \Rightarrow l'_c$$

for each object c of $\mathcal{C}$, such that the following diagram commutes for any 1-cell $h\colon c \to c'$ in $\mathcal{C}$.

$$\begin{array}{ccc}
F(h);l_{c'} & \xRightarrow{\mu_h} & l_c \\
{\scriptstyle F(h);\delta_{c'}}\Downarrow & & \Downarrow{\scriptstyle \delta_c} \\
F(h);l'_{c'} & \xRightarrow{\mu'_h} & l'_c
\end{array}$$

Definition 5.3.6 ([GHN17])**.** Let $F\colon \mathcal{C} \to \mathcal{D}$ be a lax functor of bicategories. Then a *lax colimit* of F is an oplax cocone (u, k, ν) under F such that

- if (d, l, μ) is any other oplax cocone under F then there is a unique 1-cell $\hat{l}\colon u \to d$ such that $k_c;\hat{l} = l_c$ for each object c of $\mathcal{C}$ and such that $\nu_h\hat{l} = \mu_h$ for any 1-cell $h\colon c \to c'$ in $\mathcal{C}$; and
- if (d, l', μ') is another oplax cocone under F with tip d and $\delta\colon l \Rightarrow l'$ is a modification of oplax cocones, then there is a unique 2-cell $\hat{\delta}\colon \hat{l} \Rightarrow \hat{l'}$ such that $h_c;\hat{\delta} = \delta_c$ for all objects c of $\mathcal{C}$.

For the justification of the terminology we have used whereby a *lax* limit is a limiting *oplax* cocone, see [nLa19].

Remark 5.3.7. Our definition of a lax colimit (u, k, ν) tells us that if d is an object of $\mathcal{C}$, then there is an isomorphism of categories from the category $\mathcal{C}(u, d)$ to the category of oplax cocones under F with tip d, with morphisms given by modifications of oplax cocones. As a consequence, any two lax colimits for the same diagram are uniquely isomorphic as cocones.

There is a more usual definition of an oplax cocone, in which this isomorphism of categories is weakened to an equivalence. The distinction is not important for our purposes, since the lax colimits we will be considering will all satisfy our stronger definition.

We shall sometimes write

$$\varinjlim_{c\colon \mathcal{C}}{}_{\mathrm{l}}\, F(c)$$

for the lax colimit of the functor F.

5.4 Lax natural transformations and functoriality of lax colimits

Definition 5.4.1 ([Lei98])**.** Let $F, G\colon \mathcal{C} \to \mathcal{D}$ be lax functors of bicategories. A *oplax natural transformation* $\mathcal{C} \to \mathcal{D}$ is given by a family of 1-cells

$$t_c\colon F(c) \to G(c)\,,$$

for each object c of $\mathcal{C}$, together with a family of 2-cells

$$\mu_h\colon F(h); t_{c'} \Rightarrow t_c; G(h)\colon F(c) \to G(c')$$

for each morphism $h\colon c \to c'$ in $\mathcal{C}$, such that for all 2-cells $\phi\colon h' \to h\colon c \to c'$ the diagram

$$\begin{array}{ccc} F(h'); t_{c'} & \xRightarrow{\mu_{h'}} & t_c; G(h') \\ {\scriptstyle F(\phi);t_{c'}}\Downarrow & & \Downarrow{\scriptstyle T_c;F(\phi)} \\ F(h); t_{c'} & \xRightarrow{\mu_h} & t_c; G(h) \end{array}$$

commutes, and such that for all $h\colon c \to c'$, $h'\colon c' \to c''$, the diagrams

$$\begin{array}{ccccc} (F(h); F(h')); t_{c''} & \xRightarrow{\text{assoc}} & F(h); (F(h'); t_{c''}) & \xRightarrow{F(h);\mu_{h'}} & F(h); (t_{c'}; G(h')) \\ {\scriptstyle m^F_{h,h'};t_{c''}}\Downarrow & & & & \Downarrow{\scriptstyle \text{assoc}} \\ F(h; h'); t_{c''} & & & & (F(h); t_{c'}); G(h') \\ {\scriptstyle \mu_{h;h'}}\Downarrow & & & & \Downarrow{\scriptstyle \mu_h;G(h')} \\ t_c; G(h; h') & \xRightarrow[t_c;m^G_{h,h'}]{} & t_c; (G(h); G(h')) & \xRightarrow[\text{assoc}]{} & (t_c; G(h)); G(h') \end{array}$$

$$\begin{array}{ccc} l_c & \xRightarrow{\text{lunit}_{l_c}} & \mathrm{id}_c; l_c \\ {\scriptstyle \text{runit}_{l_c}}\Downarrow & & \Vert{\scriptstyle e^F_c;l_c} \\ l_c; \mathrm{id}_c & & \Vert \\ {\scriptstyle l_c;e^G_c}\Downarrow & & \Vert \\ l_c; G(\mathrm{id}_c) & \xRightarrow[\mu_{\mathrm{id}_c}]{} & F(\mathrm{id}_c); l_c \end{array}$$

commute.

As we would expect, the horizontal/vertical composition of two oplax natural transformations is also an oplax natural transformation, and there is an identity oplax natural transformation $F \Rightarrow F$ for any lax functor F of bicategories.

Example 5.4.2. An oplax natural transformation between two lax functors $\mathbf{B}\mathcal{X} \to \mathbf{Cat}$ is the same thing as an oplax morphism between the corresponding actions, as defined in Definition 5.0.2.

Example 5.4.3. If $F\colon \mathcal{C} \to \mathcal{D}$ is a lax functor of bicategories, then an oplax cocone under F with tip $d\colon \mathcal{D}$ is the same thing as an oplax natural transformation $F \to *_d$, where $*_d$ is the constant functor taking the value d.

We can then get a functoriality result similar to the usual one for ordinary colimits.

Proposition 5.4.4. *Let $F, G\colon \mathcal{C} \Rightarrow \mathcal{D}$ be lax functors and let $t\colon F \Rightarrow G$ be an oplax natural transformation. Suppose that F and G have lax colimits with tips $\tilde{F}$ and $\tilde{G}$. Then t naturally gives rise to a lax functor $\tilde{F} \to \tilde{G}$ in a way that respects composition and identity of lax natural transformations.*

Proof. Composing the limiting oplax cocone $k^G\colon G \to *_{\tilde{G}}$ with t gives us an oplax natural transformation $F \to *_{\tilde{G}}$ and hence a unique lax functor $(t; k^G)^\curlywedge\colon \tilde{F} \to \tilde{G}$ such that $k^F_c; (t; k^G)^\curlywedge = t_c; k^G_c$ and $\nu^F_h (t; k^G)^\curlywedge = \nu^G_h$ for all objects c and morphisms h in $\mathcal{C}$.

We can prove that this procedure preserves composition and identities via the usual technique: if we have oplax natural transformations $t\colon F \Rightarrow G$ and $t'\colon G \Rightarrow H$, then the composite of the induced lax functors $\tilde{F} \to \tilde{G}$ and $\tilde{G} \to \tilde{H}$ satisfies the same property that uniquely defines the lax functor $\tilde{F} \to \tilde{H}$ induced from the composite of t and t'. □

Another similar functoriality result tells us what happens to lax colimits when we precompose with a lax functor.

Proposition 5.4.5. *Let $F\colon \mathcal{C} \to \mathcal{D}$, $G\colon \mathcal{D} \to \mathcal{E}$ be lax functors. Suppose that lax colimits exist for G and for $F; G$. Then we get a natural lax functor $\hat{F}$ from the lax colimit of $F; G$ to the lax colimit of G such that for all objects c and all morphisms h in $\mathcal{C}$ we have*

$$k^{F;G}_c; \hat{F} = k^g_{Fc} \qquad\qquad \nu^{F;G}_h \hat{F} = \nu^G_{Fh},$$

where $(k^{F;G}, \nu^{F;G})$ is the limiting cocone for the lax colimit of $F; G$ and (k^G, ν^G) is the limiting cocone for the lax colimit of G.

Proof. Composing the limiting cocone (k^G_d, ν^G_k) for the lax colimit of G with F gives us an oplax cocone (k^G_{Fc}, ν^G_{Fh}) under $F;G$, inducing a functor from the lax colimit of $F;G$ to the lax colimit of G that satisfies the required properties. □

5.5 Lax 2-colimits in Cat

Let $\mathcal{C}$ be a bicategory, and let $F\colon \mathcal{C} \to \mathbf{Cat}$ be a bifunctor.

Definition 5.5.1 ([Str80])**.** The *Grothendieck construction* associates to the bifunctor F a bicategory $\int F$, where

- the objects of $\int F$ are pairs (a, m), where a is an object of $\mathcal{C}$ and m an object of $F(a)$;
- the 1-cells $(a, m) \to (b, n)$ are pairs (h, f), where h is a 1-cell $b \to a$ and f is a morphism $m \to F(h)(n)$ in $F(a)$; and
- the 2-cells $(h, f) \Rightarrow (k, g)\colon (a, m) \to (b, n)$ are 2-cells $\phi\colon h \Rightarrow k$ in $\mathcal{C}$ that make the following diagram commute.

$$\begin{array}{ccc} m & \xrightarrow{f} & F(h)(n) \\ & \searrow^{g} & \downarrow{\scriptstyle (F\phi)_n} \\ & & F(k)(n) \end{array}$$

The identity 1-cell $(a, m) \to (a, m)$ is given by $\left(\mathrm{id}_a, m \xrightarrow{(e_a)_m} F(\mathrm{id})(m)\right)$. The composite of a 1-cell $(h, f)\colon (a, m) \to (b, n)$ with a 1-cell $(k, g)\colon (b, n) \to (c, p)$ is the pair $(k; h, f * g)$, where $f * g$ is the following composite.

$$m \xrightarrow{f} F(h)(n) \xrightarrow{F(h)(g)} F(h)(F(k)(n)) \xrightarrow{m_{h,k}} F(k;h)(n)$$

Remark 5.5.2. For the Grothendieck construction see [SGA1, VI.8] and [Joh02, B1.3.1] – in the original formulation, the category F is a pseudofunctor $\mathcal{C} \to \mathbf{Cat}$ (i.e., a lax functor where the coherences m and e are isomorphisms), where $\mathcal{C}$ is an ordinary category – then $\int F$ contains only identity 2-cells, so we can consider it as an ordinary category as well. For the version where $\mathcal{C}$ is an arbitrary bicategory (and $\int F$ is a bicategory), see [Str80].

There are many different variations of this construction, where various combinations of arrows are reversed. We have chosen the variation that suits our needs.

Definition 5.5.3. Let $\mathcal{C}$ be a bicategory, and suppose that whenever a, b are objects of $\mathcal{C}$, the category $\mathcal{C}(a,b)$ of 1-cells has a small descendent set. Then we write $\pi_*\mathcal{C}$ for the ordinary category whose objects are the objects of $\mathcal{C}$ and where the morphisms $a \to b$ are the connected components of the 1-cells $a \to b$ in $\mathcal{C}$: i.e., equivalence classes of 1-cells $a \to b$ under the equivalence relation generated by relating $f\colon a \to b$ to $g\colon a \to b$ if there is a 2-cell $\phi\colon f \to g$.

Proposition 5.5.4 ([nLa19, §3])**.** *Let $\mathcal{C}$ be a bicategory such that $\mathcal{C}(a,b)$ has a small descendent set for every pair (a,b) of objects of $\mathcal{C}$, and let $F\colon \mathcal{C} \to$ **Cat** be a lax functor. Then $\pi_* \int F$ is a lax colimit for F.*

Proof. First, we need to show that $\pi_* \int F$ is well defined; i.e., that each category $\int F((a,m),(b,n))$ of morphisms in $\int F$ has a small descendent set. Suppose that $\mathcal{D}$ is a small descendent set for $\mathcal{C}(b,a)$. We claim that

$$\{(h', f') \,:\, h' \in \mathcal{D}\,, f\colon m \to F(h)(n)\}$$

is a descendent set for $\int F((a,m),(b,n))$.

Indeed, let $(h,f)\colon (a,m) \to (b,n)$ be a morphism in $\int F$ – so $h\colon b \to a$ is a 1-cell in $\mathcal{C}$ and f is a morphism $m \to F(h)(n)$ in $F(a)$. Let $h' \in \mathcal{D}$ be such that there exists a 2-cell $\phi\colon h \Rightarrow h'$ and let $f'\colon m \to F(h')(n)$ be given by the composite

$$m \xrightarrow{f} F(h)(n) \xrightarrow{(F\phi)_n} F(h')(n)\,.$$

Then ϕ is a 2-cell from (h,f) to (h',f') in $\int F((a,m),(b,n))$.

Now we have assured ourselves that there are no size problems, we define the oplax cocone under F with tip $\pi_* \int F$. Given an object a of $\mathcal{C}$, the arrow from $F(a)$ to $\pi_* \int F$ is the functor k defined by

$$k_a(m) = (a,m) \qquad k_a\left(m \xrightarrow{f} n\right) = \left(\mathrm{id}_a, m \xrightarrow{f} n \xrightarrow{(e_a)_n} F(\mathrm{id})(n)\right)\,.$$

We should check that this is indeed a functor; clearly it preserves the identity, so we need to check that it preserves composition. Let $m \xrightarrow{f} n \xrightarrow{g} p$ be morphisms in $F(a)$. We need to show that $k_a(f); k_a(g) = k_a(f;g)$ in $\pi_* \int F$, for which it suffices to exhibit a 2-cell in $\int F$ mediating between these two 1-cells. Indeed, such a 2-cell is given by $\mathrm{lunit}_{\mathrm{id}_a}\colon \mathrm{id}_a \to \mathrm{id}_a; \mathrm{id}_a$ (see Figure 5.1).

Next, if $h\colon a \to a'$ is a morphism in $\mathcal{C}$, then we define a natural transformation

$$\nu_h\colon F(h); k_{a'} \Rightarrow k_a$$

by

$$(\nu_h)_m = (h, \mathrm{id}_{F(h)(m)})\colon (a', F(h)(m)) \to (a,m)\,.$$

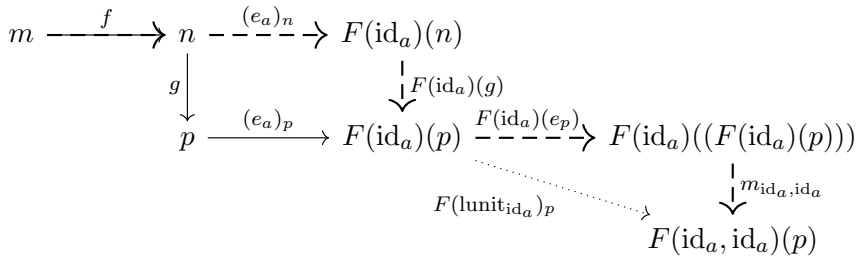


Figure 5.1: The square commutes because (e_a) is a natural transformation, while commutativity of the triangle is one of the conditions for a lax functor. The composite $k_a(f); k_a(g)$ is given by the pair $\Big(\mathrm{id}_a; \mathrm{id}_a,\ m \dashrightarrow F(\mathrm{id}_a, \mathrm{id}_a)(p)\ \Big)$ (thick dashed arrows), while $k_a(f; g)$ is given by $(\mathrm{id}_a, m \to F(\mathrm{id}_a)(p))$ (normal arrows). The 2-cell $\mathrm{lunit}_{\mathrm{id}_a} \colon \mathrm{id}_a \to \mathrm{id}_a$ from $\int F$ mediates between them (dotted arrow).

We want to show that this is a natural transformation; i.e., that for any morphism $f \colon m \to n$ in $F(a)$, the following diagram commutes.

$$\begin{array}{ccc} k_{a'}(F(h)(m)) & \xrightarrow{(\nu_h)_m} & k_a(m) \\ {\scriptstyle k_{a'}(F(h)(f))}\downarrow & & \downarrow{\scriptstyle k_a(f)} \\ k_{a'}(F(h)(n)) & \xrightarrow{(\nu_h)_n} & k_a(n) \end{array}$$

Here, the top right composite is given by

$$F(h)(m) \xrightarrow{F(h)(f)} F(h)(n) \xrightarrow{F(h)((e_a)_n)} F(h)(F(\mathrm{id}_a)(n)) \xrightarrow{m_{h,\mathrm{id}_a}} F(h; \mathrm{id}_a)(n)\,,$$

which is related to $F(h)(f)$ via runit_h, while the bottom left composite is given by

$$F(h)(m) \xrightarrow{(Fh)f} F(h)(n) \xrightarrow{(e_{a'})_{(Fh)f}} F(\mathrm{id}_{a'})(F(h)(n)) \xrightarrow{m_{\mathrm{id}_{a'},h}} F(\mathrm{id}_{a'}; F(h))(n)\,,$$

which is related to $F(h)(f)$ via lunit_h.

We now need to show that the appropriate diagrams commute to ensure that $\left(\pi_* \int F, k, \nu\right)$ is a oplax cocone under F. For the first diagram, we must show that the following commutes for any 2-cell $\phi \colon h' \Rightarrow h \colon a \to a'$ and any object m of $F(a)$.

$$\begin{array}{ccc} k_{a'}(F(h')(m)) & \xrightarrow{(\nu_{h'})_m} & k_a(m) \\ {\scriptstyle k_{a'}((F\phi)_m)}\downarrow & \nearrow_{(\nu_h)_m} & \\ k_{a'}(F(h)(m)) & & \end{array}$$

Since the arrow along the top is given by the identity on $F(h')(m)$, it will suffice to show that the arrow along the bottom is of the form $F(\psi)_m$, for some 2-cell ψ in $\mathcal{C}$, and that it is therefore equal to the identity in $\pi_* \int F$. Indeed, writing this composite out in full, we get

$$F(h')(m) \xrightarrow{(F\phi)_m} F(h)(m) \xrightarrow{(e_{a'})_{F(h)(m)}} F(\mathrm{id})(F(h)(n)) \xrightarrow{m_{\mathrm{id},h}} F(\mathrm{id}; h)(n)\,.$$

By the conditions on a lax functor, the composite of the last two morphisms is $F(\mathrm{lunit}_h)_m$, and so the whole thing is equal to $F(\phi; \mathrm{lunit}_h)_m$.

Lastly, we need to show that the following diagrams commute for any 1-cells $h\colon a \to a'$, $h'\colon a' \to a''$ and any object m of $F(a)$.

$$\begin{array}{ccc} k_{a''}(F(h')(F(h)(m))) & \xrightarrow{(\nu_h)_{F(h)(m)}} & k_{a'}(F(h)(m)) \\ {\scriptstyle k_{a''}((m_{h,h'})_m)}\downarrow & & \downarrow{\scriptstyle (\nu_h)_m} \\ k_{a''}(F(h; h')(m)) & \xrightarrow{(\nu_{h;h'})_m} & k_a(m) \end{array}$$

$$\begin{array}{ccc} k_a(m) & \xrightarrow{k_a((e_a)_m)} & k_a(F(\mathrm{id})(m)) \\ & {\scriptstyle \mathrm{id}_{k_a(m)}} \searrow & \downarrow{\scriptstyle (\nu_{\mathrm{id}})_m} \\ & & k_a(m) \end{array}$$

This time, we can compute that both composites in the first diagram are equal to

$$F(h')(F(h)(m)) \xrightarrow{m_{h,h'}} F(h; h')(m)\,,$$

while in the second diagram the top left composite is equal to

$$m \xrightarrow{(e_a)_m} F(\mathrm{id})(m) \xrightarrow{(e_a)_{F(\mathrm{id})(m)}} F(\mathrm{id})(F(\mathrm{id})(m)) \xrightarrow{\mu_{\mathrm{id},\mathrm{id}}} F(\mathrm{id}; \mathrm{id})(m)\,,$$

and the diagonal arrow is given by

$$m \xrightarrow{(\mathfrak{e}_a)_m} F(\mathrm{id})(m)\,.$$

These 1-cells in $\int F$ are related by $F(\mathrm{lunit}_m)$, so they correspond to the same morphism in $\pi_* \int F$.

This completes the definition of the universal cocone under $\pi_* \int F$.

Now, suppose that $(\mathcal{D}, l, \mu)$ is another oplax cocone under F. We define

$$\hat{l}(a, m) = l_a(m)$$

$$\hat{l}\left((a, m) \xrightarrow{(h,f)} (b, n)\right) = l_a(m) \xrightarrow{l_a(f)} l_a(F(h)(n)) \xrightarrow{(\mu_h)_n} l_b(n)\,.$$

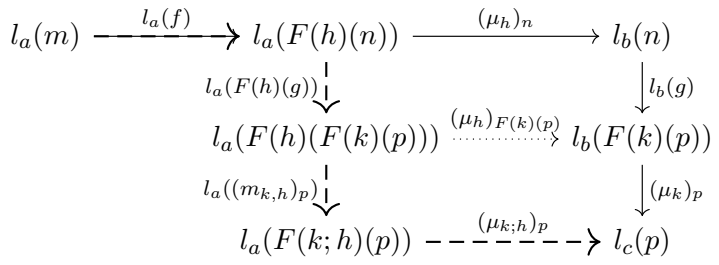


Figure 5.2: Proof that $\hat{l}$ respects composition. The thick dotted line represents $\hat{l}((h,f);(k,g))$, while the thin solid line represents $\hat{l}(h,f);\hat{l}(k,g)$. The top square commutes because μ_h is a natural transformation, while the bottom is one of the conditions on a cocone.

We need to show that $\hat{l}$ is a functor $\pi_* \int F \to \mathcal{D}$. First, observe that it sends the identity on (a,m) to the composite

$$l_a(m) \xrightarrow{l_a((e_a)_m)} l_a(F(\mathrm{id})(m)) \xrightarrow{(\mu_{\mathrm{id}})_m} l_a(m)\,,$$

which is equal to the identity on $l_a(m)$ since l is a cocone under F. Now suppose we have morphisms

$$(a,m) \xrightarrow{(h,f)} (b,n) \xrightarrow{(k,g)} (c,p)\,.$$

Using the formula for the composition of morphisms in the Grothendieck construction, we see that $\hat{l}((h,f);(k;g))$ is given by the thick dotted composite in Figure 5.2, while $\hat{l}(h,f);\hat{l}(k,g)$ is given by the thin composite. We can deal with the identity in a similar way. Therefore, $\hat{l}$ is a functor.

Moreover, if a is an object of $\mathcal{C}$ and m and object of $F(a)$, we have $\hat{l}(k_a(m)) = \hat{l}(a,m) = l_a(m)$, and if $f\colon m \to n$ is a morphism in $F(a)$, then $\hat{l}(k_a(f))$ is given by the composite

$$l_a(m) \xrightarrow{l_a(f)} l_a(n) \xrightarrow{l_a((e_a)_n)} l_a(F(\mathrm{id})(n)) \xrightarrow{(\mu_{\mathrm{id}})_n} l_a(n)\,,$$

which is equal to $l_a(f)$ by the cocone condition on μ. Therefore, $k_a;\hat{l} = l_a$ for each object a of $\mathcal{C}$.

Lastly, if $h\colon a \to b$ is a morphism in $\mathcal{C}$, then

$$\hat{l}((\nu_h)_m) = \hat{l}(h, \mathrm{id}_{F(h)(m)}) = (\mu_h)_m\,,$$

and so $\nu_h l = \mu_h$. This completes the existence part of the proof.

For uniqueness, suppose that j is another functor $\pi_* \int F \to \mathcal{D}$ such that $k_a;j = l_a$ and $\nu_h j = \mu_h$ for each object a of $\mathcal{C}$ and each morphism $h\colon a \to b$ in $\mathcal{C}$.

Let (a, m) be an object of $\pi_* \int F$. Then $j(a, m) = j(k_a(m)) = l_a(m) = \hat{l}(a, m)$.

Now let $(h, f) \colon (a, m) \to (b, n)$ be a morphism in $\pi_* \int F$. We claim that we may decompose (h, f) as the composite

$$\begin{array}{ccc} (a,m) & & \\ {\scriptstyle k_a(f)}\downarrow & \searrow^{(h,f)} & \\ (a, F(h)(n)) & \xrightarrow[(\nu_h)_n]{} & (b,n) \end{array} .$$

Indeed, if we work out what that composite is, then we get

$$m \xrightarrow{f} F(h)(n) \xrightarrow{(e_a)_n} F(\mathrm{id})(F(h)(n)) \xrightarrow{(m_{\mathrm{id},h})_n} F(\mathrm{id}; h)(n)\,,$$

which is related to (h, f) in $\int F$ by the 2-cell lunit_h (since F is a lax functor), and is therefore equal to (h, f) in $\pi_* \int F$.

We therefore have

$$j(h, f) = j(k_a(f); (\nu_h)_n) = j(k_a(f)); j((\nu_h)_n) = l_a(f); (\mu_h)_n = \hat{l}(h, f)\,,$$

and therefore $j = \hat{l}$.

Lastly, suppose that (l', μ') is another oplax cocone with tip $\mathcal{D}$ under F, and let $\delta \colon l \Rightarrow l'$ be a modification of oplax cocones. Then, for each object a of $\mathcal{C}$ and each object m of $F(a)$, we have a morphism

$$\delta_{a,m} \colon l_a(m) \to l'_a(m)\,.$$

We claim that these $\delta_{a,m}$ form the components of a natural transformation $\hat{\delta} \colon \hat{l} \Rightarrow \hat{l'}$. Indeed, if $(h, f) \colon (a, m) \to (b, n)$ is a morphism in $\int F$, then we have the following commutative diagram.

$$\begin{array}{ccc} l_a(m) & \xrightarrow{\delta_{a,m}} & l'_a(m) \\ {\scriptstyle l_a(f)}\downarrow & & \downarrow{\scriptstyle l'_a(f)} \\ l_a(F(h)(n)) & \xrightarrow{\delta_{a,F(h)(n)}} & l'_a(F(h)(n)) \\ {\scriptstyle (\mu_h)_n}\downarrow & & \downarrow{\scriptstyle (\mu'_h)_n} \\ l_b(n) & \xrightarrow{\delta_{b,n}} & l'_b(n) \end{array}$$

Here, the top square commutes because δ_a is a natural transformation, while the bottom commutes because δ is a modification of oplax cocones.

Now we have

$$(k_a; \hat{\delta})_m = \hat{\delta}_{k_a(m)} = \delta_{a,m}\,,$$

as desired. □

5.6 Examples of lax 2-colimits in Cat

Suppose that M is a monad on a category $\mathcal{C}$, considered as a lax functor

$$F\colon 1 \to \mathbf{Cat}$$

such that $F(*) = \mathcal{C}$.

If we apply the Grothendieck construction to F, then the category we get has pairs $(*, a)$ for objects, where a ranges over the objects of $\mathcal{C}$; and the morphisms from $(*, a)$ to $(*, b)$ are morphisms $a \to F(\mathrm{id})(b)$ in $\mathcal{C}$, where $F(\mathrm{id}) = M$. In other words, $\int F$ is precisely the Kleisli category for M. In this case, Proposition 5.5.4 reduces to Proposition 4.3.1.

More generally, if $\mathcal{X}$ is a monoidal category with a small descendent set, then $\mathbf{B}\mathcal{X}$ satisfies our smallness condition (all categories of 1-cells have a small descendent set). If $F\colon \mathbf{B}\mathcal{X} \to \mathbf{Cat}$ is a lax functor, corresponding to a lax action of $\mathcal{X}$ upon $\mathcal{C} = F(*)$, then it is easy to see that $\pi_* \int F$ is isomorphic to the category $\mathcal{C}/\mathcal{X}$ that we defined earlier. Indeed, in both cases the objects may be identified with the objects of $\mathcal{C}$, and the morphisms from an object a to an object b may be written as equivalence classes of morphisms $a \to x.b$ in $\mathcal{C}$ under the equivalence relation generated by relating $f\colon a \to x.b$ to $g\colon a \to y.b$ if there is a morphism $h\colon x \to y$ in $\mathcal{X}$ such that $g = f; (h.b)$.

In this case, we can recast Proposition 5.5.4 to get a result about $\mathcal{C}/\mathcal{X}$.

Corollary 5.6.1. *Let $\_.\_$ be an action of a monoidal category $\mathcal{X}$ on a category $\mathcal{C}$. If we identify this action with a lax functor $F\colon \mathbf{B}\mathcal{X} \to \mathbf{Cat}$, then $\mathcal{C}/\mathcal{X}$ is the lax colimit of this functor.*

In particular, there is a functor $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{X}$ and a natural transformation

$$\phi_{x,a}\colon J(x.a) \to Ja$$

making the following diagrams commute

$$\begin{array}{ccc} J(x.y.a) & \xrightarrow{\phi_{x,y.a}} & J(y.a) \\ {\scriptstyle J(m_{x,y,a})}\downarrow & & \downarrow{\scriptstyle \phi_{y,a}} \\ J((x \otimes y).a) & \xrightarrow{\phi_{x\otimes y,a}} & Ja \end{array} \qquad \begin{array}{ccc} Ja & & \\ {\scriptstyle J(\mathrm{l}_a)}\downarrow & \searrow^{\mathrm{id}} & \\ J(I.a) & \xrightarrow{\phi_{I,a}} & Ja \end{array},$$

such that if $F\colon \mathcal{C} \to \mathcal{D}$ is a functor and

$$\psi_{x,a}\colon F(x.a) \to Fa$$

is a natural transformation making the following diagrams commute

$$\begin{array}{ccc} F(x.y.a) & \xrightarrow{\psi_{x,y.a}} & F(y.a) \\ {\scriptstyle F(m_{x,y,a})}\downarrow & & \downarrow{\scriptstyle \psi_{y,a}} \\ F((x \otimes y).a) & \xrightarrow{\psi_{x\otimes y,a}} & Fa \end{array} \qquad \begin{array}{ccc} Fa & & \\ {\scriptstyle F(\mathrm{l}_a)}\downarrow & \searrow^{\mathrm{id}} & \\ F(I.a) & \xrightarrow{\psi_{I,a}} & Fa \end{array},$$

then there is a unique functor $\hat{F}\colon \mathcal{C}/\mathcal{X} \to \mathcal{D}$ *such that* $F = \hat{F}J$ *and* $\psi = \hat{F}\phi$.

Remark 5.6.2. Here, $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{X}$ is the identity on objects and sends a morphism $f\colon a \to b$ in $\mathcal{C}$ to the Melliès morphism

$$a \xrightarrow{f} b \xrightarrow{e_b} I.b\,,$$

while the component $\phi_{x,a}$ of ϕ is the identity

$$x.a \to x.a\,,$$

considered as a Melliès morphism $x.a \to a$.

Corollary 5.6.1 tells us that $\mathcal{C}/\mathcal{X}$ is generated as a category by the objects and morphisms in $\mathcal{C}$, together with the special morphisms making up the natural transformation $\phi_{x,a}\colon x.a \to a$.

In this special case, the proof of Proposition 5.5.4 tells us that $\mathcal{C}/\mathcal{X}$ always satisfies a 'factorization result' akin to those in [AM96] (our Proposition 3.5.1) and [HM99]: if $f\colon a \to b$ is a morphism in $\mathcal{C}/\mathcal{X}$, given by some morphism $\tilde{f}\colon a \to x.b$ in $\mathcal{C}$, then we may write f as the composite

$$a \xrightarrow{J(\tilde{f})} x.b \xrightarrow{\phi_{x,b}} b$$

in $\mathcal{C}$. This means that if we have a definability result for the category $\mathcal{C}$ – for example, that every compact morphism in $\mathcal{C}$ is definable in some language $\mathcal{L}$ – then we can automatically get a definability result for the category $\mathcal{C}/\mathcal{X}$ – for example, that every compact morphism in $\mathcal{C}/\mathcal{X}$ (i.e., every morphism $a \to b$ in $\mathcal{C}/\mathcal{X}$ that is given by a compact morphism $a \to x.b$ in $\mathcal{C}$) is definable in the language $\mathcal{L} + \Phi_{x,a}$, where $\Phi_{x,a}$ is a new family of primitives in the language whose denotations are given by the $\phi_{x,a}$.

In the case that $\mathcal{X} = 1$ and the action is a monad, this factorization result is Proposition 4.3.1, which played an important role in our main Full Abstraction result (Theorem 4.8.2). We shall use our new factorization result to prove a Full Abstraction result in a similar way.

5.7 Finite products distribute over lax colimits in Cat

The next result mirrors the result for distributivity of finite products over arbitrary colimits in a Cartesian closed category (which is essentially because the functor $A \times \_$ is a left adjoint, so preserves colimits).

Proposition 5.7.1. *Let $F\colon \mathcal{C} \to \mathbf{Cat}$ be a lax functor of bicategories, where $\mathcal{C}(a,b)$ has a small descendent set for any pair (a,b) of objects of $\mathcal{C}$. Let $\mathcal{A}$ be a category. Then the lax colimit of $\mathcal{A} \times F$ (defined by $(\mathcal{A} \times F)(c) = \mathcal{A} \times F(c)$ and $(\mathcal{A} \times F)(f) = (\mathrm{id}, f)$) is given by the product of $\mathcal{A}$ with the lax colimit of F.*

Proof. It is easiest to compute the colimit of $\mathcal{A} \times F$ directly using Proposition 5.5.4. The objects are tuples $(c,(a,m))$, where c is an object of $\mathcal{C}$, a is an object of a and m is an object of $F(c)$. 1-cells from $(c',(a',m'))$ to $(c,(a,m))$ are given by pairs $(h,(q,f))$, where h is a morphism $c \to c'$ in $\mathcal{C}$, f is a morphism $a' \to F(h)(a)$ in $F(a')$ and q is a morphism $a' \to a$ in $\mathcal{A}$. Two 1-cells $(h,(q,f))$ and $(h',(q',f'))$ from $(c',(a',m'))$ to $(c,(a,m))$ define equivalent morphisms if there is a 2-cell $\phi\colon h \Rightarrow h'$ such that $f; (F\phi)_m = f'$, and if $a' = a$.

Meanwhile, the product of $\mathcal{A}$ with the lax colimit of F has tuples $(a,(c,m))$ for objects, where a is an object of $\mathcal{A}$, c an object of $\mathcal{C}$ and m an object of $F(c)$. 1-cells $(a',(c',m')) \to (a,(c,m))$ are given by tuples $(q,(h,f))$, where q is a morphism $a' \to a$, h is a morphism $c \to c'$ and f is a morphism $m' \to F(h)(m)$. Two 1-cells $(q,(h,f))$ and $(q',(h',f'))$ from $(a',(c',m'))$ to $(a,(c,m))$ define equivalent morphisms if $a' = a$, and if there is a 2-cell $\phi\colon h \Rightarrow h'$ such that $f; (F\phi)_m = f'$. These two categories are clearly isomorphic. □

Corollary 5.7.2. *Let $F\colon \mathcal{C} \to \mathbf{Cat}$, $G\colon \mathcal{D} \to \mathbf{Cat}$ be lax functors of bicategories. Then*

$$\operatorname*{colim_l}_{c\colon \mathcal{C}, d\colon \mathcal{D}} (F(c) \times G(d)) \cong \operatorname*{colim_l}_{c\colon \mathcal{C}} F(c) \times \operatorname*{colim_l}_{d\colon \mathcal{D}} G(d)\,.$$

Proof. By Proposition 5.7.1, we have

$$\begin{aligned} &\operatorname*{colim_l}_{c\colon \mathcal{C}} F(c) \times \operatorname*{colim_l}_{d\colon \mathcal{D}} G(d) \\ \cong\ &\operatorname*{colim_l}_{c\colon \mathcal{C}} (F(c) \times \operatorname*{colim_l}_{d\colon \mathcal{D}} G(d)) \\ \cong\ &\operatorname*{colim_l}_{c\colon \mathcal{C}} \operatorname*{colim_l}_{d\colon \mathcal{D}} (F(c) \times G(d)) \\ \cong\ &\operatorname*{colim_l}_{c\colon \mathcal{C}, d\colon \mathcal{D}} (F(c) \times G(d))\,. \end{aligned}$$ □

In the particular case of a bifunctor out of $\mathbf{B}\mathcal{X}$, for $\mathcal{X}$ a monoidal category, we get the following.

Corollary 5.7.3. *Suppose we have actions of a monoidal category $\mathcal{X}$ on a category $\mathcal{C}$ and of a monoidal category $\mathcal{Y}$ on a category $\mathcal{D}$. We get an action of $\mathcal{X} \times \mathcal{Y}$ on $\mathcal{C} \times \mathcal{D}$ by*

$$(x, y).(c, d) = (x.c, y.d)\,.$$

Then

$$(\mathcal{C} \times \mathcal{D})/(\mathcal{X} \times \mathcal{Y}) \cong (\mathcal{C}/\mathcal{X}) \times (\mathcal{D}/\mathcal{Y})\,.$$

Proof. Since the $\mathcal{C}/\mathcal{X}$ construction is a special case of a lax colimit in **Cat**, this is a direct application of Corollary 5.7.2 where we are using the fact that $\mathbf{B}(\mathcal{X} \times \mathcal{Y}) \cong \mathbf{B}\mathcal{X} \times \mathbf{B}\mathcal{Y}$. □

5.8 Monoidal structure of $\mathcal{C}/\mathcal{X}$

We shall now consider the $\mathcal{C}/\mathcal{X}$ construction in more depth, looking at the properties that $\mathcal{C}/\mathcal{X}$ has when $\mathcal{C}$ is also a monoidal category. Until the end of the chapter, we assume that the monoidal category $\mathcal{X}$ has a small descendent set.

Definition 5.8.1. Suppose that $\mathcal{X}$ is a symmetric monoidal category acting on a monoidal category $\mathcal{C}$, and suppose moreover that the underlying functor $\mathcal{X} \times \mathcal{C} \to \mathcal{C}$ is a lax monoidal functor.

Then the functor $\mathcal{X} \times \mathcal{X} \times \mathcal{C} \to \mathcal{C}$ that sends (x, y, a) to $x.y.a$ is lax monoidal. Since $\mathcal{X}$ is symmetric monoidal, the tensor product $\_ \otimes \_ : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ is a strong monoidal functor and so the functor $\mathcal{X} \times \mathcal{X} \times \mathcal{C} \to \mathcal{C}$ that sends (x, y, a) to $(x \otimes y).a$ is also lax monoidal.

We say that the action of $\mathcal{X}$ on $\mathcal{C}$ is *monoidal* if the underlying functor $\mathcal{X} \times \mathcal{C} \to \mathcal{C}$ is a lax monoidal functor and the natural transformations

$$m_{x,y,a} \colon x.y.a \Rightarrow (x \otimes y).a \colon \mathcal{X} \times \mathcal{X} \times \mathcal{C} \to \mathcal{C} \qquad e_a \colon a \Rightarrow I.a \colon \mathcal{C} \to \mathcal{C}$$

are monoidal natural transformations.

Example 5.8.2. If $\mathcal{X}, \mathcal{C}$ are symmetric monoidal categories, where $\mathcal{C}$ is symmetric monoidal closed, and $j \colon \mathcal{X} \to \mathcal{C}$ is an oplax symmetric monoidal functor, then the parametric reader monad action

$$x.a = jx \multimap a$$

is a symmetric monoidal lax action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{C}$.

Remark 5.8.3. This generalizes the definition of a monad being *monoidal*; i.e., when the underlying functor $\mathcal{C} \to \mathcal{C}$ is a monoidal functor and the multiplication and unit for the monad are monoidal natural transformations. However, although a monoidal monad can be defined to be a monoid in the category of monoidal functors $\mathcal{C} \to \mathcal{C}$ and monoidal natural transformations between them, a monoidal parametric monad parameterized by a symmetric monoidal category $\mathcal{X}$ cannot be defined as a monoidal functor from $\mathcal{X}$ into this category. Indeed, that would mean we had, *inter alia*, a natural transformation

$$x.a \otimes x.b \to x.(a \otimes b)\,,$$

whereas we want to specify that there should be a natural transformation

$$x.a \otimes y.b \to (x \otimes y).(a \otimes b)\,.$$

Proposition 5.8.4. *Suppose that $\mathcal{X}$ is a symmetric monoidal category, and that we have a monoidal lax action of $\mathcal{X}$ on another monoidal category $\mathcal{C}$. Then $\mathcal{C}/\mathcal{X}$ inherits the structure of a monoidal category and the natural functor $J\colon \mathcal{C} \to \mathcal{C}/\mathcal{X}$ is strict monoidal.*

Proof. By taking the product of the action with itself, we get an action of $\mathcal{X} \times \mathcal{X}$ on $\mathcal{C} \times \mathcal{C}$; i.e., the action given by $(x, y).(a, b) = (x.a, y.b)$. Since $\mathcal{X}$ is symmetric monoidal, the tensor product functor $\mathcal{X} \times \mathcal{X} \to \mathcal{X}$ is strong monoidal, so by composing it with the action of $\mathcal{X}$ on $\mathcal{C}$ we get a lax action of $\mathcal{X} \times \mathcal{X}$ on $\mathcal{C}$; i.e., the action given by $(x, y).a = (x \otimes y).a$, with multiplicative coherence $m_{(x,y),(x',y').a}$ given by

$$(x \otimes y).(x' \otimes y').a \xrightarrow{m_{(x\otimes y),(x'\otimes y'),a}} ((x \otimes y) \otimes (x' \otimes y')).a \to ((x \otimes x') \otimes (y \otimes y')).a\,,$$

where the last arrow is given by the unique symmetric monoidal isomorphism; and with unital coherence e_a given by

$$a \xrightarrow{e_a} I.a \to (I \otimes I).a\,.$$

We claim that the functor $\_ \otimes \_ : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ and the monoidal coherence

$$m_{x,y,a,b}\colon x.a \otimes y.b \to (x \otimes y).(a \otimes b)$$

of the monoidal functor $\_ . \_ : \mathcal{X} \times \mathcal{C} \to \mathcal{C}$ give rise to an oplax morphism from the action of $\mathcal{X} \times \mathcal{X}$ on $\mathcal{C} \times \mathcal{C}$ to the action of $\mathcal{X} \times \mathcal{X}$ on $\mathcal{C}$.

Indeed, the first diagram in Definition 5.0.2 in this case is given by

$$\begin{array}{ccc}
(x.y.a) \otimes (x'.y'.a') & \xrightarrow{m_{x,x',y.a,y'.a'}} & (x \otimes x').(y.a \otimes y'.a') \\
{\scriptstyle m_{x,y,a} \otimes m_{x',y',a'}} \downarrow & & \downarrow {\scriptstyle (x \otimes x').m_{y,y',a,a'}} \\
((x \otimes y).a) \otimes ((x' \otimes y').a') & & (x \otimes x').(y \otimes y').(a \otimes a') \\
{\scriptstyle m_{x \otimes y, x' \otimes y', a, a'}} \downarrow & & \downarrow {\scriptstyle m_{x \otimes x', y \otimes y', a \otimes a'}} \\
((x \otimes y) \otimes (x' \otimes y')).(a \otimes a') & \longrightarrow & ((x \otimes x') \otimes (y \otimes y')).(a \otimes a')
\end{array},$$

which is precisely the diagram saying that m is a monoidal natural transformation. Similarly, the second diagram from Definition 5.0.2 is the same as the diagram saying that e is a monoidal natural transformation.

Therefore, by Proposition 5.4.4, this lax morphism of actions gives rise to a functor

$$(\mathcal{C} \times \mathcal{C})/(\mathcal{X} \times \mathcal{X}) \to \mathcal{C}/(\mathcal{X} \times \mathcal{X})\,.$$

The lax monoidal functor $\mathcal{X} \times \mathcal{X} \to \mathcal{X}$ gives rise by Proposition 5.4.5 to a functor $\mathcal{C}/(\mathcal{X} \times \mathcal{X}) \to \mathcal{C}/\mathcal{X}$, which we may compose with the functor above to give us a functor

$$(\mathcal{C} \times \mathcal{C})/(\mathcal{X} \times \mathcal{X}) \to \mathcal{C}/\mathcal{X}\,.$$

Moreover, Corollary 5.7.3 tells us that $(\mathcal{C} \times \mathcal{C})/(\mathcal{X} \times \mathcal{X})$ is isomorphic to $(\mathcal{C}/\mathcal{X}) \times (\mathcal{C}/\mathcal{X})$, giving us our desired functor

$$\_ \otimes \_ : (\mathcal{C}/\mathcal{X}) \times (\mathcal{C}/\mathcal{X}) \to (\mathcal{C}/\mathcal{X})\,.$$

The construction of this functor tells us that it commutes with the identity-on-objects functors out of the original categories:

$$\begin{array}{ccc}
\mathcal{C} \times \mathcal{C} & \xrightarrow{-\otimes-} & \mathcal{C} \\
\downarrow {\scriptstyle J \times J} & & \downarrow {\scriptstyle J} \\
(\mathcal{C}/\mathcal{X}) \times (\mathcal{C}/\mathcal{X}) & \xrightarrow{-\otimes-} & \mathcal{C}/\mathcal{X}
\end{array}\,.$$

This means that the functor J preserves the tensor product. It also means that we may lift the associators and unitors for the tensor product on $\mathcal{C}$ along the functor J morphisms in $\mathcal{C}/\mathcal{X}$, and that these morphisms satisfy the appropriate coherence diagrams. It remains to show that they are still natural transformations in $\mathcal{C}/\mathcal{X}$, and we do this in Figure 5.3. □

Having defined the monoidal structure on $\mathcal{C}/\mathcal{X}$ in a very formal way, let us unpack what it actually is. The tensor product on objects is defined exactly as in $\mathcal{C}$, while the tensor product of morphisms $f\colon a' \to a$, $g\colon b' \to b$

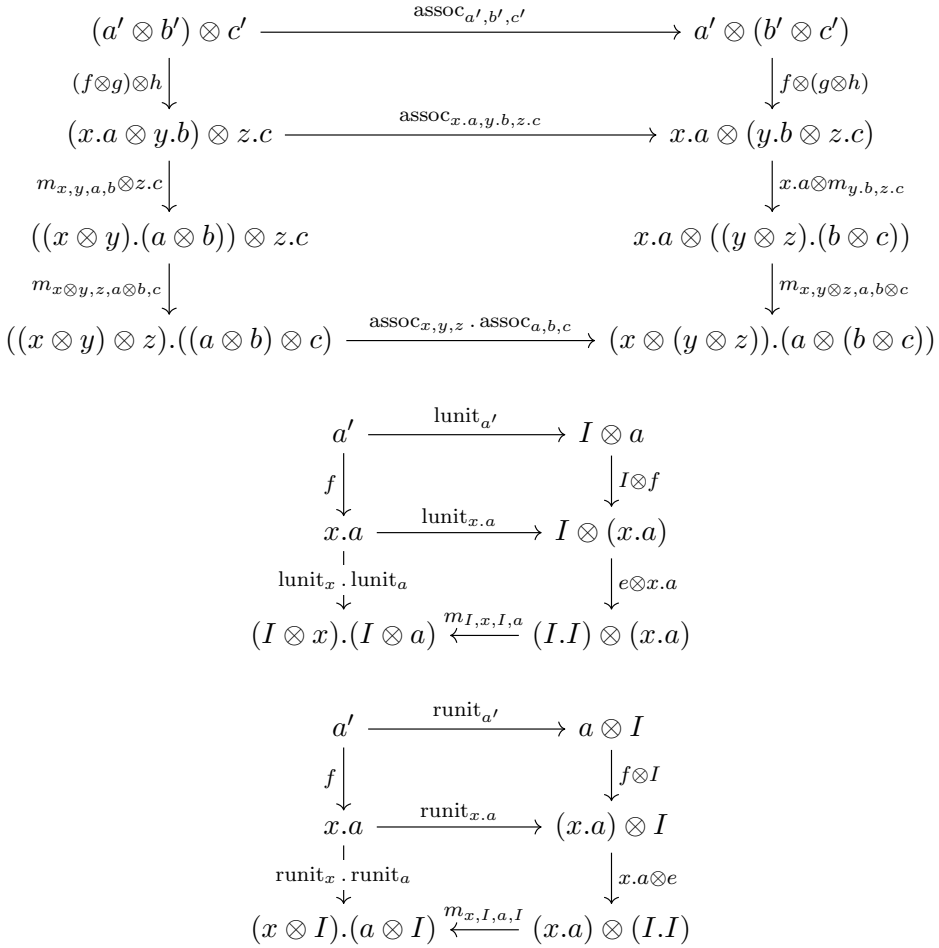


Figure 5.3: Proof that the associators and unitors in $\mathcal{C}/\mathcal{X}$ are indeed natural transformations. Here, $f\colon a' \to a$, $g\colon b' \to b$, $h\colon c' \to c$ are morphisms in $\mathcal{C}/\mathcal{X}$, considered as morphisms $a' \to x.a$, $b' \to y.b$, $c' \to z.c$ in $\mathcal{C}$.

(considered as morphisms $f\colon a' \to x.a$, $g\colon b' \to y.b$ in $\mathcal{C}$) is given by the composite

$$a' \otimes b' \xrightarrow{f \otimes g} (x.a) \otimes (y.b) \xrightarrow{m_{x,y,a,b}} (x \otimes y).(a \otimes b)\,,$$

where the right hand arrow is the multiplicative coherence for the monoidal functor $\_.\_ : \mathcal{X} \times \mathcal{C} \to \mathcal{C}$.

The reason we need some kind of symmetry in $\mathcal{X}$ is for this to be a functor: indeed, suppose that we have morphisms $f'\colon a'' \to a'$, $f\colon a' \to a$, $g'\colon b'' \to b'$ and $g\colon b' \to b$ in $\mathcal{C}/\mathcal{X}$, considered as morphisms $f'\colon a'' \to x'.a'$, $f\colon a' \to x.a$, $g'\colon b'' \to y'.b'$ and $g\colon b' \to y'b$ in $\mathcal{C}$. Then $(f' \otimes g');(f \otimes g)$ is given by the composite

$$\begin{aligned}
& a'' \otimes b'' \\
\xrightarrow{f' \otimes g'} \; & (x'.a') \otimes (y'.b') \\
\xrightarrow{m_{x',y',a',b'}} \; & (x' \otimes y').(a' \otimes b') \\
\xrightarrow{(x' \otimes y').(f \otimes g)} \; & (x' \otimes y').((x.a) \otimes (y.b)) \\
\xrightarrow{(x' \otimes y').m_{x,y,a,b}} \; & (x' \otimes y').(x \otimes y).(a \otimes b) \\
\xrightarrow{m_{x' \otimes y', x \otimes y, a \otimes b}} \; & ((x' \otimes y') \otimes (x \otimes y)).(a \otimes b)\,,
\end{aligned}$$

while $(f';f) \otimes (g';g)$ is given by

$$\begin{aligned}
& a'' \otimes b'' \\
\xrightarrow{f' \otimes g'} \; & (x'.a') \otimes (y'.b') \\
\xrightarrow{x'.f \otimes y'.g} \; & (x'.x.a) \otimes (y'.y.b) \\
\xrightarrow{m_{x',x,a} \otimes m_{y',y,b}} \; & ((x' \otimes x).a) \otimes ((y' \otimes y).b) \\
\xrightarrow{m_{x' \otimes x, y' \otimes y, a, b}} \; & ((x' \otimes x) \otimes (y' \otimes y)) \otimes (a \otimes b)\,.
\end{aligned}$$

Since $\mathcal{X}$ is symmetric, and since $m_{x,y,a}$ is a monoidal natural transformation, the natural symmetric monoidal coherence

$$((x' \otimes y') \otimes (x \otimes y)) \xrightarrow{\cong} ((x' \otimes x) \otimes (y' \otimes y))$$

mediates between these two composites, so they give us the same morphism in $\mathcal{C}/\mathcal{X}$.

5.9 Symmetric monoidal structure of $\mathcal{C}/\mathcal{X}$

Definition 5.9.1. Let a symmetric monoidal category $\mathcal{X}$ act via a monoidal lax action on a monoidal category $\mathcal{C}$. If $\mathcal{C}$ is symmetric monoidal, then we say

that the action of $\mathcal{X}$ on $\mathcal{C}$ is *symmetric monoidal* if the underlying functor $\mathcal{X} \times \mathcal{C} \to \mathcal{C}$ is a symmetric monoidal functor.

Example 5.9.2. If $j\colon \mathcal{X} \to \mathcal{C}$ is a symmetric oplax monoidal functor, where $\mathcal{C}$ is symmetric monoidal closed, then the reader action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{C}$ induced from j is a symmetric monoidal lax action.

Proposition 5.9.3. *Let a symmetric monoidal category $\mathcal{X}$ act via a symmetric monoidal lax action on a symmetric monoidal category $\mathcal{C}$. Then the category $\mathcal{C}/\mathcal{X}$ is symmetric monoidal.*

Proof. As with the associators and unitors, we can lift the symmetry isomorphisms from $\mathcal{C}$ on to $\mathcal{C}/\mathcal{X}$, and these isomorphisms will satisfy the appropriate coherence diagrams.

We need only show that they are natural transformations in $\mathcal{C}/\mathcal{X}$. Let $f\colon a' \to a$, $g\colon b' \to b$ be morphisms in $\mathcal{C}/\mathcal{X}$, considered as morphisms

$$f\colon a' \to x.a \qquad\qquad g\colon b' \to y.b$$

in $\mathcal{C}$. Then we need show that the following diagram commutes.

$$\begin{array}{ccc}
a' \otimes b' & \xrightarrow{\mathrm{sym}_{a',b'}} & b' \otimes a' \\
{\scriptstyle f\otimes g}\downarrow & & \downarrow{\scriptstyle g\otimes f} \\
(x.a) \otimes (y.b) & \xrightarrow{\mathrm{sym}_{x.a,y.b}} & (y.b) \otimes (x.a) \\
{\scriptstyle m_{x,y,a,b}}\downarrow & & \downarrow{\scriptstyle m_{y,x,b,a}} \\
(x \otimes y).(a \otimes b) & \xrightarrow{\mathrm{sym}_{x,y}\,.\,\mathrm{sym}_{a,b}} & (y \otimes x).(b \otimes a)
\end{array}$$

Indeed, the top square commutes because sym is a natural transformation in $\mathcal{C}$, and the bottom square commutes because the action is a symmetric monoidal functor. □

5.10 Monoidal closed structure of $\mathcal{C}/\mathcal{X}$

Let a symmetric monoidal category $\mathcal{X}$ act on a symmetric monoidal category $\mathcal{C}$ via a symmetric monoidal action. Since any lax monoidal functor between monoidal closed categories is automatically lax monoidal closed, we might expect that in such a situation the category $\mathcal{C}/\mathcal{X}$ would be symmetric monoidal closed.

In fact, this is not the case. For example, suppose that $\mathcal{X}$ is the unit monoidal category, and that $\mathcal{X}$ acts on the category of sets via the powerset monad. Then the category $\mathbf{Set}/\mathcal{X}$ is the Kleisli category for the powerset monad – i.e., the category of sets and relations.

Since the powerset functor is lax monoidal, it induces a monoidal structure on the Kleisli category – i.e., the familiar Cartesian product. In fact, the category of sets and relations *is* monoidal closed with respect to this choice of product, but this is a bit of a coincidence: in particular, the internal hom functor this category (which is also given by Cartesian product) does not agree with the function-space construction in the category of sets.

We are going to need to impose some stricter constraints on our action, then, in order to ensure that $\mathcal{C}/\mathcal{X}$ is monoidal closed. It turns out that we only need $\mathcal{C}$ to be monoidal closed, but that we need the action of $\mathcal{X}$ on $\mathcal{C}$ to preserve the internal hom *strictly*.

Definition 5.10.1. Let a symmetric monoidal category $\mathcal{X}$ act on a symmetric monoidal closed category $\mathcal{C}$ via a symmetric monoidal lax action. We say that the action is *symmetric monoidal closed* if there is a natural isomorphism

$$s_{x,a,b} \colon a \multimap (x.b) \xrightarrow{\cong} x.(a \multimap b)$$

that makes the following diagram commute for any object x of $\mathcal{X}$ and any objects a, b of $\mathcal{C}$.

$$\begin{array}{ccc} (x.(a \multimap b)) \otimes (I.a) & \xrightarrow{m_{x,I,a\multimap b,a}} & (x \otimes I).((a \multimap b) \otimes a) \\ {\scriptstyle s_{x,a,b}\otimes e_a}\uparrow & & \downarrow{\scriptstyle \mathrm{runit}^{-1} \otimes \mathrm{ev}_{a,b}} \\ (a \multimap x.b) \otimes a & \xrightarrow{\qquad\qquad \mathrm{ev}_{a,x.b} \qquad\qquad} & x.b \end{array}$$

Example 5.10.2. If j is an oplax symmetric monoidal functor from a symmetric monoidal category $\mathcal{X}$ to a symmetric monoidal closed category $\mathcal{C}$, then the reader action of X^{op} on $\mathcal{C}$ induced by j is symmetric monoidal closed, via the isomorphism

$$(jx \multimap b) \cong jx \multimap (a \multimap b)\,.$$

Proposition 5.10.3. *Suppose that a symmetric monoidal category $\mathcal{X}$ acts on a symmetric monoidal closed category $\mathcal{C}$ via a symmetric monoidal closed action. Then the category $\mathcal{C}/\mathcal{X}$ is symmetric monoidal closed.*

Proof. Given objects a, b of $\mathcal{C}$, we define $a \multimap b$ in $\mathcal{C}/\mathcal{X}$ to be the same as $a \multimap b$ in $\mathcal{C}$, and we define

$$\mathrm{ev}_{a,b} = J(\mathrm{ev}_{a,b}) \colon (a \multimap b) \otimes a \to b\,,$$

to be given by the same morphism as in $\mathcal{C}$.

Let $f\colon c \otimes a \to b$ be a morphism in $\mathcal{C}/\mathcal{X}$. It is necessary and sufficient to show that there is a unique morphism

$$h\colon c \to a \multimap b$$

in $\mathcal{C}/\mathcal{X}$ such that $f = (h \otimes a); \mathrm{ev}_{a,b}$.

Suppose that f is given by a morphism

$$\hat{f}\colon c \otimes a \to x.b$$

in $\mathcal{C}$. This then induces a morphism

$$\tilde{f} = W(\hat{f})\colon c \to (a \multimap x.b)\,,$$

which we can compose with $s_{x,a,b}$ to give us a morphism

$$\hat{h} = W(\hat{f}); s_{x,a,b}\colon c \to x.(a \multimap b)\,,$$

which we may consider as a morphism $h\colon c \to (a \multimap b)$ in $\mathcal{C}/\mathcal{X}$. We claim that $(h \otimes a); \mathrm{ev}_{a,b} = f$; indeed, the composite $(h \otimes \mathrm{id}_a); \mathrm{ev}_{a,b}$ is given in $\mathcal{C}$ by the composite

$$\begin{array}{ccccc} & & (x.(a \multimap b)) \otimes (I.a) & \xrightarrow{m_{x,I,a\multimap b,a}} & (x \otimes I).((a \multimap b) \otimes a) \\ & & {\scriptstyle s_{x,a,b}\otimes e_a}\uparrow & & \downarrow{\scriptstyle \mathrm{id}_{x\otimes I} \otimes \mathrm{ev}_{a,b}} \\ c \otimes a & \xrightarrow{\tilde{f}\otimes a} & (a \multimap x.b) \otimes a & & (x \otimes I).b \end{array}\,.$$

Using the diagram in Definition 5.10.1, we see that this is equal to the composite

$$c \otimes a \xrightarrow{\tilde{f}\otimes a} (a \multimap x.b) \otimes a \xrightarrow{\mathrm{ev}_{a,x.b}} x.b \xrightarrow{\mathrm{runit}_x .b} (x \otimes I).b\,.$$

The composite of the first two morphisms is equal to $\hat{f}$, and so the whole thing defines the same morphism as f in $\mathcal{C}/\mathcal{X}$, with runit_x mediating between the two.

Now suppose that $h\colon c \to a \multimap b$ is a morphism in $\mathcal{C}/X$, given by a morphism $\hat{h}\colon c \to x.(a \multimap b)$ in $\mathcal{C}$. Let $f\colon c \otimes a \to b$ be given by the morphism

$$\hat{f} = W^{-1}(\hat{h}; s^{-1}_{x,a,b})\,.$$

Then h may be recovered from f just as in the first part of this proof, and the equation $f = (h \otimes a); \mathrm{ev}_{a,b}$ determines f. It follows that the choice of h is unique. □

5.11 Cartesianness of the monoidal structure

Suppose that a symmetric monoidal category $\mathcal{X}$ acts on a Cartesian category $\mathcal{C}$ via a symmetric monoidal lax action. In this chapter we will consider what properties of the action we need in order to ensure that the induced monoidal structure on $\mathcal{C}/\mathcal{X}$ is Cartesian. Remember that the category of sets and relations is not Cartesian, even though it arises from an action of the unit category on the category of sets.

Since $\mathcal{C}$ is Cartesian, the diagonal and terminal maps in $\mathcal{C}$ gives every object a natural comonoid structure:

$$a \xrightarrow{\Delta_a} a \times a \qquad\qquad a \xrightarrow{()} 1\,.$$

The tensor product of objects A, B in $\mathcal{C}/\mathcal{X}$ is the same object $A \times B$ that defines the Cartesian product $\mathcal{C}$, so we can lift this comonoid structure through the functor J to give us a natural comonoid structure on each object of $\mathcal{C}/\mathcal{X}$:

$$a \xrightarrow{J\Delta_a} a \otimes a \qquad\qquad a \xrightarrow{J()} I\,.$$

We have replaced the symbol $\times$ with the symbol $\otimes$ when we are working in $\mathcal{C}/\mathcal{X}$, since we do not know whether this tensor product is still Cartesian in $\mathcal{C}/\mathcal{X}$.

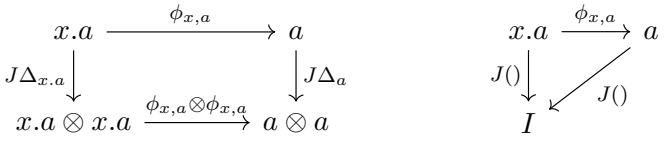
If every morphism $f\colon a \to b$ in $\mathcal{C}/\mathcal{X}$ is a comonoid homomorphism with respect to this comonoid structure; i.e., if it makes the diagrams

$$\begin{array}{ccc} a & \xrightarrow{f} & b \\ {\scriptstyle J\Delta_a}\downarrow & & \downarrow{\scriptstyle J\Delta_b} \\ a \otimes a & \xrightarrow{f\otimes f} & b \otimes b \end{array} \qquad\qquad \begin{array}{ccc} a & \xrightarrow{f} & b \\ {\scriptstyle J()}\downarrow & \swarrow_{J()} & \\ I & & \end{array}$$

commute, then we may identify $\mathcal{C}/\mathcal{X}$ with a full subcategory of its own category of comonoids, closed under tensor product. Since the category of comonoids in a monoidal category is always Cartesian, this will tell us that $\mathcal{C}/\mathcal{X}$ is Cartesian.

What does it mean for every morphism in $\mathcal{C}/\mathcal{X}$ to be a comonoid homomorphism? Firstly, since $\mathcal{C}$ itself is Cartesian, every morphism in $\mathcal{C}$ is a comonoid homomorphism with respect to the diagonal, and it follows that every morphism of the form $Jf\colon a \to b$ (where $f\colon a \to b$ is a morphism in $\mathcal{C}$) is a comonoid homomorphism in $\mathcal{C}/\mathcal{X}$. Since every morphism in $\mathcal{C}/\mathcal{X}$ may be written as the composite of a morphism of the form Jf with a morphism of the form $\phi_{x,b}$, it will suffice to show that the morphisms of the form $\phi_{x,a}$ are comonoid homomorphisms in $\mathcal{C}/\mathcal{X}$; i.e., that they make the following

diagrams commute.



Let us compute the two arms of the first square directly in $\mathcal{C}$. The composite along the top right is given in $\mathcal{C}$ by the morphism

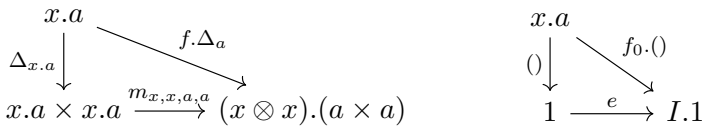
$$x.a \xrightarrow{x.\Delta_a} x.(a \times a)\,,$$

while that along the bottom right is given by the composite

$$x.a \xrightarrow{\Delta_{x.a}} x.a \times x.a \xrightarrow{m_{x,x,a,a}} (x \otimes x).(a \times a)\,.$$

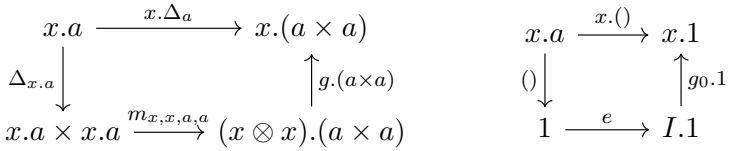
Proposition 5.11.1. *Let a symmetric monoidal category $\mathcal{X}$ act on a Cartesian category $\mathcal{C}$ via a symmetric monoidal lax action. Suppose that* ***either***

- *for every object x of $\mathcal{X}$ there are morphisms $f\colon x \to x\otimes x$ and $f_0\colon x \to I$ that make the following diagrams commute;*



  or

- *for every object x of $\mathcal{X}$ there is are morphisms $g\colon x \otimes x \to x$ and $g_0\colon I \to x$ that make the following diagrams commute.*



Then the monoidal structure on $\mathcal{C}/\mathcal{X}$ is Cartesian.

Proof. In either case, the morphisms f, f_0 or g, g_0 mediate between the composites from our earlier discussion. □

Note that these results are not the sharpest possible: there may well be more complicated zigzags mediating between the morphisms. Nevertheless, they are sufficient for our purposes.

An obvious idea for constructing the morphisms f, f_0 is to look at the case that $\mathcal{X}$ is itself Cartesian, taking f to be the diagonal and f_0 the terminal morphism. Note, however, that this does not automatically mean that the diagrams in Proposition 5.11.1 commute: for example, the trivial category and the category of sets are both presheaf categories and hence Cartesian closed, but if 1 acts on the category of sets via the powerset monad then the resulting monoidal category $\mathbf{Set}/1$ (i.e., the category of sets and relations with the set-theoretic product) is not Cartesian.

If $\mathcal{X}$ is Cartesian and $\mathcal{X}^{\mathrm{op}}$ acts on $\mathcal{C}$, then we can try taking g and g_0 to be (the opposites of) the diagonal and terminal morphism in $\mathcal{X}$. This will be particularly useful for reader actions, where an oplax monoidal functor $j\colon \mathcal{X} \to \mathcal{C}$ gives rise to an action of the opposite category $\mathcal{X}^{\mathrm{op}}$ of $\mathcal{X}$ on $\mathcal{C}$. If $\mathcal{C}$ is Cartesian and we want to show that $\mathcal{C}/\mathcal{X}$ is Cartesian, then it will be sufficient to find, for each object x of $\mathcal{X}$, morphisms $f\colon x \to x \otimes x$ and $f_0\colon x \to I$ such that the following diagrams commute.

$$\begin{array}{ccc} jx \to a & \xrightarrow{jx \to \Delta_a} & jx \to (a \times a) \\ {\scriptstyle \Delta_{jx\to a}}\downarrow & & \uparrow{\scriptstyle jf \to (a\times a)} \\ (jx \to a) \times (jx \to a) \to (jx \times jx) \to (a \times a) & \xrightarrow{m^j_x \to (a\times a)} & j(x \otimes x) \to (a \times a) \end{array}$$

$$\begin{array}{ccc} jx \to a & \xrightarrow{jx \to ()} & jx \to 1 \\ {\scriptstyle ()}\downarrow & & \uparrow{\scriptstyle jf_0 \to 1} \\ 1 \to 1 \to 1 & \xrightarrow{e^j \to 1} & jI \to 1 \end{array}$$

For these to commute, it is sufficient that the composite

$$jx \xrightarrow{jf} j(x \otimes x) \xrightarrow{m^j_{x,x}} jx \times jx$$

should be equal to the diagonal on jx, and that the composite

$$jx \xrightarrow{jf_0} jI \xrightarrow{e^j} 1$$

should be equal to the terminal morphism $jx \to 1$. Of course, this second condition is automatically satisfied by the definition of a terminal object, so all we require is that some morphism $f_0\colon x \to I$ should exist.

We have proved:

Theorem 5.11.2. *Let $j\colon \mathcal{X} \to \mathcal{C}$ be an oplax symmetric monoidal functor between symmetric monoidal categories, where $\mathcal{C}$ is Cartesian closed. Then j induces a reader action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{C}$, and the category $\mathcal{C}/\mathcal{X}^{\mathrm{op}}$ is symmetric monoidal closed.*

Suppose that for every object x of $\mathcal{X}$, there are morphisms $f\colon x \to x \otimes x$ and $f_0\colon x \to I$ in $\mathcal{X}$ such that $jf; m^j_{x,x} = \Delta_{jx}$. Then $\mathcal{C}/\mathcal{X}^{\mathrm{op}}$ is Cartesian closed.

In the next chapter, we will investigate a class of reader actions, and give a more concrete property for the category $\mathcal{C}/\mathcal{X}^{\mathrm{op}}$ to be Cartesian closed. We will start to see how we can come up with reader actions to model different computational effects and come closer to our eventual Computational Adequacy and Full Abstraction results.

Chapter 6

Reader actions on Set

6.1 Colimits of actions are monoidal functors

Proposition 6.1.1. *Let $\mathcal{C}$ be a cocomplete monoidal category, and suppose that a symmetric monoidal category $\mathcal{X}$ with a small descendent set acts on $\mathcal{C}$ via a symmetric monoidal action in the sense of Definition 5.8.1. Define a functor $\mathcal{C} \to \mathcal{C}$ by*

$$Fa = \varinjlim_{x\colon \mathcal{X}} x.a\,.$$

Then F is a lax monoidal functor.

Proof. We have morphisms

$$\begin{aligned} & \varinjlim_{x\colon \mathcal{X}} x.a \otimes \varinjlim_{y\colon \mathcal{X}} y.b \\ \to\ & \varinjlim_{x,y\colon \mathcal{X}} x.a \otimes y.b \\ \to\ & \varinjlim_{x,y\colon \mathcal{X}} (x \otimes y).(a \otimes b) \\ \hookrightarrow\ & \varinjlim_{z\colon \mathcal{X}} z.(a \otimes b)\,. \end{aligned}$$

and

$$1 \xrightarrow{e} I.1 \hookrightarrow \varinjlim_{x\colon \mathcal{X}} x.1\,.$$

Figures 6.1 and 6.2 show that these satisfy the coherence conditions for a monoidal functor. □

Proposition 6.1.2. *If $\mathcal{C}$ is a cocomplete symmetric monoidal category and the action of $\mathcal{X}$ on $\mathcal{C}$ is symmetric monoidal, then*

$$\varinjlim_{x\colon \mathcal{X}} x.\_$$

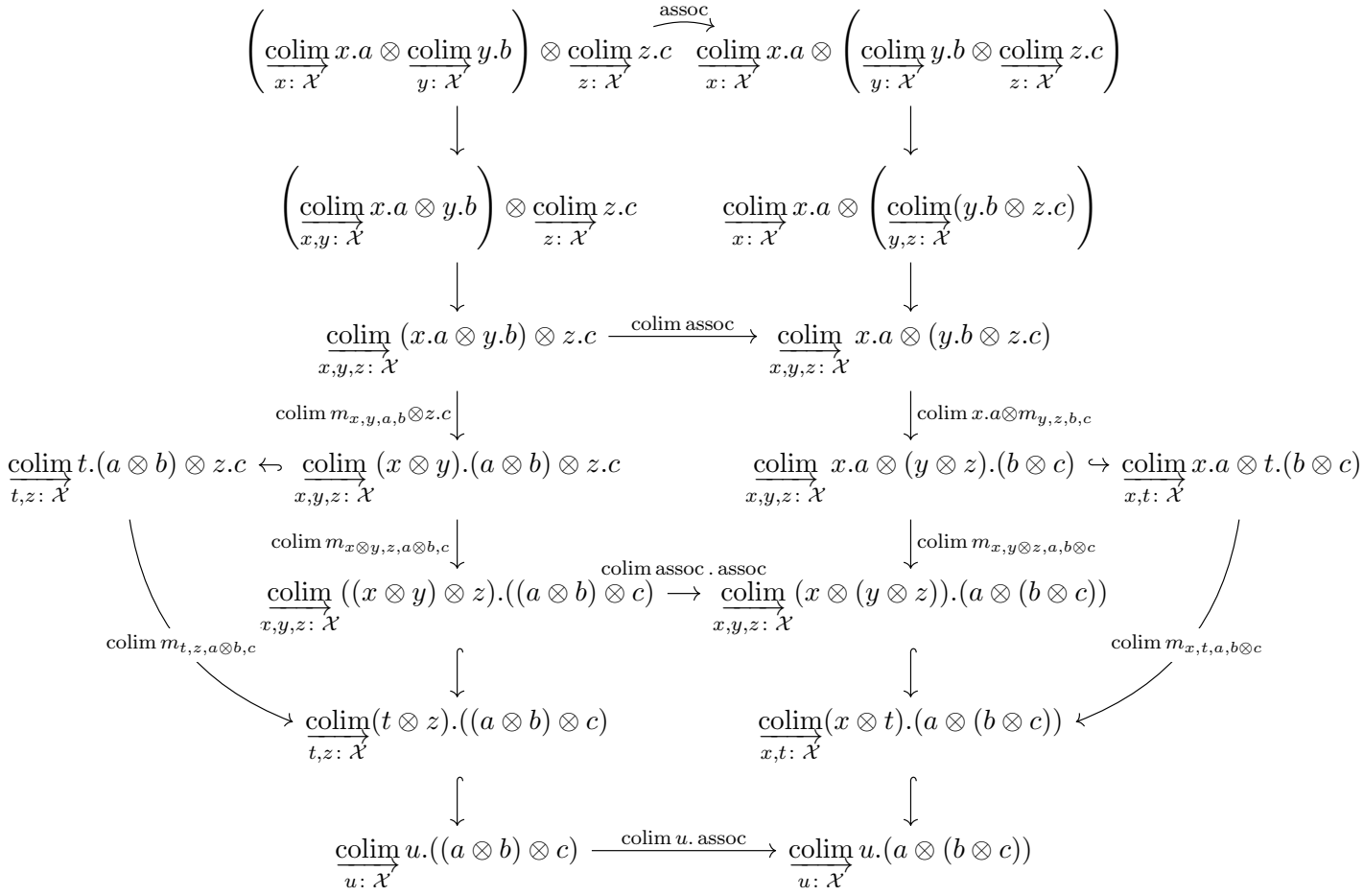


Figure 6.1: Proof that $\varinjlim_{x\colon \mathcal{X}} x.a$ satisfies the multiplicative criterion for being a monoidal functor. The bottom hexagon is the multiplicative coherence for $\_.\_$ as a monoidal functor, after applying the colimit.

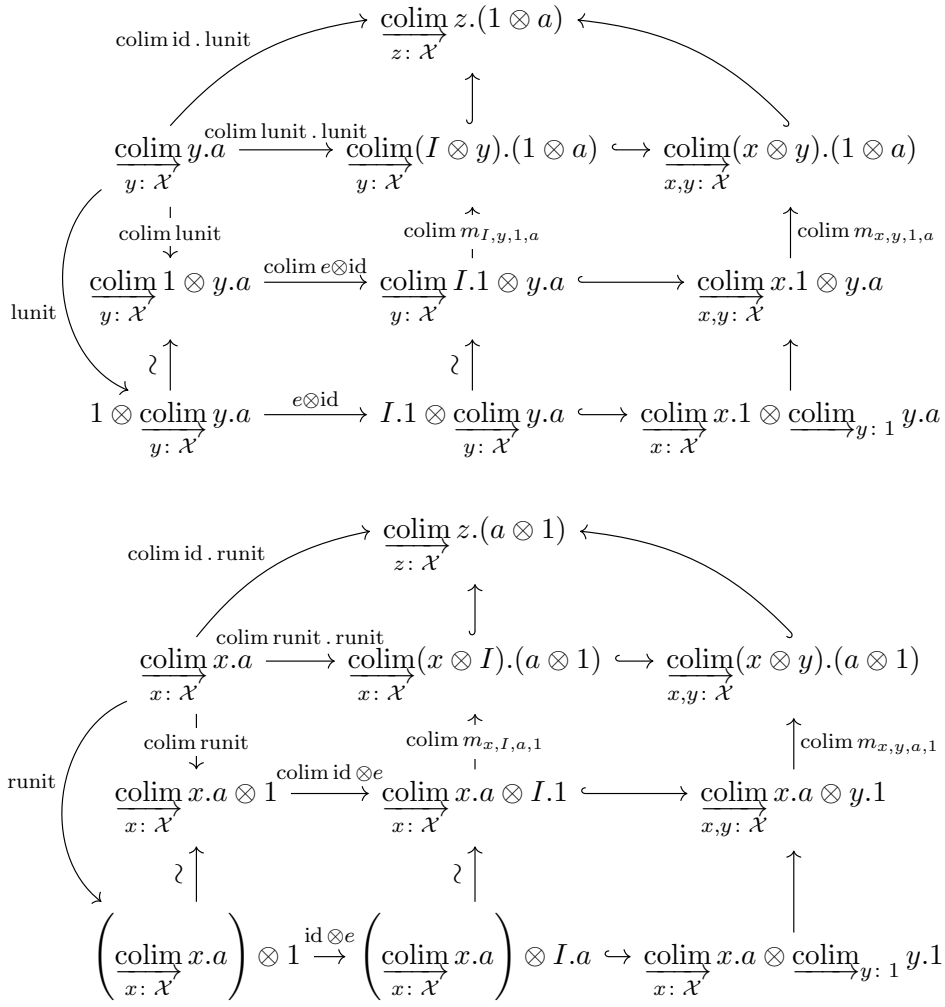


Figure 6.2: Proof that $\varinjlim_{x\colon \mathcal{X}} x.a$ satisfies the unital criteria for being a monoidal functor. The top-right squares are the unital coherence for $\_.\_$ as a monoidal functor, after applying the colimit.

is a symmetric monoidal functor.

Proof. We have a commutative diagram

$$\begin{array}{ccc}
\operatorname*{colim}_{x\colon \mathcal{X}} x.a \otimes \operatorname*{colim}_{y\colon \mathcal{X}} y.b & \xrightarrow{\text{sym}} & \operatorname*{colim}_{y\colon \mathcal{X}} y.b \otimes \operatorname*{colim}_{x\colon \mathcal{X}} x.a \\
\downarrow & & \downarrow \\
\operatorname*{colim}_{x,y\colon \mathcal{X}} x.a \otimes y.b & \xrightarrow{\operatorname{colim} \text{sym}} & \operatorname*{colim}_{x,y\colon \mathcal{X}} y.b \otimes x.a \\
{\scriptstyle \operatorname{colim} m_{x,y,a,b}}\downarrow & & \downarrow{\scriptstyle \operatorname{colim} m_{y,x,b,a}} \\
\operatorname*{colim}_{x,y\colon \mathcal{X}} (x \otimes y).(a \otimes b) & \xrightarrow{\operatorname{colim} sym.sym} & \operatorname*{colim}_{x,y\colon \mathcal{X}} (y \otimes x).(b \otimes a) \\
\downarrow & & \downarrow \\
\operatorname*{colim}_{z\colon \mathcal{X}} z.(a \otimes b) & \xrightarrow{\operatorname{colim} \text{id} \,.\, \text{sym}} & \operatorname*{colim}_{z\colon \mathcal{X}} z.(b \otimes a)
\end{array}.$$

Here, commutativity of the middle square is by applying the colimit to the diagram for $\_.\_$ to be a symmetric monoidal functor. □

6.2 Reader actions on Set vs change of base

For our next result, we will go via the Melliès category. First, we need a standard result about the Day convolution product.

Proposition 6.2.1 ([Pis14])**.** *Let $\mathcal{X}, \mathcal{Y}$ be monoidal categories, where $\mathcal{X}$ is well-tensored, and let the functor category $[\mathcal{X}, \mathbf{Set}]$ be equipped with the Day convolution product.*

Given a functor $F\colon \mathcal{Y} \times \mathcal{X} \to \mathbf{Set}$, we have an associated functor $\Lambda(F)\colon \mathcal{Y} \to [\mathcal{X}, \mathbf{Set}]$. Then F is lax monoidal if and only if $\Lambda(F)$ is lax monoidal.

In particular, if $\mathcal{X}$ acts on the category of sets via a monoidal action, then it gives rise to a monoidal functor $\mathbf{Set} \to [\mathcal{X}, \mathbf{Set}]$.

Proposition 6.2.2. *Let $j\colon \mathcal{X} \to \mathbf{Set}$ be an oplax symmetric monoidal functor between symmetric monoidal categories, where $\mathcal{X}$ is well-tensored. Let $\mathcal{X}^{\mathrm{op}}$ act on $\mathbf{Set}$ via the reader action induced by j. Then the Melliès category $\mathrm{Mell}_{\mathcal{X}^{\mathrm{op}}} \mathbf{Set}$ is isomorphic to the category obtained via base change along the functor $\mathbf{Set} \to [\mathcal{X}, \mathbf{Set}]$ obtained from the action $\_.\_$.*

Proof. The objects of both categories are sets. The morphism objects in the Melliès category are given by

$$\mathrm{Mell}_{\mathcal{X}} \mathbf{Set}(A, B)(x) = [A, [jx, B]]\,,$$

while those in the base-changed category are given by

$$(\_ \cdot \_)_* \mathbf{Set}(A, B)(x) = [jx, [A, B]]\,,$$

and these may be related by the symmetry isomorphism

$$s_{jx,A,B} \colon [A, [jx, B]] \to [jx, [A, B]]\,.$$

We need to show that this preserves composition of morphisms. Recall that composition in the Melliès category is given by

$$\begin{aligned} & \int^{y,z} [A, [jy, B]] \times [B, [jz, C]] \times \mathcal{X}(y \otimes z, x) \\ \to & \int^{y,z} [A, [j(y \otimes z), C]] \times \mathcal{X}(y \otimes z, x) \\ \cong & \; [A, [jx, C]]\,, \end{aligned}$$

where the first arrow is induced via the Melliès composition, while composition in the base changed category is given by

$$\begin{array}{ccc} & & \displaystyle\int^{y,z} [jy, [A, B]] \times [jz, [B, C]] \times \mathcal{X}(y \otimes z, x) \\ \displaystyle\int^{y,z} m_{jy,jz,[A,B],[B,C]} \times \mathcal{X}(y \otimes z, x) & \longrightarrow & \displaystyle\int^{y,z} [jy \times jz, [A, B] \times [B, C]] \times \mathcal{X}(y, \otimes z, x) \\ \displaystyle\int^{y,z} [m^j_{y,z}, ;] \times \mathcal{X}(y \otimes z, x) & \longrightarrow & \displaystyle\int^{y,z} [j(y \otimes z), [A, C]] \times \mathcal{X}(y \otimes z, x) \\ \cong & & [jx, [A, C]]\,, \end{array}$$

where ; is the internal composition in **Set**.

Using the diagram in Figure 6.3, we see that the expressions inside the coends are related by the symmetry isomorphisms as follows.

$$\begin{array}{ccc} [A, [jy, B]] \times [B, [jz, C]] & \xrightarrow{\text{Melliès}} & [A, [j(y \otimes z), C]] \\ \downarrow{\scriptstyle s_{jx,A,B} \times s_{jy,B,C}} & & \downarrow{\scriptstyle s_{j(x \otimes y),A,C}} \\ [jy, [A, B]] \times [jz, [B, C]] & \xrightarrow{\text{B.c.}} & [j(y \otimes z), [A, C]] \end{array}$$

It follows that the functor induced by s is an isomorphism of $[\mathcal{X}, \mathbf{Set}]$-enriched categories. □

By applying base change along the colimit functor to both these categories, we get the following.

Corollary 6.2.3. *Let $j \colon \mathcal{X} \to \mathbf{Set}$ be a symmetric monoidal functor, where $\mathcal{X}$ has a small ancestral set (so $\mathcal{X}^{\mathrm{op}}$ has a small descendent set). Let $\mathcal{X}^{\mathrm{op}}$*

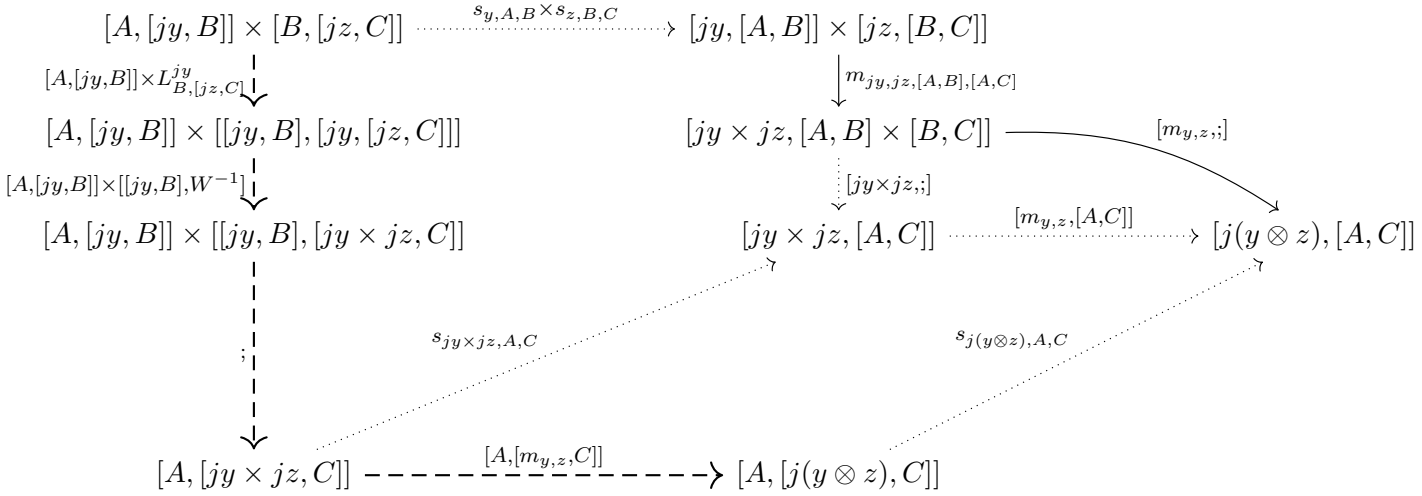


Figure 6.3: Proof that Melliès composition agrees with base-changed composition in the case of a symmetric reader action on **Set**. The Melliès composition is given by the thick dashed arrows, while the composition in the base-changed category is given by the thin arrows. The dotted lines at the top and at the bottom right – given by the symmetry isomorphisms – mediate between the two. We can verify that the main heptagon commutes by directly computing each direction: in each case, a pair $\langle f, g \rangle$ of functions is sent to the function $h \colon A \to [jy \times jz, C]$ given by

$$h(a)(Y, Z) = g(f(a)(Y))(Z)\,.$$

act on **Set** *via the induced reader action. Then the category* $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ *is isomorphic to the category obtained from* **Set** *by base change along the functor*

$$\varinjlim_{x\colon \mathcal{X}} x.\_ : \mathbf{Set} \to \mathbf{Set}\,.$$

Thus, the theory of reader actions on **Set** is subsumed into the theory of base change in **Set** along monoidal functors $\mathbf{Set} \to \mathbf{Set}$.

6.3 From monoidal endofunctors to reader actions

We can get a result in the other direction; i.e., a kind of converse to Proposition 6.1.1 in the case of reader actions on **Set**.

Proposition 6.3.1. *Let* $F\colon \mathbf{Set} \to \mathbf{Set}$ *be a lax symmetric monoidal functor. Then there is a symmetric monoidal category* $\mathcal{X}$ *and an oplax monoidal functor* $j\colon \mathcal{X} \to \mathbf{Set}$ *such that for all sets* A*, we have*

$$FA \cong \varinjlim_{x\colon \mathcal{X}} [jx, A]\,,$$

and the monoidal coherences of F *arise from the reader action of* $\mathcal{X}^{\mathrm{op}}$ *on* **Set** *as in Proposition 6.1.1.*

Proof. By considering each set $F(A)$ as a category with only identity morphisms, we can imagine F as a 2-functor $\mathbf{Set} \to \mathbf{Cat}$. Let $\mathcal{X} = \left(\int F\right)^{\mathrm{op}}$ be the opposite of its Grothendieck construction. Since **Set** is an ordinary category (as opposed to a proper bicategory), $\mathcal{X}$ has only identity 2-morphisms, so may be considered as an ordinary category.

More concretely, the objects of $\mathcal{X}$ are pairs (A, m), where A is a set and $m \in FA$, and morphisms

$$(A, m) \to (B, n)$$

are given by functions $h\colon A \to B$ such that $F(h)(m) = n$. $\mathcal{X}$ is normally called the *category of elements of* F.

We define a symmetric monoidal structure on $\mathcal{X}$ by setting

$$(A, m) \otimes (B, n) = (A \times B, m^F_{A,B}(m, n)) \qquad I = (1, e^F)\,,$$

using the fact that the monoidal coherences for F are given by functions

$$m^F_{A,B}\colon FA \times FB \to F(A \times B) \qquad e^F\colon 1 \to F1\,.$$

We now need to show that the monoidal coherences in **Set** give rise to morphisms

$$\mathrm{assoc}_{A,B,C}\colon ((A\times B)\times C, m^F_{A\times B,C}(m^F_{A,B}(m,n),p)) \to$$

$$(A\times(B\times C), m^F_{A,B\times C}(m, m^F_{B,C}(n,p))$$

$$\mathrm{lunit}_A\colon (A,m)\to(1\times A, m^F_{1,A}(e^F,m))$$

$$\mathrm{runit}_A\colon (A,m)\to(A\times a, m^F_{A,1}(m,e^F))$$

$$\mathrm{sym}_{A,B}\colon (A\times B, m^F_{A,B}(m,n))\to(B\times A, m^F_{B,A}(n,m))$$

in $\mathcal{X}$. Happily, the diagrams we need for this are precisely the coherence diagrams for m^F, e^F that we get from F being a lax symmetric monoidal functor.

Since assoc, lunit, runit, sym satisfy the pentagon, triangle and hexagon identities in **Set**, so they do in $\mathcal{X}$. Therefore, $\mathcal{X}$ is a symmetric monoidal category.

There is an obvious forgetful functor $\mathcal{X}\to\mathbf{Set}$ that is, in fact, strict monoidal.

We claim that if A is a set, then we have

$$\varinjlim_{(X,m)\colon \mathcal{X}^{\mathrm{op}}} [X,A] \cong FA\,.$$

To see this, note that for each object (X,m) of $\mathcal{X}$ there is a function

$$\begin{aligned}[X,A]&\to \quad FA\\ f\quad &\mapsto F(f)(m)\,.\end{aligned}$$

We claim that this defines a cocone under the functor

$$(X,m)\mapsto[X,A]\colon \mathcal{X}^{\mathrm{op}}\to\mathbf{Set}\,.$$

Indeed, if $h\colon (Y,n)\to(X,m)$ is a morphism then by definition we have

$$F(h)(n)=m\,,$$

and so we get a commutative triangle

$$\begin{array}{ccc}[X,A] & \xrightarrow{f\mapsto F(f)(m)} & FA\\ {\scriptstyle f\mapsto h;f}\big\downarrow & \nearrow_{g\mapsto F(g)(n)} & \\ [Y,A] & & \end{array}\,,$$

since

$$F(h;f)(n) = F(f)(F(h)(n)) = F(f)(m)\,.$$

Therefore, there is an induced map

$$\varinjlim_{(X,m)\colon \mathcal{X}} [X, A] \to FA$$

that sends $((X, m), f)$ to $F(f)(m)$.

Now we define a map in the other direction.

$$\begin{aligned} FA &\to \varinjlim_{(X,m)\colon \mathcal{X}} [X, A] \\ m &\mapsto ((A, m), \mathrm{id}_A) \end{aligned}$$

We claim that these two maps are inverses. Indeed, we certainly have

$$F(\mathrm{id}_A)(m) = id_{FA}(m) = m\,.$$

In the other direction, we need to show that

$$((A, F(f)(m)), \mathrm{id}_A) = ((X, m), f)$$

in the colimit. But we have a morphism $f\colon (X, m) \to (A, F(f)(m))$ in $\mathcal{X}^{\mathrm{op}}$, and $f; \mathrm{id}_A = f$. Therefore, our map $\varinjlim_{(X,m)\colon \mathcal{X}}[X, A] \to FA$ was a bijection.

Lastly, we need to show that this decomposition as a colimit gives rise to the monoidal structure on the functor F; i.e., that the following diagrams commute.

$$\begin{array}{ccc} FA \times FB & \xrightarrow{\quad m^F_{A,B} \quad} & F(A \times B) \\ \wr\downarrow & & \downarrow\wr \\ \varinjlim_{(X,m)\colon \mathcal{X}} [X, A] \times \varinjlim_{(Y,n)\colon \mathcal{X}} [Y, B] & \longrightarrow & \varinjlim_{(Z,p)\colon \mathcal{X}} [Z, A \times B] \end{array}$$

$$\begin{array}{ccc} 1 & \xrightarrow{e^F} & F1 \\ & \searrow & \downarrow\wr \\ & & \varinjlim_{(X,m)\colon \mathcal{X}} [X, 1] \end{array}$$

Here, the arrows marked with the isomorphism symbol $\cong$ are the isomorphisms that we have just defined, while the arrows at the bottom of the first diagram and at the bottom left of the second are as in Proposition 6.1.1.

We can check by hand that these diagrams commute: that in the first diagram, both directions send the pair $(m, n) \in FA \times FB$ to

$$((A \times B, m_{A,B}(m,n)), \mathrm{id}_{A\times B}) \in \varinjlim_{(Z,p)\colon \mathcal{X}} [Z, A \times B]\,,$$

and that in the second diagram both directions pick out the element

$$((1, e^F), \mathrm{id}_1)$$

of $\varinjlim_{(X,m)\colon \mathcal{X}} [X, 1]$. □

Remark 6.3.2. If the category $\mathcal{X}$ that we have constructed has a small ancestral set, this means that $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ is isomorphic to the category obtained from **Set** via base change through F.

If $\mathcal{X}$ has not got a small ancestral set, then this is still true, except that now $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ is not technically defined. This is not really important: we only need $\mathcal{X}^{\mathrm{op}}$ to have a small descendent set so that we can prove that $\varinjlim_{x\colon \mathcal{X}} [jx, A]$ is a small set, but in this case we know that anyway – $\varinjlim_{x\colon \mathcal{X}} [jx, A] \cong F(A)$!

In any case, our main examples will satisfy the smallness conditions on $\mathcal{X}$ that we have been working with up to this point.

Example 6.3.3. Let $\mathcal{P}_+\colon \mathbf{Set} \to \mathbf{Set}$ be the non-empty powerset functor. Then the category $\mathcal{X}$ of elements of $\mathcal{P}_+$ has pairs (A, M) as elements, where A is a (necessarily non-empty) set and M a non-empty subset of A. Morphisms $(A, M) \to (B, N)$ are functions $f\colon A \to B$ such that $f(M) = N$.

The nonempty powerset functor has a natural monoidal structure:

$$\mathcal{P}_+A \times \mathcal{P}_+B \to \mathcal{P}_+(A \times B) \qquad\qquad 1 \to \mathcal{P}_+1$$

sending (M, N) to $M \times N \subseteq A \times B$ and picking out the subset $1 \subseteq 1$. Therefore this category of elements is a monoidal category, and has a natural induced action as in Proposition 6.3.1. The colimit of this action then gives us the original functor. Therefore, by Corollary 6.2.3, the category $\mathbf{Set}/\mathcal{X}$ is isomorphic to the category $(\mathcal{P}_+)_*\mathbf{Set}$ obtained by base change along the non-empty powerset functor $\mathcal{P}_+\colon \mathbf{Set} \to \mathbf{Set}$.

Example 6.3.4. Let $\mathrm{DG}\colon \mathbf{Set} \to \mathbf{Set}$ be the functor that sends a set A to the set of discrete probability measures on A and sends a function $f\colon A \to B$ to the function $\mathrm{DG}(A) \to \mathrm{DG}(B)$ that sends a probability measure $\mathbb{P}$ on A to the probability measure $f_*\mathbb{P}$ on B given by

$$f_*\mathbb{P}(X) = \mathbb{P}(f^{-1}(X))\,.$$

Then the category of elements of DG is the category whose objects are pairs $(A, \mathbb{P})$ and where the morphisms $(A, \mathbb{P}_A) \to (B, \mathbb{P}_B)$ are functions $f\colon A \to B$ such that $\mathbb{P}_B = f_* \mathbb{P}_A$. In other words, it is the category of discrete probability spaces and probability-preserving functions.

The functor DG has a natural monoidal structure: given a discrete probability measure on a set A and a discrete probability measure on a set B, we can get a discrete probability measure on the set $A \times B$ by

$$\mathbb{P}_{A\times B}(\{(a,b)\}) = \mathbb{P}_A(\{a\}) \times \mathbb{P}_B(\{b\})\,.$$

This then gives us a monoidal structure on the category of discrete probability spaces, where the tensor product of probability spaces $(A, \mathbb{P}_A)$ and $(B, \mathbb{P}_B)$ is given by the set $A \times B$, together with the probability measure as described above.

Now suppose that $F\colon \mathbf{Set} \to \mathbf{Set}$ is a lax symmetric monoidal endofunctor. We have shown that F gives rise to a strict symmetric monoidal functor $j\colon \mathcal{X} \to \mathbf{Set}$, for some monoidal category $\mathcal{X}$, and that the induced category $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ is isomorphic to the category obtained from $\mathbf{Set}$ by base change along F.

Since the action of $\mathcal{X}$ on $\mathbf{Set}$ is the reader action of a symmetric monoidal functor, by Theorem 5.11.2 we know that the category $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ must be symmetric monoidal closed.

Now recall that in order to apply the second part of Theorem 5.11.2, and deduce that $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ is Cartesian closed, we must prove that for every object x of $\mathcal{X}$, there are morphisms $f\colon x \to x \otimes x$ and $f_0\colon x \to I$ such that $jf = \Delta_{jx}; m_x^j$ and $jf_0 = (); e^j$. This is particularly useful in our case, since the functor $\mathcal{X} \to \mathbf{Set}$ is faithful, making $\mathcal{X}$ into a concrete category. Then the morphisms between two objects in $\mathcal{X}$ may be thought of as 'structure-preserving functions' between the corresponding sets. The criterion from Theorem 5.11.2 then says that the composites $\Delta_{jx}; m_x^j$ and $(); e^j$ are structure-preserving.

In other words, for every set A and every $p \in FA$, the function Δ_A must define a morphism

$$(A, p) \to (A \times A, m^F_{A,A}(p,p))\,\text{–}$$

i.e., we must have

$$F(\Delta_A)(p) = m^F_{A,A}(p,p)\,\text{–}$$

and the function $()$ must define a morphism

$$(A, p) \to (1, e^F)\,\text{–}$$

i.e., we must have

$$F(())(p) = e^F\,.$$

Let us see what this means in some examples.

Example 6.3.5. The finite powerset functor does not satisfy the condition given above; indeed, we have

$$\mathcal{P}_+(\Delta_A)(X) = \{(x,x) \in A \times A : \ x \in X\}$$

$$m^{\mathcal{P}_+}_{A,A}(X,X) = X \times X = \{(x,y) \in A \times A : \ x \in X\}\,.$$

Similarly, the discrete probability measure functor does not satisfy the condition we have given, since the diagonal map

$$\Delta_A \colon (A, \mathbb{P}_A) \to (A \times A, \mathbb{P}_{A\times A})$$

does not preserve probability in general.

Example 6.3.6. An alternative way of dealing with probability that does satisfy the condition for Cartesianness of the resulting category. If $(\Omega, \mathcal{F}, \mathbb{P})$ is a fixed probability space, then a *discrete random variable taking values in a set* A is a measurable function

$$V \colon (\Omega, \mathcal{F}) \to (A, \mathcal{P}A)\,.$$

Given $X \subseteq A$, we define

$$\mathbb{P}(V \in X) = \mathbb{P}(V^{-1}(X))\,.$$

This then gives us a discrete probability measure on X.

Suppose that $(\Omega, \mathcal{F}, \mathbb{P})$ is such that $\mathcal{F}$ contains all singleton sets (and hence all countable sets) and that if $Y \in \mathcal{F}$ has measure 0 (i.e., $\mathbb{P}(Y) = 0$) and $Z \subseteq Y$, then $Z \in \mathcal{F}$. Write $\mathrm{RV}_\Omega \colon \mathbf{Set} \to \mathbf{Set}$ for the functor that sends a set A to the set of all random variables taking values in the set A and sends a function $f \colon A \to B$ to the function

$$\mathrm{RV}_\Omega(A) \to \mathrm{RV}_\Omega(B)$$

given by composing on the right with f.

Moreover, RV_Ω is a lax monoidal functor: if V, W are discrete random variables taking values in sets A, B, then we can define

$$\langle V, W \rangle \colon \Omega \to A \times B$$

given by pairing.

Proposition 6.3.7. *$\langle V, W\rangle$ is a measurable function, so this is a random variable.*

Proof. Any discrete random variable is countably supported. This means that there are countable $\hat{A} \subseteq A$, $\hat{B} \subseteq B$ such that $V^{-1}(\hat{A})$ and $W^{-1}(\hat{B})$ are measurable and have measure 1. Now let $U \subseteq A \times B$. Write $\hat{U} = U \cap (\hat{A} \times \hat{B})$. Then

$$\begin{aligned}\langle V, W\rangle^{-1}(\hat{U}) &= \langle V, W\rangle^{-1}(U) \cap \langle V, W\rangle^{-1}(\hat{A} \times \hat{B}) \\ &= \langle V, W\rangle^{-1}(U) \cap V^{-1}(\hat{A}) \cap W^{-1}(\hat{B})\,,\end{aligned}$$

which is countable and therefore measurable, while

$$\langle V, W\rangle^{-1}(U \setminus \hat{U}) \subseteq (\Omega \setminus \hat{A}) \cup (\Omega \setminus \hat{B})\,,$$

which is measurable of measure 0, meaning that $\langle V, W\rangle^{-1}(U \setminus \hat{U})$ is measurable and has measure 0, by hypothesis. Therefore, $\langle V, W\rangle^{-1}(U)$ is the union of two measurable sets and is therefore measurable. □

Now if we apply the construction from Proposition 6.3.1, we get the category whose objects are pairs (A, V), where A is a set and V a discrete random variable taking values in A and where the morphisms $(A, V) \to (B, W)$ are functions $f\colon A \to B$ such that $W = V; f$. This category, which we will call $\mathbf{Rv}_\Omega$, admits a strict monoidal forgetful functor into the category of sets, giving us a lax reader action of $\mathbf{Rv}_\Omega^{\mathrm{op}}$ on $\mathbf{Set}$. Once again, we can pass to the symmetric monoidal category $\mathbf{Set}/\mathbf{Rv}_\Omega^{\mathrm{op}}$, giving us a way of modelling probability that is equivalent to taking base change through the functor RV_Ω.

The difference now is that the diagonal map

$$\Delta_A\colon (A, V) \to (A \times A, \langle V, V\rangle)$$

is probability preserving. In the language of probability, the two copies of the random variable V are *dependent random variables*, so the probability of obtaining the reading (v, v) from $\langle V, V\rangle$ is the same as the probability of obtaining the reading v from V. Since the terminal morphism from (A, V) to $(1, ())$ is also probability preserving, the category $\mathbf{Set}/\mathbf{Rv}_\Omega^{\mathrm{op}}$ will be Cartesian closed.

6.4 Actions of categories with terminal objects

The probability example above lends itself to further examination. Recall that if a monoidal category $\mathcal{X}$ acts on a category $\mathcal{C}$, and x is a monoid in $\mathcal{X}$

with multiplication m^x and unit e^x, then we get a monad on $\mathcal{C}$ given by the composite

$$1 \xrightarrow{x} \mathcal{X} \xrightarrow{-\cdot-} \mathrm{End}[\mathcal{C}]$$

of lax monoidal functors.

More explicitly, this monad is given by

$$M_x a = x.a\,,$$

with the monadic coherences given by

$$M_x M_x a = x.x.a \xrightarrow{m_{x,x}} (x \otimes x).a \xrightarrow{m^x.a} x.a = M_x a$$

$$a \xrightarrow{e_a} I.a \xrightarrow{e^x.a} x.a = M_x a\,.$$

If $j\colon \mathcal{X} \to \mathcal{C}$ is an oplax monoidal (in particular, comonoid-preserving) functor, yielding a reader action of $\mathcal{X}^{\mathrm{op}}$ upon $\mathcal{C}$, and x is a monoid in $\mathcal{X}^{\mathrm{op}}$ (i.e., a comonoid in $\mathcal{X}$), then the monad M_x will be the reader monad corresponding to jx.

Now suppose that the category $\mathcal{X}$ has a terminal object 1. Then 1 automatically has the structure of a monoid in $\mathcal{X}$, via the unique morphisms

$$1 \otimes 1 \xrightarrow{()} 1 \qquad\qquad I \xrightarrow{()} 1\,.$$

Proposition 6.4.1. *Let a monoidal category $\mathcal{X}$ act on a category $\mathcal{C}$ via a lax action. Suppose that $\mathcal{X}$ has a terminal object 1. Then the category $\mathcal{C}/\mathcal{X}$ is isomorphic to the Kleisli category for the monad M_1.*

Proof. In both cases, the objects are the objects of $\mathcal{C}$. Note also that we have an isomorphism

$$(\mathcal{C}/\mathcal{X})(a,b) = \int^x \mathcal{C}(a, x.b) \cong \int^x \mathcal{C}(a, x.b) \times \mathcal{C}(x, 1) \cong \mathcal{C}(a, 1.b) = \mathrm{Kl}_{M_1}(a,b)\,.$$

More concretely, this isomorphism sends a Melliès morphism $f\colon a \to x.b$ to the Kleisli morphism given by the composite

$$a \xrightarrow{f} x.b \xrightarrow{().b} 1.b\,.$$

By Proposition 5.4.4, this gives us a functor

$$\mathcal{C}/\mathcal{X} \to \mathrm{Kl}_{M_1}\,,$$

which is fully faithful and the identity on objects. □

Now we can get a clearer idea of what is going on in the probability example: the new category $\mathbf{Rv}_\Omega$ *almost* has an initial object (i.e., a terminal object in $\mathbf{Rv}_\Omega^{\mathrm{op}}$), given by the pair

$$(\Omega, \mathrm{id}_\Omega)\,.$$

In fact, if $\mathcal{F} = \mathcal{P}(\Omega)$ then this is indeed an initial object. However, if $\mathcal{F} \neq \mathcal{P}(\Omega)$, then the function

$$\mathrm{id}_\Omega \colon (\Omega, \mathcal{F}) \to (\Omega, \mathcal{P}\Omega)$$

is not measurable, and so id_Ω is not a discrete random variable.

If $\mathbf{Rv}_\Omega$ has an initial object $(\Omega, \mathrm{id}_\Omega)$, then the category

$$\mathbf{Set}/\mathbf{Rv}_\Omega^{\mathrm{op}}$$

is isomorphic to the Kleisli category for the reader monad for the set Ω on **Set**. Now, from Theorem 4.3.5, we know that this Kleisli category is automatically Cartesian closed.

What is going on in the general case is that the category $\mathbf{Rv}_\Omega$ is just close enough to having an initial object for the category $\mathbf{Set}/\mathbf{Rv}_\Omega^{\mathrm{op}}$ to be Cartesian, even though it is not a Kleisli category.

Remark 6.4.2. We could redo this whole probability example using the nonempty powerset functor, in an effort to model nondeterminism. To do that, we'd construct a new category whose objects were pairs (A, f), where $f \colon \Omega \to A$ is a function out of some fixed set Ω. However, in that case the resulting monoidal category would always have an initial object (namely, $(\Omega, \mathrm{id}_\Omega)$), and our resulting action would collapse to an ordinary monad, taking us back to Chapter 4.

6.5 Reader actions and denotational semantics

Suppose that we have an oplax monoidal functor $j \colon \mathcal{X} \to \mathbf{Set}$, giving rise to a lax reader action of $\mathcal{X}^{\mathrm{op}}$ on **Set**. Let $\mathcal{G}$ be a (Cartesian closed) model for a programming language P, and suppose that there is a functor $D \colon \mathbf{Set} \to \mathcal{G}$ that gives us the denotation of the datatypes in $\mathcal{G}$.

For example, if $\mathcal{G}$ is the category of games, then we can define $D(A)$ to be the game given by $M_{D(A)} = \{q\} \cup A$ with $\lambda(q) = O$ and $\lambda(a) = P$ for $a \in A$, where q is initial and justifies the moves $a \in A$ and the legal plays are those of the form ϵ, q or qa.

In general, since the source category **Set** and target category $\mathcal{G}$ are Cartesian, such functors automatically carry an oplax monoidal structure, given by

$$m_{A,B} = D(A \times B) \xrightarrow{\langle D(\mathrm{pr}_1), D(\mathrm{pr}_2)\rangle} D(A) \times D(B) \qquad e = D(1) \xrightarrow{()} 1\,.$$

Then we may compose the functors j and D to get a new oplax monoidal functor $\mathcal{X} \to \mathcal{G}$, inducing a reader action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{G}$.

Now the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ has its universal natural transformation

$$\phi_{x,A} \colon (Djx \to A) \to A \,,$$

which, by the enriched Yoneda lemma, may be equivalently given by the natural transformation

$$\omega_x = 1 \xrightarrow{u_{D(jx)}} (D(jx) \to D(jx)) \xrightarrow{\phi_{x,D(jx)}} D(jx) \,.$$

We can use this natural transformation to model primitives in the language of the form

$$\mathsf{choose}_x \colon jx \,,$$

for each object x of $\mathcal{X}$, where we use jx to refer to the datatype corresponding to the set jx.

The purpose of the final chapter will be to define a language with an operational semantics for which these primitives make sense, and prove a full abstraction result for it.

But first, an interlude.

Chapter 7

Promonads and parametric promonads

The purpose of this short chapter is to shine some light on the definition of the Melliès category for a parametric monad, showing why it is natural to think of it as being an analogue for the Kleisli category on a monad.

As a technical tool to prove the results we want, we shall introduce multicategories, which are a small generalization of monoidal categories. The main purpose of this generalization is to allow us to do without coends wherever possible: for example, while we need coends to make $[\mathcal{X}, \mathbf{Set}]$ into a monoidal category, we do not need them to make it into a multicategory.

The first half of this chapter is, in the interests of completeness, fairly technical, and may be skimmed over on a first reading. In Section 7.11, we introduce the multicategory of endoprofunctors on a category $\mathcal{C}$, which generalizes the monoidal category of endofunctors on $\mathcal{C}$. As monoids in $\mathrm{End}(\mathcal{C})$ are called monads on $\mathcal{C}$, so we will call monoids in $\mathrm{Endoprof}(\mathcal{C})$ *promonads* on $\mathcal{C}$. We will observe that a promonad may be regarded as a sort of category, and that the Kleisli category for a monad M is precisely the category we get by considering M as a promonad.

The main result will then be to show that an $\mathcal{X}$-parametric promonad on a category $\mathcal{C}$ – i.e., a multifunctor $\mathcal{X} \to \mathrm{Endoprof}(\mathcal{C})$ – may be regarded as a sort of $[\mathcal{X}, \mathbf{Set}]$-enriched category, and that the Melliès category for a parametric monad may similarly be regarded as the $[\mathcal{X}, \mathbf{Set}]$-enriched category that we get by regarding that parametric monad as a parametric promonad.

7.1 Multicategories

Definition 7.1.1 ([Lei04])**.** A *multicategory* $\mathcal{M}$ is given by a set of objects $\mathrm{Ob}(\mathcal{M})$ whose elements are called *objects* and, for each (possibly empty) list $a_1, \cdots, a_n$ of objects and each object b, a set

$$\mathcal{M}_n(a_1, \cdots, a_n; b)$$

whose elements are called the (n-ary) *multimorphisms* $a_1, \cdots, a_n \to b$.

Given collections $(a_{ij} \colon i = 1, \cdots, n, j = 1, \cdots, k_i), (b_i \colon i = 1, \cdots, n), c$ of objects and multimorphisms

$$f_i \colon a_{i1}, \cdots, a_{i,k_i} \to b_i \qquad\qquad g \colon b_1, \cdots, b_n \to c\,,$$

there is an operation that forms the *composition*

$$(f_1, \cdots, f_n); g \colon a_{11}, \cdots, a_{1k_1}, \cdots, a_{n1} \cdots, a_{nk_n} \to c\,.$$

Moreover, for each object a of $\mathcal{M}$, there is a distinguished multimorphism $\mathrm{id}_a \colon a \to a$ called the *identity* on a.

The composition and identity are subject to associativity and unitality conditions. Namely, let

$$\begin{pmatrix} & p = 1, \cdots, n \\ a_{pqr} \ : & q = 1, \cdots, k_p \\ & r = 1, \cdots, l_{pq} \end{pmatrix} \qquad \begin{pmatrix} b_{pq} \ : & p = 1, \cdots, n \\ & q = 1, \cdots, k_p \end{pmatrix}$$

$$(c_p \colon p = 1, \cdots, n) \qquad\qquad d$$

be collections of objects and let

$$f_{pq} \colon a_{pq1}, \cdots, a_{p,q,l_{pq}} \to b_{pq} \qquad g_p \colon b_{p1}, \cdots, b_{p,k_p} \to c_p \qquad h \colon c_1, \cdots, c_n \to d$$

be multimorphisms. Then we require that

$$\begin{gathered} (((f_{11}, \cdots, f_{1k_1}); g_1), \cdots, ((f_{n1}, \cdots, f_{n,k_n}); g_n)); h \\ = \\ (f_{11}, \cdots, f_{1k_1}, \cdots, f_{n1}, \cdots, f_{n,k_n}); ((g_1, \cdots, g_n); h)\,. \end{gathered}$$

Furthermore, we require that if $f \colon a_1, \cdots, a_n \to b$ is a multimorphism, then

$$(\mathrm{id}_{a_1}, \cdots, \mathrm{id}_{a_n}); f = f \qquad\qquad f = (f); \mathrm{id}_b\ .$$

Remark 7.1.2. We will use a slightly different form of commutative diagrams for multicategories, which should be fairly straightforward to understand. If,

for example, we say that the following diagram commutes,

$$\begin{array}{ccc} a_1, \cdots, a_n & \xrightarrow{f_1, \cdots, f_j} & b_1, \cdots, b_j \\ {\scriptstyle g_1, \cdots, g_k}\downarrow & & \downarrow{\scriptstyle h} \\ c_1, \cdots, c_k & \xrightarrow{\quad i \quad} & d \end{array}$$

where the arities of $f_1, \cdots, f_j$ sum to n, as do the arities of $g_1, \cdots, g_k$, then we mean that the composite $(f_1, \cdots, f_k); h$ is equal to the composite $(g_1, \cdots, g_k); i$. We leave it to the reader to extend this to more complicated diagrams.

Example 7.1.3. If $\mathcal{C}$ is an ordinary category, then $\mathcal{C}$ may be regarded as a multicategory $\hat{C}$ in which $\hat{C}_1(a; b) = \mathcal{C}(a, b)$ and $\hat{C}_n(a_1, \cdots, a_n; b) = \varnothing$ for $n \neq 1$. At the same time, if $\mathcal{M}$ is a multicategory, then it has an *underlying ordinary category* $\mathcal{M}_1$ whose morphisms are the morphisms in $\mathcal{M}$ with a single source object.

Example 7.1.4. If $\mathcal{M}$ is a *monoidal* category, then $\mathcal{M}$ may be regarded as a multicategory $\tilde{\mathcal{M}}$ with

$$\tilde{\mathcal{M}}_n(a_1, \cdots, a_n; b) = \mathcal{M}(a_1 \otimes \cdots \otimes a_n, b) \quad n \geq 1 \qquad \tilde{\mathcal{M}}_0(; b) = \mathcal{M}(I, b)$$

If we're being careful, then we should note that the expression $a_1 \otimes \cdots \otimes a_n$ does not define a single object of $\mathcal{M}$. Since the tensor product is not necessarily strictly associative, the choice of bracketing in the expression $a_1 \otimes \cdots \otimes a_n$ affects which object we end up with. It is enough to fix any one of the possible bracketings (e.g., to make $\_ \otimes \_$ always associate to the right).

Composition is then given by

$$\begin{array}{rcccccc} & a_{11} \otimes \cdots \otimes a_{1k_1} & \otimes & \cdots & \otimes & a_{n1} \otimes \cdots \otimes a_{nk_n} \\ \longrightarrow & (a_{11} \otimes \cdots \otimes a_{1k_1}) & \otimes & \cdots & \otimes & (a_{n1} \otimes \cdots \otimes a_{nk_n}) \\ \xrightarrow{f_1 \otimes \cdots \otimes f_n} & b_1 & \otimes & \cdots & \otimes & b_n \\ \xrightarrow{\quad g \quad} & & & c\,, & & \end{array}$$

where the first arrow is induced from the normal monoidal coherences (exactly which ones depends on how we choose to interpret the iterated tensor product).

7.2 Representable multicategories

We call a multicategory *representable* if it isomorphic to a multicategory that arises from a monoidal category as in Example 7.1.4. The next theorem gives a criterion for a multicategory to be representable.

Theorem 7.2.1 ([Her00])**.** *Let $\mathcal{M}$ be a multicategory and suppose that for each natural number n and each sequence $a_1, \cdots, a_n$ of objects of $\mathcal{M}$ there is an object $\otimes\vec{a}$ and a multimorphism*

$$\pi_{\vec{a}} \colon a_1, \cdots, a_n \to \otimes\vec{a}\,.$$

Suppose that the $\pi_{\vec{a}}$ are strongly universal *in the sense that if*

$$b_1, \cdots b_k, c_1 \qquad\qquad \cdots, c_l$$

are two (possibly empty) lists of objects, and d is an object, then any multimorphism

$$f \colon b_1, \cdots, b_k, a_1, \cdots, a_n, c_1, \cdots, c_l \to d$$

factors uniquely through $\pi_{\vec{a}}$; i.e., there is a unique morphism

$$\hat{f} \colon b_1, \cdots, b_k, \otimes\vec{a}, c_1, \cdots, c_l \to d$$

such that

$$f = (\mathrm{id}_{b_1}, \cdots, \mathrm{id}_{b_k}, \pi_{\vec{a}}, \mathrm{id}_{c_1}, \cdots, \mathrm{id}_{c_l}); \hat{f}\,.$$

Given objects a, b of $\mathcal{M}$, define $a \otimes b = \otimes a, b$, and let I be the object $\otimes\epsilon$, where ϵ is the empty list. Then the operation $\_ \otimes \_$ and I make $\mathcal{M}$ into a monoidal category $\overline{\mathcal{M}}$ such that $\overline{\widetilde{\mathcal{M}}}$ and $\mathcal{M}$ are isomorphic multicategories.

- *$\_ \otimes \_$ and I are the monoidal product and unit of a monoidal category on $\mathcal{M}_1$, the underlying category of $\mathcal{M}$.*
- *For any sequence $a_1, \cdots a_n$ of objects of $\mathcal{M}$ there is a canonical isomorphism*

  $$a_1 \otimes \cdots \otimes a_n \cong \otimes\vec{a}\,,$$

  for any bracketing of the left hand side, and the associators and unitors in $\mathcal{M}_1$ are induced from these isomorphisms.
- *The set of multimorphisms $a_1, \cdots, a_n \to b$ is in bijection with the set of morphisms $a_1 \otimes \cdots \otimes a_n \to b$ for $n \geq 1$, and the set of multimorphisms $\to b$ is in bijection with the set of morphisms $I \to b$, and these bijections commute with the multicategory composition in $\mathcal{M}$ and the composition in $\mathcal{M}_1$.*

Definition 7.2.2. A *symmetric multicategory* is a multicategory $\mathcal{M}$ together with an action of the symmetric group on the sets $\mathcal{M}_n(a_1, \cdots, a_n; b)$ that respects composition. In other words, for each natural number n, each multimorphism $f \colon a_1, \cdots, a_n \to b$ and each permutation σ of $\{1, \cdots, n\}$ there is a multimorphism

$$\sigma_* f \colon a_{\sigma(1))}, \cdots, a_{\sigma(n)} \to b$$

such that if $(a_{ij}\colon i = 1, \cdots, n), (b_i\colon i = 1, \cdots, n)$ are objects,

$$f_i\colon a_{i1}, \cdots, a_{i,k_i} \to b_i \qquad\qquad g\colon b_1, \cdots, b_n \to c$$

are multimorphisms, σ_i is a permutation of $\{1, \cdots, k_i\}$, and τ is a permutation of $\{1, \cdots, n\}$, then

$$(\sigma_{1*} f_q, \cdots, \sigma_{n*} f_n); (\tau_* g) = (\tau * (\sigma_1, \cdots, \sigma_n))_*((f_1, \cdots, f_n); g)\,,$$

where $\tau * (\sigma_1, \cdots, \sigma_n)$ is the permutation of

$$\{(1,1), \cdots, (1,k_1), \cdots, (n,1), \cdots, (n,k_n)\}$$

that sends (i,j) to $(\tau(i), \sigma_i(j))$.

Moreover, we require that for any morphism $f\colon a_1, \cdots, a_n \to b$ and permutations σ, τ of $\{1, \cdots, n\}$ we have

$$\sigma_* \tau_* f = (\sigma \circ \tau)_* f \qquad\qquad \mathrm{id}_* f = f$$

Example 7.2.3. Any multicategory arising from an ordinary category is symmetric.

Example 7.2.4. A monoidal category is a symmetric multicategory if and only if it is a symmetric monoidal category.

7.3 Product and unit multicategories

Definition 7.3.1. Let $\mathcal{M}, \mathcal{N}$ be multicategories. The *product multicategory* $\mathcal{M} \times \mathcal{N}$ has, as objects, pairs (a,b), where a is an object of $\mathcal{M}$ and b an object of $\mathcal{N}$. The multimorphisms are given by

$$(\mathcal{M} \times \mathcal{N})_n((a_1,b_1), \cdots, (a_n,b_n); (c,d)) =$$

$$\mathcal{M}_n(a_1, \cdots, a_n; c) \times \mathcal{N}_n(b_1, \cdots, b_n; d)\,.$$

Composition and the identity are similarly defined pointwise.

Definition 7.3.2. The *unit multicategory* 1 has a single object I, and for each n, the set

$$1_n(I, \cdots, I; I)$$

is a singleton.

This is a representable multicategory; indeed, it may be identified with the usual unit monoidal category.

7.4 Multifunctors & multinatural transformations

Definition 7.4.1. Let $\mathcal{M}, \mathcal{N}$ be multicategories. A *multifunctor* from $\mathcal{M}$ to $\mathcal{N}$ is a map F from the objects of $\mathcal{M}$ to the objects of $\mathcal{N}$ together with, for each list $a_1, \cdots, a_n, b$ of objects of $\mathcal{M}$, a function

$$\mathcal{M}_n(a_1, \cdots, a_n; b) \to \mathcal{N}_n(Fa_1, \cdots, Fa_n; Fb)$$

that commutes with the composition operator.

Definition 7.4.2. Given multicategories $\mathcal{M}, \mathcal{N}$ and multifunctors

$$F, G \colon \mathcal{M} \to \mathcal{N},$$

a *multinatural transformation* $\phi \colon F \Rightarrow G$ is given by morphisms

$$\phi_a \colon Fa \to Ga$$

for each object a of $\mathcal{M}$, such that if $f \colon a_1, \cdots a_n \to b$ is any morphism in $\mathcal{M}$, then the following diagram commutes.

$$\begin{array}{ccc} Fa_1, \cdots, Fa_n & \xrightarrow{\phi_{a_1}, \cdots, \phi_{a_n}} & Ga_1, \cdots, Ga_n \\ {\scriptstyle Ff}\downarrow & & \downarrow{\scriptstyle Gf} \\ Fb & \xrightarrow{\phi_b} & Gb \end{array}$$

Proposition 7.4.3. *If $\mathcal{M}, \mathcal{N}$ are monoidal categories, considered as multicategories, then multifunctors $\mathcal{M} \to \mathcal{N}$ are the same thing as lax monoidal functors. Multinatural transformations are the same thing as monoidal natural transformations.*

Definition 7.4.4. Let $\mathcal{M}, \mathcal{N}$ be multicategories, where $\mathcal{M}$ is symmetric. Then the collection of multifunctors $\mathcal{M} \to \mathcal{N}$ forms a multicategory. A multimorphism $F_1, \cdots, F_n \Rightarrow G$, where $F_1, \cdots, F_n, G$ are multifunctors $\mathcal{M} \to \mathcal{N}$, is given by a family

$$\phi_a \colon F_1(a), \cdots, F_n(a) \to G(a)$$

such that for any multimorphism $f \colon a_1, \cdots, a_m \to b$ in $\mathcal{M}$, the diagram

$$\begin{array}{ccc} F_1(a_1), \cdots, F_n(a_1), \cdots, F_1(a_m), \cdots, F_n(a_m) & \xrightarrow{\phi_{a_1}, \cdots, \phi_{a_m}} & G(a_1), \cdots, G(a_m) \\ {\scriptstyle \sigma_*}\downarrow & & \\ F_1(a_1), \cdots, F_1(a_m), \cdots, F_n(a_1), \cdots, F_n(a_m) & & \downarrow{\scriptstyle Gf} \\ {\scriptstyle F_1 f, \cdots, F_n f}\downarrow & & \\ F_1(b), \cdots, F_n(b) & \xrightarrow{\phi_b} & G(b) \end{array}$$

commutes, where σ is the map

$$(1,1),\cdots,(n,1),\cdots,(1,m),\cdots,(n,m) \to$$

$$(1,1),\cdots,(1,m),\cdots,(n,1),\cdots,(n,m)$$

sending (i,j) to (i,j), considered as a permutation of $\{1,\cdots,mn\}$.

7.5 Monoids in multicategories

Definition 7.5.1. Let $\mathcal{M}$ be a multicategory.

Then a *monoid* in $\mathcal{M}$ is an object a of $\mathcal{M}$ together with multimorphisms

$$m\colon a,a \to a \qquad\qquad e\colon \ \to a$$

satisfying the following associativity and unitality laws.

$$\begin{array}{ccc} a,a,a & \xrightarrow{m,\mathrm{id}_a} & a,a \\ {\scriptstyle \mathrm{id}_a,m}\downarrow & & \downarrow{\scriptstyle m} \\ a,a & \xrightarrow{m} & a \end{array} \qquad \begin{array}{ccc} a & \xrightarrow{\mathrm{id}_a} & a \\ {\scriptstyle e_a,\mathrm{id}_a}\downarrow & \nearrow{\scriptstyle m} & \\ a,a & & \end{array} \qquad \begin{array}{ccc} a & \xrightarrow{\mathrm{id}_a} & a \\ {\scriptstyle \mathrm{id}_a,e_a}\downarrow & \nearrow{\scriptstyle m} & \\ a,a & & \end{array}$$

Note that a monoid in a multicategory $\mathcal{M}$ may equivalently be defined as a multifunctor $1 \to \mathcal{M}$ [Lei04, 2.1.11].

7.6 Categories enriched over multicategories

Definition 7.6.1. Let $\mathcal{V}$ be a multicategory. Then a *V-enriched category* $\mathcal{C}$ is given by a collection $\mathrm{Ob}(\mathcal{C})$ of objects together with, for each pair a,b of objects, an object

$$\mathcal{C}(a,b)$$

of $\mathcal{V}$ and, for objects a,b,c of $\mathcal{C}$, composition and identity multimorphisms

$$;_{a,b,c} : \mathcal{C}(a,b),\mathcal{C}(b,c) \to \mathcal{C}(a,c) \qquad\qquad \eta_a\colon \ \to \mathcal{C}(a,a)$$

that satisfy the following associativity and unitality laws for all objects a,b,c,d of $\mathcal{C}$.

$$\begin{array}{ccc} \mathcal{C}(a,b),\mathcal{C}(b,c),\mathcal{C}(c,d) & \xrightarrow{;_{a,b,c},\mathrm{id}_{\mathcal{C}(c,d)}} & \mathcal{C}(a,c),\mathcal{C}(c,d) \\ {\scriptstyle \mathrm{id}_{\mathcal{C}(a,b)},;_{b,c,d}}\downarrow & & \downarrow{\scriptstyle ;_{a,c,d}} \\ \mathcal{C}(a,b),\mathcal{C}(b,d) & \xrightarrow{;_{a,b,d}} & \mathcal{C}(a,d) \end{array}$$

$$\begin{array}{ccc} \mathcal{C}(a,b) & \xrightarrow{\mathrm{id}_{\mathcal{C}(a,b)}} & \mathcal{C}(a,b) \\ {\scriptstyle \eta_a,\mathrm{id}_{\mathcal{C}(a,b)}}\downarrow & \nearrow{\scriptstyle ;_{a,a,b}} & \\ \mathcal{C}(a,a),\mathcal{C}(a,b) & & \end{array} \qquad \begin{array}{ccc} \mathcal{C}(a,b) & \xrightarrow{\mathrm{id}_{\mathcal{C}(a,b)}} & \mathcal{C}(a,b) \\ {\scriptstyle \mathrm{id}_{\mathcal{C}(a,b)},\eta_b}\downarrow & \nearrow{\scriptstyle ;_{a,b,b}} & \\ \mathcal{C}(a,b),\mathcal{C}(b,b) & & \end{array}$$

Remark 7.6.2. These definitions clearly generalize the same definitions for categories enriched over a monoidal category.

In particular, a monoid in a multicategory $\mathcal{M}$ is the same thing as an $\mathcal{M}$-enriched category with a single object.

Remark 7.6.3. If $\mathcal{V}$ is a symmetric multicategory and $\mathcal{C}$ is a $\mathcal{V}$-enriched category, then we may define the *opposite category* $\mathcal{C}^{\mathrm{op}}$ whose objects are the objects of $\mathcal{C}$ and where

$$\mathcal{C}^{\mathrm{op}}(a,b) = \mathcal{C}(b,a)\,.$$

Composition is defined by

$$\mathcal{C}(b,a), \mathcal{C}(c,b) \xrightarrow{\tau_*} \mathcal{C}(c,b), \mathcal{C}(b,a) \xrightarrow{;_{c,b,a}} \mathcal{C}(c,a)\,,$$

where τ is the permutation that transposes the two values.

7.7 Multicategory-enriched functors and natural transformations

Definition 7.7.1. Let $\mathcal{C}, \mathcal{D}$ be categories enriched over some multicategory $\mathcal{V}$. An *$\mathcal{V}$-enriched functor* $F\colon \mathcal{C} \to \mathcal{D}$ is a map F from the objects of $\mathcal{C}$ to the objects of $\mathcal{D}$ together with, for each, pair a, b of objects of $\mathcal{C}$, a (unary) multimorphism

$$F\colon \mathcal{C}(a,b) \to \mathcal{D}(F(a), F(b))$$

such that for all a, b, c the following diagrams commute.

$$\begin{array}{ccc} \mathcal{C}(a,b), \mathcal{C}(b,c) & \xrightarrow{\quad ;_{a,b,c} \quad} & \mathcal{C}(a,c) \\ {\scriptstyle F,F}\downarrow & & \downarrow{\scriptstyle F} \\ \mathcal{D}(F(a),F(b)), \mathcal{D}(F(b),F(c)) & \xrightarrow{;_{F(a),F(b),F(c)}} & \mathcal{D}(F(a),F(c)) \end{array}$$

$$\begin{array}{ccc} & \xrightarrow{\eta_a} & \mathcal{C}(a,a) \\ & \searrow^{\eta_{F(a)}} & \downarrow{\scriptstyle F} \\ & & \mathcal{D}(F(a),F(a)) \end{array}$$

Definition 7.7.2. Let $\mathcal{C}, \mathcal{D}$ be categories enriched over a multicategory $\mathcal{V}$ and let $F, G\colon \mathcal{C} \to \mathcal{D}$ be $\mathcal{V}$-enriched functors. An *$\mathcal{V}$-enriched natural transformation* $\phi\colon F \Rightarrow G$ is given by a family of 0-ary multimorphisms

$$\phi_a\colon \ \to \mathcal{D}(F(a), G(a))$$

such that for all objects a, b the following diagram commutes.

$$\begin{array}{ccc} \mathcal{C}(a,b) & \xrightarrow{F,\phi_b} & \mathcal{D}(F(a),F(b)),\mathcal{D}(F(b),G(b)) \\ {\scriptstyle \phi_a,G}\downarrow & & \downarrow{\scriptstyle ;_{F(a),F(b),G(b)}} \\ \mathcal{D}(F(a),G(a)),\mathcal{D}(G(a),G(b)) & \xrightarrow{;_{F(a),G(a),G(b)}} & \mathcal{D}(F(a),G(b)) \end{array}$$

7.8 The categories enriched over a symmetric multicategory form a multicategory

Definition 7.8.1. Let $\mathcal{V}$ be a symmetric multicategory. Given $\mathcal{V}$-enriched categories $\mathcal{C}_1, \cdots, \mathcal{C}_n, \mathcal{D}$, a multimorphism

$$F\colon \mathcal{C}_1, \cdots, \mathcal{C}_n \to \mathcal{D}$$

is given by a function

$$F\colon \mathrm{Ob}(\mathcal{C}_1) \times \cdots \times \mathrm{Ob}(\mathcal{C}_n) \to \mathrm{Ob}(\mathcal{D})$$

together with, for each $a_i, b_i \in \mathrm{Ob}(\mathcal{C}_i)$, a multimorphism

$$F\colon \mathcal{C}_1(a_1,b_1), \cdots, \mathcal{C}_n(a_n,b_n) \to \mathcal{D}(F(a_1, \cdots, a_n), F(b_1, \cdots, b_n))\,,$$

such that the diagrams in Figure 7.1 commute.

In the case $n = 1$, this is the same thing as a $\mathcal{V}$-enriched functor from $\mathcal{C}_1$ to $\mathcal{D}$.

7.9 Change of base

Let $\mathcal{M}, \mathcal{N}$ be multicategories, let $F\colon \mathcal{M} \to \mathcal{N}$ be a multifunctor and let $\mathcal{C}$ be an $\mathcal{M}$-enriched category. Then we can form an $\mathcal{N}$-enriched category $F_*\mathcal{C}$ whose objects are the objects of $\mathcal{C}$ and where the morphisms are given by the formula

$$F_*\mathcal{C}(a,b) = F(\mathcal{C}(a,b))\,.$$

We get composition and identities by applying the multifunctor F to the composition and identity multimorphisms in $\mathcal{C}$. By functoriality of F, these composition and identities are associative and unital, meaning that $F_*\mathcal{C}$ is indeed an $\mathcal{N}$-enriched category.

This process is called *base change along* F.

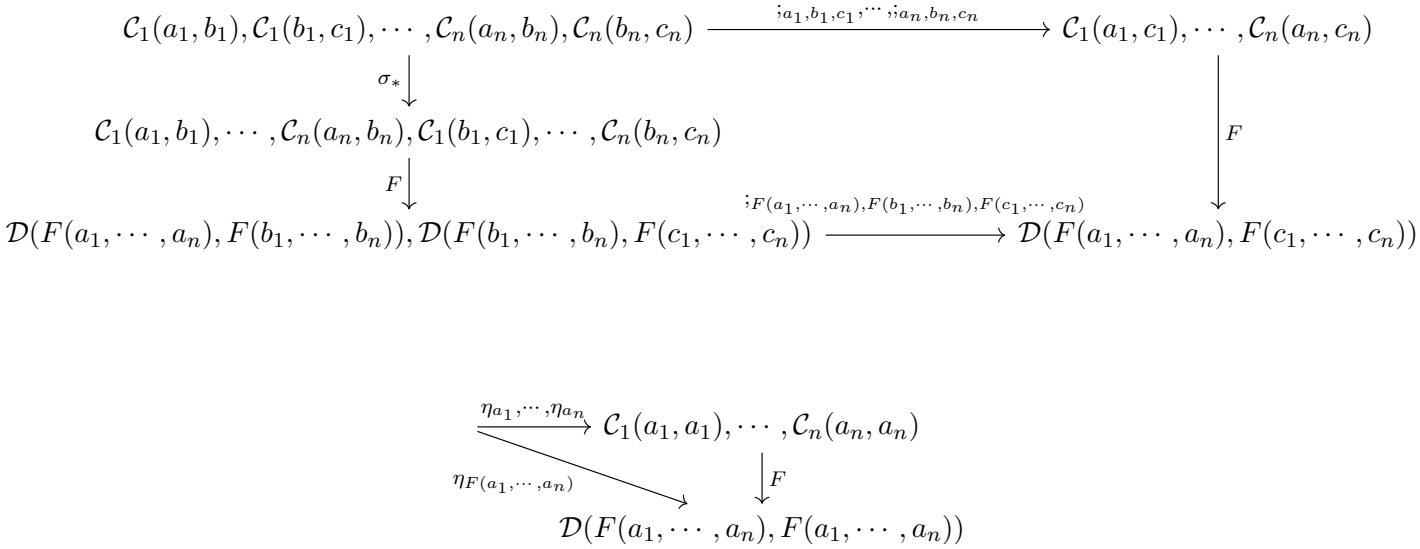


Figure 7.1: The rules for preservation of composition and identity by multimorphisms of $\mathcal{V}$-enriched functors are similar to those for ordinary enriched functors. Note that it is essential for the $\mathcal{V}$ to be a symmetric multicategory. This generalizes the usual construction for categories enriched over a symmetric monoidal category.

7.10 Closed multicategories

Definition 7.10.1 ([Man12])**.** We say that a multicategory $\mathcal{M}$ is *closed* if for any pair a, c of objects, there exists an object

$$\underline{\mathcal{M}}(a, c)$$

and a multimorphism

$$\mathrm{ev}_{a,c} \colon a, \underline{\mathcal{M}}(a, c) \to c$$

such that for any sequence $b_1, \cdots, b_n$ of objects of $\mathcal{M}$, the function

$$\begin{array}{rccc} \kappa_{a,b_1,\cdots,b_n,c} \colon & \mathcal{M}_n(b_1, \cdots, b_n; \underline{\mathcal{M}}(a, c)) & \to & \mathcal{M}_{n+1}(a, b_1, \cdots, b_n; c) \\ & f & \mapsto & (\mathrm{id}_a, f); \mathrm{ev}_{a,c} \end{array}$$

is a bijection.

Proposition 7.10.2 ([Man12])**.** *If $\mathcal{V}$ is a closed multicategory, then $\mathcal{V}$ gives rise to the structure of a $\mathcal{V}$-enriched category on the underlying category $\mathcal{V}_1$ of $\mathcal{V}$. We will also call this category $\mathcal{V}_1$, relying on context to distinguish the two. The objects of $\mathcal{V}_1$ are the objects of $\mathcal{V}$, while the morphisms are given by*

$$\mathcal{V}_1(a, b) = \underline{\mathcal{V}}(a, b)\,.$$

If $\mathcal{V}$ is a closed multicategory and $\mathcal{C}_1, \cdots, \mathcal{C}_n$ are $\mathcal{V}$-enriched categories, then a functor $\mathcal{C}_1, \cdots, \mathcal{C}_n \to \mathcal{V}$ is given by a map $\mathrm{Ob}(\mathcal{C}_1) \times \cdots \times \mathrm{Ob}(\mathcal{C}_n) \to \mathrm{Ob}(\mathcal{V})$ and, for each $a_i, b_i \in \mathrm{Ob}(\mathcal{C}_i)$, a multimorphism

$$\mathcal{C}_1(a_1, b_1), \cdots, \mathcal{C}_n(a_n, b_n) \to \underline{\mathcal{V}}(F(a_1, \cdots, a_n), F(b_1, \cdots, b_n))$$

By the definition of a closed multicategory, this is equivalent to providing a multimorphism

$$F(a_1, \cdots, a_n), \mathcal{C}_1(a_1, b_1), \cdots, \mathcal{C}_n(a_n, b_n) \to F(b_1, \cdots, b_n)\,.$$

In what follows, we will denote these multimorphisms (and their various permutations) with the letter p.

We have seen so far that multicategories provide us with a rather straightforward generalization of monoidal categories. We might ask the question, then: why make this generalization?

To answer this question, we introduce some natural multicategories that are not representable.

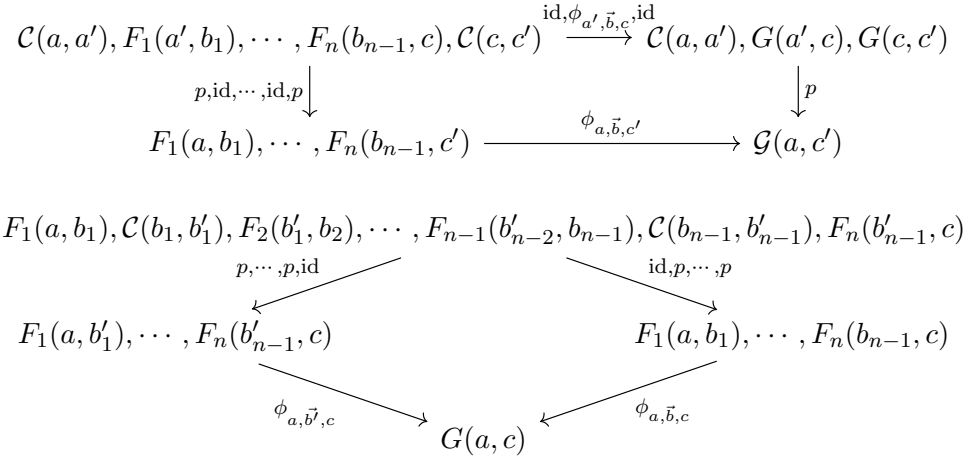


Figure 7.2: Extranatural transformations between endoprofunctors. The coherences we require on the multimorphisms between endoprofunctors are essentially the axioms for an extranatural transformation as in [EK66].

7.11 The multicategory of endoprofunctors

Let $\mathcal{C}, \mathcal{D}$ be ordinary categories. Recall that a *profunctor* $F\colon \mathcal{C} \nrightarrow \mathcal{D}$ is an ordinary functor $\mathcal{C}^{\mathrm{op}} \times \mathcal{D} \to \mathbf{Set}$.

More generally, if $\mathcal{C}, \mathcal{D}$ are enriched over some symmetric closed multicategory $\mathcal{V}$, then a $\mathcal{V}$-enriched profunctor $F\colon \mathcal{C} \nrightarrow \mathcal{D}$ is a $\mathcal{V}$-enriched functor $\mathcal{C}^{\mathrm{op}} \times \mathcal{D} \to \mathcal{V}_1$.

Let $F_1, \cdots, F_n, G\colon \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathcal{V}_1$ be $\mathcal{V}$-enriched profunctors $\mathcal{C} \nrightarrow \mathcal{C}$, where $\mathcal{C}$ is a $\mathcal{V}$-enriched category.

We then define a multimorphism $\phi\colon F_1, \cdots, F_n \Rightarrow G$ to be given by a family of multimorphisms

$$\phi_{a,b_1,\cdots,b_{n-1},c}\colon F_1(a,b_1), \cdots, F_n(b_{n-1},c) \to G(a,c)$$

that make the diagrams in Figure 7.2 commute.

A 0-ary multimorphism $\to G$ is an ordinary enriched natural transformation $\mathcal{C}(a,c) \to G(a,c)$.

We say that $\phi_{a,b_1,\cdots,b_{n-1},c}$ is *natural* in a and c and *extranatural* in the b_i.

We will often drop the component objects from ϕ and from the profunctors in question where they can be inferred from context.

We compose these multimorphisms pointwise. The following proposition shows that this is indeed a well-defined composition.

Proposition 7.11.1. *Let $\mathcal{V}$ be a symmetric closed multicategory and let $\mathcal{C}$ be a $\mathcal{V}$-enriched category. Let $F_1, \cdots F_n, G_1, \cdots, G_m, H$ be profunctors $\mathcal{C} \nrightarrow \mathcal{C}$, and let $0 = k_0, \cdots, k_m = n$ be a (not necessarily strictly) increasing subsequence of $\{0, \cdots, n\}$. Let $\phi^{(i)} \colon F_{k_i+1}, \cdots, F_{k_{i+1}} \to \mathcal{G}_i, \psi \colon G_1, \cdots, G_m \to G$ be multimorphisms of profunctors.*

Then the family of multimorphisms

$$F_1, \cdots, F_n \xrightarrow{\phi^{(1)}, \cdots, \phi^{(m)}} G_1, \cdots, G_m \xrightarrow{\psi} H$$

forms a multimorphism $F_1, \cdots, F_n \to H$.

Proof. For the first condition (naturality), we have

$$\begin{array}{ccccc} \mathcal{C}, F_1, \cdots, F_n, \mathcal{C} & \xrightarrow{\mathrm{id}, \phi^{(1)}, \cdots, \phi^{(m)}, \mathrm{id}} & \mathcal{C}, \mathcal{G}_1, \cdots, \mathcal{G}_m, \mathcal{C} & \xrightarrow{\mathrm{id}, \psi, \mathrm{id}} & \mathcal{C}, H, \mathcal{C} \\ \downarrow{\scriptstyle p, \mathrm{id}, \cdots, \mathrm{id}, p} & & \downarrow{\scriptstyle p, \mathrm{id}, \cdots, \mathrm{id}, p} & & \downarrow{\scriptstyle p} \\ F_1, \cdots, F_n & \xrightarrow{\phi^{(1)}, \cdots, \phi^{(m)}} & G_1, \cdots, G_m & \xrightarrow{\psi} & H \end{array},$$

where commutativity of the left hand square is the naturality condition on $\phi^{(1)}$ and $\phi^{(m)}$, while commutativity of the right hand square is the naturality condition for ψ.

For the second condition (extranaturality), see Figure 7.3. □

This composition is associative, because it is given pointwise by composition in $\mathcal{V}$, and its unit is given by the identity natural transformation. This gives us a multicategory.

Suppose that $\mathcal{V}$ is the category of sets, so that the multimorphisms

$$F_1, \cdots, F_n \to G$$

are ordinary extranatural transformations

$$\phi_{a, \vec{b}, c} \colon F_1(a, b_1), \cdots, F_n(b_{n-1}, c) \to G(a, c)\,.$$

Then the definition of the *coend*

$$\int_{b_1, \cdots, b_{n-1} \colon \mathcal{C}} F_1(a, b_1) \times \cdots \times F_n(b_{n-1}, c)$$

is that it is universal among all objects admitting such an extranatural transformation out of them. It follows that in this case (and more generally, if $\mathcal{V}$ is a cocomplete monoidal category), that the multicategory of endoprofunctors on $\mathcal{C}$ is representable, with monoidal product given by

$$F \otimes G(a, c) = \int_{b \colon \mathcal{C}} F(a, b) \times G(b, c)\,.$$

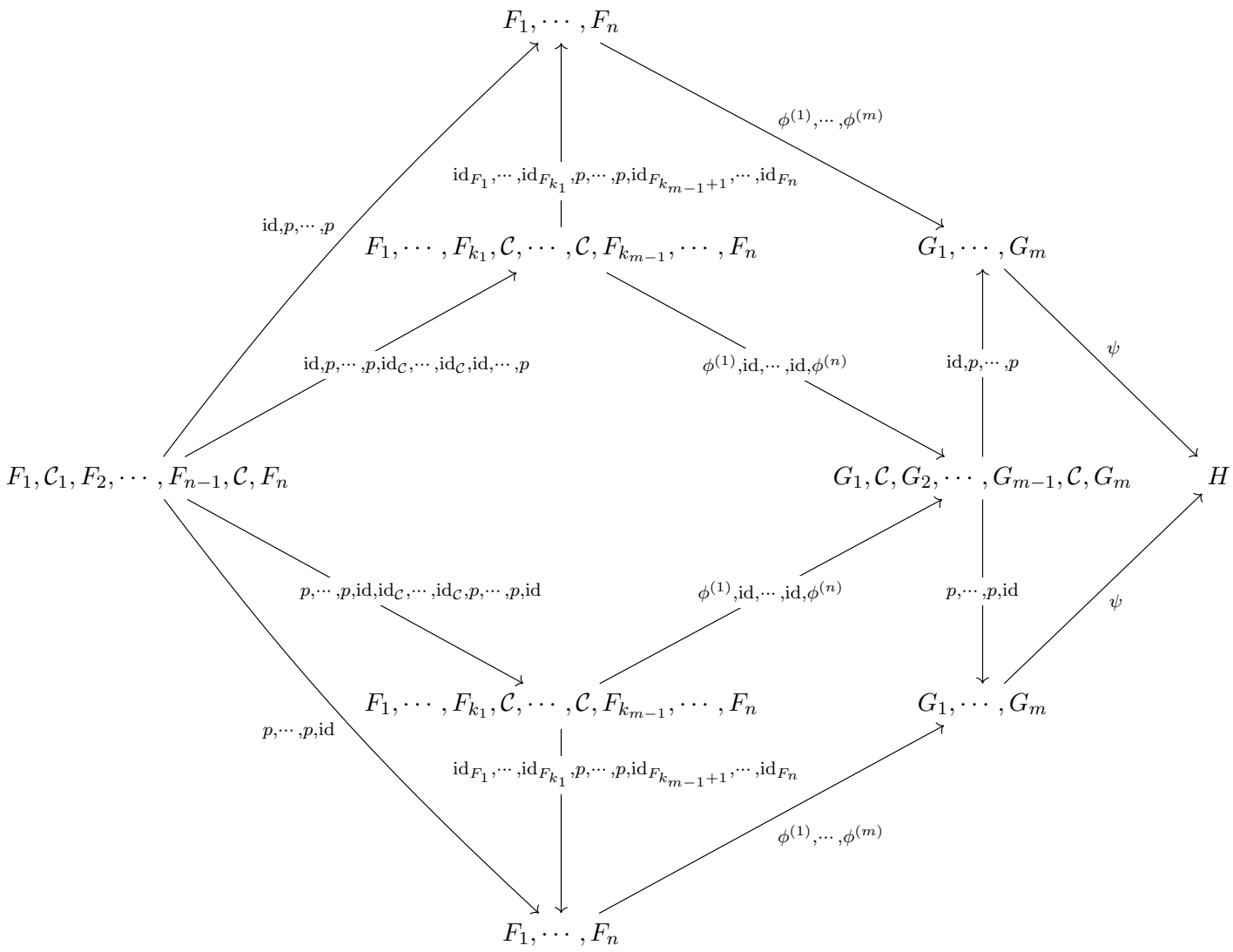


Figure 7.3: Proof that extranaturality is preserved by composition. Commutativity of the central square is by extranaturality of the $\phi^{(i)}$, while that of the four-cornered triangle at the right is by extranaturality of ψ. The triangles on the left commute automatically, while the parallelograms at the top and the bottom commute by naturality of the $\phi^{(i)}$.

This is the usual notion of composition for **Set**-enriched profunctors. However, it relies on the appropriate coends existing in **Set**; for more generally $\mathcal{V}$-enriched profunctors, the multicategory of endoprofunctors on $\mathcal{C}$ need not be representable, even if $\mathcal{V}$ is a monoidal category.

We have only considered profunctors going from a category into itself. Profunctors in general form a structure called an *fc-multicategory* [Lei04], but we shall not be using this notion.

7.12 Functors are a special case of profunctors

The reason why we refer to a functors $F\colon \mathcal{C}^{\mathrm{op}}, \mathcal{D} \to \mathcal{V}$ as a *profunctors* $\mathcal{C} \nrightarrow \mathcal{D}$ is that they generalize ordinary functors. Specifically, if $F\colon \mathcal{D} \to \mathcal{C}$ is a functor, then we can identify it with the profunctor

$$\tilde{F}(c,d) = \mathcal{C}(c, F(d))\colon \mathcal{C}^{\mathrm{op}}, \mathcal{D} \to \mathcal{V}\,.$$

This gives us an embedding of the monoidal category of endofunctors $\mathcal{C} \to \mathcal{C}$ into the multicategory of endoprofunctors $\mathcal{C} \nrightarrow \mathcal{C}$:

Proposition 7.12.1. *Let $\mathcal{V}$ be a closed symmetric multicategory and let $\mathcal{C}$ be a $\mathcal{V}$-enriched category. Let $F_1, \cdots, F_n, G\colon \mathcal{C} \to \mathcal{C}$ be functors. Then the set of natural transformations $F_1 \circ \cdots \circ F_n \to G$ is naturally in bijection with the set of extranatural transformations*

$$\tilde{F_1}, \cdots, \tilde{F_n} \to \tilde{G}\,.$$

Proof. We have a natural multimorphism

$$\begin{aligned} &\mathcal{C}(a, F_1(b_1)), \cdots, \mathcal{C}(b_{n-1}, F_n(c)) \\ \xrightarrow{\mathrm{id},\cdots,F_1\circ\cdots\circ F_{n-1}}\ &\mathcal{C}(a, F_1(b_1)), \cdots, \mathcal{C}(F_1 \circ \cdots \circ F_{n-1}(b_{n-1}), F_1 \circ \cdots \circ F_n(c)) \\ \xrightarrow{\quad ;^*\quad}\ &\mathcal{C}(a, F_1 \circ \cdots \circ F_n(c))\,, \end{aligned}$$

which is natural in a, c and extranatural in the b_i.

If $\phi\colon F_1 \circ \cdots \circ F_n \to G$ is a natural transformation, then it gives rise (via postcomposition) to a natural transformation

$$\mathcal{C}(a, F_1(\cdots(F_n(c))\cdots)) \to \mathcal{C}(a, G(c))\,,$$

which we can compose with the multimorphism above to get the required extranatural transformation

$$\mathcal{C}(a, F_1(b_1)), \cdots, \mathcal{C}(b_{n-1}, F_n(c)) \to \mathcal{C}(a, G(c))\,.$$

In the other direction, suppose that we have some extranatural transformation

$$\phi_{a,\vec{b},c} \colon \mathcal{C}(a, F_1(b_1)), \cdots, \mathcal{C}(b_{n-1}, F_n(c)) \to \mathcal{C}(a, G(c)) \,.$$

Then we can take components of the form

$$\phi_{a,F_1\circ\cdots\circ F_n(c),\cdots,F_n(c),c} \colon \mathcal{C}(a, F_1 \circ \cdots \circ F_n(c)), \cdots, \mathcal{C}(F_n(c), F_n(c)) \to \mathcal{C}(a, G(c))$$

and compose with $\mathrm{id}, \eta, \cdots, \eta$ to get our natural transformation

$$\mathcal{C}(a, F_a \circ \cdots \circ F_n(c)) \to \mathcal{C}(a, \mathcal{G}(c)) \,.$$

It is easy to check that these two constructions are inverses and that they respect composition of natural transformations. □

7.13 Promonads are categories

Since a monad was defined to be a monoid in the category of endofunctors on a category $\mathcal{C}$, we can define a *promonad* to be a monoid in the multicategory of endoprofunctors on $\mathcal{C}$.

Proposition 7.13.1 (See, e.g., [SG12])**.** *Let $\mathcal{V}$ be a symmetric closed multicategory. Let $\mathcal{C}$ be a $\mathcal{V}$-enriched category. Then a promonad $\mathcal{D} \colon \mathcal{C} \nrightarrow \mathcal{C}$ is the same thing as a $\mathcal{V}$-enriched category $\mathcal{D}$ together with an identity-on-objects functor $j \colon \mathcal{C} \to \mathcal{D}$.*

Proof. This is a matter of unwrapping the definitions.

Let $\mathcal{D} \colon \mathcal{C} \nrightarrow \mathcal{C}$ be such a promonad. So $\mathcal{D}$ is given by a $\mathcal{V}$-enriched functor $\mathcal{D} \colon \mathcal{C}^{\mathrm{op}}, \mathcal{C} \to \mathcal{V}_1$, together with extranatural transformations

$$m_{a,b,c} \colon \mathcal{D}(a,b), \mathcal{D}(b,c) \to \mathcal{D}(a,c) \qquad e_{a,b} \colon \mathcal{C}(a,b) \to \mathcal{D}(a,b)$$

such that the following diagrams commute (see Definition 7.5.1).

$$\begin{array}{ccc} \mathcal{D}(a,b), \mathcal{D}(b,c), \mathcal{D}(c,d) & \xrightarrow{m_{a,b,c},\mathrm{id}} & \mathcal{D}(a,c), \mathcal{D}(c,d) \\ {\scriptstyle \mathrm{id},m_{b,c,d}}\downarrow & & \downarrow{\scriptstyle m_{a,c,d}} \\ \mathcal{D}(a,b), \mathcal{D}(b,d) & \xrightarrow{m_{a,b,d}} & \mathcal{D}(a,d) \end{array}$$

$$\begin{array}{ccc} \mathcal{C}(a,b), \mathcal{D}(b,c) & \xrightarrow{p} & \mathcal{D}(a,c) \\ {\scriptstyle e_{a,b},\mathrm{id}}\downarrow & \nearrow_{m_{a,b,c}} & \\ \mathcal{D}(a,b), \mathcal{D}(b,c) & & \end{array} \qquad \begin{array}{ccc} \mathcal{D}(a,b), \mathcal{C}(b,c) & \xrightarrow{p} & \mathcal{D}(a,c) \\ {\scriptstyle \mathrm{id},e_{a,b}}\downarrow & \nearrow_{m_{a,b,c}} & \\ \mathcal{D}(a,b), \mathcal{D}(b,c) & & \end{array}$$

If we set $a = b$ in the second diagram and $b = c$ in the third, and compose with the identity multimorphisms η, then these are exactly the diagrams

(see Definition 7.6.1) for $\mathcal{D}$ to have the structure of a $\mathcal{V}$-enriched category on the collection of objects of $\mathcal{C}$!

Then the full versions of the second and third diagrams give us our desired enriched functor $\mathcal{C} \to \mathcal{D}$. It is the identity on objects and is the multimorphism $e_{a,b}$ on morphisms.

We can show that this is indeed a functor using the diagram in Figure 7.4. □

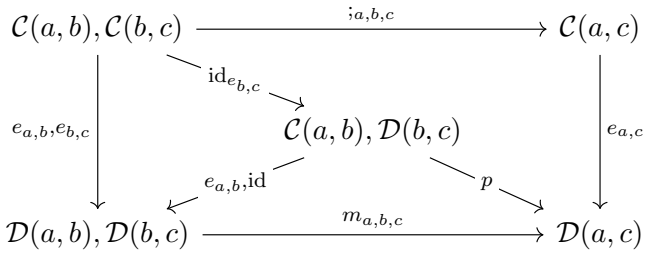


Figure 7.4: Proof that the identity-on-objects functor arising from a promonad is indeed a functor. The proof uses naturality of $e_{a,b}$ for commutativity of the large triangle at the top right.

Consider the case that $\mathcal{D}$ is an actual functor, so that $\mathcal{D}(a,b) = \mathcal{C}(a, F(b))$ for some endofunctor $F \colon \mathcal{C} \to \mathcal{C}$. Then, by Proposition 7.12.1, a promonad structure on $\mathcal{D}$ is the same thing as a monad structure on F. If we consider $\mathcal{D}$ as a category, then the objects of $\mathcal{D}$ are the objects of $\mathcal{C}$, and morphisms from a to b are morphisms from a to $F(b)$ in $\mathcal{C}$; i.e., Kleisli morphisms for F.

If we work the definitions through the proof of Proposition 7.12.1, then we see that the composition of morphisms $f \colon a \to F(b)$ and $g \colon b \to F(c)$ in $\mathcal{D}$ is given by the composite

$$a \xrightarrow{f} Fb \xrightarrow{Fg} FFc \to Fc\,,$$

where the rightmost arrow arises from the promonad structure on $\mathcal{D}$. In other words, $\mathcal{D}$ is precisely the Kleisli category for the monad F:

Slogan 7.13.2. The Kleisli category is the category we get by considering functors as profunctors.

7.14 The multicategory of functors

Let $\mathcal{X}$ be a monoidal category and let $\mathcal{M}$ be a multicategory. We define a multicategory $[\mathcal{X}, \mathcal{M}]$ where the objects are ordinary functors

$$\mathcal{X} \to \mathcal{M}_1$$

and where multimorphisms $F_1, \cdots, F_n \to G$ are natural transformations

$$\phi_{x_1,\cdots,x_n} \colon F_1(x_1), \cdots, F_n(x_n) \to G(x_1 \otimes \cdots \otimes x_n)\,.$$

Remark 7.14.1. Suppose that $\mathcal{X}$ is small and suppose that $\mathcal{M}$ is the category of sets, regarded as a multicategory through its Cartesian structure. Let $x_1, \cdots, x_n, y_1, \cdots, y_p$ be objects of $\mathcal{X}$. Then for any collection of functors

$$F_1, \cdots, F_n, G_1, \cdots, G_p, H \colon \mathcal{X} \to \mathbf{Set}\,,$$

the set of natural transformations

$$\phi_{\vec{x},\vec{y}} \colon \prod_i F_i(x_i) \times \prod_j G_j(y_j) \to H(x_1 \otimes \cdots \otimes x_n \otimes y_1 \otimes \cdots \otimes y_p)$$

may be written as the end

$$\int_{\vec{x},\vec{y}} \left[\prod_i F_i(x_i) \times \prod_j G_j(y_j), H\left(\bigotimes_i x_i \otimes \bigotimes_j y_j \right) \right]\,.$$

We may then perform some co/end calculus (See the similar computation in [Pis14], but note that that version is not quite sufficient to prove representability according to Theorem 7.2.1).

$$\begin{aligned}
&\int_{\vec{x},\vec{y}} \left[\prod_i F_i(x_i) \times \prod_j G_j(y_j), H\left(\bigotimes_i x_i \otimes \bigotimes_j y_j \right) \right] \\
&\cong \int_{\vec{x},z,\vec{y}} \left[\mathcal{X}\left(\bigotimes_i x_i, z \right), \left[\prod_i F_i(x_i) \times \prod_j G_j(y_j), H\left(z \otimes \bigotimes_j y_j \right) \right] \right] \\
&\cong \int_{\vec{x},z,\vec{y}} \left[\prod_i F_i(x_i) \times \mathcal{X}\left(\bigotimes_i x_i, z \right) \times \prod_j G_j(y_j), H\left(z \otimes \bigotimes_j y_j \right) \right] \\
&\cong \int_{z,\vec{y}} \left[\int^{\vec{x}} \left(\prod_i F_i(x_i) \times \mathcal{X}\left(\bigotimes_i x_i, z \right) \right) \times \prod_j G_j(y_j), H\left(z \otimes \bigotimes_j y_j \right) \right]
\end{aligned}$$

In other words, this multicategory is representable by the Day convolution that we met in Definition 5.1.7:

$$(F \otimes_{\mathrm{Day}} G)(z) = \int^{x,y} F(x) \times G(y) \times \mathcal{X}(x \otimes y, z)\,.$$

However, this multicategory is not representable in general, particularly in the cases when we are working with enriched multicategories (not defined here), where the enriching multicategory is not cocomplete, if $\mathcal{X}$ is large, or when the category $\mathcal{M}$ is not the enriching multicategory.

7.15 Monoids on functors are multifunctors

It might seem strange that the objects of the multicategory of functors are ordinary functors rather than multifunctors. We appear to have ignored the monoidal structure of $\mathcal{X}$ and the multicategory structure of $\mathcal{M}$.

One way to make sense of this fact is to note that an object of a category $\mathcal{C}$ is the same thing as a functor

$$1 \to \mathcal{C}\,.$$

In the same way, perhaps the correct way to think of an 'element' of a multicategory is that it is a *multifunctor*

$$1 \to \mathcal{M}\,;$$

i.e., a monoid in $\mathcal{M}$.

Then the following proposition tells us that the 'elements' of $\mathcal{X} \to \mathcal{M}$ in this sense are the multifunctors.

Proposition 7.15.1. *Let $\mathcal{X}$ be a monoidal category and let $\mathcal{M}$ be a multicategory. Then a monoid in $[\mathcal{X}, \mathcal{M}]$ is the same thing as a multifunctor $\mathcal{X} \to \mathcal{M}$.*

This can be proved by setting $\mathcal{N} = 1$ in the following stronger result.

Proposition 7.15.2 ([Pis14, 2.8])**.** *Let $\mathcal{X}$ be a monoidal category and let $\mathcal{M}, \mathcal{N}$ be multicategories. Then a multifunctor $\mathcal{N} \to [\mathcal{X}, \mathcal{M}]$ is the same thing as a multifunctor $\mathcal{N} \times \mathcal{X} \to \mathcal{M}$.*

7.16 Two perspectives on monoids in Set

We now come to our main result of the chapter. We will approach it from an oblique perspective. First note the following two rather different generalizations of the notion of an internal monoid in **Set**.

1. A monoid in **Set** may be regarded as a lax monoidal functor (i.e., a multifunctor) $1 \to \mathbf{Set}$. This generalizes to arbitrary lax monoidal functors $\mathcal{X} \to \mathbf{Set}$, for monoidal categories $\mathcal{X}$.

2. A monoid in **Set** may also be regarded as a category with a single object. This generalizes to arbitrary categories.

We shall now attempt to unify these into a single grand unifying generalization of a monoid. From Proposition 7.13.1, we know that a category with one object is the same thing as a monoid in the category $\mathrm{Endoprof}_{\mathbf{Set}}(*)$, where $*$ is the category with a single object and only an identity morphism.

We can clearly generalize this to the idea of a monoid in $\mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})$ for an arbitrary category $\mathcal{C}$. This then generalizes to the universal idea of a multifunctor

$$\mathcal{X} \to \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})\,,$$

(which we might call a *parametric promonad on $\mathcal{C}$ parameterized by $\mathcal{X}$*), which generalizes both lax monoidal functors $\mathcal{X} \to \mathbf{Set}$ (when $\mathcal{C} = *$) and **Set**-enriched categories (when $\mathcal{X} = 1$).

However, we can also do things the other way round. From Proposition 7.15.1, a lax monoidal functor $\mathcal{X} \to \mathbf{Set}$ is a monoid in the multicategory $[\mathcal{X}, \mathbf{Set}]$. This is the same thing as an $[\mathcal{X}, \mathbf{Set}]$-enriched category with a single object, so another way of generalizing monoids in **Set** is to generalize them to monoids in the multicategory

$$\mathrm{Endoprof}_{[\mathcal{X},\mathbf{Set}]}(\mathcal{C})$$

for some monoidal category $\mathcal{X}$ and some $[\mathcal{X}, \mathbf{Set}]$-enriched category $\mathcal{C}$.

This generalizes categories in the case that $\mathcal{X} = 1$. It generalizes monoidal functors $\mathcal{X} \to \mathbf{Set}$ in the case that $\mathcal{C}$ is the $[\mathcal{X}, \mathbf{Set}]$-enriched category $*_{[\mathcal{X},\mathbf{Set}]}$ with a single object $()$, where the morphisms $() \to ()$ are given by the functor $[I, \_]$, for I the monoidal unit in $\mathcal{X}$, this being the initial object in $[\mathcal{X}, \mathbf{Set}]$.

Our main result will tell us that it doesn't actually matter which way round we choose: these two ways of unifying the two generalization of a monoid in fact give the same result. The only ingredient we are missing is an appropriate change of base to move from ordinary **Set**-enriched categories to $[\mathcal{M}, \mathbf{Set}]$-enriched categories.

Definition 7.16.1. Let $\mathcal{X}$ be a small monoidal category. We have a multifunctor

$$\mathcal{X} \to [\mathbf{Set}, \mathbf{Set}]$$

given by

$$x \mapsto \mathcal{X}(I, x) \times \_\,.$$

By Proposition 7.15.2, this may equivalently be given as a multifunctor

$$\partial_{\mathcal{X}} \colon \mathbf{Set} \to [\mathcal{X}, \mathbf{Set}]$$

that sends a set A to the functor

$$\mathcal{X}(I, \_) \times A\,.$$

The important property of this particular multifunctor is as follows.

Proposition 7.16.2. *If $\mathcal{C}_1, \cdots, \mathcal{C}_n$ are categories, then $[\mathcal{X}, \mathbf{Set}]$-enriched functors $\partial_{\mathcal{X}*}\mathcal{C}_1, \cdots, \partial_{\mathcal{X}*}\mathcal{C}_n \to [\mathcal{X}, \mathbf{Set}]$ are the same thing as ordinary functors from $\mathcal{C}_1 \times \cdots \times \mathcal{C}_n$ to $[\mathcal{X}, \mathbf{Set}]$.*

Proof. Let $\mathcal{C}_1, \cdots, \mathcal{C}_n$ be categories. An $[\mathcal{X}, \mathbf{Set}]$-enriched functor

$$F\colon \partial_{\mathcal{X}*}\mathcal{C}_1, \cdots, \partial_{\mathcal{X}*}\mathcal{C}_n \to [\mathcal{X}, \mathbf{Set}]$$

is given by a map

$$F\colon \mathrm{Ob}(\mathcal{C}_1) \times \cdots \times \mathrm{Ob}(\mathcal{C}_n) \to \mathrm{Ob}([\mathcal{X}, \mathbf{Set}])\,,$$

together with, for all objects a_i, b_i of $\mathcal{C}_i$, a multimorphism

$$\partial_{X*}\mathcal{C}_1(a_1, b_1), \cdots, \partial_{X*}\mathcal{C}_n(a_n, b_n), F(a_1, \cdots, a_n) \to F(b_1, \cdots, b_n)\,;$$

i.e., a natural transformation

$$\left(\prod_i \mathcal{X}(I, x_i) \times \mathcal{C}(a_i, b_i)\right) \times F(a_1, \cdots, a_n)(y) \to$$

$$F(b_1, \cdots, b_n)(x_1 \otimes \cdots \otimes x_n \otimes y)\,.$$

By the Yoneda lemma, such a natural transformation is the same thing as a natural transformation

$$\prod_i \mathcal{C}(a_i, b_i) \times F(a_1, \cdots, a_n)(y) \to F(b_1, \cdots, b_n)(I \otimes \cdots \otimes I \otimes y)\,;$$

i.e., a natural transformation

$$\mathcal{C}(a_1, b_1) \times \cdots \times \mathcal{C}(a_n, b_n) \times F(a_1, \cdots, a_n)(y) \to F(b_1, \cdots, b_n)(y)\,.$$

But this is precisely the data of an ordinary functor $\mathcal{C}_1 \times \cdots \times \mathcal{C}_n \to [\mathcal{X}, \mathbf{Set}]$.

By naturality of the Yoneda transformation (and the left unitor), this process preserves and reflects the property of respecting composition and units. □

Theorem 7.16.3 ('Stokes's Theorem')**.** *Let $\mathcal{X}$ be a small monoidal category and let $\mathcal{C}$ be a category. Then we have an isomorphism of multicategories*

$$[\mathcal{X}, \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})] \cong \mathrm{Endoprof}_{[X, \mathbf{Set}]}(\partial_{\mathcal{X}*}\mathcal{C})\,.$$

Proof. Let $F\colon \mathcal{X} \times \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set}$ be an ordinary functor. We may view F either as the object

$$F(x, \_, \_)\colon \mathcal{X} \to \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})_1$$

of $[\mathcal{X}, \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})]$ or, by Proposition 7.16, as the object

$$F(\_, a, b)\colon \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to [\mathcal{X}, \mathbf{Set}]$$

of $\mathrm{Endoprof}_{[\mathcal{X},\mathbf{Set}]}(\partial_{\mathcal{X}*}\mathcal{C})$. Moreover, every object of each of the two categories arises in such a way. Our aim is to show that the two categories give rise to identical notions of multimorphisms between such F.

Let $F_1, \cdots, F_n, G\colon \mathcal{X} \times \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set}$ be functors. Considering the F_i as objects of $[\mathcal{X}, \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})]$, a multimorphism $F_1, \cdots, F_n \to G$ is given by a transformation

$$F_1(x_1, \_, \_), \cdots, F_n(x_n, \_, \_) \to G(x_1 \otimes \cdots \otimes x_n, \_, \_)\,;$$

natural in the x_i i.e., a transformation

$$F_1(x_1, a, b_1) \times \cdots \times F_n(x_n, b_{n-1}, c) \to G(x_1 \otimes \cdots \otimes x_n, a, c)\,.$$

natural in the x_i, a, c and extranatural in the b_i.

A multimorphism $\to G$ is given by a multimorphism

$$\to G(I, \_, \_)\,;$$

i.e., a morphism

$$\mathcal{C}(a, c) \to G(I, a, c)\,.$$

Now let us consider the F_i, G as objects of $\mathrm{Endoprof}_{[\mathcal{X},\mathbf{Set}]}(\partial_{\mathcal{X}*}\mathcal{C})$. A multimorphism $F_1, \cdots, F_n \to G$ is given by a transformation

$$F_1(\_, a, b_1), \cdots, F_n(\_, b_{n-1}, c) \to G(\_, a, c)$$

natural in a, c and extranatural in the b_i; i.e., a transformation

$$F_1(x_1, a, b_1) \times \cdots \times F_n(x_n, b_{n-1}, c) \to G(x_1 \otimes \cdots \otimes x_n, a, c)$$

natural in a, c and the x_i and extranatural in the b_i.

A multimorphism $\to G$ is given by an extranatural transformation

$$\partial_{\mathcal{X}}(\mathcal{C}(a, c)) \to G(\_, a, c)\,;$$

i.e., a natural transformation

$$\mathcal{X}(I, x) \times \mathcal{C}(a, c) \to \mathcal{G}(x, a, c)\,,$$

which by the Yoneda lemma is the same thing as a natural transformation

$$\mathcal{C}(a, c) \to \mathcal{G}(I, a, c)\,.$$

Thus, the two multicategories are isomorphic. □

Now consider the case that we have a parametric monad $\_.\_ : \mathcal{X} \times \mathcal{C} \to \mathcal{C}$ on a category $\mathcal{C}$. By considering functors as profunctors, we may identify this with a multifunctor $\mathcal{X} \to \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})$, which is the same thing as a monoid in $[\mathcal{X}, \mathrm{Endoprof}_{\mathbf{Set}}(\mathcal{C})]$. Then, by Theorem 7.16.3, we may identify this multifunctor with a monoid in $\mathrm{Endoprof}_{[\mathcal{X},\mathbf{Set}]}(\partial_{\mathcal{X}*}\mathcal{C})$; i.e., an $[\mathcal{X}, \mathbf{Set}]$-enriched promonad on $\partial_{\mathcal{X}*}\mathcal{C}$.

But now, by Proposition 7.13.1, this promonad is the same thing as an $[\mathcal{X}, \mathbf{Set}]$-enriched category that has the same objects as $\mathcal{C}$ and admits an identity-on-objects $[\mathcal{X}, \mathbf{Set}]$-enriched functor out of $\partial_{X*}(\mathcal{C})$.

The objects of this $[\mathcal{X}, \mathbf{Set}]$-enriched category are the objects of $\mathcal{C}$. By working the definitions through the proofs of Proposition 7.12.1 and Theorem 7.16.3, we see that the object of morphisms from a to b is the functor

$$x \mapsto \mathcal{C}(a, x.b) \colon \mathcal{X} \to \mathbf{Set}\,,$$

and that composition of morphisms is the multimorphism

$$\mathcal{C}(a, x.b) \times \mathcal{C}(b, y.c) \to \mathcal{C}(a, (x \otimes y).c)$$

in $[\mathcal{X}, \mathbf{Set}]$ given by sending morphisms $f \colon a \to x.b$, $g \colon b \to x.c$ to the composite

$$a \xrightarrow{f} x.b \xrightarrow{x.g} x.y.c \xrightarrow{m} (x \otimes y).c\,,$$

which is precisely the definition of composition in the Melliès category.

We get a new analogue of Slogan 7.13.2.

Slogan 7.16.4. The Melliès category is precisely the $[\mathcal{X}, \mathbf{Set}]$-enriched category that we get by considering functors as profunctors.

Chapter 8

Parametric Monads and Full Abstraction

Let a monoidal category $\mathcal{Y}$ with a small descendent set act on a category $\mathcal{G}$, where $\mathcal{G}$ is a suitable model of some programming language. In this chapter we will investigate the adequacy and full abstraction properties of the resulting category $\mathcal{G}/\mathcal{X}$, as we did with Kleisli categories in Chapter 4.

As in Chapter 4, we shall generally require $\mathcal{G}$ to be a Cartesian closed category that admits a computationally adequate denotational semantics of Idealized Algol, and we shall require that $\mathcal{G}$ may be regarded as being enriched in algebraic directed-complete partial orders in such a way that every compact morphism between the denotations of types is the denotation of some term.

As hinted at in Chapter 5, we shall also require the action of $\mathcal{Y}$ on $\mathcal{G}$ to be a reader action corresponding to an oplax symmetric monoidal functor $Y^{\mathrm{op}} \to \mathcal{G}$ that satisfies the condition in Theorem 5.11.2, so that the category $\mathcal{G}/\mathcal{X}$ is Cartesian closed.

We fix a symmetric monoidal category $\mathcal{X}$ (corresponding to $\mathcal{Y}^{\mathrm{op}}$ above) with a small ancestral set and an oplax monoidal functor $j\colon \mathcal{X} \to \mathbf{Set}$ such that for any object p of $\mathcal{X}$ there are morphisms $h\colon p \to p \otimes p$ and $h_0\colon p \to I$ such that the composite

$$j(p) \xrightarrow{jh} j(p \otimes p) \xrightarrow{m^j_{p,p}} j(p) \times j(p)$$

is equal to the diagonal on $j(p)$.

Fix a model $\mathcal{G}$ of Idealized Algol as above and suppose that the datatypes in $\mathcal{G}$ are interpreted via an oplax monoidal functor $\mathbf{Set} \to \mathcal{G}$. Then we get an oplax monoidal functor $\mathcal{X} \to \mathcal{G}$, inducing a reader action of $\mathcal{X}^{\mathrm{op}}$ on $\mathcal{G}$ such

that the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ is Cartesian closed. We will define a language and an interpretation of this language in the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$.

8.1 The language $\mathrm{IA}_{\mathcal{X}}$

Definition 8.1.1 (The language $\mathrm{IA}_{\mathcal{X}}$)**.** The language $\mathrm{IA}_{\mathcal{X}}$ is formed by taking Idealized Algol, and adding to it new constants

$$\mathsf{choose}_p$$

for each object p of $\mathcal{X}$ such that $j(p) \in \{\mathbb{C}, \mathbb{B}, \mathbb{N}\}$, with typing rule

$$\frac{}{\Gamma \vdash \mathsf{choose}_p \colon j(p)}\,.$$

The interpretation of choose_p is that it requests an element a of the set $j(p)$.

Let $\mathcal{G}$ be a model of Idealized Algol as described above, and suppose that there is an oplax monoidal functor $\mathbf{Set} \to \mathcal{G}$ that is used to interpret datatypes. We will use an underline to indicate this functor; so, for example, the object of $\mathcal{G}$ that is used to denote the natural number type is written $\underline{\mathbb{N}}$.

By our description of $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ as a lax colimit in **Cat** (i.e., Corollary 5.6.1), we have a natural functor $J \colon \mathcal{G} \to \mathcal{G}/\mathcal{X}^{\mathrm{op}}$ and a natural transformation $\phi_{p,a} \colon J(\underline{jp} \to a) \to Ja$. We define our denotational semantics of $\mathrm{IA}_{\mathcal{X}}$ in the category $\mathcal{G}/X^{\mathrm{op}}$ as follows. The denotation of any type T of Idealized Algol is the object $J(\llbracket T \rrbracket_{\mathcal{G}})$, where $\llbracket T \rrbracket_{\mathcal{G}}$ is the original denotation in $\mathcal{G}$. The denotation of any sequent $\Gamma \vdash M$ the morphism $\llbracket \Gamma \vdash M \rrbracket = J(\llbracket \Gamma \vdash M \rrbracket_{\mathcal{G}})$, where $\llbracket - \rrbracket_{\mathcal{G}}$ is the original denotation in $\mathcal{G}$. The denotation of choose_p is the morphism $\omega_p \colon 1 \to \underline{j(p)}$ given by the composite

$$1 \xrightarrow{\Lambda(\mathrm{id}_{J\underline{jp}})} (J\underline{jp} \to J\underline{jp}) \to J(\underline{jp} \to \underline{jp}) \xrightarrow{\phi_{p,\underline{jp}}} J\underline{jp}\,.$$

This denotation may alternatively be defined in a non-compositional way: given a term $\Gamma \vdash M \colon T$ in context of $\mathrm{IA}_{\mathcal{X}}$, we can write

$$M = N[\mathsf{choose}_p\,/x_p]\,,$$

where (x_p) is a finite collection of free variables in M.

Since the categories $\mathcal{G}$ and $\mathcal{G}/\mathcal{X}$ are Cartesian closed, the β-rule is valid in the semantics, and so if N is a term of $\mathrm{IA}_{\mathcal{X}}$ that refers to $(\mathsf{choose}_p)_{p \in \mathcal{P}}$ for some finite collection $\mathcal{P}$ of objects of $\mathcal{X}$, then we may write the denotation of $\Gamma \vdash N$ as the composite

$$\llbracket \Gamma \rrbracket \xrightarrow{\langle \mathrm{id}, (\omega_p) \rangle} \llbracket \Gamma, (x_p) \rrbracket \xrightarrow{\llbracket \Gamma, (x_p) \vdash N[x_p/\,\mathsf{choose}_p] \rrbracket} \llbracket T \rrbracket\,,$$

where the denotation at the right is that of ordinary Idealized Algol.

This is a morphism in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$. If we consider it as a morphism in $\mathcal{G}$, we see that it is given by the curried form of the composite

$$[\![\Gamma]\!] \times \underline{j\left(\bigotimes_p p\right)} \xrightarrow{[\![\Gamma]\!] \times m^j} [\![\Gamma]\!] \times \prod_p \underline{j(p)} \xrightarrow{[\![\Gamma, (x_p) \vdash N[x_p/\,\mathsf{choose}_p]]\!]} [\![T]\!] \ .$$

The example to have in mind is that of probability; here, the objects p of our category $\mathcal{X}$ are discrete random variables, with $j(p)$ giving the codomain of the random variable, and the term $\mathsf{choose}_p \colon j(p)$ can be thought of as randomly sampling a single element of that set with the probability distribution coming from that random variable.

8.2 Operational Semantics

We inductively define a relation

$$\Gamma, s \vdash M \Downarrow_U c, s' \,,$$

where Γ is a `Var`-context, s, s' are Γ-stores, $\Gamma \vdash M, \Gamma \vdash c$ are $\mathrm{IA}_{\mathcal{X}}$ terms-in-context such that c is an IA canonical form, and U is a sequence of pairs of the form $(p : a)$, where p is an object of $\mathcal{X}$ and $a \in j(p)$. The definition of this rule is shown in Figure 8.1.

We notice immediately that all but one of these rules are exactly the same as the corresponding rules from IA_X, except that the symbol $\Downarrow$ is now decorated with a sequence of pairs rather than a simple sequence. The one major difference is the rule for choose_p.

8.3 Translation into IA_X

Having noted the connection between the operational semantics of IA_X and $\mathrm{IA}_{\mathcal{X}}$, we now link the denotational semantics of the two languages. Recall that a morphism $A \to B$ in our model of IA_X was a Kleisli morphism; i.e., a morphism of the form

$$A \to (X \to B)$$

in the original category, while a morphism $A \to B$ in the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ is a Melliès morphism; i.e., a morphism

$$A \to (\underline{j(x)} \to B)$$

for some object x of $\mathcal{X}$. Now, if we can ensure that $j(x)$ is a set corresponding to an IA datatype, then we can treat this Melliès morphism as a Kleisli

$$\frac{}{\Gamma, s \vdash c \Downarrow_\epsilon c, s} \qquad \frac{\Gamma, s \vdash M \Downarrow_U \lambda x.M', s' \qquad \Gamma, s' \vdash M'[N/x] \Downarrow_V c, s''}{\Gamma, s \vdash MN \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M(\mathbf{Y}M) \Downarrow_U c, s'}{\Gamma, s \vdash \mathbf{Y}M \Downarrow_U c, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow_U n, s'}{\Gamma, s \vdash \mathsf{succ}\, M \Downarrow_U n+1, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow_U n+1, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow_U n, s'}$$

$$\frac{\Gamma, s \vdash M \Downarrow_U 0, s'}{\Gamma, s \vdash \mathsf{pred}\, M \Downarrow_U 0, s'} \qquad \frac{\Gamma, s \vdash M \Downarrow_U \mathsf{skip}, s' \qquad \Gamma, s' \vdash N \Downarrow_V c, s''}{\Gamma, s \vdash M; N \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_U \mathbb{t}, s' \qquad \Gamma, s' \vdash N \Downarrow_V c, s''}{\Gamma, s \vdash \mathsf{If}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_U \mathbb{f}, s' \qquad \Gamma, s' \vdash P \Downarrow_V c, s''}{\Gamma, s \vdash \mathsf{If}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_U 0, s' \qquad \Gamma, s' \vdash N \Downarrow_V c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow_U n+1, s' \qquad \Gamma, s' \vdash P \Downarrow_V c, s''}{\Gamma, s \vdash \mathsf{If0}\, M\, \mathsf{then}\, N\, \mathsf{else}\, P \Downarrow_{U \# V} c, s''}$$

$$\frac{\Gamma, s \vdash M \Downarrow c', s' \qquad \Gamma, s' \vdash N[c'/x] \Downarrow c, s''}{\Gamma, s \vdash \mathsf{let}\, x = M\, \mathsf{in}\, N \Downarrow c, s''}$$

$$\frac{\Gamma, s \vdash E \Downarrow_U n, s' \qquad \Gamma, s' \vdash V \Downarrow_V x, s''}{\Gamma, s \vdash V \leftarrow E \Downarrow_{U \# V} \mathsf{skip}, (s''|x \mapsto n)}\ x \in \Gamma$$

$$\frac{\Gamma, s \vdash V \Downarrow_U x, s'}{\Gamma, s \vdash !V \Downarrow_U n, s'}\ s'(x) = n \qquad \frac{\Gamma, x\colon \mathtt{Var}, (s|x \mapsto 0) \vdash M \Downarrow_U c, (s'|x \mapsto n)}{\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow_U c, s'}$$

$$\frac{\Gamma, s \vdash E \Downarrow_U n, s' \qquad \Gamma, s' \vdash V \Downarrow_V \mathsf{mkvar}\, W R, s'' \qquad \Gamma, s'' \vdash Wn \Downarrow_W \mathsf{skip}, s'''}{\Gamma, s \vdash V \leftarrow E \Downarrow_{U \# V \# W} \mathsf{skip}, s'''}$$

$$\frac{\Gamma, s \vdash V \Downarrow_U \mathsf{mkvar}\, W R, s' \qquad \Gamma, s' \vdash R \Downarrow_V n, s''}{\Gamma, s \vdash !V \Downarrow_{U \# V} n, s''}$$

$$\frac{}{\Gamma, s \vdash \mathsf{choose}_p \Downarrow_{(p:a)} a, s}\ a \in j(p)$$

Figure 8.1: Operational semantics for $\mathrm{IA}_{\mathcal{X}}$.

morphism and start to pull results from Chapter 4 over to the parametric monad case.

We have already required that the terms choose_p are such that $j(p)$ is always an IA datatype. However, when we compose two Melliès morphisms together, we take the tensor product of the corresponding objects of $\mathcal{X}$. Therefore, we need to ensure that we can represent $j(p_1 \otimes \cdots \otimes p_n)$ using an IA datatype, for any finite collection p_i of objects of $\mathcal{X}$ such that $j(p_i)$ is an IA datatype for all i.

We shall therefore assume that, whenever $p_1, \cdots, p_n$ is a finite collection of objects of $\mathcal{X}$ such that the set $j(p_i)$ is an IA datatype for all i.

The case we have in mind is when the $j(p_i) = \mathbb{B}$ for all i and N is the natural number object, so that we can choose some way of representing elements of jp as elements of N.

Definition 8.3.1. Let $p_1, \cdots, p_n$ be a sequence of objects of $\mathcal{X}$ and let N be an object of $\mathcal{X}$ such that $j(N)$ is a datatype of Idealized Algol (i.e., $j(N) \in \{\mathbb{C}, \mathbb{B}, \mathbb{N}\}$). Suppose we have a morphism

$$f \colon N \to p_1 \otimes \cdots \otimes p_n$$

in $\mathcal{X}$ such that jf is a surjection. Define functions $\pi_i \colon j(N) \to j(p_i)$ (depending on f) to be given by the composites

$$j(N) \xrightarrow{jf} j(p_1 \otimes \cdots \otimes p_n) \xrightarrow{m^j} j(p_1) \times \cdots \times j(p_n) \xrightarrow{\mathrm{pr}_i} j(p_i)\,.$$

Let $u \in j(N)^*$ be a sequence of elements of $j(N)$, and let U be a sequence of pairs $(p : a)$, where each p is one of the p_i. We say that u *covers* U *with respect to* f if U and u have the same length and if whenever $U^{(k)} = (p_i : a)$, we have $a = \pi_i(u^{(k)})$.

Recall that, in the definition of the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$, the Melliès morphisms are left unchanged by precomposing with a morphism in the image of the functor j; therefore, if $M \colon T$ is a closed term of $\mathrm{IA}_{\mathcal{X}}$ referring to choose_{p_1}, $\cdots$, choose_{p_n}, then we may write the denotation of M as the composite

$$\underline{jN} \xrightarrow{jf} \underline{j(p_1 \otimes \cdots \otimes p_n)} \xrightarrow{m^j} \underline{jp_1} \times \cdots \times \underline{jp_n} \xrightarrow{[\![x_1, \cdots, x_n \vdash M[x_i/\,\mathsf{choose}_{p_i}]]\!]_{\mathcal{G}}} [\![T]\!]\,;$$

i.e., as

$$\underline{jN} \xrightarrow{\langle \pi_1, \cdots, \pi_n \rangle} \underline{jp_1} \times \cdots \times \underline{jp_n} \xrightarrow{[\![x_1, \cdots, x_n \vdash M[x_i/\,\mathsf{choose}_{p_i}]]\!]_{\mathcal{G}}} [\![T]\!]\,.$$

Lemma 8.3.2. *Let $\Gamma \vdash M$ be an $\mathrm{IA}_{\mathcal{X}}$ term-in-context, where M refers to terms $\mathsf{choose}_{p_1}, \cdots, \mathsf{choose}_{p_n}$, and no other* choose *terms. Let N be an object*

of $\mathcal{X}$ such that $j(N)$ is an IA datatype and let $f\colon N \to p_1 \otimes \cdots \otimes p_n$ be a morphism, as in Definition 8.3.1. Suppose that the functions π_i are all definable in Idealized Algol; that is, that there are terms $\Pi_i\colon j(N) \to j(p_i)$ of IA such that the following inference is valid.

$$\frac{\Gamma, s \vdash M \Downarrow m, s'}{\Gamma, s \vdash \Pi_i M \Downarrow \pi_i(m), s'}$$

Let s, s' be Γ-stores, and let $\Gamma \vdash c$ be a canonical form. Suppose U is a sequence such that we have

$$\Gamma, s \vdash M \Downarrow_U c, s' .$$

Then

$$\Gamma, s \vdash M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_u c, s'$$

in $IA_{j(N)}$ for all sequences $u \in j(N)^*$ that cover U.

Proof. Induction on the derivation of $\Gamma, s \vdash M \Downarrow_U c, s'$. Suppose that the last rule in the derivation takes the following form.

$$\frac{\Gamma_1, s^{(0)} \vdash M_1 \Downarrow_{U_1} c_1, s^{(1)} \qquad \cdots \qquad \Gamma_n, s^{(n-1)} \vdash M \Downarrow_{U_n} c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow_{U_1 + \cdots + U_n} c, s^{(n)}}$$

Suppose a sequence u covers $U_1 + \cdots + U_n$. Then we may write $u = u_1 + \cdots + u_n$, where u_i covers U_i.

By induction, then, we may derive that

$$\Gamma_k, s^{(k)} \vdash M_k[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_{u_k} c_k, s^{(k)}$$

for $k = 1, \cdots, n$. Now note that Lemma 4.6.3 still holds if we use the terms choose_{p_i} instead of the ask_X; this means that we have a valid $\mathrm{IA}_{j(N)}$ inference given by

$$\frac{\begin{array}{c}\Gamma_1, s^{(0)} \vdash M_1[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_{u_1} c_1, s^{(1)} \\ \cdots \qquad \Gamma_n, s^{(n-1)} \vdash M_n[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_{u_n} c_n, s^{(n)}\end{array}}{\Gamma, s^{(0)} \vdash M[(\lambda z.\Pi_i)\,\mathsf{ask}_{j(N)}\,.\,\mathsf{choose}_{p_i}] \Downarrow_u c, s^{(n)}} ,$$

from which we can deduce that

$$\Gamma, s^{(0)} \vdash M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_i c, s^{(n)} ,$$

as desired.

The other possibility is that the final step in the derivation takes the form

$$\frac{}{\Gamma, s \vdash \mathsf{choose}_{p_j} \Downarrow_{(p_j:a)} a, s} .$$

Let U be a (length 1) sequence covering $(p_j : a)$. So $U = t$, where $t \in j(P)$ is such that $\pi_j(t) = a$.

Then

$$\mathsf{choose}_{p_j}[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] = \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v)\,,$$

and we may derive

$$\Gamma, s \vdash \mathtt{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v) \Downarrow_t a, s\,. \qquad \square$$

To prove the converse, we prove a lemma about substitution analogous to Lemma 4.7.1.

Lemma 8.3.3. *Let*

$$\frac{\Gamma, s^{(0)} \vdash M_1 \Downarrow_{u_1} c_1, s^{(1)} \qquad \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n \Downarrow_{u_n} c_n, s^{(n)}}{\Gamma, s^{(0)} \vdash M \Downarrow_{u_1 \# \cdots \# u_n} c, s^{(n)}}$$

be an inference of $IA_{j(N)}$, where every instance of $\mathsf{ask}_{j(N)}$ occurs as part of some term of the form $\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)$, and suppose that $M \neq \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v)$ for any j. Suppose we have sequences $U_1, \cdots, U_n$ such that u_k covers U_k for $k = 1, \cdots, n$. Then

$$\frac{\begin{array}{c}\Gamma, s^{(0)} \vdash M_1[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_{U_1} c_1, s^{(1)} \\ \cdots \qquad \Gamma, s^{(n-1)} \vdash M_n[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_{U_n} c_n, s^{(n)}\end{array}}{\Gamma, s^{(0)} \vdash M[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_{U_1 \# \cdots \# U_n} c, s^{(n)}}$$

is a valid inference of $IA_{\mathcal{X}}$.

Proof. As in Lemma 4.7.1, we can prove this by looking at cases. For example, consider the sequencing rule

$$\frac{\Gamma, s \vdash M \Downarrow_u \mathsf{skip}, s' \qquad \Gamma, s' \vdash N \Downarrow_v c, s''}{\Gamma, s \vdash M; N \Downarrow c, s''}\,.$$

We have

$$\begin{array}{l}(M; N)[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \\ = M[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)]; \\ \quad N[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)]\,,\end{array}$$

and so we certainly get a rule

$$\frac{\begin{array}{c}\Gamma, s \vdash M[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_U \mathsf{skip}, s' \\ \Gamma, s' \vdash N[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_V c, s''\end{array}}{\Gamma, s \vdash (M; N)[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow c, s''}\,.$$

The only case where we need to be careful is for the new rule:

$$\frac{\Gamma, x, (s|x \mapsto 0) \vdash M \Downarrow_u c, (s'|x \mapsto n)}{\Gamma, s \vdash \mathsf{new}\, \lambda x.M \Downarrow_u c, s'} \quad .$$

If $\mathsf{new}\, \lambda x.M \neq \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v)$, then we have

$$\begin{aligned} &(\mathsf{new}\, \lambda x.M)[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \\ &= \mathsf{new}\, \lambda x.(M[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)]) \,. \end{aligned}$$

Then we can apply the rule

$$\frac{\Gamma, x, (s|x \mapsto 0) \vdash M[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow_U c, (s'|x \mapsto n)}{\Gamma, s \vdash (\mathsf{new}\, \lambda x.M)[\mathsf{choose}_{p_i} / \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)] \Downarrow Uc, s'} \quad .$$

□

We now prove the converse to Lemma 8.3.2.

Lemma 8.3.4. *Let $\Gamma, y_1 : j(p_1), \cdots, y_n : j(p_n) \vdash M : T$ be a term-in-context of ordinary Idealized Algol, where Γ is a* `Var`*-context. Let U be a sequence and let N, π_i, Π_i be as above. Suppose that there exists some sequence $u \in j(N)^*$ such that u covers U and such that*

$$\Gamma, s \vdash M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/y_i] \Downarrow_u c, s' \,.$$

Then

$$\Gamma, s \vdash M[\mathsf{choose}_{p_i} / y_i] \Downarrow_U c, s' \,.$$

Proof. Induction on the derivation. Suppose that M is not one of the y_i; then $M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/y_i]$ is not equal to $\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)$. Moreover, every instance of $\mathsf{ask}_{j(N)}$ in M occurs as part of an expression of the form $\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)$, and so we win by Lemma 8.3.3 and the inductive hypothesis.

Otherwise, $M = \mathtt{new}\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v$ for some j. Now, if we have

$$\Gamma, s \vdash \mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_j!v) \Downarrow_u c, s' \,,$$

then a simple examination of the reduction tells us that we must have $s' = s$, and that u must have length 1 – say $u = m$ – where the single element m of u satisfies $\pi_j(m) = c$.

But now we certainly have

$$\Gamma, s \vdash \mathsf{choose}_{p_j} \Downarrow_{(p_j:c)} c, s \,,$$

and the sequence m covers the sequence $(p_j : c)$. □

Lemmas 8.3.2 and 8.3.4 together prove the following.

Lemma 8.3.5. *Let* $\Gamma, x_1, \cdots, x_n \vdash M$ *be a term-in-context of Idealized Algol, where* Γ *is a* `Var`*-context. Then the following are equivalent.*

i) $\Gamma, s \vdash M[\mathsf{choose}_{p_i/x_i}] \Downarrow_U c, s'$ *in* $IA_{\mathcal{X}}$.

ii) $\Gamma, s \vdash M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i !v/x_i] \Downarrow_u c, s'$ *in* IA_N *for all* u *covering* U.

iii) $\Gamma, s \vdash M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i !v/x_i] \Downarrow_u c, s'$ *in* IA_N *for some* u *covering* U.

Proof. (i) $\Rightarrow$ (ii): Lemma 8.3.2.

(ii) $\Rightarrow$ (iii): By assumption, the function $j(f)\colon N \to j(\bigotimes_i p_i)$ is surjective, so for any U there is some $u \in j(N)^*$ covering U.

(iii) $\Rightarrow$ (i): Lemma 8.3.4. □

8.4 Computational Adequacy

We are now ready to make the definitions we need to state our Computational Adequacy result.

Recall that if σ was a Kleisli morphism $1 \to \mathbb{C}$ (i.e., a morphism $1 \to (X \to \mathbb{C})$ in the original category, where X was an Idealized Algol datatype), then we wrote $\sigma \downarrow_u$ if the composite

$$1 \xrightarrow{\sigma} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathrm{Var} \to \mathbb{N}) \xrightarrow{\mathsf{new}} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

was not equal to $\bot$, where η_u was the denotation of the Idealized Algol term-in-context

$$f\colon X \to \mathtt{com} \vdash \lambda v.v \leftarrow 0; f(v \leftarrow \mathsf{succ}\, !v; \mathrm{tr}_u\, !v); !v\colon \mathtt{Var} \to \mathtt{nat}\,.$$

We want to extend this definition to morphisms in the category $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$. There are a couple of problems here.

Firstly, the morphisms in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ are equivalence classes of Melliès morphisms, and the equivalence relation does not respect this predicate $\downarrow_u$ – especially since the X in the above formula could change when we choose a different representative of the equivalence class.

Secondly, a morphism $1 \to \mathbb{C}$ in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ is given by an (equivalence class of) morphisms $1 \to (\underline{j(p)} \to \mathbb{C})$ in $\mathcal{G}$, and the object $\underline{j(p)}$ need not be an Idealized Algol datatype.

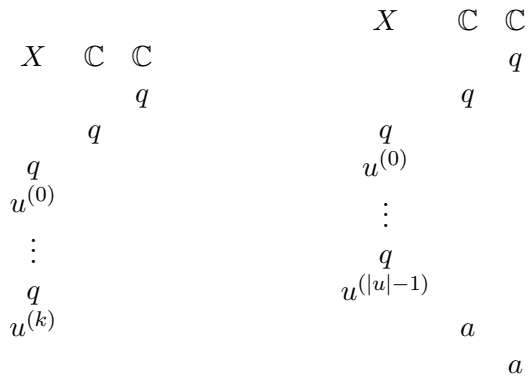
To solve the second problem, we make an additional small assumption on our category $\mathcal{G}$. We require that our category $\mathcal{G}$ contains morphisms

$$\xi_u \colon (X \to \mathbb{C}) \to \mathbb{C}$$

for any set X and any finite sequence $u \in X^*$ such that for any function $f \colon X \to Y$, we have $(\underline{f} \to \mathbb{C}); \xi_u = \xi_{f_* u}$ (where $f_* u$ is the sequence formed by applying f pointwise to u), and such that if X is an IA datatype, then

$$\xi_u = (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathrm{Var} \to \mathbb{N}) \xrightarrow{\mathsf{new}} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}\,.$$

Example 8.4.1. In the category of games, the morphisms ξ_u can be taken to be the strategies containing the plays ϵ, qq and plays of the form



(so the strategy has no reply if player O asks the question in X fewer than $|u|$ times before returning, or tries to ask it more than $|u|$ times).

Definition 8.4.2. Given a set X and a morphism $\sigma \colon 1 \to (X \to \mathbb{C})$, we say that σ *accepts* a sequence $u \in X^*$ if $\sigma; \xi_u \neq \bot$. We write $\mathrm{Acc}(\sigma)$ for the set of all sequences accepted by σ.

Recall that a morphism $1 \to \mathbb{C}$ in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ is given by an equivalence class of Melliès morphisms $1 \to (\underline{j(p)} \to \mathbb{C})$ in $\mathcal{G}$, where p ranges over the objects of $\mathcal{X}$, and where the equivalence relation is generated by identifying all pairs of morphisms $\sigma \colon 1 \to (\underline{j(p)} \to \mathbb{C})$ and $\tau \colon 1 \to (\underline{j(q)} \to \mathbb{C})$ such that there is a morphism $f \colon p \to q$ such that $\tau; (\underline{j(f)} \to A) = \sigma$.

Definition 8.4.3. We define an equivalence relation on pairs $(p, \mathcal{U})$, where p is an object of $\mathcal{X}$ and $\mathcal{U} \subseteq j(p)^*$ is a set of finite sequences drawn from $j(p)$ to be the equivalence relation generated by identifying $(p, \mathcal{U})$ and $(q, \mathcal{V})$ whenever there is a morphism $f \colon p \to q$ in $\mathcal{X}$ such that for all $u \in j(p)^*$, we have $u \in \mathcal{U}$ if and only if $j(f)_* u \in \mathcal{V}$.

It is instructive to consider the equivalence relation on pairs $(p, \mathcal{U})$ in the case that $\mathcal{X} = \mathbf{Rv}_\Omega$ is the category of random variables on some probability space Ω. Given a random variable V taking values in a set X, we get an induced notion of probability for the elements of X^*: given a sequence u of elements of x, we write

$$\mathbb{P}(u) = \prod_{i=0}^{|u|-1} \mathbb{P}(V = u^{(i)}) \,.$$

If we have a random variable W on a set Y and a function $f \colon X \to Y$ such that $W = f \circ X$, and if $\mathcal{U} \subseteq X^*$ and $\mathcal{V} \subseteq Y^*$ are such that $u \in \mathcal{U}$ if and only if $f_* u \in \mathcal{V}$, then the induced probabilities of the sets $\mathcal{U}$ and $\mathcal{V}$ are the same. So, in this case, the equivalence relation on sets of sequences is subsumed into the very natural equivalence relation of having the same probability.

Proposition 8.4.4. *Let $\sigma \colon 1 \to (\underline{j(p)} \to A)$, $\tau \colon 1 \to (\underline{j(q)} \to A)$ be two representatives of the same morphism $\overline{1 \to A}$ in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$. Then $(p, \mathrm{Acc}(\sigma))$ and $(q, \mathrm{Acc}(\tau))$ are equivalent.*

Proof. Since the relation on pairs $(p, \mathcal{U})$ is an equivalence relation, it suffices to assume that σ and σ' are related by the relation that generates the equivalence relation on Melliès morphisms; i.e., that there is a morphism $f \colon p \to q$ such that $\sigma = \tau; (j(f) \to \mathbb{C})$.

Let $u \in j(p)^*$. Then we have

$$\begin{aligned} u \in \mathrm{Acc}(\sigma) &\Leftrightarrow \sigma; \xi_u \neq \bot \\ &\Leftrightarrow \tau; (j(f) \to A); \xi_u \neq \bot \\ &\Leftrightarrow \tau \xi_{j(f)_* u} \neq \bot \\ &\Leftrightarrow j(f)_* u \in \mathrm{Acc}(\tau) \,. \end{aligned}$$

Therefore, $(p, \mathrm{Acc}(\sigma))$ and $(q, \mathrm{Acc}(\tau))$ are equivalent. □

We can now state and prove our Computational Adequacy result. For this result, given a term $M \colon \mathtt{com}$ mentioning objects $p_1, \cdots, p_n$, we shall assume the existence of some IA datatype N admitting a morphism $f \colon N \to p_1 \otimes \cdots \otimes p_n$ such that the corresponding projections π_i on to the objects $j(p_i)$ are IA-definable.

For example, if $j(p_i) = \mathbb{B}$ for all i, then we can take $N = \mathbb{N}$ and use the binary encoding.

Definition 8.4.5. Let M be a closed term of $\mathrm{IA}_\mathcal{X}$ of type $\mathtt{com}$ mentioning $p_1, \cdots, p_n$. Let $S(M)$ be the set of all sequences U such that $M \Downarrow_U \mathsf{skip}$.

We define $B(M)$, the *behaviours of* M, to be the equivalence class corresponding to the pair

$$(p_1 \otimes \cdots \otimes p_n, \mathcal{U}) \,,$$

where $\mathcal{U}$ is the set of all sequences $u \in j(p_1 \otimes \cdots \otimes p_n)^*$ that cover some sequence $U \in S(M)$, via the projections

$$j(p_1 \otimes \cdots \otimes p_n) \xrightarrow{m^j} j(p_1) \times \cdots \times j(p_n) \xrightarrow{\mathrm{pr}_i} j(p_i) \,.$$

Theorem 8.4.6 (Computational Adequacy for $\mathrm{IA}_{\mathcal{X}}$)**.** *Let* $M \colon \mathtt{com}$ *be a closed term of* $IA_{\mathcal{X}}$ *referring to* $p_1, \cdots, p_n$*. Suppose the denotation of* M *is given by a morphism* $1 \to (j(p) \to \mathbb{C})$ *in* $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$*.*

Then $(p, \mathrm{Acc}([\![M]\!]))$ *is equivalent to* $B(M)$*.*

Proof. By Proposition 8.4.4, we may assume that the denotation of M is in a particular form, namely the (curried form of) the composite

$$j(N) \xrightarrow{\langle \pi_1, \cdots, \pi_n \rangle} j(p_1) \times \cdots \times j(p_n) \xrightarrow{[\![x_1, \cdots, x_n \vdash M[x_i/\,\mathsf{choose}_{p_i}]]\!]_{\mathcal{G}}} \mathbb{C} \,.$$

But if we consider this as a Kleisli morphism in the category $\mathrm{Kl}_{R_{j(N)}}\,\mathcal{G}$, then this is the denotation of the $\mathrm{IA}_{j(N)}$ term

$$M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \,.$$

By Lemma 8.3.5, if $u \in j(N)^*$ is a sequence, then

$$M[\mathsf{new}(\lambda v.v \leftarrow \mathsf{ask}_{j(N)}; \Pi_i!v)/\,\mathsf{choose}_{p_i}] \Downarrow_u \mathsf{skip}$$

if and only if u covers a sequence U such that $M \Downarrow_U \mathsf{skip}$. By our Computational Adequacy result for $\mathrm{IA}_{\mathcal{X}}$ (Propositions 4.6.9 and 4.7.4), this means that for all $u \in j(N)^*$, $u \in \mathrm{Acc}([\![M]\!])$ (for this particular form of $[\![M]\!]$) if and only if $u \in \mathcal{U}$. Therefore, $(N, \mathrm{Acc}([\![M]\!])) = (N, \mathcal{U}')$, where $\mathcal{U}' \subseteq j(N)^*$ is the set of all sequences u that cover some U such that $M \Downarrow_U \mathsf{skip}$ via the projections π_i. Lastly, we note that $(N, \mathcal{U}')$ is equivalent to $B(M)$, through the morphism $f \colon N \to p_1 \otimes \cdots \otimes p_n$. □

8.5 Equational Soundness

We transfer to an Equational Soundness result in our standard way. First, we make a definition of observational equivalence of $\mathrm{IA}_{\mathcal{X}}$ terms.

Definition 8.5.1 (Observational Equivalence)**.** Let $M, M' \colon T$ be closed terms of $\mathrm{IA}_{\mathcal{X}}$. We say that M and M' are *observationally equivalent* if $B(C[M])$ and $B(C[M'])$ are equivalent for all contexts $C \colon \mathtt{com}$ with a hole of type T.

We then make definitions that will mirror this equivalence in the denotational semantics.

Definition 8.5.2 (Equivalence of morphisms $1 \to \mathbb{C}$)**.** Let $\sigma, \tau \colon 1 \to \mathbb{C}$ be morphisms in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$, considered as morphisms $\sigma \colon \underline{j(p)} \to \mathbb{C}$ and $\tau \colon \underline{j(q)} \to \mathbb{C}$ in $\mathcal{G}$. We say that $\sigma \approx \tau$ if $(p, \mathrm{Acc}(\sigma))$ is equivalent to $(q, \mathrm{Acc}(\tau))$.

Definition 8.5.3 (Intrinsic Equivalence)**.** Let $\sigma, \tau \colon A \to B$ be morphisms in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$. Then we say that $\sigma \sim \tau$ if for all $\alpha \colon (A \to B) \to \mathbb{C}$, we have $\Lambda(\sigma); \alpha \approx \Lambda(\tau); \alpha$.

Now we can prove Equational Soundness as we did in Proposition 3.4.13.

Theorem 8.5.4 (Equational Soundness for $\mathrm{IA}_{\mathcal{X}}$)**.** *Let $M, M' \colon T$ be closed terms of $IA_{\mathcal{X}}$ such that $[\![M]\!] \sim [\![M']\!]$. Then M and M' are observationally equivalent.*

Proof. First suppose that M and M' are not observationally equivalent – so there is some context C such that $B(C[M])$ and $B(C[M'])$ are inequivalent. Now $B(C[M])$ is equivalent to $(N, \mathcal{U})$ and $B(C[M'])$ is equivalent to $(N, \mathcal{U}')$, where $\mathcal{U} \subseteq j(N)^*$ is the set of sequences that cover some $U \in S(C[M])$ and $\mathcal{U}'$ the set of sequences that cover some $U \in S(C[M'])$ via the projections π_i.

Let α be the denotation of the term-in-context $f \colon T \vdash C[f]$. Then $\Lambda([\![M]\!]); \alpha$ is the denotation of $C[M]$ and $\Lambda([\![M']\!]); \alpha$ the denotation of $C[M']$. By Theorem 8.4.6, the sets $(N, \mathrm{Acc}(\Lambda([\![M]\!]); \alpha))$ and $(N, \mathrm{Acc}(\Lambda([\![M']\!]; \alpha)))$ are inequivalent, and so $[\![M]\!] \not\sim [\![M']\!]$. □

8.5.1 Full Abstraction for $\mathrm{IA}_{\mathcal{X}}$

In order to prove Full Abstraction, we first prove a compact definability result.

Definition 8.5.5. Let $\sigma \colon A \to B$ be a morphism in $\mathcal{G}/(\mathbf{Rv}_{\Omega}^{FS})^{\mathrm{op}}$. We say that σ is *compact* if it is compact when considered as a morphism in $\mathcal{G}$.

Remark 8.5.6. When we say 'considered as a morphism in $\mathcal{G}$' in the above definition, we mean 'in at least one of its possible interpretations as a morphism in $\mathcal{G}$'. Note, however, that the continuous image of a compact element is compact, and so if we pass to a new representative of σ by composing on the left by the image of some morphism in $\mathbf{Rv}_{\Omega}^{FS}$, then the resulting representative of σ will also be compact.

Lastly, if $\mathcal{G}$ is the category of games, this compactness property is invariant under the choice of representative for σ.

In order to prove compact definability, we need to make a further assumption.

Proposition 8.5.7 (Compact Definability for $\mathrm{IA}_{\mathcal{X}}$)**.** *Suppose the functor $j\colon \mathcal{X} \to \mathbf{Set}$ is strong monoidal and that for any object x of $\mathcal{X}$ there is a finite collection $p_1, \cdots, p_n$ such that $j(p_i)$ is an IA datatype for each i and such that there is a morphism*

$$h\colon p_1 \otimes \cdots \otimes p_n \to x\,.$$

Then any compact morphism in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$ between denotations of IA types is definable.

Proof. Let $\sigma\colon [\![S]\!] \to [\![T]\!]$ be a morphism in $\mathcal{G}/\mathcal{X}^{\mathrm{op}}$, where S and T are IA types. After composing with some suitable h as above, we may assume that σ is given by a compact morphism

$$[\![S]\!] \to \underline{j(p_1 \otimes \cdots \otimes p_n)} \to [\![T]\!]$$

in $\mathcal{G}$. After precomposing with the multiplicative coherence for j on the p_i, we get a compact morphism

$$[\![S]\!] \to (\underline{j(p_1) \times \cdots \times j(p_n)} \to [\![T]\!])\,,$$

which is the denotation of some term-in-context

$$v\colon S, a_1\colon p_1, \cdots, a_n\colon p_n \vdash M\colon T$$

by compact definability for $\mathcal{G}$. Then, by our earlier work, we know that σ itself is the denotation of the term

$$v\colon S \vdash M[\mathsf{choose}_{p_i}/a_i]\colon T\,. \qquad \square$$

We can now prove Full Abstraction.

Theorem 8.5.8 (Full Abstraction for $\mathrm{IA}_{\mathcal{X}}$)**.** *Suppose that our action satisfies the conditions of Proposition 8.5.7. Let $M, M'\colon T$ be closed terms of $IA_{\mathcal{X}}$. Then M and M' are observationally equivalent if and only if $[\![M]\!] \sim [\![M]\!]'$.*

Proof. The right-to-left direction is Theorem 8.5.4. For the other direction, suppose that $[\![M]\!] \not\sim [\![M]\!]'$. Then, without loss of generality there is some α such that the sets $(N, \mathrm{Acc}(\Lambda([\![M]\!]); \alpha))$ and $(N, \mathrm{Acc}(\Lambda([\![M']\!]\,; \alpha))$ are inequivalent. Since $\mathcal{G}$ is enriched in algebraic domains, α may be taken to be compact, and is therefore the denotation of some term-in-context $L\colon T \to \mathtt{com}$. Then, by our Computational Adequacy result, we have that $B(LM)$ and $B(LM')$ are inequivalent; i.e., that M is not observationally equivalent to N. $\square$

8.6 Probability

We now specialize to the case where $\mathcal{X}$ is a category of random variables on some fixed probability space $(\Omega, \mathcal{F}, \mathbb{P})$, in order to model a probabilistic language. For our purposes, it will suffice to take Ω to be the real interval $(0, 1)$ with its Lebesgue σ-algebra and measure. Note that the Lebesgue measure satisfies the conditions from Example 6.3.6 and Proposition 6.3.7 – every countable set is measurable and every subset of a measure-zero set is measurable.

A *random variable* on Ω is a measurable function $V \colon \Omega \to X$. Given such a random variable, and $A \subseteq X$, we write $\mathbb{P}(V \in A)$ for $\mathbb{P}(V^{-1}(A))$, and $\mathbb{P}(V = x)$ for $\mathbb{P}(V \in \{x\})$.

The category $\mathcal{X} = \mathbf{Rv}_\Omega^{FS}$ will then be the category whose objects are random variables of *finite support*; that is, discrete spaces X together with measurable functions $V \colon \Omega \to X$, such that there is some finite subset $Y \subseteq X$ satisfying $\mathbb{P}(V \in Y) = 1$.

The morphisms in $\mathbf{Rv}_\Omega^{FS}$ from $V \colon \Omega \to X$ to $W \colon \Omega \to Y$ are probability-preserving functions $X \to Y$; i.e., functions $X \to Y$ such that for all $A \subseteq Y$, we have $\mathbb{P}(f(V) \in A) = \mathbb{P}(W \in A)$.

There is a natural strict monoidal functor $\mathbf{Rv}_\Omega^{FS} \to \mathbf{Set}$ (sending $V \colon \Omega \to X$ to the set X), which gives us an oplax monoidal functor $\mathbf{Rv}_\Omega^{FS} \to \mathcal{G}$ and hence a lax reader action of $\mathbf{Rv}_\Omega^{FS}$ on $\mathcal{G}$. By our discussion in Chapter 6, this action satisfies all of the requirements we imposed in the previous section.

Recall that the tensor product of two random variables $V \colon \Omega \to X$ and $W \colon \Omega \to Y$ is their pairing $V \otimes W = \langle V, W \rangle \colon \Omega \to X \times Y$.

We define a language Probabilistic Algol (PA) to be the sublanguage of $\mathrm{IA}_{\mathbf{Rv}_\Omega^{FS}}$ generated by the terms of Idealized Algol and the terms

$$\mathsf{choose}_{V_p},$$

where $p \in [0, 1]$, and where we have identified V_p is the Bernoulli random variable

$$B_p \colon \Omega \to \mathbb{B}$$

that returns $\mathtt{t}$ if its input is less than p and $\mathtt{f}$ if it is greater than or equal to p.

The denotation of a base term of PA of type S is a Melliès morphism taking the form

$$\underline{(}1) \to [\![S]\!]$$

(for the IA terms) or the form

$$\underline{(}\mathbb{B}) \to [\![S]\!]$$

for the term choose_p. When we compose or tensor these together, we take the tensor products of the objects on the left in $\mathcal{X}$, which corresponds, after application of the oplax monoidal functor j, to taking Cartesian products in **Set**, and thence to taking Cartesian products in $\mathcal{G}$.

The denotation of any term of PA of type T, then, will be an (equivalence class of) morphisms

$$\underline{\mathbb{B}^n} \xrightarrow{m} \underline{\mathbb{B}}^n \to [\![T]\!] \,,$$

together with some random variable taking values in $\mathbb{B}^n$ (formed by taking the tensor product of Bernoulli random variables).

Lastly, given such a random variable $V \colon \Omega \to \mathbb{B}^n$, there is a random variable $\tilde{V} \colon \Omega \to \mathbb{N}$ such that for each $\vec{v} \in \mathbb{B}^n$, we have

$$\mathbb{P}\left(\tilde{V} = \sum_{i=1}^{n} 2^{i-1}\vec{v}_i\right) = \mathbb{P}(V = \vec{v})$$

and such that $\mathbb{P}(\tilde{V} = k) = 0$ for any $k \geq 2^n$. Then there is a function $f \colon \mathbb{N} \to \mathbb{B}^n$ that sends $\sum_{i=1}^{n} 2^{i-1}a_i$ to $(a_1, \cdots, a_n)$ and sends $k \geq 2^n$ to some fixed value (say, $(\mathbb{f}, \cdots, \mathbb{f})$). This function f satisfies

$$f \circ \tilde{V} = V \,.$$

Moreover, $\tilde{V}$ has finite support.

Now suppose that X is a finite discrete probability space. Then the set X^ω of all infinite sequences of elements of X may be given the product topology, and equipped with the resulting Borel σ-algebra. A basic open subset of X^ω is a set $\mathfrak{S} \subseteq X^\omega$ for which there exists some n such that if $s \in \mathfrak{S}$ and t is a sequence such that s and t are identical on the first n terms, then $t \in \mathfrak{S}$. We can define a pre-probability measure on these basic open sets by setting

$$\mathbb{P}(\mathfrak{S}) = \sum_{u \in \mathfrak{S}|_n} \prod_{i=0}^{n-1} \mathbb{P}(u^{(i)}) \,,$$

where $\mathfrak{S}|_n$ is the set of all length-n prefixes of elements of $\mathfrak{S}$.

Then the Carathéodory Extension Theorem tells us that there is a unique extension of this to a probability measure on the whole space (see, for example, [Shr04, 1.1.4]).

If $V \colon \Omega \to Z$ is a finitely-supported random variable, then V induces a probability measure on its support $\mathrm{Im}(V) \subseteq Z$. This gives us a probability measure on $\mathrm{Im}(V)^*$, which we can extend to a probability measure on Z^* by setting

$$\mathbb{P}(A) = \mathbb{P}(A \cap \mathrm{Im}(V)^*)$$

for any $A \subseteq Z^*$.

Definition 8.6.1. Let $V\colon \Omega \to X$ be a finitely supported random variable and let $\mathcal{U} \subseteq X^*$ be a set of sequences. Then we define

$$\mathbb{P}(V,\mathcal{U}) = \mathbb{P}(\mathcal{U}^\omega)\,,$$

where $\mathcal{U}^\omega \subseteq X^\omega$ is the set of all infinite sequences having some prefix in $\mathcal{U}$. Note that $\mathcal{U}^\omega$ is an open subset of X^ω, so is in particular measurable.

An easier way to define $\mathbb{P}(V,\mathcal{U})$ is that it is the sum of the probabilities of all the sequences in $\mathcal{U}$; i.e.:

$$\mathbb{P}(V,\mathcal{U}) = \sum_{u\in\mathcal{U}} \prod_{i=0}^{|u|-1} \mathbb{P}(u^{(i)})\,,$$

where the infinite sum refers to the supremum of the sums over all finite subsets of $\mathcal{U}$.

Proposition 8.6.2. *Suppose that $(V,\mathcal{U})$ and $(W,\mathcal{V})$ are equivalent pairs, in the sense of Definition 8.4.3, where $V\colon \Omega \to X$, $W\colon \Omega \to Y$ are finitely-supported random variables, and $\mathcal{U} \subseteq X^*$, $\mathcal{V} \subseteq Y^*$ are sets of sequences. Then $\mathbb{P}(V,\mathcal{U}) = \mathbb{P}(W,\mathcal{V})$.*

Proof. Without loss of generality, we may assume that there is a probability-preserving function $f\colon X \to Y$; i.e., a function such that for any $A \subseteq Y$ we have $\mathbb{P}(W \in A) = \mathbb{P}(f(V) \in A)$ and such that $\mathcal{U} = f_*^{-1}(\mathcal{V})$. Then we have

$$\begin{aligned}
\mathbb{P}(V,\mathcal{U}) &= \mathbb{P}(V, f_*^{-1}(\mathcal{V})) \\
&= \sum_{\substack{u\in X^* \\ f_*u\in\mathcal{V}}} \prod_{i=0}^{|u|-1} \mathbb{P}(V = u^{(i)}) \\
&= \sum_{v\in\mathcal{V}} \sum_{\substack{u\in X^* \\ f_*u=v}} \prod_{i=0}^{|u|-1} \mathbb{P}(V = u^{(i)}) \\
&= \sum_{v\in\mathcal{V}} \prod_{i=0}^{|v|-1} \sum_{\substack{x\in X \\ f(x)=v}} \mathbb{P}(V = x) \\
&= \sum_{v\in\mathcal{V}} \prod_{i=0}^{|v|-1} \mathbb{P}(f(V) = v^{(i)}) \\
&= \sum_{v\in\mathcal{V}} \prod_{i=0}^{|v|-1} \mathbb{P}(W = v^{(i)}) \\
&= \mathbb{P}(W,\mathcal{V})\,.
\end{aligned}$$

□

We now define the operational semantics for Probabilistic Algol.

Definition 8.6.3. Let $M\colon \mathtt{com}$ be a closed term of PA mentioning probabilities $p_1, \cdots, p_n$. We define $\mathbb{P}(M \Downarrow)$ to be $\mathbb{P}(B(M))$; i.e.,

$$\mathbb{P}(V_{p_1} \otimes \cdots \otimes V_{p_n}, \mathcal{U}),$$

where $\mathcal{U}$ is the set of all sequences $u \in j(V_{p_1} \otimes \cdots \otimes V_{p_n})^*$ that cover some sequence U such that $M \Downarrow_U \mathsf{skip}$.

We can define a corresponding notion for morphisms in the denotational semantics.

Definition 8.6.4. Let $\sigma\colon 1 \to \mathbb{C}$ be a morphism in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, considered as a morphism $\sigma\colon 1 \to (X \to \mathbb{C})$ in $\mathcal{G}$, together with a finitely-supported random variable $V\colon \Omega \to X$.

Then we define $\mathbb{P}(\sigma \downarrow)$ to be

$$\mathbb{P}(V, \mathrm{Acc}(\sigma)).$$

Remark 8.6.5. By Propositions 8.6.2 and 8.4.4, Definitions 8.6.3 and 8.6.4 are well defined.

Now we are ready to prove computational adequacy.

Proposition 8.6.6 (Computational Adequacy for PA)**.** *Let $M\colon \mathtt{com}$ be a closed term of PA. Then $\mathbb{P}(M \Downarrow) = \mathbb{P}([\![M]\!] \downarrow)$.*

Proof. By Theorem 8.4.6, $B(M)$ is equivalent to $(p, \mathrm{Acc}([\![M]\!]))$. Therefore, by Proposition 8.6.2,

$$\mathbb{P}(M \Downarrow) = \mathbb{P}(B(M)) = \mathbb{P}(V, \mathrm{Acc}([\![M]\!])) = \mathbb{P}([\![M]\!] \downarrow). \qquad \square$$

We can define observational equivalence for terms.

Definition 8.6.7. Let $M, N\colon T$ be closed terms of PA. Then we say that M and N are (probabilistically) *observationally equivalent* if for all contexts $C\colon \mathtt{com}$ with a hole of type T, we have

$$\mathbb{P}(C[M] \Downarrow) = \mathbb{P}(C[N] \Downarrow).$$

We then have the usual corresponding definition in the denotational semantics.

Definition 8.6.8. Let $\sigma, \tau \colon A \to B$ be morphisms in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$. We write $\sigma \sim_{\mathbb{P}} \tau$ if for all morphisms $\alpha \colon (A \to B) \to \mathbb{C}$ in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$ we have

$$\mathbb{P}(\Lambda(\sigma); \alpha \downarrow) = \mathbb{P}(\Lambda(\tau); \alpha \downarrow).$$

Then, by our standard argument, we may derive Equational Soundness from Computational Adequacy.

Proposition 8.6.9. *Let $M, N \colon T$ be closed terms of PA such that $[\![M]\!] \sim_{\mathbb{P}} [\![N]\!]$. Then M and N are probabilistically observationally equivalent.*

Our next goal will be to prove the converse to this result: Full Abstraction.

8.7 Full Abstraction for Probabilistic Algol

In order to prove a definability result, we would like to know that any morphism $A \to B$ in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$ may be considered as a pair

$$(V_{p_1} \otimes \cdots \otimes V_{p_n}, f \colon A \to (\mathbb{B}^n \to B))$$

for appropriately chosen $p_1, \cdots, p_n$.

This is because every morphism definable in PA takes this form. By Proposition 8.5.7, it will suffice to prove the following.

Proposition 8.7.1. *Let $V \colon \Omega \to X$ be a finitely supported random variable. Then there exist $p_1, \cdots, p_n$ and a function*

$$f \colon \mathbb{B}^n \to X$$

such that for all $x \in X$ we have $\mathbb{P}(V = x) = \mathbb{P}(f(V_{p_1}, \cdots, V_{p_n}) = x)$.

Proof. Recall that the V_p are not independent in our formulation; indeed, if $p < q$, then $V_p = \mathtt{t} \Rightarrow V_q = \mathtt{t}$.

Enumerate those elements $x \in X$ such that $\mathbb{P}(V = x) \neq 0$ as $x_1, \cdots, x_n$, and for each $k = 1, \cdots, n$, define

$$p_k = \sum_{i=1}^{n} \mathbb{P}(X = x_i).$$

Note that we must have $p_n = 1$. Then we define

$$f(\vec{b}) = \begin{cases} x_1 & \text{if } \vec{b} = \vec{\mathtt{f}} \\ \min\{k \,:\, b_k = \mathtt{t}\} & \text{otherwise}. \end{cases}$$

Fix $x \in X$. If x is not one of the x_i, then we have $\mathbb{P}(f(V_{p_1}, \cdots, V_{p_n}) = x) = 0 = \mathbb{P}(V = x)$. Otherwise, suppose $x = x_k$. If $\omega \in \Omega$ and $p_{k-1} \leq \omega < p_k$, then $V_{p_k}(\omega) = \mathbb{t}$, and $V_{p_i}(\omega) = \mathbb{f}$ for all $i \leq k$. So $f((V_{p_1} \otimes \cdots \otimes V_{p_n})(\omega)) = x_k$. If $\omega < p_{k-1}$, then $V_{p_{k-1}}(\omega) = \mathbb{t}$, so $f((V_{p_1} \otimes \cdots \otimes V_{p_n})(\omega)) \neq x_k$. If $\omega \geq p_k$, then $V_{p_k}(\omega) = \mathbb{f}$, so $f((V_{p_1} \otimes \cdots \otimes V_{p_n})(\omega)) \neq x_k$. Therefore,

$$\mathbb{P}(f(V_{p_1}, \cdots, V_{p_k}) = x_k) = \mathbb{P}([p_{k-1}, p_k)) = p_k - p_{k-1} = \mathbb{P}(X = x_k)\,.$$

It follows that f is probability preserving in the sense required. □

Remark 8.7.2. This also proves that $\mathbf{Rv}_\Omega^{FS}$ has a small ancestral set.

Then if we have an arbitrary morphism $\sigma\colon A \to B$ (given by a morphism $\tilde{\sigma}\colon A \to (j(X) \to B)$ in the base category), Proposition 8.7.1 gives us the morphism f mediating between $\tilde{\sigma}$ and a morphism of the form specified at the start of this section.

We can now prove our compact definability result.

Proposition 8.7.3 (Compact definability for PA)**.** *Let T be an Idealized Algol type and let $\sigma\colon 1 \to [\![T]\!]$ be a compact morphism in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$. Then there is some closed term $M\colon T$ such that $\sigma = [\![M]\!]$.*

Proof. Let $(V, \sigma\colon 1 \to (X \to [\![T]\!]))$ be a compact representative of σ, where X is a set and V is a finitely-supported random variable taking values in X. By Proposition 8.7.1, we may choose $p_1, \cdots, p_n$ such that there is a probability-preserving function

$$f\colon V_{p_1} \otimes \cdots \otimes V_{p_n} \to V\,.$$

After composing on the right by $(f \to [\![T]\!])$, we may assume that σ is of the form

$$(V_{p_1} \otimes \cdots \otimes V_{p_n}, \sigma\colon 1 \to (\underline{\mathbb{B}^n} \to [\![T]\!]))\,.$$

Now this σ necessarily factors as

$$1 \xrightarrow{\hat{\sigma}} (\underline{\mathbb{B}^n} \to [\![T]\!]) \xrightarrow{(m \to [\![T]\!])} (\underline{\mathbb{B}^n} \to [\![T]\!])\,,$$

where $\hat{\sigma}$ is compact. Then, by compact definability in $\mathcal{G}$, $\hat{\sigma}$ is the denotation of some term $N\colon \mathtt{bool} \to \cdots \to \mathtt{bool} \to T$, and it follows that our original morphism σ in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$ is the denotation of

$$N\ \mathsf{choose}_{p_1}\ \cdots\ \mathsf{choose}_{p_n}\ .$$ □

Theorem 8.7.4 (Full Abstraction for PA)**.** *Let $M, N\colon T$ be observationally equivalent terms of PA. Then $[\![M]\!] \sim_{\mathbb{P}} [\![N]\!]$.*

Proof. Suppose that $[\![M]\!] \not\sim_{\mathbb{P}} [\![N]\!]$. So there is some $\alpha\colon [\![T]\!] \to \mathbb{C}$ such that $\mathbb{P}([\![M]\!]\,;\alpha\downarrow) \neq \mathbb{P}([\![N]\!]\,;\alpha\downarrow)$.

Let $\mathbb{P}([\![M]\!]\,;\alpha\downarrow) = p$ and $\mathbb{P}([\![N]\!]\,;\alpha\downarrow) = q$, and suppose without loss of generality that $p > q$. Now there must be some finite subset $\mathcal{V}$ of $\mathrm{Acc}([\![M]\!]\,;\alpha\downarrow)$ such that the combined probability of the sequences in $\mathcal{V}$ is still greater than q. For each $u \in \mathcal{V}$, we can choose some compact $\alpha_u \subseteq \alpha$ such that u is still accepted by $[\![M]\!]\,;\alpha_u$, by algebraicity. Since the set of compact elements below α is directed, there is some $\alpha' \subseteq \alpha$ such that $\alpha_u \subseteq \alpha'$ for each $u \in \mathcal{V}$. Then we have

$$\mathbb{P}([\![M]\!]\,;\alpha'\downarrow) > q \qquad\qquad \mathbb{P}([\![N]\!]\,;\alpha'\downarrow) \leq q\,,$$

and therefore $\mathbb{P}([\![M]\!]\,;\alpha'\downarrow) \neq \mathbb{P}([\![N]\!]\,;\alpha'\downarrow)$.

By Proposition 8.7.3, α' is the denotation of some term $L\colon T \to \mathtt{com}$, and our Computational Adequacy result (Proposition 8.6.6) then tells us that

$$\mathbb{P}(L\,M\Downarrow) = \mathbb{P}([\![M]\!]\,;\alpha'\downarrow) \neq \mathbb{P}([\![N]\!]\,;\alpha'\downarrow) = \mathbb{P}(L\,N\Downarrow)\,.$$

Therefore, M and N are not observationally equivalent. □

8.8 Comparison with a Kleisli Category Model

The probabilistic language is our main example of an application of the theory of parametric monads that we have developed. However, it is worth noting that it is possible to model a probabilistic language within the language $\mathrm{IA}_{\mathbb{B}}$ from Chapter 4. Specifically, we can consider the language $\mathrm{IA}_{\mathbb{B}}$ as a probabilistic Algol variant, by treating the term $\mathsf{ask}_{\mathbb{B}}$ as a coin flip that returns $\mathbb{t}$ or $\mathbb{f}$ each with probability $\frac{1}{2}$.

Given a closed term $M\colon \mathtt{com}$ of $\mathrm{IA}_{\mathbb{B}}$, we define

$$\mathbb{P}(M\Downarrow) = \sum_{u\in\mathbb{B}^*\colon\, M\Downarrow_u \mathsf{skip}} 2^{-|u|}\,,$$

since $2^{-|u|}$ is the probability of a particular sequence u of $\mathbb{t}$ and $\mathbb{f}$ values occurring. Here, the infinite sum means the supremum over all sums over finite subsets. Similarly, given a Kleisli morphism $\sigma\colon 1 \to \mathbb{C}$ – i.e., a morphism $\sigma\colon 1 \to (\mathbb{B} \to \mathbb{C})$ in $\mathcal{G}$, we can define

$$\mathbb{P}(\sigma\downarrow) = \sum_{u\in\mathrm{Acc}(\sigma)} 2^{-|u|}\,.$$

Our Computational Adequacy result for IA_X (Propositions 4.6.9 and 4.7.4) then gives us a Computational adequacy result for this model.

Proposition 8.8.1. *Let $M\colon \mathtt{com}$ be a closed term of $IA_{\mathbb{B}}$. Then $\mathbb{P}(M \Downarrow) = \mathbb{P}([\![M]\!] \downarrow)$.*

Proof. Propositions 4.6.9 and 4.7.4 tell us that the set of sequences u such that $M \Downarrow_u \mathsf{skip}$ is the same as the set $\mathrm{Acc}([\![M]\!])$. □

We can define probabilistic observational equivalence and the probabilistic intrinsic equivalence $\sim_{\mathbb{P}}$ in exactly the same way as we did for PA. Then the same argument we used in Theorem 8.7.4 proves Full Abstraction for this model.

Theorem 8.8.2. *Let $M, N\colon T$ be closed terms of $IA_{\mathbb{B}}$. Then M and N are probabilistically observationally equivalent if and only if $[\![M]\!] \sim_{\mathbb{P}} [\![N]\!]$.*

Since this model relies on a lot less theory, it is worth spending a bit of time thinking about what our existing parametric monad model gives us that this one doesn't.

The most obvious answer is that our original model allowed us to work with arbitrary probabilities, rather than using a fixed coin with probability $\frac{1}{2}$. However, this is not such a great advantage as it might seem, since if $p \in [0,1]$ is any real number whose binary expansion is computable as a function $\mathbb{N} \to \mathbb{B}$, then we can simulate choose_p within the probabilistic version of $\mathrm{IA}_{\mathbb{B}}$[1].

Perhaps a better way of thinking about the difference between the two models, then, is to consider what the denotation of a term actually looks like. For comparison, we look at the denotations of the term $\mathsf{choose}_{\frac{2}{3}}$ in the two different semantics.

In the language $\mathrm{IA}_{\mathbb{B}}$, we can define a term that converges to $\mathtt{t}$ with probability $\frac{2}{3}$ and to $\mathtt{f}$ with probability $\frac{1}{3}$ by

$$\mathbf{Y}_{\mathtt{bool}}(\lambda b.\, \mathsf{If}\ \mathsf{choose}_{\frac{1}{2}}\ \mathsf{then}\ \mathtt{t}\ \mathsf{else}\ (\mathsf{If}\ \mathsf{choose}_{\frac{1}{2}}\ \mathsf{then}\ b\ \mathsf{else}\ \mathtt{f}))\,.$$

Here, we have renamed $\mathsf{ask}_{\mathbb{B}}$ to $\mathsf{choose}_{\frac{1}{2}}$ to give a better idea of what the term does in the probabilistic setup.

Now the denotation of this term in $\mathrm{Kl}_{R_{\mathbb{B}}}\mathcal{G}$ is given by the denotation of the term-in-context

$$c\colon \mathtt{bool} \vdash \mathbf{Y}_{\mathtt{bool}}(\lambda b.\, \mathsf{If}\ c\ \mathsf{then}\ \mathtt{t}\ \mathsf{else}\ (\mathsf{If}\ c\ \mathsf{then}\ b\ \mathsf{else}\ \mathtt{f}))$$

[1] Idea: build up a sequence of intervals I_n of width 2^n by repeatedly flipping the coin and using the result to choose either the lower or the upper half of I_{n-1}. If at any point the upper bound of I_n is less than or equal to p (which can be checked by computing the first n terms of the binary expansion of p), then return $\mathtt{t}$. If at any point the lower bound of I_n is greater than or equal to p, return $\mathtt{f}$.

in $\mathcal{G}$. If $\mathcal{G}$ is the category of games, then this morphism is the strategy with maximal plays taking one of the following two forms.

$$q(q\mathbb{f}q\mathbb{t})^n q\mathbb{t}\mathbb{t} \qquad\qquad q(q\mathbb{f}q\mathbb{t})^n q\mathbb{f}q\mathbb{f}\mathbb{f}$$

In other words, it is not at all clear from the denotation that the term should give $\mathbb{t}$ with probability $\frac{2}{3}$ and $\mathbb{f}$ with probability $\frac{1}{3}$.

In $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, however, we can model a term that has the same behaviour using the morphism

$$\left(V_{\frac{2}{3}}, \mathrm{id}_{\mathbb{B}}\right),$$

which makes it much clearer what the probabilistic behaviour is.

Let us now examine two more models based on Kleisli categories.

One idea we might have is to use the datatype game $(0,1)$ (corresponding to the open unit interval of reals) as our reader object. That way, we could model the term choose_p as the morphism

$$(0,1) \to \mathbb{B}$$

that returns $\mathbb{t}$ if its input is less than p and $\mathbb{f}$ if its input is greater than or equal to p. Unfortunately, however, this model allows us to construct too many strategies, including some that have no probabilistic interpretation. If $X \subseteq (0,1)$ is a non-measurable set, for instance, then there is a perfectly valid morphism[2]

$$\chi_X \colon (0,1) \to \mathbb{B}$$

corresponding to the indicator function of X, but there is now no sensible answer to the question 'What is the probability that χ_X returns $\mathbb{t}$?'[3] Fixing this problem would require us to impose further measurability conditions on strategies, taking us away from our core idea of sticking to the category-theoretic constructions as closely as possible.

A more successful approach is to replace the datatype $(0,1)$ with the (game-theoretic) product of a countably infinite collection of copies of $\mathbb{B}$, indexed by $(0,1)$, each corresponding to one of the choose_p. Computationally, this corresponds to 'encapsulating' the chosen value in $(0,1)$ behind a countably infinite family of 'methods', each of which will only tell you whether the value is less than some fixed number. Since any program can call only

[2] At least, in the category of games and any other model satisfying the condition from Definition 2.15.1.

[3] Recall also from Section 6.4 that this Kleisli category is precisely what we end up with if we relax the discreteness condition on random variables – so we are left trying to model continuous probability distributions (such as the uniform distribution on $(0,1)$) with games that are very much discrete.

finitely many of these 'methods', we avoid introducing any non-measurable behaviour.

There is then, at least in the category of games, some clear idea about how to define the probability of a particular play occurring in a strategy. Since any play must make at most countably many appeals to the product-of-booleans oracle, and since each such appeal has an associated probability, we can associate a probability to each such play by multiplying these probabilities together.

It is less clear, however, how to prove Adequacy and Full Abstraction for this model. The approach that we have adopted in this chapter – a translation into IA_X– does not work for a strategy of the form

$$\sigma\colon \prod_{p\in(0,1)} \mathbb{B} \to A\,,$$

where the left-hand side is an uncountable product of booleans and certainly cannot be embedded inside any IA datatype game.

We could reason about such a strategy by asking whether it factored through any finite product of booleans; i.e., whether there was any $\hat{\sigma}$ such that σ factors as

$$\prod_{p\in(0,1)} \mathbb{B} \xrightarrow{\langle \mathrm{pr}_{p_1},\cdots,\mathrm{pr}_{p_n}\rangle} \mathbb{B}^n \xrightarrow{\hat{\sigma}} A\,,$$

for some finite collection $p_1,\cdots,p_n$ of probabilities. We could then reason about the strategy $\hat{\sigma}$. But if we are doing that, then why not reason about the strategy $\hat{\sigma}$ from the start? And if we have two possible choices for this $\hat{\sigma}$, why not make it explicit in the model that they are equal? This is exactly what our $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$ model is doing. Rather than try and rely on some global probabilistic oracle, our strategies can form their own mini (i.e., finite/countable) oracles that give them what they need. Composition of morphisms automatically groups these mini oracles together into one, and the equivalence relation on morphisms ensures that we can always enlarge our mini oracle if we need to (for example, by converting from a product of finitely many booleans to a natural number to help the translation into IA_X).

8.9 Game Semantics and Probability

So far, we have considered things in the abstract. We now specialize to the case that $\mathcal{G}$ is the category of arenas and single-threaded strategies that we developed in Chapters 2 and 3. This will allow us to capture the close relationship between the sequences u that we have been considering and the plays in a strategy.

Definition 8.9.1. Let X be a set and let $u \in X^*$ be a sequence. Consider X as an arena $\underline{X}$. Then we write qu for the play in $\underline{X}$ given by

$$q\, u^{(1)} \cdots q\, u^{(|u|-1)}\,.$$

Note that any P-position in $\underline{X}$ is of the form qu for some sequence u.

Definition 8.9.2. Let A be an arena, let V be a random variable taking values in a set X, and let $\sigma\colon X \to A$ be a single-threaded strategy. We may consider X as a game $\underline{X}$. Let s be a legal play of A. If $t \in \sigma$, we write t/s if $t|_A = s$ and if the last move of t is the last move of s (this implies in particular that $t|_{\underline{X}}$ is a P-position in $\underline{X}$). Then we define

$$\mathrm{Acc}_s(\sigma) = \{u \in X^* :\ \exists t \in \sigma\,.\, t/s,\, t|_{\underline{X}} = qu\}\,.$$

We define

$$\mathbb{P}_V(s \in \sigma) = \mathbb{P}(V, \mathrm{Acc}_s(\sigma))\,.$$

We would like use this definition to define $\mathbb{P}(s \in \sigma)$ for σ a morphism in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, but we first need to check that this is well-defined with respect to the equivalence relation on Melliès morphisms.

Proposition 8.9.3. *Let*

$$(V\colon \Omega \to X, \sigma\colon \underline{X} \to (A \to B)) \qquad (W\colon \Omega \to Y, \sigma\colon \underline{Y} \to (A \to B))$$

be representatives of the same morphism $A \to B$ in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, where X and Y are sets. Let s be a legal play of $A \to B$. Then $(V, \mathrm{Acc}_s(\sigma))$ and $(W, \mathrm{Acc}_s(\sigma'))$ are equivalent as pairs in the sense of Definition 8.4.3.

Proof. It suffices by induction to prove this in the case that the two representatives are related by the relation that generates the equivalence relation on morphisms; i.e., that there is a probability-preserving function $f\colon X \to Y$ such that $\sigma = \sigma'; (\underline{j(f)} \to A)$.

Let $u \in X^*$. Then we have

$$\begin{aligned} u \in \mathrm{Acc}_s(\sigma) &\Leftrightarrow \exists t \in \sigma\,.\, t/s,\, t|_{\underline{X}} = qu \\ &\Leftrightarrow \exists t \in \sigma'; (\underline{j(f)} \to A)\,.\, t/s,\, t|_{\underline{X}} = qu \\ &\Leftrightarrow \exists t' \in \sigma'\,.\, t/s,\, t|_{\underline{Y}} = q(f_* u) \\ &\Leftrightarrow f_* u \in \mathrm{Acc}_s(\sigma')\,. \end{aligned}$$

Therefore, $(V, \mathrm{Acc}_s(\sigma))$ and $(W, \mathrm{Acc}_s(\sigma'))$ are equivalent. □

It follows by Proposition 8.6.2 that $\mathbb{P}_V(s \in \sigma) = \mathbb{P}_W(s \in \sigma')$ for all s. Therefore, the following is well-defined.

Definition 8.9.4. Let $\sigma\colon A \to B$ be a morphism in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, where $\mathcal{G}$ is the category of arenas and single-threaded strategies, and suppose that σ is given (after currying) by a morphism

$$\tilde{\sigma}\colon \underline{X} \to (A \to B)$$

in $\mathcal{G}$, together with a random variable V taking values in X. Then we define

$$\mathbb{P}(s \in \sigma) = \mathbb{P}_V(s \in \tilde{\sigma})\,.$$

Definition 8.9.5. Let $\sigma, \sigma'\colon A \to B$ be morphisms in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$, where $\mathcal{G}$ is the category of arenas and single-threaded strategies. We say that $\sigma \approx_{\mathbb{P}} \sigma'$ if for all legal plays s of $A \to B$ we have

$$\mathbb{P}(s \in \sigma) = \mathbb{P}(s \in \sigma')\,.$$

We now relate this definition to Melliès composition of strategies.

Definition 8.9.6. Let A, B, C be arenas and let s be a play in $A \to C$. We write

$$\mathrm{wit}_B(s) = \{\mathfrak{s} \in \mathrm{int}(A, B, C) \,:\, \mathfrak{s}|_{A,C} = s\}\,.$$

Proposition 8.9.7. *Let $\sigma\colon A \to B$, $\tau\colon B \to C$ be morphisms in the category $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$. Let s be a legal play in $A \to C$. Then*

$$\mathbb{P}(s \in \sigma;\tau) = \sum_{\mathfrak{s} \in \mathrm{wit}_B(s)} \mathbb{P}(\mathfrak{s}|_{A,B} \in \sigma)\mathbb{P}(\mathfrak{s}|_{B,C} \in \tau)\,.$$

Proof. Suppose that σ and τ are given by (equivalence classes of) pairs

$$(V\colon \Omega \to X, \tilde{\sigma}\colon A \to (\underline{X} \to B)) \qquad (W\colon \Omega \to Y, \tilde{\tau}\colon B \to (\underline{Y} \to C))\,,$$

where V and W are random variables and $\tilde{\sigma}, \tilde{\tau}$ are strategies in $\mathcal{G}$. Then the composition $\sigma;\tau$ in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$ is given by $V \otimes W\colon \Omega \to X \times Y$, together with the Melliès composition

$$A \xrightarrow{\tilde{\sigma}} (\underline{X} \to B) \xrightarrow{\underline{X} \to \tilde{\tau}} (\underline{X} \to (\underline{Y} \to C)) \to ((\underline{X} \times \underline{Y}) \to C)$$

$$\xrightarrow{\mu \to C} (\underline{X \times Y} \to C)\,,$$

where μ is $\langle \underline{\mathrm{pr}_X}, \underline{\mathrm{pr}_Y} \rangle$.

First suppose that s is a sequence in $A \to C$, and let $\mathfrak{s} \in \mathrm{wit}_B(s)$. Let $\mathfrak{t}$ be a sequence in $\tilde{\sigma} \| (\underline{X} \to \tau)$ such that $\mathfrak{t}|_{A,B,C} = \mathfrak{s}$. Then $\mathfrak{t}|_{A,\underline{X} \to B} \in \tilde{\sigma}$ and $\mathfrak{t}|_{B,\underline{Y},C} \in \tilde{\tau}$.

Moreover, since we have $\mathfrak{t}|_{A,B} = \mathfrak{s}|_{A,B}$ and $\mathfrak{t}|_{B,C} = \mathfrak{s}|_{B,C}$, we must have

$$\mathfrak{t}|_{\underline{X}} \in \mathrm{Acc}_{\mathfrak{s}|_{A,B}}(\tilde{\sigma}) \qquad\qquad \mathfrak{t}|_{\underline{Y}} \in \mathrm{Acc}_{\mathfrak{s}|_{B,C}}(\tilde{\tau}) \,,$$

where we have identified a play qu occurring in the arena $\underline{X}$ with its underlying sequence u of elements of X, and likewise for Y.

This gives us a function

$$\{\mathfrak{t} \in \tilde{\sigma} \| (\underline{X} \to \tilde{\tau}) \,:\, \mathfrak{t}|_{A,B,C} = \mathfrak{s}\} \to \mathrm{Acc}_{\mathfrak{s}|_{A,B}}(\tilde{\sigma}) \times \mathrm{Acc}_{\mathfrak{s}|_{B,C}}(\tilde{\tau}) \,.$$

We claim that this function is a bijection.

Indeed, suppose that $\mathfrak{t}, \mathfrak{t}'$ are two interactions in $\tilde{\sigma} \| (\underline{X} \to \tilde{\tau})$ such that $\mathfrak{t}|_{A,B,C} = \mathfrak{t}'|_{A,B,C} = \mathfrak{s}$, $\mathfrak{t}|_{\underline{X}} = \mathfrak{t}'|_{\underline{X}}$ and $\mathfrak{t}|_{\underline{Y}} = \mathfrak{t}'|_{\underline{Y}}$. Next we claim that $\mathfrak{t} = \mathfrak{t}'$.

To see why, suppose for a contradiction that $\mathfrak{t} \neq \mathfrak{t}'$: then there are prefixes $\mathfrak{r}p \sqsubseteq \mathfrak{t}$ and $\mathfrak{r}q \sqsubseteq \mathfrak{t}'$, where $\mathfrak{r}$ is the longest common subsequence of $\mathfrak{t}$ and $\mathfrak{t}'$ and $p \neq q$ are moves.

By our earlier analysis (see, for example, the proof of 2.4.9), we know that p and q must either both occur in the $A \to (\underline{X} \to B)$-component, or both in the $(\underline{X} \to B) \to (\underline{X} \to (\underline{Y} \to C))$-component. But since $\mathfrak{t}, \mathfrak{t}' \in \tilde{\sigma} \| (\underline{X} \to \tilde{\tau})$ are both interactions of deterministic strategies, we also know that they must both be O-moves in that component – otherwise, they would have to be equal. In particular, neither p nor q may be a move in the middle component $\underline{X} \to B$, since then it would be a P-move in one of the two components.

Therefore, p and q are both O-moves in one of the outer components A and $\underline{X} \to (\underline{Y} \to C)$. By Corollary 2.4.4, we know that only Player P may switch between games in $\underline{X} \to (\underline{Y} \to C)$, and therefore p and q must occur in the same component game – i.e., both in A, both in $\underline{X}$, both in $\underline{Y}$ or both in C. But now the conditions that $\mathfrak{t}|_{A,B,C} = \mathfrak{t}'|_{A,B,C}$, $\mathfrak{t}|_{\underline{X}} = \mathfrak{t}'|_{\underline{X}}$ and $\mathfrak{t}|_{\underline{Y}} = \mathfrak{t}'|_{\underline{Y}}$ mean that we must have $p = q$. For example, if p and q are both moves in C, then we have $\mathfrak{r}|_C p \sqsubseteq \mathfrak{t}|_C$ and $\mathfrak{r}|_C q \sqsubseteq \mathfrak{t}'|_C$; since $\mathfrak{t}|_C = \mathfrak{t}'|_C$, we must have $p = q$. This is the desired contradiction.

For surjectivity, let $u \in \mathrm{Acc}_{\mathfrak{s}|_{A,B}}(\sigma)$ and $v \in \mathrm{Acc}_{\mathfrak{s}|_{B,C}}(\tau)$. We seek a sequence $\mathfrak{t} \in \tilde{\sigma} \| (\underline{X} \to \tilde{\tau})$ such that $\mathfrak{t}|_{A,B,C} = \mathfrak{s}$, $\mathfrak{t}|_{\underline{X}} = qu$ and $\mathfrak{t}|_{\underline{Y}} = qv$.

Since $u \in \mathrm{Acc}_{\mathfrak{s}|_{A,B}}(\sigma)$, there is some sequence $s \in \sigma$ such that $s|_{A,B} = \mathfrak{s}|_{A,B}$ and $s|_{\underline{X}} = qu$. Similarly, since $v \in \mathrm{Acc}_{\mathfrak{s}|_{B,C}}(\tau)$, there is some sequence $t \in \tau$ such that $t|_{B,C} = \mathfrak{s}|_{B,C}$ and $t|_{\underline{Y}} = qv$. We form the sequence $\mathfrak{t}$ as a suitable interleaving of s and t, noting that they have the same B-components: we

start with the sequence $\mathfrak{s}$, and then insert the appropriate moves from (the left-hand copy of) $\underline{X}$ and $\underline{Y}$ between the corresponding moves from A, B and C. Lastly, we insert moves from the right-hand copy of $\underline{X}$ adjacent to the corresponding moves in the right-hand copy, inserting an O-move q in the right-hand copy of $\underline{X}$ immediately after each O-move q in the left-hand copy, and a P-move x in the right-hand copy immediately before each P-move x in the left-hand copy.

This tells us that we have

$$\begin{aligned}
&\mathbb{P}(\mathfrak{s}|_{A,B} \in \sigma)\mathbb{P}(\mathfrak{s}|_{B,C} \in \tau) \\
&= \left(\sum_{u \in \mathrm{Acc}_{\tilde{\sigma}}(\mathfrak{s}|_{A,B})} \prod_{i=0}^{|u|-1} \mathbb{P}(V = u^{(i)})\right)\left(\sum_{v \in \mathrm{Acc}_{\tilde{\tau}}(\mathfrak{s}|_{A,B})} \prod_{i=0}^{|v|-1} \mathbb{P}(W = v^{(i)})\right) \\
&= \sum_{(u,v) \in \mathrm{Acc}_{\tilde{\sigma}}(\mathfrak{s}|_{A,B}) \times \mathrm{Acc}_{\tilde{\tau}}(\mathfrak{s}|_{B,C})} \prod_{i=0}^{|u|-1} \mathbb{P}(V = u^{(i)}) \prod_{i=1}^{|v|-1} \mathbb{P}(W = v^{(i)}) \\
&= \sum_{\substack{\mathfrak{t} \in \tilde{\sigma} \| (\underline{X} \to \tilde{\tau}) \\ \mathfrak{t}|_{A,B,C} = \mathfrak{s}}} \left[\prod_{i=1}^{|u|-1} \mathbb{P}(V = u^{(i)}) \prod_{i=0}^{|v|-1} \mathbb{P}(W = v^{(i)}) \;\middle|\; \begin{array}{l} \text{where} \\ \mathfrak{t}|_{\underline{X}} = qu \\ \mathfrak{t}|_{\underline{Y}} = qv \end{array}\right].
\end{aligned}$$

Now let $s \in \mathcal{L}_{A \to C}$ and let $t \in \sigma;\tau$ be such that t/s. By the argument in Proposition 2.4.9, there is a unique interaction sequence

$$S \in \sigma \| (\underline{X} \to \tau) \| (\Lambda^{-1}; \mu)$$

such that $S|_{A,\underline{X \times Y} \to C} = t$. Define $\mathfrak{t} = S|_{A,\underline{X} \to B,\underline{X} \to (\underline{Y} \to C)}$, and $\mathfrak{s} = \mathfrak{t}|_{A,B,C}$. Now we must have $\mathfrak{s}|_{A,C} = t|_{A,C} = s$, so $\mathfrak{s} \in \mathrm{wit}_B(s)$.

Consider the sequence $\mathfrak{t}|_{\underline{X},\underline{Y}}$, which has the same length as $S|_{\underline{X \times Y}} = t|_{\underline{X \times Y}}$. This sequence is made up of pairs of moves qx for $x \in X$ or qy for $y \in Y$.

By the definition of μ, we know that if the i-th pair of moves in $\mathfrak{t}|_{\underline{X},\underline{Y}}$ is qx for $x \in X$, then the i-th pair of moves in $t|_{\underline{X \times Y}}$ is $q(x, y_0)$ for some y_0, and if the i-th pair of moves in $\mathfrak{t}|_{\underline{X},\underline{Y}}$ is qy for $y \in Y$, then the i-th pair of moves in $t|_{\underline{X \times Y}}$ is $q(x_0, y)$ for some $x_0 \in X$. Moreover, if t' is some other sequence such that t'/s and such that t' differs only from t in the choice of the 'irrelevant' moves x_0, y_0, then there is some $S' \in \sigma \| (\underline{X} \to \tau) \| (\Lambda^{-1}; \mu)$ such that $S|_{A,\underline{X \times Y} \to C} = t'$ and $S|_{A,\underline{X} \to (\underline{Y} \to C)} = \mathfrak{t}$. Then the combined probability

of all such sequences is given by

$$\sum_{\substack{\mathfrak{t}\in\tilde{\sigma}\|(\underline{X}\to\tilde{\tau})\\ \mathfrak{t}|_{A,B,C}=\mathfrak{s}}}\left[\prod_{i=1}^{|u|-1}\mathbb{P}(V=u^{(i)})\prod_{i=0}^{|v|-1}\mathbb{P}(W=v^{(i)})\;\middle|\;\begin{array}{l}\text{where}\\ \mathfrak{t}|_{\underline{X}}=qu\\ \mathfrak{t}|_{\underline{Y}}=qv\end{array}\right],$$

and we can combine this with our calculations above to get the desired result. □

A consequence of Proposition 8.9.7 is that the probabilistic equivalence relation defined in 8.9.5 is respected by composition of strategies. We may therefore take the quotient of this category by the probabilistic equivalence relation to form a new category

$$(\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}})\backslash\approx_{\mathbb{P}}\ .$$

Then next proposition shows that we do not lose anything by doing this, since the probabilistic equivalence relation gets subsumed into our intrinsic equivalence.

Proposition 8.9.8. *Suppose that $[\sigma],[\tau]\colon A\to B$ are morphisms in the quotiented category, given by equivalence classes of strategies in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\mathrm{op}}$. Then $[\sigma]$ and $[\tau]$ are probabilistically intrinsically equivalent in the quotiented category if and only if σ and τ are probabilistically intrinsically equivalent in the original category.*

Proof. If σ,τ are not intrinisically equivalent, then there is some $\alpha\colon(A\to B)\to\mathbb{C}$ such that $\mathbb{P}(\Lambda^{-1}(\sigma);\alpha\downarrow)\neq\mathbb{P}(\Lambda^{-1}(\tau);\alpha;\downarrow)$. Then we have $\mathbb{P}(qa\in\Lambda^{-1}(\sigma);\alpha)\neq\mathbb{P}(qa\in\Lambda^{-1}(\tau);\alpha)$, and therefore $\Lambda^{-1}(\sigma);\alpha\not\approx_{\mathbb{P}}\Lambda^{-1}(\tau);\alpha$.

Conversely, if $[\sigma],[\tau]$ are not intrinsically equivalent in the quotiented category, then there is some $[\alpha]\colon(A\to B)\to\mathbb{C}$ such that

$$[\Lambda^{-1}(\sigma);\alpha]\not\approx_{\mathbb{P}}[\Lambda^{-1}(\tau);\alpha]\,.$$

It follows that

$$\mathbb{P}(qa\in\Lambda^{-1}(\sigma);\alpha)\neq\mathbb{P}(qa\in\Lambda^{-1}(\tau);\alpha)\,;$$

i.e., that

$$\mathbb{P}(\Lambda^{-1}(\sigma);\alpha\downarrow)\neq\mathbb{P}(\Lambda^{-1}(\tau);\alpha\downarrow)\,,$$

and so σ and τ are not intrinsically equivalent in the original category. □

8.10 The Probabilistic Game Semantics of Danos and Harmer

The work in the previous section shows that two strategies that are related by the probabilistic equivalence relation $\approx_{\mathbb{P}}$ are automatically observationally equivalent. This suggests that the real content of a probabilistic strategy consists in the quantities $\mathbb{P}(s \in \sigma)$. An idea, then, is to take the quantities $\mathbb{P}(s \in \sigma)$ as primitives, rather than defining them indirectly.

That is the approach taken by Danos and Harmer in [DH00] – the original game semantics for a probabilistic variant of Algol. Danos and Harmer define an arena as we do, but define a probabilistic strategy on an arena A to be given by a function

$$\sigma \colon \mathcal{L}_A^{\text{even}} \to [0,1]$$

such that for any even-length s, and any extension sa, we have

$$\sigma(s) \geq \sum_{sab \in \mathcal{L}_A} \sigma(sab)\,.$$

for any even-length s, and any extension sa, we have

$$\sigma(s) \geq \sum_{sab \in \mathcal{L}_A} \sigma(sab)\,.$$

These quantities $\sigma(s)$ take the role of our $\mathbb{P}(s \in \sigma)$, but now they are part of the *definition* of the strategy σ, rather than being a calculated quantity.

Proposition 8.10.1. *For any Melliès strategy $\sigma \colon A \to B$ given by a random variable V taking values in a set X and a strategy $\tilde{\sigma} \colon A \to (\underline{X} \to B)$, we get a probabilistic strategy in the sense of Danos and Harmer by setting*

$$\sigma(s) = \mathbb{P}(s \in \sigma)\,.$$

Proof. We need to check that for any odd-length legal play $sa \in \mathcal{L}_A$ we have

$$\mathbb{P}(s \in \sigma) \geq \sum_{sab \in \mathcal{L}_A} \mathbb{P}(sab \in \sigma)\,.$$

But for any $u \in \mathrm{Acc}_{sab}(\sigma)$, there must be some prefix v of u such that $v \in \mathrm{Acc}_s(\sigma)$. Then all the sequences u arising in this way that have v as a prefix are pairwise incomparable (since $\tilde{\sigma}$ is a deterministic strategy), and so their combined probability is at most the probability of v. □

Clearly, two morphisms in $\mathcal{G}/(\mathbf{Rv}_\Omega^{FS})^{\text{op}}$ give rise to the same probabilistic strategy in this way if and only if they are probabilistically equivalent.

Danos and Harmer then *define* the composition of two probabilistic strategies using the formula that we derived in Proposition 8.9.7:

Definition 8.10.2. Let $\sigma\colon A \to B$, $\tau\colon B \to C$ be probabilistic strategies. Then their composition $\sigma;\tau$ is given by

$$(\sigma;\tau)(s) = \sum_{\mathfrak{s}\in \mathrm{wit}_B(s)} \sigma(\mathfrak{s}|_{A,B})\tau(\mathfrak{s}|_{B,C})\,.$$

By Proposition 8.9.7, the composition of strategies in the quotiented category agrees with this composition of Danos and Harmer. Our proof above then gives an alternative proof of Full Abstraction for Danos and Harmer's probabilistic game semantics.

Chapter 9

Further Directions

We conclude by examining a few avenues that we are left unexplored in this thesis.

9.1 Stateless Languages

So far, the base language we have used has been the stateful language Idealized Algol. One further question to ask is whether the techniques we have developed allow us to build models of effectful versions of stateless languages such as PCF.

One immediate problem we face is that the addition of nondeterministic effects to PCF allows us to distinguish terms that cannot be distinguished within PCF itself. For instance, the PCF term

$$M = \lambda x^{\mathtt{bool}}.\, \mathsf{If}\ x\ \mathsf{then}\ \Omega\ \mathsf{else}\ (\mathsf{If}\ x\ \mathsf{then}\ \mathtt{t}\ \mathsf{else}\ \Omega)$$

is observationally equivalent (in PCF) to Ω – informally, because the term x must have the same 'value' both time it is called, so if it is false the first time, then it must be false the second time. However, the same term inside PCF with finite nondeterminism is not observationally equivalent to Ω: if we substitute a nondeterministic oracle in for x, then the nondeterministic oracle could return false the first time and true the second, causing the term to converge to $\mathtt{t}$. Note that this problem does not arise in Idealized Algol: nondeterminism does not allow us to distinguish terms that were indistinguishable in the deterministic language, as we proved in Section 4.16.

This means that if $\mathcal{G}$ is a 'truly Fully Abstract' model of PCF – i.e., one in which observational equivalence of terms corresponds to equality of morphisms rather than intrinsic equivalence[1] – then there can be no natural

[1] As an example, take any model that is full abstract in our sense, and take the quotient by the intrinsic equivalence relation.

inclusion functor from $\mathcal{G}$ into a model of PCF with finite nondeterminism. Indeed, in such a model $[\![M]\!]$ and $[\![\Omega]\!]$ are the same morphism, so they would have to be sent to the same morphism in the model of nondeterministic PCF.

This is a problem for us, since it means that our model of PCF cannot be the Kleisli category for a monad on $\mathcal{G}$ (or a category of the form $\mathcal{G}/\mathcal{X}$), since then we *do* have a functor coming out of $\mathcal{G}$.

A way to get around this program is to insist that our category $\mathcal{G}$ is not 'truly Fully Abstract', but is still fine-grained enough to distinguish between terms that will eventually be distinguished when we add in the nondeterminism. A convenient requirement to make is that $\mathcal{G}$ should be embeddable into some model $\mathcal{G}^{IA}$ of Idealized Algol (as is the case for Hyland and Ong's game semantics). The idea is that since $\mathcal{G}^{IA}$ must necessarily distinguish between observationally inequivalent terms of nondeterministic PCF, then $\mathcal{G}$ must distinguish between them too.

The other advantage of going via an Idealized Algol model is that we can automatically apply many of the results that we have already obtained.

Let us try this out with an example. Fix $X \in \{\mathbb{B}, \mathbb{N}\}$ and let *Nondeterministic PCF* be the language PCF, together with a constant $\mathsf{ask}_X : \mathtt{bool}$ that plays the role of a nondeterministic oracle. It is the language with types given by

$$T ::= \mathtt{bool} \mid \mathtt{nat} \mid T \to T\,,$$

and with a type theory as shown in Figure 9.1. We endow this language with an operational semantics of may testing. Define a *canonical form* of the language to be a term taking one of the following forms.

- At type $\mathtt{bool}$, the constants $\mathbb{t}$ and $\mathbb{f}$;
- at type $\mathtt{nat}$, the numerals n; and
- at type $S \to T$, terms of the form $\lambda x^S.M$.

We then define a relation $M \Downarrow c$ (read 'M *may converge to* c'), for M a term of Nondeterministic PCF and c a canonical form, inductively as in Figure 9.2. This rule is the restriction of the $\Downarrow$ rule from Nondeterministic Idealized Algol to the set of those terms that are terms of Nondeterministic PCF.

It is a quick check to see that every term of Nondeterministic PCF may be regarded as a term of IA_X, giving us a denotational semantics of the Nondeterministic PCF within $\mathrm{Kl}_{R_X}\,\mathcal{G}$, where $\mathcal{G}$ is the category of arenas and single-threaded strategies. But we can be more precise than this: we know from [HO00] that the denotation of any term of ordinary PCF is an innocent strategy, so that the denotational semantics of PCF within $\mathcal{G}$ factors through the inclusion $\mathcal{G}_{inn} \hookrightarrow \mathcal{G}$, where $\mathcal{G}_{inn}$ is the category of arenas and innocent strategies. In particular, the denotation within $\mathrm{Kl}_{R_X}\,\mathcal{G}$ of any

$$\frac{}{\Gamma, x\colon T \vdash x\colon T} \qquad \frac{\Gamma \vdash M\colon S \to T \qquad \Gamma \vdash N\colon S}{\Gamma \vdash MN\colon T} \qquad \frac{\Gamma, x\colon S \vdash M\colon T}{\Gamma \vdash \lambda x^S.M\colon S \to T}$$

$$\frac{}{\Gamma \vdash \mathbb{t}\colon \mathtt{bool}} \qquad \frac{}{\Gamma \vdash \mathbb{f}\colon \mathtt{bool}}$$

$$\frac{\Gamma \vdash M\colon \mathtt{bool} \qquad \Gamma \vdash N\colon T \qquad \Gamma \vdash P\colon T}{\Gamma \vdash \mathsf{If}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P\colon T}\ T \in \{\mathtt{bool}, \mathtt{nat}\}$$

$$\frac{}{\Gamma \vdash n\colon \mathtt{nat}} \qquad \frac{\Gamma \vdash M\colon \mathtt{nat}}{\Gamma \vdash \mathsf{succ}\, M\colon \mathtt{nat}} \qquad \frac{\Gamma \vdash M\colon \mathtt{nat}}{\Gamma \vdash \mathsf{pred}\, M\colon \mathtt{nat}}$$

$$\frac{\Gamma \vdash M\colon \mathtt{nat} \qquad \Gamma \vdash N\colon T \qquad \Gamma \vdash P\colon T}{\Gamma \vdash \mathsf{If0}\, M\ \mathsf{then}\ N\ \mathsf{else}\ P\colon T}\ T \in \{\mathtt{bool}, \mathtt{nat}\}$$

$$\frac{\Gamma \vdash M\colon S \qquad \Gamma, x\colon S \vdash N\colon T}{\Gamma \vdash \mathsf{let}\, x = M\ \mathsf{in}\ N\colon T}\ S, T \in \{\mathtt{bool}, \mathtt{nat}\} \qquad \frac{\Gamma \vdash M\colon T \to T}{\Gamma \vdash \mathbf{Y}_T M\colon T}$$

$$\frac{}{\mathsf{ask}_X\colon \mathtt{bool}}$$

Figure 9.1: Type theory for a variant of PCF with nondeterminism. Note that although ordinary deterministic PCF does not include the let construct, it has to be included in the nondeterministic (see [Lai15]) and probabilistic (see [EPT18]) versions.
In ordinary PCF, we may simulate the term $\mathsf{let}\, x = M\ \mathsf{in}\ N$ by the term $(\lambda x.N)M$, which is less efficient than the version with let (If N contains multiple occurrences of the variable x, then we end up compute M multiple times), but which still gives the same result.
In nondeterministic variants of PCF, however, evaluating M multiple times may give a different answer each time, in contrast to the version with let, in which we get a single answer and use it multiple times.

$$\frac{}{c \Downarrow c} \qquad \frac{M \Downarrow \lambda x.M' \qquad M'[N/x] \Downarrow c}{M\,N \Downarrow c}$$

$$\frac{M \Downarrow \mathtt{t} \qquad N \Downarrow c}{\mathsf{If}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c} \qquad \frac{M \Downarrow \mathtt{f} \qquad P \Downarrow c}{\mathsf{If}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c}$$

$$\frac{M \Downarrow n}{\mathsf{succ}\, M \Downarrow n+1} \qquad \frac{M \Downarrow n+1}{\mathsf{pred}\, M \Downarrow n} \qquad \frac{M \Downarrow 0}{\mathsf{pred}\, M \Downarrow 0}$$

$$\frac{M \Downarrow 0 \qquad N \Downarrow c}{\mathsf{If0}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c} \qquad \frac{M \Downarrow n+1 \qquad P \Downarrow c}{\mathsf{If0}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P \Downarrow c}$$

$$\frac{M(\mathbf{Y}M) \Downarrow c}{\mathbf{Y}M \Downarrow c} \qquad \frac{}{\mathsf{ask}_X \Downarrow \mathtt{t}} \qquad \frac{}{\mathsf{ask}_X \Downarrow \mathtt{f}}$$

Figure 9.2: Big-step operational semantics for Nondeterministic PCF and may testing

term of ordinary PCF lives within $\mathrm{Kl}_{R_X}\,\mathcal{G}_{inn}$. Moreover, the denotation of the new term ask_X, being given by the identity morphism in $\mathcal{G}$, also lives in the category

$$\mathcal{G}_{inn,X} \coloneqq \mathrm{Kl}_{R_X}\,\mathcal{G}_{inn}\,,$$

giving us a denotational semantics of Nondeterministic PCF within $\mathcal{G}_{inn,X}$.

It is also quick to check that our operational semantics in Figure 9.2 may be regarded as a subset of the rules for IA_X with May Testing, as we studied in §4.10. Specifically, if M is a term of PCF and c a canonical form, then $M \Downarrow c$ if and only if

$$\_\,, () \vdash M \Downarrow c, ()$$

in IA_X.

We want to use our Computational Adequacy result for IA_X with May Testing (Corollary 4.10.3) to get a Computational Adequacy result for Nondeterministic PCF. The only slight problem is that Corollary 4.10.3 is stated for terms of type `com`, whereas our language has no such type. We get around this by using `nat` as our main ground type rather than `com`, and by using the following easy lemma.

Lemma 9.1.1. *Let $M\colon$ `nat` be a term of IA_X. Then there exists n such that $\Gamma, s \vdash M \Downarrow n, s'$ if and only if*

$$\Gamma, s \vdash \mathsf{If0}\ M\ \mathsf{then\ skip\ else\ skip} \Downarrow \mathsf{skip}, s'\,.$$

We will write $\delta\colon \mathbb{N} \to \mathbb{C}$ for the denotation of the term-in-context

$$x\colon \mathtt{nat} \vdash \mathsf{If0}\, x\ \mathsf{then\ skip\ else\ skip}\colon \mathtt{com}\,.$$

Then we immediately get the following result as a special case of Corollary 4.10.3.

Corollary 9.1.2 (Computational Adequacy for Nondeterministic PCF)**.** *Let $M\colon \mathtt{nat}$ be a closed term of Nondeterministic PCF. Consider the denotation $[\![M]\!] : 1 \to \mathbb{N}$ in $\mathrm{Kl}_{R_X}\,\mathcal{G}_{inn}$ as a morphism $1 \to (X \to \mathbb{N})$ in $\mathcal{G}_{inn}$, and thence as a morphism $1 \to (X \to \mathbb{N})$ in $\mathcal{G}$.*

Then there exists some sequence $u \in X^*$ such that the composite

$$1 \xrightarrow{[\![M]\!]} (X \to \mathbb{N}) \xrightarrow{X \to \delta} (X \to \mathbb{C}) \xrightarrow{\eta_u} (\mathrm{Var} \to \mathbb{N}) \xrightarrow{[\![\mathsf{new}]\!]} \mathbb{N} \xrightarrow{t_{|u|}} \mathbb{C}$$

is not equal to $\bot$, if and only if $M \Downarrow n$ for some n.

Note that Corollary 9.1.2, which does not depend on any specific details of the models $\mathcal{G}_{inn}$ and $\mathcal{G}$ beyond the fact that they are computationally adequate for ordinary PCF and Idealized Algol, uses morphisms δ, η_u, $[\![\mathsf{new}]\!]$ and $\mathsf{t}_{|u|}$ that do not occur in the category $\mathcal{G}_{inn}$.

In the case that $\mathcal{G}$ is the category of arenas and single-threaded strategies, and $\mathcal{G}_{inn}$ the category of arenas and innocent strategies, we can restate Corollary 9.1.2 in a way that does not refer to Idealized Algol terms at all.

Corollary 9.1.3. *Let $M\colon \mathtt{nat}$ be a closed term of Nondeterministic PCF and consider the denotation $[\![M]\!] : 1 \to \mathbb{N}$ in $\mathrm{Kl}_{R_X}\,\mathcal{G}_{inn}$ as a morphism $1 \to (X \to \mathbb{N})$ in $\mathcal{G}_{inn}$.*

Then, for all $n \in \mathbb{N}$, there exists some sequence $s \in [\![M]\!]$ such that $s|_{\mathbb{N}} = qn$ if and only if $M \Downarrow n$.

Proof. Corollary 9.1.2, together with our earlier analysis, immediately tells us that we have

$$\exists s \in [\![M]\!]\ .\ s|_{\mathbb{N}} = qn \text{ for some } n \Leftrightarrow \exists n\ .\ M \Downarrow n\,.$$

The problem is that the two n's might be different. However, we can obtain the desired result for a given n from this one by composing with an appropriate term $\mathtt{nat} \to \mathtt{nat}$ that converges if and only if its input is equal to n. □

The passage to full abstraction, via innocent definability for PCF is essentially the same as for Idealized Algol (though, as before, if $X = \mathbb{N}$, then we need to cut down to a category of *recursive* strategies).

If we so desire, we can continue with the programme from §4.10, declaring two Kleisli strategies $A \to (X \to B)$ to be equivalent if they contain the same plays after restriction to $A \to B$. We can then do away with the plays in X altogether and give a model that involves only nondeterministic strategies.

At this point, we run into a well-known problem: under what circumstances should a nondeterministic strategy be described as *innocent*? The answer that we get from our model is that it is innocent if and only if it takes the form

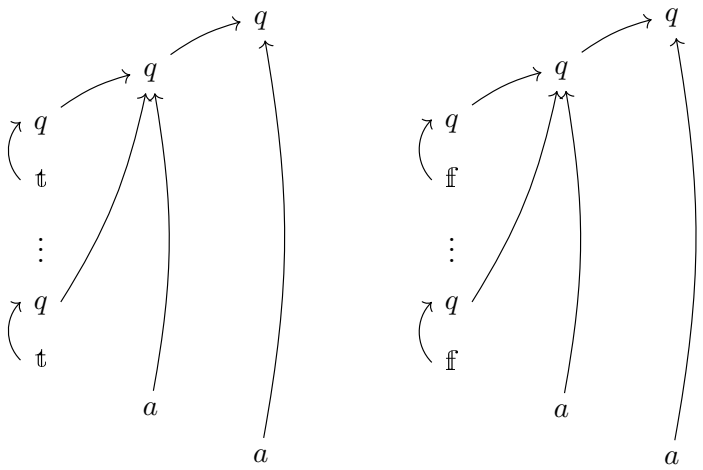
$$A \xrightarrow{\sigma} (X \to B) \xrightarrow{[\![\mathsf{ask}_X]\!] \to B} B\,,$$

where σ is an innocent deterministic strategy. This is the definition considered – and ultimately rejected for being too indirect – by Harmer in his thesis [Har99, §3.7]. This definition is correct, but, perhaps surprisingly, does not coincide with what we would get by naively applying the usual definition of innocence to nondeterministic strategies.

Consider, for example, the denotation of the term

$$\mathsf{If}\ \mathsf{ask}_{\mathbb{B}}\ \mathsf{then}\ (\lambda f.f\,\mathbb{t})\ \mathsf{else}\ (\lambda f.f\,\mathbb{f})\,,$$

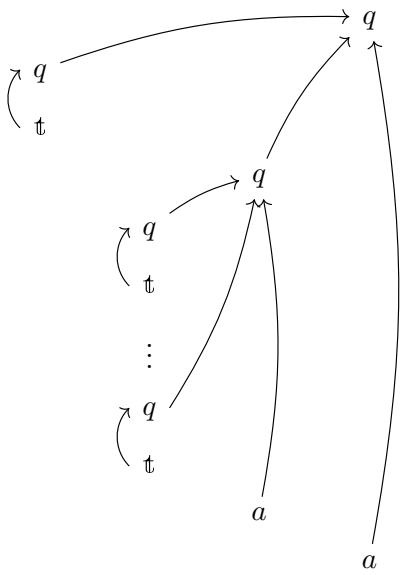
which has maximal plays taking one of the following two forms.



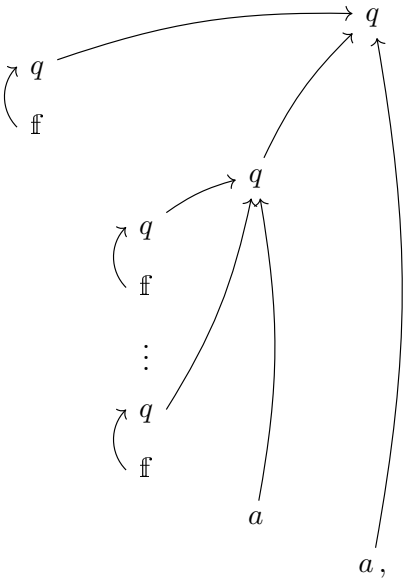
Note that this strategy displays typically non-innocent behaviour: if player P has played $\mathbb{t}$ on the left, then she must play $\mathbb{t}$ again whenever player O asks, even though the original move $\mathbb{t}$ occurs outside the current P-view.

In the Kleisli category model, the innocence of this strategy becomes clear:

plays are either of the form



or of the form



where now the move $\mathbb{t}$ or $\mathbb{f}$ in the nondeterministic oracle game (on the very left) locks the 'branching-time information' into the P-view.

Moving away from our Kleisli-category model, there is a surprising and elegant definition of nondeterministic innocence within the usual category of games (see Levy [Lev14], with the correction given by Tsukada and Ong in [TO15, Proposition 46]), which we shall not detail here. Tsukada and Ong have also demonstrated that it is instructive to consider strategies as

presheaves over plays, where we assign to each individual play a set of possible branches that lead us to that play. In this formalism, the innocent nondeterministic strategies are precisely those presheaves that are sheaves with respect to a certain natural Grothendieck topology.

Even given this theory, however, our Kleisli category model is still worthwhile, since it provides a general setting in which we can prove results such as adequacy without worrying about the specific details of a nondeterministic construction. For example, we can prove a computational adequacy result for Tsukada and Ong's model $\mathcal{G}_{TO}$ from Corollary 9.1.3 by noting that the denotational semantics in that category factors through the natural functor $\mathrm{Kl}_{R_{\mathbb{B}}}\mathcal{G} \to \mathcal{G}_{TO}$. Tsukada and Ong have provided their own proof of Computational Adequacy in the finite nondeterminism case, but this method could be useful if, for example, we wanted to extend their model to deal with countable nondeterminism.

9.2 Stateful Effects

In Chapters 2 and 3, we presented a categorical algebra of scoped state via sequoidal categories. The category theory that we used in those chapters is fairly non-standard, and a natural question to ask is whether we can come up with a treatment of state from the point of view of monads or lax actions.

Indeed, there is a well known *state monad*

$$A \mapsto (W \to (A \times W))\,,$$

which can be constructed in categories of games. More ambitiously, we might hope to capture a parametric notion of local state, via the action

$$(R, W).A = R \to (A \times W)\,.$$

of $\mathcal{G}^{\mathrm{op}} \times \mathcal{G}$ on $\mathcal{G}$.

In this section, we will present some of the problems involved in this case.

One issue, which we have touched on already, is that the state monad is not a monoidal functor, which was the condition we needed in order to ensure that its Kleisli category inherited a monoidal structure. Indeed, there is no obvious natural morphism

$$(W \to (A \times W)) \times (W \to (B \times W)) \to (W \to ((A \times B) \times W))\,.$$

The parametric version suffers from a different problem. Suppose we have two Melliès morphisms

$$f\colon A \to (R \to (B \times W)) \qquad\qquad g\colon B \to (R' \to (C \times W'))\,.$$

Then their composition will be given by a Melliès morphism

$$f; g \colon A \to ((R \times R') \to (C \times (W \times W'))) \ :$$

in other words, we will have combined the two inputs R and R' and the two outputs W and W'.

This works well up to a point, but the main point of state is that we want to be able to write to a variable once and then read from it later. So we'd need some way of wiring up the output on the right to the input on the left, and it's not clear how to deal with this in a category-theoretic way.

Linear type theory provides a perspective on this problem. Note that the state monad and the parametric action above can both be defined inside an arbitrary monoidal category. However, if we want our language to exhibit any typically stateful behaviour, then we need to be able to refer to the same variable more than once: there is little point writing a value into a cell if we are not able to read from it later. This is not a problem in a monolithic stateful system – such as we have with the state monad – but if we want to refer to individual storage cells using variables in the language, then we need our category to admit diagonals, to allow us to refer to a storage cell twice. Since the state action above does not interact in any way with the Cartesian structure of the category (i.e., the diagonals and terminal morphisms), it cannot hope to capture local state in this way.

One avenue that could shed some light on this issue is the sequoidal semantics which we developed for game semantics, in which the stateful behaviour is intimately connected to the Cartesian structure of the category, via the exponential.

There is also a *local state monad*, due to Plotkin and Power [PP02] and based on a construction by Levy [Lev01], which is defined on the category $[\mathbf{Inj}, \mathbf{Set}]$ of functors from $\mathbf{Inj}$, the category of natural numbers and injections, into the category of sets. It is given by the following formula, where V denotes a fixed set of values.

$$(TA)(n) = V^n \to \left(\int^{p \colon \mathbf{Inj}} V^p \times A(p) \times \mathbf{Inj}(n, p) \right)$$

Here, if $A \colon \mathbf{Inj} \to \mathbf{Set}$ is a functor, then we think of $A(n)$ as the set of all values that A takes on in the presence of n local variables taking values in V.

We cannot construct this coend in the category of games, but there is a clear link with our constructions with probabilistic monads. Note that the 'state action'

$$(W, A) \mapsto W \to (A \times W)$$

has mixed variance in W, suggesting a modification of our $\mathcal{C}/\mathcal{X}$ construction that uses a coend rather than an ordinary colimit.

9.3 Relational Models

The archetypal Cartesian closed category is the category **Set** of sets and functions. Most of the work in this thesis has been to do with reader actions (including reader monads as a special case), and we have already provided a classification of the reader actions on **Set** in Propositions 6.1.1 and 6.3.1: namely, they are classified by lax monoidal functors $F\colon \mathbf{Set} \to \mathbf{Set}$, in such a way that if a reader action given by an oplax monoidal functor $j\colon \mathcal{X} \to \mathbf{Set}$ corresponds to the functor $F\colon \mathbf{Set} \to \mathbf{Set}$, then $\mathbf{Set}/\mathcal{X}^{\mathrm{op}}$ is isomorphic to the category $F_*\mathbf{Set}$ formed from **Set** by base change along F. Thus, the theory in the case of **Set** reduces to the theory of **Set**-enriched base change.

There are, however, many more examples in the literature of models of programming languages that ultimately derive from Kleisli categories for monads not of the reader kind. The problem with this, from our point of view, is that the resulting Kleisli categories will not be automatically Cartesian. This means that we need different methods to ensure that we end up with a category suitable for modelling a compositional semantics.

For example, the Kleisli category for the powerset monad on **Set** is the category **Rel** of sets and relations. This is a monoidal closed category, so can be used to model multiplicative linear type theory. By using a linear exponential comonad based on finite multisets, we can transform it into a Cartesian closed category.

An important category derived from the category of sets and relations is the category **Coh** of coherence spaces. A coherence space is a set with some additional structure on it – namely, the structure of an undirected graph – and a morphism between coherence spaces is a relation between sets that respects this structure in a particular way (see [Mel14] for details).

Danos and Ehrhard in [DE11] develop a probabilistic version of coherence spaces, which is still monoidal closed. By using the finite multiset comonad, we can transform it into a Cartesian closed category. A striking result of Ehrhard, Pagani and Tasson [EPT18] is that this category is already fully abstract for a probabilistic variant of PCF.

This suggests an alternative way to get round the Cartesianness hurdle when adding effects: start with a model of the base language that is constructed by applying a linear exponential comonad to a monoidal closed category, and then delay application of the linear exponential comonad until after we have added the effect.

Unfortunately, there is no real guarantee that a linear exponential comonad on the base category will give us a linear exponential comonad on a Kleisli category or one of the other effectful categories we have considered. Some work is required to investigate the conditions in which we can lift an exponential construction in this way, given the functorial results from Chapter 5. This will pave the way for us to understand the relational and coherence-space models using the ideas developed in this thesis.

9.4 Adequacy Proofs

In Chapters 4 and 8 we proved Computational Adequacy via a factorization and operational techniques. This is good, because it allows us to assume very little about the underlying model, other than that it is itself computationally adequate. Nevertheless, our techniques rely very heavily on the fact that our base language is Idealized Algol. Although we outlined ways to get around this in Section 9.1, it would also be useful to have a conventional logical relations-based proof of Computational Adequacy.

Assume that $\mathcal{G}$ is a Cartesian closed category enriched in directed-complete partial orders (dcpos). If $M\colon \mathcal{G} \to \mathcal{G}$ is a monoidal monad, then the Kleisli category associated to M inherits an order, whereby $\sigma \leq \tau\colon A \to B$ in $\mathrm{Kl}_M\,\mathcal{G}$ if $\sigma \leq \tau\colon A \to MB$ in $\mathcal{G}$. Moreover, since composition in the Kleisli category is given by a composition in the base category, this partial order will be respected by composition, as will be limits of directed sets.

Suppose again that $\mathcal{G}$ is a dcpo-enriched category, and that a monoidal category $\mathcal{X}$ acts on $\mathcal{G}$ via a lax action. Then the construction of Melliès yields an $[\mathcal{X}, \mathbf{Dcpo}]$-enriched category $\mathrm{Mell}_{\mathcal{X}}\,\mathcal{G}$. Since **Dcpo** is cocomplete [Mes77], we can construct the coends we need to define the Day convolution product in $[\mathcal{X}, \mathbf{Dcpo}]$.

If $\mathcal{X}$ is small, or if the required colimits exist for other reasons, then we may take the change of base with respect to the colimit functor $[\mathcal{X}, \mathbf{Dcpo}] \to \mathbf{Dcpo}$ to $\mathrm{Mell}_{\mathcal{X}}\,\mathcal{G}$ to give us a **Dcpo**-enriched category $\mathcal{G}/\mathcal{X}$.

Concretely, the underling set of morphisms in $\mathcal{G}/\mathcal{X}$ contains all the usual morphisms from the **Set**-enriched case, while the order structure is given by a transfinite construction as in [Fie96, 2.17]. In particular, if $\sigma, \tau\colon A \to B$ in $\mathcal{G}/\mathcal{X}$ are given by morphisms

$$\tilde{\sigma}\colon A \to X.B \qquad\qquad \tilde{\tau}\colon A \to Y.B$$

in $\mathcal{G}$, and if there are morphisms $h\colon X \to Z$, $k\colon Y \to Z$ in $\mathcal{X}$ such that

$$\sigma; (h.B) \leq \tau; (k.B)\,,$$

then $\sigma \leq \tau$ in $\mathcal{G}/\mathcal{X}$.

A slight problem is that it is not true in general that the colimit of cpos with bottom elements has itself a bottom element. For example, let $-[n]$ be the partial order $\{-n < \cdots < 0\}$. Then, $-[n]$ is a dcpo with a bottom element for each n, but the colimit of the chain

$$-[0] \hookrightarrow -[1] \hookrightarrow -[2] \hookrightarrow \cdots$$

is the partial order of non-positive integers.

However, if all the morphisms in a diagram of dcpos are such that they preserve bottom elements, then the colimit of that diagram will have a bottom element.

In particular, if the action of $\mathcal{X}$ on $\mathcal{G}$ is a reader-style action that factors through the category of sets (as in all our examples), then every arrow in the colimit diagram that defines $\mathcal{G}/\mathcal{X}(A, B)$ will be of the form

$$\mathcal{G}(A, \underline{Y} \to B) \xrightarrow{\mathcal{G}(A, \underline{f}.B)} \mathcal{G}(A, \underline{X} \to B)$$

for some sets X, Y, considered as objects $\underline{X}, \underline{Y}$ of $\mathcal{G}$, and some function $f\colon X \to B$ considered as a morphism $\underline{f}\colon \underline{A} \to \underline{B}$ such that composition with $\underline{f}$ preserves bottom elements. Therefore, the map $\mathcal{G}(A, \underline{f}.B)$ of dcpos will also preserve bottom elements.

A second problem is that the setwise colimit of the $\mathcal{G}(A, X.B)$, with the order from [Fie96], is not necessarily itself a dcpo. In fact, to get the correct colimit of the diagram, we need to perform an extra step of *completion*. However, with a little more care, we can show that if the action is a reader action that factors through the category of sets, then the colimit is already directed complete under the order, at least in the category of games.

Having given the definition of the order-enriched structure of $\mathcal{G}$, we may define the interpretation of the fixpoint combinator $\mathbf{Y}$ using the properties of this new order, which will be enough to prove adequacy along the lines of Theorem 3.4.9.

It is worth mentioning that although we can prove computational adequacy for our various Kleisli categories (and categories of the form $\mathcal{G}/\mathcal{X}$), it does not automatically follow that we can prove adequacy in the same way for the various quotiented categories. An important example is our model of countable nondeterminism with must testing. It is well known (see, e.g., [AP86, AP81]) that countable nondeterminism is inextricably linked to discontinuity of composition, which robs us of an important ingredient in the standard order-theoretic proof of adequacy.

The solution is to prove adequacy in the normal way for the semantics of $\mathrm{IA}_{\mathbb{N}}$ in $\mathcal{G}_{\mathbb{N}}$, and then to apply the argument from Corollary 4.11.8 to reduce

this to a Computational Adequacy result for the semantics of Idealized Algol with countable nondeterminism and must testing. Here, we are using an idea due to Levy [Lev08]: that if we want to prove adequacy for countable nondeterminism, then we need a form of *explicit forcing*, in which we uncover some information about the points at which we have made an infinite nondeterministic branch. In our case, the explicit forcing is done through the language $\mathrm{IA}_{\mathbb{N}}$, which creates a complete record of the nondeterministic values we have chosen along the way. We have then applied the second part of Levy's technique – *hiding* – to remove the explicit information and yield a pure, computationally adequate model of countable nondeterminism.

9.5 Afterword

The aim of this thesis was to demonstrate how techniques of categorical algebra can be applied to the study of Full Abstraction for programming languages. I can think of two main benefits of this combination. The first is a technical one: the results that we have proved are very generally applicable, and it would not be unreasonable to hope that they could be applied in the future to help construct new Fully Abstract models of programming languages. The second reason is less precise, but equally important: throughout this thesis, we have endeavoured to clear up some of the mystery surrounding Full Abstraction, particularly in Game Semantics. The idea of using monads and Kleisli categories to model computational effects has a long history, but work in Full Abstraction, though it sometimes uses category-theoretic techniques, has often seemed fairly *ad hoc*. My hope is that the work presented here – including both the sequoidal-exponential semantics for Idealized Algol and the work on Kleisli categories and parametric monads – has gone some way towards explaining where some of the constructions used in Game Semantics come from.

One of the themes in this thesis has been an attempt to push results as far as they can go: thus, we have tried as far as possible to use, for example, Computational Adequacy results for a base language in order to prove Computational Adequacy for an extended language, rather than writing on a stand-alone proof. To an extent, we have been forced to do this by the level of generality we have been working at, but I hope that our focus on syntactic rather than semantic results has helped shift the focus away from the properties of any particular model.

Bibliography

[Adá03] Jiří Adámek. On final coalgebras of continuous functors. *Theoretical Computer Science*, 294(1):3 – 29, 2003. Category Theory and Computer Science.

[AHM98] S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science*, LICS '98, pages 334–, Washington, DC, USA, 1998. IEEE Computer Society.

[AJM00] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409 – 470, 2000.

[AM96] Samson Abramsky and Guy McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions: Extended abstract. *Electronic Notes in Theoretical Computer Science*, 3:2 – 14, 1996. Linear Logic 96 Tokyo Meeting.

[AM99] Samson Abramsky and Guy McCusker. Full abstraction for Idealized Algol with passive expressions. *Theor. Comput. Sci.*, 227(1-2):3–42, September 1999.

[AP81] K. R. Apt and G. D. Plotkin. A cook's tour of countable nondeterminism. In Shimon Even and Oded Kariv, editors, *Automata, Languages and Programming*, pages 479–494, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg.

[AP86] K. R. Apt and G. D. Plotkin. Countable nondeterminism and random assignment. *Journal of the ACM*, 33(4):724–767, August 1986.

[Bau06] Andrej Bauer. König's lemma and the Kleene tree. Published via blog post at `http://math.andrej.com/2006/04/25/konigs-lemma-and-the-kleene-tree/`, April 2006.

[Bla92] Andreas Blass. A game semantics for linear logic. *Annals of Pure and Applied Logic*, 56(1–3):183 – 220, 1992.

[CCHW18] Simon Castellan, Pierre Clairambault, Jonathan Hayman, and Glynn Winskel. Non-angelic concurrent game semantics. In Christel Baier and Ugo Dal Lago, editors, *Foundations of Software Science and Computation Structures*, pages 3–19, Cham, 2018. Springer International Publishing.

[CCPW18] Simon Castellan, Pierre Clairambault, Hugo Paquet, and Glynn Winskel. The concurrent game semantics of probabilistic PCF. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '18, pages 215–224, New York, NY, USA, 2018. ACM.

[CLM11] Martin Churchill, James Laird, and Guy McCusker. Imperative programs as proofs via game semantics. In *Proceedings of the 2011 IEEE 26th Annual Symposium on Logic in Computer Science*, LICS '11, page 65–74, USA, 2011. IEEE Computer Society.

[Day70] Brian Day. On closed categories of functors. In S. MacLane, H. Applegate, M. Barr, B. Day, E. Dubuc, Phreilambud, A. Pultr, R. Street, M. Tierney, and S. Swierczkowski, editors, *Reports of the Midwest Category Seminar IV*, pages 1–38, Berlin, Heidelberg, 1970. Springer Berlin Heidelberg.

[DE11] Vincent Danos and Thomas Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Information and Computation*, 209(6):966 – 991, 2011.

[DH00] Vincent Danos and Russell S. Harmer. Probabilistic game semantics. In *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No.99CB36332)*, LICS '00, pages 204–213. IEEE Computer Society, June 2000.

[Dij97] Edsger Wybe Dijkstra. *A Discipline of Programming*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1997.

[EK66] Samuel Eilenberg and G.M Kelly. A generalization of the functorial calculus. *Journal of Algebra*, 3(3):366 – 375, 1966.

[EPT18] Thomas Ehrhard, Michele Pagani, and Christine Tasson. Full abstraction for probabilistic PCF. *Journal of the ACM*, 65(4):23:1–23:44, April 2018.

[FH92] Matthias Felleisen and Robert Hieb. The revised report on the syntactic theories of sequential control and state. *Theoretical Computer Science*, 103(2):235 – 271, 1992.

[Fie96] Adrian Fiech. Colimits in the category dcpo. *Mathematical Structures in Computer Science*, 6:455–468, 1996.

[FKM16] Soichiro Fujii, Shin-ya Katsumata, and Paul-André Melliès. Towards a formal theory of graded monads. In Bart Jacobs and Christof Löding, editors, *Foundations of Software Science and Computation Structures*, pages 513–530, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[Ghi05] Dan R. Ghica. Slot games: A quantitative model of computation. In *Proceedings of the 32Nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '05, pages 85–97, New York, NY, USA, 2005. ACM.

[GHN17] David Gepner, Rune Haugseng, and Thomas Nikolaus. Lax colimits and free fibrations in ∞-categories. *Documenta Mathematica*, 22:1255–1266, 2017.

[GL17] William John Gowers and James Laird. Sequoidal categories and transfinite games: A coalgebraic approach to stateful objects in game semantics. In Filippo Bonchi and Barbara König, editors, *7th Conference on Algebra and Coalgebra in Computer Science, CALCO 2017, June 12-16, 2017, Ljubljana, Slovenia*, volume 72 of *LIPIcs*, pages 13:1–13:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[GL18] W. John Gowers and James D. Laird. A Fully Abstract Game Semantics for Countable Nondeterminism. In Dan Ghica and Achim Jung, editors, *27th EACSL Annual Conference on Computer Science Logic (CSL 2018)*, volume 119 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:18, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[SGA1] Alexander Grothendieck. *Revêtements étales et groupe fondamental (SGA 1)*, volume 224 of *Lecture notes in mathematics*. Springer-Verlag, 1971.

[Har99] Russell S. Harmer. Games and full abstraction for nondeterministic languages. Technical report, Imperial College London (University of London), 1999.

[Har06] Russell S. Harmer. Innocent game semantics. 2006.

[Her00] Claudio Hermida. Representable multicategories. *Advances in tics*, 151(2):164 – 225, 2000.

[HM99] Russel S. Harmer and G. McCusker. A fully abstract game semantics for finite nondeterminism. In *Proceedings. 14th An-*

nual IEEE Symposium on Logic in Computer Science (Cat. No. PR00158), LICS '99, pages 422–430. IEEE Computer Society, 1999.

[HO00] J.M.E. Hyland and C.-H.L. Ong. On full abstraction for PCF: I, II, and III. *Information and Computation*, 163(2):285 – 408, 2000.

[JK02] G. Janelidze and G. M. Kelly. A note on actions of a monoidal category. *Theory and Applications of Categories*, 9, 01 2002.

[Joh02] Peter T Johnstone. *Sketches of an elephant: a Topos theory compendium.* Oxford logic guides. Oxford Univ. Press, New York, NY, 2002.

[Jon95] Mark P. Jones. Functional programming with overloading and higher-order polymorphism. In *Advanced Functional Programming, First International Spring School on Advanced Functional Programming Techniques-Tutorial Text*, pages 97–136, Berlin, Heidelberg, 1995. Springer-Verlag.

[Joy77] André Joyal. Remarques sur la théorie des jeux à deux personnes. *Gazette des sciences mathematiques de Quebec*, 1(4), 1977.

[KK81] G.M. Kelly and V. Koubek. The large limits that all good categories admit. *Journal of Pure and Applied Algebra*, 22(3):253 – 263, 1981.

[Kle65] H. Kleisli. Every standard construction is induced by a pair of adjoint functors. *Proceedings of the American Mathematical Society*, 16(3):544–546, 1965.

[Lai01] James Laird. A fully abstract game semantics of local exceptions. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science*, LICS '01, pages 105–114. IEEE Computer Society, 2001.

[Lai02] James Laird. A categorical semantics of higher-order store. In *Proceedings of CTCS '02*, number 69 in ENTCS. Elsevier, 2002.

[Lai15] James Laird. Sequential algorithms for unbounded nondeterminism. *Electronic Notes in Theoretical Computer Science*, 319:271 – 287, 2015.

[Lai16] James Laird. Higher-order programs as coroutines. to appear, 2016.

[Lam68] Joachim Lambek. A fixpoint theorem for complete categories. *Mathematische Zeitschrift*, 103(2):151–161, 1968.

[Lam74] Joachim Lambek. Functional completeness of cartesian categories. *Annals of Mathematical Logic*, 6(3):259 – 292, 1974.

[Lei98] Tom Leinster. Basic bicategories. In *E-print math.CT/9810017*, 1998.

[Lei04] Tom Leinster. *Introduction*, page 1–6. London Mathematical Society Lecture Note Series. Cambridge University Press, 2004.

[Lev01] Paul Blain Levy. Call-by-push-value. Technical report, Queen Mary College, 2001.

[Lev08] Paul Blain Levy. Infinite trace equivalence. *Annals of Pure and Applied Logic*, 151(2):170 – 198, 2008.

[Lev14] Paul Blain Levy. Morphisms between plays. Lecture slides, GaLoP `https://www.cs.bham.ac.uk/~pbl/papers/morphismplaytalk.pdf`, 2014.

[LN15] John Longley and Dag Normann. *Higher-Order Computability*. Springer Berlin Heidelberg, 2015.

[Mac71] Saunders MacLane. *Categories for the Working Mathematician*. Springer-Verlag, New York, 1971. Graduate Texts in Mathematics, Vol. 5.

[Man12] Oleksandr Manzyuk. Closed categories vs. closed multicategories, 2012.

[McC02] Guy McCusker. A fully abstract relational model of syntactic control of interference. In Julian Bradfield, editor, *Computer Science Logic*, pages 247–261, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[Mel12] Paul-André Melliès. Parametric monads and enriched adjunctions. `https://www.irif.fr/~mellies/tensorial-logic/8-parametric-monads-and-enriched-adjunctions.pdf`, 2012.

[Mel14] Paul-André Melliès. A survival guide on coherence spaces. `https://www.irif.fr/~mellies/mpri/mpri-m2/mpri-mellies-lecture-notes-2.pdf`, 2014.

[Mel17] Paul-André Melliès. The parametric continuation monad. *Mathematical Structures in Computer Science*, 27(5):651–680, 2017.

[Mes77] J. Meseguer. On order-complete universal algebra and enriched functorial semantics. In Marek Karpiński, editor, *Fundamentals of Computation Theory*, pages 294–301, Berlin, Heidelberg, 1977. Springer Berlin Heidelberg.

[Mog91] Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55 – 92, 1991. Selections from Fourth Annual IEEE Symposium on Logic in Computer Science.

[MT16] Andrzej S. Murawski and Nikos Tzevelekos. Nominal game semantics. *Found. Trends Program. Lang.*, 2(4):191–269, March 2016.

[MTT09] Paul-André Melliès, Nicolas Tabareau, and Christine Tasson. *An Explicit Formula for the Free Exponential Modality of Linear Logic*, pages 247–260. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[Nic94] Hanno Nickau. Hereditarily sequential functionals. In *International Symposium on Logical Foundations of Computer Science*, pages 253–264. Springer, 1994.

[nLa19] nLab authors. 2-limit. `http://ncatlab.org/nlab/show/2-limit`, February 2019. Revision 52.

[OR95] P.W. O'Hearn and J.G. Riecke. Kripke logical relations and PCF. *Information and Computation*, 120(1):107 – 116, 1995.

[Pis14] Claudio Pisani. Sequential multicategories. *Theory and Applications of Categories*, 29(19):496 – 541, 2014.

[Plo77] G.D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5(3):223 – 255, 1977.

[PP02] Gordon Plotkin and John Power. Notions of computation determine monads. In Mogens Nielsen and Uffe Engberg, editors, *Foundations of Software Science and Computation Structures*, pages 342–356, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[Sch04] Andrea Schalk. What is a categorical model for linear logic? `http://www.cs.man.ac.uk/~schalk/notes/llmodel.pdf`, October 2004.

[Sco76] D. Scott. Data types as lattices. *SIAM Journal on Computing*, 5(3):522–587, 1976.

[SG12] Mike Shulman and Richard Garner. Categories, functors, and profunctors are a free cocompletion. Octoberfest 2012, October 2012.

[Shr04] Steven E Shreve. *Stochastic calculus for finance 2, Continuous-time models*. Springer, New York, NY; Heidelberg, 2004.

[Sti18] John Stillwell. *Reverse Mathematics: Proofs from the Inside Out*. Princeton University Press, USA, 2018.

[Str72a] Ross Street. The formal theory of monads. *Journal of Pure and Applied Algebra*, 2(2):149 – 168, 1972.

[Str72b] Ross Street. Two constructions on lax functors. *Cahiers de Topologie et Géométrie Différentielle Catégoriques*, 13(3):217–264, 1972.

[Str80] Ross Street. Fibrations in bicategories. *Cahiers de Topologie et Géométrie Différentielle Catégoriques*, 21(2):111–160, 1980.

[TO14] Takeshi Tsukada and C.-H. Luke Ong. Innocent strategies are sheaves over plays - deterministic, non-deterministic and probabilistic innocence. *CoRR*, abs/1409.2764, 2014.

[TO15] Takeshi Tsukada and C. H. Luke Ong. Nondeterminism in game semantics via sheaves. In *Proceedings of the 2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, LICS '15, pages 220–231, Washington, DC, USA, 2015. IEEE Computer Society.