УНИВЕРЗИТЕТ У НОВОМ САДУ

ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

Вероватносно закључивање у израчунавању и теорији функционалних типова

ДОКТОРСКА ДИСЕРТАЦИЈА

Probabilistic reasoning in computation and simple type theory

DOCTORAL DISSERTATION

Ментори:
Проф. др Силвиа Гилезан
Др Зоран Огњановић

Кандидат:
Симона Прокић

Нови Сад, 2023. године

УНИВЕРЗИТЕТ У НОВОМ САДУ **ОБРАЗАЦ – 5а**
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА[1]

Врста рада:	Докторска дисертација
Име и презиме аутора:	Симона Прокић
Ментор (титула, име, презиме, звање, институција)	Проф. др Силвиа Гилезан, редовни професор, Факултет техничких наука, Универзитет у Новом Саду др Зоран Огњановић, научни саветник, Математички институт српске академије наука и уметности, Београд
Наслов рада:	Вероватносно закључивање у израчунавању и теорији функционалних типова
Језик публикације (писмо):	Енглески (латиница)
Физички опис рада:	Унети број: Страница 179 Поглавља 6 Референци 169 Табела 0 Слика 2 Графикона 0 Прилога 0
Научна област:	Примењена математика
Ужа научна област (научна дисциплина):	Теоријска и примењена математика (Логика у рачунарству)
Кључне речи / предметна одредница:	Вероватносно програмирање, ламбда рачун, комбинаторна логика, теорија типова, Крипкеова семантика, потпуност, вероватносна логика
Резиме на језику рада:	Теза истражује два различита приступа за вероватносно закључивање у моделима израчунавања. Најчешћи приступ се састоји у проширењу ламбда рачуна вероватносним оператором избора што резултира вероватносним израчунавањем. То се показало веома корисним и примењивим у разним областима, на пример у роботици, обради природног језика и машинском учењу. Други приступ јесте да проширимо језик рачуна вероватносним операторима и добијемо модел за вероватносно закључивање о типизираном рачуну у стилу вероватносне логике. Најпре проучавамо вероватносни ламбда рачун проширен лет-ин

[1] Аутор докторске дисертације потписао је и приложио следеће Обрасце:
5б – Изјава о ауторству;
5в – Изјава о истоветности штампане и електронске верзије и о личним подацима;
5г – Изјава о коришћењу.
Ове Изјаве се чувају на факултету у штампаном и електронском облику и не кориче се са тезом.

	оператором где је примењена лења позив-по-имену стратегија евалуације, и изучавамо проблем еквиваленције програма у овом окружењу. Како је проблем доказивања контекстне еквиваленције доста изазован, истраживали смо ефикасне методе за доказивање еквиваленције програма. Вероватносна апликативна бисимулација се показала као одговарајући алат за доказивање еквиваленције програма у вероватносном окружењу. Доказујемо да је вероватносна апликативна бисимулација потпуно апстрактна у односу на контекстну еквиваленцију у вероватносном ламбда рачуну са лет-ин оператором. Затим уводимо Крипкеову семантику за целу комбинаторну логику са функционалним типовима, односно комбинаторну логику са функционалним типовима проширену типовима производа, типовима суме, празним типом и јединичним типом. Крипкеову семантику дефинишемо као Крипкеову апликативну структуру, која је екстензионална и има елементе који одговарају основним комбинаторима, и којој је придружена валуација променљивих. Доказујемо да је цела комбинаторна логика са функционалним типовима сагласна и потпуна у односу на уведене семантике. Уводимо логику комбинаторне логике, то јест исказно проширење комбинаторне логике са функционалним типовима. Доказујемо да је аксиоматизација логике комбинаторне логике сагласна и потпуна у односу на предложену семантику. Даље, показујемо да је уведена семантика нова семантика за комбинаторну логику са функционалним типовима проширену правилом типизирања које осигурава да једнаки терми имају исти тип. На крају, уводимо вероватносно проширење логике комбинаторне логике. Логику комбинаторне логике смо проширили са вероватносним операторима и добили модел за вероватносно закључивање о типизираним комбинаторним термима. Показујемо да је аксиоматизација логике сагласна и јако потпуна у односу на предложену семантику.
Датум прихватања теме од стране надлежног већа:	25. 2. 2021.
Датум одбране: (Попуњава одговарајућа служба)	
Чланови комисије: (титула, име, презиме, звање, институција)	Председник: др Јелена Иветић, ванредни професор, Факултет техничких наука, Универзитет у Новом Саду Члан: др Микеле Пагани, редовни професор, Laboratoire de l'Informatique du Parallélisme, École Normale Supérieure de Lyon, Француска

	Члан: др Зоран Петрић, научни саветник, Математички институт српске академије наука и уметности, Београд Члан: др Јована Обрадовић, научни сарадник, Математички институт српске академије наука и уметности, Београд Члан, ментор: др Силвиа Гилезан, редовни професор, Факултет техничких наука, Универзитет у Новом Саду Члан, ментор: др Зоран Огњановић, научни саветник, Математички институт српске академије наука и уметности, Београд
Напомена:	

UNIVERSITY OF NOVI SAD
FACULTY OF TECHNICAL SCIENCES

KEY WORD DOCUMENTATION[2]

Document type:	Doctoral dissertation
Author:	Simona Prokić
Supervisor (title, first name, last name, position, institution)	Prof. dr. Silvia Ghilezan, Full Professor, Faculty of Technical Sciences, University of Novi Sad dr. Zoran Ognjanović, Research Professor, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade
Thesis title:	Probabilistic reasoning in computation and simple type theory
Language of text (script):	English language
Physical description:	Number of: Pages 179 Chapters 6 References 169 Tables 0 Illustrations 2 Graphs 0 Appendices 0
Scientific field:	Applied Mathematics
Scientific subfield (scientific discipline):	Theoretical and Applied Mathematics (Logic in Computer Science)
Subject, Key words:	Probabilistic programming, lambda calculus, combinatory logic, type theory, Kripke-style semantics, completeness, probability logic
Abstract in English language:	This thesis investigates two different approaches for probabilistic reasoning in models of computation. The most usual approach is to extend the language of untyped lambda calculus with probabilistic choice operator which results in probabilistic computation. This approach has shown to be very useful and applicable in various fields, e.g. robotics, natural language processing, and machine learning. Another approach is to extend the language of a typed lambda calculus with probability operators and to obtain a framework for probabilistic reasoning about the typed calculus in the style of probability logic. First, we study the lazy call-by-name probabilistic lambda calculus extended with let-in operator, and program equivalence in the calculus. Since the proof of context equivalence is quite challenging, we

[2] The author of doctoral dissertation has signed the following Statements:
5б – Statement on the authority,
5в – Statement that the printed and e-version of doctoral dissertation are identical and about personal data,
5г – Statement on copyright licenses.
The paper and e-versions of Statements are held at he faculty and are not included into the printed thesis.

	investigate some effective methods for proving the program equivalence. Probabilistic applicative bisimilarity has proved to be a suitable tool for proving the context equivalence in probabilistic setting. We prove that the probabilistic applicative bisimilarity is fully abstract with respect to the context equivalence in the probabilistic lambda calculus with let-in operator. Next, we introduce Kripke-style semantics for the full simply typed combinatory logic, that is, the simply typed combinatory logic extended with product types, sum types, empty type and unit type. The Kripke-style semantics is defined as a Kripke applicative structure, which is extensional and has special elements corresponding to basic combinators, provided with the valuation of term variables. We prove that the full simply typed combinatory logic is sound and complete with respect to the proposed semantics. We introduce the logic of combinatory logic, that is, a propositional extension of the simply typed combinatory logic. We prove that the axiomatization of the logic of combinatory logic is sound and strongly complete with respect to the proposed semantics. In addition, we prove that the proposed semantics is the new semantics for the simply typed combinatory logic containing the typing rule that ensures that equal terms inhabit the same type. Finally, we introduce the probabilistic extension of the logic of combinatory logic. We extend the logic of combinatory logic with probability operators and obtain a framework for probabilistic reasoning about typed combinatory terms. We prove that the given axiomatization of the logic is sound and strongly complete with respect to the proposed semantics.
Accepted on Scientific Board on:	25. 2. 2021.
Defended: (Filled by the faculty service)	
Thesis Defend Board: (title, first name, last name, position, institution)	President: dr. Jelena Ivetić, Associate Professor, Faculty of Technical Sciences, University of Novi Sad Member: dr. Michele Pagani, Full Professor, Laboratoire de l'Informatique du Parallélisme, École Normale Supérieure de Lyon, France Member: dr. Zoran Petrić, Research Professor, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade Member: dr. Jovana Obradović, Research Assistant Professor, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade

	Member, Mentor: dr. Silvia Ghilezan, Full Professor, Faculty of Technical Sciences, University of Novi Sad Member, Mentor: dr. Zoran Ognjanović, Research Professor, Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade
Note:	

Acknowledgements

I would like to express my deepest appreciation to my mentors Professor Silvia Ghilezan and Professor Zoran Ognjanović for their invaluable advice, continuous support, and patience during my PhD study. Their immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life. Words cannot express my gratitude to Professor Silvia Ghilezan for making the past years much more enjoyable and showing me that research is much more than writing papers. I am extremely grateful to Professor Zoran Ognjanović who was always there for discussions and gave outstanding feedback.

I could not have undertaken this journey without my family. I am deeply grateful to my parents Đorđa and Jovanka for their unwavering support and boundless love. Your encouragement, sacrifices and faith in me have been a constant source of motivation. To my sisters Đurđina and Saška, thank you for always being there for me. You have been my rocks, my confidantes, and my biggest supporters. I would not be the same without you, and I am lucky to have grown up with you.

I want to acknowledge my friends for the support that they have given me over the years. You have been a great source of comfort, love and inspiration.

I also express my gratitude to all the colleagues at the Chair of Mathematics at Faculty of Technical Sciences for their help and support. Thank you for being such a wonderful group of people, both in and out of the office.

I would also like to thank the members of my dissertation committee, Professor Jelena Ivetić, Professor Michele Pagani, Professor Zoran Petrić, and Professor Jovana Obradović, for their valuable feedback, insights, and suggestions that helped me to improve my work.

Finally, I want to express my deepest appreciation and love to my husband, Aleksandar, your support, patience, and love have been the foundation upon which I completed this thesis. Your presence in my life has been the most cherished and motivating aspect of this journey. Thank you supporting my dreams, and encouraging me when times are tough.

To my grandfather

Abstract

Over the last decades reasoning about uncertain knowledge has gained an important role in computer science and artificial intelligence. This resulted in the development of different probabilistic models of uncertainty. Since λ-calculus and combinatory logic are models of computation that are suitable for expressing the concepts of programming languages, different approaches of introducing probabilistic reasoning in λ-calculus and combinatory logic have been studied.

This thesis investigates two different approaches for probabilistic reasoning in these calculi. The most usual approach is to extend the language of untyped λ-calculus with probabilistic choice operator which results in probabilistic computation. This approach has shown to be very useful and applicable in various fields, e.g. robotics, natural language processing, and machine learning. Another approach is to extend the language of a typed λ-calculus with probability operators and to obtain a framework for probabilistic reasoning about the typed calculus in the style of probability logic.

The thesis is organized into six chapters. The first chapter describes historical development of the logics that will be studied in the thesis, and the last chapter concludes the thesis. The remaining chapters present the results of the research.

The second chapter studies the lazy call-by-name probabilistic λ-calculus extended with let-in operator $\Lambda_{\oplus,\mathsf{let}}$, and program equivalence in the calculus. Since the proof of context equivalence is quite challenging, we investigate some effective methods for proving the program equivalence. Probabilistic applicative bisimilarity has proved to be a suitable tool for proving the context equivalence in probabilistic setting. We prove that probabilistic applicative bisimilarity is fully abstract with respect to the context equivalence in $\Lambda_{\oplus,\mathsf{let}}$. First, we use Howe's method to prove that probabilistic applicative bisimilarity is a congruence, thus included in the context equivalence. Next, we introduce the testing equivalence, which coincides with bisimilarity, and prove that the context equivalence is included in the testing equivalence, meaning that it is

included in the probabilistic applicative bisimilarity as well.

The third chapter introduces Kripke-style semantics for the full simply typed combinatory logic $CL^{\rightarrow,\times,+}$, that is, the simply typed combinatory logic extended with product types, sum types, empty type and unit type. We present $CL^{\rightarrow,\times,+}$ through its syntax, operational semantics and type assignment system. The Kripke-style semantics is defined as a Kripke applicative structure, which is extensional and has special elements corresponding to basic combinators, provided with the valuation of term variables. We prove that $CL^{\rightarrow,\times,+}$ is sound and complete with respect to the proposed semantics.

The fourth chapter introduces the logic of combinatory logic LCL, that is, a propositional extension of the simply typed combinatory logic. We present the syntax, axiomatization and semantics of LCL. The main contributions of this chapter are the soundness and strong completeness of the logic LCL with respect to the proposed semantics. First, we prove that the equational theory of the simply typed combinatory logic is sound and complete with respect to the presented semantics. Then, we prove that the axiomatization of LCL is sound and strongly complete with respect to the proposed semantics. In addition, we prove that the proposed semantics is the new semantics for the simply typed combinatory logic containing the typing rule that ensures that equal terms inhabit the same type.

The fifth chapter introduces the probabilistic extension of the logic of combinatory logic PCL. We extend the logic LCL with probability operators of the form $P_{\geq s}$ with the intended meaning "probability is at least s", and obtain a framework for probabilistic reasoning about typed combinatory terms. The semantics of PCL is based on the possible world approach, where the set of possible worlds is equipped with a finitely additive probability measure. Due to the non-compactness of the logic, we give an infinitary axiomatization. We prove that the given axiomatization is sound and strongly complete with respect to the proposed semantics.

The results of the thesis have been published in:

- S. Kašterović and M. Pagani. The discriminating power of the let-in operator in the lazy call-by-name probabilistic lambda-calculus. In H. Geuvers, editor, 4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany, volume 131 of LIPIcs, pages 26:1–26:20. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2019. doi: 10.4230/LIPIcs.FSCD.2019.26 URL: https://doi.org/10.4230/LIPIcs.FSCD.2019.26.

- S. Kašterović and S. Ghilezan. Kripke-style semantics and completeness for full simply typed lambda calculus. Journal of Logic and Computation, 30(8):1567–1608, 2020. doi: 10.1093/logcom/exaa055

URL: https://doi.org/10.1093/logcom/exaa055.

- S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Probabilistic reasoning about simply typed lambda terms. In S. N. Artemov and A. Nerode, editors, Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings, volume 10703 of Lecture Notes in Computer Science, pages 170–189. Springer, 2018. doi: 10.1007/978-3-319-72056-2_11. URL: https://doi.org/10.1007/978-3-319-72056-2_11.

- S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović and N. Savić. Towards probabilistic reasoning in type theory - the intersection type case. In A. Herzig and J. Kontinen, editors, Foundations of Information and Knowledge Systems-11th International Symposium, FoIKS 2020, Dortmund, Germany, February 17-21, 2020, Proceedings, volume 12012 of Lecture Notes in Computer Science, pages 122–139. Springer, 2020. doi: 10.1007/978-3-030-39951-1_8. URL https://doi.org/10.1007/978-3-030-39951-1_8.

Contents

Rezime

Uvod

David Hilbert je na međunarodnom kongresu matematičara održanom u Parizu 1900. godine održao izlaganje koje se smatra jednim od najuticajnijih izlaganja u oblasti matematike. U tom izlaganju je predstavio program, koji je kasnije nazvan *Hilbertov program*, u kome je predstavio 23 matematička problema za koje je smatrao da treba da budu izučavani u narednom veku. Jedan od problema koji je privukao veliku pažnju bio je Problem odlučivosti. Hilbert je verovao da se cela matematika može aksiomatizovati. U slučaju da je to moguće, postavio je sledeći problem:

> „Da li postoji procedura, tj. algoritam, koji će za proizvoljan matematički iskaz odlučiti u konačno mnogo koraka da li je taj iskaz tačan ili netačan?"

Hilbertova pretpostavka je bila da takav algoritam postoji. Međutim, kasnije je dokazano da takav algoritam ne postoji. Neki od najznačajnijih negativnih odgovora na Problem odlučivosti su:

- teoreme nepotpunosti (Kurt Gödel),
- λ-račun (Alonzo Church),
- Tjuringove mašine (Alan Turing).

Kurt Gödel je 1931. godine objavio dve teoreme nepotpunosti ([68]), kojima je postavio granicu dokazivosti u formalnim aksiomatskim sistemima. Alonzo Church je želeo da postavi osnove matematike koristeći pojam funkcije, umesto skupa. On je uveo λ-račun, formalni sistem zasnovan na funkcijama ([27]) i pokazao je da problem jednakosti λ-terma nije odlučiv. Njegovi učenici Kleene i Rosser su pokazali da je taj sistem nekonzistentan ([103]). Ipak, deo sistema je bio konzistentan, te je Church izdvojio taj deo i uveo λI-račun ([31]).

Nezavisno od Churcha, Alan Turing je uveo novi formalizam, sada poznate Tjuringove mašine ([165]), i pokazao je da problem zaustavljanja Tjuringove mašine (*Halting problem*) nije odlučiv. U isto vreme se pojavio još jedan formalni sistem, kombinatorna logika. Osnovne ideje kombinatorne logike je predstavio još Moses Schönfinkel u [155], koje je dalje razvio Haskell Curry ([37]). Ono što povezuje sva tri navedena formalna sistema jeste to da su oni iste izražajne moći.

- λ-račun i rekurzivne funkcije su ekvivalentni. (Kleene)
- λ-račun i Tjuringove mašine su ekvivalentni. (Turing)
- λ-račun i kombinatorna logika su ekvivalentni. (Curry)

Razvoj ovih formalnih sistema je nastavljen uvođenjem tipova. Tipove je u kombinatornu logiku uveo Curry ([38]) da bi kontrolisao primenu funkcija. Prvi koji su uveli tipove u logiku bili su Russell i Whitehead ([168]). Uprošćavanjem njihovog tipskog sistema nastali su funkcionalni tipovi ([144]), koji su bili osnova za razvoj mnogih tipskih sistema za λ-račun ([30]). Pregled razvoja tipova od njihovog uvođenja u Principia Mathematica ([168]) do razvoja funkcionalnih tipova u λ-računu ([30]) je predstavljen u [90]. Nakon prvog tipskog sistema za λ-račun i kombinatornu logiku, razvijeni su razni tipski sistemi kao što su tipovi sa presekom, polimorfni tipovi, zavisni tipovi i drugi. Svi ovi tipski sistemi su pronašli primenu u raznim oblastima kao što su automatsko dokazivanje teorema, interaktivno dokazivanje teorema, programski jezici.

U ovoj tezi predstavljena su i istraživana četiri različita formalna sistema: netipizirani verovatnosni λ-račun $\Lambda_{\oplus,\mathsf{let}}$, cela kombinatorna logika $CL^{\rightarrow,\times,+}$, logika kombinatorne logike LCL i verovatnosno proširenje logike kombinatorne logike PCL.

Teza je organizovana u šest poglavlja.

U poglavlju 1 dat je pregled istorijskog razvoja formalnih sistema koji su izučavani u tezi. Ovo poglavlje je podeljeno u tri celine. Najpre je dat kratak uvod u razvoj i značaj verovatnosnih programa. Zatim je opisan jedan od osnovnih problema u ovoj programskoj paradigmi, a to je ekvivalentnost programa. Dalje, detaljno je opisan istorijski razvoj semantika mogućih svetova pri čemu su istaknute osnovne karakteristike ovih semantika. Navedeni su neki od najznačajnijih dokaza potpunosti raznih logika u odnosu na semantike mogućih svetova. Naglašen je značaj Kripkeovih semantika, koje su nastale kao semantike za modalne logike, ali su kasnije prilagođene za intuicionističku logiku, druge neklasične logike, kao i za tipizirani lambda račun. Na kraju ovog poglavlja, opisan je razvoj verovatnosnih logika od antičkog vremena do danas.

Verovatnosno izračunavanje

Poglavlje 2 izučava verovatnosni λ-račun. Tokom poslednjih decenija zaključivanje u prisustvu neizvesnosti je dobilo važnu ulogu u računarstvu i veštačkoj inteligenciji. To je rezultiralo razvojem različitih alata koji se bave neizvesnošću i jedan primer takvih alata su verovatnosni modeli. Verovatnosni modeli su veoma korisni i primenjivi u različitim oblastima, kao što su robotika ([163]), mašinsko učenje ([135]) i obrada prirodnog jezika ([117]). Da bi se opisali verovatnosni modeli, razvijeni su verovatnosni programski jezici, koji su inspirisani različitim programskim paradigmama (funkcionalna, imperativna, objektno-orijentisana i druge). Dodavanjem verovatnosnog operatora determinističkom jeziku dobijamo novu programsku paradigmu verovatnosno izračunavanje. Vrsta determinističkih jezika pogodnih za verovatnosna izračunavanja jesu funkcionalni programski jezici. Kako su funkcionalni programski jezici bazirani na λ-računu, verovatnosni λ-račun je postao važna tema istraživanja.

Jedan od glavnih problema u verovatnosnom programiranju jeste dokazivanje ekvivalentnosti dva programa. Dokaz ekvivalentnosti programa u jezicima višeg reda nije jednostavan zadatak, jer treba pokazati da se programi ponašaju isto u svim kontekstima, a njih ima beskonačno mnogo. Stoga je cilj pronaći efikasniju metodu za utvrđivanje ekvivalentnosti programa. Alat koji se pokazao kao pogodan za ispitivanje ekvivalentnosti programa jeste *bisimulacija*. Bisimulacija je prvu put uvedena u konkurentnom računarstvu kao relacija koja karakteriše ponašanje procesa ([122, 133]). Kasnije je relacija bisimulacije izučavana i u λ-računu. Abramsky je uveo pojam aplikativne bisimulacije u jezike višeg reda ([1]), a Larsen i Skou su uveli pojam verovatnosne bisimulacije za označene lance Markova ([111]). Iz ove dve relacije je proizašla nova relacija bisimulacije poznata kao *verovatnosna aplikativna bisimulacija*, koja je pogodna za ispitivanje ekvivalentnosti verovatnosnih programa ([35, 36, 47]). Ono što čini bisimulaciju moćnom metodom je činjenica da je dovoljno naći jednu relaciju bisimulacije koja sadrži dva terma da bismo pokazali da su ta dva terma bisimilarna. Odnosno, bisimilarnost dva terma je definisana pomoću egzistencijalnog kvantifikatora za razliku od kontekstne ekvivalencije koja je definisana preko univerzalnog kvantifikatora.

Bisimulacija je korisna samo ukoliko je saglasna u odnosu na kontekstnu ekvivalenciju. Saglasnost bisimulacije se najčešće pokazuje primenom Howeovog metoda ([84]). Primena bisimulacije u dokazivanju verovatnosnih programa je izučavana u raznim okruženjima ([35, 36, 43, 44, 47, 48]). U izračunavanju postoje različite strategije za evaluaciju programa kao što su poziv-po-imenu (call-by-name), poziv-po-vrednosti (call-by-value), poziv-po-potrebi (call-by-need) i druge. U tezi su posmatrane dve strategije: poziv-po-imenu i

poziv-po-vrednosti. U okruženju gde je primenjena strategija *poziv-po-imenu*,

$$\text{term } (\lambda x.M)N \text{ se redukuje na term } M\{N/x\} \text{ za bilo koji term } N,$$

pri čemu $M\{N/x\}$ predstavlja term koji se dobija zamenom slobodnih pojavljivanja promenljive x u termu M termom N. Sa druge strane, ako je primenjena *poziv-po-vrednosti* strategija,

$$\text{term } (\lambda x.M)V \text{ se redukuje na term } M\{V/x\} \text{ samo ako je } V \text{ izračunata vrednost.}$$

Drugim rečima, u poziv-po-imenu strategiji neophodno je najpre izvršiti redukciju argumenta i tek onda se dobijena vrednosti prosleđuje funkciji.

Rad predstavljen u drugom poglavlju teze je motivisan rezultatima predstavljenim u [35] i [47], gde su autori izučavali verovatnosni λ-račun, to jest λ-račun proširen verovatnosnim operatorom izbora. U radu [35], Crubillé i Dal Lago posmatrali su račun u kome je primenjena poziv-po-vrednosti strategija i pokazano je da se verovatnosna aplikativna bisimulacija i kontekstna ekvivalencija poklapaju. Sa druge strane, u radu [47] Dal Lagi, Sangiorgi i Alberti posmatrali su račun u kome je primenjena poziv-po-imenu strategija i pokazano je da je verovatnosna aplikativna bisimulacija sadržana u kontekstnoj ekvivalenciji, ali da se one ne poklapaju. Termi koji čine razliku između ove dve relacije su termi

$$M = \lambda x.\lambda y.(x \oplus y) \text{ i } N = (\lambda x.\lambda y.x) \oplus (\lambda x.\lambda y.y),$$

koji su u datom računu kontekstno ekvivalentni i nisu bisimilarni. Pored toga, autori su pretpostavili da bi se dodavanjem operatora sekvenciranja dobio račun u kome se verovatnosna aplikativna bisimulacija poklapa sa kontekstnom ekvivalencijom. Ova pretpostavka je potvrđena u publikovanom radu

[99] S. Kašterović and M. Pagani. The discriminating power of the let-in operator in the lazy call-by-name probabilistic lambda-calculus. In H. Geuvers, editor, 4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany, volume 131 of LIPIcs, pages 26:1–26:20. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2019. doi: 10.4230/LIPIcs.FSCD.2019.26. URL: https://doi.org/10.4230/LIPIcs.FSCD.2019.26.

na koji se i oslanja drugo poglavlje teze.

U ovom poglavlju predstavljen je verovatnosni λ-račun $\Lambda_{\oplus,\mathsf{let}}$, odnosno netipizirani λ-račun proširen sa dva operatora: verovatnosnim operatorom izbora $\oplus$ i let-in operatorom. U računu je implementirana lenja poziv-po-imenu

strategija evaluacije. Verovatnosni operator $\oplus$ predstavlja izbor, u smislu da se term $M \oplus N$ redukuje na terme M i N sa istom verovatnoćom. Operator let-in oponaša poziv-po-vrednosti strategiju evaluacije u datom okruženju. Izučavali smo problem ekvivalencije programa u $\Lambda_{\oplus,\mathsf{let}}$-računu. Pored kontekstne ekvivalencije posmatrali smo još dve relacije ekvivalencije: bisimulaciju i ekvivalenciju testiranjem.

Najpre smo operacionu semantiku $\Lambda_{\oplus,\mathsf{let}}$-računa predstavili kao označen lanac Markova, a onda smo uveli pojam verovatnosne aplikativne bisimulacije za $\Lambda_{\oplus,\mathsf{let}}$-račun. Prvi rezultat ovog poglavlja je dokaz da je verovatnosna aplikativna bisimulacija kongruencija, što smo pokazali pomoću Hoveovog metoda ([84]). Kao posledicu ovog rezultata dobili smo da je verovatnosna aplikativna bisimulacija sadržana u kontekstnoj ekvivalenciji. Da bismo pokazali suprotan smer, odnosno da je kontekstna ekvivalencija sadržana u verovatnosnoj aplikativnoj bisimulaciji, uveli smo pojam testa u $\Lambda_{\oplus,\mathsf{let}}$-račun po ugledu na [35]. Pomoću testova smo definisali novu relaciju ekvivalencije na skupu termova, koju smo nazvali *ekvivalencija testiranjem*. Dva terma su u relaciji ekvivalencije testiranjem ako je svaki test uspešno izvršen sa istom verovatnoćom na oba terma. Ekvivalencija testiranjem se poklapa sa verovatnosnom aplikativnom bisimulacijom ([47]). Drugi značajan doprinos ovog poglavlja jeste dokaz da za svaki test postoji kontekst takav da je verovatnoća uspešnosti testa primenjenog na neki term jednaka verovatnoći konvergencije terma koji se dobija primenom konteksta na dati term. Zatim, koristeći ovaj rezultat pokazali smo da je kontekstna ekvivalencija sadržana u ekvivalenciji testiranja. Stoga, sve tri relacije ekvivalencije se poklapaju.

Kripkeove semantike za ceo račun sa funkcionalnim tipovima

Poglavlje 3 proučava $CL^{\rightarrow,\times,+}$-račun, odnosno kombinatornu logiku sa funkcionalnim tipovima, koja je proširena tipovima proizvoda, tipovima sume, praznim tipom i jediničnim tipom. Uveli smo Kripkeovu semantiku za $CL^{\rightarrow,\times,+}$-račun.

Kripkeova semantika je jedna od najpopularnijih semantika mogućih svetova. Moderno doba razvoja semantika mogućih svetova započeo je Pierce ([75, 76]). On je smatrao da kondicionale treba analizirati pomoću kvantifikacije nad mogućim svetovima:

> Kvantifikovani subjekt hipotetičkog iskaza je mogućnost ili mogući slučaj ili moguće stanje stvari.

Tokom razvoja semantika mogućih svetova, istraživači su se vodili sledećim

idejama:

- analiza modaliteta pomoću kvantifikacije nad mogućim slučajevima,
- potreba binarne relacije kao relacije dostižnosti između svetova,
- traganje za dokazom kompletnosti.

Na razvoj semantika mogućih svetova je značajan uticaj imao i Wittgenstein ([169]), koji uveo ideju logičkog prostora i koji je smatrao da su tvrdnje smeštene u prostoru. Ove ideje su dovele do razvoja formalne semantike za modalnu logiku $S5$, koju je uveo Carnap ([26, 149]). Semantika koju je uveo Carnap nije imala relaciju dostižnosti svetova.

Binarnu relaciju između svetova su prvi put uveli Prior i Meredith ([121]). Prior je razvio temporalnu logiku zamenjujući modalne operatore nužnosti i mogućnosti operatorima vremena ([140–143]), eksplicitno je uveo binarnu relaciju i interpretirao je kao relaciju dostižnosti.

Prve dokaze potpunosti predstavili su Bayart, Hintikka i Kripke. Bayart ([14]) i Kripke ([108]) su dokazali potpunost proširenja logike $S5$ kvantifikatorima. Kasnije je Kripke koristeći ovaj metod pokazao potpunosti iskaznih modalnih sistema ([109]). Hintikka je u svojim neformalnim izlaganjima, koja je držao u Bostonu, predstavio dokaze potpunosti za sisteme M, $S4$ i $S5$ sa kvantifikatorima. Značajan doprinos razvoju semantika mogućih svetova dali su i Feys ([58]), McKinsey ([120]), Becker ([15]) i Montague ([125]). Detaljan pregled razvoja semantika mogućih svetova je dat u [32].

Kripke je jedini u svojim semantikama koristio univerzalni pojam valjanosti, posmatrao svetove kao tačke u evaluaciji i formalizovao relaciju dostižnosti među svetovima. Karakterizacija svetova kao pojedinačnih tačaka u evaluaciji mu je omogućila da sistematično izvede dokaze potpunosti za razne modalne logike. Iako je Kripkeova semantika prvobitno uvedena kao semantika za modalnu logiku, ona je kasnije prilagođena intuicionističkoj logici ([110]), ali i drugim neklasičnim logikama ([91]).

Zahvaljujući Curry-Howard korespondenciji ([83]) između intuicionističke logike i tipiziranog λ-računa, Kripkeove semantike su našle svoju primenu i u λ-računu. Mitchell i Moggi su u radu ([124]) uveli Kripkeove modele za λ-račun sa funkcionalnim tipovima. Semantika za $CL^{\rightarrow,\times,+}$-račun koja je uvedena u trećem poglavlju ove teze je upravo inspirisana Kripkeovim modelima iz [124].

$CL^{\rightarrow,\times,+}$-račun smo uveli predstavljajući njegov jezik, operacionu semantiku, odnosno jednakosnu teoriju koja je proizašla iz relacije redukcije, i tipski sistem. Relacija redukcije je definisana tako da zavisi od relacije tipiziranja. Sa druge strane, da bismo osigurali da jednaki termi imaju isti tip, tipski sistem smo definisali tako da relacija tipiziranja zavisi od relacije redukcije. Stoga, jednakosna teorija, koja je proizašla iz relacije redukcije, i tipski sistem

su definisani istovremeno. Kripkeova semantika za $CL^{\rightarrow,\times,+}$, koja je predstavljena u ovom poglavlju, je uvedena u radu

[94] S. Kašterović and S. Ghilezan. Kripke-style semantics and completeness for full simply typed lambda calculus. Journal of Logic and Computation, 30(8):1567–1608, 2020. doi: 10.1093/logcom/exaa055.
URL: https://doi.org/10.1093/logcom/exaa055.

Najpre smo definisali Kripkeovu aplikativnu strukturu, a zatim smo uveli pojmove ekstenzionalne Kripkeove aplikativne strukture i Kripkeove aplikativne strukture sa kombinatorima. Kripkeova semantika je definisana kao ekstenzionalna Kripkeova aplikativna struktura sa kombinatorima, kojoj je pridružena valuacija promenljivih.

Glavni rezultati ovog poglavlja jesu dokazi saglasnosti i potpunosti $CL^{\rightarrow,\times,+}$-računa u odnosu na predloženu semantiku. Prvo smo pokazali da su jednakosna teorija i tipski sistem saglasni. U ovom dokazu smo koristili metod matematičke indukcije. Zatim, uveli smo pojam kanoničkog modela i pokazali da su jednakosna teorija i tipski sistem potpuni u odnosu na predloženu semantiku. Za konzistentnu bazu smo definisali kanonički model tako da je interpretacija terma klasa ekvivalencije datog terma u odnosu na jednakosnu teoriju. Koristeći ovu osobinu kanoničkog modela dokazali smo potpunost jednakosne teorije i tipskog sistema.

Logika kombinatorne logike

U poglavlju 4 smo predstavili logiku kombinatorne logike LCL. Logika LCL je iskazno proširenje kombinatorne logike sa funkcionalnim tipovima, tj. dobijena je definisanjem klasične iskazne logike nad kombinatornom logikom sa funkcionalnim tipovima.

Ova logika predstavlja formalni sistem za zaključivanje o tipiziranim izrazima, što nam omogućava primenu metoda kao što su DPLL procedura, metod rezolucije, SAT rešavači, SMT rešavači i drugi, na kombinatornu logiku sa funkcionalnim tipovima. Na osnovu Curry-Howard korespondencije, logiku LCL možemo posmatrati kao prvi korak ka razvoju automatskog alata za zaključivanje o tipiziranim termima i programima.

Razna proširenja kombinatorne logike su izučavana sa ciljem da se razvije formalni sistem čija će izražajna moć omogućiti uvođenje novih paradigmi. Jedan način proširenja kombinatorne logike je već predstavljen u trećem poglavlju ove teze, gde smo kombinatornoj logici sa funkcionalnim tipovima dodali nove operatore i tipove. Drugi način da se proširi kombinatorna logika jeste da se kombinuje sa nekim logičkim sistemom. Ideju kombinovanja različitih

logičkih sistema sa ciljem da se opiše zaključivanje o određenim logičkim strukturama predstavio je Scott u [158], gde je tipizirani sistem kombinatora, koji uključuje i kombinator fiksne tačke, proširen logičkim konstantama i veznicima, čime je dobijen deduktivni sistem za izračunljive funkcije. Sličan pristup koristio je Beeson za definisanje lambda logike ([16]), koja predstavlja alat za reprezentaciju funkcija, a dobijena je kao unija logike prvog reda i λ-računa. U radu [6] Axelsen, Glück i Kaarsgaard nisu predstavili proširenje kombinatorne logike, ali je ideja veoma slična našoj. Autori su definisali klasičnu iskaznu logiku nad reverzibilnim logičkim kolima i razvili formalni sistem za zaključivanje o reverzibilnim logičkim kolima. Iako su slične ideje već ranije primenjivane, logika LCL je prvi put uvedena u ovoj tezi.

Predstavili smo jezik, aksiomatizaciju i semantiku LCL logike. Atomičke formule u logici LCL su tipizirani izrazi iz kombinatorne logike sa funkcionalnim tipovima $CL_{\rightarrow}$. Skup svih tipiziranih izraza je skup svih izraza oblika $M : \sigma$, gdje je M term i σ tip takav da postoji baza u kojoj će term M dobiti tip σ.

Skup formula logike LCL je definisan sledećom gramatikom:

$$\boxed{\alpha := M : \sigma \mid \alpha \Rightarrow \alpha \mid \neg\alpha}$$

Aksiomatizacija logike LCL je dobijena iz tipskog sistema kombinatorne logike sa funkcionalnim tipovima i aksiomatskog sistema za klasičnu iskaznu logiku. Semantika je definisana kao ekstenzionalna aplikativna struktura proširena specijalnim elementima, koji odgovaraju osnovnim kombinatorima, i kojoj je pridružena valuacija promenljivih. Sličan pristup smo koristili za definisanje semantike kombinatorne logike sa tipovima sa presekom u radu

[62] S. Ghilezan, S. Kašterović. Semantics for Combinatory Logic With Intersection Types, Frontiers in Computer Science, volume 4, 2022. doi: 10.3389/fcomp.2022.792570.
URL: https://www.frontiersin.org/articles/10.3389/fcomp.2022.792570

Glavni rezultati ovog poglavlja su dokazi saglasnosti i jake potpunosti logike LCL u odnosu na uvedenu semantiku. Najpre smo dokazali da je jednakosna teorija kombinatorne logike sa funkcionalnim tipovima saglasna i potpuna u odnosu na definisanu semantiku. Zatim smo dokazali da je aksiomatizacija logike LCL saglasna. Kao i u prethodnom poglavlju, u dokazu saglasnosti koristili smo matematičku indukciju. U dokazu potpunosti jednakosne teorije definisali smo LCL-model takav da je interpretacija terma u datom modelu jednaka klasi ekvivalencije u odnosu na jednakosnu teoriju kombinatorne logike sa funkcionalnim tipovima. Na osnovu navedene osobine modela pokazali smo da važi potpunost jednakosne teorije. Dokaz jake potpunosti aksiomatizacije logike LCL je nešto složeniji i sastoji se iz sledećih koraka:

1. Najpre smo dokazali teoremu dedukcije, koja je neophodna za dokaz jake potpunosti.

2. Zatim, pokazali smo da svaki konzistentan skup T može biti proširen do maksimalno konzistentnog skupa $T^\star$.

3. Koristeći maksimalno konzistentan skup $T^\star$, definisali smo kanonički model.

4. Pokazali smo da kanonički model jeste LCL-model i da zadovoljava samo formule iz skupa $T^\star$.

5. Koristeći kanonički model dokazali smo da je svaki konzistentan skup zadovoljiv.

6. Konačno, izveli smo dokaz jake potpunosti za logiku LCL.

Dalje, pokazali smo da je uvedena semantika za logiku LCL takođe i nova semantika za kombinatornu logiku sa funkcionalnim tipovima. Posmatrali smo kombinatnornu logiku sa funkcionalnim tipovima proširenu sa pravilom za tipiziranje koje osigurava da jednaki termi imaju isti tip i pokazali smo da je ona saglasna i potpuna u odnosu na semantiku za LCL. Time je dokazano da je logika LCL zapravo konzervativno proširenje kombinatorne logike sa funkcionalnim tipovima.

Verovatnosno zaključivanje u teoriji funkcionalnih tipova

Poglavlje 5 uvodi logiku PCL, koja je verovatnosno proširenje logike kombinatorne logike. Prateći pristup koji je korišćen u razvoju verovatnosnih proširenja različitih logika, kao što su klasična iskazna logika, intuicionistička iskazna logika, logika opravdanja ([132]), proširili smo jezik logike LCL verovatnosnim operatorima oblika $P_{\geq s}$, koji imaju značenje: „verovatnoća je bar s".

Priča o verovatnosnoj logici potiče još iz antičkog vremena. Sofisti Corax i Tisias koristili su pojam neizveznosti u svojim argumentima o zakonskim, medicinskim i političkim pitanjima ([107]). Cardano, Tartaglia, de Fermat, Pascal, Huygens i brojni matematičari izučavali su igre na sreću i uveli su nove ideje u vezi zaključivanja u slučaju neizvesnosti. Ideju da se verovatnoća može numerički kvantifikovati i da se metode korišćene u igrama na sreću mogu primeniti na izračunavanje verovatnoće prvi put su predstavili Arnauld i Pierre u [3]. Reč „verovatnoća" u sadašnjem značenju prvi je uveo Leibniz kada je definisao verovatnoću kao odnos broja poželjnih slučajeva i

broja ukupnih slučajeva ([113]). Pascal, Huygens i Leibniz su imali veliki uticaj na rad Jacoba Bernoullija ([18]), koji se smatra osnivačem teorije verovatnoće. Značajan doprinos razvoju teorije verovatnoće dali su i De Morgan, Bayes i Boole. Boole je među prvima izučavao vezu između logike i verovatnoće ([23]).

Nakon rada Leibniza i Boola najveći napredak u razvoju verovatnosne logike napravili su Keisler, Hoover, Hamblin i Hailperin. Keisler ([100, 101]) je proučavao raspodelu verovatnoće na domenima struktura prvog reda i uveo verovatnosne kvantifikatore oblika $Px > r$ gde formula $(Px > r)\varphi(x)$ ima značenje:

„verovatnoća skupa $\{x \mid \varphi(x)\}$ je veća od r".

Hoover ([82]) je dao potpunu aksiomatizaciju za Keislerove logike i zajedno za Keislerom je dokazao teoreme potpunosti za različite modele kao što su verovatnosni, analitički, i drugi. Vezu između verovatnoće i modalne logike uočio je Hamblin ([74]) i on je uveo verovatnosni, modalni operator. Hailperin ([72]) je iskoristio metode linearnog programiranja da izvede proceduru za dobijanje najboljih mogućih granica za verovatnoću iskaznih formula u slučaju da su verovatnoće potformula poznate. Moderni razvoj verovatnosne logike je započeo Nilsson ([127]) kada je uveo prvi sistem za formalno verovatnosno zaključivanje.

Nilssonov rad je imao veliki uticaj na mnoge istraživače koji su razvijali formalne sisteme za verovatnosno zaključivanje ([56, 57, 73, 167]). Naš rad se najviše oslanja na beskonačne verovatnosne logike uvedene u [104, 119, 129–131, 147], gde su predstavljena verovatnosna proširenja raznih logika kao što su klasična iskazna logika, intuicionistička iskazna logika, temporalna logika, logika opravdanja i druge.

Ideju o verovatnosnom proširenju λ-računa i kombinatorne logike smo prvi put predstavili u publikovanim radovima

[64] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Probabilistic reasoning about simply typed lambda terms. In S. N. Artemov and A. Nerode, editors, Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings, volume 10703 of Lecture Notes in Computer Science, pages 170–189. Springer, 2018. doi: 10.1007/978-3-319-72056-2_11. URL https://doi.org/10.1007/978-3-319-72056-2_11.

[66] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Towards probabilistic reasoning in type theory - the intersection type case. In A. Herzig and J. Kontinen, editors, Foundations of Information and Knowledge Systems-11th International Symposium, FoIKS 2020, Dortmund, Germany, February 17-21, 2020, Proceedings, volume 12012 of

Lecture Notes in Computer Science, pages 122–139. Springer, 2020. doi: 10.1007/978-3-030-39951-1_8. URL https://doi.org/10.1007/978-3-030-39951-1_8.

Modeli uvedeni u ovim radovima su bazirani na dobro poznatim modelima λ-računa: term modelima i filter modelima. Jezik ovih sistema je definisan tako što smo najpre definisali iskazno proširenje tipiziranog λ-računa i dobili skup formula, koje smo nazvali osnovne formule. Zatim smo primenili verovatnosne operatore na osnovne formule i posmatrali bulovske kombinacije tako dobijenih verovatnosnih formula. Problem kod ovih sistema je taj što poznati modeli λ-računa nisu pogodni da opišu iskazno zaključivanje o tipiziranim termima. Drugim rečima, baza ove logike, a to su bulovske kombinacije tipiziranih izraza, nije potpun sistem u odnosu na navedene modele λ-računa. Ovaj problem je motivisao razvoj logike LCL i modela za ovu logiku, koji su predstavljeni u četvrtom poglavlju. U petom poglavlju smo primenili dobro poznat metod za razvoj logike PCL, koja predstavlja verovatnosno proširenje logike LCL.

Predstavili smo jezik, semantiku i aksiomatizaciju logike PCL. Jezik logike PCL je proširenje logike LCL sa verovatnosnim operatorima, a skup formula je definisan kao unija dva skupa: skupa osnovnih formula i skupa verovatnosnih formula. Osnovne formule su zapravo formule logike LCL, dok su verovatnosne formule dobijene primenom verovatnosnih operatora na osnovne formule. Najpre definišemo atomičke verovatnosne formule.

> Za $s \in [0,1] \cap \mathbb{Q}$ i formulu α logike LCL, $P_{\geq s}\alpha$ je atomička verovatnosna formula.

Skup svih verovatnosih formula je skup svih bulovskih kombinacija atomičkih verovatnosnih formula, odnosno generisan je sintaksom

$$\boxed{\varphi ::= P_{\geq s}\alpha \mid \varphi \wedge \varphi \mid \neg\varphi}$$

Semantika verovatnosnih logika je zasnovana na strukturama definisanim nad mogućim svetovima, gde u svakom svetu postoji konačno aditivna mera. Semantika logike PCL je definisana tako da svaki mogući svet određuje jedan LCL-model. Aksiomatizacija logike PCL je dobijena iz aksiomatizacije logike LCL i aksiomatizacije verovatnosne logike. Logika PCL ima beskonačnu aksiomatizaciju u smislu da sadrži jedno beskonačno pravilo, to jest pravilo sa prebrojivim skupom premisa. Beskonačna aksiomatizacija je posledica nekompaktnosti logike PCL. Logika PCL ne zadovoljava teoremu kompaktnosti jer je skup

$$X = \{\neg P_{=0}(x : \sigma)\} \cup \{P_{<\frac{1}{n}}(x : \sigma) \mid n \in \mathcal{N}\}$$

takav da je svaki njegov konačan podskup zadovoljiv, dok sam skup X nije zadovoljiv. Dakle, nijedna konačna aksiomatizacija neće biti saglasna i jako potpuna.

Najvažniji rezultati ovog poglavlja su dokazi saglasnosti i jake potpunosti date aksiomatizacije logike PCL u odnosu na uvedenu semantiku. Slično kao u prethodnim poglavljima, dokaz saglasnosti je dobijen primenom matematičke indukcije. Dokaz jake potpunosti je nešto složeniji. Primenili smo metod kao i za logiku LCL, te smo najpre dokazali teoremu dedukcije. Zatim smo dokazali da svaki konzistentan skup može biti proširen do maksimalno konzistentnog skupa. Konstruisali smo PCL-model u kome su tačne samo one formule koje pripadaju datom maksimalno konzistentnom skupu. Koristeći ovaj model, pokazali smo da je svaki konzistentan skup zadovoljiv, odakle sledi jaka potpunost logike PCL.

Rezultati

Poglavlje 6 sadrži sažetak postignutih rezultata, pregled literature i razmatra pravce daljih istraživanja.

Disertacija daje značajan doprinos razvoju formalnih modela za verovatnosno zaključivanje u izračunavanju i teoriji tipova i razvoju Kripkeovih semantika za modele izračunavanja.

- Proučavan je verovatnosni λ-račun sa let-in operatorom gde je primenjena lenja poziv-po-imenu strategija evaluacije i izučavan je problem ekvivalencije verovatnosnih programa u ovom okruženju. Cilj je bio pronaći efikasan metod za dokazivanje kontekstne ekvivalencije. Dati su dokazi da se u posmatranom računu relacija verovatnosne aplikativne bisimulacije i relacija kontekstne ekvivalencije poklapaju, što čini verovatnosnu aplikativnu bisimulaciju pogodnim metodom za dokazivanje kontekstne ekvivalencije. Ovi rezultati su predstavljeni u poglavlju 2.

- Uvedena je nova Kripkeova semantika za kombinatornu logiku sa funkcionalnim tipovima koja je proširena sa tipovima proizvoda, tipovima sume, praznim tipom i jediničnim tipom. Dati su detaljni dokazi da je tipski sistem posmatranog računa saglasan i potpun u odnosu na uvedenu semantiku. Ovi rezultati su predstavljeni u poglavlju 3.

- Razvijen je potpuno nov koncept verovatnosnog zaključivanja u teoriji tipova i računima sa tipovima koji je zasnovan na novom modelu za verovatnosno zaključivanje o tipiziranim programima. Najpre je uvedena nova logika, pod nazivom Logika kombinatorne logike, koja predstavlja

iskazno proširenje kombinatorne logike sa funkcionalnim tipovima. Pokazano je da je uvedena logika saglasna i potpuna u odnosu na predloženu semantiku. Zatim je logika kombinatorne logike proširena verovatnosnim operatorima, čime je dobijena logika koja omogućava verovatnosno zaključivanje o tipiziranim programima. Takođe, dati su dokazi saglasnosti i potpunosti dobijene logike u odnosu na predloženu semantiku. Ovi rezultati su predstavljeni u poglavljima 4 i 5.

xx

Chapter 1

Introduction

In 1900, David Hilbert gave a talk at the International Congress of Mathematicians held in Paris, France. In this talk, which is considered as the most influential talk ever given by a mathematician, he presented Hilbert's Program. Hilbert's Program addressed 23 major mathematical problems that should be studied in the coming century. Many researchers were influenced by Hilbert's Program and worked on solutions of the presented problems. One of the challenges presented in Paris was *the Entscheidungsproblem*, or *Decision Problem*. Hilbert believed that all of mathematics could be axiomatized. In the case this axiomatization is done, he addressed the following question:

> "Is there an "effective procedure", i.e. an algorithm that takes as input any precise mathematical statement and after a finite number of steps decides whether the statement is true or false?"

Hilbert assumed that such an algorithm exists and just has to be found. However, it turned out that this is not true. The most popular results which refuted Hilbert's conjecture are:

- Incompleteness Theorems by Kurt Gödel,
- λ-calculus by Alonzo Church,
- Turing Machines by Alan Turing.

In 1931, Kurt Gödel ([68]) published two incompleteness theorems that give the limit of provability in formal axiomatic theories. The incompleteness theorems stand as two of the most important results in the history of the mathematical logic.

> **The first incompleteness theorem:** Any consistent formal system F within which a certain amount of elementary arithmetic can be carried out is incomplete; i.e. there are statements of the language of F which can neither be proved nor disproved in F.
>
> **The second incompleteness theorem:** For any consistent system F within which a certain amount of elementary arithmetic can be carried out, the consistency of F cannot be proved in F itself.

Alonzo Church was also interested in the Entscheidungsproblem. He wanted to redefine the very foundations of mathematics using functions instead of sets. In [27], Church has introduced λ-calculus, a formal system based on functions, which included primitively a notion of abstraction and application, and many other notions: a two-place predicate for extensional equality, an existential quantifier, negation, conjunction and the unique solution of a function. He has proved that the problem of equality of terms is undecidable. However, Church's students Kleene and Rosser have discovered Kleene-Rosser paradox ([103]) and proved the inconsistency of Church's original system. The computational part of the system was proved to be consistent, so Church isolated this part and introduced λI-calculus in [31] as a formalism for defining the notion of computability. He has defined an algorithm in terms of λ-calculus and proved that the Entscheidungsproblem is unsolvable in [28, 29]. Kleene ([102]) proved that the set of λ-definable functions coincides with the set of recursive functions, i.e. computable functions in the sense of Herbrand and Gödel.

Independently, Alan Turing also gave a negative answer to the Entscheidungsproblem. In order to formally introduce the notion of an algorithm he invented a new formalism, called *Turing machines* ([165]), and defined an algorithm as anything that can be computed by a Turing Machine. Turing has proved that Halting problem is undecidable. Then he proved that there exist undecidable problems that cannot be solved by any Turing Machine such as Halting Problem. Later, Turing became Church's graduate student at Princeton and proved that the λ-calculus and Turing Machines are computationally equivalent: they define the same class of computable functions.

Another formalism that was independently invented is *Combinatory logic*. The basic idea of combinatory logic was presented by Moses Schönfinkel in 1920s ([155]), but the foundations of combinatory logic have been established by Haskell Curry in 1930s ([37]). Combinatory logic does not use bound variables, which results in a simpler syntax and avoids the obstacles that emerge from bound variables in λ-calculus. Curry has proved that λ-calculus and combinatory logic are computationally equivalent. So, all three formalisms, λ-calculus, Turing machines and combinatory logic, have the same expressive

power:

Kleene: Equivalence of λ-calculus and recursive functions.

Turing: Equivalence of λ-calculus and Turing machines.

Curry: Equivalence of λ-calculus and Combinatory logic.

In order to control the application of functions, types in combinatory logic were introduced in [38]. Russell and Whitehead were first to introduce types in logic ([168]) and by simplifying their type theory the simple theory of types was introduced in [144]. Many modern type systems such as the simply typed λ-calculus ([30]) are based on the theory introduced in [144]. The development of types from the first types introduced in Principia Mathematica ([168]) until the simple types for λ-calculus ([30]) has been described in [90]. The simple types for λ-calculus is the first type system introduced. It is followed by various type systems such as intersection types, polymorphic types, dependent types, and others. These type systems found applications in various fields, e.g. automated theorem provers, proof assistants, programming languages.

1.1 Probabilistic programming

Over the last decades reasoning about uncertain knowledge has played an important role in computer science and artificial intelligence. Sometimes, dealing with uncertainty and incomplete information is not an alternative but rather a necessity. For example, in computational cryptography secure public key encryption schemes have to be probabilistic ([69]). Therefore, it was essential to develop tools that will deal with uncertainty such as probabilistic models. The probabilistic models have proved to be extremely applicable and useful in various areas, such as robotics ([163]), machine learning ([135]) and natural language processing ([117]).

In order to describe probabilistic models, the mechanics to perform inference in those models in various probabilistic programming languages have been introduced ([70, 89, 134, 136, 145]). These languages have been inspired by different programming paradigms such as functional, imperative, object oriented. One approach used to obtain probabilistic models is to add primitives for probabilistic choice to the deterministic language. In this way, we shift from the usual, deterministic computation to a new paradigm, called *probabilistic computation*. Deterministic languages that get well with probabilistic computation are functional languages. Functional programming languages such as Lisp, Scheme, Miranda, ML and others are based on λ-calculus and many existing probabilistic programming languages ([70, 136]) are designed around

λ-calculus or one of its incarnations, like Scheme. All this has influenced the foundational research about probabilistic λ-calculi.

One of the most challenging problems in probabilistic programming is proving equivalence of programs. Two programs are considered equivalent if they behave "in the same manner" in any possible context ([126]). Proving that two terms are equivalent is rather difficult because of the universal quantifier in the definition of equivalence, since one should consider behaviour of the program in any context and there are infinitely many contexts. On account of this, the goal is to find effective methods for context equivalence proofs in higher-order languages.

A technique which has proved to be suitable for characterizing program equivalence is *bisimilarity*. Bisimilarity has emerged in Concurrency Theory as the notion that characterizes the behavioural equality for processes. It was first introduced by Milner and Park ([122, 133]) and since then it has become a fundamental concept in the theory of concurrency. Bisimilarity is defined as the union of all bisimulations, where a bisimulation is a relation on the set of terms of a language. So, in order to prove that two terms are bisimilar, it is enough to find a bisimulation which contains a pair of the terms. The use of the existential quantifier instead of the universal one makes this a powerful proof method.

Over the years, the bisimilarity has been extensively studied in λ-calculi as well. Today it is employed in a number of areas of computer science such as object-oriented languages, functional languages, types, data types, program analysis, verification tools, etc. Abramsky ([1]) introduced the notion of bisiumulation, called applicative bisimulation, into higher-order languages. This notion of bisimulation has been studied by a number of researchers ([71, 112, 114, 139, 150]). Applicative bisimilarity is useful only if it is sound with respect to the context equivalence. For this to hold, it is necessary that bisimilarity is a congruence. The proof that bisimilarity is a congruence is not trivial and in the case of applicative bisimilarity, a common scheme consists in following Howe's approach [84]. Trying to simplify the proof of congruence and accommodate language extensions, different forms of bisimulation have been proposed ([106, 151, 152]).

Another form of bisimulation, called *probabilistic bisimulation* has been introduced for labelled Markov chains by Larsen and Skou ([111]). From applicative bisimulation and probabilistic bisimulation, *probabilistic applicative bisimulation* has emerged. Probabilistic applicative bisimilarity has shown to be an effective method for equivalence proof of probabilistic programs ([35, 36, 47]).

1.2 Possible world semantics

The modern era of possible world semantics started with the work of Peirce ([75, 76]), who argued that the conditional should be analysed in terms of quantification over possible worlds:

> The quantified subject of a hypothetical proposition is a possibility, or possible case, or possible state of things.

In the history of possible world semantics, there are three trains of thoughts that were followed:

1. the analysis of modalities in terms of quantifications over possibilia;
2. the use of a binary relation as an accessibility relation between worlds;
3. the quest for completeness proofs.

The development of the possible world semantics was greatly influenced by the work of Wittgenstein ([169]), who introduced the idea of logical space and thought of claims as being "located" in a kind of space. These ideas evolved into formal semantics for modal logic S5 introduced by Carnap ([26, 149]). The first technical work in possible worlds semantics is presented in [26]. However, Carnap did not have a binary relation between worlds.

The first to use a binary relation were Prior and Meredith ([121]). Prior invented the modern temporal logic by replacing the possibility and necessity operators of the standard modal logic with tense operators ([140–143]). He introduced a binary relation in an explicitly modal context and employed an accessibility-like interpretation of the relation.

The first completeness proofs were obtained by Bayart, Hintikka and Kripke. Smiley has announced the completeness proofs for propositional M, $S4$ and $S5$ with respect to possible world semantics in [159, 160]. Bayart and Kripke proved the completeness of an extension of $S5$ with quantifiers in [14] and [108], respectively. Later, Kripke adapted his method to prove the completeness of propositional modal systems T, $S4$, $S5$ and B ([109]). Hintikka presented completeness proof for versions of M, $S4$ and $S5$ with quantifiers at seminars in Boston area, where he gave a series of informal talks. His modal completeness proofs were variants of the completeness proof for the first-order predicated calculus published in [79, 80]. A significant contribution to the development of the possible world semantics is given by Feys ([58]), McKinsey ([120]), Becker ([15]) and Montague ([125]). A historical survey of possible world semantics is given in [32].

The crucial ingredients in the possible world semantics are: the universal notion of validity, considering possible worlds as indices or points of evaluation

and accessibility relation between worlds. Kripke was the only one who included all these ingredients in the semantics, since he characterized the worlds as simple points of evaluation ([109]). This characterization enabled him to observe the link between the algebra of modal logic and the model theoretic semantics, and to obtain model theoretic completeness results for various modal systems in a systematic way. Kripke semantics was first introduced for modal logics, but later it was adapted to the intuitionistic logic ([110]) and other non-classical logics ([91]).

Due to the correspondence between the intuiotionistic logic and typed λ-calculus known as *Curry-Howard correspondence* ([83]), Mitchell and Moggi suggested to employ the semantics of the intuiotinistic logic as a semantics for typed λ-calculus and presented Kripke-style semantics for the simply typed λ-calculus in [124]. In [59], Gallier generalised the results from [124] to the second-order λ-calculus. Kripke semantics for various typed calculi were introduced in [5, 33, 85, 88, 116].

1.3 Probability logics

The story about probability logic, as many others mathematical stories, begins in ancient times. The plausible reasoning was invented by sophists Corax and Tisias in Plato's Phaedrus and Aristotle's Rhetoric ([107]), who used the notion of uncertainty in their argument about legal, medical or political questions. A number of mathematicians such as Cardano, Tartaglia, de Fermat, Pascal and Huygens were dealing with games of chance and introduced new ideas about uncertainty. The idea that probability can be numerically quantified and that the methods designed for games of chance can be used to calculate probability was suggested in [3], where the probability is consider as something epistemic and related to arguments and opinion. The word probability in the contemporary sense was first used by Leibnitz, who defined a probability as the ratio of favorable cases to the total number of cases ([113]). The work of Pascal, Huygens and Leibnitz had a great influence on the work of Jacob Bernoulli ([18]), who is considered the founder of the probability theory ([154]). De Morgan, Bayes and Boole also contributed significantly to the development of the probability theory. Boole has studied the relationship between logic and probability ([23]).

After the work of Leibnitz and Boole, the greatest progress in the probability logic was made by Keisler, Hoover, Hamblin and Hailperin. Keisler ([100, 101]) studied the probability distributions on domains of first-order structures and introduced probability quantifiers of the form $Px > r$, where the formula $(Px > r)\varphi(x)$ has the following meaning:

"the probability of the set $\{x \mid \varphi(x)\}$ is greater than r."

Hoover ([82]) gave a complete axiomatization of Keisler-like logics and together with Keisler proved completeness theorems for various kinds of models such as probability, graded, analytic, hyperfinite, etc. The connection between probability and modal logic was observed by Hamblin, who introduced probability, modal operator ([74]). Hailperin ([72]) has used the methods for linear programming to derive an effective procedure for obtaining the best possible boundaries for probabilities of propositional formulas, when the probabilities of subformulas are known. Reasoning with uncertainty found its applications in many fields such as artificial intelligence, computer science, economics and philosophy. The development in these fields triggered off the development of probability logic. The modern development of probability logic started with Nilsson ([127]), who has introduced a first framework for formalizing probabilistic reasoning.

The logic introduced in [127] has influenced the work of many researchers, who have developed frameworks for probabilistic reasoning ([56, 57, 73, 167]). This work is followed by a number of infinitary probability logics introduced in [104, 119, 129–131, 147]. The term infinitary concerns the meta language only, more precisely the language of the logic is countable and formulas are finite, but the proofs are allowed to be infinite. It turns out that this approach can be used for combining probability with different logics, e.g. classical logic, intuitionistic logic, temporal logic, justification logic and others. Systematic overview of some of these infinitary logics is given in [132].

1.4 Main contributions and the structure of the thesis

Probabilistic λ-calculus endowed with let-in operator is studied in Chapter 2. It is proved that, in the case that let-in operator is present in the language, probabilistic applicative bisimilarity is an effective method for proving the equivalence of probabilistic programs. More precisely, we prove that in the probabilistic λ-calculus with let-in operator, bisimilarity and context equivalence coincide. Chapter 2 is based on the paper:

[99] S. Kašterović and M. Pagani. The discriminating power of the let-in operator in the lazy call-by-name probabilistic lambda-calculus. In H. Geuvers, editor, 4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany, volume 131 of LIPIcs, pages 26:1–26:20. Schloss Dagstuhl-Leibniz-

Zentrum für Informatik, 2019. doi: 10.4230/LIPIcs.FSCD.2019.26.
URL: https://doi.org/10.4230/LIPIcs.FSCD.2019.26.

In Chapter 3, the full simply typed combinatory logic, which is simply typed combinatory logic extended with product types, sum types, the empty type and the unit type, is studied. We introduce Kripke-style semantics for the full simply typed combinatory logic and prove that the full simply typed combinatory logic is sound and complete with respect to the proposed semantics. Chapter 3 is based on the paper:

[94] S. Kašterović and S. Ghilezan. Kripke-style semantics and completeness for full simply typed lambda calculus. J. Log. Comput., 30(8):1567–1608, 2020. doi: 10.1093/logcom/exaa055.
URL: https://doi.org/10.1093/logcom/exaa055.

The semantics presented here are introduced in the mentioned paper, however the calculus in the paper differs from the one presented here. The paper studies full simply typed λ-calculus and combinatory logic that do not include typing rule for equal terms, i.e. the rule that guarantees that equal term inhabit the same type. In the calculus studied in the thesis, this rule is included. The motivation for adding this rule is explained in Section 3.2.

Chapter 4, introduces the logic of combinatory logic (LCL), which is a propositional extension of the simply typed combinatory logic. We present the language of LCL, its semantics and axiomatization, and prove that the given axiomatization is sound and complete with respect to the proposed semantics. The results of Chapter 4 are in the preparation for the publication and have been presented in

[97] S. Kašterović and S. Ghilezan. Logic of combinatory logic. CoRR, abs/2212.06675, 2022. doi: 10.48550/arXiv.2212.06675.
URL: https://doi.org/10.48550/arXiv.2212.06675.

The semantics of LCL is based on applicative structures extended with special elements corresponding to primitive combinators. Similar approach was used for the semantics of the combinatory logic with intersection types introduced in the paper:

[62] S. Ghilezan, S. Kašterović. Semantics for Combinatory Logic With Intersection Types, Frontiers in Computer Science, volume 4, 2022. doi: 10.3389/fcomp.2022.792570.
URL: https://www.frontiersin.org/articles/10.3389/fcomp.2022.792570

Chapter 5 introduces the probabilistic extension (PCL) of the logic of combinatory logic. Following approach used by Ognjanović, Rašković and

Marković in [132], we extend the logic of combinatory logic with probability operators and obtain a formal model for reasoning about simply typed combinatory terms. The idea of developing a framework for probabilistic reasoning about typed terms has been introduced in the papers:

[64] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Probabilistic reasoning about simply typed lambda terms. In S. N. Artemov and A. Nerode, editors, Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings, volume 10703 of Lecture Notes in Computer Science, pages 170–189. Springer, 2018. doi: 10.1007/978-3-319-72056-2_11. URL https://doi.org/10.1007/978-3-319-72056-2_11.

[66] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović and N. Savić. Towards probabilistic reasoning in type theory - the intersection type case. In A. Herzig and J. Kontinen, editors, Foundations of Information and Knowledge Systems-11th International Symposium, FoIKS 2020, Dortmund, Germany, February 17-21, 2020, Proceedings, volume 12012 of Lecture Notes in Computer Science, pages 122–139. Springer, 2020. doi: 10.1007/978-3-030-39951-1_8. URL https://doi.org/10.1007/978-3-030-39951-1_8.

The semantics for these systems were based on the well-known models for λ-calculus such as term models and filter models. The languages of the systems are defined as the union of two sets of formulas: basic formulas and probabilistic formulas. Basic formulas are formulas from the propositional extensions of the typed calculi and probabilistic formulas are obtained by applying probability operator to basic formulas. However, term models and filter models did not prove to be suitable for the propositional reasoning about typed terms. Thus, we develop the propositional extension of the simply typed combinatory logic in Chapter 4 and introduce its semantics. So, the semantics for PCL are based on the semantics for LCL. We present the logic PCL by introducing its syntax, semantics and axiomatization, and we prove that the given axiomatization is sound and complete with respect to the proposed semantics.

Chapter 2

Probabilistic computation

In this chapter, we study a probabilistic λ-calculus $\Lambda_{\oplus,\mathsf{let}}$ defined by endowing the pure, untyped λ-calculus with two operators: a probabilistic choice operator $\oplus$ and a let-in operator. The chapter presents the results of [98, 99].

There are different evaluation strategies that can be adopted to evaluate programs, such as call-by-name, call-by-value, call-by-need and others. In $\Lambda_{\oplus,\mathsf{let}}$, both call-by-name and call-by-value strategy are implemented. In call-by-name setting,

$$\text{a term } (\lambda x.M)N \text{ evaluates to a term } M\{N/x\} \text{ for any term } N,$$

where $M\{N/x\}$ denotes the capture-avoiding substitution of N for the free occurrences of x in M. On the other hand, in call-by-value setting,

$$\text{a term } (\lambda x.M)V \text{ can be evaluated to a term } M\{V/x\} \text{ only if } V \text{ is a value,}$$

that is, we need to first evaluate the term V before we pass it to the calling parameter x. In many functional programming languages, let-in operator is used to allow the local definition of an expression, which is used in another expression. We define the let-in operator to represent the substitution of a variable for a value. Although $\Lambda_{\oplus,\mathsf{let}}$ is a call-by-name probabilistic λ-calculus, the presence of the let-in operator gives us a possibility of evaluating terms in a call-by-value way, thus both strategies, call-by-name and call-by-value, are combined in $\Lambda_{\oplus,\mathsf{let}}$.

One of the most challenging problems in the probabilistic programming is to check if two programs enjoy the same behavioural properties. If two programs behave in the same manner in any possible context, we say that they are *context equivalent*. Proving that two programs are context equivalent in higher-order languages is not always easy. Hence, the aim is to find an

effective method for checking the context equivalence, that is, to find the characterization of context equivalence which enables to check the equality of programs more easily. The work presented in this chapter is inspired by results in [35] and [47], where the authors study the probabilistic λ-calculus, that is the pure, deterministic λ-calculus extended with a probabilistic choice operator. In [47], Dal Lago, Sangiorgi and Alberti considered call-by-name evaluation and it is shown that the probabilistic applicative bisimilarity is included in context equivalence and that they do not coincide. Terms which distinguish these two relations are

$$M = \lambda x.\lambda y.(x \oplus y) \text{ and } N = (\lambda x.\lambda y.x) \oplus (\lambda x.\lambda y.y),$$

which are context equivalent in call-by-name probabilistic λ-calculus, but are not bisimilar. On the other hand, in [35] Crubillé and Dal Lago proved that the probabilistic applicative bisimulation coincide with the context equivalence if call-by-value evaluation is considered. The question that we address here is if the mismatch between call-by-name and call-by-value calculus is the presence of let-in operator.

In this chapter, three different notions of equivalence are defined and compared. The first notion of equivalence we consider is *a context equivalence*, also called observable (behavioural) equivalence. Two terms M and N are context equivalent if we can replace all occurrences of M with N in any program, without changing the observable behaviour of a program. A notion of *probabilistic applicative bisimulation (bisimilarity)* is introduced using the fact that a labelled Markov chain can model the evaluation of programs in $\Lambda_{\oplus,\mathsf{let}}$. The third equivalence relation, called *testing equivalence*, is induced by a testing language defined on $\Lambda_{\oplus,\mathsf{let}}$.

Contributions of the chapter

- Using Howe's technique ([84]), we show that a probabilistic applicative bisimilarity is a congruence. The structure of the proof is similar to the ones in [35], [36] and [47], where Howe's technique is also used.

- The proof that the probabilistic applicative bisimilarity is included in the context equivalence follows from the previous result (probabilistic applicative bisimilarity being a congruence).

- We introduce the testing equivalence and prove that, for every test, there is a context such that the success probability of the test applied to a term and the convergence probability of the context applied to the same term are equal.

- As a consequence, the context equivalence is included in the testing equivalence.
- Furthermore, we prove that the context equivalence is also included in the probabilistic applicative bisimilarity, hence those two relations coincide.

Overview of the chapter We start with introducing $\Lambda_{\oplus,\mathsf{let}}$-calculus in Section 2.1. We define the syntax of $\Lambda_{\oplus,\mathsf{let}}$-calculus in Section 2.1.1, give its operational semantics in Section 2.1.2 and introduce a notion of context equivalence in Section 2.1.3. The probabilistic applicative simulation and bisimulation are introduced in Section 2.2, followed by the proof that bisimilarity is a congruence and that it is included in the context equivalence, which is given in Section 2.2.1. Section 2.3 introduces the testing equivalence and presents the proof of full abstraction. In Section 2.4, concluding remarks and future work are presented.

2.1 Probabilistic λ-calculus

In this section, we introduce formally the lazy call-by-name probabilistic λ-calculus $\Lambda_{\oplus,\mathsf{let}}$. We introduce its syntax, operational semantics and define the context equivalence.

2.1.1 Syntax

Probabilistic λ-calculus $\Lambda_{\oplus,\mathsf{let}}$ is a pure, untyped λ-calculus extended with two new operators: a probabilistic binary operator $\oplus$ and let-in operator. Terms and values are expressions generated by the following grammar:

$$\boxed{\begin{array}{lrl} \text{(values)} & V & ::= x \mid \lambda x.M \\ \text{(terms)} & M & ::= V \mid MM \mid M \oplus M \mid \mathsf{let}\ x = M\ \mathsf{in}\ M \end{array}} \tag{2.1}$$

where x belongs to a countable set of term variables, $X = \{x, y, z, \ldots, x_1, \ldots\}$. $\Lambda_{\oplus,\mathsf{let}}$ denotes the set of all terms and $\mathcal{V}_{\oplus,\mathsf{let}}$ denotes the set of all values. We let $M, N, \ldots$ range over $\Lambda_{\oplus,\mathsf{let}}$ and $V, W, \ldots$ range over $\mathcal{V}_{\oplus,\mathsf{let}}$. We use $\lambda x_1 x_2 \ldots x_n.M$ to abbreviate $\lambda x_1.\lambda x_2.\ldots.\lambda x_n.M$.

The lambda abstraction $\lambda x.M$ *binds* the *free* variable x in term M and the let-in operator $\mathsf{let}\ x = M\ \mathsf{in}\ N$ binds the free variable x in term N. Following Barendregt's Variable Convention ([10]), we assume that the bound variables that occur in a certain expression are different from the free ones. For a term M, the set of its free variables $FV(M)$ is defined as follows:

- $FV(x) \stackrel{\text{def}}{=} \{x\}$,
- $FV(\lambda x.M) \stackrel{\text{def}}{=} FV(M) \setminus \{x\}$,
- $FV(MN) \stackrel{\text{def}}{=} FV(M) \cup FV(N)$,
- $FV(M \oplus N) \stackrel{\text{def}}{=} FV(M) \cup FV(N)$,
- $FV(\mathsf{let}\ x = M\ \mathsf{in}\ N) \stackrel{\text{def}}{=} (FV(N) \setminus \{x\}) \cup FV(M)$.

A term M without free variables, $FV(M) = \emptyset$, is called *closed term (program)*. We write $\Lambda^{\Phi}_{\oplus,\mathsf{let}}$ (resp. $\mathcal{V}^{\Phi}_{\oplus,\mathsf{let}}$) to denote the set of terms (resp. values) whose free variables are within $\Phi = \{x_1, x_2, \ldots, x_n\}$. The capture-avoiding substitution of N for the free occurrences of x in M, denoted by $M\{N/x\}$, is defined inductively as follows:

- $x\{N/x\} \stackrel{\text{def}}{=} N$,
- $y\{N/x\} \stackrel{\text{def}}{=} y, if x \not\equiv y$,
- $(\lambda y.M)\{N/x\} \stackrel{\text{def}}{=} \lambda y.(M\{N/x\})$,
- $(M_1 M_2)\{N/x\} \stackrel{\text{def}}{=} (M_1\{N/x\})(M_2\{N/x\})$,
- $(M_1 \oplus M_2)\{N/x\} \stackrel{\text{def}}{=} (M_1\{N/x\}) \oplus (M_2\{N/x\})$,
- $(\mathsf{let}\ y = M_1\ \mathsf{in}\ M_2)\{N/x\} \stackrel{\text{def}}{=} \mathsf{let}\ y = (M_1\{N/x\})\ \mathsf{in}\ (M_2\{N/x\})$.

By Barendregt's Variable Convention, we have that the variables x and y in the third and fifth clause are not the same.

Example 2.1. *We define some terms that will be used in the sequel.*

- *Term* $\mathbf{I} \stackrel{\text{def}}{=} \lambda x.x$ *is called the identity.*
- *Terms* $\mathbf{T} \stackrel{\text{def}}{=} \lambda xy.x$ *and* $\mathbf{F} \stackrel{\text{def}}{=} \lambda xy.y$ *represent boolean projections.*
- *The duplicator is the term defined as* $\mathbf{\Delta} \stackrel{\text{def}}{=} \lambda x.xx$ *and it enables defining the ever looping term* $\mathbf{\Omega} \stackrel{\text{def}}{=} \mathbf{\Delta}\mathbf{\Delta}$.
- *The term* $\mathbf{\Delta}^{\ell} \stackrel{\text{def}}{=} \lambda x.\mathsf{let}\ y = x\ \mathsf{in}\ yy$ *is the call-by-value duplicator defined using* let-in *operator.*

2.1.2 Operational semantics

In this subsection, we introduce the operational semantics of $\Lambda_{\oplus,\mathsf{let}}$. The operational semantics of probabilistic λ-calculus, both call-by-name and call-by-value have been introduced in [45].

We first introduce the one-step reduction relation with the following reduction rules:

$$\begin{aligned}(\lambda x.M)N &\to M\{N/x\},\\ M \oplus N &\to M,\\ M \oplus N &\to N,\\ \mathsf{let}\ x = V\ \mathsf{in}\ M &\to M\{V/x\}.\end{aligned}$$

We denote by $\to^\star$ the reflexive and transitive closure of the relation $\to$. For terms M and N, if $M \to^\star N$, then we say that the term M reduces (evaluates) to the term N.

Due to the presence of the probabilistic operator $\oplus$ in the language, a closed term in $\Lambda_{\oplus,\mathsf{let}}$ does not evaluate to a single value. For example, the term $M \oplus N$ reduces to the term M with probability $\frac{1}{2}$ and to the term N with probability $\frac{1}{2}$. Thus, a closed term evaluates to a function which assigns a probability to values, that is, to a discrete probability distribution of the outcomes.

Definition 2.2 (Distribution)**.** *A (value) distribution is a map* $\mathscr{D} : \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}} \to \mathbb{R}_{[0,1]}$*, such that* $\sum_{V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}} \mathscr{D}(V) \leq 1$.

We denote with $\mathcal{P}_{\mathcal{D}}$ the set of all value distributions $\mathscr{D}$. The support of $\mathscr{D}$, denoted by $\mathsf{S}(\mathscr{D})$, is the subset of $\mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ whose elements are values to which $\mathscr{D}$ assigns a positive probability. In the sequel, we use $\sum \mathscr{D}$ to abbreviate $\sum_{V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}} \mathscr{D}(V)$. If a distribution $\mathscr{D}$ has a finite support $\{V_1, \ldots, V_n\}$ and for every $i \in \{1, \ldots, n\}$, $\mathscr{D}(V_i) = p_i$, we denote the distribution $\mathscr{D}$ with $p_1V_1 + \ldots + p_nV_n$. In this case, we have $\sum \mathscr{D} = \sum_{k=1}^{n} p_k$. The empty distribution is denoted by 0, and V denotes both the value V and the distribution which assigns probability 1 to the value V.

Definition 2.3. *The order over distributions is defined pointwise:* $\mathscr{D} \leq \mathscr{E}$ *if and only if* $\mathscr{D}(V) \leq \mathscr{E}(V)$ *for every value* $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ *.*

The structure $(\mathcal{P}_{\mathcal{D}}, \leq)$ is a partially ordered set, that is $\leq$ is reflexive and transitive relation defined on the set $\mathcal{P}_{\mathcal{D}}$. Moreover, it is an ω-complete partial

order: every ω-chain (countable chain) in $\mathcal{P}_{\mathcal{D}}$ has a supremum. Following [45], we give *the operational semantics* of $\Lambda_{\oplus,\mathsf{let}}$ in two steps. First, we define a big-step approximation relation and then we define the semantics.

Definition 2.4. *A big-step approximation relation $M \Downarrow \mathscr{D}$ is a relation between closed terms and finite value distributions and it is inductively defined by the derivation rules in Figure 2.1.*

$$\frac{}{M \Downarrow 0}\ (\text{be}) \qquad \frac{}{V \Downarrow V}\ (\text{bv}) \qquad \frac{M \Downarrow \mathscr{D} \quad N \Downarrow \mathscr{E}}{M \oplus N \Downarrow \frac{1}{2}\mathscr{D} + \frac{1}{2}\mathscr{E}}\ (\text{bs})$$

$$\frac{M \Downarrow \mathscr{D} \quad \{P\{N/x\} \Downarrow \mathscr{E}_{P,N}\}_{\lambda x.P \in \mathsf{S}(\mathscr{D})}}{MN \Downarrow \sum_{\lambda x.P \in \mathsf{S}(\mathscr{D})} \mathscr{D}(\lambda x.P) \cdot \mathscr{E}_{P,N}}\ (\text{ba})$$

$$\frac{N \Downarrow \mathscr{G} \quad \{M\{V/x\} \Downarrow \mathscr{H}_V\}_{V \in \mathsf{S}(\mathscr{G})}}{\mathsf{let}\ x = N\ \mathsf{in}\ M \Downarrow \sum_{V \in \mathsf{S}(\mathscr{G})} \mathscr{G}(V) \cdot \mathscr{H}_V}\ (\text{bl})$$

FIGURE 2.1: Rules for the approximation relation $M \Downarrow \mathscr{D}$, with $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $\mathscr{D}$ being a value distribution.

The rule (be) in Figure 2.1 ensures that every term evaluates to the empty distribution and the rule (bv) guarantees that every value V evaluates to the distribution which assigns probability 1 to the value V. Further, the rule (bs) gives semantics to a binary choice. Let us consider terms $\mathbf{\Omega}$ and $\mathbf{I}$. Although $\mathbf{\Omega}$ only evaluates to the empty distribution $\mathbf{\Omega} \Downarrow 0$, the sum $\mathbf{\Omega} \oplus \mathbf{I}$ evaluates to a non-empty distribution. From $\mathbf{I} \Downarrow \lambda x.x$, we derive $\mathbf{\Omega} \oplus \mathbf{I} \Downarrow \frac{1}{2}\lambda x.x$. The rule (ba) gives semantics to an application MN. This rule reflects the call-by-name evaluation, since it is enough to evaluate the term M to distribution $\mathscr{D}$ and for every $\lambda x.P \in \mathsf{S}(\mathscr{D})$ obtain distribution $\mathscr{E}_{P,N}$ by evaluating $P\{N/x\}$. On the other hand, in the call-by-value setting, it is not enough to evaluate the term M to distribution $\mathscr{D}$, the term N also has to be evaluated to some distribution $\mathscr{E}$, and for every $\lambda x.P \in \mathsf{S}(\mathscr{D})$ and $V \in \mathsf{S}(\mathscr{E})$, the distribution $\mathscr{E}_{P,V}$ is obtained by evaluating $P\{V/x\}$. The call-by-value passing policy is implemented by the rule (bl). In order to evaluate the term $\mathsf{let}\ x = N\ \mathsf{in}\ M$, we need to evaluate the term N before passing it to the term M. In addition, the rule (ba) implements lazy call-by-name evaluation, where *lazy* means that term does not reduce within the body of an abstraction

Proposition 2.5. *Let N be a term. If $N \Downarrow \mathscr{E}$ and $N \Downarrow \mathscr{F}$, then there exists a distribution $\mathscr{D}$ such that $N \Downarrow \mathscr{D}$, $\mathscr{E} \leq \mathscr{D}$ and $\mathscr{F} \leq \mathscr{D}$.*

Proof. The proof is by induction on the derivation of $N \Downarrow \mathscr{E}$. □

Proposition 2.5 proves that the set $\{\mathscr{D} \mid M \Downarrow \mathscr{D}\}$ is a directed set in the sense of Definition 2.6 below.

Definition 2.6 ([148])**.** *A nonempty subset X of a partially ordered set P is directed if every pair $\{a, b\}$ of elements of X has an upper bound in X, that is, if for every $a, b \in X$, there exists $c \in D$ with the property that $a \leq c$ and $b \leq c$.*

Since the set $\{\mathscr{D} \mid M \Downarrow \mathscr{D}\}$ is a countable, directed set (Proposition 2.5) and $(\mathcal{P}_{\mathcal{D}}, \leq)$ is an ω-complete partial order, the least upper bound of the set $\{\mathscr{D} \mid M \Downarrow \mathscr{D}\}$ is well-defined.

Definition 2.7. *The semantics of M, denoted by $[\![M]\!]$, is the least upper bound of all distributions which are related to M via the big-step approximation relation, that is*

$$[\![M]\!] = \sup\{\mathscr{D} \mid M \Downarrow \mathscr{D}\}. \tag{2.2}$$

If $\sum[\![M]\!] = p$, then we say that M converges with probability p.

As we have already discussed, we consider the lazy call-by-name λ-calculus. The lazy call-by-name strategy is implemented by the rules in Figure 2.1. In the rule (ba) of Figure 2.1, an argument is passed to a function without evaluating it. However, the call-by-value strategy is also present in the calculus. If we want to evaluate term $\mathsf{let}\ x = Q\ \mathsf{in}\ P$, we first need to evaluate the term Q to a value V and then we can evaluate the given term to $P\{V/x\}$. Thus, the let-in operator implements the call-by-value strategy.

The following examples illustrate the operational semantics.

Example 2.8. *For term $\mathbf{\Delta}(\mathbf{F} \oplus \mathbf{T})$, we can derive $\mathbf{\Delta}(\mathbf{F} \oplus \mathbf{T}) \Downarrow \mathscr{D}$ for any*

$$\mathscr{D} \in \mathcal{P}_1 = \{0, \frac{1}{2}\mathbf{I}, \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T}), \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T}) + \frac{1}{2}\mathbf{I}\},$$

by the rules of Figure 2.1. The derivation of

$$\mathbf{\Delta}(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T})$$

is given in Figure 2.2. The least upper bound of the set $\mathcal{P}_1$ is the distribution $\frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T})$, thus this distribution is the semantics of $\mathbf{\Delta}(\mathbf{F} \oplus \mathbf{T})$,

$$[\![\mathbf{\Delta}(\mathbf{F} \oplus \mathbf{T})]\!] = \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T}).$$

$$\dfrac{\Delta \Downarrow \Delta \qquad \dfrac{\dfrac{\mathbf{F} \Downarrow \mathbf{F} \qquad \mathbf{T} \Downarrow \mathbf{T}}{\mathbf{F} \oplus \mathbf{T} \Downarrow \frac{1}{2}\mathbf{F} + \frac{1}{2}\mathbf{T}}\,(\mathrm{bs}) \qquad \{\mathbf{I} \Downarrow \mathbf{I},\ \lambda y.(\mathbf{F} \oplus \mathbf{T}) \Downarrow \lambda y.(\mathbf{F} \oplus \mathbf{T})\}}{(\mathbf{F} \oplus \mathbf{T})(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T})}\,(\mathrm{ba})}{\Delta(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T})}\,(\mathrm{ba})$$

FIGURE 2.2: A derivation of the big-step approximation $\Delta(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.(\mathbf{F} \oplus \mathbf{T})$

Example 2.9. *If, in the Example 2.8, the operator* Δ *is replaced with the operator* Δ^ℓ *introduced in Example 2.1, we obtain the term* $\Delta^\ell(\mathbf{F} \oplus \mathbf{T})$*. For any*

$$\mathscr{D} \in \mathcal{P}_2 = \{0, \frac{1}{2}\mathbf{I}, \frac{1}{2}\lambda y.\mathbf{T}, \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}\},$$

we can derive $\Delta^\ell(\mathbf{F} \oplus \mathbf{T}) \Downarrow \mathscr{D}$ *by the rules given in Figure 2.1. Again, we give the derivation for just one distribution (Figure 2.3). The distribution* $\frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}$ *is the least upper bound of the set* $\mathcal{P}_2$*, hence*

$$[\![\Delta^\ell(\mathbf{F} \oplus \mathbf{T})]\!] = \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}.$$

$$\dfrac{\Delta^\ell \Downarrow \Delta^\ell \qquad \dfrac{\dfrac{\mathbf{F} \Downarrow \mathbf{F} \qquad \mathbf{T} \Downarrow \mathbf{T}}{\mathbf{F} \oplus \mathbf{T} \Downarrow \frac{1}{2}\mathbf{F} + \frac{1}{2}\mathbf{T}}\,(\mathrm{bs}) \qquad \{\mathbf{FF} \Downarrow \mathbf{I},\ \mathbf{TT} \Downarrow \lambda y.\mathbf{T}\}}{\mathsf{let}\ y = \mathbf{F} \oplus \mathbf{T}\ \mathsf{in}\ yy \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}}\,(\mathrm{bl})}{\Delta^\ell(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}}\,(\mathrm{ba})$$

FIGURE 2.3: A derivation of the big-step approximation $\Delta^\ell(\mathbf{F} \oplus \mathbf{T}) \Downarrow \frac{1}{2}\mathbf{I} + \frac{1}{2}\lambda y.\mathbf{T}$

Example 2.10. *In the probabilistic* λ*-calculus, normalizing terms are terms* M *such that their semantics are of total mass* $\sum[\![M]\!] = 1$ *and that there exists a unique finite derivation giving* $M \Downarrow [\![M]\!]$*. The probabilistic* λ*-calculus is a framework which also allows for almost sure terminating terms, i.e. terms* M *with* $\sum[\![M]\!] = 1$*, but such that there is no finite derivation giving* $M \Downarrow [\![M]\!]$*. An example of almost sure terminating term is the term* $M \stackrel{\text{def}}{=} VV$*, with* $V \stackrel{\text{def}}{=} \lambda x.(\mathbf{I} \oplus xx)$*. As Figure 2.4 shows, any finite approximation of* M *gives a distribution bounded by* $\sum_{i=1}^{n} \frac{1}{2^i}\mathbf{I}$ *for some* $n \geq 0$*. However, only the limit sum* $\sup_n \sum_{i=1}^{n} \frac{1}{2^i}\mathbf{I}$ *is equal to* $[\![M]\!] = \mathbf{I}$*.*

$$\dfrac{V \Downarrow V \qquad \dfrac{\mathbf{I} \Downarrow \mathbf{I} \qquad \dfrac{VV \Downarrow \sum_{i=1}^{n-1} \frac{1}{2^i}\mathbf{I}}{\vdots}\,(\text{ba})}{\mathbf{I} \oplus VV \Downarrow \sum_{i=1}^{n} \frac{1}{2^i}\mathbf{I}}\,(\text{bs})}{VV \Downarrow \sum_{i=1}^{n} \frac{1}{2^i}\mathbf{I}}\,(\text{ba}) \qquad \dfrac{\dfrac{\mathbf{I} \Downarrow \mathbf{I} \qquad VV \Downarrow 0}{\mathbf{I} \oplus VV \Downarrow \frac{1}{2}\mathbf{I}}\,(\text{bs})}{\vdots}\,(\text{ba})$$

FIGURE 2.4: A derivation of the big-step approximation $VV \Downarrow \sum_{i=1}^{n} \frac{1}{2^i}\mathbf{I}$ for $V = \lambda x.(\mathbf{I} \oplus xx)$.

Some fundamental properties of the semantics are stated in the following proposition, for which the respective proofs can be found in [45].

Proposition 2.11 ([45]). *For any terms M and N,*

1. $[\![(\lambda x.M)N]\!] = [\![M\{N/x\}]\!]$.
2. $[\![M \oplus N]\!] = \frac{1}{2}[\![M]\!] + \frac{1}{2}[\![N]\!]$.

2.1.3 Context equivalence

One of the most challenging problems in probabilistic programming is checking whether two programs M and N behave the same, in the sense that we can replace any occurrence of M within some program L with N without changing the behaviour of program L. In order to formalize this idea, a notion of context is introduced.

Definition 2.12. A $\Lambda_{\oplus,\mathsf{let}}$-term context *is a term with a unique hole* $[\cdot]$ *generated by the following grammar:*

$$\begin{array}{l} C ::= [\cdot] \mid \lambda x.C \mid CM \mid MC \mid C \oplus M \mid M \oplus C \mid \mathsf{let}\ x = C\ \mathsf{in}\ M \\ \qquad\quad \mid \mathsf{let}\ x = M\ \mathsf{in}\ C \end{array} \tag{2.3}$$

A $\Lambda_{\oplus,\mathsf{let}}$-term obtained from a context by replacing a hole in C with a $\Lambda_{\oplus,\mathsf{let}}$-term N, denoted by $C[N]$, is defined inductively as follows.

- $[\cdot][N] \stackrel{def}{=} N$,
- $(\lambda x.C)[N] \stackrel{def}{=} \lambda x.C[N]$,
- $(C \oplus M)[N] \stackrel{def}{=} C[N] \oplus M$,

- $(M \oplus C)[N] \stackrel{def}{=} M \oplus C[N]$,
- $(CM)[N] \stackrel{def}{=} C[N]M$,
- $(MC)[N] \stackrel{def}{=} MC[N]$,
- $(\text{let } x = M \text{ in } C)[N] \stackrel{def}{=} (\text{let } x = M \text{ in } C[N])$,
- $(\text{let } x = C \text{ in } M)[N] \stackrel{def}{=} (\text{let } x = C[N] \text{ in } M)$.

Substituting a hole in the context C with a term N allows capturing free variables of N; still, some free variables of N can remain free in $C[N]$. In the sequel, we will work with contexts C which bound all free variables of the term N, resulting in closed terms $C[N]$, called closing contexts. Therefore, the important features of a context are the set of variables it bounds and the set of variables it keeps free.

In order to keep the track of free variables, we introduce the set of contexts $\mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}$ inductively defined by the following rules.

$$\frac{}{[\cdot] \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Phi)}} \ (\mathsf{Ctx1})$$

$$\frac{C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi\cup\{x\})} \quad x \notin \Psi}{\lambda x.C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx2})$$

$$\frac{C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)} \quad M \in \Lambda^{\Psi}_{\oplus,\mathsf{let}}}{CM \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx3})$$

$$\frac{M \in \Lambda^{\Psi}_{\oplus,\mathsf{let}} \quad C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}}{MC \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx4})$$

$$\frac{C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)} \quad M \in \Lambda^{\Psi}_{\oplus,\mathsf{let}}}{C \oplus M \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx5})$$

$$\frac{M \in \Lambda^{\Psi}_{\oplus,\mathsf{let}} \quad C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}}{M \oplus C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx6})$$

$$\frac{C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)} \quad M \in \Lambda^{\Psi\cup\{x\}}_{\oplus,\mathsf{let}}}{(\text{let } x = C \text{ in } M) \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}} \ (\mathsf{Ctx7})$$

$$\frac{M \in \Lambda^{\Psi}_{\oplus,\mathsf{let}} \quad C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi\cup\{x\})}}{(\mathsf{let}\ x = M\ \mathsf{in}\ C) \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}}\ (\mathsf{Ctx8})$$

We use the sets Φ and Ψ of variables to indicate the sets of free variables before and after the filling of the hole by a term. The idea is explained by the following two lemmas.

Lemma 2.13. *If $M \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$ and $C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}$, then $C[M] \in \Lambda^{\Psi}_{\oplus,\mathsf{let}}$.*

Lemma 2.14. *If $C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}$ and $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Psi;\Theta)}$, then $D[C] \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Theta)}$.*

Lemma 2.13 and Lemma 2.14 are direct consequences of the definition of $\mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Psi)}$, the proofs follow by induction on the derivation of context and are omitted.

In the probabilistic language, we check the program equivalence by observing the probability of program convergence to a value. We say that two programs are context equivalent if they converge to a value with the same probability in all contexts. A context preorder, denoted by $\leq$, and a context equivalence, denoted by $\simeq$, are defined as follows.

Definition 2.15. *For all $M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, we define:*

$$M \leq N \text{ iff } \left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right) \sum [\![C[M]]\!] \leq \sum [\![C[N]]\!], \tag{2.4}$$

$$M \simeq N \text{ iff } \left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right) \sum [\![C[M]]\!] = \sum [\![C[N]]\!]. \tag{2.5}$$

We may observe that $M \simeq N$ is equivalent to $M \leq N$ and $N \leq M$. The context equivalence is illustrated by the following examples.

Example 2.16. *Two terms that show the difference between the call-by-name probabilistic λ-calculus without the* let-in *operator and the call-by-name probabilistic λ-calculus with the* let-in *operator are terms $M \stackrel{\text{def}}{=} \lambda xy.(x \oplus y)$ and $N \stackrel{\text{def}}{=} (\lambda xy.x) \oplus (\lambda xy.y)$. In the call-by-name probabilistic λ-calculus without the* let-in *operator ([47]) these terms are context equivalent; however in the call-by-name probabilistic λ-calculus with the* let-in *operator they can be discriminated by the context $C \stackrel{\text{def}}{=} (\mathsf{let}\ y = [\cdot]\ \mathsf{in}\ (\mathsf{let}\ z_1 = y\mathbf{I}\mathbf{\Omega}\ \mathsf{in}\ (\mathsf{let}\ z_2 = y\mathbf{I}\mathbf{\Omega}\ \mathsf{in}\ \mathbf{I})))$. Following the rules of Figure 2.1, we obtain $\sum [\![C[M]]\!] = \frac{1}{4}$ and $\sum [\![C[N]]\!] = \frac{1}{2}$. Since M and N have different probability of convergence in the same context they are not context equivalent.*

Example 2.17. *The two duplicators $\mathbf{\Delta}$ and $\mathbf{\Delta}^{\ell}$ introduced in (Example 2.1) are not context equivalent. For example, for the context $C \stackrel{\text{def}}{=} [\cdot](\mathbf{I} \oplus \mathbf{\Omega})$ we get $\sum [\![C[\mathbf{\Delta}]]\!] = \frac{1}{4}$ and $\sum [\![C[\mathbf{\Delta}^{\ell}]]\!] = \frac{1}{2}$. Thus, the context C discriminates the two duplicators.*

Proposition 2.18. *For $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, we have that, if $[\![M]\!] \leq [\![N]\!]$ then $M \leq N$. Therefore, $[\![M]\!] = [\![N]\!]$ implies $M \simeq N$.*

Proof. Following the definition of the context preorder, we can see that $[\![M]\!] \leq [\![N]\!]$ is equivalent to: for all $\mathscr{D}$ such that $M \Downarrow \mathscr{D}$, there exists $\mathscr{E} \geq \mathscr{D}$ such that $N \Downarrow \mathscr{E}$. The proof that $[\![M]\!] \leq [\![N]\!]$ implies $[\![C(M)]\!] \leq [\![C(N)]\!]$ follows by induction on the structure of the context C. □

Example 2.19. *Proposition 2.18 allows us to prove that two terms, which seem to be quite different, are actually context equivalent. For example, let us consider the term VV introduced in Example 2.10 and the term $\mathbf{I}$. They have the same semantics and as a consequence we can conclude that they are context equivalent. The opposite direction does not hold. There are context equivalent terms that do not have the same semantics, as for example terms $\lambda x.(x \oplus x)$ and $\mathbf{I}$.*

2.2 Probabilistic applicative bisimulation

The notion of the applicative bisimulation for the lazy call-by-name λ-calculus was introduced by Abramsky in [1]. Later, Larsen and Scou ([111]) have introduced a notion of probabilistic bisimulation for labelled Markov chains. Mixing these two notions resulted in emerging a new notion of bisimulation, called probabilistic applicative bisimulation ([47]), which is a relation between terms of probabilistic λ-calculus. In order to define probabilistic applicative bisimulation, we first notice that the operational semantics of probabilistic λ-calculus can be seen as a probabilistic transition system, or more precisely as a labelled Markov chain. The states in this system will be closed terms and the set of transitions will comprise two kinds of transitions:

- evaluating a term to a value, and
- applying a value to a term.

Looking at the probabilistic λ-calculus as labelled Markov chain allows us to define the notion of bisimilarity over it. As one of the main results, we will show that probabilistic applicative bisimilarity implies context equivalence, meaning that in order to prove that two program are context equivalent, it is enough to prove that they are bisimilar.

First, we introduce the notions of labelled Markov chain, probabilistic simulation and bisimulation (as in [111]).

Definition 2.20. *A* labelled Markov chain *is a triple $\mathcal{M} = (\mathcal{S}, \mathcal{L}, P)$ where*

- *$\mathcal{S}$ is a countable set of states,*

- $\mathcal{L}$ *is a set of labels (actions), and*
- P *is a transition probability matrix, i.e. a function* $P : \mathcal{S} \times \mathcal{L} \times \mathcal{S} \to \mathbb{R}_{[0,1]}$ *satisfying the following condition:* $(\forall s \in \mathcal{S})\,(\forall l \in \mathcal{L}) \sum_{t \in \mathcal{S}} P(s,l,t) \leq 1$.

Throughout the following subsections, we use the following notational conventions. We denote $\sum_{t \in X} P(s,l,t)$ by $P(s,l,X)$. For a relation $\mathcal{R}$, the image of the set X under $\mathcal{R}$ is denoted by $\mathcal{R}(X)$, i.e. $\mathcal{R}(X) = \{y \mid \exists x \in X \text{ such that } x\mathcal{R}y\}$. If $\mathcal{R}$ is a binary relation, then $\mathcal{R}^{op}$ denotes the relation $\{(b,a) \mid (a,b) \in \mathcal{R}\}$. For an equivalence relation $\mathcal{R}$, the set of all equivalence classes of $\mathcal{S}$ modulo $\mathcal{R}$ will be denoted by $\mathcal{S}/\mathcal{R}$.

Definition 2.21. *Let* $(\mathcal{S}, \mathcal{L}, P)$ *be a labelled Markov chain and* $\mathcal{R}$ *be a relation over* $\mathcal{S}$*:*

- $\mathcal{R}$ *is a* probabilistic simulation *if it is a preorder and*

$$(\forall (s,t) \in \mathcal{R})\,(\forall X \subseteq \mathcal{S})\,(\forall l \in \mathcal{L})\, P(s,l,X) \leq P(t,l,\mathcal{R}(X))$$

- $\mathcal{R}$ *is a* probabilistic bisimulation *if it is an equivalence and*

$$(\forall (s,t) \in \mathcal{R})\,(\forall E \in \mathcal{S}/\mathcal{R})\,(\forall l \in \mathcal{L})\, P(s,l,E) = P(t,l,E)$$

It has been proved in [47] that the union of all probabilistic simulations (resp. bisimulations) is still a simulation (resp. a bisimulation).

Definition 2.22. *The union of all probabilistic simulations is the largest probabilistic simulation, called* probabilistic similarity *and denoted by* $\lesssim$*. Similarly, the union of all probabilistic bisimulations is the largest probabilistic bisimulation, called* probabilistic bisimilarity *and denoted by* $\sim$*.*

$$M \lesssim N \text{ iff there exists a probabilistic simulation } \mathcal{R} \text{ such that } M\ \mathcal{R}\ N \tag{2.6}$$

$$M \sim N \text{ iff there exists a probabilistic bisimulation } \mathcal{R} \text{ such that } M\ \mathcal{R}\ N \tag{2.7}$$

It is straightforward to prove that $\sim = \lesssim \cap \lesssim^{op}$, i.e. $M \lesssim N$ and $N \lesssim M$ is equivalent to $M \sim N$ (Proposition 2.13, [47]).

We present the operational semantics of $\Lambda_{\oplus,\mathsf{let}}$ as a labelled Markov chain. For this purpose we define a set of distinguished values, denoted by $\mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ such that for every closed value $V = \lambda x.P \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ there is a distinguished value $\widetilde{V} = \nu x.P$ that belongs to $\mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. For example, value $\lambda yz.z$ belongs to the set $\mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$, whereas the distinguished value $\nu yz.z$ belongs to the set $\mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$.

Definition 2.23 ([47, 99]). *The $\Lambda_{\oplus,\mathsf{let}}$-Markov chain is defined as the triple $(\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}, \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \cup \{\tau\}, P)$ such that*

- *the set of states $\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ is the disjoint union of the set of closed terms and the set of closed distinguished values,*
- *labels (actions), $\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \cup \{\tau\}$, are either closed terms, which model parameter passing, or the action τ, which is a distinguished action that models evaluation,*
- *the transition probability matrix P is defined as follows:*
  - *for every closed term M and a distinguished value $\nu x.N$,*

$$P(M, \tau, \nu x.N) = [\![M]\!](\lambda x.N)\,,$$

  - *for every closed term M and a distinguished value $\nu x.N$,*

$$P(\nu x.N, M, N\{M/x\}) = 1\,,$$

  - *in all other cases, P returns 0.*

A probabilistic applicative simulation (resp. bisimulation) is a probabilistic simulation (resp. bisimulation) on $\Lambda_{\oplus,\mathsf{let}}$-Markov chain. Further, by applying Definition 2.22, we define *the probabilistic applicative similarity* $\lesssim$ and *the probabilistic applicative bisimilarity* $\sim$, respectively.

We extend the notions of probabilistic applicative similarity and bisimilarity to open terms by considering all closing substitutions. Let M and N be terms whose free variables belong to the set $\Phi = \{y_1, \ldots, y_m\}$, i.e. $M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$. If, for all terms $P_1, \ldots, P_n \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$,

$$M\{P_1/y_1, \ldots, P_n/y_n\} \lesssim N\{P_1/y_1, \ldots, P_n/y_n\},$$

we say that the terms M and N are similar, i.e. $M \lesssim N$. The notion of bisimilarity is analogously extended to open terms.

Example 2.24. *In Example 2.19, we have observed that there are terms that are context equivalent and that do not have the same semantics, as for example the terms $\lambda x.(x \oplus x)$ and $\lambda x.x$. Proving that these two terms are context equivalent is not an easy task, since we should check that the terms will converge to a value with the same probability in every context. In general, proofs of statements that use universal quantification in its definition can be hard to deal with, which is why the characterizations that use existential quantification are preferable. In this example, we will prove that the terms $\lambda x.(x \oplus x)$ and*

$\lambda x.x$ are bisimilar. We need to find a bisimulation that contains the pair of terms $(\lambda x.(x \oplus x), \lambda x.x)$. Let us consider the relation

$$\begin{aligned}\mathcal{R} =&\{(\lambda x.(x \oplus x), \lambda x.x)\} \cup \{(\lambda x.x, \lambda x.(x \oplus x))\} \cup \{(\nu x.(x \oplus x), \nu x.x)\} \\ &\cup \{(\nu x.x, \nu x.(x \oplus x))\} \cup \{(M \oplus M, M) \mid M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}\} \\ &\cup \{(M, M \oplus M) \mid M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}\} \cup \{(M, M) \mid M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}\} \\ &\cup \{(\widetilde{W}, \widetilde{W}) \mid \widetilde{W} \in \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}\}.\end{aligned}$$

The pair of terms $(\lambda x.(x \oplus x), \lambda x.x)$ belongs to the relation $\mathcal{R}$, so we need to prove that $\mathcal{R}$ is a bisimulation. By its definition, $\mathcal{R}$ is an equivalence relation, thus it remains to prove that is satisfies the condition $(\forall (M, N) \in \mathcal{R})$ $\Big(\forall E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}\Big)\Big(\forall \ell \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \cup \{\tau\}\Big)$, $P(M, \ell, E) = P(N, \ell, E)$ (Definition 2.21). We illustrate just some cases of the proof, since other cases are analogous.

- *First, let us consider the pair $(\lambda x.(x \oplus x), \lambda x.x) \in \mathcal{R}$. From Definition 2.21, it follows that $P(\lambda x.(x \oplus x), L, E) = 0 = P(\lambda x.x, L, E)$ holds for all $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}$. Furthermore, we have that $(\nu x.(x \oplus x), \nu x.x) \in \mathcal{R}$, thus $\nu x.(x \oplus x)$ and $\nu x.x$ belong to the same equivalence class E. In the case that $\nu x.(x \oplus x), \nu x.x \in E$, it holds that $P(\lambda x.(x \oplus x), \tau, E) = 1 = P(\lambda x.x, \tau, E)$, and if $\nu x.(x \oplus x), \nu x.x \notin E$, then $P(\lambda x.(x \oplus x), \tau, E) = 0 = P(\lambda x.x, \tau, E)$. Consequently $P(\lambda x.(x \oplus x), \ell, E) = P(\lambda x.x, \ell, E)$ for all $\ell \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \cup \{\tau\}$ and all $E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}$.*

- *Next, by Definition 2.23 the equality $P(\nu x.(x \oplus x), \tau, E) = 0 = P(\nu x.x, \tau, E)$ holds for any equivalence class E. For $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, we have that $L \oplus L$ and L belong to the same equivalence class, since $(L \oplus L, L) \in \mathcal{R}$. Thus, $P(\nu x.(x \oplus x), L, E) = 1 = P(\nu x.x, L, E)$ if $L \in E$, and $P(\nu x.(x \oplus x), L, E) = 0 = P(\nu x.x, L, E)$, otherwise. Again, we conclude that $P(\nu x.(x \oplus x), \ell, E) = P(\nu x.x, \ell, E)$ for all $\ell \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \cup \{\tau\}$ and all $E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}$.*

- *Finally, let us consider the pair $(M \oplus M, M) \in \mathcal{R}$, where $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. By Definition 2.23, it follows that $P(M \oplus M, L, E) = 0 = P(M, L, E)$ for any closed term $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and equivalence class $E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}$. By Proposition 2.11 and Definition 2.23, for an equivalence class $E \in (\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}})/\mathcal{R}$ the following holds:*

$$\begin{aligned}
P(M \oplus M, \tau, E) &= \sum_{\nu x.N \in E} P(M \oplus M, \tau, \nu x.N) \\
&= \sum_{\{\lambda x.N \mid \nu x.N \in E\}} [\![M \oplus M]\!](\lambda x.N) \\
&= \sum_{\{\lambda x.N \mid \nu x.N \in E\}} (\frac{1}{2}[\![M]\!] + \frac{1}{2}[\![M]\!])(\lambda x.N) \\
&= \sum_{\{\lambda x.N \mid \nu x.N \in E\}} [\![M]\!](\lambda x.N) \\
&= \sum_{\nu x.N \in E} P(M, \tau, \nu x.N) \\
&= P(M, \tau, E).
\end{aligned}$$

This concludes the proof that the relation $\mathcal{R}$ is a probabilistic bisimulation, thus terms $\lambda x.x \oplus x$ and $\lambda x.x$ are bisimilar. As we will see, if we want to show that two terms are context equivalent, it is enough to show that they are bisimilar.

Example 2.25. *Let us consider the terms given in Example 2.16, $\lambda xy.(x \oplus y)$ and $(\lambda xy.x) \oplus (\lambda xy.y)$. We discussed in Example 2.16 that these terms are context equivalent in the call-by-name probabilistic λ-calculus without the* let-in *operator and can be discriminated by a context in $\Lambda_{\oplus,\mathsf{let}}$. Now, we will show that these terms are not bisimilar, i.e. that there is no bisimulation which contains both terms. Suppose the opposite, that there is a bisimulation $\mathcal{R}$ such that $(\lambda xy.(x \oplus y), (\lambda xy.x) \oplus (\lambda xy.y)) \in \mathcal{R}$. Then*

$$1 = P(\lambda xy.(x \oplus y), \tau, E) = P((\lambda xy.x) \oplus (\lambda xy.y), \tau, E)$$

holds for an equivalence class E of $\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \uplus \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ with respect to $\mathcal{R}$ such that $\nu x.\lambda y.(x \oplus y)$ belongs to E. Since $P((\lambda xy.x) \oplus (\lambda xy.y), \tau, \nu x.\lambda y.x) = \frac{1}{2}$ and $P((\lambda xy.x) \oplus (\lambda xy.y), \tau, \nu x.\lambda y.y) = \frac{1}{2}$, both $\nu x.\lambda y.x$ and $\nu x.\lambda y.y$ belong to E. From $\nu x.\lambda y.x \in E$, it follows that $(\nu x.\lambda y.(x \oplus y), \nu x.\lambda y.x) \in \mathcal{R}$. Let F be an equivalence class such that $\lambda y.(\mathbf{\Omega} \oplus y) \in F$. Then

$$1 = P(\nu x.\lambda y.(x \oplus y), \mathbf{\Omega}, F) = P(\nu x.\lambda y.x, \mathbf{\Omega}, F)$$

From $P(\nu x.\lambda y.x, \mathbf{\Omega}, \lambda y.\mathbf{\Omega}) = 1$, it follows that $\lambda y.\mathbf{\Omega}$ belongs to F. Therefore, $\lambda y.(\mathbf{\Omega} \oplus y), \lambda y.\mathbf{\Omega} \in F$ and consequently $(\lambda y.(\mathbf{\Omega} \oplus y), \lambda y.\mathbf{\Omega}) \in \mathcal{R}$. Let G be an equivalence class which contains $\nu y.(\mathbf{\Omega} \oplus y)$. Then

$$1 = P(\lambda y.(\mathbf{\Omega} \oplus y), \tau, G) = P(\lambda y.\mathbf{\Omega}, \tau, G)$$

holds. It follows that both $\nu y.(\mathbf{\Omega} \oplus y)$ *and* $\nu y.\mathbf{\Omega}$ *belong to the same equivalence class* G*, thus* $(\nu y.(\mathbf{\Omega} \oplus y), \nu y.\mathbf{\Omega}) \in \mathcal{R}$*. If* H *is an equivalence class such that* $\mathbf{\Omega} \oplus \mathbf{I} \in H$*, then by observing that*

$$1 = P(\nu y.(\mathbf{\Omega} \oplus y), \mathbf{I}, H) = P(\nu y.\mathbf{\Omega}, \mathbf{I}, H),$$

we conclude $\mathbf{\Omega} \in H$ *and as a consequence* $(\mathbf{\Omega} \oplus \mathbf{I}, \mathbf{\Omega}) \in \mathcal{R}$*. Let* J *be an equivalence class which contains* $\nu x.x$*. Then*

$$\frac{1}{2} = P(\mathbf{\Omega} \oplus \mathbf{I}, \tau, J) = P(\mathbf{\Omega}, \tau, J).$$

However, this contradicts $P(\mathbf{\Omega}, \tau, J) = 0$*, which follows from the definition of a transition probability matrix (Definition 2.23). Finally, we can conclude that a bisimulation which contains the terms* $\lambda xy.(x \oplus y)$ *and* $(\lambda xy.x) \oplus (\lambda xy.y)$ *does not exist, hence these terms are not bisimilar.*

In Proposition 2.18, we have proved that the context preorder and the context equivalence are sound with respect to the operational semantics. Similarly, the probabilistic applicative similarity and bisimilarity are sound with respect to the operational semantics.

Proposition 2.26. *Let* $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$*. If* $[\![M]\!] \leq [\![N]\!]$ *then* $M \lesssim N$*. So,* $[\![M]\!] = [\![N]\!]$ *implies* $M \sim N$*.*

Proof. The relation

$$\mathcal{R} = \{(M, N) \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \times \Lambda^{\emptyset}_{\oplus,\mathsf{let}} \mid [\![M]\!] \leq [\![N]\!]\} \cup \{(\widetilde{V}, \widetilde{V}) \in \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}} \times \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}\}$$

is a probabilistic applicative simulation (Lemma 3.4, [47]), thus included in the largest probabilistic applicative simulation $\lesssim$. Soundness of bisimilarity follows from $\sim = \lesssim \cap \lesssim^{op}$.

□

The first step towards the proof that the similarity (resp. bisimilarity) implies the context preorder (resp. context equivalence) is proving that the similarity is a precongruence relation. We introduce a new notion of relation called $\Lambda_{\oplus,\mathsf{let}}$-relation, which is a set of triplets (Φ, M, N) such that $M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$. For any relation $\mathcal{R}'$ on the set of $\Lambda_{\oplus,\mathsf{let}}$-terms, we can define a $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$ such that if $(M, N) \in \mathcal{R}'$ and $M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, then $(\Phi, M, N) \in \mathcal{R}$. If $(\Phi, M, N) \in \mathcal{R}$, we will write $\Phi \vdash M \; \mathcal{R} \; N$. We denote the set of all finite subsets of X by $P_{\mathsf{FIN}}(X)$.

Definition 2.27. *A* $\Lambda_{\oplus,\mathsf{let}}$*-relation* $\mathcal{R}$ *is* a precongruence *(resp.* a congruence*) if it is a preorder (resp. an equivalence) and for every* $\Phi \vdash M \; \mathcal{R} \; N$ *and every context* $C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$*, it holds that* $\emptyset \vdash C[M] \; \mathcal{R} \; C[N]$*.*

A precongruence relation can be defined by using a notion of compatibility and this definition is equivalent to Definition 2.27.

Definition 2.28. *A $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$ is compatible if and only if the following five conditions hold:*

(Com1) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in \Phi) : \Phi \vdash x \;\mathcal{R}\; x$

(Com2) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in X \setminus \Phi)\left(\forall M, N \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)$
$\Phi \cup \{x\} \vdash M \;\mathcal{R}\; N \Rightarrow \Phi \vdash (\lambda x.M) \;\mathcal{R}\; (\lambda x.N)$

(Com3) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M, N, L, P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$
$\Phi \vdash M \;\mathcal{R}\; N \;\wedge\; \Phi \vdash L \;\mathcal{R}\; P \Rightarrow \Phi \vdash (ML) \;\mathcal{R}\; (NP)$

(Com4) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M, N, L, P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$
$\Phi \vdash M \;\mathcal{R}\; N \;\wedge\; \Phi \vdash L \;\mathcal{R}\; P \Rightarrow \Phi \vdash (M \oplus L) \;\mathcal{R}\; (N \oplus P)$

(Com5) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in X \setminus \Phi)\left(\forall M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\left(\forall L, P \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)$
$\Phi \vdash M \;\mathcal{R}\; N \;\wedge\; \Phi \cup \{x\} \vdash L \;\mathcal{R}\; P \Rightarrow$
$\Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L) \;\mathcal{R}\; (\mathsf{let}\ x = N\ \mathsf{in}\ P)$

The following auxiliary results will be useful in the sequel.

Lemma 2.29. *Let us consider the properties*

(Com3L) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M, N, L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$
$\Phi \vdash M \;\mathcal{R}\; N \Rightarrow \Phi \vdash (ML) \;\mathcal{R}\; (NL)$

(Com3R) $(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M, N, L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$
$\Phi \vdash M \;\mathcal{R}\; N \Rightarrow \Phi \vdash (LM) \;\mathcal{R}\; (LN)$

If $\mathcal{R}$ is transitive, then (Com3L) *and* (Com3R) *together imply* (Com3).

Proof. Let $\mathcal{R}$ be a transitive relation that satisfies (Com3L) and (Com3R). We prove that $\mathcal{R}$ also satisfies (Com3). Let $\Phi \in P_{\mathsf{FIN}}(X)$ and $M, N, L, P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$ such that $\Phi \vdash M \;\mathcal{R}\; N$ and $\Phi \vdash L \;\mathcal{R}\; P$. By applying (Com3L) to $\Phi \vdash M \;\mathcal{R}\; N$ and term L, we obtain $\Phi \vdash (ML) \;\mathcal{R}\; (NL)$. Further, by applying (Com3R) to $\Phi \vdash L \;\mathcal{R}\; P$ and term N, we obtain $\Phi \vdash (NL) \;\mathcal{R}\; (NP)$. Therefore, $\Phi \vdash (ML) \;\mathcal{R}\; (NP)$ by the transitivity of the relation $\mathcal{R}$. □

Lemma 2.30. *Let us consider the properties*

$$\begin{aligned}(\mathsf{Com4L})\ \ &(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M,N,L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\\ &\Phi \vdash M\ \mathcal{R}\ N \Rightarrow \Phi \vdash (M \oplus L)\ \mathcal{R}\ (N \oplus L)\end{aligned}$$

$$\begin{aligned}(\mathsf{Com4R})\ \ &(\forall \Phi \in P_{\mathsf{FIN}}(X))\left(\forall M,N,L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\\ &\Phi \vdash M\ \mathcal{R}\ N \Rightarrow \Phi \vdash (L \oplus M)\ \mathcal{R}\ (L \oplus N)\end{aligned}$$

If $\mathcal{R}$ is transitive, then (Com4L) *and* (Com4R) *together imply* (Com4).

Proof. Let $\mathcal{R}$ be a transitive relation, which satisfies (Com4L) and (Com4R). We prove that $\mathcal{R}$ also satisfies (Com4). We assume $\Phi \in P_{\mathsf{FIN}}(X)$, $M,N,L,P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, $\Phi \vdash M\ \mathcal{R}\ N$ and $\Phi \vdash L\ \mathcal{R}\ P$. By applying (Com4L) to $\Phi \vdash M\ \mathcal{R}\ N$ and term L, we derive $\Phi \vdash (M \oplus L)\ \mathcal{R}\ (N \oplus L)$. Similarly, by applying (Com4R) to $\Phi \vdash L\ \mathcal{R}\ P$ and term N we get $\Phi \vdash (N \oplus L)\ \mathcal{R}\ (N \oplus P)$. Consequently, $\Phi \vdash (M \oplus L)\ \mathcal{R}\ (N \oplus P)$ by the transitivity of the relation $\mathcal{R}$. □

Lemma 2.31. *Let us consider the properties*

$$\begin{aligned}(\mathsf{Com5L})\ \ &(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in X \setminus \Phi)\left(\forall M,N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\left(\forall L \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)\\ &\Phi \vdash M\ \mathcal{R}\ N \Rightarrow \Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L)\ \mathcal{R}\ (\mathsf{let}\ x = N\ \mathsf{in}\ L)\\ (\mathsf{Com5R})\ \ &(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in X \setminus \Phi)\left(\forall L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\left(\forall M,N \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)\\ &\Phi \cup \{x\} \vdash M\ \mathcal{R}\ N \Rightarrow \Phi \vdash (\mathsf{let}\ x = L\ \mathsf{in}\ M)\ \mathcal{R}\ (\mathsf{let}\ x = L\ \mathsf{in}\ N)\end{aligned}$$

If $\mathcal{R}$ is transitive, then (Com5L) *and* (Com5R) *together imply* (Com5).

Proof. To prove (Com5) we have to show that the hypotheses $\Phi \vdash M\ \mathcal{R}\ N$ and $\Phi \cup \{x\} \vdash L\ \mathcal{R}\ P$ imply $\Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L)\ \mathcal{R}\ (\mathsf{let}\ x = N\ \mathsf{in}\ P)$. If we apply (Com5L) to the first hypothesis, with respect to $L \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}$, we get $\Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L)\ \mathcal{R}\ (\mathsf{let}\ x = N\ \mathsf{in}\ L)$. Similarly, applying (Com5R) to the second hypothesis, with respect to $N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, we obtain $\Phi \vdash (\mathsf{let}\ x = N\ \mathsf{in}\ L)\ \mathcal{R}\ (\mathsf{let}\ x = N\ \mathsf{in}\ P)$. Then by the transitivity property of $\mathcal{R}$ we conclude the claim. □

Definition 2.32. *A $\Lambda_{\oplus,\mathsf{let}}$-relation is a precongruence (resp. congruence) if it is a preorder relation (resp. equivalence) and compatible.*

The context preorder is a precongruence relation, whereas the context equivalence is a congruence relation.

Proposition 2.33. *The context preorder $\leq$ is a precongruence relation.*

Proof. In order to prove that $\leq$ is a precongruence, we need to show that $\leq$ is a preorder (reflexive and transitive) relation, which is compatible. The relation $\leq$ is reflexive by its definition and we prove it is transitive, i.e.

$$(\forall \Phi \in P_{\mathsf{FIN}}(X))\ (\forall M, N, L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}})$$
$$\Phi \vdash M \leq N \wedge \Phi \vdash N \leq L \Rightarrow \Phi \vdash M \leq L.$$

Let us assume that $\Phi \vdash M \leq N$ and $\Phi \vdash N \leq L$, then

(1) $\left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right) \sum [\![C[M]]\!] \leq \sum [\![C[N]]\!]$,

(2) $\left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right) \sum [\![C[N]]\!] \leq \sum [\![C[L]]\!]$.

To prove $\Phi \vdash M \leq L$ we need to show that for every $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$, $\sum [\![D[M]]\!] \leq \sum [\![D[L]]\!]$. For any such context D, from the hypothesis (1) and (2) we have $\sum [\![D[M]]\!] \leq \sum [\![D[N]]\!] \leq \sum [\![D[L]]\!]$. Thus, $\leq$ is transitive. In order to prove that $\leq$ is compatible, we show that it satisfies the conditions (Com1), (Com2), (Com3), (Com4) and (Com5). The proof of (Com1) follows from the reflexivity of $\leq$.

(Com2) To prove that (Com2) holds, we show $(\forall \Phi \in P_{\mathsf{FIN}}(X))\ (\forall x \in X \setminus \Phi)$ $\left(\forall M, N \in \Lambda^{\Phi \cup \{x\}}_{\oplus,\mathsf{let}}\right)$

$$\Phi \cup \{x\} \vdash M \leq N \Rightarrow \Phi \vdash \lambda x.M \leq \lambda x.N.$$

From the assumption $\Phi \cup \{x\} \vdash M \leq N$, we have $\left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi \cup \{x\};\emptyset)}\right)$ $\sum [\![C[M]]\!] \leq \sum [\![C[N]]\!]$ as the hypothesis. Let us consider a context $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi,\emptyset)}$. Since the context $\lambda x.[\cdot]$ belongs to the set $\mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi \cup \{x\};\Phi)}$, then $E = D[\lambda x.[\cdot]] \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi \cup \{x\};\emptyset)}$ by Lemma 2.14. We can apply the hypothesis to the context E and obtain $\sum [\![E[M]]\!] \leq \sum [\![E[N]]\!]$, that is $\sum [\![D[\lambda x.M]]\!] \leq \sum [\![D[\lambda x.N]]\!]$. Thus, $\Phi \vdash \lambda x.M \leq \lambda x.N$.

(Com3) As already proved, $\leq$ is a transitive relation, thus by Lemma 2.29 it is enough to prove two characterizations (Com3L) and (Com3R). We prove (Com3L): $(\forall \Phi \in P_{\mathsf{FIN}}(X))\ \left(\forall M, N, L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$

$$\Phi \vdash M \leq N \Rightarrow \Phi \vdash ML \leq NL.$$

Let us assume $\Phi \vdash M \leq N$, then

$$\left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right) \sum [\![C[M]]\!] \leq \sum [\![C[N]]\!]$$

holds as the hypothesis. We want to show that for any context $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$, $\sum[\![D[ML]]\!] \leq \sum[\![D[NL]]\!]$ holds. For an arbitrary context $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$ and $[\cdot]L \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Phi)}$, we get $E = D[[\cdot]L] \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$ by Lemma 2.14. From the hypothesis, we conclude that $\sum[\![E[M]]\!] \leq \sum[\![E[N]]\!]$ holds, i.e. $\sum[\![D[ML]]\!] \leq \sum[\![D[NL]]\!]$. Thus, $\Phi \vdash ML \leq NL$. We omit the proof of (Com3R), since it is analogous to the proof of (Com3L).

(Com4) As in the previous case, the fact that $\leq$ is transitive and Lemma 2.30 ensure that (Com4L) and (Com4R) imply (Com4), so it is enough to prove these two characterizations. We omit the proof of (Com4L) and (Com4R), since we prove it by a similar reasoning as in the proof of (Com3L).

(Com5) By Lemma 2.31 it is enough to prove two characterizations (Com5L) and (Com5R). We prove (Com5L): $(\forall \Phi \in P_{\mathsf{FIN}}(X))\ (\forall x \in X \setminus \Phi)\ \left(\forall M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)\ \left(\forall L \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)$

$$\Phi \vdash M \leq N \Rightarrow \Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L) \leq (\mathsf{let}\ x = N\ \mathsf{in}\ L).$$

If we assume $\Phi \vdash M \leq N$, then we have $\left(\forall C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}\right)\ \sum[\![C[M]]\!] \leq \sum[\![C[N]]\!]$. We want to show that for any context $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$, $\sum[\![D[\mathsf{let}\ x = M\ \mathsf{in}\ L]]\!] \leq \sum[\![D[\mathsf{let}\ x = N\ \mathsf{in}\ L]]\!]$ holds. For an arbitrary context $D \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$ and the context $\mathsf{let}\ x = [\cdot]\ \mathsf{in}\ L \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\Phi)}$, we get $E = D[\mathsf{let}\ x = [\cdot]\ \mathsf{in}\ L] \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$ by Lemma 2.14. From the hypothesis, we can conclude that $\sum[\![E[M]]\!] \leq \sum[\![E[N]]\!]$ holds, i.e. $\sum[\![D[\mathsf{let}\ x = M\ \mathsf{in}\ L]]\!] \leq \sum[\![D[\mathsf{let}\ x = N\ \mathsf{in}\ L]]\!]$. Thus, $\Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ L) \leq (\mathsf{let}\ x = N\ \mathsf{in}\ L)$. The characterization (Com5R) can be proved in a similar way. □

Corollary 2.34. *The context equivalence $\simeq$ is a congruence relation.*

Proof. This statement is a consequence of Proposition 2.33 and the fact that $\simeq = \leq \cap \leq^{op}$. □

In the following two definitions we introduce the notions of substitutive relation and relation closed under term-substitution, which will be used in the sequel.

Definition 2.35. *A $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$ is* (term) substitutive *if for all $\Phi \in P_{\mathsf{FIN}}(X), x \in X \setminus \Phi, M, N \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}, L, P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, we have*

$$\Phi \cup \{x\} \vdash M\ \mathcal{R}\ N \wedge \Phi \vdash L\ \mathcal{R}\ P \Rightarrow \Phi \vdash M\{L/x\}\ \mathcal{R}\ N\{P/x\}.$$

Definition 2.36. *A $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$ is* closed under term-substitution *if for all $\Phi \in P_{\mathsf{FIN}}(X), x \in X \setminus \Phi, M, N \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}, L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, we have*

$$\Phi \cup \{x\} \vdash M\ \mathcal{R}\ N \wedge L \in \Lambda^{\Phi}_{\oplus,\mathsf{let}} \Rightarrow \Phi \vdash M\{L/x\}\ \mathcal{R}\ N\{L/x\}.$$

Notice that the similarity and bisimilarity are closed under term-substitution by their definition. Compared to the proof of Proposition 2.33, proving that similarity is a precongruence is more involved. The proof is very technical and we have followed the technique used in [35, 36, 47]. Given that similarity is a preorder relation, by Definition 2.32, it remains to be proven that it is a compatible relation. We do this in the next subsection using Howe's technique ([84]), which is a commonly used technique for proving that similarity (resp. bisimilarity) is a precongruence (resp. congruence).

2.2.1 Similarity is a precoungruence

First, we introduce Howe's lifting $\mathcal{R}^H$ of an arbitrary $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$, which is defined by the rules in Figure 2.5. Next, we present some auxiliary results.

$$\frac{\Phi \vdash x\ \mathcal{R}\ M}{\Phi \vdash x\ \mathcal{R}^H\ M}\ (\text{How1})$$

$$\frac{\Phi \cup \{x\} \vdash M\ \mathcal{R}^H\ L \quad \Phi \vdash (\lambda x.L)\ \mathcal{R}\ N \quad x \notin \Phi}{\Phi \vdash (\lambda x.M)\ \mathcal{R}^H\ N}\ (\text{How2})$$

$$\frac{\Phi \vdash M\ \mathcal{R}^H\ P \quad \Phi \vdash N\ \mathcal{R}^H\ Q \quad \Phi \vdash (PQ)\ \mathcal{R}\ L}{\Phi \vdash (MN)\ \mathcal{R}^H\ L}\ (\text{How3})$$

$$\frac{\Phi \vdash M\ \mathcal{R}^H\ P \quad \Phi \vdash N\ \mathcal{R}^H\ Q \quad \Phi \vdash (P \oplus Q)\ \mathcal{R}\ L}{\Phi \vdash (M \oplus N)\ \mathcal{R}^H\ L}\ (\text{How4})$$

$$\frac{\Phi \vdash M\ \mathcal{R}^H\ P \quad \Phi \cup \{x\} \vdash N\ \mathcal{R}^H\ Q \quad \Phi \vdash (\mathsf{let}\ x = P\ \mathsf{in}\ Q)\ \mathcal{R}\ L}{\Phi \vdash (\mathsf{let}\ x = M\ \mathsf{in}\ N)\ \mathcal{R}^H\ L}\ (\text{How5})$$

FIGURE 2.5: Howe's lifting for $\Lambda_{\oplus,\mathsf{let}}$

Lemma 2.37. *If $\mathcal{R}$ is reflexive, then $\mathcal{R}^H$ is compatible.*

Proof. The proof proceeds similarly to the proof of Lemma 3.10 in [46]. □

Lemma 2.38. *If $\mathcal{R}$ is transitive, then $\Phi \vdash M\ \mathcal{R}^H\ N$ and $\Phi \vdash N\ \mathcal{R}\ L$ imply $\Phi \vdash M\ \mathcal{R}^H\ L$.*

Proof. The proof proceeds by induction on the derivation of $\Phi \vdash M\ \mathcal{R}^H\ N$. □

Lemma 2.39. *If $\mathcal{R}$ is reflexive, then $\Phi \vdash M\ \mathcal{R}\ N$ implies $\Phi \vdash M\ \mathcal{R}^H\ N$.*

Proof. The proof proceeds by induction on the structure of M. □

Lemma 2.40. *If $\mathcal{R}$ is reflexive, transitive and closed under term-substitution, then $\mathcal{R}^H$ is (term) substitutive and hence also closed under term-substitution.*

Proof. We need to show that: $(\forall \Phi \in P_{\mathsf{FIN}}(X))\,(\forall x \in X \setminus \Phi)\left(\forall M, N \in \Lambda^{\Phi\cup\{x\}}_{\oplus,\mathsf{let}}\right)\left(\forall L, P \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}\right)$

$$\Phi \cup \{x\} \vdash M\ \mathcal{R}^H\ N\ \wedge\ \Phi \vdash L\ \mathcal{R}^H\ P\ \Rightarrow \Phi \vdash M\{L/x\}\ \mathcal{R}^H\ N\{P/x\}.$$

The proof proceeds by induction on the derivation of $\Phi \cup \{x\} \vdash M\ \mathcal{R}^H\ N$. □

Definition 2.41. *For a relation $\mathcal{R}$, its transitive closure, denoted by $\mathcal{R}^+$, is defined by the rules in Figure 2.6.*

$$\frac{\Phi \vdash M\ \mathcal{R}\ N}{\Phi \vdash M\ \mathcal{R}^+\ N}\ (\text{TC1})$$

$$\frac{\Phi \vdash M\ \mathcal{R}^+\ N \qquad \Phi \vdash N\ \mathcal{R}^+\ L}{\Phi \vdash M\ \mathcal{R}^+\ L}\ (\text{TC2})$$

FIGURE 2.6: Transitive closure for $\Lambda_{\oplus,\mathsf{let}}$

In the following lemmas we identify some properties of a transitive closure of a relation.

Lemma 2.42. *For the transitive closure $\mathcal{R}^+$ of a relation $\mathcal{R}$ the following holds:*

1. *If $\mathcal{R}$ is compatible, then so is $\mathcal{R}^+$.*
2. *If $\mathcal{R}$ is closed under term-substitution, then so is $\mathcal{R}^+$.*

Proof. The proofs of the two claims follow the same reasoning as the proofs of Lemma 3.14 and Lemma 3.15, respectively, in [46]. □

Lemma 2.43. *If a $\Lambda_{\oplus,\mathsf{let}}$-relation $\mathcal{R}$ is a preorder, then so is $(\mathcal{R}^H)^+$.*

Proof. Let $\mathcal{R}$ be preorder. Since $\mathcal{R}$ is reflexive, we get that $\mathcal{R}^H$ is compatible by Lemma 2.37 and then, by Lemma 2.42 it follows that $(\mathcal{R}^H)^+$ is also compatible. Therefore, $(\mathcal{R}^H)^+$ is reflexive. Relation $(\mathcal{R}^H)^+$ is a transitive closure of the relation $\mathcal{R}^H$, so the transitivity of the relation $(\mathcal{R}^H)^+$ follows from its definition.

□

A key step in proving that similarity is a precongruence is Key Lemma (Lemma 2.48 below). In the proof of Key Lemma we will use auxiliary lemmas about probability assignment (Lemma 2.45) and similarity (Proposition 2.46 and Proposition 2.47).

Definition 2.44. *An ordered pair* $\mathbb{P} = (\{p_i\}_{1\leq i\leq n}, \{r_I\}_{I\subseteq\{1,\ldots,n\}})$, *where* $p_1, \ldots, p_n \in [0,1]$ *and for each* $I \subseteq \{1,\ldots,n\}$, $r_I \in [0,1]$, *is a probability assignment if for each* $I \subseteq \{1,\ldots,n\}$ *it holds that* $\sum_{i\in I} p_i \leq \sum_{J\cap I\neq\emptyset} r_J$.

Lemma 2.45. *Let* $\mathbb{P} = (\{p_i\}_{1\leq i\leq n}, \{r_I\}_{I\subseteq\{1,\ldots,n\}})$ *be a probability assignment. Then for every nonempty set* $I \subseteq \{1,\ldots,n\}$ *and for every* $k \in I$ *there exists* $s_{k,I} \in [0,1]$ *which satisfies the following properties:*

1. *for every* I, *it holds that* $\sum_{k\in I} s_{k,I} \leq 1$;
2. *for every* $k \in \{1,\ldots,n\}$, *it holds that* $p_k \leq \sum_{k\in I} s_{k,I} \cdot r_I$.

Lemma 2.45 is proved in [46]. In the proof of Key Lemma, we will also use the following auxiliary results.

Proposition 2.46. *For every* $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$, *it holds that* $\lesssim(\nu x.X) = \nu x.(\lesssim(X))$, *where* $\nu x.(\lesssim(X))$ *stands for the set* $\{\nu x.M \mid \exists N \in X, N \lesssim M\}$.

Proof. The proof follows from the definition of similarity. □

The following proposition is a direct consequence of Proposition 2.46.

Proposition 2.47. *If* $M \lesssim N$, *then for every* $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$, $[\![M]\!](\lambda x.X) \leq [\![N]\!](\lambda x.\lesssim(X))$.

Lemma 2.48. (Key Lemma) *Let* $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. *If* $M \lesssim^H N$, *then it holds that* $[\![M]\!](\lambda x.X) \leq [\![N]\!](\lambda x.(\lesssim^H(X)))$, *for every* $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$.

Proof. From the definition of semantics, we know that

$$[\![M]\!] = \sup\{\mathscr{D} \mid M \Downarrow \mathscr{D}\}.$$

Thus, it is sufficient to prove that whenever $M \lesssim^H N$ and $M \Downarrow \mathscr{D}$, we have $\mathscr{D}(\lambda x.X) \leq [\![N]\!](\lambda x.(\lesssim^H(X))$ for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$. The proof proceeds by induction on the derivation of $M \Downarrow \mathscr{D}$, performing the case analysis on the last rule applied.

- If $M \Downarrow \emptyset$, then we have $\mathcal{D}(\lambda x.X) = 0 \leq [\![N]\!](\lambda x.(\lesssim^H(X)))$ for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$.

- Next, we consider the case where M is a value $\lambda x.Q$ and the distribution $\mathcal{D}$ has all of its mass on $\lambda x.Q$, that is $\mathscr{D}(\lambda x.Q) = 1$. Since M is a value, the last rule used in the derivation of $M \lesssim^H N$ has to be (How2). Thus, for some $P \in \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$, it holds that $x \vdash Q \lesssim^H P$ and $\emptyset \vdash \lambda x.P \lesssim N$. For $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$ we consider two cases:
  - If $Q \notin X$, then $\mathcal{D}(\lambda x.X) = 0$ and the statement holds.
  - If $Q \in X$, then $\mathcal{D}(\lambda x.X) = 1$ and $P \in \lesssim^H (X)$. For every $L \in \lesssim (P)$, we have that $x \vdash Q \lesssim^H P$ and $x \vdash P \lesssim L$. By Lemma 2.38 we conclude that $x \vdash Q \lesssim^H L$ holds. Thus, $L \in \lesssim^H (X)$ and it holds that $\lesssim (P) \subseteq \lesssim^H (X)$. By Proposition 2.47 we obtain the following

$$\begin{aligned}\mathcal{D}(\lambda x.X) &= 1\\ &= [\![\lambda x.P]\!](\lambda x.P)\\ &\leq [\![N]\!](\lambda x.\lesssim (P))\\ &\leq [\![N]\!](\lambda x.\lesssim^H (X)).\end{aligned}$$

- Let M be an application LP. Then we have $\mathcal{D} = \sum_{\lambda x.Q} \mathcal{F}(\lambda x.Q) \cdot \mathcal{H}_{Q,P}$ where $L \Downarrow \mathcal{F}$ and for any $\lambda x.Q \in \mathsf{S}(\mathcal{F})$, $\{Q\{P/x\} \Downarrow \mathcal{H}_{Q,P}\}$. The last rule used in the derivation of $M \lesssim^H N$ has to be (How3), thus we get $\emptyset \vdash L \lesssim^H R$, $\emptyset \vdash P \lesssim^H S$ and $\emptyset \vdash RS \lesssim N$. If we apply the induction hypothesis to $L \Downarrow \mathcal{F}$ and $\emptyset \vdash L \lesssim^H R$, we obtain that, for any $Y \subseteq \Lambda_{\oplus,\mathsf{let}}(x)$, it holds that

$$\mathcal{F}(\lambda x.Y) \leq [\![R]\!](\lambda x.\lesssim^H (Y)). \tag{2.8}$$

  Since $\mathcal{F}$ is a finite distribution, the distribution $\mathcal{D} = \sum_{\lambda x.Q} \mathcal{F}(\lambda x.Q) \cdot \mathcal{H}_{Q,P}$ is a sum of finitely many summands. Let us assume that $\mathsf{S}(\mathcal{F}) = \{\lambda x.Q_1, \ldots, \lambda x.Q_n\}$. From equation (2.8) we conclude

$$\mathcal{F}(\bigcup_{i\in I} \lambda x.Q_i) \leq [\![R]\!](\bigcup_{i\in I} \lambda x.\lesssim^H (Q_i)),$$

  for every $I \subseteq \{1, \ldots, n\}$ which allows us to apply Lemma 2.45. Hence, for every $U \in \bigcup_{i=1}^n \lesssim^H (Q_i)$ there exist real numbers $r_1^{U,R}, \ldots, r_n^{U,R}$ such that:

$$[\![R]\!](\lambda x.U) \geq \sum_{i=1}^n r_i^{U,R}, \qquad \forall U \in \bigcup_{i=1}^n \lesssim^H (Q_i);$$

$$\mathcal{F}(\lambda x.Q_i) \leq \sum_{U \in \lesssim^H (Q_i)} r_i^{U,R}, \qquad \forall i \in \{1, \ldots, n\}.$$

From these equations we can conclude the following

$$\mathcal{D} \leq \sum_{i=1}^{n} \left(\sum_{U \in \lesssim^H(Q_i)} r_i^{U,R} \right) \cdot \mathcal{H}_{Q_i,P} = \sum_{i=1}^{n} \sum_{U \in \lesssim^H(Q_i)} r_i^{U,R} \cdot \mathcal{H}_{Q_i,P}.$$

Since $Q_i \lesssim^H U$ and $P \lesssim^H S$ holds, by Lemma 2.40 we have $Q_i\{P/x\} \lesssim^H U\{S/x\}$. Now, by applying the induction hypothesis to the derivations $Q_i\{P/x\} \Downarrow \mathcal{H}_{Q_i,P}$, $i \in \{1,\ldots,n\}$, for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$ we obtain

$$\begin{aligned}
\mathcal{D}(\lambda x.X) &\leq \sum_{i=1}^{n} \sum_{U \in \lesssim^H(Q_i)} r_i^{U,R} [\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&\leq \sum_{i=1}^{n} \sum_{U \in \bigcup_{i=1}^{n} \lesssim^H(Q_i)} r_i^{U,R} [\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&= \sum_{U \in \bigcup_{i=1}^{n} \lesssim^H(Q_i)} \sum_{i=1}^{n} r_i^{U,R} [\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&= \sum_{U \in \bigcup_{i=1}^{n} \lesssim^H(Q_i)} \left(\sum_{i=1}^{n} r_i^{U,R} \right) [\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&\leq \sum_{U \in \bigcup_{i=1}^{n} \lesssim^H(Q_i)} [\![R]\!](\lambda x.U)[\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&\leq \sum_{U \in \Lambda^{\{x\}}_{\oplus,\mathsf{let}}} [\![R]\!](\lambda x.U)[\![U\{S/x\}]\!](\lambda x.\, \lesssim^H (X)) \\
&= [\![RS]\!](\lambda x.\, \lesssim^H (X)) \\
&\leq [\![N]\!](\lambda x.\, \lesssim (\lesssim^H (X))) \\
&\leq [\![N]\!](\lambda x.\, \lesssim^H (X)).
\end{aligned}$$

- If M is a probabilistic sum $L \oplus P$, then $\mathcal{D} = \frac{1}{2}\mathcal{F} + \frac{1}{2}\mathcal{E}$ where $L \Downarrow \mathcal{F}$ and $P \Downarrow \mathcal{E}$. The last used rule in the derivation of $M \lesssim^H N$ has to be (How4), hence for some $R, S \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, we have that $\emptyset \vdash L \lesssim^H R$, $\emptyset \vdash P \lesssim^H S$ and $\emptyset \vdash R \oplus S \lesssim N$ hold. If we apply the induction hypothesis to $L \Downarrow \mathcal{F}$ and $\emptyset \vdash L \lesssim^H R$, for any $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$ we obtain $\mathcal{F}(\lambda x.X) \leq [\![R]\!](\lambda x.\, \lesssim^H (X))$. Similarly, if we apply the induction hypothesis to $P \Downarrow \mathcal{E}$ and $\emptyset \vdash P \lesssim^H S$, for any $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$ we obtain $\mathcal{E}(\lambda x.X) \leq [\![S]\!](\lambda x.\, \lesssim^H (X))$. Since $\emptyset \vdash R \oplus S \lesssim N$, we have $[\![R \oplus S]\!](\lambda x.\, \lesssim^H (X)) \leq [\![N]\!](\lambda x.\, \lesssim^H (X))$. By

Proposition 2.11 and the previously concluded statements the following holds

$$\begin{aligned}\mathcal{D}(\lambda x.X) &= \frac{1}{2}\mathcal{F}(\lambda x.X) + \frac{1}{2}\mathcal{E}(\lambda x.X)\\ &\leq \frac{1}{2}[\![R]\!](\lambda x.\lesssim^H (X)) + \frac{1}{2}[\![S]\!](\lambda x.\lesssim^H (X))\\ &= [\![R \oplus S]\!](\lambda x.\lesssim^H (X))\\ &= [\![N]\!](\lambda x.\lesssim^H (X)).\end{aligned}$$

- If $M = (\mathsf{let}\ x = L\ \mathsf{in}\ P)$, then $\mathscr{D} = \sum_{\lambda x.Q} \mathscr{F}(\lambda x.Q) \cdot \mathscr{H}_{Q,P}$ where $L \Downarrow \mathscr{F}$ and for any $\lambda x.Q \in \mathsf{S}(\mathscr{F})$, $\{P\{\lambda x.Q/x\} \Downarrow \mathscr{H}_{Q,P}\}$ holds. In the derivation of $\emptyset \vdash M \lesssim^H N$, the last rule used has to be (How5), meaning that for some terms R and S, we have $\emptyset \vdash L \lesssim^H R$, $x \vdash P \lesssim^H S$ and $\emptyset \vdash (\mathsf{let}\ x = R\ \mathsf{in}\ S) \lesssim N$. From $L \Downarrow \mathscr{F}$ and $\emptyset \vdash L \lesssim^H R$, by the induction hypothesis we get

$$\mathscr{F}(\lambda x.Y) \leq [\![R]\!](\lambda x.\lesssim^H (Y)), \tag{2.9}$$

for any $Y \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$. The distribution $\mathscr{F}$ is a finite distribution and as a consequence the sum $\mathscr{D} = \sum_{\lambda x.Q} \mathscr{F}(\lambda x.Q) \cdot \mathscr{H}_{Q,P}$ has finitely many summands. Let $\mathsf{S}(\mathscr{F}) = \{\lambda x.Q_1, \ldots, \lambda x.Q_n\}$ be the the support of the distribution $\mathscr{F}$. From Equation (2.9), it follows that

$$\mathscr{F}(\bigcup_{i\in I} \lambda x.Q_i) \leq [\![R]\!](\bigcup_{i\in I} \lambda x.\lesssim^H (Q_i))$$

holds for every $I \subseteq \{1, \ldots, n\}$. By Lemma 2.45, we get that for every $U \in \bigcup_{i=1}^n \lesssim^H (Q_i)$, there exist real numbers $r_1^{U,R}, \ldots, r_n^{U,R}$ such that:

$$[\![R]\!](\lambda x.U) \geq \sum_{i=1}^n r_i^{U,R}, \qquad \forall U \in \bigcup_{i=1}^n \lesssim^H (Q_i);$$

$$\mathscr{F}(\lambda x.Q_i) \leq \sum_{U\in\lesssim^H(Q_i)} r_i^{U,R}, \qquad \forall i \in \{1, \ldots, n\}.$$

We derive

$$\mathscr{D} \leq \sum_{i=1}^n \left(\sum_{U\in\lesssim^H(Q_i)} r_i^{U,R} \right) \cdot \mathscr{H}_{Q_i,P} = \sum_{i=1}^n \sum_{U\in\lesssim^H(Q_i)} r_i^{U,R} \cdot \mathscr{H}_{Q_i,P}.$$

By Lemma 2.37, the relation $\lesssim^H$ is compatible. From $Q_i \lesssim^H U$, we get $\lambda x.Q_i \lesssim^H \lambda x.U$. From the latter and $P \lesssim^H S$, we obtain

$P\{\lambda x.Q_i/x\} \lesssim^H S\{\lambda x.U/x\}$ by Lemma 2.40. Finally, from the derivations $P\{\lambda x.Q_i/x\} \Downarrow \mathscr{H}_{Q_i,P}$, $i \in \{1,\ldots,n\}$, by the induction hypothesis and by similar reasoning as in the case of application, we get

$$\mathscr{D}(\lambda x.X) \leq [\![N]\!](\lambda x.\lesssim^H (X)),$$

for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$.

This concludes the proof. □

Using Key Lemma and other presented auxiliary results, we derive the most important result of this section.

Proposition 2.49. *The similarity $\lesssim$ (resp. bisimilarity $\sim$) is a precongruence (resp. congruence) relation for $\Lambda_{\oplus,\mathsf{let}}$-terms.*

Proof. The idea of the proof is the following: we first prove that the relation $(\lesssim^H)^+$ is a precongruence and then we prove that relations $(\lesssim^H)^+$ and $\lesssim$ coincide. The relation $\lesssim$ is a preorder, so it follows that the relation $(\lesssim^H)^+$ is also a preorder by Lemma 2.43. From the reflexivity of relation $\lesssim$ and Lemma 2.37, we conclude that the relation $\lesssim^H$ is compatible and it follows that $(\lesssim^H)^+$ is also compatible by Lemma 2.42. We have proved that $(\lesssim^H)^+$ is a precongruence. As a direct consequence of the definition of Howe's lifting and transitive closure we have $\lesssim \subseteq (\lesssim^H)^+$. It remains to prove $(\lesssim^H)^+ \subseteq \lesssim$. Since $\lesssim$ is the largest probabilistic simulation, it is sufficient to prove that $(\lesssim^H)^+$ is included in some probabilistic simulation. We consider the relation $\mathcal{R} = \{(M,N) : M\ (\lesssim^H)^+\ N\} \cup \{(\nu x.M, \nu x.N) : M\ (\lesssim^H)^+\ N\}$. From the definition of $\mathcal{R}$, we see that $(\lesssim^H)^+ \subseteq \mathcal{R}$. We prove that $\mathcal{R}$ is a probabilistic simulation. By Lemma 2.40 and Lemma 2.42, we have that $(\lesssim^H)^+$ is closed under term-substitution, thus it is sufficient to consider only closed terms and distinguished values. The relation $(\lesssim^H)^+$ is a preorder relation, so it follows that $\mathcal{R}$ is also a preorder relation. It remains to prove the following:

1. If $M\ (\lesssim^H)^+\ N$, then for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$ it holds that $P(M,\tau,\nu x.X) \leq P(N,\tau,\mathcal{R}(\nu x.X))$.

2. If $M\ (\lesssim^H)^+\ N$, then for every $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and for every $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$, $P(\nu x.M, L, X) \leq P(\nu x.N, L, \mathcal{R}(X))$.

We prove the first point by induction on the length of the derivation $M\ (\lesssim^H)^+\ N$, performing the case analysis on the last rule applied. The base case is when the rule (TC1) is the last rule used, that is $M\ (\lesssim^H)^+\ N$ is

obtained from $M \lesssim^H N$. From the latter and Key Lemma, we have

$$\begin{aligned} P(M, \tau, \nu x.X) &= [\![M]\!](\lambda x.X) \\ &\leq [\![N]\!](\lambda x.\lesssim^H (X)) \\ &\leq [\![N]\!](\lambda x.(\lesssim^H)^+(X)) \\ &\leq [\![N]\!](\mathcal{R}(\lambda x.X)) \\ &= P(N, \tau, \mathcal{R}(\nu x.X)). \end{aligned}$$

If the last rule used was (TC2), then $M\ (\lesssim^H)^+\ P$ and $P\ (\lesssim^H)^+\ N$ hold for some $P \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. By the induction hypothesis, we have

$$P(M, \tau, X) \leq P(P, \tau, \mathcal{R}(X)),$$

$$P(P, \tau, \mathcal{R}(X)) \leq P(N, \tau, \mathcal{R}(\mathcal{R}(X))).$$

From these inequalities and the fact that $\mathcal{R}(\mathcal{R}(X)) \subseteq \mathcal{R}(X)$ holds for any transitive relation $\mathcal{R}$ and set X, we obtain

$$P(M, \tau, X) \leq P(N, \tau, \mathcal{R}(X)).$$

This concludes the proof of the first point.

Next, we prove the second point, that is whenever $M\ (\lesssim^H)^+\ N$, $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $X \subseteq \Lambda^{\{x\}}_{\oplus,\mathsf{let}}$, then $P(\nu x.M, L, X) \leq P(\nu x.N, L, \mathcal{R}(X))$. Let $M\ (\lesssim^H)^+\ N$ and $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. Since the relation $(\lesssim^H)^+$ is closed under term-substitution, we have $M\{L/x\}\ (\lesssim^H)^+\ N\{L/x\}$. It follows that if $M\{L/x\} \in X$, then $N\{L/x\} \in (\lesssim^H)^+(X)$ and we obtain

$$\begin{aligned} P(\nu x.M, L, X) &= 1 \\ &= P(\nu x.N, L, (\lesssim^H)^+(X)) \\ &= P(\nu x.N, L, \mathcal{R}(X)). \end{aligned}$$

If $M\{L/x\} \notin X$, then we have $P(\nu x.M, L, X) = 0 \leq P(\nu x.N, L, \mathcal{R}(X))$.

This proves the second point and concludes the proof that similarity is a precongruence.

Finally, we prove that bisimilarity is a congruence. The relation $\sim$ is an equivalence relation by its definition, hence the compatibility of the relation $\sim$ follows from the compatibility of relation $\lesssim$ and definition $\sim = \lesssim \cap \lesssim^{op}$. Thus, bisimilarity is a congruence.

□

As a direct consequence of the previous lemma we have that similarity (resp. bisimilarity) is sound with respect to the context preorder (resp. context equivalence).

Theorem 2.50 (Soundness)**.** *For every* $M, N \in \Lambda^{\Phi}_{\oplus,\mathsf{let}}$, $\Phi \vdash M \lesssim N$ *implies* $\Phi \vdash M \leq N$. *Therefore,* $M \sim N$ *implies* $\Phi \vdash M \simeq N$.

Proof. Let $\Phi \vdash M \lesssim N$. Then for every context $C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\Phi;\emptyset)}$, we have $\emptyset \vdash C[M] \lesssim C[N]$ by Proposition 2.49, meaning that there exists a simulation relation which contains the pair $(C[M], C[N])$. By Definition 2.21, it follows that $\sum[\![C[M]]\!] \leq \sum[\![C[N]]\!]$. Thus, $\Phi \vdash M \leq N$. The proof that the bisimilarity is included in context equivalence follows from the definitions $\sim = \lesssim \cap \lesssim^{op}$ and $\simeq = \leq \cap \leq^{op}$. □

2.3 Full abstraction

In this section we present the proof of full abstraction. We prove that whenever two terms are context equivalent, they are also bisimilar, which is the converse of Theorem 2.50. In the proof we will use a new notion of equivalence, called *testing equivalence*.

In [111], Larsen and Skou have introduced a notion of testing language for a discrete probabilistic transition system and proved that two processes are bisimilar if and only if the success probability of any test is the same for both processes. In [166], Van Breugel, Mislove, Ouaknine and Worrell have extended the results of [111] to labelled Markov processes. We consider the discrete-time version of Markov processes, namely Markov chains, so we adapt the result of [166] to the labelled Markov chains (as in [35, 47]). First we give the general definitions of a testing language and the success probability of a test for a labelled Markov chain.

Definition 2.51 ([35])**.** *Let* $(\mathcal{S}, \mathcal{L}, \mathcal{P})$ *be a labelled Markov chain. The testing language* $\mathcal{T}_{(\mathcal{S},\mathcal{L},\mathcal{P})}$ *for* $(\mathcal{S}, \mathcal{L}, \mathcal{P})$ *is given by the grammar*

$$\boxed{t ::= \omega \mid a.t \mid (t, t)} \tag{2.10}$$

where ω *is a symbol for termination and* $a \in \mathcal{L}$ *is an action (label).*

A test is an algorithm that consists of finite sequence of actions. The symbol ω represents a test which always succeeds, meaning that none of the action is performed. The test $a.t$ consists in performing an action a and in the case of success performing the test t. Performing the test (t, s) consists in making two copies of the current state and performing independently tests t and s on the same state. Performing the test on a state can result in either a success or a failure with a given probability. In the following definition we introduce the notion of a success probability of a test.

Definition 2.52 ([35]). *Let $(\mathcal{S}, \mathcal{L}, \mathcal{P})$ be a labelled Markov chain. We define the family $\{P_t : \mathcal{S} \to \mathbb{R}_{[0,1]}\}_{t \in \mathcal{T}_{(\mathcal{S},\mathcal{L},\mathcal{P})}}$ by induction on the structure of t:*

- $P_\omega(s) = 1$,
- $P_{a.t}(s) = \sum_{s' \in \mathcal{S}} \mathcal{P}(s, a, s') P_t(s')$,
- $P_{(t_1,t_2)}(s) = P_{t_1}(s) \cdot P_{t_2}(s)$.

For $t \in \mathcal{T}_{(\mathcal{S},\mathcal{L},\mathcal{P})}$ and $s \in \mathcal{S}$, we refer to $P_t(s)$ as the success probability *of t applied to s.*

In Section 2.2 we have shown that probabilistic λ-calculus $\Lambda_{\oplus,\mathsf{let}}$ can be seen as a labelled Markov chain, so we can define the testing language for $\Lambda_{\oplus,\mathsf{let}}$. We illustrate how to perform the test and how to compute the success probability in the following example.

Example 2.53. *We consider the terms $\lambda xy.(x \oplus y)$ and $(\lambda xy.x) \oplus (\lambda xy.y)$. In Example 2.16 we have proved that these terms are not context equivalent. These terms can be discriminated by the test $t = \tau.(\mathbf{I}.\tau.\mathbf{\Omega}.\tau.\omega, \mathbf{I}.\tau.\mathbf{\Omega}.\tau.\omega)$. The success probability of the test t for the term $\lambda xy.(x \oplus y)$ is $\frac{1}{4}$, whereas the success probability of the test t for the term $(\lambda xy.x) \oplus (\lambda xy.y)$ is $\frac{1}{2}$. We have sketched the computation of success probability in Figure 2.7.*

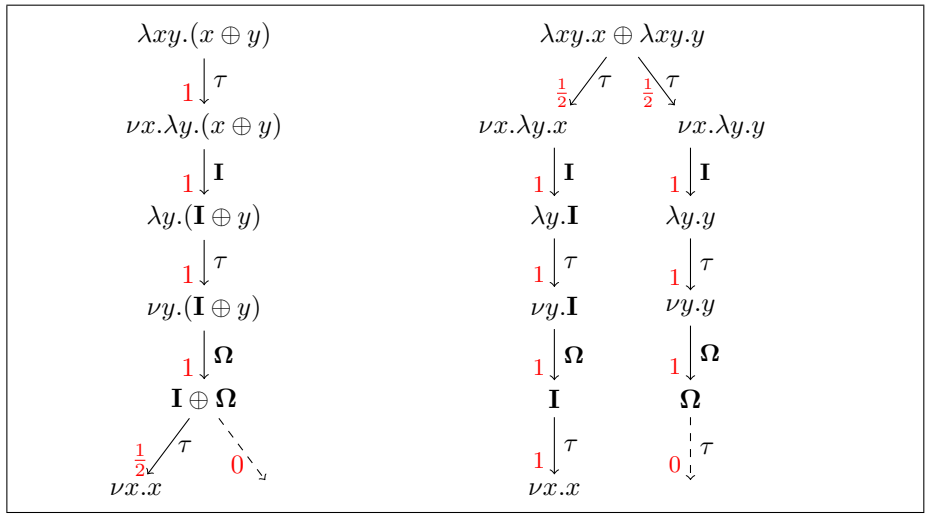

FIGURE 2.7: The test $t = \tau.(\mathbf{I}.\tau.\mathbf{\Omega}.\tau.\omega, \mathbf{I}.\tau.\mathbf{\Omega}.\tau.\omega)$ over the terms of Example 2.53.

We see that terms, as states of a labelled Markov chain, can be discriminated by tests. Thus, a testing language induces a new relation on the set of terms such that two terms are related if the success probability of every test is the same for both terms. We formally define this relation as follows.

Definition 2.54 (Testing equivalence)**.** *Let $(\mathcal{S}, \mathcal{L}, \mathcal{P})$ be a labelled Markov chain. Two states $s, s' \in \mathcal{S}$ are* testing equivalent *if and only if for every test $t \in \mathcal{T}_{(\mathcal{S},\mathcal{L},\mathcal{P})}$, we have that $P_t(s) = P_t(s')$.*

The testing equivalence and bisimilarity over some labelled Markov chain coincide. The same result has been proved for labelled Markov processes in [166]. In [47] these results have been adapted to labelled Markov chains.

Theorem 2.55 ([47])**.** *Let $(\mathcal{S}, \mathcal{L}, \mathcal{P})$ be a labelled Markov chain. Then $s, s' \in \mathcal{S}$ are bisimilar if and only if $P_t(s) = P_t(s')$ for every test $t \in \mathcal{T}_{(\mathcal{S},\mathcal{L},\mathcal{P})}$.*

In the case of inequalities the previous theorem does not hold. In [166], the authors gave examples of states s and s' in labelled Markov process such that $P_t(s) \leq P_t(s')$ and the states s and s' are not similar.

In order to prove that context equivalence implies bisimilarity, it is enough to prove that context equivalence implies testing equivalence, since bisimilarity and testing equivalence coincide by Theorem 2.55. We prove the implication between context equivalence and testing equivalence by proving that for every test t related to the $\Lambda_{\oplus,\mathsf{let}}$-Markov chain, there is a context C_t in $\Lambda_{\oplus,\mathsf{let}}$ such that for every closed term M the success probability of the test t on the term M is equal to the convergence probability of the term $C_t[M]$, i.e. $P_t(M) = \sum[\![C_t[M]]\!]$.

Lemma 2.56. *For every test $t \in \mathcal{T}_{\Lambda_{\oplus,\mathsf{let}}}$, there are contexts $C_t, D_t \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\emptyset;\emptyset)}$ such that for every term $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and value $V = \lambda x.M \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ it holds that:*

$$P_t(M) = \sum[\![C_t[M]]\!] \quad \text{and} \quad P_t(\widetilde{V}) = \sum[\![D_t[V]]\!],$$

where $\widetilde{V}$ denotes the distinguished value $\nu x.M \in \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}$.

Proof. The proof proceeds by induction on the structure of test t.

- Let $t = \omega$. By Definition 2.52, we have $P_\omega(M) = 1$ for every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $P_\omega(\widetilde{V}) = 1$ for every $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$. We take both context $C_\omega[\cdot]$ and $D_\omega[\cdot]$ to be the context $(\lambda xy.x)[\cdot]$. For every closed term $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, we have

$$\sum[\![C_\omega[M]]\!] = \sum[\![(\lambda xy.x)M]\!] = \sum[\![\lambda y.M]\!] = 1 = P_\omega(M).$$

Further, for every value $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ we have

$$\sum [\![D_\omega[V]]\!] = \sum [\![(\lambda xy.x)V]\!] = \sum [\![\lambda y.V]\!] = 1 = P_\omega(\widetilde{V}).$$

- Let the test t be of the form $a.t'$ for some action a and test t'. By the induction hypothesis we have that there exist contexts $C_{t'} \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\emptyset;\emptyset)}$ and $D_{t'} \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\emptyset;\emptyset)}$ such that $P_{t'}(M) = \sum [\![C_{t'}[M]]\!]$ for every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $P_{t'}(\widetilde{V}) = \sum [\![D_{t'}[V]]\!]$ for every $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$. In $\Lambda_{\oplus,\mathsf{let}}$-Markov chain an action is either a closed term or τ action, so we distinguish two cases depending on the action a.

  - If $a = \tau$, we have $P_{\tau.t'}(\widetilde{V}) = 0$ for any value $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ by Definition 2.23 and Definition 2.52. For the context $D_{\tau.t'} = \Omega[\cdot]$, we have $P_{\tau.t'}(\widetilde{V}) = 0 = \sum [\![D_{\tau.t'}[V]]\!]$, for every $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$. By Definition 2.23 and the induction hypothesis we have that for every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ the following holds:

$$\begin{aligned} P_{\tau.t'}(M) &= \sum_{\widetilde{V} \in \mathsf{V}\Lambda^{\emptyset}_{\oplus,\mathsf{let}}} P(M,\tau,\widetilde{V}) P_{t'}(\widetilde{V}) \\ &= \sum_{V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}} [\![M]\!](V) \cdot \sum [\![D_{t'}[V]]\!]. \end{aligned}$$

    We define the context $C_{\tau.t'}$ as $C_{\tau.t'} = (\mathsf{let}\ y = [\cdot]\ \mathsf{in}\ D_{t'}[y])$. From the definition of semantics it follows that for any closed term $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ we have

$$\begin{aligned} \sum [\![C_{\tau.t'}[M]]\!] &= \sum [\![\mathsf{let}\ y = M\ \mathsf{in}\ D_{t'}[y]]\!] \\ &= \sum_{V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}} [\![M]\!](V) \cdot \sum [\![D_{t'}[V]]\!]. \end{aligned}$$

    We conclude $P_{\tau.t'}(M) = \sum [\![C_{\tau.t'}[M]]\!]$.

  - If $a = L$ for some $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, then $P_{L.t'}(M) = 0$ for any term $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, by Definition 2.23 and Definition 2.52. For the context $C_{L.t'} = \Omega[\cdot]$, we have $P_{L.t'}(M) = 0 = \sum [\![C_{L.t'}[M]]\!]$. For a value $V = \lambda x.N$ ($\widetilde{V} = \nu x.N$) and every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$, by Definition 2.23,

Definition 2.52 and the induction hypothesis we have

$$\begin{aligned} P_{L.t'}(\widetilde{V}) &= \sum_{N' \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}} P(\widetilde{V}, L, N') P_{t'}(N') \\ &= P(\nu x.N, L, N\{L/x\}) \cdot P_{t'}(N\{L/x\}) \\ &= 1 \cdot P_{t'}(N\{L/x\}) = \sum [\![C_{t'}[N\{L/x\}]]\!] \end{aligned}$$

By Proposition 2.11 we know that the term $N\{L/x\}$ has the same semantics as the term $(\lambda x.N)L$. Then terms $N\{L/x\}$ and $(\lambda x.N)L$ are bisimilar by Proposition 2.26 and as a consequence they are also context equivalent (Theorem 2.50). Thus, $\sum [\![C[N\{L/x\}]]\!] = \sum [\![C[(\lambda x.N)L]]\!]$ holds for every context C. We conclude

$$P_{L.t'}(\widetilde{V}) = \sum [\![C_{t'}[N\{L/x\}]]\!] = \sum [\![C_{t'}[(\lambda x.N)L]]\!] = \sum [\![C_{t'}[VL]]\!].$$

For the context $D_{L.t'} = C_{t'}[[\cdot]L]$, we have that $\sum [\![D_{L.t'}[V]]\!] = \sum [\![C_{t'}[VL]]\!]$ holds for any value $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$. Thus, we conclude $P_{L.t'}(\widetilde{V}) = \sum [\![D_{L.t'}[V]]\!]$.

- If the test t is of the form (t', t''), then there exist contexts $C_{t'}$,$D_{t'}$, $C_{t''}$, $D_{t''}$ from $\mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\emptyset;\emptyset)}$ such that for every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ and $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ the following holds:

$$\begin{gathered} P_{t'}(M) = \sum [\![C_{t'}[M]]\!], \quad P_{t'}(\widetilde{V}) = \sum [\![D_{t'}[V]]\!], \\ P_{t''}(M) = \sum [\![C_{t''}[M]]\!] \quad \text{and} \quad P_{t''}(\widetilde{V}) = \sum [\![D_{t''}[V]]\!]. \end{gathered}$$

For every $M \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$ we have

$$P_{(t',t'')}(M) = P_{t'}(M) \cdot P_{t''}(M) = \sum [\![C_{t'}[M]]\!] \cdot \sum [\![C_{t''}[M]]\!],$$

by Definition 2.52. We define the context $C_{(t',t'')}$ as follows

$$C_{(t',t'')} = (\lambda y.(\mathsf{let}\ z_1 = C_{t'}[y]\ \mathsf{in}\ (\mathsf{let}\ z_2 = C_{t''}[y]\ \mathsf{in}\ I)))[\cdot] \tag{2.11}$$

From the definition of semantics we have $\sum [\![C_{(t',t'')}[M]]\!] = \sum [\![C_{t'}[M]]\!] \cdot \sum [\![C_{t''}[M]]\!]$. Thus, $P_{(t',t'')}(M) = \sum [\![C_{(t',t'')}[M]]\!]$.

Similarly, for every value $V \in \mathcal{V}^{\emptyset}_{\oplus,\mathsf{let}}$ we have

$$P_{(t',t'')}(\widetilde{V}) = P_{t'}(\widetilde{V}) \cdot P_{t''}(\widetilde{V}) = \sum [\![D_{t'}[V]]\!] \cdot \sum [\![D_{t''}[V]]\!],$$

so we define $D_{(t',t')} = (\lambda y.(\mathsf{let}\ z_1 = D_{t'}[y]\ \mathsf{in}\ (\mathsf{let}\ z_2 = D_{t''}[y]\ \mathsf{in}\ I)))[\cdot]$ and obtain $P_{(t',t'')}(\widetilde{V}) = \sum[\![D_{(t',t'')}[V]]\!]$.

This concludes the proof. □

Proposition 2.57. *Let $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. If $M \leq N$, then $P_t(M) \leq P_t(N)$, for every test t.*

Proof. Let M and N be closed terms such that $\emptyset \vdash M \leq N$. By the definition of the context preorder, we have $\sum[\![C[M]]\!] \leq \sum[\![C[N]]\!]$, for every context $C \in \mathsf{C}\Lambda_{\oplus,\mathsf{let}}{}^{(\emptyset;\emptyset)}$. Next, let t be a test. Lemma 2.56 ensures that for the test t, there is a context C_t, so that $P_t(L) = \sum[\![C_t[L]]\!]$ holds for every $L \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. From the latter and the assumption that terms M and N are in context preorder, we obtain that for each test t, $P_t(M) = \sum[\![C_t[M]]\!] \leq \sum[\![C_t[N]]\!] = P_t(N)$. □

As a direct consequence of Proposition 2.57 we obtain that context equivalence implies testing equivalence.

Theorem 2.58. *Let $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$. Then $M \simeq N$ implies $P_t(M) = P_t(N)$, for every test t.*

Proof. The proof follows directly from Proposition 2.57 and the definition of context equivalence. Recall that $M \simeq N$ is equivalent to $M \leq N$ and $N \leq M$. □

Examples 2.16 and 2.53 gave terms that are distinguished by a test and cannot be distinguished by contexts in call-by-name probabilistic λ-calculus without let-in operator, but can be distinguished by a context in call-by-name probabilistic λ-calculus with let-in operator, so they illustrate that the let-in operator is necessary for achieving full abstraction. In the call-by-name setting the tests where we have to copy a term after an evaluation cannot be characterized by a context and as a consequence the context equivalence and testing equivalence do not coincide. On the other hand, in the call-by-value setting we do not have this problem, since arguments are first evaluated and then passed to a function. We overcome this issue in call-by-name probabilistic λ-calculus by adding the let-in operator. The equivalent terms of the $\Lambda_{\oplus}$-calculus, call-by-name calculus without the let-in operator, which are discriminated in the calculus we propose, are terms where a body of a lambda abstraction contains a probabilistic choice, e.g. a term of the form $\lambda x.M \oplus N$ and a probabilistic choice of lambda abstractions, e.g. a term of the form $(\lambda x.M) \oplus (\lambda x.N)$. Also, the terms which represent passing a probabilistic choice to a function (call-by-name application) and the terms which represent evaluating a probabilistic choice before it is passed to a function (let-in operator) are discriminated.

Finally, we conclude that the three equivalence relations, namely context equivalence, bisimilarity and testing equivalence, coincide. These results are stated in the following theorem and sketched in Figure 2.8.

Theorem 2.59 (Full Abstraction)**.** *For any* $M, N \in \Lambda^{\emptyset}_{\oplus,\mathsf{let}}$*, the following notions are equivalent:*

(context equivalence) $M \simeq N$*,*

(bisimilarity) $M \sim N$*,*

(testing equivalence) $P_t(M) = P_t(N)$ *for all tests* t*.*

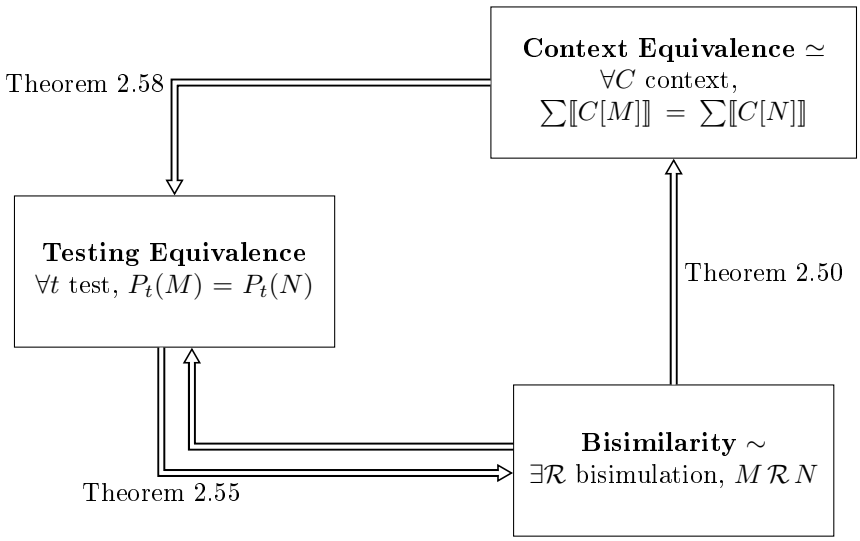

FIGURE 2.8: Sketch of the main results in Chapter 2, which lead to Theorem 2.59.

2.4 Concluding remarks

In this chapter, the $\Lambda_{\oplus,\mathsf{let}}$-calculus, a pure untyped λ-calculus extended with two operators: a probabilistic choice operator $\oplus$ and a let-in operator has been studied. In $\Lambda_{\oplus,\mathsf{let}}$ both call-by-name and call-by-value strategy are implemented. The main evaluation strategy in the calculus is call-by-name strategy and the let-in operator allows for a call-by-value passing policy. The main

result of the chapter is the proof of full abstraction, that is the proof that context equivalence and probabilistic applicative bisimilarity coincide. This chapter is based on [99].

Three equivalence relations on the set of $\Lambda_{\oplus,\mathsf{let}}$-terms are considered: the context equivalence, the probabilistic applicative bisimilarity and the testing equivalence. First, it is proved that probabilistic applicative bisimilarity is a congruence. Consequently, the probabilistic applicative bisimilarity implies context equivalence. Next, the testing language is introduced, which induces a new equivalence relation, the testing equivalence. An important known result is that the testing equivalence and the probabilistic applicative bisimilarity coincide. Finally, proving that for every test there is an equivalent context ensured that the context equivalence implies the testing equivalence. All these results are sketched in Figure 2.8 and they imply that the three equivalence relations coincide.

The presented results confirm a conjecture stated in [35]. In [35], the authors studied the call-by-value probabilistic λ-calculus and proved that the probabilistic applicative bisimilarity coincide with the context equivalence. However, in the call-by-name probabilistic λ-calculus ([47]) the probabilistic applicative bisimilarity is included in the context equivalence, but these two relations do not coincide. So, the authors conjectured that adding a let-in operator to the call-by-name setting will recover full abstraction.

In [99], we have conjectured that the need for the let-in operator was not due to the call-by-name evaluation strategy, but due to the lazyness of the calculus. This conjecture has been proved in [40].

Concerning the inequalities associated with these equivalences: the full abstraction of similarity with respect to the context preorder remains an open question. As it has been discussed, the similarity and testing preorder do not coincide, that is Theorem 2.55 does not hold in the case of inequalities. Thus, we cannot use the same method as in the case of bisimilarity and context equivalence.

There are few directions for the future work.

- Since the method used for proving the full abstraction of the probabilistic applicative bisimilarity with respect to the context equivalence can not be used if we consider the inequalities associated with these equivalences, it is necessary to explore other options for dealing with inequalities.
- Another interesting research path is studying the let-in operator in call-by-name languages with effects other than the probabilistic one such as the non-determinism.
- In this chapter, a non-typed calculus is considered. Similar questions should be addressed in typed languages, as well.

Chapter 3

Kripke-style semantics for full simply typed calculus

In this chapter, we study the simply typed combinatory logic extended with product types and sum types, called *the full simply typed combinatory logic* $CL^{\rightarrow,\times,+}$ ([67, 83, 123, 138]), and present its Kripke-style semantics, which has been introduced in [61, 93, 94]. We prove that the logic $CL^{\rightarrow,\times,+}$ is sound and complete with respect with the proposed semantics.

Combinatory logic, untyped and typed, has a wide range of applications in developing fields, e.g. program synthesis ([53]), machine learning ([115]), artificial intelligence ([60]), cognitive representation ([52]), natural language processing ([162]), physics ([164]). Consequently, it has been the object of many studies, e.g. [17, 19, 20, 25, 53, 54, 161].

Typed calculi have proved to be related to different logics via the Curry-Howard correspondence ([83]), also known as formulae-as-types, proofs-as-terms, proofs-as-programs correspondence. The Curry-Howard correspondence is a correspondence between type systems for models of computation (λ-calculus and combinatory logic) and formal proof calculi. It gives a relationship between types in models of computation and formulas in logics, in the following way: the set of formulas provable in logic coincides with the set of types inhabited by terms of the corresponding calculus. Another relationship given by the Curry-Howard correspondence is the one between terms and proofs, a term which inhabits a type σ in the calculus is actually a proof of formula σ in the logic. The analogy between types and formulas of logic was first discovered by Curry in 1934 ([38]) when he noted that types of the combinators correspond to the axioms of intuitionistic implicational logic. More precisely, Curry observed that the fragment of Hilbert-style deduction system coincides

with the type system of combinatory logic ([39]). He noted that there is the correspondence between the natural deduction system for intuitionistic implicational logic and simply typed λ-calculus, which was later formulated by Howard in [83].

The full simply typed combinatory logic presented in this chapter is related to the intuitionistic propositional logic with all connectives via the Curry-Howard correspondence. The analogy of the full intuitionistic propositional logic (with all connectives) and the simply typed λ-calculus extended with product types and sum types has been established in [83].

Different extensions of the simply typed calculus have been studied ([34, 50, 153]). In [34], Cosmo and Kesner study the typed λ-calculus with functional types, product types, terminal object, sum types and recursion and they provide a rewriting system which has proved to be confluent. In [50], de Vrijer shows that with surjective pairing axioms the extension of the extensional λ-calculus is conservative. In [153], Scherer studies the $\beta\eta$-equivalence of terms in the full simply typed λ-calculus with atoms, functions, pairs, the unit type, sums and the empty type and shows that this equivalence is decidable, it coincides with the context equivalence and the finite model property holds. We may notice that all these studies have considered the computational part of the calculus, namely its reduction relation and the induced equational theory. To the best of our knowledge, this chapter provides the first result on the completeness of the type assignment system.

The semantics we propose in this chapter is a Kripke-style semantics. Kripke semantics has been introduced by Kripke in 1950s as a semantics of modal logic ([108]). Later, it was adapted to the intuitionistic logic ([110]) and other non-classical logics. Kripke-style semantics has also been employed as a semantics of models of computation ([5, 33, 59, 124]). Inspired by the work of Mitchell and Moggi in [124] we have introduced a Kripke-style semantics of the full simply typed λ-calculus and combinatory logic in [94].

Contributions of the chapter

- We present a novel Kripke-style semantics of the full simply typed combinatory logic.
- The Kripke-style semantics that we present have been introduced in [94], however the calculus considered in [94] is not the same as the one we study in this chapter. The similarities and differences of the results presented in [94] and the result of this chapter are discussed in 3.2.
- The main results of the chapter are the soundness and completeness of the full simply typed combinatory logic with respect to the proposed semantics.

Overview of the chapter We start by introducing the full simply typed combinatory logic, $CL^{\rightarrow,\times,+}$, in 3.1. A Kripke-style semantics of $CL^{\rightarrow,\times,+}$ is introduced in 3.2. Section 3.3 presents the main results of the chapter, the proof of soundness and completeness of $CL^{\rightarrow,\times,+}$ with respect to the proposed Kripke-style semantics. Section 3.4 concludes the chapter.

3.1 Full simply typed combinatory logic

In this section, we formally introduce the full simply typed combinatory logic $CL^{\rightarrow,\times,+}$, which is the simply typed combinatory logic extended with product types, sum types, the unit type and the empty type. We start by defining the language of $CL^{\rightarrow,\times,+}$. The set of all terms is built up from the set of term variables $X = \{x, y, z, \ldots, x_1, \ldots\}$ and the set of term constants $\{\mathsf{K}, \mathsf{S}, \mathsf{P}_1, \mathsf{P}_2, \mathsf{P}, \mathsf{I}_1, \mathsf{I}_2, \mathsf{C}, \mathsf{Z}, \mathsf{U}\}$. Terms are expressions generated by the following grammar:

$$\boxed{\begin{array}{c} M ::= x \mid \mathsf{K} \mid \mathsf{S} \mid \mathsf{P}_1 \mid \mathsf{P}_2 \mid \mathsf{P} \mid \mathsf{I}_1 \mid \mathsf{I}_2 \mid \mathsf{C} \mid \mathsf{Z} \mid \mathsf{U} \mid MM \\ \mid \pi_1(M) \mid \pi_2(M) \mid \mathrm{in}_1(M) \mid \mathrm{in}_2(M) \mid \langle\rangle \end{array}} \tag{3.1}$$

The set of all terms is denoted by $CL^{\rightarrow,\times,+}$ and we let $M, N, \ldots, M_1, \ldots$ range over $CL^{\rightarrow,\times,+}$. By $FV(M)$ we denote the set of variables that occur in the term M. The substitution of N for the occurrences of x in M is denoted by $M\{N/x\}$. We write $M \equiv N$ for syntactic identity. We are interested in typed terms, so we introduce the set of types, which is built up from a countable set of type variables $V_{\mathsf{Type}} = \{a, b, c, ..., a_1, \ldots\}$ and the set of type constants $\{0, 1\}$ using three type constructors: $\rightarrow$ (functional type), $\times$ (product type) and $+$ (sum type). The set of all types, denoted by Types, is generated by the following grammar:

$$\boxed{\sigma ::= a \mid \sigma \rightarrow \sigma \mid \sigma \times \sigma \mid \sigma + \sigma \mid 0 \mid 1} \tag{3.2}$$

We let $\sigma, \tau, \ldots, \sigma_1, \ldots$ range over Types.

Following [123], we formalize the typing system. First, some auxiliary notions such as *statements*, *declarations* and *bases* are introduced.

Definition 3.1.

(i) *A* (typed) statement *is an expression of the form* $M : \sigma$*, where* $M \in CL^{\rightarrow,\times,+}$ *and* $\sigma \in \mathsf{Types}$*. The term* M *is the* subject *and the type* σ *is the* predicate *of the statement.*

(ii) *A* declaration *is a statement of the form* $x : \sigma$*, i.e. a statement with a term variable as subject.*

(iii) *A set of declarations with distinct variables as subjects is called* a basis *(*context*).*

(iv) *For a basis* $\Gamma = \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\}$, the domain *of* Γ *is the set* $dom(\Gamma) = \{x_1, \ldots, x_n\}$.

(v) *For a basis* $\Gamma = \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\}$ *the codomain of* Γ *is the set* $|\Gamma| = \{\sigma_1, \ldots, \sigma_n\}$.

One of the properties that we want the typing system to satisfy is to type equal terms with the same type. More precisely, whenever a type σ is assigned to a term M and terms M and N are equal, the type σ should also be assigned to the term N. For this reason, we first define the equality of terms.

Combinatory logic is a model of computation and the computational aspect is modeled by the operational semantics which is given by the reduction relation. The one-step reduction is defined with the following contraction rules

$$
\begin{array}{lcl}
(\mathsf{K}M)N & \to & M \\
((\mathsf{S}M)N)L & \to & (ML)(NL) \\
\mathsf{P}_1 M & \to & \pi_1(M) \\
\mathsf{P}_2 M & \to & \pi_2(M) \\
\mathsf{P}_1((\mathsf{P}M)N) & \to & M \\
\mathsf{P}_2((\mathsf{P}M)N) & \to & N \\
(\mathsf{P}(\mathsf{P}_1 M))(\mathsf{P}_2 M) & \to & M \\
\mathsf{I}_1 M & \to & \mathrm{in}_1(M) \\
\mathsf{I}_2 M & \to & \mathrm{in}_2(M) \\
((\mathsf{C}F)G)(\mathsf{I}_1 M) & \to & FM \\
((\mathsf{C}F)G)(\mathsf{I}_2 M) & \to & GM \\
((\mathsf{C}(\mathsf{S}(\mathsf{K}F)\mathsf{I}_1))(\mathsf{S}(\mathsf{K}F)\mathsf{I}_2))M & \to & FM \\
M & \to & \langle\rangle, \quad \text{if } \Gamma \vdash M : 1 \\
M\{N/x\} & \to & \mathsf{Z}N, \quad \text{if } \Gamma \vdash N : 0, \\
 & & \quad \text{and } \Gamma, x : 0 \vdash M : \sigma.
\end{array}
$$

Notice that the reduction relation depends on the type assignment system. In the last two contraction rules, the reduction depends on a type assigned to a term. A precise formulation of the rule $M \to \langle\rangle$ is:

$$\text{If } \Gamma \vdash M : 1 \text{ for some basis } \Gamma, \text{ then } M \to_\Gamma \langle\rangle.$$

The reduction $\rightarrow_\Gamma$ will be called *reduction with respect to the basis* Γ. The first twelve contraction rules hold for every Γ. The subscript Γ can be omitted if there is no ambiguity from the context. Reflexive, symmetric, transitive and contextual closure of the reduction relation $\rightarrow_\Gamma$ is denoted by $=_\Gamma$.

We have discussed that the equality will be used in the definition of the type assignment system and the typing derivations will be used in the definition of the equality, thus these two relations have to be defined simultaneously.

Definition 3.2. *The equivalence relation* $=_\Gamma$ *and the type assignment system for* $CL^{\rightarrow,\times,+}$ *are defined by the axioms and rules in Figure 3.1 and Figure 3.2, respectively.*

We briefly discuss the axioms and rules in Figure 3.1, which define the equational theory $\mathcal{EQ}_{FCL}$.

- The axioms $(1)-(12)$ and rules $(13)-(14)$ correspond to the contraction rules that define the one-step reduction.

- The reflexivity, transitivity and symmetry of the relation $=_\Gamma$ is ensured by the axiom (15), and rules (16) and (17), respectively.

- The rules $(18)-(23)$ guarantee that the relation is closed under contexts.

- The extensionality of the equational theory is established by the rule (24).

In Figure 3.2, the typing axioms and rules are given. Notice that the type assignment system for $CL^{\rightarrow,\times,+}$ consists of:

- axioms: (Axiom $\in$), (Axiom K), (Axiom S), (Axiom P_1), (Axiom P_2), (Axiom P), (Axiom I_1), (Axiom I_2), (Axiom C), (Axiom Z), (Axiom U) and (Axiom 1-intro),

- rules with one premise: ($\times$ elim1), ($\times$ elim2), ($+$ intro1), ($+$ intro2), and

- rules with two premises: ($\rightarrow$ elim), (Eq).

$$\mathsf{K}MN =_\Gamma M \quad (1) \qquad \mathsf{S}MNL =_\Gamma (ML)(NL) \quad (2)$$

$$\mathsf{P}_1 M =_\Gamma \pi_1(M) \quad (3) \qquad \mathsf{P}_2 M =_\Gamma \pi_2(M) \quad (4)$$

$$\mathsf{P}_1(PMN) =_\Gamma M \quad (5) \qquad \mathsf{P}_2(\mathsf{P}MN) =_\Gamma N \quad (6)$$

$$\mathsf{P}(\mathsf{P}_1 M)(\mathsf{P}_2 M) =_\Gamma M \quad (7) \qquad \mathsf{I}_1 M =_\Gamma \mathrm{in}_1(M) \quad (8)$$

$$\mathsf{I}_2 M =_\Gamma \mathrm{in}_2(M) \quad (9) \qquad ((\mathsf{C}F)G)(\mathsf{I}_1 M) =_\Gamma FM \quad (10)$$

$$((\mathsf{C}F)G)(\mathsf{I}_2 M) =_\Gamma GM \quad (11) \qquad ((\mathsf{C}(\mathsf{S}(\mathsf{K}F)\mathsf{I}_1))(\mathsf{S}(\mathsf{K}F)\mathsf{I}_2))M =_\Gamma FM \quad (12)$$

$$\frac{\Gamma \vdash M : 1}{M =_\Gamma \langle\rangle} \ (13) \qquad \frac{\Gamma \vdash N : 0 \qquad \Gamma, x : 0 \vdash M : \sigma}{M\{N/x\} =_\Gamma \mathsf{Z}N} \ (14)$$

$$M =_\Gamma M \quad (15) \qquad \frac{M =_\Gamma N \qquad N =_\Gamma L}{M =_\Gamma L} \ (16)$$

$$\frac{M =_\Gamma M'}{M' =_\Gamma M} \ (17) \qquad \frac{M =_\Gamma M'}{MN =_\Gamma M'N} \ (18)$$

$$\frac{M =_\Gamma M'}{NM =_\Gamma NM'} \ (19) \qquad \frac{M =_\Gamma M'}{\pi_1(M) =_\Gamma \pi_1(M')} \ (20)$$

$$\frac{M =_\Gamma M'}{\pi_2(M) =_\Gamma \pi_2(M')} \ (21) \qquad \frac{M =_\Gamma M'}{\mathrm{in}_1(M) =_\Gamma \mathrm{in}_1(M')} \ (22)$$

$$\frac{M =_\Gamma M'}{\mathrm{in}_2(M) =_\Gamma \mathrm{in}_2(M')} \ (23) \qquad \frac{Mx =_\Gamma Nx \qquad x \notin FV(M) \cup FV(N)}{M =_\Gamma N} \ (24)$$

FIGURE 3.1: Reflexive, symmetric, transitive and contextual closure of reduction relation $\to_\Gamma$

We briefly discuss the axioms and rules in Figure 3.2.

- (Axiom $\in$) ensured that every variable that belong to the domain of the basis is typed which the corresponding type in the basis.
- (Axiom K), (Axiom S), (Axiom P_1), (Axiom P_2), (Axiom P), (Axiom I_1), (Axiom I_2), (Axiom C), (Axiom Z) and (Axiom U) assign types to the term constants.
- ($\to$ elim) is the rule for typing an application.

- In the rules ($\times$ elim1) and ($\times$ elim 2) the type is assigned to the first and second projection, respectively.
- The rules (+ intro1) and (+ intro2) give the typing derivation for the left and right injection, respectively.
- The empty pair is typed by the axiom (Axiom 1-intro).
- The rule (Eq) ensures that equal terms inhabit the same type.

$$\Gamma, x : \sigma \vdash x : \sigma \quad \text{(Axiom } \in\text{)} \qquad \Gamma \vdash \mathsf{K} : \sigma \to (\tau \to \sigma) \quad \text{(Axiom } K\text{)}$$

$$\Gamma \vdash \mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho)) \quad \text{(Axiom } S\text{)}$$

$$\Gamma \vdash \mathsf{P}_1 : (\sigma \times \tau) \to \sigma \quad \text{(Axiom } P_1\text{)} \qquad \Gamma \vdash \mathsf{P}_2 : (\sigma \times \tau) \to \tau \quad \text{(Axiom } P_2\text{)}$$

$$\Gamma \vdash \mathsf{P} : \sigma \to (\tau \to (\sigma \times \tau)) \quad \text{(Axiom } P\text{)}$$

$$\Gamma \vdash \mathsf{I}_1 : \sigma \to (\sigma + \tau) \quad \text{(Axiom } I_1\text{)} \qquad \Gamma \vdash \mathsf{I}_2 : \tau \to (\sigma + \tau) \quad \text{(Axiom } I_2\text{)}$$

$$\Gamma \vdash \mathsf{C} : (\sigma \to \rho) \to ((\tau \to \rho) \to ((\sigma + \tau) \to \rho)) \quad \text{(Axiom } C\text{)}$$

$$\Gamma \vdash \mathsf{Z} : 0 \to \sigma \quad \text{(Axiom } Z\text{)} \qquad \Gamma \vdash \mathsf{U} : 1 \quad \text{(Axiom } U\text{)}$$

$$\frac{\Gamma \vdash M : \sigma \to \tau \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau} \ (\to \text{elim})$$

$$\frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_1(M) : \sigma} \ (\times \text{elim1}) \qquad \frac{\Gamma \vdash M : \sigma \times \tau}{\Gamma \vdash \pi_2(M) : \tau} \ (\times \text{elim2})$$

$$\frac{\Gamma \vdash M : \sigma}{\Gamma \vdash \text{in}_1(M) : \sigma + \tau} \ (+ \text{intro1}) \qquad \frac{\Gamma \vdash M : \tau}{\Gamma \vdash \text{in}_2(M) : \sigma + \tau} \ (+ \text{intro2})$$

$$\Gamma \vdash \langle\rangle : 1 \quad \text{(Axiom 1-intro)} \qquad \frac{\Gamma \vdash M : \sigma \qquad M =_\Gamma N}{\Gamma \vdash N : \sigma} \ \text{(Eq)}$$

FIGURE 3.2: Type assignment system for $CL^{\to,\times,+}$

Remark 3.3. *Let us consider the terms* $\mathsf{K}xy$ *and* x*. They are equal with respect to any basis* Γ *by the axiom* (1) *in Figure 3.1. We take the basis* Γ *to be* $\Gamma = \{x : \sigma\}$*. In the basis* Γ*, the variable* x *inhabits the type* σ*. From* $x =_\Gamma \mathsf{K}xy$ *we obtain* $\Gamma \vdash \mathsf{K}xy : \sigma$ *by the rule (Eq), although the the variable* y *is not typable in the basis* Γ*.*

Terms M and N are equal with respect to the basis Γ, $M =_\Gamma N$, if we can derive $M =_\Gamma N$ by the rules in Figure 3.1. We say that a term M can be typed with a type σ in a basis Γ, or that the type σ is inhabited by the term M in the basis Γ, denoted by $\Gamma \vdash M : \sigma$, if $\Gamma \vdash M : \sigma$ can be derived by the rules in Figure 3.2.

The following example illustrates some typing derivations.

Example 3.4. *Let us consider the terms* $\mathsf{P}_1((\mathsf{PK})\mathsf{K})$ *and* K. *These terms are equal with respect to any basis by the axiom* (5) *in Figure 3.1. Further, we have* $\Gamma \vdash \mathsf{K} : \sigma \to (\tau \to \sigma)$ *by the rule (Axiom K) and we obtain* $\Gamma \vdash \mathsf{P}_1((\mathsf{PK})\mathsf{K}) : \sigma \to (\tau \to \sigma)$ *by the rule (Eq). However, it can be proved that the term* $\mathsf{P}_1((\mathsf{PK})\mathsf{K})$ *inhabits the type* $\sigma \to (\tau \to \sigma)$ *without using the rule (Eq). Let* $\tau_1 = ((\sigma \to (\tau \to \sigma)) \times (\sigma \to (\tau \to \sigma))) \to (\sigma \to (\tau \to \sigma))$, $\tau_2 = (\sigma \to (\tau \to \sigma)) \to ((\sigma \to (\tau \to \sigma)) \to ((\sigma \to (\tau \to \sigma)) \times (\sigma \to (\tau \to \sigma))))$, $\tau_3 = (\sigma \to (\tau \to \sigma)) \to ((\sigma \to (\tau \to \sigma)) \times (\sigma \to (\tau \to \sigma)))$ *and* $\tau_4 = (\sigma \to (\tau \to \sigma)) \times (\sigma \to (\tau \to \sigma))$. *Then we have the following derivation:*

$$\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \mathsf{P} : \tau_2} \quad \dfrac{}{\Gamma \vdash \mathsf{K} : \sigma \to (\tau \to \sigma)}}{\Gamma \vdash \mathsf{PK} : \tau_3} \quad \dfrac{}{\Gamma \vdash \mathsf{K} : \sigma \to (\tau \to \sigma)}}{\dfrac{\dfrac{}{\Gamma \vdash \mathsf{P}_1 : \tau_1} \quad \Gamma \vdash \mathsf{PKK} : \tau_4}{\Gamma \vdash \mathsf{P}_1(\mathsf{PKK}) : \sigma \to (\tau \to \sigma)}}$$

Thus, the rule (Eq) is not necessary for proving that terms $\mathsf{P}_1(\mathsf{PKK})$ *and* K *inhabit the same type in every* Γ. *Nevertheless, this is not true for all terms that are equal and all bases as we have discussed it in Remark 3.3.*

Definition 3.5. *A basis* Γ *is* consistent *if and only if there does not exist a term* M *such that* $\Gamma \vdash M : 0$, *otherwise* Γ *is* inconsistent.

From Definition 3.5 it follows that if a basis Γ is consistent, then there exists a type σ that cannot be inhabited in the basis Γ.

As we have already discussed, the full simply typed combinatory logic is related to the full intuitionistic propositional logic with all the connectives via the Curry-Howard correspondence ([83]).

Theorem 3.6 (Curry-Howard correspondence)([83]). *For a basis* Γ *and a type* σ, *there exists a term* M *such that* $\Gamma \vdash M : \sigma$ *if and only if* $|\Gamma| \vdash \sigma$, *that is* σ *is derivable from the set* $|\Gamma|$ *in natural deduction system for intuitionistic propositional logic.*

Now, we compare a notion of a consistent set in the intuitionistic propositional logic with all the connectives and the full simply typed combinatory logic. In the intuitionistic propositional logic a set of formulas is consistent if

there is a formula which cannot be derived from the set, whereas it is inconsistent if and only if every formula can be derived from it.

Let Γ be an inconsistent basis. Then, $\Gamma \vdash M : 0$, for some $M \in CL$, by Definition 3.5. Further, by (Axiom Z) and ($\rightarrow$ elim) we have that $\Gamma \vdash \mathsf{Z}M : \sigma$, for any type σ. We conclude that $|\Gamma| \vdash \sigma$ by Theorem 3.6. Therefore, the set $|\Gamma|$ is also inconsistent.

The derivation length of $M =_\Gamma N$ and $\Gamma \vdash M : \sigma$ is the number of applied axioms and rules from Figure 3.1 and 3.2.

A standard property of the typing system is that whenever a type σ is assigned to a term M in some basis Γ, it can also be assigned in a basis which is a superset of Γ. Similarly, we will show that equality of terms is also monotone with respect to the preorder, that is whenever two terms are equal with respect to some basis Γ, they will be equal with respect to a superset of Γ. The equality and the type assignment system are defined simultaneously, thus we need to consider both properties within one statement.

Proposition 3.7. *If* $M =_{\Gamma_1} N$, $\Gamma_2 \vdash P : \sigma$, $\Gamma_1 \subseteq \Gamma_1'$ *and* $\Gamma_2 \subseteq \Gamma_2'$, *then* $M =_{\Gamma_1'} N$ *and* $\Gamma_2' \vdash P : \sigma$.

Proof. The proof is by induction on the sum $k = n + m$ of the length n of the derivation $M =_{\Gamma_1} N$ and the length m of the derivation of $\Gamma_2 \vdash P : \sigma$.

The base case is when $k = 2$, i.e. when both derivations are obtained from axioms ($n = 1$ and $m = 1$). If $n = 1$, then $M =_{\Gamma_1} N$ is obtained by one of the axioms $(1) - (12)$ and (15) from Figure 3.1. As all equalities in these axioms hold for every basis Γ, they will also hold for the basis Γ_1', i.e. $M =_{\Gamma_1'} N$. If $m = 1$, then $\Gamma_2 \vdash P : \sigma$ is obtained by applying an axiom of Figure 3.2. Again, typing statements in all axioms hold for every basis Γ, so they will also hold for the basis Γ_2', i.e. $\Gamma_2' \vdash P : \sigma$.

Let us assume that the statement holds for every $i < k$, $k \geq 3$:

$$\begin{aligned} &\text{if } M =_{\Gamma_1} N, \Gamma_2 \vdash P : \sigma, \Gamma_1 \subseteq \Gamma_1', \Gamma_2 \subseteq \Gamma_2', \\ &\text{the length of } M =_{\Gamma_1} N \text{ is } n' \text{ the length of } \Gamma_2 \vdash P : \sigma \text{ is } m' \\ &\text{and } n' + m' = i < k, \text{ then } M =_{\Gamma_1'} N \text{ and } \Gamma_2' \vdash P : \sigma. \end{aligned} \tag{IH}$$

We prove that the statement holds when the sum of the lengths of derivations is equal to k. Let us assume that $M =_{\Gamma_1} N$ can be derived with the derivation length n and $\Gamma_2 \vdash P : \sigma$ with the derivation length m, $n+m = k \geq 3$, $\Gamma_1 \subseteq \Gamma_1'$ and $\Gamma_2 \subseteq \Gamma_2'$. We perform the case analysis on the last rule applied in the derivation of $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash P : \sigma$.

1. First, we consider the last applied rule in the derivation $M =_{\Gamma_1} N$.

Let (13) be the last applied rule. Then the term N is $\langle\rangle$, and $M =_{\Gamma_1} \langle\rangle$ is obtained from $\Gamma_1 \vdash M : 1$, i.e. $\dfrac{\Gamma_1 \vdash M : 1}{M =_{\Gamma_1} \langle\rangle}$ is the last applied rule. The length of the derivation $\Gamma_1 \vdash M : 1$ is $n-1$. Directly from the rules of Figure 3.1 it follows that $\mathsf{K}MN =_{\Gamma_2} M$ and the length of this derivations is 1. Since $n - 1 + 1 = n < n + m = k$, we can apply the induction hypothesis (IH) to $\mathsf{K}MN =_{\Gamma_2} M$ and $\Gamma_1 \vdash M : 1$ and we obtain $\mathsf{K}MN =_{\Gamma_2'} M$ and $\Gamma_1' \vdash M : 1$. From $\Gamma_1' \vdash M : 1$, it follows that $M =_{\Gamma'} \langle\rangle$. We also need to prove $\Gamma_2' \vdash P : \sigma$. We will distinguish three cases performing an analysis on the last applied rule in the derivation of $\Gamma_2 \vdash P : \sigma$.

- If $\Gamma_2 \vdash P : \sigma$ is obtained by an axiom, then we have the base case and for any Γ_2' it holds that $\Gamma_2' \vdash P : \sigma$.

- If the last applied rule is the rule with only one premise, then the derivation is of the form $\dfrac{\Gamma_2 \vdash Q : \tau}{\Gamma_2 \vdash P : \sigma}\,(r)$ for some term Q, type τ and the rule (r) which has only one premise. The length of the derivation $\Gamma_2 \vdash Q : \tau$ is $m - 1$. The length of the derivation of $\mathsf{K}MN =_{\Gamma_1} M$ is 1. Since $m - 1 + 1 = m < n + m = k$, we can apply the induction hypothesis (IH) to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \tau$ and we obtain $\mathsf{K}MN =_{\Gamma_1'} M$ and $\Gamma_2' \vdash Q : \tau$. From the latter, we get $\Gamma_2' \vdash P : \sigma$ by the rule (r).

- If $\Gamma_2 \vdash P : \sigma$ is obtained by one of the rules with two premises, then we have two possibilities. If the last rule is ($\to$ elim), then we have

$$\frac{\Gamma_2 \vdash Q : \tau \to \sigma \qquad \Gamma_2 \vdash R : \tau}{\Gamma_2 \vdash QR : \sigma}$$

  for some terms Q and R ($P \equiv QR$) and type τ. The lengths of $\Gamma_2 \vdash Q : \tau \to \sigma$ and $\Gamma_2 \vdash R : \tau$ are less than $m - 1$. We can apply the induction hypothesis to $KMN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \tau \to \sigma$ and we obtain $\mathsf{K}MN =_{\Gamma_1'} M$ and $\Gamma_2' \vdash Q : \tau \to \sigma$. By applying the induction hypothesis to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash R : \tau$, we obtain $\mathsf{K}MN =_{\Gamma_1'} M$ and $\Gamma_2' \vdash R : \tau$. Since we have $\Gamma_2' \vdash Q : \tau \to \sigma$ and $\Gamma_2' \vdash R : \tau$, we conclude $\Gamma_2' \vdash QR : \sigma$ by the rule ($\to$ intro). The last case is when $\Gamma_2 \vdash P : \sigma$ is obtained by the rule (Eq) and we have

$$\frac{\Gamma_2 \vdash Q : \sigma \qquad Q =_{\Gamma_2} P}{\Gamma_2 \vdash P : \sigma}$$

  where the lengths of $\Gamma_2 \vdash Q : \sigma$ and $Q =_{\Gamma_2} P$ are less than $m - 1$. By applying IH to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \sigma$, we obtain $\mathsf{K}MN =_{\Gamma_1'} M$

and $\Gamma_2' \vdash Q : \sigma$. If we apply IH to $Q =_{\Gamma_2} P$ and $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$, we get $Q =_{\Gamma_2'} P$ and $\Gamma_1' \vdash \mathsf{K} : \sigma \to \tau \to \sigma$. Finally, from $\Gamma_2' \vdash Q : \sigma$ and $Q =_{\Gamma_2'} P$ we obtain $\Gamma_2' \vdash P : \sigma$.

Let (14) be the last applied rule. Then we have that term M is of the form $R\{L/x\}$, term N is of the form $\mathsf{Z}L$ for some terms R and L and the last step in the derivation is

$$\frac{\Gamma_1 \vdash L : 0 \qquad \Gamma_1, x : 0 \vdash R : \sigma}{R\{L/x\} =_{\Gamma_1} \mathsf{Z}L}$$

The derivation lengths of $\Gamma_1 \vdash L : 0$ and $\Gamma_1, x : 0 \vdash R : \sigma$ are less than $n-1$. By applying the induction hypothesis to $\mathsf{KII} =_{\Gamma_2} \mathsf{I}$ and $\Gamma_1 \vdash L : 0$, we obtain $\mathsf{KII} =_{\Gamma_2'} \mathsf{I}$ and $\Gamma_1' \vdash L : 0$. If $x \notin dom(\Gamma_1')$ and $\Gamma_1 \subseteq \Gamma_1'$, then $\Gamma_1, x : 0 \subseteq \Gamma_1', x : 0$. We can apply IH to $\mathsf{KII} =_{\Gamma_2} \mathsf{I}$ and $\Gamma_1, x : 0 \vdash R : \sigma$ and we derive $\mathsf{KII} =_{\Gamma_2'} \mathsf{I}$ and $\Gamma_1', x : 0 \vdash R : \sigma$. The result $R\{L/x\} =_{\Gamma_1'} \mathsf{Z}L$ follows from $\Gamma_1' \vdash L : 0$ and $\Gamma_1', x : 0 \vdash R : \sigma$ by the rule (14). In the case $x \in dom(\Gamma_1')$, we can choose a fresh variable y that occurs in neither terms L and R nor in $dom(\Gamma_1) \cup dom(\Gamma_1')$. It is straightforward to show that if $\Gamma_1, x : 0 \vdash R : \sigma$, then $\Gamma_1, y : 0 \vdash R\{y/x\} : \sigma$ with the equal derivation length. Since $y : 0 \notin \Gamma_1'$, we derive $\Gamma_1', y : 0 \vdash R\{y/x\} : \sigma$ by the same reasoning as in the case $x : 0 \notin \Gamma_1'$. Now, $\Gamma_1' \vdash L : 0$ and $\Gamma_1', y : 0 \vdash R\{y/x\} : \sigma$ implies $R\{y/x\}\{L/y\} =_{\Gamma_1'} \mathsf{Z}L$. By induction on the structure of term R it can be proved that $R\{y/x\}\{L/y\} \equiv R\{L/x\}$, so $R\{L/x\} =_{\Gamma_1'} \mathsf{Z}L$. Again, we have to prove $\Gamma_2 \vdash P : \sigma$ and the proof follows similar reasoning as in the previous case.

Let (16) be the last applied rule. Then we have $\dfrac{M =_{\Gamma_1} L \qquad L =_{\Gamma_1} N}{M =_{\Gamma_1} N}$ where the derivation lengths of $M =_{\Gamma_1} L$ and $L =_{\Gamma_1} N$ are less than $n-1$. By applying the induction hypothesis to $M =_{\Gamma_1} L$ and $\Gamma_2 \vdash P : \sigma$, we get $M =_{\Gamma_1'} L$ and $\Gamma_2' \vdash P : \sigma$. Similarly, we can apply the induction hypothesis to $L =_{\Gamma_1} N$ and $\Gamma_2 \vdash P : \sigma$, which results in $L =_{\Gamma_1'} N$ and $\Gamma_2' \vdash P : \sigma$. From $M =_{\Gamma_1'} L$ and $L =_{\Gamma_1'} N$, we conclude $M =_{\Gamma_1'} N$ by the rule (16).

Let (17) be the last applied rule. Then. the last step in the derivation is $\dfrac{N =_{\Gamma_1} M}{M =_{\Gamma_1} N}$ and the length of the derivation of $N =_{\Gamma_1} M$ is $n-1$. By the induction hypothesis we get $N =_{\Gamma_1'} M$ and $\Gamma_2' \vdash P : \sigma$ and by rule (17) we derive $M =_{\Gamma_1'} N$.

Let (18) be the last applied rule. Then the term M is of the form LR, the term N is of the form $L'R$ and the last step in the derivation is

$$\frac{L =_{\Gamma_1} L'}{LR =_{\Gamma_1} L'R}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$, so we can apply IH to $L =_{\Gamma_1} L'$ and $\Gamma_2 \vdash P : \sigma$ and we get $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$. By the rule (18) we obtain $LR =_{\Gamma_1'} L'R$.

Let (19) be the last applied rule. Then the term M is of the form RL, the term N is of the form RL' and the last step in the derivation is

$$\frac{L =_{\Gamma_1} L'}{RL =_{\Gamma_1} RL'}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$. By the induction hypothesis we get $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$ and by the rule (19) we derive $RL =_{\Gamma_1'} RL'$.

Let (20) be the last applied rule. Then we have

$$\frac{L =_{\Gamma_1} L'}{\pi_1(L) =_{\Gamma_1} \pi_1(L')}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$. By applying the induction hypothesis to $L =_{\Gamma_1} L'$ and $\Gamma_2 \vdash P : \sigma$, we obtain $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$. From $L =_{\Gamma_1'} L'$ we derive $\pi_1(L) =_{\Gamma_1'} \pi_1(L')$ by rule (20).

Let (21) be the last applied rule. Then we have

$$\frac{L =_{\Gamma_1} L'}{\pi_2(L) =_{\Gamma_1} \pi_2(L')}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$. By applying the induction hypothesis to $L =_{\Gamma_1} L'$ and $\Gamma_2 \vdash P : \sigma$, we obtain $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$. From $L =_{\Gamma_1'} L'$ we derive $\pi_2(L) =_{\Gamma_1'} \pi_2(L')$ by rule (21).

Let (22) be the last applied rule. Then the last step in the derivation is

$$\frac{L =_{\Gamma_1} L'}{\mathrm{in}_1(L) =_{\Gamma_1} \mathrm{in}_1(L')}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$. By applying the induction hypothesis to $L =_{\Gamma_1} L'$ and $\Gamma_2 \vdash P : \sigma$, we obtain $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$. From $L =_{\Gamma_1'} L'$ we derive $\mathrm{in}_1(L) =_{\Gamma_1'} \mathrm{in}_1(L')$ by rule (22).

Let (23) be the last applied rule. Then the last step in the derivation is

$$\frac{L =_{\Gamma_1} L'}{\mathrm{in}_2(L) =_{\Gamma_1} \mathrm{in}_2(L')}$$

The length of the derivation $L =_{\Gamma_1} L'$ is $n-1$. By applying the induction hypothesis to $L =_{\Gamma_1} L'$ and $\Gamma_2 \vdash P : \sigma$, we obtain $L =_{\Gamma_1'} L'$ and $\Gamma_2' \vdash P : \sigma$. From $L =_{\Gamma_1'} L'$ we derive $\mathrm{in}_2(L) =_{\Gamma_1'} \mathrm{in}_2(L')$ by rule (23).

2. Next, we consider the last applied rule in the derivation $\Gamma \vdash P : \sigma$.

If the last applied rule is ($\to$ elim), then we have

$$\frac{\Gamma_2 \vdash Q : \tau \to \sigma \qquad \Gamma_2 \vdash R : \tau}{\Gamma_2 \vdash QR : \sigma}$$

and the derivation lengths of $\Gamma_2 \vdash Q : \tau \to \sigma$ and $\Gamma_2 \vdash R : \tau$ are less than $m-1$. By applying the induction hypothesis to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \tau \to \sigma$, we derive $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \tau \to \sigma$. Further, by applying the induction hypothesis to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash R : \tau$, we derive $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash R : \tau$. From $\Gamma_2' \vdash Q : \tau \to \sigma$ and $\Gamma_2' \vdash R : \tau$, it follows that $\Gamma_2' \vdash QR : \sigma$.

If the last applied rule is ($\times$ elim1), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma \times \tau}{\Gamma_2 \vdash \pi_1(Q) : \sigma}$$

The derivation $\Gamma_2 \vdash Q : \sigma \times \tau$ is of length $m-1$. By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma \times \tau$, we get $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \sigma \times \tau$. From $\Gamma_2' \vdash Q : \sigma \times \tau$, we derive $\Gamma_2' \vdash \pi_1(Q) : \sigma$ by the rule ($\times$ elim1).

If the last applied rule is ($\times$ elim2), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma \times \tau}{\Gamma_2 \vdash \pi_2(Q) : \tau}$$

The derivation $\Gamma_2 \vdash Q : \sigma \times \tau$ is of length $m-1$. By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma \times \tau$, we obtain $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \sigma \times \tau$. From $\Gamma_2' \vdash Q : \sigma \times \tau$, we derive $\Gamma_2' \vdash \pi_2(Q) : \tau$ by the rule ($\times$ elim2).

If the last applied rule is ($+$ intro1), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma}{\Gamma_2 \vdash \mathrm{in}_1(Q) : \sigma + \tau}$$

The derivation $\Gamma_2 \vdash Q : \sigma$ is of length $m-1$. By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma$, we obtain $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \sigma$. From $\Gamma_2' \vdash Q : \sigma$, we derive $\Gamma_2' \vdash \mathrm{in}_1(Q) : \sigma + \tau$ by the rule ($+$ intro1).

If the last applied rule is ($+$ intro2), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \tau}{\Gamma_2 \vdash \mathrm{in}_2(Q) : \sigma + \tau}$$

The derivation $\Gamma_2 \vdash Q : \tau$ is of length $m-1$. By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \tau$, we get $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \tau$. From $\Gamma_2' \vdash Q : \tau$, we derive $\Gamma_2' \vdash \mathrm{in}_2(Q) : \sigma + \tau$ by the rule ($+$ intro2).

Finally, we consider the case when the last applied rule is (Eq). Then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma \qquad Q =_{\Gamma_2} P}{\Gamma_2 \vdash P : \sigma}$$

The derivation lengths of $\Gamma_2 \vdash Q : \sigma$ and $Q =_{\Gamma_2} P$ are less than $m-1$. By applying the induction hypothesis to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma$, we get $M =_{\Gamma_1'} N$ and $\Gamma_2' \vdash Q : \sigma$. Further, typing $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$ has the derivation length 1. If we denote the derivation length of $Q =_{\Gamma_2} P$ by j, we have $j - 1 + 1 = j < m - 1 < n + m = k$. So, we can apply the induction hypothesis to $Q =_{\Gamma_2} P$ and $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$ and we get $Q =_{\Gamma_2'} P$ and $\Gamma_1' \vdash \mathsf{K} : \sigma \to \tau \to \sigma$. Finally, from $\Gamma_2' \vdash Q : \sigma$ and $Q =_{\Gamma_2'} P$, we conclude $\Gamma_2' \vdash P : \sigma$ by the rule (Eq).

□

3.2 Kripke-style semantics of $CL^{\to,\times,+}$

In this section, we propose a Kripke-style semantics of $CL^{\to,\times,+}$, which has been introduced in [94]. The motivation for the proposed Kripke-style semantics comes from the work of Mitchell and Moggi [124], where they have introduced the Kripke-style semantics of the simply typed λ-calculus. The similarities and differences of the semantics introduced in [124] and the one we present in this chapter will be pointed out throughout the section.

A disjoint union of sets X and Y will be denoted by $X \uplus Y$, i.e.

$$X \uplus Y = \{\langle 0, x\rangle \mid x \in X\} \cup \{\langle 1, y\rangle \mid y \in Y\}.$$

First, a notion of a Kripke applicative structure is introduced.

Definition 3.8. A Kripke applicative structure $\mathcal{K}$ *for* $CL^{\to,\times,+}$ *is a tuple*

$$\langle W, \preceq, \{D_w\}, \{A_w^\sigma\}, \{App_w\}, \{Proj_{1,w}\}, \{Proj_{2,w}\}, \{Inl_w\}, \{Inr_w\}, \{i_{w,w'}\}\rangle$$

that consists of:

(i) *a set* W *of* possible worlds *partially ordered by* $\preceq$,

(ii) *a family* $\{D_w\} = \{D_w\}_{w \in W}$ *of sets indexed by worlds* w*, where the set* D_w *is referred to as the domain of the world* w,

(iii) *a family* $\{A_w^\sigma\} = \{A_w^\sigma\}_{w \in W, \sigma \in \mathsf{Types}}$ *of sets indexed by types* σ *and worlds* w *that satisfies the following:*

- *for all* $w \in W$*, for all* $\sigma \in \mathsf{Types}$*,* $A_w^\sigma \subseteq D_w$*,* A_w^0 *is empty, i.e.* $A_w^0 = \emptyset$*, and* A_w^1 *has one element, i.e.* $A_w^1 = \{1_w\}$*,* $1_w \in D_w$*,*
- *there exists an injective function* $H : D_w \uplus D_w \to D_w$ *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* H *to the set* $A_w^\sigma \uplus A_w^\tau$ *is* $A_w^{\sigma+\tau}$*,*

- *there exists an injective function* $G : D_w \to D_w \times D_w$ *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* G *to the set* $A_w^{\sigma\times\tau}$ *is* $A_w^\sigma \times A_w^\tau$*,*

(iv) *a family* $\{App_w\} = \{App_w\}_{w\in W}$ *of* application functions $App_w : D_w \times D_w \to D_w$ *indexed by worlds* w *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* App_w *to the set* $A_w^{\sigma\to\tau} \times A_w^\sigma$ *is* A_w^τ*,*

(v) *a family* $\{Proj_{1,w}\} = \{Proj_{1,w}\}_{w\in W}$ *of* first projection functions $Proj_{1,w} : D_w \to D_w$ *indexed by worlds* w *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* $Proj_{1,w}$ *to the set* $A_w^{\sigma\times\tau}$ *is* A_w^σ*,*

(vi) *a family* $\{Proj_{2,w}\} = \{Proj_{2,w}\}_{w\in W}$ *of* second projection functions $Proj_{2,w} : D_w \to D_w$ *indexed by worlds* w *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* $Proj_{2,w}$ *to the set* $A_w^{\sigma\times\tau}$ *is* A_w^τ*,*

(vii) *a family* $\{Inl_w\} = \{Inl_w\}_{w\in W}$ *of* left injection functions $Inl_w : D_w \to D_w$ *indexed by worlds* w *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* Inl_w *to the set* A_w^σ *is* $A_w^{\sigma+\tau}$*,*

(viii) *a family* $\{Inr_w\} = \{Inr_w\}_{w\in W}$ *of* right injection functions $Inr_w : D_w \to D_w$ *indexed by worlds* w *such that for all* $\sigma, \tau \in \mathsf{Types}$*, the codomain of the restriction of the function* Inr_w *to the set* A_w^τ *is* $A_w^{\sigma+\tau}$*,*

(ix) *a family* $\{i_{w,w'}\} = \{i_{w,w'}\}_{w,w'\in W, w\preceq w'}$ *of* transition functions $i_{w,w'} : D_w \to D_{w'}$ *indexed by pairs of worlds* $w \preceq w'$ *such that* $i_{w,w'}$ *is a surjective function, for all* $\sigma \in \mathsf{Types}$*, the codomain of the restriction of the function* $i_{w,w'}$ *to the set* A_w^σ *is* $A_{w'}^\sigma$*, and all transition functions satisfy the following conditions:*

$$i_{w,w} : D_w \to D_w \text{ is the identity} \tag{id}$$

$$i_{w',w''} \circ i_{w,w'} = i_{w,w''} \text{ for all } w \preceq w' \preceq w'' \tag{comp}$$

We also require that the application functions, the projection functions and the injection functions commute with the transition functions in a natural way:

$$(\forall w \in W)(\forall f \in D_w)\ (\forall a \in D_w)\ (\forall w' \in W, w \preceq w')$$

$$i_{w,w'}(App_w(f,a)) = App_{w'}(i_{w,w'}(f), i_{w,w'}(a)) \qquad \text{(comm1)}$$

$$i_{w,w'}(Proj_{1,w}(a)) = Proj_{1,w'}(i_{w,w'}(a)) \qquad \text{(comm2)}$$

$$i_{w,w'}(Proj_{2,w}(a)) = Proj_{2,w'}(i_{w,w'}(a)) \qquad \text{(comm3)}$$

$$i_{w,w'}(Inl_w(a)) = Inl_{w'}(i_{w,w'}(a)) \qquad \text{(comm4)}$$

$$i_{w,w'}(Inr_w(a)) = Inr_{w'}(i_{w,w'}(a)) \qquad \text{(comm5)}$$

We will sometimes omit writing *App* for the application function and write fa instead of $App_w(f,a)$, as in Definition 3.10 below.

Our goal is to define a model such that every term has a unique meaning in the model. For that reason, we consider only Kripke applicative structures which are *extensional* and have special elements in the domain, called *combinators*, as specified in Definition 3.9 and Definition 3.10 below.

Definition 3.9. *A Kripke applicative structure $\mathcal{K}$ as in Definition 3.8 is* extensional *if for all $w \in W$ and $f, g, p, q \in D_w$ the following holds*

- *if $Proj_{1,w}(p) = Proj_{1,w}(q)$ and $Proj_{2,w}(p) = Proj_{2,w}(q)$, then $p = q$*
- *if $(\forall w' \succeq w)(\forall x, y \in D_{w'})$*

$$App_{w'}(i_{w,w'}(f), Inl_{w'}(x)) = App_{w'}(i_{w,w'}(g), Inl_{w'}(x))$$
$$\text{and}$$
$$App_{w'}(i_{w,w'}(f), Inr_{w'}(y)) = App_{w'}(i_{w,w'}(g), Inr_{w'}(y)),$$

then $f = g$.

The second condition of extensionality implies that if a Kripke applicative structure is extensional, then for elements $f, g \in D_w$ holds the following:

if $(\forall w' \geq w)(\forall a \in D_{w'})(App_{w'}(i_{w,w'}(f), a) = App_{w'}(i_{w,w'}(g), a))$, then $f = g$.

This condition was the definition of an extensional applicative structure in [124], where Mitchell and Moggi have considered simply typed λ-calculus. We may notice that the condition ensures the extensionality of application functions App_w in a Kripke applicative structure. In the Kripke-style semantics of simply typed λ-calculus, presented in [124], there are only application functions *App*, that is there is neither projection functions nor injection functions in an applicative structure, so this condition was sufficient to ensure extensionality.

Definition 3.10. *A Kripke applicative structure $\mathcal{K}$ as in Definition 3.8* has combinators *if there exist elements:*

- $\boldsymbol{k}_w \in A_w^{\sigma\to(\tau\to\sigma)}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{s}_w \in A_w^{(\sigma\to(\rho\to\tau))\to((\sigma\to\rho)\to(\sigma\to\tau))}$, *for every* $\sigma,\rho,\tau \in \mathsf{Types}$,
- $\boldsymbol{p}_{1,w} \in A_w^{\sigma\times\tau\to\sigma}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{p}_{2,w} \in A_w^{\sigma\times\tau\to\tau}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{p}_w \in A_w^{\sigma\to(\tau\to(\sigma\times\tau))}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{i}_{1,w} \in A_w^{\sigma\to(\sigma+\tau)}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{i}_{2,w} \in A_w^{\tau\to(\sigma+\tau)}$, *for every* $\sigma,\tau \in \mathsf{Types}$,
- $\boldsymbol{c}_w \in A_w^{(\sigma\to\rho)\to((\tau\to\rho)\to((\sigma+\tau)\to\rho))}$, *for every* $\sigma,\tau,\rho \in \mathsf{Types}$,
- $\boldsymbol{z}_w \in A_w^{0\to\sigma}$, *for every* $\sigma \in \mathsf{Types}$,
- $\boldsymbol{u}_w \in A_w^1$,

such that for every combinator

$$A_w \in \{\boldsymbol{k}_w, \boldsymbol{s}_w, \boldsymbol{p}_{1,w}, \boldsymbol{p}_{2,w}, \boldsymbol{p}_w, \boldsymbol{i}_{1,w}, \boldsymbol{i}_{2,w}, \boldsymbol{c}_w, \boldsymbol{z}_w, \boldsymbol{u}_w\}$$

whenever $w \preceq w'$, we have that $A_{w'} = i_{w,w'}(A_w)$. The combinators have to satisfy the following equations:

$$(\boldsymbol{k}_w x)y = x \tag{3.3}$$
$$((\boldsymbol{s}_w x)y)z = (xz)(yz) \tag{3.4}$$
$$\boldsymbol{p}_{1,w}x = Proj_{1,w}(x) \tag{3.5}$$
$$\boldsymbol{p}_{2,w}x = Proj_{2,w}(x) \tag{3.6}$$
$$\boldsymbol{p}_{1,w}((\boldsymbol{p}_w x)y) = x \tag{3.7}$$
$$\boldsymbol{p}_{2,w}((\boldsymbol{p}_w x)y) = y \tag{3.8}$$
$$\boldsymbol{i}_{1,w}x = Inl_w(x) \tag{3.9}$$
$$\boldsymbol{i}_{2,w}x = Inr_w(x) \tag{3.10}$$
$$((\boldsymbol{c}_w f)g)(\boldsymbol{i}_{1,w}x) = fx \tag{3.11}$$
$$((\boldsymbol{c}_w f)g)(\boldsymbol{i}_{2,w}x) = gx \tag{3.12}$$
$$((\boldsymbol{c}_w((\boldsymbol{s}_w(\boldsymbol{k}_w f))\boldsymbol{i}_{1,w}))((\boldsymbol{s}_w(\boldsymbol{k}_w f))\boldsymbol{i}_{2,w}))z = fz \tag{3.13}$$

When an applicative structure has combinators, we also say that it satisfies *the combinatory model condition*. The combinatory model condition guarantees that there are sufficiently many elements in a model so that each program (term without variables) has the meaning in the model, i.e. each program can be interpreted in the model. The same approach has been used in [124] and [123]. We may notice that there is a correspondence between combinators and axioms of Hilbert-style system for full intuitionistic propositional logic (with all the connectives). For each combinator there are two conditions: it has to belong to a certain set A^{σ}_{w} and it has to satisfy a certain equation. The superscripts of the sets A^{σ}_{w}, which have to contain combinators, correspond to the axioms of Hilbert-style system for the full intuitionistic propositional logic and to the types of term constants in Figure 3.2.

The monotonicity in Kripke semantics models the growth of knowledge in time: whatever is true at this moment (possible world w), it will remain true in the future (possible world w', $w \preceq w'$) ([110]). We define Kripke CL model to be monotone in that sense. More precisely, a model should satisfy the following: if a term M has type σ in a world $w \in W$, then it will have the type σ in every possible world w' accessible from w ($w \preceq w'$). We ensure this by requiring the monotonicity of valuations of term variables with respect to the accessibility relation.

Definition 3.11. An environment ρ *for a Kripke applicative structure* $\mathcal{K}$ *is a mapping from the set of term variables and the set of possible worlds to domains* $\rho : X \times W \to \bigcup D_w$ *such that for* $x \in X$ *and* $w \in W$, $\rho(x, w) \in D_w$ *holds, and the mapping* ρ *satisfies the following condition:*

$$\text{if } \rho(x,w) \in D_w \text{ and } w \preceq w', \text{ then } \rho(x,w') = i_{w,w'}(\rho(x,w)). \tag{3.14}$$

The important difference between Definition 3.11 and the definition of an environment in [124] is the fact that the environment in [124] was a partial mapping, whereas herein it is a total mapping. The motivation for defining environments as a total mappings will be given in Remark 3.20.

If ρ is an environment for a Kripke applicative structure $\mathcal{K}$ and $a \in D_w$, then $\rho(x := a)$ is an environment such that,

$$\rho(x := a)(y, w') = \begin{cases} i_{w,w'}(a), & y = x \text{ and } w \preceq w' \\ i^{-1}_{w',w}(a), & y = x \text{ and } w' \preceq w \\ \rho(y, w'), & y \neq x \text{ or neither } w \preceq w' \text{ nor } w' \preceq w \text{ holds} \end{cases}$$

where $i^{-1}_{w',w}(a)$ denotes the element $b \in D_{w'}$ such that $i_{w',w}(b) = a$. The existence of this element for every $w, w' \in W$, $w' \preceq w$ and $a \in D_w$ is a

consequence of the requirement that for every two worlds w and w' such that $w' \preceq w$ the transition function $i_{w',w}$ is surjective.

Remark 3.12. *The valuation $\rho(x := a)$ given above differs from the valuation $\rho(x := a)$ introduced in [124] and this difference is caused by the differences in the definitions of environments. In [124], an environment is a partial mapping, so it was enough to define the value of $\rho(x := a)(y, w')$ for $w \preceq w'$, where $a \in D_w$, and leave it undefined in other cases. Since we have defined an environment as a total mapping, the value $\rho(x := a)(y, w')$ has to be defined for every world w' and its definition has to ensure that the condition (3.14) is satisfied. For this reason, we had to add a requirement for the transition functions to be surjective, in contrast to [124], where this condition was not necessary.*

Finally, we define a Kripke CL model by providing a Kripke applicative structure with an environment.

Definition 3.13 (Kripke CL model)**.** A Kripke CL model *is $\mathcal{K}_\rho$ is a Kripke applicative structure which is extensional and has combinators and ρ is an environment for $\mathcal{K}$.*

We give some remarks about the valuation of term variables. As we have already explained, an environment is hereditary and this is ensured by the condition 3.14. Further, it is possible that $\bigcup_{\sigma \in \mathsf{Types}} A^\sigma_w \subset D_w$ and that there is no $\sigma \in \mathsf{Types}$ such that $\rho(x, w)$ belongs to A^σ_w.

In order to define the interpretation of terms, we extend the valuation of term variables to the interpretation map $[\![\ ]\!]^w_\rho$. The interpretation map $[\![\ ]\!]^w_\rho$ is a mapping from the set of all terms to the domain of world w, namely D_w. By $[\![M]\!]^w_\rho$ we denote the meaning of a term M in the environment ρ at world w and we define it inductively as follows.

Definition 3.14. *Let $\mathcal{K}_\rho$ be a Kripke CL model and $w \in W$ one possible world of the model $\mathcal{K}_\rho$. We define the interpretation map $[\![\ ]\!]^w_\rho : CL^{\to,\times,+} \to D_w$ as follows:*

1. $[\![x]\!]^w_\rho = \rho(x, w)$,
2. $[\![\mathsf{K}]\!]^w_\rho = \boldsymbol{k}_w$,
3. $[\![\mathsf{S}]\!]^w_\rho = \boldsymbol{s}_w$,
4. $[\![\mathsf{P}_1]\!]^w_\rho = \boldsymbol{p}_{1,w}$,
5. $[\![\mathsf{P}_2]\!]^w_\rho = \boldsymbol{p}_{2,w}$,

6. $[\![\mathsf{P}]\!]^w_\rho = \boldsymbol{p}_w$,
7. $[\![\mathsf{I}_1]\!]^w_\rho = \boldsymbol{i}_{1,w}$,
8. $[\![\mathsf{I}_2]\!]^w_\rho = \boldsymbol{i}_{2,w}$,
9. $[\![\mathsf{C}]\!]^w_\rho = \boldsymbol{c}_w$,
10. $[\![\mathsf{Z}]\!]^w_\rho = \boldsymbol{z}_w$,
11. $[\![\mathsf{U}]\!]^w_\rho = \boldsymbol{u}_w$,
12. $[\![MN]\!]^w_\rho = App_w([\![M]\!]^w_\rho, [\![N]\!]^w_\rho)$,
13. $[\![\pi_1(M)]\!]^w_\rho = Proj_{1,w}([\![M]\!]^w_\rho)$,
14. $[\![\pi_2(M)]\!]^w_\rho = Proj_{2,w}([\![M]\!]^w_\rho)$,
15. $[\![in_1(M)]\!]^w_\rho = Inl_w([\![M]\!]^w_\rho)$,
16. $[\![in_2(M)]\!]^w_\rho = Inr_w([\![M]\!]^w_\rho)$,
17. $[\![\langle\,\rangle]\!]^w_\rho = 1_w$, *the unique element of* A^1_w.

For every term M, the interpretation of the term M in the environment ρ at world w, $[\![M]\!]^w_\rho$, is defined and it belongs to the domain of the world w, $[\![M]\!]^w_\rho \in D_w$. The interpretation of a term M does not depend on the variables that do not appear in the term, it depends only on the interpretation of subterms of the term M. As a consequence, if two environments assign the same value to each variable that appears in a term M, then the interpretations of the term M in those environments are equal.

Lemma 3.15. *Let $\mathcal{K}$ be an extensional Kripke applicative structure with combinators, ρ_1 and ρ_2 environments for $\mathcal{K}$ and M a term. If $\rho_1(x,w) = \rho_2(x,w)$ for all $x \in FV(M)$, then $[\![M]\!]^w_{\rho_1} = [\![M]\!]^w_{\rho_2}$.*

Proof. The proof is by induction on the structure of the term M.

- If M is a variable x, then $FV(x) = \{x\}$. Hence, $[\![x]\!]^w_{\rho_1} = \rho_1(x,w) = \rho_2(x,w) = [\![x]\!]^w_{\rho_2}$.
- If M is a term constant, then the interpretation of M does not depend on the environment and it is determined by the Kripke applicative structure. Since both ρ_1 and ρ_2 are the environments for the same Kripke applicative structure, a term constant has the same interpretation in both environments.

- If M is the application NL, then by the induction hypothesis on N and L, we get $[\![N]\!]^w_{\rho_1} = [\![N]\!]^w_{\rho_2}$ and $[\![L]\!]^w_{\rho_1} = [\![L]\!]^w_{\rho_2}$. From Definition 3.14 it follows that

$$[\![NL]\!]^w_{\rho_1} = App_w([\![N]\!]^w_{\rho_1}, [\![L]\!]^w_{\rho_1}) = App_w([\![N]\!]^w_{\rho_2}, [\![L]\!]^w_{\rho_2}) = [\![NL]\!]^w_{\rho_2}.$$

- Let M be the first projection $\pi_1(N)$. We can apply the induction hypothesis to N and obtain $[\![N]\!]^w_{\rho_1} = [\![N]\!]^w_{\rho_2}$. By Definition 3.14 we get:

$$[\![\pi_1(N)]\!]^w_{\rho_1} = Proj_{1,w}([\![N]\!]^w_{\rho_1}) = Proj_{1,w}([\![N]\!]^w_{\rho_2}) = [\![\pi_1(N)]\!]^w_{\rho_2}.$$

- Let M be the second projection $\pi_2(N)$. We can apply the induction hypothesis to N and obtain $[\![N]\!]^w_{\rho_1} = [\![N]\!]^w_{\rho_2}$. By Definition 3.14 we get:

$$[\![\pi_2(N)]\!]^w_{\rho_1} = Proj_{2,w}([\![N]\!]^w_{\rho_1}) = Proj_{2,w}([\![N]\!]^w_{\rho_2}) = [\![\pi_2(N)]\!]^w_{\rho_2}.$$

- If M is the left injection $\mathrm{in}_1(N)$, then we apply the induction hypothesis to N and derive $[\![N]\!]^w_{\rho_1} = [\![N]\!]^w_{\rho_2}$. Following Definition 3.14 we conclude:

$$[\![\mathrm{in}_1(N)]\!]^w_{\rho_1} = Inl_w([\![N]\!]^w_{\rho_1}) = Inl_w([\![N]\!]^w_{\rho_2}) = [\![\mathrm{in}_1(N)]\!]^w_{\rho_2}.$$

- If M is the right injection $\mathrm{in}_2(N)$, then we can apply the induction hypothesis to N and obtain $[\![N]\!]^w_{\rho_1} = [\![N]\!]^w_{\rho_2}$. Following Definition 3.14 we have:

$$[\![\mathrm{in}_2(N)]\!]^w_{\rho_1} = Inr_w([\![N]\!]^w_{\rho_1}) = Inr_w([\![N]\!]^w_{\rho_2}) = [\![\mathrm{in}_2(N)]\!]^w_{\rho_2}.$$

- Let M be the empty pair $\langle\rangle$. Then we have

$$[\![\langle\rangle]\!]^w_{\rho_1} = 1_w = [\![\langle\rangle]\!]^w_{\rho_2}.$$

□

Definition 3.16. *We define the* satisfiability *of a statement $M : \sigma$ in a model and* semantical consequence *in the following way:*

1. *A world w satisfies the statement $M : \sigma$, denoted by $w \models M : \sigma$, if and only if $[\![M]\!]^w_\rho \in A^\sigma_w$. If the statement $M : \sigma$ is not satisfied in a world w, we write $w \not\models M : \sigma$.*

2. *A Kripke CL model $\mathcal{K}_\rho$ satisfies the statement $M : \sigma$ if and only if every world w of the model $\mathcal{K}_\rho$ satisfies the statement $M : \sigma$, i.e.*

$$\mathcal{K}_\rho \models M : \sigma \text{ if and only if } w \models M : \sigma, \text{ for all } w \in W.$$

If a Kripke CL model $\mathcal{K}_\rho$ does not satisfy the statement $M : \sigma$, we write $\mathcal{K}_\rho \not\models M : \sigma$.

3. *A possible world w satisfies the basis $\Gamma = \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\}$, denoted by $w \models \Gamma$, if and only if it satisfies every declaration from Γ, i.e. $w \models x_i : \sigma_i$, for all $i \in \{1, \ldots, n\}$. A Kripke CL model $\mathcal{K}_\rho$ satisfies the basis Γ, denoted by $\mathcal{K}_\rho \models \Gamma$, if and only if the basis Γ is satisfied in every world w of the model $\mathcal{K}_\rho$, i.e. $w \models \Gamma$ for all $w \in W$.*

4. *A statement $M : \sigma$ is a semantical consequence of a basis Γ, denoted by $\Gamma \models M : \sigma$, if whenever a Kripke CL model satisfies the basis Γ ($\mathcal{K}_\rho \models \Gamma$) it also satisfies the statement $M : \sigma$ ($\mathcal{K}_\rho \models M : \sigma$).*

A valuation of terms is hereditary by its definition, in the sense that if a type σ is assigned to the variable x in a world w, then the same type will be assigned to a variable x in every world accessible form w. The same property holds for all terms, as we will show. We prove that the interpretation of a term M at a world w uniquely determines the interpretation of the term M at any world w' accessible from w, using transition function $i_{w,w'}$.

Lemma 3.17. *Let $\mathcal{K}_\rho$ be a Kripke CL model. If $[\![M]\!]^w_\rho$ is defined and $w \preceq w'$, then $[\![M]\!]^{w'}_\rho = i_{w,w'}([\![M]\!]^w_\rho)$.*

Proof. The proof is by induction on the structure of the term M.

- Let M be a variable x. Then $[\![x]\!]^{w'}_\rho = \rho(x, w') = i_{w,w'}(\rho(x, w)) = i_{w,w'}([\![x]\!]^w_\rho)$ by Definition 3.11 and 3.14.

- If M is a constant K, then $[\![\mathsf{K}]\!]^{w'}_\rho = \boldsymbol{k}_{w'} = i_{w,w'}(\boldsymbol{k}_w) = i_{w,w'}([\![\boldsymbol{k}]\!]^w_\rho)$ by Definition 3.10 and 3.14. The cases for all other term constants proceed similarly.

- Assume M is the application NL. Then by Definition 3.8, 3.14 and the induction hypothesis we get

$$\begin{aligned}[\![NL]\!]^{w'}_\rho &= App_{w'}([\![N]\!]^{w'}_\rho, [\![L]\!]^{w'}_\rho) = App_{w'}(i_{w,w'}([\![N]\!]^w_\rho), i_{w,w'}([\![L]\!]^w_\rho)) \\ &= i_{w,w'}(App_w([\![N]\!]^w_\rho, [\![L]\!]^w_\rho)) = i_{w,w'}([\![NL]\!]^w_\rho).\end{aligned}$$

- If M is the first projection $\pi_1(N)$, then by Definition 3.8, 3.14 and the induction hypothesis we get

$$\begin{aligned}[\![\pi_1(N)]\!]^{w'}_\rho &= Proj_{1,w'}([\![N]\!]^{w'}_\rho) = Proj_{1,w'}(i_{w,w'}([\![N]\!]^w_\rho)) \\ &= i_{w,w'}(Proj_{1,w}([\![N]\!]^w_\rho)) = i_{w,w'}([\![\pi_1(N)]\!]^w_\rho).\end{aligned}$$

- The case when M is the second projection $\pi_2(N)$ proceeds similarly.

$$\begin{aligned}[\![\pi_2(N)]\!]^{w'}_\rho &= Proj_{2,w'}([\![N]\!]^{w'}_\rho) = Proj_{2,w'}(i_{w,w'}([\![N]\!]^w_\rho)) \\ &= i_{w,w'}(Proj_{2,w}([\![N]\!]^w_\rho)) = i_{w,w'}([\![\pi_2(N)]\!]^w_\rho).\end{aligned}$$

- Let M be the left injection $\mathrm{in}_1(N)$. Then by Definition 3.8, 3.14 and the induction hypothesis we derive

$$\begin{aligned}[\![\mathrm{in}_1(N)]\!]_\rho^{w'} &= Inl_{w'}([\![N]\!]_\rho^{w'}) = Inl_{w'}(i_{w,w'}([\![N]\!]_\rho^{w})) \\ &= i_{w,w'}(Inl_w([\![N]\!]_\rho^{w})) = i_{w,w'}([\![\mathrm{in}_1(N)]\!]_\rho^{w}).\end{aligned}$$

- Similarly, if M is the right injection $\mathrm{in}_2(N)$, then we have

$$\begin{aligned}[\![\mathrm{in}_2(N)]\!]_\rho^{w'} &= Inr_{w'}([\![N]\!]_\rho^{w'}) = Inr_{w'}(i_{w,w'}([\![N]\!]_\rho^{w})) \\ &= i_{w,w'}(Inr_w([\![N]\!]_\rho^{w})) = i_{w,w'}([\![\mathrm{in}_2(N)]\!]_\rho^{w}).\end{aligned}$$

- Assume M is the empty pair $\langle\rangle$. Then

$$[\![\langle\rangle]\!]_\rho^{w'} = 1_{w'} = i_{w,w'}(1_w) = i_{w,w'}([\![\langle\rangle]\!]_\rho^{w}).$$

This concludes the proof. □

Lemma 3.18 (Substitution Lemma)**.** *Let $\mathcal{K}_\rho$ be a Kripke CL model. For terms M, N, a variable x that appears in the term M and a world $w \in W$, it holds that $[\![M\{N/x\}]\!]_\rho^w = [\![M]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}$.*

Proof. The proof proceeds by induction on the structure of the term M.

- If the term M is a variable, then we distinguish two cases.
  - The case when the term M is a variable x:

$$[\![x\{N/x\}]\!]_\rho^w = [\![N]\!]_\rho^w = [\![x]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}.$$

  - The case when the term M is a variable y such that $y \not\equiv x$: by Lemma 3.15 we have

$$[\![y\{N/x\}]\!]_\rho^w = [\![y]\!]_\rho^w = [\![y]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}.$$

- If the term M is the application PQ, then by induction hypothesis we have

$$\begin{aligned}[\![(PQ)\{N/x\}]\!]_\rho^w &= [\![(P\{N/x\})(Q\{N/x\})]\!]_\rho^w \\ &= App_w([\![P\{N/x\}]\!]_\rho^w, [\![Q\{N/x\}]\!]_\rho^w) \\ &= App_w([\![P]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}, [\![Q]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}) \\ &= [\![PQ]\!]^w_{\rho(x:=[\![N]\!]_\rho^w)}.\end{aligned}$$

- Next, we assume the term M is the first projection $\pi_1(L)$. We apply the induction hypothesis to L and obtain

$$\begin{aligned}[\![\pi_1(L)\{N/x\}]\!]^w_\rho &= [\![\pi_1(L\{N/x\})]\!]^w_\rho = Proj_{1,w}([\![L\{N/x\}]\!]^w_\rho)\\ &= Proj_{1,w}([\![L]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}) = [\![\pi_1(L)]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}.\end{aligned}$$

- If M is the second projection $\pi_2(L)$, then again we can apply the induction hypothesis to L and we have

$$\begin{aligned}[\![\pi_2(L)\{N/x\}]\!]^w_\rho &= [\![\pi_2(L\{N/x\})]\!]^w_\rho = Proj_{2,w}([\![L\{N/x\}]\!]^w_\rho)\\ &= Proj_{2,w}([\![L]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}) = [\![\pi_2(L)]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}.\end{aligned}$$

- Assume M is the left injection $\mathrm{in}_1(L)$. Then by applying the induction hypothesis to L we derive

$$\begin{aligned}[\![\mathrm{in}_1(L)\{N/x\}]\!]^w_\rho &= [\![\mathrm{in}_1(L\{N/x\})]\!]^w_\rho = Inl_w([\![L\{N/x\}]\!]^w_\rho)\\ &= Inl_w([\![L]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}) = [\![\mathrm{in}_1(L)]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}.\end{aligned}$$

- If M is the right injection $\mathrm{in}_2(L)$, then similarly to the previous case we have

$$\begin{aligned}[\![\mathrm{in}_2(L)\{N/x\}]\!]^w_\rho &= [\![\mathrm{in}_2(L\{N/x\})]\!]^w_\rho = Inr_w([\![L\{N/x\}]\!]^w_\rho)\\ &= Inr_w([\![L]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}) = [\![\mathrm{in}_2(L)]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}.\end{aligned}$$

- The case when M is the empty pair $\langle\rangle$ follows from Lemma 3.15.

$$[\![\langle\rangle\{N/x\}]\!]^w_\rho = [\![\langle\rangle]\!]^w_\rho = [\![\langle\rangle]\!]^w_{\rho(x:=[\![N]\!]^w_\rho)}.$$

□

The Kripke-style semantics we have presented has been introduced in [94] and is inspired by the Kripke-style semantics of the simply typed λ-calculus introduced in [124]. In [94], we have introduced the proposed semantics, but the calculus was not exactly the same as the one we study in this chapter. The idea was to introduce a Kripke-style semantics of the full simply typed λ-calculus. Inspired by the work of [124], we have introduced the Kripke-style semantics of the full simply typed λ-calculus and the full simply typed combinatory logic with the goal to prove the completeness of the full simply typed λ-calculus using the translation of λ-calculus into combinatory logic. However, we did not include the typing rule that ensures that equal terms inhabit the

same type and the proof failed. For this reason, we had to add the typing rule (Eq) in Figure 3.2. The proof that the full simply typed combinatory logic is sound and complete with respect to the proposed semantics is given in the following section. The proof of the soundness and completeness of the full simply typed λ-calculus is left for future work, since the translation of λ-calculus with the typing rule for equal terms into the combinatory logic with the same rule is more involved.

Although the semantics introduced in [124] and [94] are defined as an applicative structure which is extensional and has combinators, provided with a valuation of term variable, there are some significant differences. As the main difference, we would single out that we have defined the interpretation of a term $[\![M]\!]^w_\rho$ independently of its type, whereas in [124] the authors define an interpretation of a well-typed terms only, more precisely they define an interpretation of a typing statement $[\![\Gamma \vdash M : \sigma]\!]^w_\rho$ considering the term and its type at the same time. We have presented a denotational semantics which can be used for the analysis of computations that involve untypable terms. If we compare Kripke-style semantics we presented and Kripke semantics of the intuitionistic propositional logic, we may notice that our semantics concerns statement $M : \sigma$, which by the Curry-Howard correspondence means it interprets both a provable formula and its proof, whereas Kripke semantics of the intuitionistic propositional logic takes into account only a provable formula.

3.3 Soundness and completeness of $CL^{\rightarrow,\times,+}$

In this section, we present the main results of the chapter: the soundness and the completeness of the full simply typed combinatory logic. We give two soundness and completeness results. First, we prove that the equational theory $\mathcal{EQ}_{FCL}$, defined by the rules in Figure 3.1, is sound and complete with respect to the proposed Kripke-style semantics. Second, we prove soundness and completeness of the full simply typed combinatory logic, more precisely we prove that the type assignment system $CL^{\rightarrow,\times,+}$, introduced in Figure 3.2, is sound and complete with respect to the proposed Kripke-style semantics.

We start with soundness results. Proving soundness of the equational theory means proving that every two terms that are equal, in the sense that $M =_\Gamma N$ for some Γ, have the same interpretation in every Kripke CL model which satisfies Γ. We prove soundness of the type assignment system by proving that for every Kripke CL model $\mathcal{K}_\rho$ and world w in that model, if $\Gamma \vdash M : \sigma$ and $w \models \Gamma$, then $w \models M : \sigma$. Similarly as in Proposition 3.7, we consider these properties together, since the equality of terms (Figure 3.1) and the type assignment system (Figure 3.2) are defined simultaneously.

Theorem 3.19. *For every Kripke CL model $\mathcal{K}_\rho$ and a possible world w of the model, if $M =_{\Gamma_1} N$, $\Gamma_2 \vdash L : \sigma$, $w \models \Gamma_1$ and $w \models \Gamma_2$, then $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models L : \sigma$.*

Proof. The proof proceeds by induction on the sum $k = n+m$ of the derivation length n of $M =_{\Gamma_1} N$ and the derivation length m of $\Gamma_2 \vdash L : \sigma$.

Step 1: The base case is when both derivations are obtained by applying just an axiom from Figure 3.1 and Figure 3.2, i.e. $n = 1$, $m = 1$ and $k = 2$. Since $n = 1$, $M =_{\Gamma_1} N$ has to be obtained by some axioms in Figure 3.1 (axioms $(1) - (12)$ and (15)). The proof follows from Definition 3.14 and Definition 3.10. Assume that $M =_{\Gamma_1} N$ is obtained by the axiom (1). Then $M \equiv \mathsf{K}QR$ and $N \equiv Q$ for some terms Q and R and we have

$$\begin{aligned}[\![\mathsf{K}QR]\!]^w_\rho &= App_w(App_w([\![\mathsf{K}]\!]^w_\rho, [\![Q]\!]^w_\rho), [\![R]\!]^w_\rho) \\ &= App_w(App_w(\boldsymbol{k}_w, [\![Q]\!]^w_\rho), [\![R]\!]^w_\rho) \\ &= [\![Q]\!]^w_\rho.\end{aligned}$$

All other cases for the axioms in Figure 3.1 proceed similarly.

If $m = 1$, then $\Gamma_2 \vdash L : \sigma$ is obtained from an axiom in Figure 3.2. Let it be (Axiom $\in$). Then $L : \sigma$ is a declaration which belongs to Γ and if world w satisfies Γ, it satisfies every declaration in Γ, thus $w \models P : \sigma$. If the derivation is obtained by (Axiom K), then the term L is the term constant K. By Definition 3.14 and Definition 3.10, we obtain

$$[\![\mathsf{K}]\!]^w_\rho = \boldsymbol{k}_w \in A^{\sigma\to\tau\to\sigma}_w.$$

Thus, $w \models \mathsf{K} : \sigma \to \tau \to \sigma$ for every model $\mathcal{K}_\rho$ and world w. The proof proceeds similarly for all other axioms in Figure 3.2.

Step 2: Let us assume that the statement holds for every $i < k$, $k \geq 3$.

$$\begin{aligned}&\text{For every Kripke } CL \text{ model } \mathcal{K}_\rho \text{ and a possible world } w \\ &\text{if } M =_{\Gamma_1} N, \Gamma_2 \vdash L : \sigma, \text{ the length of derivation } M =_{\Gamma_1} N \text{ is } n, \\ &\text{the length of derivation } \Gamma_2 \vdash L : \sigma \text{ is } m, n + m = i < k, \\ &w \models \Gamma_1 \text{ and } w \models \Gamma_2, \text{ then } [\![M]\!]^w_\rho = [\![N]\!]^w_\rho \text{ and } w \models L : \sigma.\end{aligned} \tag{IH*}$$

Step 3: We prove that the statement holds when the sum of derivation lengths is equal to k. We assume that the length of derivation $M =_{\Gamma_1} N$ is n, the length of derivation $\Gamma_2 \vdash L : \sigma$ is m and $w \models \Gamma_1$ and $w \models \Gamma_2$. We perform case analysis on the last rule applied in the derivations.

a) First, we consider the last applied rule in the derivation of $M =_{\Gamma_1} N$.

1. The rule (13) is the last rule applied. In this case we have $N \equiv \langle\rangle$ and the last step in the derivation is

$$\frac{\Gamma_1 \vdash M : 1}{M =_{\Gamma_1} \langle\rangle}$$

The length of derivation $\Gamma_1 \vdash M : 1$ is $n-1$. The length of the derivation $\mathsf{K}MN =_{\Gamma_2} M$ is 1. Since $n-1+1 = n < n+m = k$, we can apply the induction hypothesis to $\Gamma_1 \vdash M : 1$ and $\mathsf{K}MN =_{\Gamma_2} M$ and obtain $[\![\mathsf{K}MN]\!]^w_\rho = [\![M]\!]^w_\rho$ and $w \models M : 1$. The latter implies $[\![M]\!]^w_\rho \in A^1_w$. From the fact that A^1_w is a singleton $\{1_w\}$ we get $[\![M]\!]^w_\rho = 1_w = [\![\langle\rangle]\!]^w_\rho$. Still, it remains to prove $w \models L : \sigma$, under the assumptions $\Gamma_2 \vdash L : \sigma$ and $w \models \Gamma_2$. We distinguish three cases.

   - If $\Gamma_2 \vdash L : \sigma$ is obtained by an axiom, then we have a base case ($m = 1$), which has been already proved.
   - If $\Gamma_2 \vdash L : \sigma$ is obtained from a rule with one premise, then the result follows directly from the induction hypothesis applied to its premise and some equality, whose derivation length is 1. For example, let us assume that $\Gamma_2 \vdash L : \sigma$ is obtained by the rule ($\times$ elim1), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma \times \tau}{\Gamma_2 \vdash \pi_1(Q) : \sigma}$$

     The length of the derivation $\Gamma_2 \vdash Q : \sigma \times \tau$ is $m-1$. Since the derivation length of $\mathsf{K}MN =_{\Gamma_1} M$ is 1 and $m-1+1 = m < n+m = k$, we can apply the induction hypothesis to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \sigma \times \tau$ and we derive $[\![\mathsf{K}MN]\!]^w_\rho = [\![M]\!]^w_\rho$ and $w \models Q : \sigma \times \tau$. The latter implies $[\![Q]\!]^w_\rho \in A^{\sigma\times\tau}_w$ and by Definition 3.14 we have

$$[\![\pi_1(Q)]\!]^w_\rho = Proj_{1,2}([\![Q]\!]^w_\rho) \in A^\sigma_w.$$

     Thus, $w \models \pi_1(Q) : \sigma$. The rest of the cases with just one premise are analogous.
   - If the last applied rule has two premises, then we have two possibilities. If the last applied rule is ($\to$ elim), then the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \tau \to \sigma \qquad \Gamma_2 \vdash R : \tau}{\Gamma_2 \vdash QR : \sigma}$$

     The derivation lengths of both $\Gamma_2 \vdash Q : \tau \to \sigma$ and $\Gamma_2 \vdash R : \tau$ are less than $m-1$. By the induction hypothesis applied to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \tau \to \sigma$ we derive $[\![\mathsf{K}MN]\!]^w_\rho = [\![M]\!]^w_\rho$ and $w \models Q : \tau \to \sigma$. Similarly, we obtain $w \models R : \tau$. Now,

$w \models Q : \tau \to \sigma$ and $w \models R : \tau$ imply $w \models QR : \sigma$ by Definition 3.8 and 3.14. If the last applied rule is (Eq), then we have

$$\frac{\Gamma_2 \vdash Q : \sigma \qquad Q =_{\Gamma_2} L}{\Gamma_2 \vdash L : \sigma}$$

The derivation lengths of $\Gamma_2 \vdash Q : \sigma$ and $Q =_{\Gamma_2} L$ are less than $m-1$. By applying the induction hypothesis to $\mathsf{K}MN =_{\Gamma_1} M$ and $\Gamma_2 \vdash Q : \sigma$, we derive $[\![\mathsf{K}MN]\!]^w_\rho = [\![M]\!]^w_\rho$ and $w \models Q : \sigma$. Similarly, we can apply the induction hypothesis to $Q =_{\Gamma_2} L$ and $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$ and we obtain $[\![Q]\!]^w_\rho = [\![L]\!]^w_\rho$ and $w \models \mathsf{K} : \sigma \to \tau \to \sigma$. From $w \models Q : \sigma$ and $[\![Q]\!]^w_\rho = [\![L]\!]^w_\rho$, we derive $[\![L]\!]^w_\rho = [\![Q]\!]^w_\rho \in A^\sigma_w$, i.e, $w \models L : \sigma$.

2. The rule (14) is the last applied rule. Then $M \equiv R\{L/x\}$ and $N \equiv \mathsf{Z}L$, for some terms R and L. The last step in the derivation is

$$\frac{\Gamma_1 \vdash L : 0 \qquad \Gamma_1, x : 0 \vdash R : \sigma}{R\{L/x\} =_{\Gamma_1} \mathsf{Z}L}$$

The length of the derivation $\mathsf{KII} =_{\Gamma_2} \mathsf{I}$ is 1, whereas the derivation length of $\Gamma_1 \vdash L : 0$ is less than $n-1$, so we can apply the induction hypothesis to the latter two and obtain $[\![\mathsf{KII}]\!]^w_\rho = [\![\mathsf{I}]\!]^w_\rho$ and $w \models L : 0$. However, $w \models N : 0$ contradicts the condition that A^0_w is the empty set (Definition 3.8). Thus, it is not possible that (14) is the last rule applied.

3. The rule (16) is the last rule applied. Then we have

$$\frac{M =_{\Gamma_1} Q \qquad Q =_{\Gamma_1} N}{M =_{\Gamma_1} N}$$

By the induction hypothesis applied to $M =_{\Gamma_1} Q$ and $\Gamma_2 \vdash L : \sigma$, we obtain $[\![M]\!]^w_\rho = [\![Q]\!]^w_\rho$ and $w \models L : \sigma$. Similarly, by the induction hypothesis applied to $Q =_{\Gamma_1} N$ and $\Gamma_2 \vdash L : \sigma$, we get $[\![Q]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models L : \sigma$. From $[\![M]\!]^w_\rho = [\![Q]\!]^w_\rho$ and $[\![Q]\!]^w_\rho = [\![N]\!]^w_\rho$, we conclude $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$.

4. The rule (17) is the last rule applied. Then the last step in the derivation is

$$\frac{N =_{\Gamma_1} M}{M =_{\Gamma_1} N}$$

and the derivation length of $N =_{\Gamma_1} M$ is $n-1$. We can apply the induction hypothesis to $N =_{\Gamma_1} M$ and $\Gamma_2 \vdash L : \sigma$ and we obtain $[\![N]\!]^w_\rho = [\![M]\!]^w_\rho$ and $w \models L : \sigma$.

5. The rule (18) is the last rule applied. Then $M \equiv QR$ and $N \equiv Q'R$ for some terms Q, Q', R such that $Q =_{\Gamma_1} Q'$ and the last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{QR =_{\Gamma_1} Q'R}$$

By the induction hypothesis we derive $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$. From Definition 3.14 it follows that

$$[\![QR]\!]^w_\rho = App_w([\![Q]\!]^w_\rho, [\![R]\!]^w_\rho) = App_w([\![Q']\!]^w_\rho, [\![R]\!]^w_\rho) = [\![Q'R]\!]^w_\rho.$$

6. The rule (19) is the last rule applied. Then $M \equiv RQ$ and $N \equiv RQ'$ for some terms Q, Q', R such that $Q =_{\Gamma_1} Q'$ and the last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{RQ =_{\Gamma_1} RQ'}$$

By the induction hypothesis we derive $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$. Again, from Definition 3.14 it follows that

$$[\![RQ]\!]^w_\rho = App_w([\![R]\!]^w_\rho, [\![Q]\!]^w_\rho) = App_w([\![R]\!]^w_\rho, [\![Q']\!]^w_\rho) = [\![RQ']\!]^w_\rho.$$

7. The rule (20) is the last rule applied. Then the last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{\pi_1(Q) =_{\Gamma_1} \pi_1(Q')}$$

so $M \equiv \pi_1(Q)$ and $N \equiv \pi_1(Q')$ for some terms Q and Q'. By the induction hypothesis, we get $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$. Using the definition of the interpretation of a term we derive

$$[\![\pi_1(Q)]\!]^w_\rho = Proj_{1,w}([\![Q]\!]^w_\rho) = Proj_{1,w}([\![Q']\!]^w_\rho = [\![\pi_1(Q')]\!]^w_\rho.$$

8. The rule (21) is the last rule applied. Then $M \equiv \pi_2(Q)$ and $N \equiv \pi_2(Q')$ for some terms Q and Q' and the last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{\pi_2(Q) =_{\Gamma_1} \pi_2(Q')}$$

By the induction hypothesis, we obtain $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$. Similarly to the previous case, by Definition 3.14 we derive

$$[\![\pi_2(Q)]\!]^w_\rho = Proj_{2,w}([\![Q]\!]^w_\rho) = Proj_{2,w}([\![Q']\!]^w_\rho = [\![\pi_2(Q')]\!]^w_\rho.$$

9. The rule (22) is the last rule applied. In this case, the last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{\mathrm{in}_1(Q) =_{\Gamma_1} \mathrm{in}_1(Q')}$$

By the induction hypothesis, we have $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$ and by Definition 3.14 we derive

$$[\![\mathrm{in}_1(Q)]\!]^w_\rho = Inl_w([\![Q]\!]^w_\rho) = Inl_w([\![Q']\!]^w_\rho) = [\![\mathrm{in}_1(Q')]\!]^w_\rho.$$

10. The rule (23) is the last rule applied. The last step in the derivation is

$$\frac{Q =_{\Gamma_1} Q'}{\mathrm{in}_2(Q) =_{\Gamma_1} \mathrm{in}_2(Q')}$$

Now, we have $[\![Q]\!]^w_\rho = [\![Q']\!]^w_\rho$ and $w \models L : \sigma$ by the induction hypothesis. From Definition 3.14 it follows that

$$[\![\mathrm{in}_2(Q)]\!]^w_\rho = Inr_w([\![Q]\!]^w_\rho) = Inr_w([\![Q']\!]^w_\rho) = [\![\mathrm{in}_2(Q')]\!]^w_\rho.$$

11. The rule (24) is the last rule applied. Then $M =_{\Gamma_1} N$ is obtained from $Mx =_{\Gamma_1} Nx$ for some variable x which appears neither in M nor in N, that is the last step in the derivation is

$$\frac{Mx =_{\Gamma_1} Nx \qquad x \notin FV(M) \cup FV(N)}{M =_{\Gamma_1} N}$$

By the induction hypothesis, we have $[\![Mx]\!]^w_\rho = [\![Nx]\!]^w_\rho$ and $w \models L : \sigma$ for every model $\mathcal{K}_\rho$ and world w which satisfies Γ_1. Let $\mathcal{K}_\rho$ be a Kripke CL model and w a world of that model such that $w \models \Gamma_1$. Further, let $w \preceq w'$ and d be an element of the domain $D_{w'}$ in the model $\mathcal{K}_\rho$. Terms Mx and Nx have the same interpretation in every model, so they also have the same interpretation in model $\mathcal{K}_{\rho(x:=d)}$. So, by Lemma 3.15 and Lemma 3.17 we have for every $w' \succeq w$

$$\begin{aligned}
App_{w'}(i_{w,w'}([\![M]\!]^w_\rho), d) &= App_{w'}([\![M]\!]^{w'}_\rho, [\![x]\!]^{w'}_{\rho(x:=d)}) \\
&= App_{w'}([\![M]\!]^{w'}_{\rho(x:=d)}, [\![x]\!]^{w'}_{\rho(x:=d)}) \\
&= [\![Mx]\!]^{w'}_{\rho(x:=d)} \\
&= [\![Nx]\!]^{w'}_{\rho(x:=d)} \\
&= App_{w'}([\![N]\!]^{w'}_{\rho(x:=d)}, [\![x]\!]^{w'}_{\rho(x:=d)}) \\
&= App_{w'}(i_{w,w'}([\![N]\!]^w_\rho), d).
\end{aligned}$$

Since $\mathcal{K}_\rho$ is a model, Kripke applicative structure $\mathcal{K}$ is extensional. We proved that for every $w' \succeq w$ and $d \in D_{w'}$,

$$App_{w'}(i_{w,w'}([\![M]\!]^w_\rho), d) = App_{w'}(i_{w,w'}([\![N]\!]^w_\rho), d).$$

By the extensionality of the Kripke applicative structure, we conclude $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$.

b) Next, we consider the last rule in the derivation of $\Gamma_2 \vdash L : \sigma$.

1. If $\Gamma_2 \vdash L : \sigma$ is obtained by the rule ($\to$ elim), then $L \equiv QR$ for some terms Q and R and the last step on the derivation is
$$\frac{\Gamma_2 \vdash Q : \tau \to \sigma \qquad \Gamma_2 \vdash R : \tau}{\Gamma_2 \vdash QR : \sigma}$$
By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \tau \to \sigma$ we obtain $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \tau \to \sigma$. Similarly, by the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash R : \tau$, we have $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models R : \sigma$. By Definition 3.8 and 3.14 we conclude that $w \models Q : \tau \to \sigma$ and $w \models R : \tau$ imply $w \models QR : \sigma$.
2. If the last applied rule is ($\times$ elim1), then $L \equiv \pi_1(Q)$ for some term Q and the last step in the derivation is
$$\frac{\Gamma_2 \vdash Q : \sigma \times \tau}{\Gamma_2 \vdash \pi_1(Q) : \sigma}$$
By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma \times \tau$, we derive $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \sigma \times \tau$. The latter implies $[\![Q]\!]^w_\rho \in A^{\sigma\times\tau}_w$ and by Definition 3.8 and Definition 3.14 we get
$$[\![\pi_1(Q)]\!]^w_\rho = Proj_{1,w}([\![Q]\!]^w_\rho) \in A^\sigma_w.$$
Thus, $w \models \pi_1(Q) : \sigma$.
3. Similarly, if the last applied rule is ($\times$ elim2), then $L \equiv \pi_2(Q)$ for some term Q and the last step in the derivation is
$$\frac{\Gamma_2 \vdash Q : \tau \times \sigma}{\Gamma_2 \vdash \pi_2(Q) : \sigma}$$
By the induction hypothesis, we obtain $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \tau \times \sigma$. The latter implies $[\![Q]\!]^w_\rho \in A^{\tau\times\sigma}_w$ and by Definition 3.8 and 3.14, we have
$$[\![\pi_2(Q)]\!]^w_\rho = Proj_{2,w}([\![Q]\!]^w_\rho) \in A^\sigma_w.$$
Hence, $w \models \pi_2(Q) : \sigma$.

4. Let the last applied rule in the derivation $\Gamma_2 \vdash L : \sigma$ be the rule (+ intro1). We have $L \equiv \mathrm{in}_1(Q)$ for some term Q, $\sigma \equiv \tau_1 + \tau_1$ for some types τ_1 and τ_2 and the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \tau_1}{\Gamma_2 \vdash \mathrm{in}_1(Q) : \tau_1 + \tau_2}$$

By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \tau_1$ we get $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \tau_1$. From the latter it follows that $[\![Q]\!]^w_\rho \in A^{\tau_1}_w$. Further, by Definition 3.8 and 3.14 we derive

$$[\![\mathrm{in}_1(Q)]\!]^w_\rho = Inl_w([\![Q]\!]^w_\rho) \in A^{\tau_1+\tau_2}_w.$$

5. If the last applied rule in the derivation $\Gamma_2 \vdash L : \sigma$ is the rule (+ intro2), then $L \equiv \mathrm{in}_2(Q)$ for some term Q, $\sigma \equiv \tau_1 + \tau_2$ for some types τ_1 and τ_2 and the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \tau_2}{\Gamma_2 \vdash \mathrm{in}_2(Q) : \tau_1 + \tau_2}$$

By the induction hypothesis, we have $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \tau_2$. From the latter, it follows that $[\![Q]\!]^w_\rho \in A^{\tau_2}_w$ and by Definition 3.8 and 3.14 we get

$$[\![\mathrm{in}_2(Q)]\!]^w_\rho = Inr_w([\![Q]\!]^w_\rho) \in A^{\tau_1+\tau_2}_w.$$

6. Finally, we consider the case when the last applied rule is (Eq). Then we have that the last step in the derivation is

$$\frac{\Gamma_2 \vdash Q : \sigma \qquad Q =_{\Gamma_2} L}{\Gamma_2 \vdash L : \sigma}$$

By the induction hypothesis applied to $M =_{\Gamma_1} N$ and $\Gamma_2 \vdash Q : \sigma$ we get $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ and $w \models Q : \sigma$. Since the derivation $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$ has the length 1 and the length of the derivation $Q =_{\Gamma_2} L$ is less than $m - 1$, we can apply the induction hypothesis to $Q =_{\Gamma_2} L$ and $\Gamma_1 \vdash \mathsf{K} : \sigma \to \tau \to \sigma$ and we obtain $[\![Q]\!]^w_\rho = [\![L]\!]^w_\rho$ and $w \models \mathsf{K} : \sigma \to \tau \to \sigma$. We conclude $[\![L]\!]^w_\rho = [\![Q]\!]^w_\rho \in A^\sigma_w$, thus $w \models L : \sigma$.

This concludes the proof of Theorem 3.19. □

Remark 3.20. *We now give the motivation for defining an environment as a total mapping. In order to prove that equal terms have the same interpretation in every model, an environment has to be a total mapping. Otherwise, if ρ is a partial mapping and $\rho(x, w)$ is defined, whereas $\rho(y, w)$ is not, then $[\![\mathsf{K}xy]\!]^w_\rho$ is not defined. As a consequence, we would have that the interpretation of $\mathsf{K}xy$ is not the same as the interpretation of x, although they are equal terms.*

A direct consequence of the previous theorem is the soundness of the type assignment system.

Corollary 3.21 (Soundness of $CL^{\rightarrow,\times,+}$)**.** *If* $\Gamma \vdash M : \sigma$*, then* $\Gamma \models M : \sigma$.

The equivalence class of a term M with respect to the equivalence relation generated by the rules in Figure 3.1 is denoted by $[M]_\Gamma$, i.e.

$$[M]_\Gamma = \{N \mid N \in CL^{\rightarrow,\times,+} \text{ and } M =_\Gamma N\}.$$

We prove the completeness of the equational theory and the type assignment system separately. First, a notion of a canonical model is introduced.

Definition 3.22 (Canonical model)**.** *Let* Γ_0 *be a consistent basis. A* canonical model $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ *is a pair* $\langle \mathcal{K}^{\Gamma_0}, \rho^\star \rangle$ *such that the tuple*

$$\mathcal{K}^{\Gamma_0} = \langle W_{\Gamma_0}, \preceq, \{D_{w_\Gamma}\}, \{A^\sigma_{w_\Gamma}\}, \{App_{w_\Gamma}\}, \{Proj_{1,w_\Gamma}\}, \{Proj_{2,w_\Gamma}\}, \{Inl_{w_\Gamma}\}, \{Inr_{w_\Gamma}\}, \{i_{w_\Gamma, w_{\Gamma'}}\}\rangle$$

consists of:

(i) the set W_{Γ_0} *of possible worlds, one for each consistent superset of the basis* Γ_0*, i.e.* $W_{\Gamma_0} = \{w_\Gamma \mid \Gamma_0 \subseteq \Gamma$ *and* Γ *is a consistent basis*$\}$*,*

(ii) the relation $\preceq$ *on* W_{Γ_0} *defined as follows:*

$$w_\Gamma \preceq w_{\Gamma'} \text{ if and only if } \Gamma \subseteq \Gamma', \tag{3.15}$$

(iii) the family $\{D_{w_\Gamma}\} = \{D_{w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$D_{w_\Gamma} = \{[M]_\Gamma \mid M \in CL^{\rightarrow,\times,+}\},$$

(iv) the family $\{A^\sigma_{w_\Gamma}\} = \{A^\sigma_{w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}, \sigma \in \mathsf{Types}}$ *where*

$$A^\sigma_{w_\Gamma} = \{[N]_\Gamma \mid N \in CL^{\rightarrow,\times,+} \text{ and } \Gamma \vdash N : \sigma\},$$

(v) the family $\{App_{w_\Gamma}\} = \{App_{w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$App_{w_\Gamma}([M]_\Gamma, [N]_\Gamma) = [MN]_\Gamma,$$

(vi) the family $\{Proj_{1,w_\Gamma}\} = \{Proj_{1,w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$Proj_{1,w_\Gamma}([M]_\Gamma) = [\pi_1(M)]_\Gamma,$$

(vii) *the family* $\{Proj_{2,w_\Gamma}\} = \{Proj_{2,w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$Proj_{2,w_\Gamma}([M]_\Gamma) = [\pi_2(M)]_\Gamma,$$

(viii) *the family* $\{Inl_{w_\Gamma}\} = \{Inl_{w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$Inl_{w_\Gamma}([M]_\Gamma) = [in_1(M)]_\Gamma,$$

(ix) *the family* $\{Inr_{w_\Gamma}\} = \{Inr_{w_\Gamma}\}_{w_\Gamma \in W_{\Gamma_0}}$ *where*

$$Inr_{w_\Gamma}([M]_\Gamma) = [in_2(M)]_\Gamma,$$

(x) *the family* $\{i_{w_\Gamma,w_{\Gamma'}}\} = \{i_{w_\Gamma,w_{\Gamma'}}\}_{w_\Gamma,w_{\Gamma'} \in W_{\Gamma_0}, w_\Gamma \preceq w_{\Gamma'}}$ *where*

$$i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma) = [M]_{\Gamma'}.$$

The valuation $\rho^\star$ *is defined by* $\rho^\star(x, w_\Gamma) = [x]_\Gamma$.

We prove that a canonical model introduced in Definition 3.22 is a Kripke CL model.

Lemma 3.23. *The canonical model* $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ *is a Kripke* CL *model.*

Proof. We prove that the tuple $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ satisfies the conditions of Definition 3.13. We show that $\mathcal{K}^{\Gamma_0}$ is a Kripke applicative structure (Definition 3.8), which is extensional (Definition 3.9) and has combinators (Definition 3.10).

By the definition of a canonical model (Definition 3.22) we have that the set of possible worlds is partially ordered by relation $\preceq$, since the set $\{\Gamma \mid \Gamma_0 \subseteq \Gamma$ and Γ is a consistent basis$\}$ is partially ordered by inclusion.

Let $\sigma \in \mathsf{Types}$. By Definition 3.22 we have

$$\begin{aligned} A^\sigma_{w_\Gamma} &= \{[N]_\Gamma \mid N \in CL^{\to,\times,+} \text{ and } \Gamma \vdash N : \sigma\} \\ &\subseteq \{[M]_\Gamma \mid M \in CL^{\to,\times,+}\} = D_{w_\Gamma} \end{aligned}$$

Further, as a direct consequence of the condition that every Γ is a consistent basis (Definition 3.5), we have that $A^0_{w_\Gamma} = \{[N]_\Gamma \mid \Gamma \vdash N : 0\} = \emptyset$ for every $w_\Gamma \in W_{\Gamma_0}$. Now, we look at the set $A^1_{w_\Gamma} = \{[N]_\Gamma \mid N \in CL^{\to,\times,+}$ and $\Gamma \vdash N : 1\}$. If $[M]_\Gamma \in A^1_{w_\Gamma}$, then by the definition of the set $A^1_{w_\Gamma}$, we have $\Gamma \vdash M : 1$. By the rule (13) in Figure 3.1 we conclude $M =_\Gamma \langle\rangle$, i.e. $[M]_\Gamma = [\langle\rangle]_\Gamma$. Thus, the only element of $A^1_{w_\Gamma}$ is $[\langle\rangle]_\Gamma$.

We define the injective function $H : D_{w_\Gamma} \uplus D_{w_\Gamma} \to D_{w_\Gamma}$ as follows:

$$H(\langle 0, [M]_\Gamma\rangle) = [\mathrm{in}_1(M)]_\Gamma$$
$$H(\langle 1, [M]_\Gamma\rangle) = [\mathrm{in}_2(M)]_\Gamma$$

It is an easy task to prove that the codomain of the restriction of the function H to the set $A^\sigma_{w_\Gamma} \uplus A^\tau_{w_\Gamma}$ is $A^{\sigma+\tau}_{w_\Gamma}$.

We define the injective function $G : D_{w_\Gamma} \to D_{w_\Gamma} \times D_{w_\Gamma}$ as $G([M]_\Gamma) = \langle [\pi_1(M)]_\Gamma, [\pi_2(M)]_\Gamma\rangle$. Again, we have that the codomain of the restriction of the function G to the set $A^{\sigma\times\tau}_{w_\Gamma}$ is $A^\sigma_{w_\Gamma} \times A^\tau_{w_\Gamma}$.

Next, the family of application functions App_{w_Γ} has to satisfy the condition (vi) in Definition 3.8. For $[M]_\Gamma \in D_{w_\Gamma}$ and $[N]_\Gamma \in D_{w_\Gamma}$ we have $App_{w_\Gamma}([M]_\Gamma, [N]_\Gamma) = [MN]_\Gamma \in D_{w_\Gamma}$ by Definition 3.22. If $[M]_\Gamma \in A^{\sigma\to\tau}_{w_\Gamma}$ and $[N]_\Gamma \in A^\sigma_{w_\Gamma}$ for some types $\sigma, \tau \in \mathsf{Types}$, then $\Gamma \vdash M : \sigma \to \tau$ and $\Gamma \vdash N : \sigma$ by Definition 3.22. By the rule $(\to$ elim) in Figure 3.2 we derive $\Gamma \vdash MN : \sigma$, i.e. $[MN]_\Gamma \in A^\tau_{w_\Gamma}$. Thus, if we restrict the function App_{w_Γ} to $A^{\sigma\to\tau}_{w_\Gamma} \times A^\sigma_{w_\Gamma}$, the codomain is $A^\tau_{w_\Gamma}$.

Let $[M]_\Gamma \in D_{w_\Gamma}$. By Definition 3.22 we have $Proj_{1,w_\Gamma}([M]_\Gamma) = [\pi_1(M)]_\Gamma \in D_{w_\Gamma}$ and $Proj_{2,w_\Gamma}([M]_\Gamma) = [\pi_2(M)]_\Gamma \in D_{w_\Gamma}$. If $[M]_\Gamma \in A^{\sigma\times\tau}_{w_\Gamma}$, then $\Gamma \vdash M : \sigma \times \tau$ and $\Gamma \vdash \pi_1(M) : \sigma$ follows by the rule ($\times$ elim1) in Figure 3.2. Thus, the codomain of the restriction of the function $Proj_{1,w_\Gamma}$ to the set $A^{\sigma\times\tau}_{w_\Gamma}$ is $A^\sigma_{w_\Gamma}$. Similarly, if $[M]_\Gamma \in A^{\sigma\times\tau}_{w_\Gamma}$, then $\Gamma \vdash M : \sigma \times \tau$ and by rule ($\times$ elim2) in Figure 4.2 we obtain $\Gamma \vdash \pi_2(M) : \tau$. Hence, we conclude that the codomain of the restriction of the function $Proj_{2,w_\Gamma}$ to the set $A^{\sigma\times\tau}_{w_\Gamma}$ is $A^\tau_{w_\Gamma}$.

For $[M]_\Gamma \in D_{w_\Gamma}$, we have

$$Inl_{w_\Gamma}([M]_\Gamma) = [\mathrm{in}_1(M)]_\Gamma \in D_{w_\Gamma}$$
$$Inr_{w_\Gamma}([M]_\Gamma) = [\mathrm{in}_2(M)]_\Gamma \in D_{w_\Gamma}$$

Let $[M]_\Gamma \in A^\sigma_{w_\Gamma}$. Then $\Gamma \vdash M : \sigma$ and we conclude $\Gamma \vdash \mathrm{in}_1(M) : \sigma + \tau$ by rule (+ intro1), i.e. the codomain of the restriction of the function Inl_{w_Γ} to the set $A^\sigma_{w_\Gamma}$ is $A^{\sigma+\tau}_{w_\Gamma}$. Similarly, if $[M]_\Gamma \in A^\sigma_{w_\Gamma}$, then $\Gamma \vdash M : \sigma$ and $\Gamma \vdash \mathrm{in}_2(M) : \tau + \sigma$ follows by rule (+ intro2) in Figure 4.2. Thus, if we restrict Inr_{w_Γ} to $A^\sigma_{w_\Gamma}$ the codomain is $A^{\tau+\sigma}_{w_\Gamma}$.

Let us consider the family of transition functions $i_{w_\Gamma, w_{\Gamma'}}$. By Definition 3.22, we have $i_{w_\Gamma, w_{\Gamma'}} : D_{w_\Gamma} \to D_{w_{\Gamma'}}$. In Proposition 3.7 we have proved that if $M =_\Gamma N$ and $\Gamma \subseteq \Gamma'$, then $M =_{\Gamma'} N$. This property ensures that he function $i_{w_\Gamma, w_{\Gamma'}}$ is well defined. More precisely, it cannot happen that $[M]_\Gamma = [N]_\Gamma$ and $i_{w_\Gamma, w_{\Gamma'}}([M]_\Gamma) \neq i_{w_\Gamma, w_{\Gamma'}}([N]_\Gamma)$. Let $[M]_\Gamma \in A^\sigma_{w_\Gamma}$ and $\Gamma \subseteq \Gamma'$ ($w_\Gamma \preceq w_{\Gamma'}$). By Definition 3.22, $[M]_\Gamma \in A^\sigma_{w_\Gamma}$ implies $\Gamma \vdash M : \sigma$ and by Proposition 3.7 we obtain $\Gamma' \vdash M : \sigma$, i.e. $[M]_{\Gamma'} \in A^\sigma_{w_{\Gamma'}}$. Hence, the codomain of the restriction of the function $i_{w,w'}$ to the set $A^\sigma_{w_\Gamma}$ is $A^\sigma_{w_{\Gamma'}}$.

It remains to prove that transition functions satisfy conditions id and comp and that they commute with functions $App_{w_\Gamma}, Proj_{1,w_\Gamma}, Proj_{2,w_\Gamma}, Inl_{w_\Gamma}$ and Inr_{w_Γ}. The transition function $i_{w_\Gamma,w_{\Gamma'}}$ is a surjective function from D_{w_Γ} to $D_{w_{\Gamma'}}$. By the definition of a canonical model, the function i_{w_Γ,w_Γ} is the identity, since $\Gamma \subseteq \Gamma$ and $i_{w_\Gamma,w_\Gamma}([M]_\Gamma) = [M]_\Gamma$. Let w_Γ, $w_{\Gamma'}$ and $w_{\Gamma''}$ be possible worlds of W_{Γ_0} such that $w_\Gamma \preceq w_{\Gamma'} \preceq w_{\Gamma''}$. The latter is equivalent to $\Gamma \subseteq \Gamma' \subseteq \Gamma''$ by Definition 3.22. By the definition of the transition function in Definition 3.22 we derive

$$\begin{aligned} i_{w_{\Gamma'},w_{\Gamma''}} \circ i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma) &= i_{w_{\Gamma'},w_{\Gamma''}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma)) = i_{w_{\Gamma'},w_{\Gamma''}}([M]_{\Gamma'}) \\ &= [M]_{\Gamma''} = i_{w_\Gamma,w_{\Gamma''}}([M]_\Gamma). \end{aligned}$$

This proves that a transition function satisfies the condition (comp). Let $[M]_\Gamma, [N]_\Gamma \in D_{w_\Gamma}$ and $w_\Gamma \preceq w_{\Gamma'}$. We then have the following sequences of equalities

$$\begin{aligned} i_{w_\Gamma,w_{\Gamma'}}(App_{w_\Gamma}([M]_\Gamma, [N]_\Gamma)) &= i_{w_\Gamma,w_{\Gamma'}}([MN]_\Gamma) \\ &= [MN]_{\Gamma'} \\ &= App_{w_{\Gamma'}}([M]_{\Gamma'}, [N]_{\Gamma'}) \\ &= App_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma), i_{w_\Gamma,w_{\Gamma'}}([N]_\Gamma)), \end{aligned}$$

$$\begin{aligned} i_{w_\Gamma,w_{\Gamma'}}(Proj_{1,w_\Gamma}([M]_\Gamma)) &= i_{w_\Gamma,w_{\Gamma'}}([\pi_1(M)]_\Gamma) \\ &= [\pi_1(M)]_{\Gamma'} \\ &= Proj_{1,w_{\Gamma'}}([M]_{\Gamma'}) \\ &= Proj_{1,w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma)), \end{aligned}$$

$$\begin{aligned} i_{w_\Gamma,w_{\Gamma'}}(Proj_{2,w_\Gamma}([M]_\Gamma)) &= i_{w_\Gamma,w_{\Gamma'}}([\pi_2(M)]_\Gamma) \\ &= [\pi_2(M)]_{\Gamma'} \\ &= Proj_{2,w_{\Gamma'}}([M]_{\Gamma'}) \\ &= Proj_{2,w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma)), \end{aligned}$$

$$\begin{aligned} i_{w_\Gamma,w_{\Gamma'}}(Inl_{w_\Gamma}([M]_\Gamma)) &= i_{w_\Gamma,w_{\Gamma'}}([\mathrm{in}_1(M)]_\Gamma) \\ &= [\mathrm{in}_1(M)]_{\Gamma'} \\ &= Inl_{w_{\Gamma'}}([M]_{\Gamma'}) \\ &= Inl_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma)), \end{aligned}$$

$$\begin{aligned} i_{w_\Gamma,w_{\Gamma'}}(Inr_{w_\Gamma}([M]_\Gamma)) &= i_{w_\Gamma,w_{\Gamma'}}([\mathrm{in}_2(M)]_\Gamma) \\ &= [\mathrm{in}_2(M)]_{\Gamma'} \\ &= Inr_{w_{\Gamma'}}([M]_{\Gamma'}) \\ &= Inr_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma)), \end{aligned}$$

which prove that $\mathcal{K}^{\Gamma_0}$ is a Kripke applicative structure. In order to prove that $\mathcal{K}^{\Gamma_0}$ has combinators we define the following elements:

- $\boldsymbol{k}_{w_\Gamma} = [\mathsf{K}]_\Gamma$
- $\boldsymbol{s}_{w_\Gamma} = [\mathsf{S}]_\Gamma$
- $\boldsymbol{p}_{w_\Gamma} = [\mathsf{P}]_\Gamma$
- $\boldsymbol{p}_{1,w_\Gamma} = [\mathsf{P}_1]_\Gamma$
- $\boldsymbol{p}_{2,w_\Gamma} = [\mathsf{P}_2]_\Gamma$
- $\boldsymbol{i}_{1,w_\Gamma} = [\mathsf{I}_1]_\Gamma$
- $\boldsymbol{i}_{2,w_\Gamma} = [\mathsf{I}_2]_\Gamma$
- $\boldsymbol{c}_{w_\Gamma} = [\mathsf{C}]_\Gamma$
- $\boldsymbol{z}_{w_\Gamma} = [\mathsf{Z}]_\Gamma$
- $\boldsymbol{u}_{w_\Gamma} = [\mathsf{U}]_\Gamma$

The elements defined above represent combinators of the applicative structure $\mathcal{K}^{\Gamma_0}$. The proof that these elements satisfy the conditions from Definition 3.10 follows straightforwardly from the rules in Figure 3.1 and Figure 3.2.

The extensionality of the applicative structure $\mathcal{K}^{\Gamma_0}$ is a direct consequence of the extensionality of combinatory logic, i.e. of the extensionality of the equational theory given in Figure 3.1. We prove that the two conditions from Definition 3.9 are satisfied. First, we show that for all $[M]_\Gamma, [N]_\Gamma \in D_{w_\Gamma}$, if $Proj_{1,w_\Gamma}([M]_\Gamma) = Proj_{1,w_\Gamma}([N]_\Gamma)$ and $Proj_{2,w_\Gamma}([M]_\Gamma) = Proj_{2,w_\Gamma}([N]_\Gamma)$, then $[M]_\Gamma = [N]_\Gamma$. By Definition 3.22 we have $Proj_{1,w_\Gamma}([M]_\Gamma) = [\pi_1(M)]_\Gamma$ and $Proj_{2,w_\Gamma}([M]_\Gamma) = [\pi_2(M)]_\Gamma$. So, from the assumption we conclude $[\pi_1(M)]_\Gamma = [\pi_1(N)]_\Gamma$ and $[\pi_2(M)]_\Gamma = [\pi_2(N)]_\Gamma$, i.e. $\pi_1(M) =_\Gamma \pi_1(N)$ and $\pi_2(M) =_\Gamma \pi_2(N)$. Now, by the rules in Figure 3.1 we derive

$$\begin{aligned} M &=_\Gamma \mathsf{P}(\mathsf{P}_1 M)(\mathsf{P}_2 M) =_\Gamma \mathsf{P}(\pi_1(M))(\pi_2(M)) \\ &=_\Gamma \mathsf{P}(\pi_1(N))(\pi_2(N)) =_\Gamma \mathsf{P}(\mathsf{P}_1 N)(\mathsf{P}_2 N) \\ &=_\Gamma N. \end{aligned}$$

Next we prove that the second condition of extensionality (Definition 3.9) is satisfied. Let $[M]_\Gamma$ and $[N]_\Gamma$ be the elements of D_{w_Γ} such that for all $\Gamma \subseteq \Gamma'$ and $[P]_{\Gamma'}, [Q]_{\Gamma'} \in D_{w_{\Gamma'}}$, the equalities

$$App_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma), Inl_{w_{\Gamma'}}([P]_{\Gamma'})) = App_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([N]_\Gamma), Inl_{w_{\Gamma'}}([P]_{\Gamma'})) \tag{3.16}$$

and

$$App_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([M]_\Gamma), Inr_{w_{\Gamma'}}([Q]_{\Gamma'})) = App_{w_{\Gamma'}}(i_{w_\Gamma,w_{\Gamma'}}([N]_\Gamma), Inr_{w_{\Gamma'}}([Q]_{\Gamma'})) \tag{3.17}$$

hold.

Since $\Gamma \subseteq \Gamma$, we take $\Gamma' \equiv \Gamma$. By Definition 3.22 we have the following equations:

$$App_{w_\Gamma}(i_{w_\Gamma,w_\Gamma}([M]_\Gamma), Inl_{w_\Gamma}([P]_\Gamma)) = [M(\mathrm{in}_1(P))]_\Gamma \tag{3.18}$$

$$App_{w_\Gamma}(i_{w_\Gamma,w_\Gamma}([M]_\Gamma), Inr_{w_\Gamma}([Q]_\Gamma)) = [M(\mathrm{in}_2(Q))]_\Gamma \tag{3.19}$$

$$App_{w_\Gamma}(i_{w_\Gamma,w_\Gamma}([N]_\Gamma), Inl_{w_\Gamma}([P]_\Gamma)) = [N(\mathrm{in}_1(P))]_\Gamma \tag{3.20}$$

$$App_{w_\Gamma}(i_{w_\Gamma,w_\Gamma}([N]_\Gamma), Inr_{w_\Gamma}([Q]_\Gamma)) = [N(\mathrm{in}_2(Q))]_\Gamma \tag{3.21}$$

The assumptions 3.16 and 3.17 are equivalent to the equations

$$[M(\mathrm{in}_1(P))]_\Gamma = [N(\mathrm{in}_1(P))]_\Gamma \text{ and } [M(\mathrm{in}_2(Q))]_\Gamma = [N(\mathrm{in}_2(Q))]_\Gamma$$

i.e. $M(\mathrm{in}_1(P)) =_\Gamma N(\mathrm{in}_1(P))$ and $M(\mathrm{in}_2(Q)) =_\Gamma N(\mathrm{in}_2(Q))$ for any terms P and Q. We take P and Q to be variables x and y which do not appear in terms M and N and we obtain $M(\mathrm{in}_1(x)) =_\Gamma N(\mathrm{in}_1(x))$ and $M(\mathrm{in}_2(y)) =_\Gamma N(\mathrm{in}_2(y))$. By the rules of Figure 3.1 we have

$$\begin{aligned} \mathsf{S}(\mathsf{K}M)\mathsf{I}_1 x &=_\Gamma (\mathsf{K}Mx)(\mathsf{I}_1 x) =_\Gamma M(\mathsf{I}_1 x) \\ &=_\Gamma M(\mathrm{in}_1(x)) =_\Gamma N(\mathrm{in}_1(x)) \\ &=_\Gamma (\mathsf{K}Nx)(\mathsf{I}_1 x) =_\Gamma \mathsf{S}(\mathsf{K}N)\mathsf{I}_1 x. \end{aligned}$$

Since x is not a variable of M and N, it appears neither in $\mathsf{S}(\mathsf{K}M)\mathsf{I}_1$ nor in $\mathsf{S}(\mathsf{K}N)\mathsf{I}_1$, therefore by the rule (24) in Figure 3.1 we conclude $\mathsf{S}(\mathsf{K}M)\mathsf{I}_1 =_\Gamma \mathsf{S}(\mathsf{K}N)\mathsf{I}_1$. Similarly,

$$\begin{aligned} \mathsf{S}(\mathsf{K}M)\mathsf{I}_2 y &=_\Gamma (\mathsf{K}My)(\mathsf{I}_2 y) =_\Gamma M(\mathsf{I}_2 y) \\ &=_\Gamma M(\mathrm{in}_2(y)) =_\Gamma N(\mathrm{in}_2(y)) \\ &=_\Gamma (\mathsf{K}Ny)(\mathsf{I}_2 y) =_\Gamma \mathsf{S}(\mathsf{K}N)\mathsf{I}_2 y. \end{aligned}$$

As a consequence, we have $\mathsf{S}(\mathsf{K}M)\mathsf{I}_2 =_\Gamma \mathsf{S}(\mathsf{K}N)\mathsf{I}_2$. The relation $=_\Gamma$ is closed under contexts, so for a variable z that does not appear in terms M and N we have

$$Mz =_\Gamma ((\mathsf{C}(\mathsf{S}(\mathsf{K}M)\mathsf{I}_1))(\mathsf{S}(\mathsf{K}M)\mathsf{I}_2))z =_\Gamma ((\mathsf{C}(\mathsf{S}(\mathsf{K}N)\mathsf{I}_1))(\mathsf{S}(\mathsf{K}N)\mathsf{I}_2))z =_\Gamma Nz.$$

Again, by the rule (24) in Figure 3.1 we conclude $M =_\Gamma N$, i.e. $[M]_\Gamma = [N]_\Gamma$. This concludes the proof of the extensionality.

It remains to prove that $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ is a Kripke CL model, i.e. that the mapping $\rho^\star$ satisfies the conditions of Definition 3.11. The environment $\rho^\star$ is a mapping from a set of variables and worlds to the set of domains by its definition, $\rho^\star(x, w_\Gamma) = [x]_\Gamma \in D_{w_\Gamma}$, where w_Γ is an arbitrary world of $\mathcal{K}^{\Gamma_0}_{\rho^\star}$. The environment $\rho^\star$ satisfies the condition 3.14. Let w_Γ and $w_{\Gamma'}$ be worlds in $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ such that $\rho^\star(x, w_\Gamma) \in D_{w_\Gamma}$ and $w_\Gamma \preceq w_{\Gamma'}$. By Definition 3.22 we have

$$\rho(x, w_{\Gamma'}) = [x]_{\Gamma'} = i_{w_\Gamma, w_{\Gamma'}}([x]_\Gamma) = i_{w_\Gamma, w_{\Gamma'}}(\rho(x, w_\Gamma)).$$

This concludes the proof that $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ is a Kripke CL model. □

The notion of an environment of a canonical model (Definition 3.22) is defined as the map which assigns to a variable its equivalence class. We prove that the interpretation of any term in a canonical model is its equivalence class.

Lemma 3.24. *Let Γ_0 be a consistent basis and $\mathcal{K}^{\Gamma_0}_{\rho^\star}$ a canonical model. For every term M and a possible world w_Γ in $\mathcal{K}^{\Gamma_0}_{\rho^\star}$, the equality $[\![M]\!]^{w_\Gamma}_{\rho^\star} = [M]_\Gamma$ holds.*

Proof. The proof is by induction on the structure of the term M.

If M is a variable x, then we have $[\![x]\!]^{w_\Gamma}_{\rho^\star} = \rho^\star(x, w_\Gamma) = [x]_\Gamma$ by Definition 3.22.

Let the term M be a constant K. Then by Definition 3.14 and Definition 3.22 we have $[\![\mathsf{K}]\!]^{w_\Gamma}_{\rho^\star} = \boldsymbol{k}_{w_\Gamma} = [\mathsf{K}]_\Gamma$. The remaining cases when M is a term constant are analogous.

If term M is an application NL, then by the induction hypothesis and Definition 3.14 we have

$$[\![NL]\!]^{w_\Gamma}_{\rho^\star} = App_{w_\Gamma}([\![N]\!]^{w_\Gamma}_{\rho^\star}, [\![L]\!]^{w_\Gamma}_{\rho^\star}) = App_{w_\Gamma}([N]_\Gamma, [L]_\Gamma) = [NL]_\Gamma.$$

Assume that the term M is the first projection $\pi_1(N)$. By Definition 3.14, Definition 3.22 and the induction hypothesis we derive

$$[\![\pi_1(N)]\!]^{w_\Gamma}_{\rho^\star} = Proj_{1,w_\Gamma}([\![N]\!]^{w_\Gamma}_{\rho^\star}) = Proj_{1,w_\Gamma}([N]_\Gamma) = [\pi_1(N)]_\Gamma.$$

The case when the term M is the second projection $\pi_2(N)$ proceeds analogously.

If the term M is the left injection $\mathrm{in}_1(N)$, then by Definition 3.14, Definition 3.22 and the induction hypothesis we have

$$[\![\mathrm{in}_1(N)]\!]^{w_\Gamma}_{\rho^\star} = Inl_{w_\Gamma}([\![N]\!]^{w_\Gamma}_{\rho^\star}) = Inl_{w_\Gamma}([N]_\Gamma) = [\mathrm{in}_1(N)]_\Gamma.$$

The case when the term M is the right injection $\mathrm{in}_2(N)$ proceeds analogously.

Finally, if M is the empty pair $\langle\rangle$, then

$$[\![\langle\rangle]\!]^{w_\Gamma}_{\rho^\star} = 1_{w_\Gamma} = [\langle\rangle]_\Gamma,$$

since we have showed in the proof of Lemma 3.23 that $[\langle\rangle]_\Gamma$ is the unique element of $A^1_{w_\Gamma}$.

□

We may now show the other direction of Theorem 3.19. First, we prove that the equational theory given in Figure 3.1 is complete with respect to the proposed semantics.

Theorem 3.25. *Let M and N be terms and Γ a consistent basis. If for every Kripke CL model $\mathcal{K}_\rho$, such that $\mathcal{K}_\rho \models \Gamma$, and a world w of the model we have $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$, then $M =_\Gamma N$.*

Proof. Let us assume that for every Kripke CL model $\mathcal{K}_\rho$ such that $\mathcal{K}_\rho \models \Gamma$ and world w of the model, $[\![M]\!]^w_\rho = [\![N]\!]^w_\rho$ holds. We consider a canonical model $\mathcal{K}^\Gamma_{\rho^\star}$. As a direct consequence of Definition 3.22 we have $\mathcal{K}^\Gamma_{\rho^\star} \models \Gamma$. From the assumption we conclude that for a world w_Γ of model $\mathcal{K}^\Gamma_{\rho^\star}$, $[\![M]\!]^{w_\Gamma}_{\rho^\star} = [\![N]\!]^{w_\Gamma}_{\rho^\star}$ and by Lemma 3.24 we obtain $[M]_\Gamma = [N]_\Gamma$, i.e. $M =_\Gamma N$. □

We now state the main result of this chapter, the completeness of full simply typed combinatory logic with respect to the proposed semantics.

Theorem 3.26. *Let Γ be a consistent basis. If $\Gamma \models M : \sigma$, then $\Gamma \vdash M : \sigma$.*

Proof. Suppose that, for a consistent basis Γ, we have $\Gamma \models M : \sigma$. For a canonical model $\mathcal{K}^\Gamma_\rho$, it holds that $\mathcal{K}^\Gamma_{\rho^\star} \models \Gamma$, as stated in the proof of Theorem 3.25. Thus, in the world w_Γ of the model $\mathcal{K}^\Gamma_{\rho^\star}$ we have $[\![M]\!]^{w_\Gamma}_{\rho^\star} \in A^\sigma_{w_\Gamma}$. By Definition 3.22 and Lemma 3.24 we derive

$$[M]_\Gamma = [\![M]\!]^{w_\Gamma}_{\rho^\star} \in A^\sigma_{w_\Gamma} = \{[N]_\Gamma \mid N \in CL^{\to,\times,+} \text{ and } \Gamma \vdash N : \sigma\}.$$

So, we conclude $\Gamma \vdash M : \sigma$. □

3.4 Concluding remarks

In this chapter, the full simply typed combinatory logic is studied and the Kripke-style semantics of the full simply typed combinatory logic is presented.

The full simply typed combinatory logic is the simply typed combinatory logic extended with product types, sum types, the empty type and the unit type. Via the Curry-Howard correspondence, the full simply typed combinatory logic is related to the full intuitionistic logic (with all the connectives). Since the intuitionistic logic is sound and complete with respect to the Kripke semantics, the question of the soundness and completeness of the full simply typed combinatory logic naturally arises. This is not the first time that Kripke-style semantics is employed as a semantics for the model of computation. In [124], the authors introduced the Kripke-style semantics of the simply typed λ-calculus and proved the soundness and completeness of the simply typed λ-calculus with respect to the proposed semantics. Motivated by their work we have introduced the Kripke-style semantics of the full simply typed calculi (λ-calculus and combinatory logic) in [94].

This chapter presents the Kripke-style semantics introduced in [94], but for a different calculus. The full simply typed λ-calculus and combinatory logic studied in [94] did not include a typing rule which ensures that the equal terms inhabit the same type. The calculus without this typing rule cannot be complete with respect to the proposed semantics. In order to obtain a sound and complete calculus, we added the typing rule for equal terms. The main results of the chapter are soundness and completeness of the full simply typed combinatory logic with respect to the proposed Kripke-style semantics.

At the end, we give some open questions and the ideas for the future work.

- The next step is to employ the presented Kripke-style semantics as a semantics of the full simply typed λ-calculus. As it has been discussed, the translation of the full simply typed λ-calculus with the typing rule for equal terms into the combinatory logic with the same rule is more involved than in the case without the typing rule for equal terms. Thus, we leave this for the future work.

- Another possibility to further develop this approach to different frameworks in logic and computation, e.g. polymorphic types.

- Kripke-style semantics has found its application in various fields. One of the recent applications of Kripke-style semantics is in the blockchain technology. In [118], Marinković et al. introduce a temporal epistemic logic with probabilities as an extension of temporal epistemic logic and use this framework to model and reason about probabilistic properties of the blockchain protocol. The semantics of the logic is defined as a

Kripke-style semantics. Further, in [24] Brünnler et al. introduce BCL, a dynamic logic to reason about blockchain updates and the semantics of the logic is again Kripke-style semantics. Kripke models for modal logic have been used by Hirai in [81] for analysing the protocol of blockchain. We see that Kripke-style semantics is a structure suitable for describing and analysing properties of blockchain, so it is a promising research topic.

Chapter 4

Logic of combinatory logic

In this chapter, we present the logic of combinatory logic, a propositional extension of the simply typed combinatory logic, which has been introduced in [95, 97]. We give its syntax, axiomatization and semantics, and prove that the axiomatization is sound and complete with respect to the proposed semantics.

The importance of the logic of combinatory logic can be seen from different perspectives. From the perspective of reasoning, by defining the classical propositional logic over the simply typed combinatory logic we obtain a formal system for reasoning about type assignment statements. As a consequence, we can employ the well-established methods such as DPLL, resolution method, SAT solvers and SMT solvers to reason about the simply typed combinatory logic. Due to the Curry-Howard correspondence this is a first step towards the development of a tool for automated reasoning about combinatory terms and programs. From the perspective of programming language theory, we can see the logic of combinatory logic as a convenient framework for abstract syntax: the structure of programming languages, disregarding the superficial details of concrete syntax. From the proof-theoretic perspective, we have a precise correspondence between syntactic and semantic structure: an internal language for reasoning about the simply typed combinatory logic is given by the rules of classical propositional logic.

Contribution of the chapter

- We develop a framework for reasoning about simply typed terms.
- The introduced logic LCL is an extension of the simply typed combinatory logic.

- We prove that the equational theory of the simply typed combinatory logic is sound and complete with respect to the semantics of LCL.
- The axiomatization of the logic LCL is proved to be sound and complete with respect to the proposed semantics.
- The proposed semantics is a new semantics of the simply typed combinatory logic. We give the proof of soundness and completeness of the simply typed combinatory logic extended with the typing rule that ensures that equal terms inhabit the same type.

Overview of the chapter We start this chapter with preliminaries for the simply typed combinatory logic in Section 4.1. In Section 4.2 we present the logic of combinatory logic by giving its syntax in Section 4.2.1, axiomatization in Section 4.2.2 and semantics in Section 4.2.3. Section 4.3 presents the proofs of soundness and completeness of the equational theory with respect to the proposed semantics. The main results, namely the soundness and strong completeness of the given axiomatization with respect to the proposed semantics, are given in Section 4.4.

4.1 Simply typed combinatory logic

In this section, we present the simply typed combinatory logic $CL_{\to}$ ([11, 21, 78]), through its syntax, equational theory and type assignment system.

We start with the syntax of untyped combinatory logic CL. The set of terms of CL is built up from a countable set of term variables $V = \{x, y, z, \ldots, x_1, \ldots\}$ and a set of term constants $\{\mathsf{S}, \mathsf{K}, \mathsf{I}\}$, and it is generated by the following grammar

$$\boxed{M ::= x \mid \mathsf{S} \mid \mathsf{K} \mid \mathsf{I} \mid MM} \tag{4.1}$$

where x is a term variable. Term constants S, K, and I are called primitive combinators. Other well-known combinators such as $\mathsf{B}, \mathsf{W}, \mathsf{C}, \ldots$ are described in [21]. The set of all untyped terms is denoted by CL and we let $M, N, \ldots, M_1, \ldots$ range over CL. As in the previous chapters, the set of variables that occur in a term M is denoted by $FV(M)$, and the substitution of a term N for the occurrences of variable x in a term M is denoted by $M\{N/x\}$. A subterm of a term M is any term that is a part of M. For example, the term $\mathsf{K}x(\mathsf{IS})$ is a subterm of the term $(\mathsf{S}(\mathsf{K}x(\mathsf{IS})))(\mathsf{II})$.

Primitive combinators represent functions, so they are characterized by the following rewriting rules:

$$\begin{aligned} \mathsf{S}MNL &\to (ML)(NL) \\ \mathsf{K}MN &\to M \\ \mathsf{I}M &\to M \end{aligned}$$

Terms of the form $\mathsf{S}MNL, \mathsf{K}MN, \mathsf{I}M$ are called redexes. The rewriting rules presented above induce relations between CL terms, which are an important aspect of the combinatory logic. We mainly focus on the equational theory that arises from these rules and in order to formally introduce this theory, we define one-step reduction.

Definition 4.1. *If N is a redex such that $N \to N'$ and N is a subterm of a term M, then $M \triangleright_1 M'$, where M' is obtained from M by replacing the subterm N with N'. The relation $\triangleright_1$ is called* the one-step reduction.

The weak reduction, denoted by $\triangleright_w$, is the reflexive and transitive closure of the one-step reduction $\triangleright_1$. The one-step reduction also induces an equivalence relation, called the weak equality.

Definition 4.2. *The weak equality, denoted by $=_w$, is the reflexive, transitive and symmetric closure of the reduction $\triangleright_1$. It can be characterized inductively as follows:*

1. *If $M \triangleright_1 N$, then $M =_w N$.*
2. *If M is a CL-term, then $M =_w M$.*
3. *If $M =_w N$, then $N =_w M$.*
4. *If $M =_w N$ and $N =_w L$, then $M =_w L$.*

Another equivalence relation obtained from the weak reduction is *the extensional weak equality*, denoted by $=_{w,\eta}$, and introduced in Definition 4.3 below.

Definition 4.3 ([21])**.** *CL-terms Q and $Q\{M_1/N_1, \ldots, M_m/N_m\}$ are extensionally weakly equal, denoted by $Q =_{w,\eta} Q\{M_1/N_1, \ldots, M_m/N_m\}$ if and only if for every $i \in \{1, \ldots, m\}$, there exists n_i such that for a series of distinct variables $x_{i,1}, x_{i,2}, \ldots, x_{i,n_i}$, that occur neither in M_i nor in N_i, there exists a CL-term P_i with $M_i x_{i,1}, x_{i,2}, \ldots x_{i,n_i} \triangleright_w P_i$ and $N_i x_{i,1} x_{i,2} \ldots x_{i,n_i} \triangleright_w P_i$.*

An alternative way to formally introduce weak equality and extensional weak equality is axiomatically.

$$M = M \quad (id) \qquad \mathsf{S}MNL = (ML)(NL) \quad (S)$$

$$\mathsf{K}MN = M \quad (K) \qquad \mathsf{I}M = M \quad (I)$$

$$\frac{M = N}{N = M}\ (\text{sym}) \qquad \frac{M = N \qquad N = L}{M = L}\ (\text{trans})$$

$$\frac{M = N}{MP = NP}\ (\text{app-l}) \qquad \frac{M = N}{PM = PN}\ (\text{app-r})$$

FIGURE 4.1: Equational theory $\mathcal{EQ}$

Definition 4.4. *The equational theory $\mathcal{EQ}$ is given by the axioms and rules in Figure 4.1.*

If $M = N$ can be derived from the set of axioms and rules in Figure 4.1, then we say that CL-terms M and N are equal. The equational theory $\mathcal{EQ}$ introduces a new equivalence relation on the set of CL-terms, which coincides with the weak equality. This is stated in the following proposition, for which the respective proof can be found in [21].

Proposition 4.5 ([21])**.** *An equation $M = N$ is provable in $\mathcal{EQ}$ if and only if $M =_w N$.*

The theory obtained by extending the equational theory $\mathcal{EQ}$ with the rule

$$\frac{Mx = Nx \qquad x \notin FV(M) \cup FV(N)}{M = N}\ (\text{ext})$$

is called *the extensional equational theory* and it is denoted by $\mathcal{EQ}^\eta$. We write $M = N$, when $M = N$ can be derived in the equational theory $\mathcal{EQ}^\eta$. The equivalence class of a term M with respect to the equivalence relation generated by $\mathcal{EQ}^\eta$ is denoted by $[M]$, i.e.

$$[M] = \{N \mid M = N \text{ is provable in } \mathcal{EQ}^\eta\}$$

The equivalence relation generated by the extensional equational theory $\mathcal{EQ}^\eta$ coincides with the extensional weak equality.

Proposition 4.6 ([21])**.** *An equation $M = N$ is provable in $\mathcal{EQ}^\eta$ if and only if $M =_{w,\eta} N$.*

In this chapter, we focus on the equational theory $\mathcal{EQ}^\eta$. In [21], the inequational theory, which is related to the weak reduction, has also been studied.

This chapter focuses on the simply typed terms. Simple types are built up from a countable set of type variables $V_{\mathsf{Type}} = \{a, b, c, ..., a_1, \ldots\}$ using only one type operator $\to$, called *the functional type*. The set of all simple types is denoted by $\mathsf{Types}_{\to}$ and is generated by the following grammar

$$\boxed{\sigma ::= a \mid \sigma \to \sigma} \tag{4.2}$$

We let $\sigma, \tau, \ldots, \sigma_1, \ldots$ range over $\mathsf{Types}_{\to}$. The notions of type statement, declaration, basis and the domain of the basis are defined as in Definition 3.1 (Chapter 3).

The type assignment system for *the simply typed combinatory logic*, denoted by $CL_{\to}$, is given in Figure 4.2. We say that a term M can be typed with a type σ in a basis Γ, or that $M : \sigma$ is derivable from Γ, if we can derive the typing judgment $\Gamma \vdash_{\mathrm{CL}} M : \sigma$ by the rules of Figure 4.2. The set of all statements $M : \sigma$ that are typable from some basis Γ is also denoted by $CL_{\to}$, i.e.

$$CL_{\to} = \{M : \sigma \mid \exists \Gamma \text{ such that } \Gamma \vdash_{\mathrm{CL}} M : \sigma\}.$$

$$\Gamma, x : \sigma \vdash_{\mathrm{CL}} x : \sigma \quad (\text{axiom} \in)$$

$$\Gamma \vdash_{\mathrm{CL}} \mathsf{S} : (\sigma \to (\rho \to \tau)) \to (\sigma \to \rho) \to (\sigma \to \tau) \quad (\text{axiom } S)$$

$$\Gamma \vdash_{\mathrm{CL}} \mathsf{K} : \sigma \to (\tau \to \sigma) \quad (\text{axiom } K) \qquad \Gamma \vdash_{\mathrm{CL}} \mathsf{I} : \sigma \to \sigma \quad (\text{axiom } I)$$

$$\frac{\Gamma \vdash_{\mathrm{CL}} M : \sigma \to \tau \qquad \Gamma \vdash_{\mathrm{CL}} N : \sigma}{\Gamma \vdash_{\mathrm{CL}} MN : \tau} \ (\to \text{elim})$$

FIGURE 4.2: Type assignment system $CL_{\to}$

At the end of this section, we give some auxiliary properties of the typing system. The proofs follow by the induction on the length of derivations and are omitted.

Proposition 4.7. *If* $\Gamma \vdash_{CL} M : \sigma$ *and* $\Gamma \subseteq \Gamma'$ *then* $\Gamma' \vdash_{CL} M : \sigma$.

For a set X of term variables, we write $\Gamma \restriction X = \{x : \sigma \in \Gamma \mid x \in X\}$.

Proposition 4.8. *1. If* $\Gamma \vdash_{CL} M : \sigma$, *then* $FV(M) \subseteq dom(\Gamma)$.

2. If $\Gamma \vdash_{CL} M : \sigma$, *then* $\Gamma \restriction FV(M) \vdash_{CL} M : \sigma$.

4.2 Logic of combinatory logic

In this section, we formally introduce the propositional extension of the simply typed combinatory logic, called *the logic of combinatory logic* and denoted by LCL. The logic of combinatory logic is obtained by extending the simply typed combinatory logic with classical propositional connectives of negation and implication. We introduce LCL through its syntax, axiomatization and semantics.

4.2.1 Syntax LCL

The language of LCL obtained from the language of the simply typed combinatory logic and the language of the classical propositional logic. The set of LCL-formulas is the set of all type statements that are typable from some basis Γ closed under the propositional connectives (negation and implication) and it is given by the following grammar:

$$\boxed{\alpha := M : \sigma \mid \neg\alpha \mid \alpha \Rightarrow \alpha} \tag{4.3}$$

where $M : \sigma \in CL_{\rightarrow}$. We let $\alpha, \beta, \ldots$ range over the set of LCL-formulas. Other classical propositional connectives $\wedge, \vee, \Leftrightarrow$ are defined in the standard way.

$$\begin{aligned} \alpha \vee \beta &\text{ stands for } \neg\alpha \Rightarrow \beta, \\ \alpha \wedge \beta &\text{ stands for } \neg(\neg\alpha \vee \neg\beta), \\ \alpha \Leftrightarrow \beta &\text{ stands for } (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha). \end{aligned}$$

A formula $\alpha \wedge \neg\alpha$ is denoted by $\bot$.

The logic of combinatory logic is a step towards formalization of meta-language of the simply typed combinatory logic. In the simply typed combinatory logic, if a term M has type $\sigma \rightarrow \tau$ and a term N has type σ in some basis Γ, then MN is typable in Γ with the type τ. We can formally express this in LCL with formula

$$(M : \sigma \rightarrow \tau \wedge N : \sigma) \Rightarrow MN : \tau.$$

4.2.2 Axiomatization of LCL

This subsection presents the axiomatic system of the logic of combinatory logic LCL, which is obtained from the type assignment system for simply typed combinatory logic and the axiomatic system of classical propositional logic. The axiomatic system of LCL is given by the axiom schemes and inference rule in Figure 4.3.

Axiom schemes:

(1) $\mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$
(2) $\mathsf{K} : \sigma \to (\tau \to \sigma)$
(3) $\mathsf{I} : \sigma \to \sigma$
(4) $(M : \sigma \to \tau) \Rightarrow ((N : \sigma) \Rightarrow (MN : \tau))$,
$M : \sigma \to \tau, N : \sigma, MN : \tau \in CL_{\to}$
(5) $M : \sigma \Rightarrow N : \sigma$, if $M = N$, $M : \sigma, N : \sigma \in CL_{\to}$
(6) $\alpha \Rightarrow (\beta \Rightarrow \alpha)$
(7) $(\alpha \Rightarrow (\beta \Rightarrow \gamma)) \Rightarrow ((\alpha \Rightarrow \beta) \Rightarrow (\alpha \Rightarrow \gamma))$
(8) $(\neg\alpha \Rightarrow \neg\beta) \Rightarrow ((\neg\alpha \Rightarrow \beta) \Rightarrow \alpha)$

Inference rule:

$$\frac{\alpha \Rightarrow \beta \qquad \alpha}{\beta}\ (\mathrm{MP})$$

FIGURE 4.3: Axiom schemes and inference rules for LCL

The axiomatic system of LCL comprises eight axiom schemes, which can be classified as follows.

- The first three axiom schemes, (1), (2) and (3), are non-logical and correspond to the axioms for typing primitive combinators.
- The axiom schemes (4) and (5) correspond to the typing rules of the simply typed combinatory logic. The axiom scheme (4) corresponds to the rule ($\to$ *elim*) for typing an application and the axiom scheme (5) corresponds to the rule that guarantees that equal terms have the same type. The condition $M = N$ in the axiom scheme (5) requires that $M = N$ is provable in the equational theory $\mathcal{EQ}^{\eta}$. This rule is not included in the typing system in Figure 4.2. In order to obtain the completeness of the type assignment system, it is necessary to add this rule to the original system as it will be discussed in Section 4.5.
- The last three axiom schemes, (6), (7) and (8), are logical axioms from the axiomatic system of the classical propositional logic.

The axiomatic system of LCL also includes one inference rule , the classical Modus Ponens (MP). The notion of a proof in LCL is formally introduced in the next definition, followed by an example, which illustrates a proof in LCL.

Definition 4.9. *Let T be a set of LCL-formulas and α an LCL-formula. A formula α can be derived from T, denoted by $T \vdash \alpha$, if there exists a sequence of formulas $\alpha_0, \alpha_1, \ldots, \alpha_n$ such that α_n is the formula α and for every $i \in \{0, 1, \ldots, n\}$, α_i is either an axiom instance, or $\alpha_i \in T$, or α_i is a formula which can be derived by the inference rule (MP) applied to some previous members of the sequence.*

A formula α is a theorem, denoted by $\vdash \alpha$, if it is deducible from the empty set, i.e. $\emptyset \vdash \alpha$.

If an LCL-formula α can be derived from a set T using only axiom schemes $6 - 8$ and inference rule (MP), then we say that $T \vdash \alpha$ is obtained by propositional reasoning.

Example 4.10. *We prove that if $M : \sigma \to \tau$, $N : \sigma$ and $MN : \tau$ are LCL-formulas, then $M : \sigma \to \tau, N : \sigma \vdash MN : \tau$. We write the proof as the sequence of formulas, starting with the formulas $M : \sigma \to \tau$ and $N : \sigma$ as hypotheses.*

1. $M : \sigma \to \tau$, *hypothesis*
2. $N : \sigma$, *hypothesis*
3. $(M : \sigma \to \tau) \Rightarrow ((N : \sigma) \Rightarrow (MN : \tau))$ *an instance of (4)*
4. $(N : \sigma) \Rightarrow (MN : \tau)$, *from* (1) *and* (3) *by rule (MP)*
5. $MN : \tau$, *from* (2) *and* (4) *by rule (MP)*

We may notice that for almost every axiom scheme and typing rule in Figure 4.2, except (axiom $\in$), there is an axiom scheme in Figure 4.3. Although in the axiomatic system of LCL there is no corresponding axiom scheme for (axiom $\in$), by Definition 4.9 we can derive $\Gamma, x : \sigma \vdash_{\mathrm{CL}} x : \sigma$. As we have already discussed, the logic LCL is an extension of the simply typed combinatory logic and this is stated in the following proposition.

Proposition 4.11. *If $\Gamma \vdash_{CL} M : \sigma$, then $\Gamma \vdash M : \sigma$.*

Proof. The proof is by induction on the derivation of $\Gamma \vdash_{\mathrm{CL}} M : \sigma$ in $CL_{\to}$. □

The notion of a consistent set is introduced in the previous chapter, when a consistent basis is defined and compared with a consistent set in the intuitionistic propositional logic. In the logic LCL, the definition of a consistent set is equivalent to the definition of a consistent set in the classical propositional logic.

Definition 4.12. *A set of LCL-formulas T is consistent if there exists at least one formula which is not derivable from T. Otherwise, T is inconsistent.*

Alternatively, we say that T is inconsistent if and only if $T \vdash \bot$. We will write $T, \alpha \vdash \beta$ for $T \cup \{\alpha\} \vdash \beta$.

A key step in proving the strong completeness of the axiomatic system of LCL is the proof of Deduction theorem.

Theorem 4.13 (Deduction theorem)**.** *Let T be a set of formulas and α, β formulas of LCL. If $T, \alpha \vdash \beta$, then $T \vdash \alpha \Rightarrow \beta$.*

Proof. The proof is by induction on the length n of the proof of $T, \alpha \vdash \beta$. For the base case suppose that $n = 1$. Then β is either an axiom or $\beta \in T \cup \{\alpha\}$.

Let β be an axiom. The proof proceeds as follows:

1. $T \vdash \beta$, since β is an axiom
2. $T \vdash \beta \Rightarrow (\alpha \Rightarrow \beta)$, since $\beta \Rightarrow (\alpha \Rightarrow \beta)$ is an instance of (6)
3. $T \vdash \alpha \Rightarrow \beta$, from (1) and (2) by rule (MP)

Next, let $\beta \in T$. Then

1. $T \vdash \beta$, since $\beta \in T$
2. $T \vdash \beta \Rightarrow (\alpha \Rightarrow \beta)$, since $\beta \Rightarrow (\alpha \Rightarrow \beta)$ is an instance of (6)
3. $T \vdash \alpha \Rightarrow \beta$, from (1) and (2) by rule (MP)

If $\beta = \alpha$, then $T \vdash \alpha \Rightarrow \alpha$, since $\alpha \Rightarrow \alpha$ is a theorem in LCL.

Let us assume that the statement holds if the proof length is less than k and consider the case when the proof length is $k > 1$. In the case that β is an axiom or belongs to the set $T \cup \{\alpha\}$ the proof proceeds as above. Assume the formula β is derived from $T \cup \{\alpha\}$ by an application of inference rule (MP). If β is obtained by an application of rule (MP) on formulas α_1 and $\alpha_1 \Rightarrow \beta$ such that $T, \alpha \vdash \alpha_1$ and $T, \alpha \vdash \alpha_1 \Rightarrow \beta$, then the derivation lengths of $T, \alpha \vdash \alpha_1$ and $T, \alpha \vdash \alpha_1 \Rightarrow \beta$ are less than k. By the induction hypothesis we have $T \vdash \alpha \Rightarrow \alpha_1$ and $T \vdash \alpha \Rightarrow (\alpha_1 \Rightarrow \beta)$. The proof proceeds as follows:

1. $T \vdash \alpha \Rightarrow \alpha_1$, by the induction hypothesis
2. $T \vdash \alpha \Rightarrow (\alpha_1 \Rightarrow \beta)$, by the induction hypothesis
3. $T \vdash (\alpha \Rightarrow (\alpha_1 \Rightarrow \beta)) \Rightarrow ((\alpha \Rightarrow \alpha_1) \Rightarrow (\alpha \Rightarrow \beta))$, by (7)
4. $T \vdash (\alpha \Rightarrow \alpha_1) \Rightarrow (\alpha \Rightarrow \beta)$, from (2) and (3) by rule (MP)
5. $T \vdash \alpha \Rightarrow \beta$, from (1) and (4) by rule (MP)

This concludes the proof. □

4.2.3 Semantics of LCL

In this section, we introduce a semantics of the logic of combinatory logic.

Dana Scott introduced the first models for the untyped combinatory logic in [156, 157]. Since then different models for both the untyped and typed combinatory logic have been proposed, starting with *a term model* introduced by Barendregt ([9, 12]) , which is constructed from the language of the combinatory logic. The algebraic and set theoretical semantics of dual and symmetric combinatory calculi have been introduced by Bimbó in [20]. Additionally, in [21] Bimbó has presented several models for the combinatory logic such as the operational models for the untyped combinatory logic and two models for the typed combinatory logic: combinatory algebras and relational models. More models can be found in [2, 11, 13].

Building models for the typed combinatory logic from the models of the untyped combinatory logic is a standard approach, and in that case types are interpreted as subsets of the untyped model. Over the past decades, different approaches have been proposed such as the ones introduced in [124] and [94]. The semantics introduced in [124] is a Kripke-style semantics, which interprets a typing judgment $\Gamma \vdash M : \sigma$, considering term M and its type σ at the same type. Although the main result of [124] is Kripke-style semantics for the simply typed λ-calculus, they can also be employed as semantics of the simply typed combinatory logic. In fact, some features of the semantics are proved using a translation of the λ-calculus into the combinatory logic. This approach and the general concept of a model of untyped calculus were combined in [94] with the goal to define Kripke-style semantics of the full simply typed λ-calculus and combinatory logic. The Kripke-style semantics of the full simply typed combinatory logic is presented in Chapter 3. As it has been discussed in the previous chapter, the interpretation of a term is defined without considering its type, which is the main difference with respect to the semantics introduced in [124]. The semantics of the logic of combinatory logic is not a Kripke-style semantics, but it has been inspired by the semantics introduced in [124] and [94]. Furthermore, we have used the similar approach to define the semantics of the combinatory logic with intersection types in [62].

In order to define semantics of LCL, the notion of an applicative structure for LCL is introduced.

Definition 4.14. *An applicative structure for LCL is a tuple*

$$\mathcal{M} = \langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i} \rangle$$

where

- *D is a nonempty set, called* domain.

- $\{A^\sigma\}_\sigma = \{A^\sigma\}_{\sigma \in \mathsf{Types}_\to}$ *is a family of sets indexed by* σ *such that* $A^\sigma \subseteq D$ *for all* σ*,*
- $\cdot$ *is a binary operation on* D*, i.e.* $\cdot : D \times D \to D$*, which is extensional: for* $d_1, d_2 \in D$*,* $d_1 = d_2$ *whenever* $(\forall e \in D)(d_1 \cdot e = d_2 \cdot e)$*, and it holds that the codomain of the restriction of function* $\cdot$ *to the set* $A^{\sigma\to\tau} \times A^\sigma$ *is* A^τ *for every* $\sigma, \tau \in \mathsf{Types}_\to$*.*
- $\boldsymbol{s} \in D$ *such that for every* $\sigma, \tau, \rho \in \mathsf{Types}_\to$

$$\boldsymbol{s} \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))} \tag{4.4}$$

and for every $d, e, f \in D$*,*

$$((\boldsymbol{s} \cdot d) \cdot e) \cdot f = (d \cdot f) \cdot (e \cdot f) \tag{4.5}$$

- $\boldsymbol{k} \in D$ *such that for every* $\sigma, \tau \in \mathsf{Types}_\to$

$$\boldsymbol{k} \in A^{\sigma\to(\tau\to\sigma)} \tag{4.6}$$

and for every $d, e \in D$*,*

$$(\boldsymbol{k} \cdot d) \cdot e = d \tag{4.7}$$

- $\boldsymbol{i} \in D$ *such that for every* $\sigma \in \mathsf{Types}_\to$

$$\boldsymbol{i} \in A^{\sigma\to\sigma} \tag{4.8}$$

and for every $d \in D$*,*

$$\boldsymbol{i} \cdot d = d \tag{4.9}$$

The uniqueness of the elements $\boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}$ is a consequence of the extensionality of the operation $\cdot$. We provide an applicative structure with a valuation of term variables, called *an environment.*

Definition 4.15. *Let* $\mathcal{M}$ *be an applicative structure. An environment* ρ *for* $\mathcal{M}$ *is a map from the set of term variables to the domain of the applicative structure* $\mathcal{M}$*,* $\rho : V \to D$*.*

Let $\mathcal{M}$ be an applicative structure, ρ an environment for $\mathcal{M}$ and d an element from the domain of $\mathcal{M}$. We write $\rho(x := d)$ to denote an environment for $\mathcal{M}$ that is identical to ρ on all variables except x, i.e.

$$\rho(x := d)(y) = \begin{cases} d, & y = x \\ \rho(y), & y \neq x \end{cases}.$$

Definition 4.16. *An LCL-model is a tuple $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$, where $\mathcal{M}$ is an applicative structure and ρ is an environment for $\mathcal{M}$.*

An environment interprets term variables. The interpretation of terms is defined by extending the environment ρ to the interpretation map $[\![\cdot]\!]_\rho$.

Definition 4.17. *Let $\mathcal{M}$ be an applicative structure and ρ an environment for $\mathcal{M}$.* The interpretation (meaning) of a term M *in the environment ρ, denoted by $[\![M]\!]_\rho$, is defined inductively as follows:*

- $[\![x]\!]_\rho = \rho(x)$,
- $[\![\mathsf{S}]\!]_\rho = \boldsymbol{s}$,
- $[\![\mathsf{K}]\!]_\rho = \boldsymbol{k}$,
- $[\![\mathsf{I}]\!]_\rho = \boldsymbol{i}$,
- $[\![MN]\!]_\rho = [\![M]\!]_\rho \cdot [\![N]\!]_\rho$.

The interpretation map $[\![\cdot]\!]_\rho$ is a total mapping from the set of all CL-terms to the domain of an applicative structure. The existence of elements $\boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}$ in an applicative structure guarantees that the interpretation map is well defined, i.e. that the interpretation of every CL-term in an environment ρ is defined.

The following lemmas present some auxiliary results. The proofs proceed by induction on the structure of term and are omitted. The first result shows that the interpretation of a term depends only on variables that occur in the term.

Lemma 4.18. *Let $\mathcal{M}$ be an applicative structure, ρ_1 and ρ_2 two environments for $\mathcal{M}$ and M a CL-term. If $\rho_1(x) = \rho_2(x)$ for every x that occurs in M, then $[\![M]\!]_{\rho_1} = [\![M]\!]_{\rho_2}$.*

The second result gives the interpretation of a term obtained by a substitution.

Lemma 4.19 (Substitution lemma)**.** *Let M, N be CL-terms and ρ an environment. Then*

$$[\![M\{N/x\}]\!]_\rho = [\![M]\!]_{\rho(x:=[\![N]\!]_\rho)}.$$

Next, we introduce the notion of satisfiability of a formula in a model.

Definition 4.20. The satisfiability of an LCL-formula in a model $\mathcal{M}_\rho$ *is defined as follows:*

- $\mathcal{M}_\rho \models M : \sigma$ *if and only if* $[\![M]\!]_\rho \in A^\sigma$.

- $\mathcal{M}_\rho \models \alpha \wedge \beta$ *if and only if* $\mathcal{M}_\rho \models \alpha$ *and* $\mathcal{M}_\rho \models \beta$.
- $\mathcal{M}_\rho \models \neg\alpha$ *if and only if it is not true that* $\mathcal{M}_\rho \models \alpha$.

The satisfiability of a set of formulas and the notion of semantical consequence are defined as in Chapter 3. To make this chapter self-contained we revisit the definition.

Definition 4.21. *1. A model $\mathcal{M}_\rho$ satisfies a set T of LCL-formulas, denoted by $\mathcal{M}_\rho \models T$, if and only if $\mathcal{M}_\rho \models \alpha$ for every $\alpha \in T$.*

2. An LCL-formula α is a semantical consequence of a set T, denoted by $T \models \alpha$, if and only if $\mathcal{M}_\rho \models T$ implies $\mathcal{M}_\rho \models \alpha$.

4.3 Soundness and completeness of the equational theory

In this section, we prove that the equational theory $\mathcal{EQ}^\eta$, given in Section 4.1, is sound and complete with respect to the semantics of LCL introduced in Section 4.2.3.

In every LCL-model, the interpretation of every CL-term is defined. Naturally, we expect that equal terms have equal interpretations in every model and this is the first result we present in this section (Theorem 4.22). Another question that arises is whether two terms that have equal interpretations in every model are equal. The positive answer to this question is proved in Theorem 4.23.

Theorem 4.22 (Soundness of $\mathcal{EQ}^\eta$)**.** *If $M = N$ in provable in $\mathcal{EQ}^\eta$, then $[\![M]\!]_\rho = [\![N]\!]_\rho$ for any LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$.*

Proof. The proof is by induction on the length of the proof of $M = N$.

In the case $M = N$ is obtained from axiom (id), terms M and N are identical and we have $[\![M]\!]_\rho = [\![N]\!]_\rho$, for every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$.

If $M = N$ falls under axiom (S), then the term M is of the form $((\mathsf{S}P)Q)R$ and the term N is the term $(PR)(QR)$ for some terms P, Q, R. By Definition 4.14 and Definition 4.17 we derive

$$\begin{aligned}
[\![((\mathsf{S}P)Q)R]\!]_\rho &= (([\![\mathsf{S}]\!]_\rho \cdot [\![P]\!]_\rho) \cdot [\![Q]\!]_\rho) \cdot [\![R]\!]_\rho \\
&= ((\boldsymbol{s} \cdot [\![P]\!]_\rho) \cdot [\![Q]\!]_\rho) \cdot [\![R]\!]_\rho \\
&= ([\![P]\!]_\rho \cdot [\![R]\!]_\rho) \cdot ([\![Q]\!]_\rho \cdot [\![R]\!]_\rho) \\
&= [\![(PR)\,(QR)]\!]_\rho.
\end{aligned}$$

Similarly, in the case that $M = N$ falls under axiom (K), term M is an application $\mathsf{K}PQ$ and N is the term P, for some terms P, Q. Again, by Definition 4.14 and Definition 4.17 we obtain

$$[\![\mathsf{K}PQ]\!]_\rho = ([\![\mathsf{K}]\!]_\rho \cdot [\![P]\!]_\rho) \cdot [\![Q]\!]_\rho = (\boldsymbol{k} \cdot [\![P]\!]_\rho) \cdot [\![Q]\!]_\rho = [\![P]\!]_\rho.$$

Similarly, if $M = N$ is obtained by axiom (I), then M is of the form $\mathsf{I}P$, N is a term P, for some term P, and we derive

$$[\![\mathsf{I}P]\!]_\rho = [\![\mathsf{I}]\!]_\rho \cdot [\![P]\!]_\rho = \boldsymbol{i} \cdot [\![P]\!]_\rho = [\![P]\!]_\rho.$$

Let $M = N$ be obtained from $N = M$ by rule (sym). We obtain $[\![N]\!]_\rho = [\![M]\!]_\rho$ for every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$ by the induction hypothesis. Consequently, it holds that $[\![M]\!]_\rho = [\![N]\!]_\rho$.

If $M = N$ is obtained from $M = P$ and $P = N$ by the rule (trans), then by the induction hypothesis we have $[\![M]\!]_\rho = [\![P]\!]_\rho$ and $[\![P]\!]_\rho = [\![N]\!]_\rho$, for every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$. We conclude that $[\![M]\!]_\rho = [\![N]\!]_\rho$.

If $M = N$ is obtained from $L = Q$ by rule (app-l) , then the term M is of the form LP and the term N is of the form QP, for some term P. For every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$, it holds that $[\![L]\!]_\rho = [\![Q]\!]_\rho$ by the induction hypothesis. Now, by Definition 4.17, we obtain

$$[\![LP]\!]_\rho = [\![L]\!]_\rho \cdot [\![P]\!]_\rho = [\![Q]\!]_\rho \cdot [\![P]\!]_\rho = [\![QP]\!]_\rho.$$

In the case when $M = N$ is obtained from $L = Q$ by rule (app-r), the terms M and N are of the form PL and PQ, respectively, for some term P. For every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$, we have that $[\![L]\!]_\rho = [\![Q]\!]_\rho$ by the induction hypothesis. Further, by Definition 4.17, we derive

$$[\![PL]\!]_\rho = [\![P]\!]_\rho \cdot [\![L]\!]_\rho = [\![P]\!]_\rho \cdot [\![Q]\!]_\rho = [\![PQ]\!]_\rho.$$

Finally, assume that $M = N$ is obtained by rule (ext) from $Mx = Nx$ where the variable x appears neither in M nor in N. We show that $[\![M]\!]_\rho = [\![N]\!]_\rho$ for every LCL-model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$. Let d be an element from the domain of the model $\mathcal{M}_\rho$. Since $[\![Mx]\!]_\rho = [\![Nx]\!]_\rho$ holds for any model $\mathcal{M}_\rho = \langle \mathcal{M}, \rho \rangle$ by the induction hypothesis, it also holds for the model $\langle \mathcal{M}, \rho(x := d) \rangle$. The variable x does not appear in the term M, thus by Lemma 4.18 we have $[\![M]\!]_\rho = [\![M]\!]_{\rho(x:=d)}$ and the same holds for the term N.

We derive

$$\begin{aligned}
[\![M]\!]_\rho \cdot d &= [\![M]\!]_{\rho(x:=d)} \cdot d \\
&= [\![M]\!]_{\rho(x:=d)} \cdot [\![x]\!]_{\rho(x:=d)} \\
&= [\![Mx]\!]_{\rho(x:=d)} \\
&= [\![Nx]\!]_{\rho(x:=d)} \\
&= [\![N]\!]_{\rho(x:=d)} \cdot [\![x]\!]_{\rho(x:=d)} \\
&= [\![N]\!]_{\rho(x:=d)} \cdot d \\
&= [\![N]\!]_\rho \cdot d.
\end{aligned}$$

We have proved that $[\![M]\!]_\rho \cdot d = [\![N]\!]_\rho \cdot d$ holds for every element d from the domain. We conclude that $[\![M]\!]_\rho = [\![N]\!]_\rho$ by the extensionality of the operation $\cdot$.

This concludes the proof. □

Next, we prove the completeness of the equational theory $\mathcal{EQ}^\eta$, which is the converse of Theorem 4.22.

Let us recall that $[M]$ denotes the equivalence class of a term M with respect to the equivalence relation generated by the equational theory $\mathcal{EQ}^\eta$, $[M] = \{N \mid M = N \text{ is provable in } \mathcal{EQ}^\eta\}$.

Theorem 4.23 (Completeness of $\mathcal{EQ}^\eta$)**.** *If $[\![M]\!]_\rho = [\![N]\!]_\rho$ in every LCL-model, then $M = N$ is provable in $\mathcal{EQ}^\eta$.*

Proof. We define the LCL-model $\mathcal{M}_{\rho^\star} = \langle \mathcal{M}_0, \rho^\star \rangle$ as follows. The applicative structure $\mathcal{M}_0$ is a tuple $\langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i} \rangle$ where:

- $D = \{[M] \mid M \in CL\}$,
- $\{A^\sigma\}_\sigma = \{A^\sigma\}_{\sigma \in \mathsf{Types}_\to}$ is a family of sets indexed by σ such that

$$A^\sigma = \{[M] \mid \ M \in CL \text{ and } \ \vdash M : \sigma\}$$

- $[M] \cdot [N] = [MN]$,
- $\boldsymbol{s} = [\mathsf{S}]$,
- $\boldsymbol{k} = [\mathsf{K}]$,
- $\boldsymbol{i} = [\mathsf{I}]$.

The environment $\rho^\star$ is defined by

$$\rho^\star(x) = [x].$$

Notice that a set A^σ is defined as the set of all equivalence classes of typable terms. The judgment $\vdash M : \sigma$ indicates that we can derive $\vdash M : \sigma$ by the axioms and rules given in Figure 4.3, that is $M : \sigma$ is a theorem in LCL.

We have to prove that $\mathcal{M}_{\rho^\star}$ is an LCL-model. Fist, we prove that the tuple $\mathcal{M}_0 = \langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}\rangle$ is an applicative structure, i.e. that it satisfies the conditions of Definition 4.14. The domain D is a nonempty set $\{[M] \mid M \in CL\}$. For every type σ, we have

$$A^\sigma = \{[M] \mid \ M \in CL \text{ and } \ \vdash M : \sigma\} \subseteq \{[M] \mid M \in CL\} = D.$$

The operation $\cdot$ given by $[M] \cdot [N] = [MN]$ is a binary operation on the domain D. Its extensionality follows from the extensionality of the equational theory $\mathcal{EQ}^\eta$. Assume that $[M]$ and $[N]$ are the elements of the domain D such that $[M] \cdot [L] = [N] \cdot [L]$, for every $[L] \in D$. Let x be a variable that does not appear in M or N. If we take L to be x, then $[M] \cdot [x] = [N] \cdot [x]$, i.e. $[Mx] = [Nx]$. So, we have that $Mx = Nx$ is provable in $\mathcal{EQ}^\eta$ and by rule (ext) we conclude that $M = N$ is provable in $\mathcal{EQ}^\eta$, i.e. $[M] = [N]$. Thus, $\cdot$ is extensional.

Further, assume that $[M] \in A^{\sigma\to\tau}$ and $[N] \in A^\sigma$ for some $\sigma, \tau \in \mathsf{Types}_\to$. Then $\vdash M : \sigma \to \tau$ and $\vdash N : \sigma$ directly follow from the definition of sets $A^{\sigma\to\tau}$ and A^σ, respectively. As an instance of axiom scheme (4) we have $(M : \sigma \to \tau) \Rightarrow ((N : \sigma) \Rightarrow (MN : \tau))$. By the rule Modus Ponens we derive $\vdash MN : \tau$. Thus, we conclude that $[MN] \in A^\tau$ and that the codomain of the restriction of function $\cdot$ to the set $A^{\sigma\to\tau} \times A^\sigma$ is A^τ.

We prove that elements $\boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}$ belong to the respective subsets of the domain and satisfy equations given in Definition 4.14. Since $\vdash \mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$ is an axiom, for every $\sigma, \rho, \tau \in \mathsf{Types}_\to$, we have $[\mathsf{S}] \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}$ by the definition of the structure $\mathcal{M}_0$. So, the element $[\mathsf{S}]$ satisfies the equation (4.4). For any $[M], [N], [L] \in D$, we have

$$\begin{aligned}((([\mathsf{S}] \cdot [M]) \cdot [N]) \cdot [L] &= [((\mathsf{S}M)N)L] = [(ML)(NL)] \\ &= [ML] \cdot [NL] \\ &= ([M] \cdot [L]) \cdot ([N] \cdot [L]).\end{aligned}$$

Hence, the element $[\mathsf{S}]$ also satisfies the equation (4.5).

By similar reasoning, we prove that the element $[\mathsf{K}] \in D$ satisfies equations (4.6) and (4.7). We have that $\mathsf{K} : \sigma \to (\tau \to \sigma)$ is an axiom for every

$\sigma, \tau \in \mathsf{Types}_{\to}$, hence $[\mathsf{K}] \in A^{\sigma\to(\tau\to\sigma)}$. Next, for every $[M], [N] \in D$ we derive

$$([\mathsf{K}] \cdot [M]) \cdot [N] = [(\mathsf{K}M)N] = [M].$$

Similarly, the element $[\mathsf{I}]$ satisfies equations (4.8) and (4.9). For every $\sigma \in \mathsf{Types}_{\to}$, $\mathsf{I} : \sigma \to \sigma$ is an axiom and as a consequence we have $[\mathsf{I}] \in A^{\sigma\to\sigma}$ for every $\sigma \in \mathsf{Types}_{\to}$. Further, for every $[M] \in D$, we have

$$[\mathsf{I}] \cdot [M] = [\mathsf{I}M] = [M].$$

We conclude that the tuple $\mathcal{M}_0$ is an applicative structure for LCL. Since $\rho^\star$ is a total function from the set of term variables to the domain D, we have that $\mathcal{M}_{\rho^\star}$ is an LCL-model.

By similar reasoning as in the proof of Lemma 3.24 (Chapter 3), we can prove that $[\![M]\!]_{\rho^\star} = [M]$, for every term M.

If we assume that $[\![M]\!]_\rho = [\![N]\!]_\rho$ holds in every LCL-model, then we have that it also holds in $\mathcal{M}_{\rho^\star}$, i.e. $[\![M]\!]_{\rho^\star} = [\![N]\!]_{\rho^\star}$. Since the interpretation of a term in $\mathcal{M}_{\rho^\star}$ is its equivalence class with respect to the equivalence relation generated by the equational theory $\mathcal{EQ}^\eta$, we obtain that $[M] = [\![M]\!]_{\rho^\star} = [\![N]\!]_{\rho^\star} = [N]$, i.e. that $M = N$ is provable in $\mathcal{EQ}^\eta$. □

We may notice that we deal only with the interpretation of a term and we do not consider its type here, so the sets A^σ were not of utmost importance in the proof of completeness of the equational theory $\mathcal{EQ}^\eta$. Any family $\{A^\sigma\}_\sigma$ of subsets of the domain D that satisfies the conditions of Definition 4.14 could be used in the proof.

The soundness of the equational theory is crucial for the proof of the soundness and strong completeness of the axiomatization LCL, which is proved in the following section. In particular, this result was necessary in order to prove that every instance of axiom scheme (5) is satisfied in every LCL-model. An instance of (5) is of the form $M : \sigma \Rightarrow N : \sigma$, for some terms M, N such that $M = N$ is provable in the equational theory $\mathcal{EQ}^\eta$. Then $[\![M]\!]_\rho = [\![N]\!]_\rho$ in every LCL-model and as a consequence we have that the formula $M : \sigma \Rightarrow N : \sigma$ is satisfied in every LCL-model.

We have proved that $[\![M]\!]_\rho = [\![N]\!]_\rho$ holds in every LCL-model if and only if $M = N$ is provable in $\mathcal{EQ}^\eta$. Additionally, if we want to prove $M = N$ in $\mathcal{EQ}^\eta$ it is enough to prove $[\![M]\!]_{\rho^\star} = [\![N]\!]_{\rho^\star}$ in the model $\mathcal{M}_{\rho^\star}$, given in the proof of Theorem 4.23. Nevertheless, this does not make it easier to determine if two terms are equal, since both problems are undecidable. Proving $M = N$ in $\mathcal{EQ}^\eta$ is undecidable and as a consequence proving $N \in [M]$ is undecidable. Thus, the problem of determining if $[\![M]\!]_{\rho^\star} = [\![N]\!]_{\rho^\star}$ is undecidable.

4.4 Soundness and strong completeness of the axiomatization of LCL

In this section, we give the proof of the soundness and strong completeness of the logic of combinatory logic, i.e. of its axiomatization given in Section 4.2.2, with respect to the semantics proposed in Section 4.2.3. In addition, we also prove that the simply typed combinatory logic with equality is sound and complete with respect to the proposed semantics.

4.4.1 Soundness of LCL

This subsection presents the first result, soundness of the axiomatization of LCL. We prove that if a formula is a deductive consequence of a set of formulas, then it is also a semantical consequence of that set.

Theorem 4.24 (Soundness of LCL)**.** *If $T \vdash \alpha$, then $T \models \alpha$.*

Proof. We show that each instance of an axiom scheme is satisfied by any model and that the inference rule Modus Ponens preserves satisfiability.

By Definition 4.14 we have that $\boldsymbol{s} \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}$ for every $\sigma,\tau,\rho \in \mathsf{Types}_{\to}$ and every model $\mathcal{M}_\rho$, and by Definition 4.17 we have that $[\![\mathsf{S}]\!]_\rho = \boldsymbol{s}$. Hence, it follows that $[\![\mathsf{S}]\!]_\rho \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}$ and that every instance of axiom scheme (1) is satisfied in every LCL-model $\mathcal{M}_\rho$.

Similarly, since $\boldsymbol{k} \in A^{\sigma\to(\tau\to\sigma)}$ for every $\sigma,\tau \in \mathsf{Types}_{\to}$ by Definition 4.14 and $[\![\mathsf{K}]\!]_\rho = \boldsymbol{k}$ by Definition 4.17, we conclude that $[\![\mathsf{K}]\!]_\rho \in A^{\sigma\to(\tau\to\sigma)}$ for every $\sigma,\tau \in \mathsf{Types}_{\to}$. This implies that every instance of axiom scheme (2) is satisfied in every model.

By similar reasoning, we obtain that every instance of axiom scheme (3) is satisfied in every model. From $\boldsymbol{i} \in A^{\sigma\to\sigma}$ (Definition 4.14) and $[\![\mathsf{I}]\!]_\rho = \boldsymbol{i}$ (Definition 4.17), it follows that $[\![\mathsf{I}]\!]_\rho \in A^{\sigma\to\sigma}$.

The satisfiability of propositional connectives is defined as usual, that is an implication $\alpha \Rightarrow \beta$ is not satisfied by a model if and only if α is satisfied and β is not satisfied by the model. If we suppose that there exists a model $\mathcal{M}_\rho$ such that an instance of (4), $(M : \sigma \to \tau) \Rightarrow ((N : \sigma) \Rightarrow (MN : \tau))$, is not satisfied by a model $\mathcal{M}_\rho$. Then $\mathcal{M}_\rho \models M : \sigma \to \tau$, $\mathcal{M}_\rho \models N : \sigma$ and $\mathcal{M}_\rho \not\models MN : \tau$. We have that $\mathcal{M}_\rho \models M : \sigma \to \tau$ implies $[\![M]\!]_\rho \in A^{\sigma\to\tau}$, and $\mathcal{M}_\rho \models N : \sigma$ implies $[\![N]\!]_\rho \in A^{\sigma}$. Further, we obtain $[\![MN]\!]_\rho = [\![M]\!]_\rho \cdot [\![N]\!]_\rho \in A^{\tau}$ by Definition 4.14 and Definition 4.17. This contradicts the assumption $\mathcal{M}_\rho \not\models MN : \tau$, so we conclude that every instance of (4) is satisfied by any LCL-model.

Next, we consider an instance of axiom scheme (5), $M : \sigma \Rightarrow N : \sigma$, where $M = N$ is provable in $\mathcal{EQ}^\eta$. If a model $\mathcal{M}_\rho$ satisfies $M : \sigma$, i.e. $\mathcal{M}_\rho \models M : \sigma$,

then we have $[\![M]\!]_\rho \in A^\sigma$ by Definition 4.20. Since $M = N$ is provable in $\mathcal{EQ}^\eta$, we know that $[\![M]\!]_\rho = [\![N]\!]_\rho$ by Theorem 4.22. It directly follows that $[\![N]\!]_\rho \in A^\sigma$. Hence, an instance of axiom scheme (5) is satisfied by any LCL-model.

Every instance of axiom scheme (6), $\alpha \Rightarrow (\beta \Rightarrow \alpha)$, is satisfied by any LCL-model, otherwise we would have that α and β are satisfied by the model and α is not satisfied, which leads to a contradiction.

The satisfiability of an instance of axiom scheme (7) follows by the propositional reasoning. Let $\mathcal{M}_\rho$ be a model such that

(1) $\mathcal{M}_\rho \models \alpha \Rightarrow (\beta \Rightarrow \gamma)$,

(2) $\mathcal{M}_\rho \models \alpha \Rightarrow \beta$,

(3) $\mathcal{M}_\rho \models \alpha$.

Then (2) and (3) imply $\mathcal{M}_\rho \models \beta$. Further, (1), (3) and $\mathcal{M}_\rho \models \beta$ imply $\mathcal{M}_\rho \models \gamma$. Hence, every instance of axiom scheme (7) is satisfied by any LCL-model.

Now, we prove that every instance of axiom scheme (8) is satisfied by any model. If we assume that there exists a model $\mathcal{M}_\rho$ such that an instance of (8), $(\neg\alpha \Rightarrow \neg\beta) \Rightarrow ((\neg\alpha \Rightarrow \beta) \Rightarrow \alpha)$, is not satisfied by $\mathcal{M}_\rho$, then we have

(1) $\mathcal{M}_\rho \models \neg\alpha \Rightarrow \neg\beta$,

(2) $\mathcal{M}_\rho \models \neg\alpha \Rightarrow \beta$,

(3) $\mathcal{M}_\rho \not\models \alpha$.

From (3), it follows that $\mathcal{M}_\rho \models \neg\alpha$. Further, (1) and $\mathcal{M}_\rho \models \neg\alpha$ imply $\mathcal{M}_\rho \models \neg\beta$. Similarly, (2) and $\mathcal{M}_\rho \models \neg\alpha$ imply $\mathcal{M}_\rho \models \beta$. However, the latter contradicts $\mathcal{M}_\rho \models \neg\beta$. We conclude that the assumption is not true, so every instance of axiom scheme (8) is satisfied by any LCL-model.

Finally, we prove that the inference rule Modus Ponens preserves satisfiability. By Definition 4.20 we have that $\mathcal{M}_\rho \models \beta$, whenever $\mathcal{M}_\rho \models \alpha \Rightarrow \beta$ and $\mathcal{M}_\rho \models \alpha$. This concludes the proof. □

4.4.2 Strong completeness of LCL

In this subsection, we prove the strong completeness of the logic of combinatory logic, which is the converse of Theorem 4.36.

When logical system is given with two different presentations, e.g. semantics (model theory) and axiomatization (proof theory), we can study two

kinds of completeness results: *(weak) completeness* and *strong completeness*. The logical system is complete, i.e. weakly complete, if the following holds:

$$\text{If } \models \varphi \text{ then } \vdash \varphi. \tag{4.10}$$

In other words, completeness means that if a formula is true in every model of the logic, then it is provable in the logic. On the other hand, the logic is strongly complete if the following holds:

$$\text{If } T \models \varphi \text{ then } T \vdash \varphi. \tag{4.11}$$

where T is a set of formulas. Therefore, the strong completeness means that if a formula is a semantical consequence of a set of formulas, then it is also a deductive consequence of that set. The completeness only addresses the special case of strong completeness when the set of premises is empty. Thus, strong completeness implies completeness.

We adapt the Henkin-style completeness method, developed for the proof of completeness of modal logic. The proof of strong completeness comprises the following steps:

- the proof of Deduction theorem given in Section 4.2.2,
- the proof that every consistent set of LCL-formulas can be extended to a maximal consistent set,
- the construction of a canonical model using the maximal consistent set,
- the proof that the canonical model is LCL-model,
- the proof that every consistent set is satisfiable,
- the proof of strong completeness of LCL.

First, an auxiliary result, which describes the property of a consistent set, is given and a notion of a maximal consistent set is introduced.

Lemma 4.25. *Let T be a consistent set of formulas. For any formula α, either $T \cup \{\alpha\}$ is consistent, or $T \cup \{\neg\alpha\}$ is consistent.*

Proof. Assume the opposite, i.e. that there exists a consistent set T and a formula α such that both $T \cup \{\alpha\}$ and $T \cup \{\neg\alpha\}$ are inconsistent. It follows that $T, \alpha \vdash \bot$ and $T, \neg\alpha \vdash \bot$. By Theorem 4.13 we get $T \vdash \alpha \Rightarrow \bot$ and $T \vdash \neg\alpha \Rightarrow \bot$. Now, $T \vdash \alpha \Rightarrow \bot$ is equivalent to $T \vdash \neg\alpha$, and $T \vdash \neg\alpha \Rightarrow \bot$ is equivalent to $T \vdash \alpha$. However, $T \vdash \neg\alpha$ and $T \vdash \alpha$ contradict the assumption that T is a consistent set, so $T \cup \{\alpha\}$ and $T \cup \{\neg\alpha\}$ cannot be both inconsistent. □

Definition 4.26. *Let T be a set of formulas. T is a maximal consistent set if it is consistent and for any formula α, either $\alpha \in T$ or $\neg\alpha \in T$.*

A key point in proving the strong completeness of the logic of combinatory logic is the proof that every consistent set can be extended to a maximal consistent set.

Theorem 4.27 (Lindenbaum's theorem)**.** *Every consistent set of LCL-formulas can be extended to a maximal consistent set.*

Proof. Let T be a consistent set. We construct a set $T^\star$, which is a maximal consistent set and includes the set T, i.e. $T \subseteq T^\star$. We let $\alpha_0, \alpha_1, \ldots$ be an enumeration of all LCL-formulas. The sequence of sets T_i, $i = 0, 1, \ldots$ and the set $T^\star$ are defined as follows:

(1) $T_0 = T$,

(2) for every $i \geq 0$

  (a) if $T_i \cup \{\alpha_i\}$ is consistent, then $T_{i+1} = T_i \cup \{\alpha_i\}$, otherwise

  (b) $T_{i+1} = T_i \cup \{\neg\alpha_i\}$,

(3) $T^\star = \bigcup_{i=0}^{\infty} T_i$.

We may notice from the definition of set $T^\star$ that $T \subseteq T^\star$. We show that $T^\star$ is a maximal consistent set. From the definition of the set $T^\star$, it follows that it is maximal, so it remains to prove it is consistent. In order to do so, we prove that it is deductively closed set, which does not contain all formulas. From the definition of the set $T^\star$ it follows that it does not contains all formulas. Further, every set T_i, $i \in \{0, 1, 2, \ldots\}$, is consistent by its definition.

We prove that the set $T^\star$ is deductively closed, i.e. that for every formula α such that $T^\star \vdash \alpha$, it holds that $\alpha \in T^\star$. For $\alpha = \alpha_j$ and the set T_i such that $T_i \vdash \alpha$ we have $\alpha \in T^\star$, due to the consistency of the set $T_{\max\{i,j\}+1}$. If $T^\star \vdash \alpha$, then there is a sequence of formulas $\alpha_0, \alpha_1, \ldots, \alpha_n$ that is a proof of α from $T^\star$. The sequence $\alpha_0, \alpha_1, \ldots, \alpha_n$ is finite, so there is a set T_i such that $T_i \vdash \alpha$. It follows that $\alpha \in T^\star$, thus $T^\star$ is deductively closed.

□

Remark 4.28. *We may notice that if $T^\star$ is a maximal consistent set which contains the set T, defined as in the proof of Theorem 4.27, and $T \vdash \alpha$, then $\alpha \in T^\star$. This is the consequence of the fact that $T^\star$ is deductively closed.*

Using the maximal consistent set $T^\star$, we construct a canonical applicative structure. Providing a canonical applicative structure with an environment results in a canonical model. The canonical model is defined in the way that it satisfies exactly the formulas from the maximal consistent set.

Definition 4.29. *Let $T^\star$ be a maximal consistent set. A canonical applicative structure is a tuple $\mathcal{M}_{T^\star} = \langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}\rangle$ where*

- $D = \{[M] \mid M \in CL\}$,
- $\{A^\sigma\}_\sigma = \{A^\sigma\}_{\sigma \in \mathsf{Types}_\to}$ *is a family of sets indexed by σ such that*

$$A^\sigma = \{[M] \mid M \in CL \text{ and } M : \sigma \in T^\star\},$$

- $[M] \cdot [N] = [MN]$,
- $\boldsymbol{s} = [\mathsf{S}]$,
- $\boldsymbol{k} = [\mathsf{K}]$,
- $\boldsymbol{i} = [\mathsf{I}]$.

Lemma 4.30. *A canonical applicative structure $\mathcal{M}_{T^\star}$ is an applicative structure for LCL.*

Proof. We show that the tuple $\mathcal{M}_{T^\star}$ given in Definition 4.29 satisfies the conditions of Definition 4.14.

The set $D = \{[M] \mid M \in CL\}$ is a nonempty set. For every type σ, we have

$$A^\sigma = \{[M] \mid M \in CL \text{ and } M : \sigma \in T^\star\} \subseteq \{[M] \mid M \in CL\} = D.$$

We may notice that the set A^σ is well-defined. If $M = N$ is provable in $\mathcal{EQ}^\eta$, then $M : \sigma \Rightarrow N : \sigma$ is an instance of axiom scheme and as a consequence $M : \sigma \Rightarrow N : \sigma \in T^\star$. Since $T^\star$ is deductively closed, we have that for terms M, N such that $M = N$ is provable in $\mathcal{EQ}^\eta$, $M : \sigma \in T^\star$ if and only if $N : \sigma \in T^\star$. This implies that if $[M] \in A^\sigma$ and $N \in [M]$, then $[N] \in A^\sigma$, that is A^σ is well-defined.

The operation $\cdot$ given by $[M] \cdot [N] = [MN]$ is a binary operation on D. Its extensionality follows from the extensionality of the equational theory $\mathcal{EQ}^\eta$ by the same reasoning as in the proof of Theorem 4.23, where the domain and the operation are defined the same as here.

Let $[M] \in A^{\sigma\to\tau}$ and $[N] \in A^\sigma$ for some $\sigma, \tau \in \mathsf{Types}_\to$. By Definition 4.29 we have that $[M] \in A^{\sigma\to\tau}$ is equivalent to $M : \sigma \to \tau \in T^\star$ and $[N] \in A^\sigma$

is equivalent to $N : \sigma \in T^\star$. The set $T^\star$ is deductively closed, so we get $MN : \tau \in T^\star$, which implies $[MN] \in A^\tau$. We have shown that the codomain of the restriction of function $\cdot$ to the set $A^{\sigma\to\tau} \times A^\sigma$ is A^τ.

Next, we prove that elements $\boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}$ belong to the respective subsets of the domain and satisfy equations given in Definition 4.14. We have already proved that the elements $\boldsymbol{s} = [\mathsf{S}]$, $\boldsymbol{k} = [\mathsf{K}]$ and $\boldsymbol{i} = [\mathsf{I}]$ satisfy equations 4.5, 4.7 and 4.9, respectively, in the proof of Theorem 4.23. Since $\mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$ is an instance of axiom scheme (1), we have $\mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho)) \in T^\star$. Thus, $\boldsymbol{s} = [\mathsf{S}] \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}$.

Similarly, we have that $\mathsf{K} : \sigma \to (\tau \to \sigma)$ is an instance of axiom scheme (2) for every $\sigma, \tau \in \mathsf{Types}_\to$, hence $\mathsf{K} : \sigma \to (\tau \to \sigma) \in T^\star$. We conclude that $\boldsymbol{k} = [\mathsf{K}] \in A^{\sigma\to(\tau\to\sigma)}$, for every $\sigma, \tau \in \mathsf{Types}_\to$.

For every $\sigma \in \mathsf{Types}_\to$, $\mathsf{I} : \sigma \to \sigma$ is an instance of axiom scheme (3), so it belongs to the set $T^\star$ and we have that $\boldsymbol{i} = [\mathsf{I}] \in A^{\sigma\to\sigma}$ for every $\sigma \in \mathsf{Types}_\to$,

This concludes the proof that $\mathcal{M}_{T^\star}$ is an applicative structure for LCL.

□

Definition 4.31. *A canonical model is a tuple $\mathcal{M}_{T^\star,\rho^\star} = \langle \mathcal{M}_{T^\star}, \rho^\star \rangle$, where $\mathcal{M}_{T^\star}$ is a canonical applicative structure and $\rho^\star$ is the environment defined by $\rho^\star(x) = [x]$.*

As we have already discussed in the proof of Theorem 4.23, if an environment $\rho^\star$ is defined by $\rho^\star(x) = [x]$ on the domain $D = \{[M] \mid M \in CL\}$, then the interpretation of every term M is given by $[\![M]\!]_{\rho^\star} = [M]$.

The main property of a canonical model $\mathcal{M}_{T^\star,\rho^\star}$ is that it satisfies exactly the formulas from $T^\star$.

Lemma 4.32. *Let $\mathcal{M}_{T^\star,\rho^\star}$ be a canonical model and α a formula. Then*

$$\mathcal{M}_{T^\star,\rho^\star} \models \alpha \text{ if and only if } \alpha \in T^\star.$$

Proof. The proof is by induction on the structure of the formula α.

In the case when α is a statement $M : \sigma$, we have

$$\begin{aligned} \mathcal{M}_{T^\star,\rho^\star} \models M : \sigma \text{ if and only if } & [\![M]\!]_{\rho^\star} \in A^\sigma \\ \text{if and only if } & [M] \in A^\sigma \\ \text{if and only if } & M : \sigma \in T^\star. \end{aligned}$$

If formula α is a conjunction $\beta \wedge \gamma$, then

$$\begin{aligned} \mathcal{M}_{T^\star,\rho^\star} \models \beta \wedge \gamma \text{ if and only if } & \mathcal{M}_{T^\star,\rho^\star} \models \beta \text{ and } \mathcal{M}_{T^\star,\rho^\star} \models \gamma \\ \text{if and only if } & \beta \in T^\star \text{ and } \gamma \in T^\star \\ \text{if and only if } & \beta \wedge \gamma \in T^\star. \end{aligned}$$

Finally, assume α is a negation $\neg\beta$. Then

$$\begin{aligned}\mathcal{M}_{T^\star,\rho^\star} \models \neg\beta \text{ if and only if } & \mathcal{M}_{T^\star,\rho^\star} \not\models \beta \\ \text{if and only if } & \beta \notin T^\star \\ \text{if and only if } & \neg\beta \in T^\star.\end{aligned}$$

□

Now, we are ready to prove that every consistent set is satisfiable.

Theorem 4.33. *Every consistent set is satisfiable.*

Proof. Assume that T is a consistent set. The set T can be extended to a maximal consistent set $T^\star$ by Theorem 4.27. We consider the canonical model $\mathcal{M}_{T^\star,\rho^\star}$. By Lemma 4.32 we have that $\mathcal{M}_{T^\star,\rho^\star}$ satisfies all formulas from set $T^\star$. Consequently, $\mathcal{M}_{T^\star,\rho^\star} \models T$, since $T \subseteq T^\star$. Thus, $\mathcal{M}_{T^\star,\rho^\star}$ is a model of the set T. □

Theorem 4.34 (Strong completeness of LCL)**.** *If $T \models \alpha$, then $T \vdash \alpha$.*

Proof. Assume that $T \models \alpha$. Then the set $T \cup \{\neg\alpha\}$ is not satisfiable and as a consequence it is not consistent, by Theorem 4.33. The latter implies $T \cup \{\neg\alpha\} \vdash \bot$, and by Theorem 4.13 and the propositional reasoning it follows that $T \vdash \alpha$. □

Corollary 4.35. *$T \vdash \alpha$ if and only if $T \models \alpha$.*

4.5 Soundness and completeness of the simply typed combinatory logic

As a consequence of the soundness and strong completeness of the logic of combinatory logic, we have that the simply typed combinatory logic is sound and complete with respect to the proposed semantics. The soundness of the simply typed combinatory logic, presented in Section 4.1, is a direct consequence of Proposition 4.11 and Theorem 4.24.

Theorem 4.36 (Soundness of $CL_\to$)**.** *If $\Gamma \vdash_{CL} M : \sigma$, then $\Gamma \models M : \sigma$.*

Proof. Let $\Gamma \vdash_{\mathrm{CL}} M : \sigma$. By Proposition 4.11, there exists a proof for $M : \sigma$ from Γ in LCL, i.e. $\Gamma \vdash M : \sigma$. Then we obtain $\Gamma \models M : \sigma$ by Theorem 4.24. □

If we consider the type assignment system given in Figure 4.2, then the converse of Theorem 4.36 does not hold. As a counterexample, we take statements $x : \sigma$ and $\mathsf{K}xy : \sigma$. In the equational theory $\mathcal{EQ}^{\eta}$ we can prove the equation $\mathsf{K}xy = x$. Thus, by Theorem 4.22 we have that the terms $\mathsf{K}xy$ and x have equal interpretations in every LCL-model, i.e. $[\![\mathsf{K}xy]\!]_{\rho} = [\![x]\!]_{\rho}$. In other words, whenever a model satisfies $x : \sigma$, then it also satisfies $\mathsf{K}xy : \sigma$, i.e. $x : \sigma \models \mathsf{K}xy : \sigma$. If the simple type assignment system, given in Figure 4.2, was complete, then we would conclude $x : \sigma \vdash_{\mathrm{CL}} \mathsf{K}xy : \sigma$. However, $x : \sigma \vdash_{\mathrm{CL}} \mathsf{K}xy : \sigma$ can not be derived in $CL_{\to}$.

In order to obtain a type assignment system that is sound and complete, we have to extend the system given in Figure 4.2 by the following rule

$$\frac{\Gamma \vdash_{\mathrm{CL}^{=}} M : \sigma \qquad M = N \ \text{ is provable in } \ \mathcal{EQ}^{\eta}}{\Gamma \vdash_{\mathrm{CL}^{=}} N : \sigma} \ (\mathrm{eq})$$

The rule (eq) guarantees that equal terms inhabit the same types. The type assignment system obtained by adding the rule (eq) to the type assignment system in Figure 4.2 is denoted by $CL^{=}_{\to}$. The derivability in $CL^{=}_{\to}$ is denoted by $\vdash_{\mathrm{CL}^{=}}$. The typing rule (eq) corresponds to the axiom scheme (5) in the axiomatization of LCL. We may notice that Proposition 4.11 holds also for the system $CL^{=}_{\to}$, that is if $\Gamma \vdash_{\mathrm{CL}^{=}} M : \sigma$, then $\Gamma \vdash M : \sigma$ (in LCL). The type assignment system $CL^{=}_{\to}$ is sound and complete with respect to the semantics proposed in Section 4.2.3.

Theorem 4.37 (Soundness of $CL^{=}_{\to}$)**.** *If* $\Gamma \vdash_{CL^{=}} M : \sigma$*, then* $\Gamma \models M : \sigma$*.*

Proof. The proof is a direct consequence of Theorem 4.24 and Proposition 4.11 and proceeds by similar reasoning as the proof of Theorem 4.36. □

The converse of Theorem 4.37 also holds. We follow the approach used in Chapter 3 and for a basis Γ we define a model $\mathcal{M}^{\Gamma}$ such that $\mathcal{M}^{\Gamma} \models M : \sigma$ if and only if $\Gamma \vdash_{\mathrm{CL}^{=}} M : \sigma$.

Definition 4.38. *Let* Γ *be a basis. We define a tuple* $\mathcal{M}^{\Gamma}$ *as follows.*

$$\mathcal{M}^{\Gamma} = \langle D, \{A^{\sigma}\}_{\sigma}, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i} \rangle$$

where

- $D = \{[M] \mid M \in CL\}$,
- $\{A^{\sigma}\}_{\sigma} = \{A^{\sigma}\}_{\sigma \in \mathsf{Types}_{\to}}$ *is a family of sets indexed by* σ *such that*

$$A^{\sigma} = \{[M] \mid M \in CL \text{ and } \Gamma \vdash_{CL^{=}} M : \sigma\},$$

- $[M] \cdot [N] = [MN]$,
- $\boldsymbol{s} = [\mathsf{S}]$,
- $\boldsymbol{k} = [\mathsf{K}]$,
- $\boldsymbol{i} = [\mathsf{I}]$.

Lemma 4.39. *Let Γ be a basis. The tuple $\mathcal{M}^\Gamma$ introduced in Definition 4.38 is an applicative structure for LCL.*

Proof. Similarly as in the proof of Theorem 4.23 we prove that the tuple $\mathcal{M}^\Gamma = \langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}\rangle$ satisfies the conditions of Definition 4.14. By Definition 4.38, we have that the set D is nonempty and for every $\sigma \in \mathsf{Types}_\to$,

$$A^\sigma = \{[M] \mid M \in CL \text{ and } \Gamma \vdash_{\mathrm{CL}^=} M : \sigma\} \subseteq \{[M] \mid M \in CL\} = D.$$

The operation $\cdot$ is a binary, extensional operation on the set D. The extensionality of $\cdot$ follows from the extensionality of the equational theory $\mathcal{EQ}^\eta$ and it has been shown in the proof of Theorem 4.23. By the similar reasoning as in the proof of Theorem 4.23, we obtain that the codomain of the restriction of function $\cdot$ to the set $A^{\sigma\to\tau} \times A^\sigma$ is A^τ. Let $[M] \in A^{\sigma\to\tau}$ and $[N] \in A^\sigma$. It follows that $\Gamma \vdash_{\mathrm{CL}^=} M : \sigma \to \tau$ and $\Gamma \vdash_{\mathrm{CL}^=} N : \sigma$ by Definition 4.38. So, we derive $\Gamma \vdash_{\mathrm{CL}^=} MN : \tau$ by the rule ($\to$ elim). The latter implies $[MN] \in A^\tau$.

It remains to show that elements $\boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}$ belong to the respective subsets of the set D, i.e. that they satisfy the conditions 4.4, 4.6 and 4.8, and the equations 4.5, 4.7 and 4.9, respectively. By the rules in Figure 4.2 we have $\Gamma \vdash_{\mathrm{CL}^=} \mathsf{S} : (\sigma \to (\rho \to \tau)) \to ((\sigma \to \rho) \to (\sigma \to \tau))$, for all $\sigma, \rho, \tau \in \mathsf{Types}_\to$. So, we can conclude $\boldsymbol{s} = [\mathsf{S}] \in A^{(\sigma\to(\rho\to\tau))\to((\sigma\to\rho)\to(\sigma\to\tau))}$ for every $\sigma, \rho, \tau \in \mathsf{Types}_\to$, i.e. element $\boldsymbol{s}$ satisfies the condition 4.4. The proof that elements $\boldsymbol{k}$ and $\boldsymbol{i}$ satisfy conditions 4.6 and 4.8, respectively, follows by similar reasoning. □

We use the applicative structure $\mathcal{M}^\Gamma$ to define a model $\mathcal{M}^\Gamma_{\rho^\star}$ that satisfies only typing statements derivable from Γ in the system $CL^=_\to$.

Lemma 4.40. *Let $\mathcal{M}^\Gamma = \langle D, \{A^\sigma\}_\sigma, \cdot, \boldsymbol{s}, \boldsymbol{k}, \boldsymbol{i}\rangle$ be an applicative structure given in Definition 4.38 and $\rho^\star$ an environment defined by $\rho^\star(x) = [x]$. Then $\mathcal{M}^\Gamma_{\rho^\star} \models M : \sigma$ if and only if $\Gamma \vdash_{CL^=} M : \sigma$.*

Proof. As it was discussed in the proof of Theorem 4.23, if the domain is $D = \{[M] \mid M \in CL\}$ and an environment is defined by $\rho^\star(x) = [x]$, then $[\![M]\!]_{\rho^\star} = [M]$ for every term M. It follows that

$$\begin{aligned}\mathcal{M}^{\Gamma}_{\rho^\star} \models M : \sigma \text{ if and only if } & [\![M]\!]_{\rho^\star} \in A^\sigma \\ \text{if and only if } & [M] \in \{[N] \mid N \in CL \text{ and } \Gamma \vdash_{\mathrm{CL}^=} N : \sigma\} \\ \text{if and only if } & \Gamma \vdash_{\mathrm{CL}^=} M : \sigma.\end{aligned}$$

□

Finally, we are ready to prove that the type assignment system $CL^{=}_{\rightarrow}$ is complete with respect to the semantics introduced in Section 4.2.3.

Theorem 4.41 (Completeness of $CL^{=}_{\rightarrow}$)**.** *Let* Γ *be a basis. If* $\Gamma \models M : \sigma$*, then* $\Gamma \vdash_{CL^=} M : \sigma$*.*

Proof. Let Γ be a basis such that $\Gamma \models M : \sigma$. We consider the model $\mathcal{M}^{\Gamma}_{\rho^\star}$ given in Definition 4.38, which is a model of Γ by Lemma 4.40, i.e. $\mathcal{M}^{\Gamma}_{\rho^\star} \models \Gamma$. From $\mathcal{M}^{\Gamma}_{\rho^\star} \models \Gamma$ and $\Gamma \models M : \sigma$, we obtain $\mathcal{M}^{\Gamma}_{\rho^\star} \models M : \sigma$. Then we get $\Gamma \vdash_{\mathrm{CL}^=} M : \sigma$ by Lemma 4.40 . □

We end this section with the proof that the logic of combinatory logic is a conservative extension of the simply typed combinatory logic. Proposition 4.11 has shown that the logic of combinatory logic is an extension of the simply typed combinatory logic, and we prove that this extension is conservative if we consider the simply typed combinatory logic with equality .

Theorem 4.42. *Let* Γ *be a basis. If* $\Gamma \vdash M : \sigma$*, then* $\Gamma \vdash_{CL^=} M : \sigma$*.*

Proof. Assume Γ is a basis such that $\Gamma \vdash M : \sigma$. By Theorem 4.34 we have that $\Gamma \models M : \sigma$. Then by Theorem 4.41, we obtain $\Gamma \vdash_{\mathrm{CL}^=} M : \sigma$. □

4.6 Concluding remarks

In this chapter, the classical propositional extension of the simply typed combinatory logic, called the logic of combinatory logic LCL, is presented. The logic LCL is introduced in [95, 97].

The logic of combinatory logic is presented through its syntax, axiomatization and semantics. The language of the logic LCL is defined as the set of all typed terms closed under the classical propositional connectives. The logic LCL can be explained in two ways:

- LCL is the simply typed combinatory logic extended with the classical propositional connectives, and corresponding axioms and rules.

- LCL is obtained from classical the propositional logic by replacing propositional primitives with typed statements $M : \sigma$, where M is a CL-term and σ is a simple type.

The axiomatization of the logic LCL is obtained from the axiomatization of the classical propositional logic and the type assignment system for the simply typed combinatory logic. The semantics of LCL is an applicative structure provided with a valuation of term variables.

First, the soundness and completeness of the equational theory of untyped combinatory logic is proved. The soundness of the equational theory ensures that equal terms have the same interpretation in every model, whereas the completeness guarantees that if two terms have the same interpretation in every model, then they are equal. As the main results, we have proved the soundness and strong completeness of the axiomatization of LCL with respect to the proposed semantics. In the proof of strong completeness, we have adapted the Henkin-style completeness method, which is developed for the completeness proof of modal logic and became the standard way to prove completeness.

In addition, the proposed semantics of LCL has proven to be a semantics of the simply typed combinatory logic with equality. More precisely, if we consider the simply typed combinatory logic that includes the typing rule which ensures that equal terms inhabit the same type, then it is sound and complete with respect to the proposed semantics.

We end the chapter with several topics for further work.

- The motivation for the study of the propositional extension of the simply typed combinatory logic is the development of a framework for probabilistic reasoning about typed terms. So, the next step is the development of the probabilistic extension of LCL and this is the topic of the next chapter.

- The combinatory logic and λ-calculus are equivalent theories, so a natural question arises: Can the same approach be used to develop the propositional extension of the simply typed λ-calculus?

- Herein, we considered the propositional extension of typed calculus. It would be interesting to enrich the language and to study first-order extensions of simply typed calculi. This would be a step forward towards the formalization of the metalogic for simply typed theories.

- We have studied simple types, but various type systems have been developed for both the combinatory logic and λ-calculus such as intersection types, polymorphic types, higher-order types, etc. So, the extensions of

different type systems with appropriate logics in order to enable formal reasoning about these type systems is another interesting topic for the future work.

Chapter 5

Probabilistic reasoning in type theory

In this chapter, we introduce the logic PCL, which is a probabilistic extension of the logic of combinatory logic introduced in Chapter 4.

The motivation for developing a new formal model for reasoning about typed terms is the fact that reasoning with uncertainty has gained an important role in various fields of computer science, artificial intelligence and cognitive science, while it is underdeveloped in typed calculi.

Different approaches that deal with introducing non-determinism and probabilities into the typed calculi have been investigated, e.g. [4, 22, 35, 36, 45, 47, 55, 99]. However, the goal of these papers was to formalize computation in the presence of uncertainty and not to provide a framework that enables probabilistic reasoning about typed terms. Our goal is to introduce the logic in which we can express the following sentence:

The probability that a term M inhabits a type σ is at least s.

We develop our formal model along the lines of the method used for the logic LPP_2 ([128, 132]), a probabilistic extension of the classical propositional logic. The logic LPP_2 enriches the classical propositional logic with probability operators of the form $P_{\geq s}$ with the intended meaning "the probability is at least s". The iterations of the probability operators, e.g. $P_{\geq r}P_{\geq s}$, are not allowed in the logic LPP_2, therefore the logic LPP_2 cannot express higher-order probabilities, i.e. it cannot express the following: "the probability that the event has the probability s is r."

The idea of formalization of probabilistic reasoning about typed terms has been presented in [63–66, 92, 96]. In [64], we have introduced a formal model

for reasoning about probabilities of simply typed lambda terms, whereas in [66] a probabilistic extension of the λ-calculus with intersection types has been introduced. These models are based on well-known models for λ-calculus: term models for the simply typed λ-calculus ([77]) and filter models for λ-calculus with intersection types ([7]). However, these models are not suitable for propositional reasoning about typed terms. In order to achieve the strong completeness of the probabilistic extension of typed calculus, we first need to obtain the strong completeness of the propositional extension of the calculus. We could not achieve these results with respect to the existing models for λ-calculus and combinatory logic. For this reason, we have developed the logic of combinatory logic (LCL), which has been proven to be sound and complete with respect to the proposed semantics in Chapter 4.

In thus chapter, we propose a probabilistic extension of LCL as a logical framework for reasoning about typed terms.

Contribution of the chapter

- We introduce the logic PCL, which is a probabilistic extension of the logic of combinatory logic introduced in Chapter 4.
- We propose a semantics of the logic PCL based on the possible world approach.
- We prove that the logic PCL is not compact and an infinitary axiomatization is given.
- We prove that the given axiomatization of PCL is sound and complete with respect to the proposed semantics.

Overview of the chapter Section 5.1 recalls the logic LPP_2. In Section 5.2, the syntax of the logic PCL is introduced. The semantics and axiomatization of the logic PCL are given in Sections 5.3 and 5.4, respectively. Section 5.5 presents the main results about the logic PCL: soundness and strong completeness. Finally, some concluding remarks and future work are given in Section 5.6.

5.1 The logic LPP_2

In this section, we present the probability logic LPP_2 introduced in [128] by Ognjanović. The syntax, semantics and axiomatization of LPP_2 are given, followed by the main results about LPP_2, namely the soundness and strong completeness of the given axiomatization with respect to the given semantics.

5.1.1 Syntax LPP_2

The logic LPP_2 is a logic for formal reasoning about probabilities of formulas. It is a probabilistic extension of the classical propositional logic and the language of the LPP_2 logic is obtained by extending the language of the classical propositional logic with probability operators.

Let S be the set of rational numbers from the real unit interval $[0,1]$, i.e. $\mathsf{S} = \mathbb{Q} \cap [0,1]$. The alphabet of the LPP_2 consists of:

- the countable set of propositional letters (primitive propositions) $P = \{p, q, \ldots, p_1, \ldots\}$,
- classical propositional connectives $\neg$ and $\wedge$,
- the list of probability operators $P_{\geq s}$, where $s \in \mathsf{S}$.

In the logic LPP_2, a probability operator can only be applied to a classical propositional formula, thus the language of LPP_2 is defined as the union of two sets of formulas: *classical propositional formulas* and *probabilistic formulas*. The set of the classical propositional formulas is defined by the following grammar:

$$\boxed{\alpha ::= p \mid \alpha \wedge \alpha \mid \neg\alpha} \tag{5.1}$$

The set For_C denotes the set of all classical propositional formulas and is ranged over by $\alpha, \beta, \ldots, \alpha_1, \ldots$.

Definition 5.1. *Let $s \in \mathsf{S}$ and $\alpha \in \mathsf{For}_C$. The formula*

$$P_{\geq s}\alpha$$

is called a basic probabilistic formula.

The set of all probabilistic formulas, denoted by For_P, is the smallest set that contains all basic probabilistic formulas and is closed under the classical propositional connectives $\neg$ and $\wedge$.

The probabilistic formulas are generated by the grammar:

$$\boxed{\varphi := P_{\geq s}\alpha \mid \varphi \wedge \varphi \mid \neg\varphi} \tag{5.2}$$

Let $\varphi, \psi, \ldots, \varphi_1, \ldots$ range over For_P. The intended meaning of the formula $P_{\geq s}$ is "the probability of α is at least s".

The set of all LPP_2 formulas is

$$\mathsf{For}_{LPP_2} = \mathsf{For}_C \cup \mathsf{For}_P.$$

Let $A, B, \ldots$ range over For_{LPP_2}. Other classical propositional connectives $\vee, \Rightarrow$ and $\Leftrightarrow$, both for classical propositional formulas and probabilistic formulas, are defined as usual:

$$\begin{aligned} A \vee B &\text{ stands for } \neg(\neg A \wedge \neg B) \\ A \Rightarrow B &\text{ stands for } \neg A \vee B \\ A \Leftrightarrow B &\text{ stands for } (A \Rightarrow B) \wedge (B \Rightarrow A) \end{aligned}$$

The following abbreviations are used to introduce other probability operators:

$$\begin{aligned} P_{<s}\alpha &\text{ stands for } \neg P_{\geq s}\alpha \\ P_{\leq s}\alpha &\text{ stands for } P_{\geq 1-s}\neg\alpha \\ P_{>s}\alpha &\text{ stands for } \neg P_{\leq s}\alpha \\ P_{=s}\alpha &\text{ stands for } P_{\geq s}\alpha \wedge \neg P_{>s}\alpha \end{aligned}$$

The logic LPP_2 allows neither mixing of the classical propositional formulas and the probabilistic formulas nor nested probability operators. For example, the following formulas are not well-defined formulas in LPP_2:

- $(p \wedge \neg q) \wedge P_{\geq \frac{1}{2}}(\neg r)$,
- $P_{\geq \frac{1}{3}} P_{\geq \frac{1}{5}}(p \wedge \neg q)$.

Thus, in the logic LPP_2 it is not possible to express higher order probabilities.

5.1.2 Semantics of LPP_2

The semantics of LPP_2 is based on the possible world approach, where a set of possible worlds is equipped with a finitely additive probability measure.

Definition 5.2 ([105]). *A collection X of subsets of a non-empty set Ω is called* an algebra *of subsets of Ω if it has the following three properties.*

1. $\Omega \in X$.
2. *For $\Delta \subseteq \Omega$, if $\Delta \in X$, then $\Omega \setminus \Delta \in X$.*
3. *For $\Delta_1, \Delta_2, \ldots, \Delta_n \subseteq \Omega$, if $\Delta_1, \Delta_2, \ldots, \Delta_n \in X$, then $\bigcup_{i=1}^{n} \Delta_i \in X$.*

Definition 5.3 ([105]). *Let X be an algebra of subsets of Ω. A function $\mu : X \to [0,1]$ is called a finitely additive probability measure if*

- $\mu(\Omega) = 1$,
- *for any* $Y_1, Y_2 \in X$, *if* $Y_1 \cap Y_2 = \emptyset$, *then*

$$\mu(Y_1 \cup Y_2) = \mu(Y_1) + \mu(Y_2).$$

Definition 5.4 ([128]). *An LPP_2-model is a structure $\mathcal{M} = (W, H, \mu, v)$ where:*

- *W is a nonempty set of objects called worlds,*
- *H is an algebra of subsets of W,*
- *μ is a finitely additive probability measure, $\mu : H \to [0, 1]$, and*
- *$v : W \times P \to \{true, false\}$ provides a two-valued valuation of primitive propositions, for each world $w \in W$.*

The valuation of primitive propositions interprets each primitive proposition as true or false in every possible world. In order to provide interpretation for all classical propositional formulas in world w, the valuation $v(w, \cdot)$ is extended to all classical propositional formulas as usual.

Definition 5.5. *Let $\mathcal{M} = (W, H, \mu, v)$ be an LPP_2-model and $w \in W$ a possible world in $\mathcal{M}$.*

1. *The interpretation of a classical propositional formula α in world w, denoted by $I_v(w, \alpha)$, is defined as follows:*
   - $I_v(w, p) = v(w, p)$,
   - $I_v(w, \alpha \wedge \beta) = true$ *if and only if* $I_v(w, \alpha) = I_v(w, \beta) = true$, *otherwise* $I_v(w, \alpha \wedge \beta) = false$,
   - $I_v(w, \neg\alpha) = true$ *if and only if* $I_v(w, \alpha) = false$, *otherwise* $I_v(w, \neg\alpha) = false$.
2. $[\alpha]_{\mathcal{M}}$ *is the set of all worlds of model $\mathcal{M}$ in which α is true, i.e.*

$$[\alpha]_{\mathcal{M}} = \{w \mid w \in W \text{ and } I_v(w, \alpha) = true\}.$$

For a formula α and a world w, if $I_v(w, \alpha) = true$, then we write $w \models \alpha$, otherwise we write $w \not\models \alpha$. If $\mathcal{M}$ is clear from the context, then the subscript $\mathcal{M}$ from $[\alpha]_{\mathcal{M}}$ can be omitted.

In contrast to the classical propositional formulas, which are interpreted as *true* or $false$ in a possible world, probabilistic formulas are interpreted as *true* or $false$ in a model by introducing the satisfiability relation. The satisfiability relation is defined for the measurable LPP_2-models.

Definition 5.6 ([128]). *An LPP_2-model $\mathcal{M} = (W, H, \mu, v)$ is measurable if $[\alpha]_{\mathcal{M}} \in H$ for every formula $\alpha \in \mathsf{For}_C$. The class of all measurable LPP_2-models is denoted by $LPP_{2,\mathsf{Meas}}$.*

The following example illustrates the notion of a measurable LPP_2-model.

Example 5.7. *Let us consider the primitive propositions p, q, r and the structure $\mathcal{M} = (W, H, \mu, v)$ such that*

- $W = \{w_1, w_2, w_3\}$,
- $H = \mathcal{P}(W)$, *where $\mathcal{P}(W)$ is the power set of W,*
- $\mu(\emptyset) = 0$, $\mu(w_1) = \frac{3}{7}$, $\mu(w_2) = \frac{2}{7}$, $\mu(w_3) = \frac{2}{7}$, $\mu(\{w_1, w_2\}) = \frac{5}{7}$, $\mu(\{w_1, w_3\}) = \frac{5}{7}$, $\mu(\{w_2, w_3\}) = \frac{4}{7}$, $\mu(W) = 1$,
- $v(w_1, p) = v(w_1, q) = v(w_1, r) = true$, $v(w_2, q) = v(w_2, r) = true$, $v(w_2, p) = false$, $v(w_3, p) = v(w_3, q) = false$ *and* $v(w_3, r) = true$.

The structure $\mathcal{M}$ is an LPP_2-model. Indeed, W is a non-empty set and H is an algebra of subsets of W. The function $\mu : \mathcal{P}(W) \to [0, 1]$ is a finitely additive probability measure since it assigns 0 to the empty set, 1 to the entire space W and it is finitely additive, i.e. for every $E_1, E_2, \ldots E_n \subseteq W$ such that if $i \neq j$, then $E_i \cap E_j = \emptyset$, it holds that $\mu(E_1 \cup E_2 \cup \ldots \cup E_n) = \mu(E_1) + \mu(E_2) + \ldots + \mu(E_n)$. Moreover, since the function μ is defined for all subsets of W, for every $\alpha \in \mathsf{For}_C$ the set $[\alpha]_{\mathcal{M}}$ is measurable, i.e. $[\alpha]_{\mathcal{M}} \in H$. Thus, the model $\mathcal{M}$ is an $LPP_{2,\mathsf{Meas}}$-model.

Definition 5.8 ([128]). *The satisfiability relation $\models \subseteq LPP_{2,\mathsf{Meas}} \times \mathsf{For}_{LPP_2}$ is defined in the following way:*

- $\mathcal{M} \models \alpha$ *if and only if for every* $w \in W$, $v(w, \alpha) = true$.
- $\mathcal{M} \models P_{\geq s}\alpha$ *if and only if* $\mu([\alpha]) \geq s$.
- $\mathcal{M} \models \neg\varphi$ *if and only if it is not the case that* $\mathcal{M} \models \varphi$.
- $\mathcal{M} \models \varphi \wedge \psi$ *if and only if* $\mathcal{M} \models \varphi$ *and* $\mathcal{M} \models \psi$.

If $\mathcal{M} \models A$, then we say $\mathcal{M}$ is a model of formula A. Directly from Definition 5.5 and Definition 5.8 it follows that every valid classical propositional formula is true in every LPP_2-model.

The notions of satisfiable and valid formulas, satisfiable set of formulas and semantical consequence are given in the following definition.

Definition 5.9. *Let $A \in \mathsf{For}_{LPP_2}$ be a formula and let $T \subseteq \mathsf{For}_{LPP_2}$.*

- *A is satisfiable if there is a $LPP_{2,\mathsf{Meas}}$-model $\mathcal{M}$ such that $\mathcal{M} \models A$.*
- *A set of formulas T is satisfiable if there is a $LPP_{2,\mathsf{Meas}}$-model $\mathcal{M}$ such that $\mathcal{M} \models A$ for every $A \in T$.*
- *A is valid if for every $LPP_{2,\mathsf{Meas}}$-model $\mathcal{M}$, $\mathcal{M} \models A$ holds.*
- *A is a semantical consequence of the set T, denoted by $T \models A$, if it holds that A is satisfied in a model $\mathcal{M}$ (denoted by $\mathcal{M} \models A$) whenever T is satisfied in that model (denoted by $\mathcal{M} \models T$).*

The satisfiability of a formula in a model is illustrated in the following example.

Example 5.10. *Let us consider the model $\mathcal{M}$ given in 5.7. From the definition of the valuation v in $\mathcal{M}$ and Definition 5.5 it follows that*

- *$w_1 \models p \vee r$, $w_2 \models p \vee r$, $w_3 \models p \vee r$,*
- *$w_1 \models q \Rightarrow p$, $w_2 \not\models q \Rightarrow p$ and $w_3 \models q \Rightarrow p$.*

So, we conclude $\mathcal{M} \models p \vee r$. Thus, $\mathcal{M} \models P_{\geq 1}(p \vee r)$. On the other hand, the formula $q \Rightarrow p$ is not true in the world w_2, hence it holds that $\mathcal{M} \not\models q \Rightarrow p$. Since $[q \Rightarrow p]_{\mathcal{M}} = \{w_1, w_3\}$ and $\mu(\{w_1, w_3\}) = \frac{5}{7}$, we have that $\mathcal{M} \models P_{=\frac{5}{7}}(q \Rightarrow p)$. From $[\neg p]_{\mathcal{M}} = \{w_2, w_3\}$ and $\mu(\{w_2, w_3\}) = \frac{4}{7}$, we derive $\mathcal{M} \models P_{<\frac{5}{7}}(\neg p)$.

Classical propositional formulas in the logic LPP_2 do not behave in the usual way. In the classical propositional logic, if $\alpha \vee \beta$ is true in some model, then either α or β is true in the model. However, this is not the case in the logic LPP_2. Also, in the logic LPP_2, it is possible to have $\mathcal{M} \not\models \alpha$ and $\mathcal{M} \not\models \neg\alpha$. We illustrate this in the following example.

Example 5.11. *Let us consider the model $\mathcal{M} = (W, H, \mu, v)$ where:*

- *$W = \{w_1, w_2, w_3\}$,*
- *$H = \mathcal{P}(W)$,*
- *$\mu(\emptyset) = 0$, $\mu(w_1) = \mu(w_2) = \mu(w_3) = \frac{1}{3}$, $\mu(\{w_1, w_2\}) = \mu(\{w_1, w_3\}) = \mu(\{w_2, w_3\}) = \frac{2}{3}$, $\mu(W) = 1$,*
- *$v(w_1, p) = v(w_2, p) = true$ and $v(w_3, p) = false$.*

  Since $w_1 \not\models \neg p$, we have that $\mathcal{M} \not\models \neg p$. Similarly, from $w_3 \not\models p$, it follows that $\mathcal{M} \not\models p$. However, $p \vee \neg p$ is true in every world w_i, $i \in \{1, 2, 3\}$, thus $\mathcal{M} \models p \vee \neg p$.

The logic LPP_2 is not compact. We do not discuss non-compactness of the logic LPP_2, since we prove non-compactness of the logic PCL in Section 5.3. It follows by the similar reasoning as the non-compactness of the logic LPP_2.

5.1.3 Axiomatization of LPP_2

The axiomatic system of LPP_2 consists of six axiom schemes and three inference rules given in Figure 5.1.

(1) all instances of the classical propositional tautologies (atoms are any LPP_2-formulas)
(2) $P_{\geq 0}\alpha$
(3) $P_{\leq r}\alpha \Rightarrow P_{<s}\alpha,\ s > r$
(4) $P_{<s}\alpha \Rightarrow P_{\leq s}\alpha$
(5) $(P_{\geq r}\alpha \wedge P_{\geq s}\beta \wedge P_{\geq 1}(\neg\alpha \vee \neg\beta)) \Rightarrow P_{\geq \min\{1, r+s\}}(\alpha \vee \beta)$
(6) $(P_{\leq r}\alpha \wedge P_{<s}\beta) \Rightarrow P_{<r+s}(\alpha \vee \beta),\ r + s \leq 1$

(R1) $\dfrac{A \qquad A \Rightarrow B}{B}$ (R2) $\dfrac{\alpha}{P_{\geq 1}\alpha}$ (R3) $\dfrac{\{\psi \Rightarrow P_{\geq s-\frac{1}{k}}\alpha \mid k \geq \frac{1}{s}\}}{\psi \Rightarrow P_{\geq s}\alpha}$

FIGURE 5.1: The axiomatic system of LPP_2

Axiom scheme (1) and inference rule (R1) ensure that the classical propositional logic is a sublogic of LPP_2. Axiom schemes (2) – (6) deal with probabilistic part of LPP_2. The measure of a set of worlds that satisfy a classical propositional formula is at least 0 by the axiom scheme (2). The monotonicity of measures is ensured by the axiom schemes (3) and (4). Axiom schemes (5) and (6) correspond to the additivity of measures.

Inference rule (R1) is Modus Ponens. The rule (R2) can be considered as the rule of necessitation in modal logics. The iterations of probability operators are not allowed in LPP_2, so the rule (R2) can be applied only to classical propositional formulas. The third rule is an infinitary rule with the countable set of assumptions and one conclusion. Intuitively, the rule (R3) guarantees that if the probability is arbitrary close to s, then it is at least s.

The presented axiomatization of LPP_2 is not minimal, in the sense that the set of axioms can be reduced. The first axiom scheme comprises all tautologies of classical propositional logic, however instead of including all tautologies

we could include just three axiom schemes from the axiomatic system of the classical propositional logic, as it was done in Chapter 4. Then all tautologies of the classical propositional logic are derivable from the axiom schemes using the first inference rule.

The proof is defined similarly as in Chapter 4. The only difference is that it can be a countable sequence of formulas, whereas in Chapter 4 it is a finite sequence. This is due to the infinitary rule in the axiomatization of LPP_2.

Definition 5.12. *A formula φ is deducible from a set T of formulas (denoted by $T \vdash_{LPP_2} \varphi$) if there is a sequence $\varphi_0, \ldots, \varphi_n$ (n is a finite or countable ordinal) of For_{LPP_2}-formulas, such that*

- *$\varphi_n = \varphi$, and*
- *every φ_i , $i \leq n$, is an axiom-instance, or $\varphi_i \in T$, or φ_i is derived by an inference rule applied to some previous members of the sequence.*

A proof for φ from T is the corresponding sequence of formulas.

The definition of a consistent set of LPP_2-formulas is again similar to the definition of a consistent set of LCL-formulas in Chapter 4.

Definition 5.13. *A set T of formulas is* consistent *if there is at least a formula from For_C and at least a formula from For_P that are not deducible from T, otherwise T is* inconsistent.

Recall that in the definition of a consistent set of LCL-formulas, it is required that there is at least one LCL-formula not derivable from the set. However, in the case of a consistent set of LPP_2-formulas it is not enough to have at least one LPP_2-formula not derivable from the set. There have to be at least one formula from the set For_C and at least one formula from the set For_P that are not derivable from the set, because For_C and For_P are disjoint.

5.1.4 Soundness and strong completeness of LPP_2

The main results about the logic LPP_2 presented in [128] are soundness and strong completeness of the given axiomatization with respect to the proposed semantics. We state these theorems without their proof, which can be found in [128, 132].

Theorem 5.14 (Soundness of LPP_2)**.** *The axiomatic system of LPP_2 is sound with respect to the class of $LPP_{2,\mathsf{Meas}}$-models.*

Theorem 5.15 (Strong completeness of LPP_2)**.** *Every consistent set of LPP_2-formulas T is $LPP_{2,\mathsf{Meas}}$-satisfiable.*

The proof of the strong completeness of LPP_2 is obtained by adapting the Henkin-style completeness method, as in the proof of the strong completeness of LCL in Chapter 4. The main steps in the proof are as follows:

- the proof of Deduction theorem,
- the proof that a consistent set of formulas can be extended to a maximal consistent set,
- the construction of a canonical model,
- the proof that a canonical model is an LPP_2-model,
- the proof that every consistent set is satisfiable,
- the proof of strong completeness.

5.2 Syntax PCL

In this section, the language of PCL is introduced. As it is discussed, PCL is a probabilistic extension of the logic of combinatory logic LCL introduced in Chapter 4. The alphabet of PCL comprises the alphabet of LCL and the alphabet of probability logic. So, it consists of:

- the alphabet of the simply typed combinatory logic, that is all symbols needed to define simply typed CL-terms, given in Section 4.1,
- the classical propositional connectives $\neg$, $\wedge$ and $\Rightarrow$,
- the list of probability operators $P_{\geq s}$, for every $s \in \mathsf{S} = [0,1] \cap \mathbb{Q}$.

Similarly to the language of LPP_2, the language of PCL is layered into two sets of formulas: *basic formulas* and *probabilistic formulas*.

Basic formulas are LCL formulas introduced in Section 4.2.1. Let us recall, the LCL-formulas are generated by the following grammar

$$\boxed{\alpha := M : \sigma \mid \neg\alpha \mid \alpha \Rightarrow \alpha} \tag{5.3}$$

where $M : \sigma \in CL_{\rightarrow}$. Herein, the set of LCL-formulas (i.e. basic formulas) is denoted by $\mathsf{For_B}$.

Probabilistic formulas are obtained by applying the probability operator to LCL-formulas. For $s \in \mathsf{S}$ and an LCL-formula α, the formula $P_{\geq s}\alpha$ is called *a basic probabilistic formula*. The set of all probabilistic formulas is the set of basic probabilistic formulas closed under the classical propositional connectives $\neg$ and $\wedge$, i.e

$$\boxed{\varphi ::= P_{\geq s}\alpha \mid \varphi \wedge \varphi \mid \neg\varphi} \tag{5.4}$$

$\mathsf{For_P}$ denotes the set of all probabilistic formulas and $\varphi, \psi, \ldots, \varphi_1, \ldots$ range over $\mathsf{For_P}$.

The language of PCL is the union of basic and probabilistic formulas.

$$\mathsf{For}_{PCL} = \mathsf{For_B} \cup \mathsf{For_P}$$

The set of all PCL-formulas is denoted by For_{PCL} and is ranged over by $A, B, C, \ldots$.

The same abbreviations as in Section 5.1 are used to introduce other probability operators (e.g. $P_{<s}$, $P_{>s}$, $P_{\leq s}$ and $P_{=s}$).

In the definition of LCL-formulas only two classical propositional connectives are used, namely negation and implication, whereas in the definition of probabilistic formulas only negation and conjunction are used. Nevertheless, other classical propositional connectives are defined as usual (see Section 5.1.1). Both $\alpha \wedge \neg\alpha$ and $\varphi \wedge \neg\varphi$ are denoted by $\bot$. The logic PCL is based on the probability logic LPP_2, so mixing of basic formulas and probabilistic formulas, and nested probability operators are not allowed. For example, the following formulas are not well-defined formulas in PCL:

- $(x : \sigma) \wedge P_{\geq \frac{1}{3}}(x : \sigma \to \tau)$,
- $P_{\geq \frac{1}{2}} P_{\geq \frac{1}{3}}(x : \sigma \to \tau)$.

5.3 Semantics of PCL

In this section, the semantics of PCL is introduced. Following the approach used for the logic LPP_2, the semantics of the logic PCL is based on the possible world approach, where the set of possible worlds is equipped with a finitely additive probability measure.

Definition 5.16. *A PCL-model is a structure*

$$\mathcal{M} = (W, \{D_w\}, \{A^\sigma_w\}, \{\cdot_w\}, \{\boldsymbol{s}_w\}, \{\boldsymbol{k}_w\}, \{\boldsymbol{i}_w\}, H, \mu, \rho)$$

where:

- *W is a non-empty set of objects, called* possible worlds*,*
- *$\{D_w\} = \{D_w\}_{w \in W}$ is a family of sets indexed by worlds, where the set D_w is referred to as* the domain *of the world w,*

- $\{A^\sigma_w\} = \{A^\sigma_w\}_{w\in W,\sigma\in \mathsf{Types}_\to}$ *is a family of sets indexed by types* σ *and worlds* w *such that* $A^\sigma_w \subseteq D_w$ *for all* $w \in W$ *and* $\sigma \in \mathsf{Types}_\to$.
- $\{\cdot_w\} = \{\cdot_w\}_{w\in W}$ *is a family of binary operations indexed by worlds such that the following hold:*
  - $\cdot_w$ *is a binary operation on* D_w, *i.e.* $\cdot_w : D_w \times D_w \to D_w$,
  - $\cdot_w$ *is extensional, that is for every* $w \in W$ *and every* $d_1, d_2 \in D_w$, *if* $(\forall e \in D_w)(d_1 \cdot_w e = d_2 \cdot_w e)$, *then* $d_1 = d_2$,
  - *for every* $\sigma, \tau \in \mathsf{Types}_\to$, *it holds that the codomain of the restriction of function* $\cdot_w$ *to the set* $A^{\sigma\to\tau}_w \times A^\sigma_w$ *is* A^τ_w,
- $\{\boldsymbol{s}_w\} = \{\boldsymbol{s}_w\}_{w\in W}$ *is a family of elements indexed by worlds such that for every* $w \in W$ *the following hold:*
  - $\boldsymbol{s}_w \in D_w$,
  - *for every* $\sigma, \tau, \rho \in \mathsf{Types}_\to$,
$$\boldsymbol{s}_w \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}_w \tag{5.5}$$
  - *and for every* $d, e, f \in D_w$,
$$((\boldsymbol{s}_w \cdot_w d) \cdot_w e) \cdot_w f = (d \cdot_w f) \cdot (e \cdot_w f) \tag{5.6}$$
- $\{\boldsymbol{k}_w\} = \{\boldsymbol{k}_w\}_{w\in W}$ *is a family of elements indexed by worlds such that for every* $w \in W$ *the following hold:*
  - $\boldsymbol{k}_w \in D_w$,
  - *for every* $\sigma, \tau \in \mathsf{Types}_\to$,
$$\boldsymbol{k}_w \in A^{\sigma\to(\tau\to\sigma)}_w \tag{5.7}$$
  - *and for every* $d, e \in D_w$,
$$(\boldsymbol{k}_w \cdot_w d) \cdot_w e = d \tag{5.8}$$
- $\{\boldsymbol{i}_w\} = \{\boldsymbol{i}_w\}_{w\in W}$ *is a family of elements indexed by worlds such that for every* $w \in W$ *the following hold:*
  - $\boldsymbol{i}_w \in D_w$,
  - *for every* $\sigma \in \mathsf{Types}_\to$,
$$\boldsymbol{i}_w \in A^{\sigma\to\sigma}_w \tag{5.9}$$

– *and for every* $d \in D_w$,

$$\boldsymbol{i}_w \cdot_w d = d \tag{5.10}$$

- H *is an algebra of subsets of* W.
- μ *is a finitely additive probability measure,* $\mu : H \to [0,1]$.
- $\rho : W \times V \to \bigcup_{w \in W} D_w$ *provides for each world a two-valued valuation of term variables such that for every* $w \in W$, $\rho(w, \cdot)$ *is a map from the set of term variables to the domain* D_w, *i.e.* $\rho(w, \cdot) : V \to D_w$.

The following proposition gives the connection between LCL-models and PCL-models and it is a direct consequence of Definition 5.16.

Proposition 5.17. *Let*

$$\mathcal{M} = (W, \{D_w\}, \{A^\sigma_w\}, \{\cdot_w\}, \{\boldsymbol{s}_w\}, \{\boldsymbol{k}_w\}, \{\boldsymbol{i}_w\}, H, \mu, \rho)$$

be a PCL*-model. For each* $w \in W$, *the structure*

$$\mathcal{M}_w = \langle D_w, \{A^\sigma_w\}_\sigma, \cdot_w, \boldsymbol{s}_w, \boldsymbol{k}_w, \boldsymbol{i}_w \rangle$$

is an applicative structure for LCL*and* $\mathcal{M}_{\rho_w} = \langle \mathcal{M}_w, \rho(w, \cdot) \rangle$ *is an* LCL*-model.*

Note that the similar proposition holds for an LPP_2-model. For every world w in an LPP_2-model, the function $v(w, \cdot)$ is a valuation of primitive propositions, that is each world generates one model of classical propositional logic. Similarly, in a PCL-model each world generates one LCL-model.

In order to define satisfiability of a formula in a model, we first introduce the notion of satisfiability of a basic formula α in a possible world w of a model $\mathcal{M}$.

Definition 5.18. *Let* $\mathcal{M} = (W, \{D_w\}, \{A^\sigma_w\}, \{\cdot_w\}, \{\boldsymbol{s}_w\}, \{\boldsymbol{k}_w\}, \{\boldsymbol{i}_w\}, H, \mu, \rho)$ *be a* PCL*-moodel,* w' *a possible world in* $\mathcal{M}$ *and* α *a basic formula. The formula* α *is satisfied in a world* w', *denoted by* $w' \models \alpha$ *if and only if* α *is satisfied by the* LCL*-model* $\mathcal{M}_{\rho_{w'}} = \langle \mathcal{M}_{w'}, \rho_{w'} \rangle$ *where* $\mathcal{M}_{w'} = \langle D_{w'}, \{A^\sigma_{w'}\}_\sigma, \cdot_{w'}, \boldsymbol{s}_{w'}, \boldsymbol{k}_{w'}, \boldsymbol{i}_{w'} \rangle$ *and* $\rho_{w'}(x) = \rho(w', x)$.

We define the class of measurable PCL-models, as in the logic LPP_2. The set $[\alpha]_\mathcal{M}$ is defined as in the logic LPP_2 (Definition 5.5).

Definition 5.19. *A* PCL*-model*

$$\mathcal{M} = (W, \{D_w\}, \{A^\sigma_w\}, \{\cdot_w\}, \{\boldsymbol{s}_w\}, \{\boldsymbol{k}_w\}, \{\boldsymbol{i}_w\}, H, \mu, \rho)$$

is measurable *if* $[\alpha]_\mathcal{M} \in H$ *for every formula* $\alpha \in \mathsf{For_B}$. *The class of all measurable* PCL*-models is denoted by* PCL_{Meas}.

Similarly as in the logic LPP_2, the satisfiability relation is defined for the measurable PCL-models.

Definition 5.20. *The satisfiability relation* $\models \subseteq PCL_{\mathsf{Meas}} \times \mathsf{For}_{PCL}$ *is defined in the following way:*

- $\mathcal{M} \models \alpha$ *if and only if for every* $w \in W$, $w \models \alpha$.
- $\mathcal{M} \models P_{\geq s}\alpha$ *if and only if* $\mu([\alpha]) \geq s$.
- $\mathcal{M} \models \neg\varphi$ *if and only if it is not the case that* $\mathcal{M} \models \varphi$.
- $\mathcal{M} \models \varphi \wedge \psi$ *if and only if* $\mathcal{M} \models \varphi$ *and* $\mathcal{M} \models \psi$.

The notions of satisfiable formulas, valid formulas and semantical consequences are defined the same as in the logic LPP_2 (see Definition 5.9).

Compactness We say that the compactness theorem holds for some logic L if the following holds:

> The set X of formulas of the logic L is satisfiable if and only if every finite subset of X is satisfiable.

The compactness theorem does not hold for the logic PCL. In the following example, we present a set X of PCL-formulas such that each finite subset of X is satisfiable, but the set X is not satisfiable.

Example 5.21. *Let us consider the set*

$$X = \{\neg P_{=0}(x : \sigma)\} \cup \{P_{<\frac{1}{n}}(x : \sigma) \mid n \in \mathbb{N}\}.$$

For every finite subset X' *of* X, *there exists the largest* $k \in \mathbb{N}$ *such that* $P_{<\frac{1}{k}}(x : \sigma) \in X'$. *We show that* X' *is satisfiable. Since* $\{x : \sigma\}$ *is a consistent set of LCL-formulas, there is an LCL-model* $\mathcal{M}^1_{\rho_1}$ *such that* $\mathcal{M}^1_{\rho_1} \models x : \sigma$ *by Theorem 4.33. Let* $\mathcal{M}^1_{\rho_1} = \langle \mathcal{M}^1, \rho_1 \rangle$ *and* $\mathcal{M}^1 = \langle D_1, \{A^\sigma_1\}, \cdot_1, \boldsymbol{s}_1, \boldsymbol{k}_1, \boldsymbol{i}_1 \rangle$. *On the other hand,* $x : \sigma$ *is not a theorem of the logic LCL, so it is not true in every LCL-model by Theorem 4.24. Hence, there exists an LCL-model* $\mathcal{M}^2_{\rho_2}$ *such that* $\mathcal{M}^2_{\rho_2} \not\models x : \sigma$. *Let* $\mathcal{M}^2_{\rho_2} = \langle \mathcal{M}^2, \rho_2 \rangle$ *and* $\mathcal{M}^2 = \langle D_2, \{A^\sigma_2\}, \cdot_2, \boldsymbol{s}_2, \boldsymbol{k}_2, \boldsymbol{i}_2 \rangle$.

Now, we construct a PCL-model $\mathcal{M}'$ *such that* $\mathcal{M}' \models X'$. *Let* $\mathcal{M}'$ *be the following structure*

$$\mathcal{M}' = (W, \{D_w\}, \{A^\sigma_w\}, \{\cdot_w\}, \{\boldsymbol{s}_w\}, \{\boldsymbol{k}_w\}, \{\boldsymbol{i}_w\}, H, \mu, \rho),$$

where:

- $W = \{w_1, w_2\}$,
- $D_{w_i} = D_i$, $i \in \{1, 2\}$,
- $A^{\sigma}_{w_i} = A^{\sigma}_i$ *for every* σ *and* $i \in \{1, 2\}$,
- $\cdot_{w_i} = \cdot_i$, $i \in \{1, 2\}$,
- $\boldsymbol{s}_{w_i} = \boldsymbol{s}_i$, $i \in \{1, 2\}$,
- $\boldsymbol{k}_{w_i} = \boldsymbol{k}_i$, $i \in \{1, 2\}$,
- $\boldsymbol{i}_{w_i} = \boldsymbol{i}_i$, $i \in \{1, 2\}$,
- $H = \mathcal{P}(W)$,
- $\mu(w_1) = \frac{1}{k+1}$ *and* $\mu(w_2) = \frac{k}{k+1}$,
- $\rho(w_i, x) = \rho_i(x)$.

From the construction of the model $\mathcal{M}'$*, we have that* $w_1 \models x : \sigma$ *and* $w_2 \not\models x : \sigma$*. Thus,* $\mu([x : \sigma]) = \frac{1}{k+1}$ *and* $\mathcal{M}' \models X'$*. However, the set* X *is not satisfiable. Let* $\mathcal{M}'$ *be a PCL-model. For every* $m > 0$*, if* $\mu([x : \sigma]) = m$*, then there exists* $n_0 \in \mathbb{N}$ *such that* $\frac{1}{n_0} < m$ *and* $\mathcal{M}' \not\models P_{< \frac{1}{n_0}} x : \sigma$*. If* $m = 0$*, then* $\mathcal{M}' \not\models \neg P_{=0}(x : \sigma)$.

A consequence of the non-compactness is that any finite axiomatization of the logic PCL which is sound cannot be strongly complete (meaning that if a formula semantically follows from a set T, then this formula is derivable from T). Let us suppose the opposite, i.e. that there is a finite axiomatization of PCL that is sound and strongly complete. Let X be an infinite set of formulas such that every subset of X is satisfiable and X itself is not. From the strong completeness of the axiomatization, it follows that the set X is inconsistent, since it is unsatisfiable. So, it holds that $X \vdash \bot$. Since the axiomatization is finite, the proof of $X \vdash \bot$ has to be a finite sequence of formulas. Thus, there exists a finite subset $X' \subseteq X$ such that $X' \vdash \bot$. Then X' is also inconsistent. Furthermore, we conclude X' is unsatisfiable by the soundness of the axiomatization. This contradicts the assumption that every finite subset of X is satisfiable.

Since the goal is to give the axiomatization which is sound and strongly complete with respect to the proposed semantics, in the following section we give an infinitary axiomatization of the logic PCL.

5.4 Axiomatization of PCL

In this section, the axiomatic system of PCL, denoted by Ax_{PCL}, is introduced. The axiomatic system is obtained from the axiomatic system of the logic LCL and the axiomatic system of the logic LPP_2, in the way we explain below. It consists of 14 axiom schemes and three inference rules given in Figure 5.2.

We briefly discuss the axiomatic system of PCL.

- Axiom schemes $(1)-(5)$ are axiom schemes for the logic LCL given in Figure 4.3.
- Axiom schemes $(6)-(10)$ are axiom schemes from the axiomatic system of the logic LPP_2 and they are concerned with the probabilistic aspect of the logic PCL.
- Axiom scheme (11) ensures that equivalent formulas have equal measures.
- The last three axiom schemes are axiom schemes for the classical propositional logic. Atoms in the axiom schemes $(12)-(14)$ are any probabilistic formulas.
- The inference rules are the same as in the axiomatization of LPP_2.

Note that we did not follow the approach used for LPP_2 and listed all classical tautologies as axioms. We left out the first axiom scheme of LPP_2 and added three axioms schemes for the classical propositional logic, where atoms can be any probabilistic formulas of PCL. From the axiom schemes $(12)-(14)$ and the inference rule (R1) all classical propositional tautologies, where atoms are any PCL-formulas, can be derived.

Proposition 5.22. *Axiom schemes* (7) *and* (8) *are equivalent to the following formulas:*

$$(7')\ P_{\geq t}\alpha \Rightarrow P_{>s}\alpha,\ t > s,$$
$$(8')\ P_{>s}\alpha \Rightarrow P_{\geq s}\alpha,$$

respectively.

Proof. The proof follows directly from the definition of probability operators. □

The notions of proof and consistent set in the logic PCL are defined as in the logic LPP_2 in Definition 5.12 and Definition 5.13, respectively. If there

Axiom schemes:

(1) $\mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$
(2) $\mathsf{K} : \sigma \to (\tau \to \sigma)$
(3) $\mathsf{I} : \sigma \to \sigma$
(4) $(M : \sigma \to \tau) \Rightarrow ((N : \sigma) \Rightarrow (MN : \tau))$,
$M : \sigma \to \tau, N : \sigma, MN : \tau \in CL_{\to}$
(5) $M : \sigma \Rightarrow N : \sigma,$ if $M = N, M : \sigma, N : \sigma \in CL_{\to}$
(6) $P_{\geq 0}\alpha$
(7) $P_{\leq r}\alpha \Rightarrow P_{<s}\alpha, s > r$
(8) $P_{<s}\alpha \Rightarrow P_{\leq s}\alpha$
(9) $(P_{\geq r}\alpha \wedge P_{\geq s}\beta \wedge P_{\geq 1}(\neg\alpha \vee \neg\beta)) \Rightarrow P_{\geq \min\{1, r+s\}}(\alpha \vee \beta)$
(10) $(P_{\leq r}\alpha \wedge P_{<s}\beta) \Rightarrow P_{<r+s}(\alpha \vee \beta), r + s \leq 1$
(11) $P_{\geq 1}(\alpha \Rightarrow \beta) \Rightarrow (P_{\geq s}\alpha \Rightarrow P_{\geq s}\beta)$
(12) $A \Rightarrow (B \Rightarrow A)$
(13) $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$
(14) $(\neg A \Rightarrow \neg B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow A)$

Inference rules:

$$\text{(R1)}\ \frac{A \quad A \Rightarrow B}{B}$$

$$\text{(R2)}\ \frac{\alpha}{P_{\geq 1}\alpha}$$

$$\text{(R3)}\ \frac{\{\varphi \Rightarrow P_{\geq s - \frac{1}{k}}\alpha \mid k \geq \frac{1}{s}\}}{\varphi \Rightarrow P_{\geq s}\alpha}$$

FIGURE 5.2: The axiomatic system of PCL

is a proof for formula A from a set T of formulas in the logic PCL, we write $T \vdash_{PCL} A$. We will omit the subscript PCL when there is no ambiguity in the context. For a formula A, we will write $T, A \vdash B$ to denote $T \cup \{A\} \vdash B$.

If the proof of formula $A \in \mathsf{For}_{PCL}$ from the set T is obtained using only axiom schemes $(12) - (14)$ and the inference rule (R1), then we say that $T \vdash A$ is obtained by propositional reasoning.

In the following definitions the notions of a maximal consistent set and a deductively closed set are introduced.

Definition 5.23. *A set T is* maximally consistent *if it is consistent and:*

1. *for every $\alpha \in \mathsf{For_B}$, if $T \vdash \alpha$, then $\alpha \in T$ and $P_{\geq 1}\alpha \in T$,*
2. *for every $\varphi \in \mathsf{For_P}$, either $\varphi \in T$ or $\neg\varphi \in T$.*

Definition 5.24. *A set T is* deductively closed *if $T \vdash A$ implies $A \in T$, for every $A \in \mathsf{For}_{PCL}$.*

A crucial step in the completeness proof is the Deduction theorem.

Theorem 5.25 (Deduction theorem)**.** *Let T be a set of PCL-formulas and $\varphi, \psi \in \mathsf{For_P}$. If $T, \varphi \vdash \psi$, then $T \vdash \varphi \Rightarrow \psi$.*

Proof. The proof proceeds by the induction on the length of the proof of $T, \varphi \vdash \psi$.

If the length of the proof is 1, then ψ is either an instance of an axiom scheme or it belongs to the set $T \cup \{\varphi\}$.

- In the case that ψ is an axiom, we have

  1. $T \vdash \psi$, by Definition 5.12,
  2. $T \vdash \psi \Rightarrow (\varphi \Rightarrow \psi)$, by axiom scheme (12),
  3. $T \vdash \varphi \Rightarrow \psi$, from (1) and (2) by (R1).

- If $\psi \in T$, then we have

  1. $T \vdash \psi$, by Definition 5.12,
  2. $T \vdash \psi \Rightarrow (\varphi \Rightarrow \psi)$, by axiom scheme (12),
  3. $T \vdash \varphi \Rightarrow \psi$, from (1) and (2) by (R1).

- If $\psi \in \{\varphi\}$, i.e. $\varphi = \psi$ then

  $$T \vdash \varphi \Rightarrow \varphi,$$

  since $\varphi \Rightarrow \varphi$ is a tautology in the classical propositional logic.

Let us assume that the statement holds when the length of the proof i is $1 < i < k$ and prove that it holds when the length of the proof is k.

If the length of the proof is $k > 1$, then ψ is either an axiom or belongs to the set $T \cup \{\varphi\}$ or is obtained by an application of some inference rule. If it is an axiom or belongs to the set $T \cup \{\varphi\}$, the proof proceeds the same as in the base case.

Let ψ be obtained by applying the rule (R1) to some formulas ψ_1 and $\psi_1 \Rightarrow \psi$ such that $T, \varphi \vdash \psi_1$ and $T, \varphi \vdash \psi_1 \Rightarrow \psi$. Then

1. $T, \varphi \vdash \psi_1$, the assumption,
2. $T, \varphi \vdash \psi_1 \Rightarrow \psi$, the assumption,
3. $T \vdash \varphi \Rightarrow \psi_1$, by the induction hypothesis,
4. $T \vdash \varphi \Rightarrow (\psi_1 \Rightarrow \psi)$, by the induction hypothesis,
5. $T \vdash (\varphi \Rightarrow (\psi_1 \Rightarrow \psi)) \Rightarrow ((\varphi \Rightarrow \psi_1) \Rightarrow (\varphi \Rightarrow \psi))$, by axiom scheme(13),
6. $T \vdash (\varphi \Rightarrow \psi_1) \Rightarrow (\varphi \Rightarrow \psi)$, from (2) and (3) by (R1),
7. $T \vdash \varphi \Rightarrow \psi$, from (1) and (4) by (R1).

If ψ is of the form $P_{\geq 1}\alpha$ and it is obtained by applying the rule (R2) to $T, \varphi \vdash \alpha$, then we have $T \vdash \alpha$. This is due to the fact that $\alpha \in \mathsf{For_B}$ and $\varphi \in \mathsf{For_P}$, so φ cannot affect the proof of α from $T \cup \{\varphi\}$. Furthermore, we obtain

1. $T \vdash \alpha$, by the assumption,
2. $T \vdash P_{\geq 1}\alpha$, by the rule (R2) applied to (1),
3. $T \vdash P_{\geq 1}\alpha \Rightarrow (\varphi \Rightarrow P_{\geq 1}\alpha)$, by axiom scheme (12),
4. $T \vdash \varphi \Rightarrow P_{\geq 1}\alpha$, from (2) and (3) by (R1).

Finally, we consider the case when ψ is of the form $\psi_1 \Rightarrow P_{\geq s}\alpha$ and is obtained by applying the rule (R3) to the set of formulas $\{\psi_1 \Rightarrow P_{\geq s-\frac{1}{k}}\alpha \mid k \geq \frac{1}{s}\}$ such that $T, \varphi \vdash \psi_1 \Rightarrow P_{\geq s-\frac{1}{k}}\alpha$, for all $k \geq \frac{1}{s}$. Then we derive

1. $T, \varphi \vdash \psi_1 \Rightarrow P_{\geq s-\frac{1}{k}}\alpha$, the assumption,
2. $T \vdash \varphi \Rightarrow (\psi_1 \Rightarrow P_{\geq s-\frac{1}{k}}\alpha)$, by the induction hypothesis,
3. $T \vdash (\varphi \wedge \psi_1) \Rightarrow P_{\geq s-\frac{1}{k}}\alpha$, by propositional reasoning,
4. $T \vdash (\varphi \wedge \psi_1) \Rightarrow P_{\geq s}\alpha$, by the rule (R3),
5. $T \vdash \varphi \Rightarrow (\psi_1 \Rightarrow P_{\geq s}\alpha)$, by propositional reasoning.

□

5.5 Soundness and strong completeness of PCL

This section presents the main results about the logic PCL: its soundness and strong completeness with respect to the proposed semantics.

Theorem 5.26 (Soundness of PCL)**.** *The axiomatic system Ax_{PCL} is sound with respect to the class of PCL_{Meas}-models.*

Proof. We prove that every instance of an axiom scheme is true in every PCL-model and that the inference rules preserve validity.

Axioms schemes $(1)-(5)$ belong to the axiomatic system of LCL. Since each world w of a PCL-model $\mathcal{M}$ generates an LCL-model $\mathcal{M}_w$ by Proposition 5.17 and each instance α of the axiom schemes $(1)-(5)$ is true in every LCL-model by Theorem 4.24, we conclude $\mathcal{M}_w \models \alpha$. Similarly, every instance of axiom schemes $(12)-(14)$ where atoms are LCL-formulas, i.e. basic formulas, is an instance of an axiom in the logic LCL. Thus, each instance of axiom schemes $(12)-(14)$ is satisfied in model $\mathcal{M}_w$. Now, for every possible world w from the model $\mathcal{M}$, $w \models \alpha$ by Definition 5.18. By Definition 5.20, we obtain $\mathcal{M} \models \alpha$, for every PCL_{Meas}-model $\mathcal{M}$.

Let us consider the axiom scheme (6); let $P_{\geq 0}\alpha$ be one of its instances and $\mathcal{M}$ a PCL_{Meas}-model. We have that the range of μ is $[0,1]$ by the definition of probability measure μ, so $\mu(X) \geq 0$, for every $X \in H$. Thus, $\mu([\alpha]_{\mathcal{M}}) \geq 0$ for every $\alpha \in \mathsf{For_B}$. We conclude that $\mathcal{M} \models P_{\geq o}\alpha$.

An instance of the axiom scheme (7) is of the form $P_{\leq r}\alpha \Rightarrow P_{<s}\alpha$, for some formula $\alpha \in \mathsf{For_B}$ and numbers $r, s \in \mathsf{S}$ such that $s > r$. We prove that if $P_{\leq r}\alpha$ holds in a PCL-model $\mathcal{M}$, then $P_{<s}\alpha$ also holds in $\mathcal{M}$. From $\mathcal{M} \models P_{\leq r}\alpha$, we obtain $\mu([\alpha]_{\mathcal{M}}) \leq r$ by Definition 5.20. Since $r < s$, we have that $\mu([\alpha]_{\mathcal{M}}) < s$. Thus, $\mathcal{M} \models P_{<s}\alpha$ and each instance of axiom scheme (7) is satisfied by any PCL_{Meas}-model.

Similarly, in order to prove that each instance of the axiom scheme (8) is satisfied by any PCL_{Meas}-model $\mathcal{M}$, we have to prove that if $\mathcal{M} \models P_{<s}\alpha$, then $\mathcal{M} \models P_{\leq s}\alpha$. From $\mathcal{M} \models P_{<s}\alpha$, it follows that $\mu([\alpha]_{\mathcal{M}}) < s$, and since $s \geq s$ we conclude $\mu([\alpha]_{\mathcal{M}}) \leq s$. Hence, $\mathcal{M} \models P_{\leq s}\alpha$.

Next, let us consider an instance of the axiom scheme (9). We prove that if $P_{\geq r}\alpha$, $P_{\geq s}\beta$ and $P_{\geq 1}(\neg\alpha \vee \neg\beta)$ hold in a PCL_{Meas}-model $\mathcal{M}$, then $P_{\geq min\{1, r+s\}}(\alpha \vee \beta)$ also holds in $\mathcal{M}$. Let us assume that $\mathcal{M} \models P_{\geq r}\alpha$, $\mathcal{M} \models P_{\geq s}\beta$ and $\mathcal{M} \models P_{\geq 1}(\neg\alpha \vee \neg\beta)$, which is equivalent to $\mu([\alpha]_{\mathcal{M}}) \geq r$, $\mu([\beta]_{\mathcal{M}}) \geq s$ and $\mu([\neg\alpha \vee \neg\beta]_{\mathcal{M}}) \geq 1$, respectively. From $\mu([\neg\alpha \vee \neg\beta]_{\mathcal{M}}) \geq 1$, we obtain $\mu([\alpha \wedge \beta]_{\mathcal{M}}) \leq 0$, which implies $\mu([\alpha \wedge \beta]_{\mathcal{M}}) = 0$. Thus, the sets

$[\alpha]_{\mathcal{M}}$ and $[\beta]_{\mathcal{M}}$ are disjoint. By the definition of finitely additive probability measure, we have

$$\mu([\alpha \vee \beta]_{\mathcal{M}}) = \mu([\alpha]_{\mathcal{M}} \cup [\beta]_{\mathcal{M}}) = \mu([\alpha]_{\mathcal{M}}) + \mu([\beta]_{\mathcal{M}}) = r + s.$$

Hence, $\mathcal{M} \models P_{\geq \min\{1, r+s\}}(\alpha \vee \beta)$.

In order to prove that an instance of the axiom scheme (10) is satisfied by any PCL_{Meas}-model $\mathcal{M}$, we prove that if $\mathcal{M} \models P_{\leq r}\alpha$ and $\mathcal{M} \models P_{<s}\beta$, where $r + s \leq 1$, then $\mathcal{M} \models P_{<r+s}(\alpha \vee \beta)$. From $\mathcal{M} \models P_{\leq r}\alpha$ it follows that $\mu([\alpha]_{\mathcal{M}}) \leq r$. Similarly, from $P_{<s}\beta$, we obtain $\mu([\beta]_{\mathcal{M}}) < s$. For the probability measure μ it holds that $\mu(Y_1 \cup Y_2) = \mu(Y_1) \cup \mu(Y_2) - \mu(Y_1 \cap Y_2)$, for any sets $Y_1, Y_2 \in H$. Thus, $\mu([\alpha]_{\mathcal{M}} \cup [\beta]_{\mathcal{M}}) = \mu([\alpha]_{\mathcal{M}}) + \mu([\beta]_{\mathcal{M}}) - \mu([\alpha]_{\mathcal{M}} \cap [\beta]_{\mathcal{M}})$. Since $\mu([\alpha]_{\mathcal{M}} \cup [\beta]_{\mathcal{M}}) \geq 0$ by the definition of finitely additive probability measure, we conclude $\mu([\alpha]_{\mathcal{M}} \cup [\beta]_{\mathcal{M}}) \leq \mu([\alpha]_{\mathcal{M}}) + \mu([\beta]_{\mathcal{M}}) < r + s$. Hence, $\mathcal{M} \models P_{<r+s}(\alpha \vee \beta)$.

Let us consider an instance of the axiom scheme (11). We prove that if $P_{\geq 1}(\alpha \Rightarrow \beta)$ and $P_{\geq s}\alpha$ hold in some PCL_{Meas}-model $\mathcal{M}$, then $P_{\geq s}\beta$ also hold in the model. From $\mathcal{M} \models P_{\geq 1}(\alpha \Rightarrow \beta)$, we get $\mu([\alpha \Rightarrow \beta]_{\mathcal{M}}) = 1$, i.e. $[\alpha \Rightarrow \beta]_{\mathcal{M}} = W$. It follows that if a world $w \in W$ satisfies the formula α, then it also satisfies the formula β. Thus, $[\alpha]_{\mathcal{M}} \subseteq [\beta]_{\mathcal{M}}$. Now, by the definition of finitely additive probability measure, we have $\mu([\beta]_{\mathcal{M}}) \geq \mu([\alpha]_{\mathcal{M}})$. So, if $\mathcal{M} \models P_{\geq s}\alpha$, then we have $\mu([\alpha]_{\mathcal{M}}) \geq s$. Consequently, $\mu([\beta]_{\mathcal{M}}) \geq s$, i.e. $\mathcal{M} \models P_{\geq s}\beta$.

The satisfiability of the axiom schemes (12) – (14) where atoms are probabilistic formulas follows directly from the definition of satisfiability of classical connectives.

Next, we consider the inference rules and prove that they preserve validity. The inference rule (R1) is Modus Ponens. The satisfiability of logical connectives is defined as in the classical propositional logic and an implication $A \Rightarrow B$ does not hold only if A holds and B does not. So, if A and $A \Rightarrow B$ hold, then B has to hold as well.

Let us consider the inference rule (R2). If $\mathcal{M} \models \alpha$, then $w \models \alpha$ for each possible world w in $\mathcal{M}$ and $[\alpha]_{\mathcal{M}} = W$. Thus, $\mathcal{M} \models P_{\geq 1}\alpha$.

Finally, we consider the infinitary rule (R3). Let us assume that $\mathcal{M}$ is a PCL_{Meas}-model such that $\mathcal{M} \models \varphi \Rightarrow P_{\geq s - \frac{1}{k}}\alpha$ for all $k \geq \frac{1}{s}$. We prove that $\mathcal{M} \models \varphi$ implies $\mathcal{M} \models P_{\geq s}\alpha$. If $\mathcal{M} \models \varphi$, then $\mathcal{M} \models P_{\geq s - \frac{1}{k}}\alpha$ for all $k \geq \frac{1}{s}$, i.e. $\mu([\alpha]_{\mathcal{M}}) \geq s - \frac{1}{k}$. We prove that it is not possible that $\mu([\alpha]_{\mathcal{M}}) < s$. From $\mu([\alpha]_{\mathcal{M}}) < s$, it follows that $s - \mu([\alpha]_{\mathcal{M}}) > 0$. Due to the Archimedean property of real numbers, we have that there exists $n \in \mathbb{N}$ such that $\frac{1}{n} < s - \mu([\alpha]_{\mathcal{M}})$.

Then $\mu([\alpha]_{\mathcal{M}}) < s - \frac{1}{n}$. In addition, from $\mu([\alpha]_{\mathcal{M}}) \geq 0$, we get $n \geq \frac{1}{s}$. This contradicts the assumption that $\mu([\alpha]_{\mathcal{M}}) \geq s - \frac{1}{k}$ for all $k \geq \frac{1}{s}$. Hence, $\mu([\alpha]_{\mathcal{M}}) \geq s$ holds, i.e. $\mathcal{M} \models P_{\geq s}\alpha$.

□

The strong completeness proof for the logic PCL is more involved. Following the approach used for the logic LPP_2, we adapt the Henkin-style completeness method. The proof of strong completeness comprises the following steps:

- the proof of Deduction theorem,
- the proof that every consistent set of PCL-formulas can be extended to a maximal consistent set,
- the construction of a canonical model using maximal consistent set,
- the proof that a canonical model is a PCL-model,
- the proof that every consistent set is satisfiable,
- the proof of strong completeness.

The proof of Deduction theorem is given in Section 5.4. We present some auxiliary results about properties of consistent sets.

Lemma 5.27. *Let α be a basic formula and let $s, r \in \mathsf{S}$ be such that $s \geq r$. Then the formulas*

$$P_{\geq s}\alpha \Rightarrow P_{\geq r}\alpha \text{ and } P_{\leq r}\alpha \Rightarrow P_{\leq s}\alpha$$

are theorems in the logic PCL.

Proof. We first prove that the formula $P_{\geq s}\alpha \Rightarrow P_{\geq r}\alpha$ can be derived using axiom schemes and inference rules of Figure 5.2. We distinguish two cases.

- If $s = r$, then $P_{\geq s}\alpha \Rightarrow P_{\geq s}\alpha$ is an instance of classical tautology $\varphi \Rightarrow \varphi$, which is derivable using axiom schemes (12) and (13) and the inference rule (R1) of Figure 5.2.
- Let us assume that $s > r$. Then instances of axiom schemes (7) and (8) are formulas $P_{\leq r}\alpha \Rightarrow P_{<s}\alpha$ and $P_{<r}\alpha \Rightarrow P_{\leq r}\alpha$, respectively. Using the definition of probability operators and logical connectives introduced in Section 5.1.1 we obtain that

$$P_{\leq r}\alpha \Rightarrow P_{<s}\alpha \text{ is equivalent to } P_{\geq s}\alpha \Rightarrow P_{>r}\alpha, \text{ and}$$

$$P_{<r}\alpha \Rightarrow P_{\leq r}\alpha \text{ is equivalent to } P_{>r}\alpha \Rightarrow P_{\geq r}\alpha.$$

From $P_{\geq s}\alpha \Rightarrow P_{>r}\alpha$, $P_{>r}\alpha \Rightarrow P_{\geq r}\alpha$ and the transitivity of the implication in the classical propositional logic, we obtain $P_{\geq s}\alpha \Rightarrow P_{\geq r}\alpha$.

The proof that $P_{\leq r}\alpha \Rightarrow P_{\leq s}\alpha$ is a theorem proceeds similarly. □

Lemma 5.28. *Let T be a consistent set of PCL-formulas.*

(1) *For any formula $\varphi \in \mathsf{For_P}$, either $T \cup \{\varphi\}$ is consistent or $T \cup \{\neg\varphi\}$ is consistent.*

(2) *If $\neg(\varphi \Rightarrow P_{\geq s}\alpha) \in T$, then there is some $n > \dfrac{1}{s}$ such that $T \cup \{\varphi \Rightarrow \neg P_{\geq s-\frac{1}{n}}\alpha\}$ is consistent.*

Proof. (1) Let T be a consistent set of PCL-formulas and $\varphi \in \mathsf{For_P}$. If both $T \cup \{\varphi\}$ and $T \cup \{\neg\varphi\}$ are inconsistent, then we have $T \cup \{\varphi\} \vdash \bot$ and $T \cup \{\neg\varphi\} \vdash \bot$. By Deduction theorem, we obtain $T \vdash \varphi \Rightarrow \bot$ and $T \vdash \neg\varphi \Rightarrow \bot$, that is $T \vdash \neg\varphi$ and $T \vdash \varphi$. We further derive $T \vdash \neg\varphi \wedge \varphi$, that is $T \vdash \bot$. However, this contradicts the assumption that T is consistent.

(2) Let T be a consistent set and $\neg(\varphi \Rightarrow P_{\geq s}\alpha) \in T$. Suppose that for every $n > \dfrac{1}{s}$, $T \cup \{\varphi \Rightarrow \neg P_{\geq s-\frac{1}{n}}\alpha\}$ is inconsistent. Then $T \cup \{\varphi \Rightarrow \neg P_{\geq s-\frac{1}{n}}\alpha\} \vdash \bot$. We obtain the following

1. $T \vdash (\varphi \Rightarrow \neg P_{\geq s-\frac{1}{n}}\alpha) \Rightarrow \bot$, by Deduction theorem,
2. $T \vdash \neg(\neg\varphi \vee \neg P_{\geq s-\frac{1}{n}}\alpha) \vee \bot$, by propositional reasoning,
3. $T \vdash (\varphi \wedge P_{\geq s-\frac{1}{n}}\alpha) \vee (\varphi \wedge \neg\varphi)$, by propositional reasoning,
4. $T \vdash ((\varphi \wedge P_{\geq s-\frac{1}{n}}\alpha) \vee (\varphi \wedge \neg\varphi)) \Rightarrow (\varphi \wedge (P_{\geq s-\frac{1}{n}}\alpha \vee \neg\varphi))$, the distributive law,
5. $T \vdash \varphi \wedge (P_{\geq s-\frac{1}{n}}\alpha \vee \neg\varphi)$, from (3) and (4) by (R1),
6. $T \vdash \neg\varphi \vee P_{\geq s-\frac{1}{n}}\alpha$, by propositional reasoning,
7. $T \vdash \varphi \Rightarrow P_{\geq s-\frac{1}{n}}\alpha$, by propositional reasoning.

We have used the distributive law and the property of the classical propositional logic: if $T \vdash \varphi \wedge \psi$, then $T \vdash \varphi$.

We have derived $T \vdash \varphi \Rightarrow P_{\geq s-\frac{1}{n}}\alpha$ for every $n > \dfrac{1}{s}$. If $n = \dfrac{1}{s}$, then it also holds that $T \vdash \varphi \Rightarrow P_{\geq s-\frac{1}{n}}\alpha$ for every $n > \dfrac{1}{s}$. By the inference rule (R3) in Figure 5.2 we obtain $T \vdash \varphi \Rightarrow P_{\geq s}\alpha$. However, this contradicts the assumption that T is consistent set and that $\neg(\varphi \Rightarrow P_{\geq s}\alpha) \in T$. □

Lemma 5.29. *Let T be a maximal consistent set of formulas.*

(1) *For $\psi \in \mathsf{For_P}$, if $T \vdash \psi$, then $\psi \in T$.*

(2) *For any formula $\alpha \in \mathsf{For_B}$, if $t = sup\{s \mid P_{\geq s}\alpha \in T\}$ and $t \in \mathsf{S}$, then $P_{\geq t}\alpha \in T$.*

(3) *For all formulas $\varphi, \psi \in \mathsf{For_P}$,*

$$\varphi \vee \psi \in T \text{ if and only if } \varphi \in T \text{ or } \psi \in T.$$

(4) *For all formulas $A, B \in \mathsf{For}_{PCL}$ such that either $A, B \in \mathsf{For_B}$ or $A, B \in \mathsf{For_P}$,*

$$A \wedge B \in T \text{ if and only if } A \in T \text{ and } B \in T.$$

(5) *For all formulas $A, B \in \mathsf{For}_{PCL}$ such that either $A, B \in \mathsf{For_B}$ or $A, B \in \mathsf{For_P}$,*

$$\text{if } A \in T \text{ and } A \Rightarrow B \in T, \text{ then } B \in T.$$

Proof. (1) Let T be a maximal consistent set and ψ a PCL-formula such that $T \vdash \psi$. By Definition 5.23 we have that either $\psi \in T$ or $\neg\psi \in T$. If $\neg\psi \in T$, then $T \vdash \neg\psi$. Nevertheless, this contradicts the assumption that T is consistent and $T \vdash \psi$. Thus, $\psi \in T$. This proves that T is deductively closed set.

(2) Let us assume that T is a maximal consistent set, $\alpha \in \mathsf{For_B}$, $t = sup\{s \mid P_{\geq s}\alpha \in T\}$ and $t \in \mathsf{S}$. By the monotonicity of the measure proved in Lemma 5.27, we have that for every $s \in \mathsf{S}$ such that $s < t$ it holds that $T \vdash P_{\geq s}\alpha$. Then we obtain $T \vdash P_{\geq t}\alpha$ by the inference rule (R3) in Figure 5.2 . Since T is a maximal consistent set, $P_{\geq t}\alpha \in T$ follows from the first statement of this lemma.

Proofs $(3) - (5)$ follow from the deductive closeness of the maximal consistent set T. □

We prove Lindenbaum's theorem, which states that every consistent set can be extended to a maximal consistent set.

Theorem 5.30 (Lindenbaum's theorem)**.** *Every consistent set of PCL-formulas can be extended to a maximal consistent set.*

Proof. Let T be a consistent set. Let us denote by $\mathsf{Cn_B}(T)$ the consistent set of formulas that contains all basic formulas derivable from T, i.e.

$$\mathsf{Cn_B}(T) = \{\alpha \mid \alpha \in \mathsf{For_B} \text{ and } T \vdash \alpha\}.$$

We take $\varphi_0, \varphi_1, \ldots$ to be an enumeration of all probabilistic formulas, that is formulas from $\mathsf{For_P}$, and define a sequence of sets T_i, $i = 0, 1, 2, \ldots$ as follows:

(1) $T_0 = T \cup \mathsf{Cn_B}(T) \cup \{P_{\geq 1}\alpha \mid \alpha \in \mathsf{Cn_B}(T)\}$,

(2) for every $i \geq 0$,

  (a) if $T_i \cup \{\varphi_i\}$ is consistent, then $T_{i+1} = T_i \cup \{\varphi_i\}$, otherwise

  (b) if φ_i is of the form $\psi \Rightarrow P_{\geq s}\beta$, then $T_{i+1} = T_i \cup \{\neg\varphi_i, \psi \Rightarrow \neg P_{\geq s-\frac{1}{n}}\beta\}$, for some positive integer n, so that T_{i+1} is consistent, otherwise,

  (c) $T_{i+1} = T_i \cup \{\neg\varphi_i\}$.

We define the set $T^\star = \bigcup_{i=0}^{\infty} T_i$.

We prove that the set $T^\star$ is a maximal consistent set that includes T. By the definition of the sequence of sets, we have that each set T_i, $i = 0, 1, 2, \ldots$ is consistent. The existence of the natural number n from the step $2(b)$ is ensured by Lemma 5.28.

The set $T^\star$ is maximal by the steps (1) and (2) in its definition. In order to prove that $T^\star$ is consistent, it is enough to prove that it is a deductively closed set which does not contain all formulas.

From the definition of the set $T^\star$, it follows that it does not contain all formulas. For a formula $\alpha \in \mathsf{For_B}$, the formulas α and $\neg\alpha$ cannot both belong to T_0. For a probabilistic formula $\varphi \in \mathsf{For_P}$, let $\varphi = \varphi_i$ and $\neg\varphi = \varphi_j$. Since the set $T_{\max\{i,j\}+1}$ is consistent, the set $T^\star$ does not contain both φ and $\neg\varphi$.

It remains to prove that $T^\star$ is deductively closed. For a basic formula $\alpha \in \mathsf{For_B}$, if $T \vdash \alpha$, then by the construction of T_0, $\alpha \in T^\star$ and $P_{\geq 1}\alpha \in T^\star$.

From the definition of a proof and the set $T^\star$, it follows that if $\varphi = \varphi_j$ and $T_i \vdash \varphi$, then $\varphi \in T^\star$ due to the consistency of $T_{\max\{i,j\}+1}$. Using the induction on the length of a proof, we prove that for a probabilistic formula $\varphi \in \mathsf{For_P}$,

$$\text{if } T^\star \vdash \varphi, \text{then } \varphi \in T^\star.$$

Let the sequence $\varphi_1, \varphi_2, \ldots, \varphi$ be the proof of φ from $T^\star$. The sequence can be countably infinite. We prove that for each i, if the formula φ_i is obtained by applying an inference rule to premises that belong to the set $T^\star$, then the formula φ also belongs to $T^\star$. In the case that a finitary rule is applied, there exists a set T_j such that all the premises belong to T_j and $T_j \vdash \varphi_i$. Thus, $\varphi_i \in T^\star$. If the infinitary rule is applied, then the formula φ_i is of the form $\psi \Rightarrow P_{\geq s}\alpha$ and it is obtained from the set of premises $\{\varphi_i^k = \psi \Rightarrow P_{\geq s_k}\alpha \mid s_k \in \mathsf{S}\}$ such that $T^\star \vdash \varphi_i^k$. By the induction hypothesis, we get $\psi \Rightarrow P_{\geq s_k}\alpha \in T^\star$, for every k. If we assume that $\varphi_i = \psi \Rightarrow P_{\geq s}\alpha \notin T^\star$, then there exist l and j such that $\neg(\psi \Rightarrow P_{\geq s}\alpha)$, $\psi \Rightarrow \neg P_{\geq s-\frac{1}{l}}\alpha \in T_j$ by the step $(2)(b)$ in the definition of $T^\star$. Now, for some $j' \geq j$, we have the following:

- $\psi \wedge \neg P_{\geq s}\alpha \in T_{j'}$, using the definition of the propositional connectives,
- $\psi \in T_{j'}$, from $\psi \wedge \neg P_{\geq s}\alpha \in T_{j'}$ by the propositional reasoning,
- $\neg P_{\geq s-\frac{1}{l}}\alpha \in T_{j'}$, follows from $\psi \Rightarrow \neg P_{\geq s-\frac{1}{l}}\alpha \in T_j$ and $\psi \in T_{j'}$
- $P_{\geq s-\frac{1}{l}}\alpha \in T_{j'}$, follows from $\psi \Rightarrow P_{\geq s-\frac{1}{l}}\alpha \in T^\star$ and $\psi \in T_{j'}$.

The last two conclusions contradict each other, so it is not possible that $\varphi_i = \psi \Rightarrow P_{\geq s}\alpha \notin T^\star$. Thus, $\varphi_i = \psi \Rightarrow P_{\geq s}\alpha \in T^\star$.

Since $T^\star$ is a deductively closed set that does not contain all formulas, we conclude it is consistent.

□

Definition 5.31. *Let T be a consistent set, $T^\star$ a maximal consistent set introduced in the proof of Theorem 5.30 and $Models_{LCL}(T)$ the set of all LCL-models $\mathcal{M}_{\rho_i} = \langle \mathcal{M}_i, \rho_i \rangle$ that satisfy the set $\mathsf{Cn}_\mathsf{B}(T)$, where $\mathcal{M}_i = \langle D_i, \{A_i^\sigma\}_\sigma, \cdot_i, \boldsymbol{s}_i, \boldsymbol{k}_i, \boldsymbol{i}_i \rangle$. A canonical model $\mathcal{M}_{T^\star}$ is a tuple*

$$\mathcal{M}_{T^\star} = (W, \{D_i\}, \{A_i^\sigma\}, \{\cdot_i\}, \{\boldsymbol{s}_i\}, \{\boldsymbol{k}_i\}, \{\boldsymbol{i}_i\}, H, \mu, \rho)$$

such that

- *W is a set of possible worlds, one for each LCL-model that satisfies $\mathsf{Cn}_\mathsf{B}(T)$; $W = \{w_i \mid i \in I\}$ where I is the cardinality of the set $Models_{LCL}(T)$.*
- *$\{D_i\} = \{D_i\}_{i \in I}$ is a family of sets, where D_i is the domain of LCL-model $\mathcal{M}_i$,*
- *$\{A_i^\sigma\} = \{A_i^\sigma\}_{i \in I, \sigma \in \mathsf{Types}_\to}$ is a family of sets, where $\{A_i^\sigma\}_\sigma$ is a family given in the LCL-model $\mathcal{M}_i$,*
- *$\{\cdot_i\} = \{\cdot_i\}_{i \in I}$ is a family of binary operations given in the LCL-model $\mathcal{M}_i$,*
- *$\{\boldsymbol{s}_i\} = \{\boldsymbol{s}_i\}_{i \in I}, \{\boldsymbol{k}_i\} = \{\boldsymbol{k}_i\}_{i \in I}, \{\boldsymbol{i}_i\} = \{\boldsymbol{i}_i\}_{i \in I}$ are families of elements of the domains such that $\boldsymbol{s}_i, \boldsymbol{k}_i$ and $\boldsymbol{i}_i$ are the elements given in the LCL-model $\mathcal{M}_i$,*
- *$[\alpha] = \{w_i \mid \mathcal{M}_i \models \alpha\}$ and $H = \{[\alpha] \mid \alpha \in \mathsf{For}_\mathsf{B}\}$,*
- *$\mu([\alpha]) = \sup\{s \mid P_{\geq s}\alpha \in T^\star\}$,*
- *ρ is a two-valued valuation defined by $\rho(w_i, x) = \rho_i(x)$, where ρ_i is a valuation of the LCL-model $\mathcal{M}_{\rho_i}$.*

Lemma 5.32. *Let $\mathcal{M}_{T^\star}$ be a canonical model and $\alpha, \beta \in \mathsf{For}_\mathsf{B}$. Then the following claims hold*

(1) *H is an algebra of subsets of W,*

(2) *if $[\alpha] = [\beta]$, then $\mu([\alpha]) = \mu([\beta])$,*

(3) *$\mu([\alpha]) \geq 0$,*

(4) *$\mu(W) = 1$ and $\mu(\emptyset) = 0$,*

(5) *$\mu([\alpha]) = 1 - \mu([\neg\alpha])$,*

(6) *$\mu([\alpha] \cup [\beta]) = \mu([\alpha]) + \mu([\beta])$, for all disjoint $[\alpha]$ and $[\beta]$.*

Proof. (1) For basic formulas $\alpha, \alpha_1, \alpha_2, \ldots, \alpha_n$ it holds that

- $W = [\alpha \vee \neg\alpha] \in H$,
- if $[\alpha] \in H$, then its complement $[\neg\alpha] \in H$,
- if $[\alpha_1], [\alpha_2], \ldots, [\alpha_n] \in H$, then

$$[\alpha_1] \cup [\alpha_2] \cup \ldots \cup [\alpha_n] = [\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n] \in H$$

So, we conclude that H is an algebra of subsets of W.

(2) We prove that $[\alpha] \subseteq [\beta]$ implies $\mu([\alpha]) \leq \mu([\beta])$. From $[\alpha] \subseteq [\beta]$, we conclude that every LCL-model of α is also a model of β, thus the formula $\alpha \Rightarrow \beta$ is true in every LCL-model. From the strong completeness of the logic LCL, proved in Theorem 4.34, it follows that $\alpha \Rightarrow \beta$ is a theorem and $\alpha \Rightarrow \beta \in \mathsf{Cn}_\mathsf{B}(T)$. Then we obtain that $P_{\geq 1}(\alpha \Rightarrow \beta) \in T^\star$ by the definition of the set $T^\star$. The latter and the fact that $P_{\geq 1}(\alpha \Rightarrow \beta) \Rightarrow (P_{\geq s}\alpha \Rightarrow P_{\geq s}\beta)$ is an instance of axiom scheme (11) imply that $P_{\geq s}\alpha \Rightarrow P_{\geq s}\beta \in T^\star$ for every $s \in S$. Consequently, $\mu([\alpha]) \leq \mu([\beta])$.

(3) $\mu([\alpha]) = 0$ follows from the fact that $P_{\geq 0}\alpha$ is an axiom and it belongs to $T^\star$.

(4) For any $\alpha \in \mathsf{For}_\mathsf{B}$, the formula $\alpha \vee \neg\alpha$ is a theorem, thus it belongs to $\mathsf{Cn}_\mathsf{B}(T)$ and $P_{\geq 1}(\alpha \vee \neg\alpha) \in T^\star$. Therefore, we have $W = [\alpha \vee \neg\alpha]$ and $\mu(W) = 1$. Using the definition of probability operators, we get

$$P_{\geq 1}(\alpha \vee \neg\alpha) = P_{\geq 1-0}(\alpha \vee \neg\alpha) = P_{\leq 0}\neg(\alpha \vee \neg\alpha) = \neg P_{>0}(\alpha \wedge \neg\alpha).$$

From the latter and Theorem 5.22, we conclude that

$$\sup\{t \mid P_{\geq t}(\alpha \wedge \neg\alpha)\} = 0$$

and $\mu(\emptyset) = 0$.

(5) Let $\mu([\alpha]) = \sup\{s \mid P_{\geq s}\alpha \in T^\star\} = r$. If $r = 1$, then we obtain $P_{\geq 1}\alpha \in T^\star$ by Lemma 5.29(2). Since $P_{\geq 1}\alpha = \neg P_{>0}\neg\alpha$, it follows that $\neg P_{>0}\neg\alpha \in T^\star$. As it was discussed earlier, the formula $P_{\geq t}\alpha \Rightarrow P_{>s}\alpha$ for $t > s$ is a theorem. So, we conclude $\mu([\neg\alpha]) = 0$ and $\mu([\alpha]) = 1 - \mu([\neg\alpha])$. On the other hand, if $r < 1$, then for each rational number $r' \in (r, 1]$, $\neg P_{\geq r'}\alpha = P_{<r'}\alpha \in T^\star$. The axiom scheme (8) ensures that $P_{\leq r'}\alpha, P_{\geq 1-r'}\neg\alpha \in T^\star$.

So, it has to hold that $r_1 = \sup\{s \mid P_{\geq s}\neg\alpha \in T^\star\} \geq 1-r$. Otherwise, for $r_2 \in (r_1, 1-r)$ it holds that $P_{\geq r_2}\neg\alpha \notin T^\star$. From $r_2 < 1-r$ it follows that $1 - r_2 > r$, so we have $P_{\geq r_2}\neg\alpha \in T^\star$. This contradicts $P_{\geq r_2}\neg\alpha \notin T^\star$.

If we assume that $\sup\{s \mid P_{\geq s}\neg\alpha \in T^\star\} > 1 - r$, then there exists a rational number $r'' \in [0, r)$ such that $P_{\geq 1-r''}\neg\alpha \in T^\star$ and $\neg P_{>r''}\alpha \in T^\star$. Since $r > r''$, we have that $P_{\geq r}\alpha \Rightarrow P_{>r''} \in T^\star$ by Proposition 5.22. Hence, $P_{>r''}\alpha \in T^\star$. This contradicts $\neg P_{>r''}\alpha \in T^\star$.

Thus,

$$\sup\{s \mid P_{\geq s}\neg\alpha \in T^\star\} = 1 - \sup\{s \mid P_{\geq s}\alpha \in T^\star\},$$

i.e., $\mu([\alpha]) = 1 - \mu([\neg\alpha])$.

(6) Let us assume that $[\alpha] \cap [\beta] = \emptyset$, $\mu([\alpha]) = r$ and $\mu([\beta]) = s$. Since $[\beta] \subset [\neg\alpha]$, we obtain $r + s \leq r + (1 - r) = 1$ using the previously proved properties in the lemma. If $r > 0$ and $s > 0$, then for every rational number $r' \in [0, r)$ and for every rational number $s' \in [0, s)$, we have that $P_{\geq r'}\alpha, P_{\geq s'}\beta \in T^\star$ due to the properties of the supremum. Then, using the axiom scheme (9), we derive $P_{\geq r'+s'}(\alpha \vee \beta) \in T^\star$. The latter implies $r + s \leq t_0 = \sup\{t \mid P_{\geq t}(\alpha \vee \beta) \in T^\star)\}$. In the case that $r + s = 1$, the statement holds. Let us consider the case when $r + s < 1$. If $r + s < t_0$, then for every rational number $t' \in (r + s, t_0)$ it holds that $P_{\geq t'}(\alpha \vee \beta) \in T^\star$. Further, there exist rational numbers $r'' > r$ and $s'' > s$ such that:

$$\begin{aligned}
&\neg P_{\geq r''}\alpha \in T^\star, \text{ i.e. } P_{<r''}\alpha \in T^\star,\\
&\neg P_{\geq s''}\alpha \in T^\star, \text{ i.e. } P_{<s''}\alpha \in T^\star,\\
&\text{and } r'' + s'' = t' \leq 1.
\end{aligned}$$

The axiom scheme (8) ensures that $P_{\leq r''}\alpha \in T^\star$. Using the axiom scheme (10), we derive

$$P_{<r''+s''}(\alpha \vee \beta) \in T^\star, \quad \text{i.e. } \neg P_{\geq r''+s''}(\alpha \vee \beta) \in T^\star,$$

and

$$\neg P_{\geq t'}(\alpha \vee \beta) \in T^\star.$$

This contradicts $P_{\geq t'}(\alpha \vee \beta) \in T^\star$. Hence, $r + s = t_0$ and we conclude that $\mu([\alpha] \cup [\beta]) = \mu([\alpha]) + \mu([\beta])$. Finally, in the case that $r = 0$ or $s = 0$, we can reason as above, where $r' = 0$ or $s' = 0$.

□

Lemma 5.33. *A canonical model $\mathcal{M}_{T^\star}$ is a PCL-model.*

Proof. We prove that the tuple $\mathcal{M}_{T^\star}$ given in Definition 5.31 satisfies the conditions of Definition 5.16.

Since $\mathsf{Cn}_\mathsf{B}(T)$ is a consistent set of LCL-formulas, there is an LCL-model that satisfies the set $\mathsf{Cn}_\mathsf{B}(T)$, so the set $Models_{LCL}(T)$ is not empty. Consequently, the set W is not empty. For $\mathcal{M}_{\rho_i} = \langle \mathcal{M}_i, \rho_i \rangle \in Models_{LCL}(T)$, we have that D_i, $\{A_i^\sigma\}_\sigma$, $\cdot_i$, $\boldsymbol{s}_i$, $\boldsymbol{k}_i$, $\boldsymbol{i}_i$ satisfy the conditions of Definition 5.16, since they form an LCL-model $\mathcal{M}_i$. In Lemma 5.32(1) we have proved that H is an algebra of subsets of W. By Lemma 5.32, we conclude that the function μ is a finitely additive probability measure. Finally, the function ρ defined by $\rho(w_i, x) = \rho_i(x)$ is a two-valued valuation $\rho : W \times V \to \bigcup_{w_i \in W} D_{w_i}$ such that $\rho(w_i, \cdot) = \rho_i : V \to D_{w_i}$. This concludes the proof that $\mathcal{M}_{T^\star}$ is a PCL-model. □

Proposition 5.34. *Let $\mathcal{M}_{T^\star}$ be a canonical model. For every $A \in \mathsf{For}_{PCL}$, $\mathcal{M}_{T^\star} \models A$ if and only if $A \in T^\star$.*

Proof. The proof proceeds by induction on the structure of formula A.

Let A be a basic formula α. If $\alpha \in T^\star$, then by the definition of $T^\star$ we have that $\alpha \in \mathsf{Cn}_\mathsf{B}(T)$. Thus, for every $\mathcal{M}_i \in Models_{LCL}(T)$, it holds that $\mathcal{M}_i \models \alpha$. Each world w_i of $\mathcal{M}_{T^\star}$ corresponds to one LCL-model $\mathcal{M}_i$ and it holds that $w_i \models \alpha$ if and only if $\mathcal{M}_i \models \alpha$. Hence, we conclude $w_i \models \alpha$, for each $i \in I$ and it follows that $\mathcal{M}_{T^\star} \models \alpha$. If $\mathcal{M}_{T^\star} \models \alpha$, then $w_i \models \alpha$, for each $i \in I$, that is each model $\mathcal{M}_i \in Models_{LCL}(T)$ of the set $\mathsf{Cn}_\mathsf{B}(T)$ is also the model of α. Hence, $\mathsf{Cn}_\mathsf{B}(T) \models \alpha$ in the logic LCL. By the strong completeness of the logic LCL proved in Theorem 4.34 (Chapter 4), we obtain $\mathsf{Cn}_\mathsf{B}(T) \vdash \alpha$. Consequently, $\alpha \in T^\star$ due to the definition of $\mathsf{Cn}_\mathsf{B}(T)$ and $T^\star$.

Next, let us assume that A is a basic probabilistic formula $P_{\geq s}\alpha$. If $P_{\geq s}\alpha \in T^\star$, then $\mu([\alpha]) = \sup\{t \mid P_{\geq t}\alpha \in T^\star\} \geq s$. So, we conclude $\mathcal{M}_{T^\star} \models P_{\geq s}\alpha$ by Definition 5.20. On the other hand, suppose that $\mathcal{M}_{T^\star} \models P_{\geq s}\alpha$, i.e. $\mu([\alpha]) = \sup\{t \mid P_{\geq t} \in T^\star\} \geq s$. If $\mu([\alpha]) > s$, then $P_{\geq s}\alpha \in T^\star$ by the definition of supremum and Lemma 5.27. If $\mu([\alpha]) = s$, then $P_{\geq s}\alpha \in T^\star$ by Lemma 5.29(2).

In the case when A is a negation $\neg\varphi$, by Definition 5.20 and the induction

hypothesis we obtain

$$\begin{aligned}\mathcal{M}_{T^\star} \models \neg\varphi \text{ if and only if } \mathcal{M}_{T^\star} &\not\models \varphi\\ \text{if and only if } \varphi &\notin T^\star\\ \text{if and only if } \varphi &\in T^\star.\end{aligned}$$

Similarly, if A is a conjunction $\varphi \wedge \psi$, then by Definition 5.20 and the induction hypothesis we conclude

$$\begin{aligned}\mathcal{M}_{T^\star} \models \varphi \wedge \psi \text{ if and only if } \mathcal{M}_{T^\star} &\models \varphi \text{ and } \mathcal{M}_{T^\star} \models \psi\\ \text{if and only if } \varphi &\in T^\star \text{ and } \psi \in T^\star\\ \text{if and only if } \varphi \wedge \psi &\in T^\star.\end{aligned}$$

□

Theorem 5.35. *Every consistent set T of PCL-formulas is satisfiable.*

Proof. Let T be a consistent set. By Theorem 5.30, it can be extended to a maximal consistent $T^\star$. Let $\mathcal{M}_{T^\star}$ be a canonical model introduced in Definition 5.31. Since $T \subseteq T^\star$ by the definition of $T^\star$ and the model $\mathcal{M}_{T^\star}$ satisfies all formulas from the set $T^\star$ by Proposition 5.34, we obtain that the model $\mathcal{M}_{T^\star}$ satisfies all formulas from the set T as well, i.e. $\mathcal{M}_{T^\star} \models T$. Thus, T is satisfiable. □

Theorem 5.36 (Strong completeness)**.** *If $T \models A$, then $T \vdash A$.*

Proof. Let T be a set of formulas and A a formula such that $T \models A$. From $T \models A$ it follows that the set $T \cup \{\neg A\}$ is not satisfiable. Then we conclude that $T \cup \{\neg A\}$ is inconsistent by Theorem 5.35 , that is $T \cup \{\neg A\} \vdash \bot$. By Deduction theorem, we derive $T \vdash \neg A \Rightarrow \bot$, which is equivalent to $T \vdash A$. □

5.6 Concluding remarks

In this chapter, we have introduced the logic PCL, which is a probabilistic extension of the logic of combinatory logic.

Following the approach used for the logic LPP_2 and some other probability logics, e.g. first-order probability logic and intuitionistic probability logic ([132]), the language of the logic PCL is layered into two sets: basic formulas, which are actually LCL-formulas, and probabilistic formulas. The probabilistic formulas are obtained by applying probability operators of the form $P_{\geq s}$ to the basic formulas. The proposed semantics of the logic PCL is based on

the possible world approach. We have proved that the compactness theorem does not hold for the logic PCL, i.e. there is an infinite set X that is not satisafiable, although each finite subset of X is satisfiable. For this reason, a finite axiomatization of the logic PCL cannot be sound and complete with respect to the proposed semantics. Thus, we give an infinitary axiomatization of the logic PCL. The main results of the chapter are the soundness and strong completeness of the given axiomatization with respect to the proposed semantics.

First, we have proved the soundness of the axiomatization of PCL, that is we proved that every instance of an axiom scheme holds in every PCL-model and that inference rules preserve validity. The proof of strong completeness is more involved. Herein, we adapted the Henkin-style completeness method, which was also used for the logic LPP_2 ([128]). We proved that every consistent set can be extended to a maximal consistent set. The maximal consistent set is used for the definition of a canonical model that is a PCL model which satisfies exactly formulas from the maximal consistent set. Consequently, we obtained that every consistent set is satisfiable, which implies the strong completeness of the logic PCL.

At the end, we list some ideas for the future work.

- The presented logic is a formal model for reasoning about probabilities of simply typed combinatory terms. The next step is to adapt the approach for probabilistic reasoning about simply typed λ-calculus. The adaptation is not straightforward since it relies on the translations from combinatory logic to λ-calculus which are rather involved.

- Simplifying the semantics by using finite sets of probability values yields a compact logic. One line of research is to provide finite axiomatizations of those logics.

- The basic formulas in the logic PCL are obtained as the set of typed statements closed under classical propositional connectives. Another possibility is to have intuitionistic reasoning at the basic level.

- Also, intuitionistic reasoning about probabilities has been studied by Marković, Ognjanović and Rašković in [119]. Instead of considering classical propositional connectives in the definition of probabilistic formulas, one can consider the intuitionistic ones.

- The development of the first order extension of the logic PCL is a topic for the future work.

- Various type disciplines have been introduced for λ-calculus and combinatory logic such as polymorphic, intersection and higher-order types.

In the future, we plan to consider probabilistic extensions of different typed calculi.

Chapter 6

Conclusion

In this chapter, we summarise the contributions, present the related work and give some initial ideas for future work.

6.1 Summary of contributions

In this thesis, we have studied four different formal systems:

- the probabilistic λ-calculus with let-in operator,
- the full simply typed combinatory logic,
- the logic of combinatory logic and
- the probabilistic extension of the logic of combinatory logic.

We considered two approaches used to introduce probability into calculus:

1. to add a probabilistic choice to the language of untyped calculus as primitive in order to obtain probabilistic computation;
2. to extend the language of a typed calculus with probability operators in order to obtain a framework for probabilistic reasoning about the typed calculus in the style of probability logic.

In Chapter 2, we have studied the probabilistic λ-calculus $\Lambda_{\oplus,\mathsf{let}}$, that is the pure, untyped λ-calculus extended with two operators: the probabilistic choice operator $\oplus$ and let-in operator. The implemented evaluation strategy is a lazy call-by-name evaluation. The probabilistic choice operator $\oplus$ represents a fair choice, in the sense that the term $M \oplus N$ evaluates to M or N with the

equal probability. The let-in operator simulates the call-by-value evaluation in the call-by-name setting. A problem addressed in Chapter 2 is the program equivalence in $\Lambda_{\oplus,\mathsf{let}}$. The proof of context equivalence of two programs in higher-order languages is challenging, since it has to be shown that programs behave the same in any context and there are infinitely many contexts. For this reason, we aimed to find an effective method for checking program equivalence. Besides context equivalence, we have considered two other equivalence relations: bisimilarity and testing equivalence. First, we have presented the operational semantics of $\Lambda_{\oplus,\mathsf{let}}$ as a labelled Markov chain, then we have introduced the notion of probabilistic applicative bisimilarity for $\Lambda_{\oplus,\mathsf{let}}$. The first contribution of Chapter 2 is the proof that the probabilistic applicative bisimilarity is a congruence. We have proved this using Howe's technique ([84]). As a consequence, the probabilistic applicative bisimilarity is included in the context equivalence. In order to prove that the context equivalence is included in the probabilistic applicative bisimilarity, we have presented the testing language ([35]). The induced testing equivalence is proved to coincide with probabilistic applicative bisimilarity ([47]). The second main contribution of the chapter is the proof of the property that for every test there is a context such that the success probability of the test applied to a term is equal to the convergence probability of the context applied to the term. Consequently, the context equivalence is included in the testing equivalence. We have proved that all three equivalence relations coincide.

Chapter 3 has studied the full simply typed combinatory logic $CL^{\rightarrow,\times,+}$, that is, the simply typed combinatory logic extended with product types, sum types, empty type and unit type. We have presented the language of the full simply typed combinatory logic, its operational semantics, that is the equational theory induced by reduction relation and the type assignment system. The reduction relation in $CL^{\rightarrow,\times,+}$ depends on the typing relation. In order to ensure that equal terms inhabit the same type, the type assignment system is defined so that the typing relation depends on the reduction relation. Thus, the equational theory induced by reduction relation and the type assignment system are defined simultaneously. We have presented a Kripke-style semantics of $CL^{\rightarrow,\times,+}$ introduced in [94], which is inspired by the Kripke-style semantics of the simply typed λ-calculus introduced in [124]. The semantics is defined as an extensional Kripke applicative structure, which has special elements corresponding to combinators and is provided with a valuation of variables. The main contributions of the chapter are the soundness and completeness of $CL^{\rightarrow,\times,+}$ with respect to the presented semantics. First, the soundness of the equational theory and the type assignment system are proved. The proof method used for the soundness proof is mathematical induction. Second, the completeness of the equational theory and the type assignment system are

proved using the notion of a canonical model. The canonical model is based on a consistent basis and it is defined so that the interpretation of a term is the set of equivalence classes of terms typable in the basis. The completeness proofs follow directly from this property of the canonical model.

In Chapter 4, we have introduced the logic of combinatory logic LCL. The logic LCL is the propositional extension of the simply typed combinatory logic. It is a formal system for reasoning about typed statements. We have introduced the syntax, axiomatization and semantics of LCL. The language of LCL is obtained by closing the set of typed statements under classical propositional connectives. The axiomatization of LCL has arisen from the type assignment system of the simply typed combinatory logic and the axiomatic system of classical propositional logic. The semantics of LCL is based on the extensional applicative structures extended with special elements corresponding to primitive combinators. The main contributions of the chapter are the soundness and strong completeness of LCL with respect to the proposed semantics. First, we have proved that the equational theory of the simply typed combinatory logic is sound and complete with respect to the proposed semantics. Then the proof of soundness and strong completeness of the axiomatization of LCL with respect to the proposed semantics is given. Similarly to Chapter 3, the proof method used for proving soundness is the mathematical induction. In the proof of completeness of the equational theory we have constructed the model such that the interpretation of a term in the model is the equivalence class of the term with respect to the equational theory of the simply typed combinatory logic. As a consequence, we have obtained the completeness of the equational theory. The proof that the axiomatization of LCL is strongly complete is more involved. First, we prove that every consistent set can be extended to a maximal consistent set. Then we have defined an LCL-model such that it satisfies only formulas from the maximal consistent set. Using this model, we have proved that every consistent set is satisfiable. The strong completeness of the axiomatization follows directly from this result. Additionally, we have shown that the proposed semantics are novel semantics of the simply typed combinatory logic. More precisely, we proved that the simply typed combinatory logic containing the typing rule that ensures that equal terms inhabit the same type is sound and complete with respect to the proposed semantics. Thus, the logic of combinatory logic is a conservative extension of the simply typed combinatory logic.

Chapter 5 has introduced the logic PCL, the probabilistic extension of the logic of combinatory logic. Following the approach used for probabilistic extensions of different logics such as classical propositional logic, intuitionistic propositional logic, justification logic ([132]), we extend the logic LCL with probability operators of the form $P_{\geq s}$ with the intended meaning "probability

is at least s". We have introduced the syntax, semantics and axiomatization of PCL. The set of formulas is layered into two sets: basic formulas, i.e. LCL-formulas, and probabilistic formulas obtained by applying probability operators to basic formulas. The semantics of PCL is based on the possible world approach, where the set of possible worlds is equipped with a finitely additive probability measure. The axiomatization of PCL comprises the axioms and rules from the axiomatization of LCL and the axioms and rules from the axiomatization of probability logic. The axiomatization of PCL is infinitary, since it has one infinitary rule, i.e. a rule with a countable set of premises. This is due to the non-compactness of the logic PCL which we have proved. The main contributions of the chapter are the soundness and strong completeness of the given axiomatization with respect to the proposed semantics. The proof of soundness is straightforward. Similarly to the previous chapters, we have used mathematical induction. The proof of strong completeness is more involved. Following the approach used in Chapter 4, we first proved that every consistent set can be extended to a maximal consistent set. Then we constructed a PCL-model that satisfies only formulas from the maximal consistent set. We have used this model to prove that every consistent set is satisfiable and as a consequence we obtained strong completeness.

6.2 Related work

Probabilistic programming is a new programming paradigm which has proven to be extremely applicable and useful in various areas, such as robotics [163], machine learning [135] and natural language processing [117]. For this reason, questions about probabilistic program equivalence have been addressed in different settings ([35, 36, 43, 44, 47, 48]). The probabilistic λ-calculi with call-by-value and call-by-name passing policies have been investigated in [35] and [47], respectively. In [35], Crubillé and Dal Lago have proved the full abstraction, that is, they have proved that the bisimilarity and the context equivalence coincide in the call-by-value setting. In turn, in [47] Dal Lago, Sangiorgi and Alberti have proved that in the call-by-name setting bisimilarity implies context equivalence, but they do not coincide. Also, they have conjectured that adding a sequencing operator can recover the full abstraction. This conjecture is proved in Chapter 2. Both papers use the same approach as we have used in the thesis. They prove that the bisimilarity is a congruence using Howe's technique. Further, in [35] the authors introduced the testing language and they proved that the context equivalence implies bisimilarity using the testing equivalence, which coincides with bisimilarity. On the other hand, in [47] the authors gave examples of terms that are context equivalent in the call-by-name setting but not bisimilar, meaning that the context equivalence

does not imply the bisimilarity in this setting. A similar approach was used in [36], where Crubillé, Dal Lago, Sangiorgi and Vignudelli proved that similarity (resp. bisimilarity) is fully abstract with respect to the context preorder (resp. context equivalence) in call-by-value probabilistic λ-calculus endowed with Plotkin's parallel disjunction operator. The soundness of the applicative bisimilarity with respect to the context equivalence in linear λ-calculus extended with probabilistic binary choice and quantum data has been proved in [44] by Dal Lago and Rioli. In [48], Dal Lago, Gavazzo and Levy have studied Abramsky's applicative bisimilarity in call-by-value λ-calculi with algebraic effects. The authors have generalised Howe's technique in order to show that the applicative similarity is a precongruence. As a consequence, the applicative similarity is a sound proof technique for the contextual preorder. In [43], Dal Lago and Gavazzo have proved that the applicative bisimiliarity behaves well in a λ-calculus endowed with an operator performing sampling from continuous distributions. In [99], we have conjectured that the need for the let-in operator in lazy call-by-name λ-calculus was not due to the call-by-name evaluation strategy, but due to the lazyness of the calculus. This conjecture has been proved by Curzi and Pagani in [40].

Kripke-style semantics presented in Chapter 3 has been inspired by the Kripke-style semantics of the simply typed λ-calculus introduced in [124]. The semantics is defined as an extensional Kripke applicative structure with special elements that correspond to primitive combinators endowed with valuation of variables. In [124], the authors have considered the equations between terms of the same type that are described by an axiomatic system and they proved that the axiomatic system is sound and complete with respect to the proposed Kripke-style semantics. The semantics for the simply typed λ-calculus and semantics of the full simply typed λ-calculus introduced in [123] are not Kripke-style semantics, still they have certain similarities with the Kripke-style semantics presented in the thesis. They are similar in the sense that the applicative structure, the extensionality of applicative structure and the existence of elements called combinators are defined in a similar way. However, the soundness and completeness results are not presented in [123]. The results from [124] have been generalised to the second-order λ-calculus in [59], where the author considered not only equalities but also inequalities and obtained the soundness and completeness results for both inequalities and equalities. Another extension of the work introduced in [124] is the development of modified Kripke models for syntactic realizability and dependent type theory. The generalization of the interpretation of intuitionistic first-order logic in Kripke models to a dependent type theory is given in [5] with the aim to establish the coherence of interpretations of dependent type theory.

Chapter 4 has introduced the propositional extension of the simply typed

combinatory logic. Various extensions of combinatory logic have already been investigated with the goal of obtaining formalisms capable to express new features and paradigms. For example, in the calculus extended with new constructors such as pairs, records and variants ([137]), compound data structures can be built. Moreover, it is possible to organise data in a better way and to deal with heterogeneous collections of values. We have already discussed adding probabilistic choice operator and shifting to the probabilistic computation. However, in this approach the calculus is extended with new operators, whereas the logic introduced in Chapter 4 is obtained as the combination of different logical systems. This idea of combining different logical systems in order to capture reasoning about certain logical structures has been introduced by Scott in [158], where the typed system of combinators, including fixed-point combinator is extended with logical constants and connectives, and a deductive system for computable functions is developed. A similar approach was used by Beeson in [16] to define the λ-logic as the union of first-order logic and λ-calculus, and to develop a powerful tool for representing functions. The extensions of the theory of combinators with additional constants and corresponding axioms and rules with aim to capture inference are called systems of illative combinatory logic, and they have been investigated in [8, 41, 42, 51]. Another approach related to ours is introduced in [6], where Axelsen, Glück and Kaarsgaard used a classical propositional logic to reason about reversible logic circuits. The authors extended the system of reversible logic circuits with classical propositional logic extended with ordered multiplicative conjunction, so that reversible logic circuits play the role of propositions. This work and the work presented in Chapter 4 share the same philosophy of extending a basic logical system with classical propositional logic to capture reasoning about the basic logical system.

The work presented in Chapter 5 follows the approach used for probabilistic extensions of different logics. The probabilistic extension of the classical propositional logic LPP has been introduced by Rašković in [146], where the classical propositional logic is extended with probability operators of the form P_s with the intended meaning "the probability is at least s". In [128], Ognjanović extended the classical propositional logic with probability operators of the form $P_{\geq s}$ and obtained the logic LPP_2. The probability operator $P_{\geq s}$ has the same meaning as in [146], but s takes the value from an infinite set, whereas in [146] the set of values is finite. This was followed by the work presented in [129] where Ognjanović and Rašković extended the classical propositional logic with two kinds of probability operators $P_{\geq s}$ and Q_F such that the meaning of Q_F is "the probability is in F". The probabilistic extensions of the intuitionistic propositional logic and justification logic are introduced by Marković, Ognjanović and Rašković in [119] and by Kokkinis, Ognjanović and Studer in

[104], respectively. The probabilistic extension of the first-order logic is introduced by Ognjanović and Rašković in [130]. All these systems are obtained in the way that the language of a basic system, e.g. classical propositional logic, intuitionistic propositional logic, justification logic, is extended with probability operators applied to the formulas of the logic. In this way, a framework for probabilistic reasoning about different systems is obtained. The questions addressed in the mentioned papers are soundness, completeness and decidability of the obtained probability logics. The proof method that we have used in Chapter 5 follows the proof methods used in these papers. Recently, new probability logics have been developed ([49, 86, 87]). In [49], Dautović, Doder and Ognjanović have formalised the quantitative concept of confirmation, first within a propositional logical framework and then using its first-order extension. A new probabilistic extension of the intuitionistic propositional logic has been introduced by Ilić-Stepić, Knežević and Ognjanović in [86], where contrary to other works on this topic (e.g. [119]) reasoning with probability operators is also intuitionistic. The most recent work has introduced a framework for reasoning about quantum observations ([87]).

6.3 Future work

Every investigated topic leaves open questions and ideas for future work.

In Chapter 2, we have addressed the question of program equivalence in the probabilistic setting. However, another question is what happens if the inequalities associated with these equivalences are considered. It is known that similarity and testing preorder do not coincide, so in the case of inequalities we cannot use the same method as in Chapter 2. Thus, other options have to be explored. In this work, we added the let-in operator to the calculus with probabilistic operator, however we can also add the operator to a calculus with different effects, e.g. non-determinism. All these questions can also be addressed in the typed language.

Although the first idea was to develop the Kripke-style semantics of the full simply typed λ-calculus ([94]), in Chapter 3 the semantics are introduced for the full simply typed combinatory logic. If we add the typing rule that ensures that equal terms inhabit the same type to the calculi, then the translation from combinatory logic to λ-calculus is much more involved. For this reason, we have considered only combinatory logic in Chapter 3 and left the semantics of λ-calculus for future work. Besides the considered typing system, many different type systems have been introduced for λ-calculus and combinatory logic, so naturally it would be interesting to investigate if the same approach can be adapted to other type disciplines.

The work presented in Chapter 4 can be continued in several directions. We

have considered the propositional extension of the simply typed combinatory logic. The combinatory logic and λ-calculus are computationally equivalent theories, so it would be interesting to see if the same approach can be used to develop the propositional extension of the simply typed λ-calculus. Also, we could consider different type disciplines. As we have already discussed, many type systems have been developed for λ-calculus and combinatory logic. So, we could develop frameworks for reasoning about different typed calculi. In order to obtain a more expressive system, we can enrich the language and extend calculus with first-order logic. The propositional logic considered in Chapter 4 is classical and it would be interesting to investigate extending calculus with some non-classical logic such as intuitionistic logic.

Regarding the framework for probabilistic reasoning introduced in Chapter 5, there are also several directions to which the work can be extended. The basis of the presented system is the logic LCL, that is propositional extension of the simply typed combinatory logic. In the case of adapting the approach used in Chapter 4 to the simply typed λ-calculus, the next step would be developing the framework for reasoning about simply typed λ-terms. Also, changing the calculus at the basis level would result in the framework for probabilistic reasoning about different typed calculi. The introduced logic is not compact. However, if we simplify the semantics so that the set of probability values is finite, we will obtain compact logic and we can provide a finite axiomatization of it. The propositional connectives both in basic formulas and probabilistic formulas are classical connectives. Nevertheless, we can change the reasoning at the basic level to intuitionistic reasoning as in [119]. The reasoning about probabilities can also be shifted to intuitionistic, as in [86]. As for the logic PCL, the next step is the development of first-order extension of PCL.

Bibliography

[1] S. Abramsky. *Research Topics in Functional Programming*, chapter The Lazy Lambda Calculus, pages 65–116. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990. ISBN 0-201-17236-4. URL http://dl.acm.org/citation.cfm?id=119830.119834.

[2] N. Arkor and M. Fiore. Algebraic models of simple type theories: A polynomial approach. In H. Hermanns, L. Zhang, N. Kobayashi, and D. Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 88–101. ACM, 2020. doi: 10.1145/3373718.3394771. URL https://doi.org/10.1145/3373718.3394771.

[3] A. Arnauld and N. Pierre. *La logique, ou l'art de penser (Port-Royal Logic, Logique de Port-Royal, Ars cogitandi)*. C. Savreux (Paris), 1662.

[4] P. Audebaud and C. Paulin-Mohring. Proofs of randomized algorithms in coq. *Sci. Comput. Program.*, 74(8):568–589, 2009. doi: 10.1016/j.scico.2007.09.002. URL https://doi.org/10.1016/j.scico.2007.09.002.

[5] S. Awodey and F. Rabe. Kripke semantics for Martin-Löf's extensional type theory. *Log. Methods Comput. Sci.*, 7(3), 2011. doi: 10.2168/LMCS-7(3:18)2011. URL https://doi.org/10.2168/LMCS-7(3:18)2011.

[6] H. B. Axelsen, R. Glück, and R. Kaarsgaard. A classical propositional logic for reasoning about reversible logic circuits. In J. A. Väänänen, Å. Hirvonen, and R. J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation - 23rd International Workshop, WoLLIC 2016, Puebla, Mexico, August 16-19th, 2016. Proceedings*, volume 9803 of *Lecture Notes in Computer Science*, pages 52–67. Springer, 2016. doi: 10.1007/978-3-662-52921-8_4. URL https://doi.org/10.1007/978-3-662-52921-8_4.

[7] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *The Journal of Symbolic Logic*, 48(4):931–940, 1983. doi: 10.2307/2273659.

[8] H. Barendregt, M. W. Bunder, and W. Dekkers. Systems of illative combinatory logic complete for first-order propositional and predicate calculus. *J. Symb. Log.*, 58(3):769–788, 1993. doi: 10.2307/2275096. URL https://doi.org/10.2307/2275096.

[9] H. P. Barendregt. *Some extensional term models for combinatory logics and l - calculi*. PhD thesis, Univ. Utrecht, 1971.

[10] H. P. Barendregt. *The lambda calculus: its syntax and semantics*. Studies in logic and the foundations of mathematics. North-Holland, 1984. ISBN 9780444867483. URL https://books.google.rs/books?id=eMtTAAAAYAAJ.

[11] H. P. Barendregt. *The lambda calculus - its syntax and semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1985. ISBN 978-0-444-86748-3.

[12] H. P. Barendregt. *Some extensional term models for combinatory logics and lambda-calculi: motivation, the making of, 2020 hindsight*. Kindle Direct Publishing, 2020.

[13] H. P. Barendregt, W. Dekkers, and R. Statman. *Lambda Calculus with Types*. Perspectives in logic. Cambridge University Press, 2013. ISBN 978-0-521-76614-2. URL http://www.cambridge.org/de/academic/subjects/mathematics/logic-categories-and-sets/lambda-calculus-types.

[14] A. Bayart. Quasi-adéquation de la logique modale du second ordre s5 et adéquation de la logique modale du premier ordre s5 [quasi-completeness of second-order s5 modal logic and completeness of first-order s5 modal logic]. *Logique Et Analyse*, 2(6):99–121, 1959.

[15] O. Becker. *Untersuchungen Über den Modalkalkül [Investigations into Modal Calculus]*. A. Hain, 1952.

[16] M. Beeson. Lambda logic. In D. A. Basin and M. Rusinowitch, editors, *Automated Reasoning - Second International Joint Conference, IJCAR 2004, Cork, Ireland, July 4-8, 2004, Proceedings*, volume 3097 of *Lecture Notes in Computer Science*, pages 460–474. Springer, 2004. doi: 10.1007/978-3-540-25984-8_34. URL https://doi.org/10.1007/978-3-540-25984-8_34.

[17] M. Bendkowski, K. Grygiel, and M. Zaionc. Asymptotic properties of combinatory logic. In R. Jain, S. Jain, and F. Stephan, editors, *Theory and Applications of Models of Computation*, pages 62–72, Cham, 2015. Springer International Publishing. ISBN 978-3-319-17142-5.

[18] J. Bernoulli. *Ars coniectandi*. Impensis Thurnisiorum, Fratrum, 1713.

[19] J. Bessai, A. Dudenhefner, B. Düdder, M. Martens, and J. Rehof. Combinatory logic synthesizer. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change - 6th International Symposium, ISoLA 2014, Imperial, Corfu, Greece, October 8-11, 2014, Proceedings, Part I*, volume 8802 of *Lecture Notes in Computer Science*, pages 26–40. Springer, 2014. doi: 10.1007/978-3-662-45234-9_3. URL https://doi.org/10.1007/978-3-662-45234-9_3.

[20] K. Bimbó. Semantics for dual and symmetric combinatory calculi. *J. Philos. Log.*, 33(2):125–153, 2004. doi: 10.1023/B:LOGI.0000021709.73522.34. URL https://doi.org/10.1023/B:LOGI.0000021709.73522.34.

[21] K. Bimbó. *Combinatory Logic: Pure, Applied, and Typed*. CRC Press, Taylor & Francis Group, Boca Raton, Florida, 2012.

[22] A. Bizjak and L. Birkedal. Step-indexed logical relations for probability. In A. M. Pitts, editor, *Foundations of Software Science and Computation Structures - 18th International Conference, FoSSaCS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9034 of *Lecture Notes in Computer Science*, pages 279–294. Springer, 2015. doi: 10.1007/978-3-662-46678-0_18. URL https://doi.org/10.1007/978-3-662-46678-0_18.

[23] G. Boole. *An Investigation of the Laws of Thought: On which are Founded the Mathematical Theories of Logic and Probabilities*. Walton and Maberly, 1854.

[24] K. Brünnler, D. Flumini, and T. Studer. A logic of blockchain updates. *J. Log. Comput.*, 30(8):1469–1485, 2020. doi: 10.1093/logcom/exaa045. URL https://doi.org/10.1093/logcom/exaa045.

[25] M. W. Bunder. Intersection types for lambda-terms and combinators and their logics. *Log. J. IGPL*, 10(4):357–378, 2002. doi: 10.1093/jigpal/10.4.357. URL https://doi.org/10.1093/jigpal/10.4.357.

[26] R. Carnap. Modalities and quantification. *The Journal of Symbolic Logic*, 11(2):33–64, 1946. ISSN 00224812. URL http://www.jstor.org/stable/2268610.

[27] A. Church. A set of postulates for the foundation of logic. *Annals of Mathematics*, 33(2):346–366, 1932. ISSN 0003486X. URL http://www.jstor.org/stable/1968337.

[28] A. Church. A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1(1):40–41, 1936. doi: 10.2307/2269326.

[29] A. Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58:345, 1936.

[30] A. Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5(2):56–68, 1940. doi: 10.2307/2266170. URL https://doi.org/10.2307/2266170.

[31] A. Church. *The Calculi of Lambda Conversion. (AM-6)*. Princeton University Press, 1941. ISBN 9780691083940. URL http://www.jstor.org/stable/j.ctt1b9x12d.

[32] B. J. Copeland. The genesis of possible worlds semantics. *J. Philos. Log.*, 31(2):99–137, 2002. doi: 10.1023/A:1015273407895. URL https://doi.org/10.1023/A:1015273407895.

[33] C. Coquand. A formalised proof of the soundness and completeness of a simply typed lambda-calculus with explicit substitutions. *High. Order Symb. Comput.*, 15(1):57–90, 2002. doi: 10.1023/A:1019964114625. URL https://doi.org/10.1023/A:1019964114625.

[34] R. D. Cosmo and D. Kesner. A confluent reduction for the extensional typed lambda-calculus with pairs, sums, recursion and terminal object. In *Automata, Languages and Programming, 20nd International Colloquium, ICALP93, Lund, Sweden, July 5-9, 1993, Proceedings*, pages 645–656, 1993. doi: 10.1007/3-540-56939-1_109. URL https://doi.org/10.1007/3-540-56939-1_109.

[35] R. Crubillé and U. Dal Lago. On probabilistic applicative bisimulation and call-by-value λ-calculi. In *ESOP*, volume 8410 of *Lecture Notes in Computer Science*, pages 209–228. Springer, 2014.

[36] R. Crubillé, U. Dal Lago, D. Sangiorgi, and V. Vignudelli. On applicative similarity, sequentiality, and full abstraction. In *Correct System Design*, volume 9360 of *Lecture Notes in Computer Science*, pages 65–82. Springer, 2015.

[37] H. B. Curry. Grundlagen der kombinatorischen logik. *American Journal of Mathematics*, 52(3):509–536, 1930. ISSN 00029327, 10806377. URL http://www.jstor.org/stable/2370619.

[38] H. B. Curry. Functionality in combinatory logic. *Proceedings of the National Academy of Sciences of the United States of America*, 20(11): 584–590, 1934. ISSN 00278424. URL http://www.jstor.org/stable/86796.

[39] H. B. Curry and R. Feys. *Combinatory Logic*. Number v. 1 in Combinatory Logic. North-Holland Publishing Company, 1958. URL https://books.google.ba/books?id=fEnuAAAAMAAJ.

[40] G. Curzi and M. Pagani. The benefit of being non-lazy in probabilistic λ-calculus: Applicative bisimulation is fully abstract for non-lazy probabilistic call-by-name. In H. Hermanns, L. Zhang, N. Kobayashi, and D. Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 327–340. ACM, 2020. doi: 10.1145/3373718.3394806. URL https://doi.org/10.1145/3373718.3394806.

[41] L. Czajka. A semantic approach to illative combinatory logic. In M. Bezem, editor, *Computer Science Logic, 25th International Workshop / 20th Annual Conference of the EACSL, CSL 2011, September 12-15, 2011, Bergen, Norway, Proceedings*, volume 12 of *LIPIcs*, pages 174–188. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011. doi: 10.4230/LIPIcs.CSL.2011.174. URL https://doi.org/10.4230/LIPIcs.CSL.2011.174.

[42] L. Czajka. Higher-order illative combinatory logic. *J. Symb. Log.*, 78(3): 837–872, 2013. doi: 10.2178/jsl.7803080. URL https://doi.org/10.2178/jsl.7803080.

[43] U. Dal Lago and F. Gavazzo. On bisimilarity in lambda calculi with continuous probabilistic choice. In B. König, editor, *Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2019, London, UK, June 4-7, 2019*, volume 347 of *Electronic Notes in Theoretical Computer Science*, pages 121–141. Elsevier, 2019. doi: 10.1016/j.entcs.2019.09.007. URL https://doi.org/10.1016/j.entcs.2019.09.007.

[44] U. Dal Lago and A. Rioli. Applicative bisimulation and quantum λ-calculi. In M. Dastani and M. Sirjani, editors, *Fundamentals of Software Engineering - 6th International Conference, FSEN 2015*

Tehran, Iran, April 22-24, 2015, Revised Selected Papers, volume 9392 of *Lecture Notes in Computer Science*, pages 54–68. Springer, 2015. doi: 10.1007/978-3-319-24644-4_4. URL https://doi.org/10.1007/978-3-319-24644-4_4.

[45] U. Dal Lago and M. Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications*, 46(3):413–450, 2012. doi: 10.1051/ita/2012012. URL http://www.numdam.org/item/ITA_2012__46_3_413_0.

[46] U. Dal Lago, D. Sangiorgi, and M. Alberti. On coinductive equivalences for higher-order probabilistic functional programs (long version). *CoRR*, abs/1311.1722, 2013.

[47] U. Dal Lago, D. Sangiorgi, and M. Alberti. On coinductive equivalences for higher-order probabilistic functional programs. In *POPL*, pages 297–308. ACM, 2014.

[48] U. Dal Lago, F. Gavazzo, and P. B. Levy. Effectful applicative bisimilarity: Monads, relators, and howe's method. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi: 10.1109/LICS.2017.8005117. URL https://doi.org/10.1109/LICS.2017.8005117.

[49] S. Dautović, D. Doder, and Z. Ognjanović. Logics for reasoning about degrees of confirmation. *J. Log. Comput.*, 31(8):2189–2217, 2021. doi: 10.1093/logcom/exab033. URL https://doi.org/10.1093/logcom/exab033.

[50] R. C. de Vrijer. Extending the lambda calculus with surjective pairing is conservative. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 204–215, 1989. doi: 10.1109/LICS.1989.39175. URL https://doi.org/10.1109/LICS.1989.39175.

[51] W. Dekkers, M. W. Bunder, and H. Barendregt. Completeness of two systems of illative combinatory logic for first-order propositional and predicate calculus. *Arch. Math. Log.*, 37(5-6):327–341, 1998. doi: 10.1007/s001530050102. URL https://doi.org/10.1007/s001530050102.

[52] J. Desclés. Combinatory logic, language, and cognitive representations. In P. Weingartner, editor, *Alternative Logics. Do Sciences Need Them?*, pages 115–148. Springer Verlag, 2004.

[53] B. Düdder, M. Martens, J. Rehof, and P. Urzyczyn. Bounded combinatory logic. In P. Cégielski and A. Durand, editors, *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 243–258. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012. doi: 10.4230/LIPIcs.CSL.2012.243. URL `https://doi.org/10.4230/LIPIcs.CSL.2012.243`.

[54] J. M. Dunn and R. K. Meyer. Combinators and structurally free logic. *Log. J. IGPL*, 5(4):505–537, 1997. doi: 10.1093/jigpal/5.4.505. URL `https://doi.org/10.1093/jigpal/5.4.505`.

[55] T. Ehrhard, M. Pagani, and C. Tasson. The computational meaning of probabilistic coherence spaces. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011, June 21-24, 2011, Toronto, Ontario, Canada*, pages 87–96. IEEE Computer Society, 2011. doi: 10.1109/LICS.2011.29. URL `https://doi.org/10.1109/LICS.2011.29`.

[56] R. Fagin, J. Y. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87(1/2):78–128, 1990. doi: 10.1016/0890-5401(90)90060-U. URL `https://doi.org/10.1016/0890-5401(90)90060-U`.

[57] M. Fattorosi-Barnaba and G. Amati. Modal operators with probabilistic interpretations, I. *Studia Logica*, 46(4):383–393, 1987. doi: 10.1007/BF00370648. URL `https://doi.org/10.1007/BF00370648`.

[58] R. Feys. La transcription logistique du raisonnement: son intérét et ses limites [the logistic transcription of reasoning: Its importance and its limits]. *Revue néoscolastique de philosophie*, 26:299–324, 1924. ISSN 0776555X, 22959122. URL `http://www.jstor.org/stable/26343715`.

[59] J. H. Gallier. Kripke models and the (in)equational logic of the second-order lambda-calculus. *Ann. Pure Appl. Log.*, 84(3):257–316, 1997. doi: 10.1016/S0168-0072(96)00039-5. URL `https://doi.org/10.1016/S0168-0072(96)00039-5`.

[60] D. Garrette, C. Dyer, J. Baldridge, and N. A. Smith. Weakly-supervised grammar-informed bayesian CCG parser learning. In B. Bonet and

S. Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2246–2252. AAAI Press, 2015. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9835.

[61] S. Ghilezan and S. Kašterović. Towards completeness of full simply typed lambda calculus. In *26th International Conference on Types for Proofs and Programs - TYPES 2020*, pages 164–166, March 2-5, 2020.

[62] S. Ghilezan and S. Kašterović. Semantics for combinatory logic with intersection types. *Frontiers Comput. Sci.*, 4, 2022. doi: 10.3389/FCOMP.2022.792570. URL https://doi.org/10.3389/fcomp.2022.792570.

[63] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Towards probabilistic reasoning about simply typed lambda terms. In *7th Conference on Probability Logics and Applications - VLP 2017*, Belgrade, Serbia, November 8, 2017.

[64] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Probabilistic reasoning about simply typed lambda terms. In S. N. Artëmov and A. Nerode, editors, *Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings*, volume 10703 of *Lecture Notes in Computer Science*, pages 170–189. Springer, 2018. doi: 10.1007/978-3-319-72056-2_11. URL https://doi.org/10.1007/978-3-319-72056-2_11.

[65] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Towards probabilistic reasoning about typed lambda terms. In *24th International Conference on Types for Proofs and Programs - TYPES 2018*, pages 41–42, Braga, Portugal, June 18 - 21, 2018.

[66] S. Ghilezan, J. Ivetić, S. Kašterović, Z. Ognjanović, and N. Savić. Towards probabilistic reasoning in type theory - the intersection type case. In A. Herzig and J. Kontinen, editors, *Foundations of Information and Knowledge Systems - 11th International Symposium, FoIKS 2020, Dortmund, Germany, February 17-21, 2020, Proceedings*, volume 12012 of *Lecture Notes in Computer Science*, pages 122–139. Springer, 2020. doi: 10.1007/978-3-030-39951-1_8. URL https://doi.org/10.1007/978-3-030-39951-1_8.

[67] J.-Y. Girard, P. Taylor, and Y. Lafont. *Proofs and Types*. Cambridge University Press, New York, NY, USA, 1989. ISBN 0-521-37181-3.

[68] K. Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für Mathematik und Physik*, 38: 173–198, 1931.

[69] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. doi: 10.1016/0022-0000(84)90070-9. URL https://doi.org/10.1016/0022-0000(84)90070-9.

[70] N. D. Goodman. The principles and practice of probabilistic programming. In R. Giacobazzi and R. Cousot, editors, *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '13, Rome, Italy - January 23 - 25, 2013*, pages 399–402. ACM, 2013. doi: 10.1145/2429069.2429117. URL https://doi.org/10.1145/2429069.2429117.

[71] A. D. Gordon. Bisimilarity as a theory of functional programming. *Theor. Comput. Sci.*, 228(1-2):5–47, 1999. doi: 10.1016/S0304-3975(98)00353-3. URL https://doi.org/10.1016/S0304-3975(98)00353-3.

[72] T. Hailperin. Best possible inequalities for the probability of a logical function of events. *The American Mathematical Monthly*, 72(4):343–359, 1965. ISSN 00029890, 19300972. URL http://www.jstor.org/stable/2313491.

[73] J. Y. Halpern and R. Pucella. A logic for reasoning about evidence. *J. Artif. Int. Res.*, 26(1):1–34, may 2006. ISSN 1076-9757.

[74] C. L. Hamblin. The modal "probably". *Mind*, 68(270):234–240, 1959. doi: 10.1093/mind/lxviii.270.234.

[75] C. Hartshorne and P. Weiss, editors. *Collected Papers of Charles Sanders Peirce, Vol. II: Elements of Logic*. Harvard University Press, Cambridge, Mass, 1932.

[76] C. Hartshorne and P. Weiss, editors. *Collected Papers of Charles Sanders Peirce, Vol. III: Exact Logic*. Harvard University Press, Cambridge, Mass, 1933.

[77] J. R. Hindley. The completeness theorem for typing lambda-terms. *Theor. Comput. Sci.*, 22:1–17, 1983. doi: 10.1016/0304-3975(83)90136-6. URL https://doi.org/10.1016/0304-3975(83)90136-6.

[78] J. R. Hindley and J. P. Seldin. *Lambda-Calculus and Combinators: An Introduction*. Cambridge University Press, 2 edition, 2008. doi: 10.1017/CBO9780511809835.

[79] J. Hintikka. Form and content in quantification theory. *Acta Philosophica Fennica*, 8:7–55, 1955.

[80] J. Hintikka. Notes on quantification theory. *Societas Scientiarum Fennica, Commentationes Physico-Mathematicae*, 17(12), 1955.

[81] Y. Hirai. Blockchains as kripke models: An analysis of atomic cross-chain swap. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice - 8th International Symposium, ISoLA 2018, Limassol, Cyprus, November 5-9, 2018, Proceedings, Part IV*, volume 11247 of *Lecture Notes in Computer Science*, pages 389–404. Springer, 2018. doi: 10.1007/978-3-030-03427-6_29. URL https://doi.org/10.1007/978-3-030-03427-6_29.

[82] D. N. Hoover. Probability logic. *Annals of Mathematical Logic*, 14(3): 287, 1978. doi: 10.1016/0003-4843(78)90022-0.

[83] W. A. Howard. The formulae-as-types notion of construction. In *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490. London : Academic Press, 1980 (originally circulated 1969). ISBN 978-0-12-349050-6.

[84] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Inf. Comput.*, 124(2):103–112, 1996. doi: 10.1006/inco.1996.0008. URL https://doi.org/10.1006/inco.1996.0008.

[85] J. Z. S. Hu and B. Pientka. A categorical normalization proof for the modal lambda-calculus. *CoRR*, abs/2211.12318, 2022. doi: 10.48550/arXiv.2211.12318. URL https://doi.org/10.48550/arXiv.2211.12318.

[86] A. Ilić-Stepić, M. Knežević, and Z. Ognjanović. Intuitionistic propositional probability logic. *Math. Log. Q.*, 68(4):479–495, 2022. doi: 10.1002/malq.202100052. URL https://doi.org/10.1002/malq.202100052.

[87] A. Ilić-Stepić, Z. Ognjanović, and A. Perović. Probability logics for reasoning about quantum observations. *Logica Universalis*, 17(2):175–219, 2023. doi: 10.1007/s11787-023-00326-y. URL https://doi.org/10.1007/s11787-023-00326-y.

[88] S. S. Ishtiaq and D. J. Pym. Kripke resource models of a dependently-typed, bunched lambda-calculus. In J. Flum and M. Rodríguez-Artalejo, editors, *Computer Science Logic, 13th International Workshop, CSL '99,*

8th Annual Conference of the EACSL, Madrid, Spain, September 20-25, 1999, Proceedings, volume 1683 of *Lecture Notes in Computer Science*, pages 235–249. Springer, 1999. doi: 10.1007/3-540-48168-0_17. URL https://doi.org/10.1007/3-540-48168-0_17.

[89] C. Jones and G. D. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 186–195. IEEE Computer Society, 1989. doi: 10.1109/LICS.1989.39173. URL https://doi.org/10.1109/LICS.1989.39173.

[90] F. Kamareddine, T. Laan, and R. Nederpelt. Types in logic and mathematics before 1940. *Bull. Symb. Log.*, 8(2):185–245, 2002. doi: 10.2178/bsl/1182353871. URL https://doi.org/10.2178/bsl/1182353871.

[91] N. Kamide, Y. Shramko, and H. Wansing. Kripke completeness of bi-intuitionistic multilattice logic and its connexive variant. *Studia Logica*, 105(6):1193–1219, 2017. doi: 10.1007/s11225-017-9752-x.

[92] S. Kašterovć and S. Ghilezan. Towards probabilistic reasoning about typed combinatory terms. In *28th International Conference on Types for Proofs and Programs - TYPES 2022*, Nantes, France, June 20-23, 2022.

[93] S. Kašterović and S. Ghilezan. Kripke-style semantics for full simply typed lambda calculus. In *9th International Conference on Logic and Applications - LAP 2020*, pages 12–14, September 21 - 25, 2020.

[94] S. Kašterović and S. Ghilezan. Kripke-style semantics and completeness for full simply typed lambda calculus. *J. Log. Comput.*, 30(8):1567–1608, 2020. doi: 10.1093/logcom/exaa055. URL https://doi.org/10.1093/logcom/exaa055.

[95] S. Kašterović and S. Ghilezan. Towards logic of combinatory logic. In *10th International Conference on Logic and Applications - LAP 2021*, pages 34–36, Dubrovnik, Croatia, September 20 - 24, 2021.

[96] S. Kašterović and S. Ghilezan. Probabilistic reasoning about typed combinatory logic. In *11th International Conference on Logic and Applications - LAP 2022*, pages 29–31, Dubrovnik, Croatia, September 26-29, 2022.

[97] S. Kašterović and S. Ghilezan. Logic of combinatory logic. *CoRR*, abs/2212.06675, 2022. doi: 10.48550/arXiv.2212.06675. URL https://doi.org/10.48550/arXiv.2212.06675.

[98] S. Kašterović and M. Pagani. Towards probabilistic testing of lambda terms. In *7th International Conference on Logic and Applications - LAP 2018*, pages 21–23, Dubrovnik, Croatia, September 24-28, 2018.

[99] S. Kašterović and M. Pagani. The discriminating power of the let-in operator in the lazy call-by-name probabilistic lambda-calculus. In H. Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPIcs*, pages 26:1–26:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi: 10.4230/LIPIcs.FSCD.2019.26. URL `https://doi.org/10.4230/LIPIcs.FSCD.2019.26`.

[100] H. J. Keisler. Hyperfinite model theory. In R. Gandy and J. Hyland, editors, *Logic Colloquim*, volume 76, pages 5–110. North-Holland, 1977.

[101] H. J. Keisler. Probability quantifiers. In J. Barwise and S. Feferman, editors, *Model-Theoretic Logics*, Perspectives in Logic, page 509–556. Springer, Berlin, 1985. doi: 10.1017/9781316717158.021.

[102] S. C. Kleene. λ-definability and recursiveness. *Duke Mathematical Journal*, 2:340–353, 1936.

[103] S. C. Kleene and J. B. Rosser. The inconsistency of certain formal logics. *Annals of Mathematics*, 36(3):630–636, 1935. ISSN 0003486X. URL `http://www.jstor.org/stable/1968646`.

[104] I. Kokkinis, Z. Ognjanović, and T. Studer. Probabilistic justification logic. In S. N. Artëmov and A. Nerode, editors, *Logical Foundations of Computer Science - International Symposium, LFCS 2016, Deerfield Beach, FL, USA, January 4-7, 2016. Proceedings*, volume 9537 of *Lecture Notes in Computer Science*, pages 174–186. Springer, 2016. doi: 10.1007/978-3-319-27683-0_13. URL `https://doi.org/10.1007/978-3-319-27683-0_13`.

[105] L. Koralov and Y. G. Sinai. *Theory of Probability and Random Processes*. Universitext (Berlin. Print). Springer, 2007. ISBN 9783540254843. URL `https://books.google.rs/books?id=QILj7e_iPZEC`.

[106] V. Koutavas, P. B. Levy, and E. Sumii. From applicative to environmental bisimulation. In M. W. Mislove and J. Ouaknine, editors, *Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics, MFPS 2011, Pittsburgh, PA, USA, May 25-28, 2011*, volume 276 of *Electronic Notes in Theoretical Computer Science*, pages 215–235. Elsevier, 2011. doi: 10.1016/j.entcs.2011.09.023. URL `https://doi.org/10.1016/j.entcs.2011.09.023`.

[107] M. Kraus. Early greek probability arguments and common ground in dissensus. In *Ontario Society for the Study of Argumentation (OSSA) Proceedings*, pages 1–11. OSSA Conference Archive, 2007.

[108] S. A. Kripke. A completeness theorem in modal logic. *Journal of Symbolic Logic*, 24(1):1–14, 1959. doi: 10.2307/2964568.

[109] S. A. Kripke. Semantical analysis of modal logic i. normal propositional calculi. *Zeitschrift fur mathematische Logik und Grundlagen der Mathematik*, 9(5-6):67–96, 1963. doi: 10.1002/malq.19630090502.

[110] S. A. Kripke. Semantical analysis of intuitionistic logic i. In *Formal Systems and Recursive Functions*, volume 40 of *Studies in Logic and the Foundations of Mathematics*, pages 92 – 130. Elsevier, 1965. doi: https://doi.org/10.1016/S0049-237X(08)71685-9. URL `http://www.sciencedirect.com/science/article/pii/S0049237X08716859`.

[111] K. G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991.

[112] S. B. Lassen. *Relational Reasoning about Functions and Nondeterminism*. PhD thesis, University of Aarhus, 1998.

[113] G. W. Leibniz. *De incerti aestimatione*. PhD thesis, 1678.

[114] S. Lenglet, A. Schmitt, and J. Stefani. Howe's method for calculi with passivation. In M. Bravetti and G. Zavattaro, editors, *CONCUR 2009 - Concurrency Theory, 20th International Conference, CONCUR 2009, Bologna, Italy, September 1-4, 2009. Proceedings*, volume 5710 of *Lecture Notes in Computer Science*, pages 448–462. Springer, 2009. doi: 10.1007/978-3-642-04081-8_30. URL `https://doi.org/10.1007/978-3-642-04081-8_30`.

[115] P. Liang, M. I. Jordan, and D. Klein. Learning programs: A hierarchical bayesian approach. In J. Fürnkranz and T. Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 639–646. Omnipress, 2010. URL `https://icml.cc/Conferences/2010/papers/568.pdf`.

[116] J. Lipton. Kripke semantics for dependent type theory and realizability interpretations. In J. P. M. Jr. and M. J. O'Donnell, editors, *Constructivity in Computer Science, Summer Symposium, San Antonio, Texas, USA, June 19-22, Proceedings*, volume 613 of *Lecture Notes in Computer Science*, pages 22–32. Springer, 1991. doi: 10.1007/BFb0021080. URL `https://doi.org/10.1007/BFb0021080`.

[117] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-13360-1.

[118] B. Marinković, P. Glavan, Z. Ognjanović, D. Doder, and T. Studer. Probabilistic consensus of the blockchain protocol. In G. Kern-Isberner and Z. Ognjanović, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 15th European Conference, ECSQARU 2019, Belgrade, Serbia, September 18-20, 2019, Proceedings*, volume 11726 of *Lecture Notes in Computer Science*, pages 469–480. Springer, 2019. doi: 10.1007/978-3-030-29765-7_39. URL https://doi.org/10.1007/978-3-030-29765-7_39.

[119] Z. Marković, Z. Ognjanović, and M. Rašković. A probabilistic extension of intuitionistic logic. *Math. Log. Q.*, 49(4):415–424, 2003. doi: 10.1002/malq.200310044. URL https://doi.org/10.1002/malq.200310044.

[120] J. C. C. Mckinsey. On the syntactical construction of systems of modal logic. *The Journal of Symbolic Logic*, 10(3):83–94, 1945. doi: 10.2307/2267027.

[121] C. A. Meredith and A. N. Prior. Interpretations of different modal logics in the 'property calculus'. Mimeograph, University of Canterbury, Philosophy Department, 1956.

[122] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980. ISBN 3-540-10235-3. doi: 10.1007/3-540-10235-3. URL https://doi.org/10.1007/3-540-10235-3.

[123] J. C. Mitchell. *Foundations for programming languages*. Foundation of computing series. MIT Press, 1996. ISBN 978-0-262-13321-0.

[124] J. C. Mitchell and E. Moggi. Kripke-style models for typed lambda calculus. *Annals of Pure and Applied Logic*, 51(1-2):99–124, 1991. doi: 10.1016/0168-0072(91)90067-V. URL https://doi.org/10.1016/0168-0072(91)90067-V.

[125] R. Montague. Logical necessity, physical necessity, ethics, and quantifiers. *Inquiry: An Interdisciplinary Journal of Philosophy*, 3(1-4):259–269, 1960. doi: 10.1080/00201746008601312.

[126] J. H. Morris. *Lambda-calculus models of programming languages*. PhD thesis, Massachusetts Institute of Technology, 1969.

[127] N. J. Nilsson. Probabilistic logic. *Artificial Intelligence*, 28(1):71–87, 1986. doi: 10.1016/0004-3702(86)90031-7. URL https://doi.org/10.1016/0004-3702(86)90031-7.

[128] Z. Ognjanović. *Some probability logics and their applications in computer sciences.* PhD Thesis (in Serbian), University of Kragujevac, 1999.

[129] Z. Ognjanović and M. Rašković. Some probability logics with new types of probability operators. *J. Log. Comput.*, 9(2):181–195, 1999. doi: 10.1093/logcom/9.2.181. URL https://doi.org/10.1093/logcom/9.2.181.

[130] Z. Ognjanović and M. Rašković. Some first-order probability logics. *Theor. Comput. Sci.*, 247(1-2):191–212, 2000. doi: 10.1016/S0304-3975(98)00341-7. URL https://doi.org/10.1016/S0304-3975(98)00341-7.

[131] Z. Ognjanović, A. Perović, and M. Rašković. Logics with the qualitative probability operator. *Log. J. IGPL*, 16(2):105–120, 2008. doi: 10.1093/jigpal/jzm031. URL https://doi.org/10.1093/jigpal/jzm031.

[132] Z. Ognjanović, M. Rašković, and Z. Marković. *Probability Logics: Probability-Based Formalization of Uncertain Reasoning.* Springer Publishing Company, Incorporated, 1st edition, 2016. ISBN 3319470116.

[133] D. Park. A new equivalence notion for communicating systems. In G. Maurer, editor, *Bulletin EATCS*, volume 14, pages 78–80, 1981. Abstract of the talk presented at the Second Workshop on the Semantics of Programming Languages, Bad Honnef, March 16–20 1981. Abstracts collected in the Bulletin by B. Mayoh.

[134] S. Park, F. Pfenning, and S. Thrun. A probabilistic language based on sampling functions. *ACM Trans. Program. Lang. Syst.*, 31(1):4:1–4:46, 2008. doi: 10.1145/1452044.1452048. URL https://doi.org/10.1145/1452044.1452048.

[135] J. Pearl. *Probabilistic reasoning in intelligent systems - networks of plausible inference.* Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann, 1989.

[136] A. Pfeffer. IBAL: A probabilistic rational programming language. In B. Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, pages 733–740. Morgan Kaufmann, 2001.

[137] B. C. Pierce. *Types and programming languages*. MIT Press, 2002. ISBN 978-0-262-16209-8.

[138] B. C. Pierce, A. A. de Amorim, C. Casinghino, M. Gaboardi, M. Greenberg, C. Hriţcu, V. Sjöberg, A. Tolmach, and B. Yorgey. *Programming Language Foundations*. Software Foundations series, volume 2. Electronic textbook, May 2018. URL `http://www.cis.upenn.edu/~bcpierce/sf`. Version 5.5.

[139] A. M. Pitts. Howe's method for higher-order languages. In D. Sangiorgi and J. J. M. M. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge tracts in theoretical computer science*, pages 197–232. Cambridge University Press, 2012.

[140] A. Prior. *Past, Present and Future*. Oxford, England: Clarendon Press, 1967.

[141] A. N. Prior. *Time and Modality*. Oxford University Press, Greenwood Press, 1957.

[142] A. N. Prior. The syntax of time-distinctions. *Franciscan Studies*, 18(2): 105–120, 1958. doi: 10.1353/frc.1958.0008.

[143] A. N. Prior. Tense-logic and the continuity of time. *Studia Logica*, 13 (1):133–151, 1962. doi: 10.1007/bf02317267.

[144] F. P. Ramsey. The foundations of mathematics. *Proceedings of the London Mathematical Society*, s2-25(1):338–384, 1926. doi: https://doi.org/10.1112/plms/s2-25.1.338. URL `https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s2-25.1.338`.

[145] N. Ramsey and A. Pfeffer. Stochastic lambda calculus and monads of probability distributions. In J. Launchbury and J. C. Mitchell, editors, *Conference Record of POPL 2002: The 29th SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Portland, OR, USA, January 16-18, 2002*, pages 154–165. ACM, 2002. doi: 10.1145/503272.503288. URL `https://doi.org/10.1145/503272.503288`.

[146] M. Rašković. Classical logic with some probability operators. *Publications de l'Institut Mathématique*, 53(67), 1993.

[147] M. Rašković, Z. Ognjanović, and Z. Marković. A logic with conditional probabilities. In J. J. Alferes and J. A. Leite, editors, *Logics in Artificial Intelligence, 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004, Proceedings*, volume 3229 of

Lecture Notes in Computer Science, pages 226–238. Springer, 2004. doi: 10.1007/978-3-540-30227-8_21. URL https://doi.org/10.1007/978-3-540-30227-8_21.

[148] S. Roman. *Lattices and ordered sets*. Springer New York, NY, 2008.

[149] G. Ryle. Meaning and necessity. *Philosophy*, 24(88):69–76, 1949. ISSN 00318191, 1469817X. URL http://www.jstor.org/stable/3747236.

[150] D. Sands. From SOS rules to proof principles: An operational metatheory for functional languages. In P. Lee, F. Henglein, and N. D. Jones, editors, *Conference Record of POPL'97: The 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Papers Presented at the Symposium, Paris, France, 15-17 January 1997*, pages 428–441. ACM Press, 1997. doi: 10.1145/263699.263760. URL https://doi.org/10.1145/263699.263760.

[151] D. Sangiorgi, N. Kobayashi, and E. Sumii. Logical bisimulations and functional languages. In F. Arbab and M. Sirjani, editors, *International Symposium on Fundamentals of Software Engineering, International Symposium, FSEN 2007, Tehran, Iran, April 17-19, 2007, Proceedings*, volume 4767 of *Lecture Notes in Computer Science*, pages 364–379. Springer, 2007. doi: 10.1007/978-3-540-75698-9_24. URL https://doi.org/10.1007/978-3-540-75698-9_24.

[152] D. Sangiorgi, N. Kobayashi, and E. Sumii. Environmental bisimulations for higher-order languages. *ACM Trans. Program. Lang. Syst.*, 33(1): 5:1–5:69, 2011. doi: 10.1145/1889997.1890002. URL https://doi.org/10.1145/1889997.1890002.

[153] G. Scherer. Deciding equivalence with sums and the empty type. In G. Castagna and A. D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 374–386. ACM, 2017. doi: 10.1145/3009837.3009901. URL https://doi.org/10.1145/3009837.3009901.

[154] I. A. Schneider. The role of Leibnitz and Jacob Bernoulli for the development of probability theory. In *Boletin de la Sociedad Espanola de Historia de las Ciencias*, volume 7, pages 68–89, 1984.

[155] M. Schönfinkel. Über die bausteine der mathematischen logik. *Mathematische Annalen*, 1924.

[156] D. S. Scott. A construction of a model for the λ-calculus. Informally distributed, Notes for a November 1969 seminar, Oxford University, 1969.

[157] D. S. Scott. Lattice-theoretic models for the λ-calculus. Incomplete typescript, 50 pp., Merton College, Oxford University, 1969.

[158] D. S. Scott. A type-theoretical alternative to ISWIM, CUCH, OWHY. *Theor. Comput. Sci.*, 121(1&2):411–440, 1993. doi: 10.1016/0304-3975(93)90095-B. URL `https://doi.org/10.1016/0304-3975(93)90095-B`.

[159] T. Smiley. Modal logic. lecture handout, Department of Philosophy, University of Cambridge, 1957.

[160] T. Smiley. *Natural Systems of Logic*. PhD thesis, University of Cambridge, 1955.

[161] R. Smullyan. *To Mock a Mockingbird: And Other Logic Puzzles*. Oxford University Press, 1985.

[162] M. Steedman and J. Baldridge. Combinatory categorial grammar. In R. Borsley and K. Börjars, editors, *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, pages 181–224. Wiley-Blackwell, 2011.

[163] S. Thurn. *Exploring Artificial Intelligence in the New Millennium*, chapter Robotic Mapping: A Survey, pages 1–35. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003. ISBN 1-55860-811-7. URL `http://dl.acm.org/citation.cfm?id=779343.779345`.

[164] Z. Toffano and F. Dubois. Adapting logic to physics: The quantum-like eigenlogic program. *Entropy*, 22(2):139, 2020. doi: 10.3390/e22020139. URL `https://doi.org/10.3390/e22020139`.

[165] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1936. doi: https://doi.org/10.1112/plms/s2-42.1.230. URL `https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/plms/s2-42.1.230`.

[166] F. Van Breugel, M. W. Mislove, J. Ouaknine, and J. Worrell. Domain theory, testing and simulation for labelled markov processes. *Theor. Comput. Sci.*, 333(1-2):171–197, 2005.

[167] W. van der Hoek. Some considerations on the logic P_FD. *J. Appl. Non Class. Logics*, 7(3), 1997.

[168] A. N. Whitehead and B. Russell. *Principia Mathematica.* Cambridge University Press, 1925–1927.

[169] L. Wittgenstein. *Tractatus Logico-Philosophicus.* Routledge and Kegan Paul, London, 1922.

Simona Prokić (née Kašterović) is born on March 12, 1992 in Brčko, Bosnia and Herzegovina. She completed her undergraduate academic studies at the program of studies Bachelor with Honours in Mathematics Teaching at the Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad in June 2015 with GPA 9.42 (on a scale from 6.00 to 10.00). In 2015, she started the master academic studies in Applied Mathematics Program at the Faculty of Technical Sciences, University of Novi Sad. By June 2016 she finished all her exams, with GPA 9.78 (on a scale from 6.00 to 10.00) and defended her Masters thesis "Kripke models for intuitionistic logic and lambda calculus" in July 2017, under the supervision of Prof. Silvia Ghilezan. In October 2017, she enrolled in the doctoral program Applied Mathematics at the Faculty of Technical Sciences, University of Novi Sad. Since 2016, she works as a teaching assistant at the Faculty of Technical Sciences, University of Novi Sad.

As a recipient of Erasmus+ scholarship for PhD students, she spent three months on a research visit at University Paris Diderot (Paris 7) in Paris, France. She has participated as a speaker in several national and international conferences and attended four summer schools and one winter school. She is a co-author of four papers in international conference proceedings and one paper in a refereed international journal. She has been a participant in the project "ON 174026: Representations of logical structures and formal languages and their application in computing" financed by the Ministry of Education, Science and Technological Development and in the project "AI4TrustBC: Advanced artificial intelligence techniques for analysis and design of system components based on trustworthy BlockChain technology" (2020-2023) financed by Science Fund of the Republic of Serbia. At the moment, she participates in the project "TaRDIS: Trustworthy and Resilient Decentralised Intelligence for Edge Systems" funded by the European Union. She has been a participant in COST action CA15123: "EUTYPES: The European research network on types for programming and verification" (2015-2020). Currently, she is a grant awarding coordinator in COST action CA20111 "EuroProofNet: European Research Network on Formal Proofs".

План третмана података

Назив пројекта/истраживања
Вероватносно закључивање у израчунавању и теорији функционалних типова Probabilistic reasoning in computation and simple type theory
Назив институције/институција у оквиру којих се спроводи истраживање
а) Факултет техничких наука, Универзитет у Новом Саду
Назив програма у оквиру ког се реализује истраживање
Истраживање се врши у оквиру израде докторске дисертације на студијском програму Математика у техници.
1. Опис података

1.1 Врста студије

Укратко описати тип студије у оквиру које се подаци прикупљају

У овој студији нису прикупљани подаци.

1.2 Врсте података

а) квантитативни

б) квалитативни

1.3. Начин прикупљања података

а) анкете, упитници, тестови

б) клиничке процене, медицински записи, електронски здравствени записи

в) генотипови: навести врсту _____

г) административни подаци: навести врсту _____

д) узорци ткива: навести врсту_____

ђ) снимци, фотографије: навести врсту_____

е) текст, навести врсту _____

ж) мапа, навести врсту _____

з) остало: описати _____

1.3 Формат података, употребљене скале, количина података

1.3.1 Употребљени софтвер и формат датотеке:

a) Excel фајл, датотека _____

b) SPSS фајл, датотека _____

c) PDF фајл, датотека _____

d) Текст фајл, датотека _____

e) JPG фајл, датотека _____

f) Остало, датотека _____

1.3.2. Број записа (код квантитативних података)

а) број варијабли _____

б) број мерења (испитаника, процена, снимака и сл.) _____

1.3.3. Поновљена мерења

а) да

б) не

Уколико је одговор да, одговорити на следећа питања:

а) временски размак између поновљених мера је _____

б) варијабле које се више пута мере односе се на _____

в) нове верзије фајлова који садрже поновљена мерења су именоване као _____

Напомене: _____

Да ли формати и софтвер омогућавају дељење и дугорочну валидност података?

а) Да

б) Не

Ако је одговор не, образложити _____

_____

2. Прикупљање података

2.1 Методологија за прикупљање/генерисање података

2.1.1. У оквиру ког истраживачког нацрта су подаци прикупљени?

а) експеримент, навести тип _____

б) корелационо истраживање, навести тип _____

ц) анализа текста, навести тип _____

д) остало, навести шта _____

2.1.2 Навести врсте мерних инструмената или стандарде података специфичних за одређену научну дисциплину (ако постоје).

_____

_____

2.2 Квалитет података и стандарди

2.2.1. Третман недостајућих података

а) Да ли матрица садржи недостајуће податке? Да Не

Ако је одговор да, одговорити на следећа питања:

а) Колики је број недостајућих података? _____

б) Да ли се кориснику матрице препоручује замена недостајућих података? Да Не

в) Ако је одговор да, навести сугестије за третман замене недостајућих података

_____

2.2.2. На који начин је контролисан квалитет података? Описати

_____

_____

2.2.3. На који начин је извршена контрола уноса података у матрицу?

_____

_____

3. Третман података и пратећа документација

3.1. Третман и чување података

3.1.1. Подаци ће бити депоновани у _____ репозиторијум.

3.1.2. URL адреса _____

3.1.3. DOI _____

3.1.4. Да ли ће подаци бити у отвореном приступу?

а) Да

б) Да, али после ембарга који ће трајати до _____

в) Не

Ако је одговор не, навести разлог _____

3.1.5. Подаци неће бити депоновани у репозиторијум, али ће бити чувани.

Образложење

_____

_____

3.2 Метаподаци и документација података

3.2.1. Који стандард за метаподатке ће бити примењен? _____

3.2.1. Навести метаподатке на основу којих су подаци депоновани у репозиторијум.

_____

_____

Ако је потребно, навести методе које се користе за преузимање података, аналитичке и процедуралне информације, њихово кодирање, детаљне описе варијабли, записа итд.

_____

_____

_____

_____

3.3 Стратегија и стандарди за чување података

3.3.1. До ког периода ће подаци бити чувани у репозиторијуму? _____

3.3.2. Да ли ће подаци бити депоновани под шифром? Да Не

3.3.3. Да ли ће шифра бити доступна одређеном кругу истраживача? Да Не

3.3.4. Да ли се подаци морају уклонити из отвореног приступа после извесног времена?

Да Не

Образложити

_____

_____

4. Безбедност података и заштита поверљивих информација

Овај одељак МОРА бити попуњен ако ваши подаци укључују личне податке који се односе на учеснике у истраживању. За друга истраживања треба такође размотрити заштиту и сигурност података.

4.1 Формални стандарди за сигурност информација/података

Истраживачи који спроводе испитивања с људима морају да се придржавају Закона о заштити података о личности *(https://www.paragraf.rs/propisi/zakon_o_zastiti_podataka_o_licnosti.html)* и одговарајућег институционалног кодекса о академском интегритету.

4.1.2. Да ли је истраживање одобрено од стране етичке комисије? Да Не

Ако је одговор Да, навести датум и назив етичке комисије која је одобрила истраживање

_____

4.1.2. Да ли подаци укључују личне податке учесника у истраживању? Да Не

Ако је одговор да, наведите на који начин сте осигурали поверљивост и сигурност информација везаних за испитанике:

а) Подаци нису у отвореном приступу

б) Подаци су анонимизирани

ц) Остало, навести шта

_____

_____

5. Доступност података

5.1. Подаци ће бити

а) јавно доступни

б) доступни само уском кругу истраживача у одређеној научној области

ц) затворени

Ако су подаци доступни само уском кругу истраживача, навести под којим условима могу да их користе:

_____

_____

Ако су подаци доступни само уском кругу истраживача, навести на који начин могу приступити подацима:

_____

_____

5.4. Навести лиценцу под којом ће прикупљени подаци бити архивирани.

_____

6. Улоге и одговорност

6.1. Навести име и презиме и мејл адресу власника (аутора) података

_____

6.2. Навести име и презиме и мејл адресу особе која одржава матрицу с подацима

_____

6.3. Навести име и презиме и мејл адресу особе која омогућује приступ подацима другим истраживачима

_____