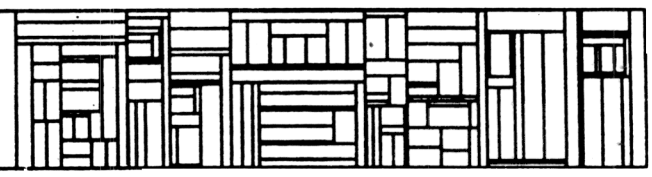
ISSN 0105-8517

EVENT STRUCTURE SEMANTICS FOR CCS AND RELATED LANGUAGES

Glynn Winskel

DAIMI PB-159
April 1983

Computer Science Department
AARHUS UNIVERSITY
Ny Munkegade – DK 8000 Aarhus C – DENMARK
Telephone: 06 – 12 83 55



PB-159

G. Winskel: Event Structure Semantics for CCS and Related Languages

EVENT STRUCTURE SEMANTICS FOR CCS AND RELATED LANGUAGES

Glynn Winskel

Abstract

We give denotational semantics to a wide range of parallel programming languages based on the idea of Milner's CCS [M1], that processes communicate by events of mutual synchronisation. Processes are denoted by labelled event structures. Event structures represent concurrency rather directly as in net theory [NT] The semantics does not simulate concurrency by non-deterministic interleaving.

We first define a category $\mathbb{E}$ of event structures ([NPW1, 2], [W]) appropriate to synchronised communication. The category bears a natural relation to a subcategory of trees through an interleaving functor; so results transfer to trees neatly. Then we introduce the concept of a synchronisation algebra (S.A.) on labels by adopting an idea of Milner [M2]. An S.A. specifies how two processes synchronise via labels on their events. From each S.A., L, we derive a category $\mathbb{E}_L$ of labelled event structures with natural operations for composing labelled event structures. In particular the parallel composition L is derived from the product in $\mathbb{E}$. We obtain semantics for a class of CCS-like languages by varying the S.A. Synchronisation algebras are very general so the class is very broad, handling synchrony and asynchrony in a common framework.

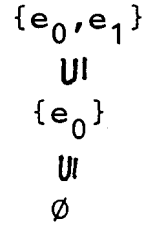
As a corollary we get an event structure semantics for CCS. When interleaved our semantics is Milner's synchronisation/communication tree semantics [M1]. However our semantics distinguishes more terms as it reflects concurrency. Event structure semantics is at a rather basic level of abstraction but should support all abstract notions of equivalence (see [M1] for examples), including those which take concurrency into account.

TABLE OF CONTENTS

0. INTRODUCTION

We consider languages which are related to Robin Milner's "Calculus of Communicating Systems" - CCS, described in [M1]. The most important feature of the languages is the form of parallel composition. The idea is that two processes communicate by events of mutual synchronisation which we illustrate by a simple example. (The reader familiar with [M1] is warned that our approach is not quite the same as Milner's; we do not serialise, or interleave, event occurrences. We promise a neat connection with Milner's synchronisation trees later.)

Consider a simple reading machine M capable of performing only two events e_0 - the event of accepting a coin - and e_1 - the event of delivering an item. By event we mean what others might call an event occurrence, so the machine is really quite short-lived; it accepts one coin and delivers one item. Naturally it only delivers the item after accepting a coin. We can represent the machine as all the sets of events it can have performed up to various stages. We call each set a configuration. Ordered by inclusion the configurations are:

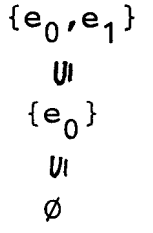


Initially M has performed no events of interest, the configuration $\emptyset$; then it can perform e_0 to realise the configuration $\{e_0\}$ and afterwards e_1 to realise the configuration $\{e_0,e_1\}$. Notice how the machine's behaviour in time is reflected by the inclusion relation on configurations; configurations of events which have occurred later include those which have occurred earlier. Such diagrams can be simplified by using the "covering" relation. In a partial order $\sqsubseteq$ one point x is covered by another y if x and y are distinct and no point can be inserted in between. Formally,

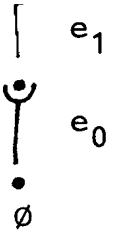
x is covered by y is written $x \prec y$ and defined by:

$$x \prec y \text{ iff } x \sqsubset_{\neq} y \;\&\; \forall z.x \sqsubseteq z \sqsubseteq y \Rightarrow (x=z \text{ or } x=y).$$

For the above partial order of inclusion, if one point covers another it just means one action event has occurred so we can draw

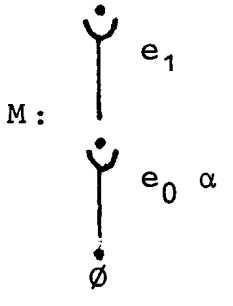


equivalently as



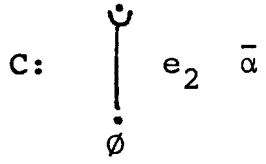
specifying what the extra event occurs at each covering.

To be of use the machine will be set in an environment consisting perhaps of other machines and possible customers. To the environment the events e_0, e_1 are not generally of interest in themselves. Rather it is their nature, what kinds of event they are, that determines how, for instance, a customer interacts with the machine. The machine M performs two kinds of events, accepting a coin, abbreviated to α, and delivering an item abbreviated to ι. We label e_0 by α and e_1 by ι to indicate their kind, so:



Imagine our single machine in use. It relates to customers by accepting a coin from them. At the very least a customer should be able to perform an event of inserting a coin. This kind of event is, in a sense, complementary to accepting a coin so we

label it by $\bar{\alpha}$. A typical customer C is modelled by:

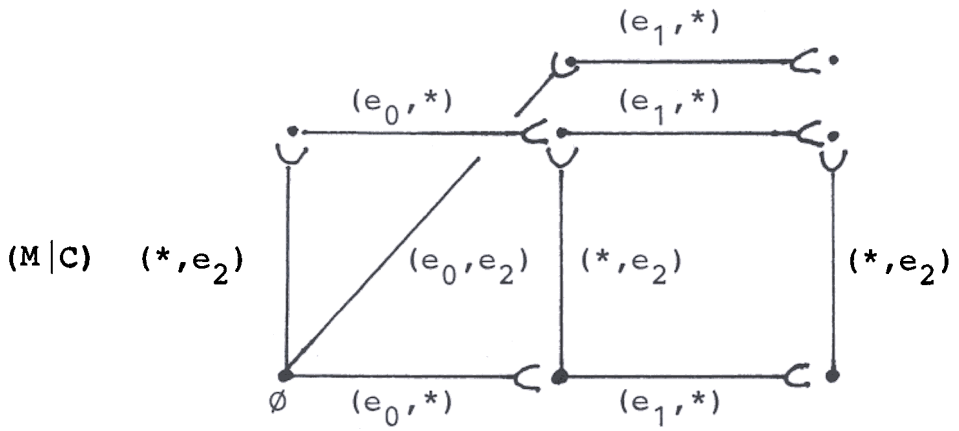


Recall this is really just an abbreviation for $\{e_2\}$ where e_2 is an event of kind $\bar{\alpha}$. A customer can do one event of the kind insert a coin.

Now M can accept a coin from its environment and C can insert a coin to his. In particular, when they are set together M can accept a coin from C. This produces a new kind of event, an event of synchronisation between M and C, which we label by τ. In the world of just customers that can only insert coins and machines that can only accept coins and deliver - we would not expect this new event to synchronise further. In a more varied world it might. Of course, the synchronised event need not occur; quite possibly M could accept a coin from elsewhere, perhaps from another customer, just as C could spend his coin differently. How are we to model this parallel composition of M and C?

Firstly it is natural to take the synchronisation event as a combination of the event e_0 of M, accepting a coin, and the event e_2 of C, inserting a coin. Name the synchronised event by a pair (e_0,e_2) because to M the event looks like e_0 and to C the event looks like e_2. As explained we label it by τ. What about a name for the event where M accepts a coin from something in its environment other than C? To M the event looks like e_0 while to C, who is only sensitive to having his coin accepted, it is invisible. We introduce *, a sort of undefined, and name the event $(e_0,*)$. Clearly, it is the same kind of event as e_0 so we label it the same by α. Similarly there is an event $(*,e_2)$ labelled $\bar{\alpha}$ corresponding to C inserting his coin into something other than M and an event $(e_1,*)$ labelled ι.

What form do the configurations of the parallel composition of M and C take? Suppose first M and C do not synchronise. Then M can deliver an item $(e_1,*)$ only after accepting a coin $(e_0,*)$ and both events are independent of C inserting a coin, $(*,e_2)$. All these events are performed with the environment and not with each other. Alternatively together they synchronise to perform the event (e_0,e_2) whereupon M can do $(e_1,*)$. It is not possible for $(e_0,*)$ and (e_0,e_2) to occur together. This informal argument should convince the reader that the parallel composition $(M|C)$ of M and C has the following configurations



There are several points to note about this diagram. Notice that intuitively the events $(e_0,*)$ and $(*,e_2)$ (and similarly $(e_1,*)$, $(*,e_2)$) are concurrent in that they can occur independently, and this fact is reflected by the little commuting square  .

Notice too there are obvious projections from the parallel composition $(M|C)$ back to the component processes M and C; for example the configuration $\{(e_0,e_2),(e_1,*)\}$ in $(M|C)$ projects to the configuration $\{e_0,e_1\}$ of M and to the configuration $\{e_2\}$ of C. This is natural and expresses the intuition that the behaviour of a compound process should be consistent with the behaviours of its processes. Interestingly we shall derive parallel composition from a product, in a category suitable for synchronised communication, thus giving mathematical leverage to the idea of projecting down to a subprocess.

The category will have event structures as its objects; an event structure consists of a pair, a set of events and a set of configurations, satisfying suitable axioms. Processes will be denoted by labelled event structures where the labels specify the kinds of events. In the machine-customer example it is intuitively clear how two events of certain kinds may or may not combine to form synchronised events. But of course it can all be done more abstractly. We just need a general way to say when and how pairs of labelled events can combine to form synchronised events and what labels such combinations carry. We shall do this by using synchronisation algebras on labels. The idea is to have a binary composition operation, $\bullet$, on a set of labels. When a pair of events of which two labels do not synchronise we make the composition of the labels give 0. For example we would make $\alpha \bullet \alpha = 0$ and $\alpha \bullet \iota = 0$ for our machine and customer. When two labelled events can combine we make the labelled compositions give the new kind of the synchronised event e.g. in our example $\alpha \bullet \bar{\alpha} = \tau$. Our machine-customer example also makes clear that there may be some asynchrony in the parallel composition. In fact there, every event of M and C could occur asynchronously in the parallel composition; every event of M need not be synchronised with an event of C and vice versa, reflected by all those events of the form $(e_0,*)$, $(e_1,*)$ and $(*,e_2)$ in the parallel composition. To allow asynchrony we introduce another constraint * into the algebra. Then for example $\alpha \bullet * = \alpha$ shows an event of kind α can occur asynchronously in a parallel composition and that its new kind in the parallel composition is still the same, viz. α. Our machine-customer example would have this synchronisation algebra:

$\bullet$	*	α	$\bar{\alpha}$	ι	τ	0
*	*	α	$\bar{\alpha}$	ι	τ	0
α	α	0	τ	0	0	0
$\bar{\alpha}$	$\bar{\alpha}$	τ	0	·	·	·
ι	ι	0	·	(0 elsewhere)		
τ	τ	0				
0	0					

The parallel composition (M|C) consists of events determined by the synchronisation algebra and configurations which are subsets of these events which "project down" to configurations of M and C.

This gives a rough idea of how we shall model the parallel composition of two processes. Of course we shall model other operations on processes and need techniques for defining infinite event structures recursively. Then we can give denotational semantics to a range of such languages of which CCS is typical. Of course we also want methods for relating our semantics to others especially Milner's. The details follow.

. EVENT STRUCTURES

Processes are modelled by event structures. An event structure consists of a set of possible event occurrences together with a family of configurations; a configuration is a set of events which occur by some stage in the process, possibly after infinite time. To define operations on event structures neatly we modify the definition of [NPW1, 2] so that an event can occur in several incompatible ways. The definition is motivated further in proposition 1.8.

Notation Let F be a family of subsets of a set E. Let $X \subseteq F$. We write $X\uparrow^F$ for $\exists y \in F \forall x \in X. x \subseteq y$ and say X is compatible. When $x,y \in F$ we write $x\uparrow^F y$ for $\{x,y\}\uparrow^F$.

1.1 Definition An event structure is a pair (E,F), where E is a set of events and $F \subseteq P(E)$ is a family of configurations, which is:

coherent $\forall X \subseteq F.(\forall x,y \in X, x\uparrow^F y) \Rightarrow \bigcup X \in F$

stable $\forall X \subseteq F. X \neq \emptyset \ \& \ X\uparrow^F \Rightarrow \bigcap X \in F$

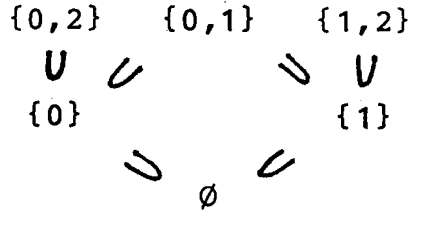
(iii coincidence-free $\forall x \in F \ \forall e,e' \in x. e \neq e' \Rightarrow \exists y \in F. y \subseteq x \ \&$
$((e \in y \ \& \ e' \notin y) \text{ or } (e \notin y \ \& \ e' \in y))$

finitary $\forall x \in F \ \forall e \in x \ \exists y \in F. e \in y \ \& \ y \subseteq x \ \& \ |y| < \infty$

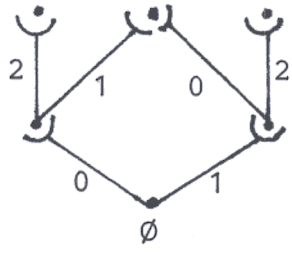
In addition, we say an event structure is full when it satisfies

$\forall e \in E \ \exists x \in F. e \in x$ (i.e. $E = \bigcup F$).

1.2 Example Let $E = \{0,1,2\}$ and F be



or equivalently



where $\prec$ is the covering-relation representing the occurrence of one event. Then (E,F) is an event structure. The events 0 and 1 are concurrent, neither depends on the occurrence or non-occurrence of the other to occur (see [NPW1, 2] and [NT]). The event 2 can occur in two incompatible ways, either through event 0 having occurred or event 1 having occurred. This possibility makes event structures of 1.1 easier to work with than those of [NPW1, 2].

1.3 Example "A ticking clock". Let Ω consist of events ω and configurations the sets $\emptyset,\{0\},\{0,1\},\ldots,\{0,\ldots,n\},\ldots\omega$. Then Ω is an event structure which models a clock ticking $0,1,2,\ldots$.

1.4 Example Coincidence-freeness. Let $E = \{0,1\}$ and $F = \{\emptyset,\{0,1\}\}$. Then (E,F) is not an event structure. It is not coincidence-free. The "events" 0 and 1 are coincident in that together they behave like a single event with respect to F.

1.5 Example Finite causes. Let $E = \omega \cup \{\infty\}$ and $F = \{\emptyset,\{1\},\ldots,\{1,2,\ldots,n\}, \omega,\omega\cup\{\infty\}\}$. Then (E,F) is an event structure which is not finitary. The event ∞ can only occur after the finite set of events ω. Nor is the event structure $(E,P(\omega)\cup \{\omega \cup \{\infty\}\})$ finitary. Such processes are unnatural in computer science because they require an infinite set of events to occur within a finite time.

1.6 Example Fullness. The event structure $(\{e\},\{\emptyset,\{e\}\})$ is full while the event structure $(\{e\},\{\emptyset\})$ is not full. For convenience we do not assume all event structures are full. Clearly any event structure (E,F) determines a full event structure (UF,F) with the same configurations. With trivial modifications all our results hold with the assumption of fullness.

The next proposition motivates the axioms of 1.1. It shows that event structures possess an intrinsic causal dependency relation local to each configuration. The stability axiom ensures that when an event is in some configuration its occurrence has depended on a unique set of events. The set on which the event depends will be finite because of the finitary axiom and the dependency relation

will be a partial order because of coincidence freeness. The ways in which events can occur correspond to complete primes of configurations ordered by inclusion; they form a subbasis making the domain of configurations prime algebraic [NPW1, 2].

1.7 Definition Let $(D,\sqsubseteq)$ be a partial order. Let $p\in D$. Say p is a complete prime iff for all $X\subseteq D$ when the lub $\bigsqcup X$ exists and $p \sqsubseteq \bigsqcup X$ then $p \sqsubseteq x$ for some $x\in X$. Say D is prime algebraic iff

$$\forall x\in D.x = \bigsqcup\{p\sqsubseteq x \quad p \text{ is a complete prime}\}$$

1.8 Proposition Let E be a set and $F\subseteq P(E)$. Then
(i) (E,F) is coherent according to 1.1 iff $(F,\subseteq)$ is a coherent cpo such that for all $X\subseteq F$ if the lub of X exists it is $\bigcup X$. (Thus $\emptyset\in F$.)

For $x\in F$ define the causal dependency relation $\leq_x$ on x by $e\leq_x e' \iff \forall y\in F.y\subseteq x \Rightarrow (e'\in y \Rightarrow e\in y)$ and for $e\in x$ define $[e]_x = \{e'\in x \mid e'\leq_x e\}$. Then $[e]_x=\bigcap\{z\in F \mid e\in z\subseteq x\}$, and we have

(ii If (E,F) is coherent then
(E,F) is stable according to 1.1 iff $\forall x\in F\forall e\in x.[e]_x\in F$
and $\forall x,y\in F\forall e\in x\cap y.x\uparrow^F y\Rightarrow[e]_x=[e]_y$

(iii (E,F) is coincidence-free according to 1.1 iff
$\leq_x$ is a partial order for all $x\in F$.

(iv) If (E,F) is stable then
(E,F) is finitary according to 1.1 iff $\forall x\in F\forall e\in x. \quad |[e]_x| < \infty$.

Suppose (E,F) is coherent and stable. Then $(F,\subseteq)$ is a coherent prime algebraic partial order [NPW1, 2]; the complete primes are of the form $[e]_x$ for $x\in F$ and $e\in x$. Further (E,F) is finitary iff each isolated element of the domain $(F,\subseteq)$ dominates only a finite number of elements.

Proof Let E be a set and $F \subseteq \mathcal{P}(E)$ as above

is obvious.

These two facts follow from the definitions of $\leq_x$ and $[e]_x$:
For $x \in F$ the relation $\leq_x$ defined above is a preorder on x and for $e \in x$ we have $[e]_x = \bigwedge\{z \in F \mid e \in z \subseteq x\}$, a more workable characterisation of $[e]_x$ than its definition.

(ii) Assume (E,F) is coherent.
"=>" Suppose (E,F) is stable. Let $e \in x$ and $x \in F$. Then as $\{y \in F \mid e \in y \subseteq x\}\uparrow^F$ we have $[e]_x = \bigwedge\{y \in F \mid e \in y \subseteq x\} \in F$. Let $x,y \in F$ and $x \uparrow^F y$ and $e \in x \cap y$. Then $[e]_x$, $[e]_y \in F$ and $[e]_x \uparrow^F [e]_y$ so $e \in [e]_x \cap [e]_y \subseteq x$ with $[e]_x \cap [e]_y \in F$. Thus $[e]_x \subseteq [e]_y$, and similarly $[e]_y \subseteq [e]_x$. Therefore $[e]_x = [e]_y$ as required.
"<=" Suppose $\forall x \in F \forall e \in x.[e]_x \in F$ and $\forall x,y \in F \forall e \in x \cap y.x \uparrow^F y \Rightarrow [e]_x = [e]_y$. Let $\emptyset \neq X \subseteq F$ and $X\uparrow^F$. Choose $x \in X$. Let $e \in \bigwedge X$. Then $[e]_x = [e]_y$ for all $y \in X$. Thus $\bigwedge X = \bigcup_{e \in \bigwedge X} [e]_x$. Now by coherence $\bigwedge X \in F$. As required, (E,F) is stable.

(iii) Follows directly from the definitions of $\leq_x$ and coincidence-freeness.

(iv) Assume (E,F) is stable.
"=>" Suppose (E,F) is finitary. Let $e \in x$ and $x \in F$. Then for some finite $z \in F$ we have $e \in z \subseteq x$. By the characterisation of $[e]_x$ it must also be finite.
"<=" Let $e \in x \in F$. Then as (E,F) is stable $[e]_x \in F$ and clearly $e \in [e]_x \subseteq x$. Thus if $[e]_x$ is finite for all $x \in F$ and $e \in x$ we get that (E,F) is finitary.

Assume (E,F) is coherent and stable.
Let $y \in F$ and $e \in y$. Then as (E,F) is stable $[e]_y \in F$. We show $[e]_y$ is a complete prime. Let $X \subseteq F$ and $X\uparrow^F$. Suppose $[e]_y \subseteq \bigcup X$. Then for some $x \in X$ we have $e \in x$. Also as $[e]_y \uparrow^F x$ we have $e \in [e]_y \cap x \subseteq y$ with $[e]_y \cap x \in F$. Thus by the characterisation of $[e]_y$ we have $[e]_y \subseteq [e]_y \cap x$ so $[e]_y \subseteq x$. Thus $[e]_y$ is a complete prime.

Clearly for $x \in F$ we have $x = \bigcup_{e \in x} [e]_x$. Thus each element of F is the l.u.b. of the complete primes it dominates. This means $(F, \subseteq)$ is a prime algebraic po. It is obviously coherent.

Suppose (E, F) is also finitary. Let x be an isolated element of $(F, \subseteq)$. Take S to be the directed set of all finite unions of complete primes below x. Then $x = \bigcup S$ and as x is isolated $x \subseteq s$ for some $s \in S$. Thus x is a finite union of finite sets and so finite. Conversely as complete primes are isolated assuming isolated elements are finite implies that (E, F) is finitary. This means (E, F) is finitary iff each isolated element dominates only a finite number of elements. ∎

As a corollary to 1.8 (ii) we can relate the stability axiom of 1.1 to the concept of stable function due to Gérard Berry (see [B] and [BC]). It is thus axiom (ii) of 1.1 derives its name.

1.9 Corollary Let E be a set and $F \subseteq P(E)$ satisfy the coherence axiom. Let $\mathbb{O}$ be the two element cpo $\bot \sqsubseteq T$. For each $e \in E$ define $\chi_e : (F, \subseteq) \to \mathbb{O}$ by $\chi_e(x) = T$ if $e \in x$, $\bot$ otherwise. Then (E, F) is stable according to 1.1 iff for all $e \in E$ the function χ_e is stable in the sense of Berry [B].

Proof Recall the definition of a stable function. Let A, B be cpos. Let $f : A \to B$ be continuous. Then $f : A \to B$ is stable iff $\forall x \in A \forall y \in B\ y \sqsubseteq f(x)\ \exists M(f,x,y) \in A$ such that $\forall z \sqsubseteq x. y \sqsubseteq f(z) \Leftrightarrow M(f,x,y) \sqsubseteq z$. Hence $M(f,x,y)$ is the least element z less than x such that $y \sqsubseteq f(z)$. Clearly each χ_e above is continuous as (E, F) satisfies the coherence axiom. If (E, F) is stable in addition then take $M(\chi_e, x, T) = [e]_x$ to show χ_e is a stable function. Conversely supposing each χ_e is stable if $e \in x \in F$ we have $M(\chi_e, x, T) = \bigcap \{z \in F \mid e \in z \subseteq y\} = [e]_x$ so $[e]_x \in F$, while if $e \in x \cap y$ for $x \uparrow y$ in F we have $[e]_x = M(\chi_e, x \cup y, T) = [e]_x$. Then by 1.8 (ii) we have (E, F) is stable. ∎

1.10 Example Let (E,F) be the event structure of example 1.2. Let $x = \{0,2\}$ and $y = \{1,2\}$. Then $[2]_x = x$ and $[2]_y = y$ correspond to the two ways the event 2 can occur.

Proposition 1.8 suggests a subclass of event structures for which each event can occur and always causally depends on the same set of events, no matter in what configuration it occurs; so then events correspond to complete primes.

1.11 Definition Let (E,F) be an event structure. Say $(E,F$ is prime iff it is full and $\forall x,y\in F \forall e\in x\cap y.[e]_x = [e]_y$.

For prime event structures the local causal dependency relations ($\leqq_x$ for configurations x) are restrictions of one global causal dependency ($\leqq$) and incompatibility of configurations stems from a pairwise incompatibility, or conflict ($\#$), between events. In accord with intuitions the configurations are then precisely the left-closed consistent subsets (w.r.t. $\leqq$ and $\#$).

1.12 Definition Let $(E,\leqq,\#)$ be a set E with partial order $\leqq$ and binary symmetric relation $\#$. Define the left-closed consistent subsets of E by $x\in L(E,\leqq,\#)$ iff $x\subseteq E$

$$\& \ \forall e,e'.e'\leqq e\in x \Rightarrow e'\in x \quad \text{(left-closed)}$$
$$\& \ \forall e,e'\in x.\neg(e\#e') \quad \text{(consistent)}$$

1.13 Proposition Let (E,F) be a prime event structure. Define the relations $\leqq$ (called the causal dependency relation) and $\#$ (called the conflict relation) on E by

$$e'\leqq e \text{ iff } \forall x\in F.e\in x \Rightarrow e'\in x$$
$$e\#e' \text{ iff } \forall x\in F.e\in x \Rightarrow e'\notin x$$

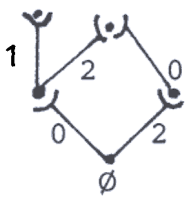
Then $\leqq$ is a partial order s.t. $[e]=_{def}\{e'\in E \mid e'\leqq e\}$ is finite for all $e\in E$ and $\#$ is a binary irreflexive symmetric relation s.t. $e\#e'\leqq e'' \Rightarrow e\#e''$ for all $e,e',e''\in E$. Further the configurations F are precisely the left closed consistent subsets $L(E,\leqq,\#)$.

Conversely, suppose $(E,\leqq,\#)$ consists of a partial order $\leqq$ and binary symmetric relation $\#$ s.t. $|[e]|<\infty$ and $e\#e'\leqq e'' \Rightarrow e\#e''$ for all e,e',e''. Then $(E,\mathcal{L}(E,\leqq,\#))$ is a prime event structure.

Proof Let (E,F) be a prime event structure. Take $\leqq$ and $\#$ as defined above. From 1.8 clearly they satisfy the properties stated above and any configuration is a left closed consistent subset w.r.t. $\leqq$ and $\#$. Also any left-closed consistent subset is a configuration by the coherence of (E,F).

Let $(E,\leqq,\#)$ consist of a p.o. $\leqq$ and symmetric relation $\#$ s.t. $|[e]|<\infty$ and $e\#e'\leqq e'' \Rightarrow e\#e''$ for events e,e',e''. Then it is easily verified that $(E,\mathcal{L}(E,\leqq,\#))$ is a prime event structure. ∎

1.14 Example We show the configurations of a prime event structure alongside its causal dependency $\leqq$ and conflict relation $\#$. Its events are $\{0,1,2\}$.



Consequently prime event structures are in 1-1 correspondence with structures $(E,\leqq,\#)$ which consist of a set of events with causal dependency and conflict relations satisfying simple axioms. They give a simple, intuitive model of concurrent processes related to net theory in [NPW1, 2] and [W]. In fact any event structure of 1.1 determines a prime event structure with an isomorphic domain of configurations by taking the complete primes as the new events.

1.15 Definition Let (E,F) be an event structure. Define $Pr(E,F)$ to consist of events $P = \{[e]_x \mid e\in x\in F\}$ and configurations F_p where

$$\Leftrightarrow \exists x\in F.\ z = \{[e]_x \quad e\in x\}$$

1.16 Proposition Let (E,F) be an event structure.
Then $Pr(E,F)$ is a prime event structure; its events are the complete primes P of $(F,\subseteq)$, its causal dependency relation is $\subseteq \upharpoonright P$ and its conflict relation is $\#^F \upharpoonright P$.

There is an isomorphism $(F,\subseteq) \cong (F_p,\subseteq)$ where F_p are the configuration of $Pr(E,F)$; it is given by $x \longmapsto \{[e]_x \mid e \in x\}$ with inverse $y \longmapsto \bigcup y$.

Proof Let (E,F) be an event structure. Take $P = \{[e]_x \mid e \in x \in F\}$ - then P is the set of complete primes of $(F,\subseteq)$ by proposition 1.8. Take $\leqq = \subseteq \upharpoonright P$ and $\# = \#^F \upharpoonright P$ as above. Certainly $(P, L(P,\leqq,\#))$ is a prime event structure. For any configuration x of $Pr(E,F)$ we have $x \in L(P,\leqq,\#)$. Conversely if $y \in L(P,\leqq,\#)$ then by coherence $\bigcup y \in F$. But then $y = \{p \subseteq \bigcup y \mid p \text{ is a complete prime}\}$. (The inclusion "$\subseteq$" is obvious. Suppose $p \in$ r.h.s. Then $p \subseteq p' \in y$ as p is a complete prime, which as y is left-closed means $p \in y$.) Thus $y = \{[e]_{\bigcup y} \mid e \in \bigcup y\}$ and y is a configuration of $Pr(E,F)$.

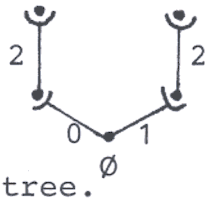
isomorphism follows directly from prime algebraicity. ∎

We work with more general event structures because it is difficult to define parallel composition directly on prime event structures; for prime event structures events correspond to the ways they can occur so to compose them in parallel we must duplicate as many copies of an event as there are ways introduced for it to occur. In the more general class we avoid a messy inductive naming of events, and can "tap out" prime event structures by the construction Pr.

Trees are another simple kind of event structure.

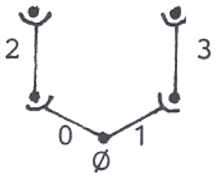
1.17 Definition An event structure (E,F) is a pre-tree iff $\forall x,y \in F.\, x \uparrow^F y \Rightarrow x \subseteq y$ or $y \subseteq x$. A tree is an event structure which is prime and a pre-tree.

1.18 Example The event structure with events {0,1,2} and configurations



is a pre-tree but not a tree.

The event structure with events {0,1,2,3} and configurations



is a tree, with configurations clearly order isomorphic to those of the pre-tree.

The reader may check that the configurations of a pre-tree are isomorphic to sequences of events ordered by extension - so configurations correspond to partial and maximal branches - and that for a tree the events correspond to arcs. For this reason we shall often write a tree as (A,B) consisting of events A - for "arcs" - and configurations B - for "branches". By insisting a tree is prime we have "abstracted away" from the events of which it is built. (This is justified formally when morphisms are introduced; then the tree and pre-tree above will not be isomorphic.)

To sum up we have a class of event structures which includes trees and those event structures of [NPW1, 2] which satisfy a simple finiteness restriction.

With an eye to possible generalisations we note: The coherence axiom is rather strong, too strong for the event structures of 1.1 to model processes such as "fair merge" in a natural way; not all infinite configurations would correspond to a possible infinite behaviour of a fair merge. Perhaps there is an appropriate weaker substitute for which much of the following work still goes

through. One advantage of the coherence axiom, however, is that it allows a smooth connection with Petri nets via the work of [NPW1,2]. The stability axiom would go if one wished to model processes which had an event which could be caused in several compatible ways - see [KP] and [W] for examples; then I expect complete irreducibles would play a similar role to complete primes here. The axioms in 1.1 are like those for a topology. Possibly they can be modified to model continuous processes but, of course, then the finiteness axiom should be dropped.

Those familiar with [KP], [BC], or [W] may wonder why we do not work with event structures $(E,\vdash,\#)$ where E is a set of events, $\vdash \subseteq \mathcal{P}(E)\times E$ is an enabling relations and # is a conflict relation. The main reason is that our morphisms will only be interested in events and configurations, not the exact nature of $\vdash$ and #. Besides the complete primes (the $[e]_x$'s) give us an enabling relation, a rather special one because in a configuration an event is enabled in a unique way, a property unfortunately called "deterministic" in [BC]. (Note incidentally that because of example 1.2 configurations and events are a bit more general than those of "deterministic" event structures $(E,\vdash,\#)$ with a <u>binary</u> conflict relation.)

2. A "CPO" OF EVENT STRUCTURES

By restricting the configurations of an event structure (E,F) to those inside a subset E' of E a new event structure is formed.

2.1 Definition Let (E,F) be an event structure. Let $E' \subseteq E$ Define the restriction $(E,F)\restriction E'$ to be (E',F') where $F' = \{x \in F \mid x \subseteq E'\}$.

2.2 Lemma The restriction $(E,F)\restriction E'$ above is an event structure

Proof All the properties (i) - (iv) of 1.1 required for $(E,F)\restriction E'$ to be an event structure follow directly from the corresponding properties of (E,F). ∎

Such restriction accompanies an idea of substructure - the relation $\trianglelefteq$ below.

2.3 Definition Let (E_0,F_0), (E_1,F_1) be event structures

$(E_0,F_0) \trianglelefteq (E_1,F_1)$ iff $E_0 \subseteq E_1$

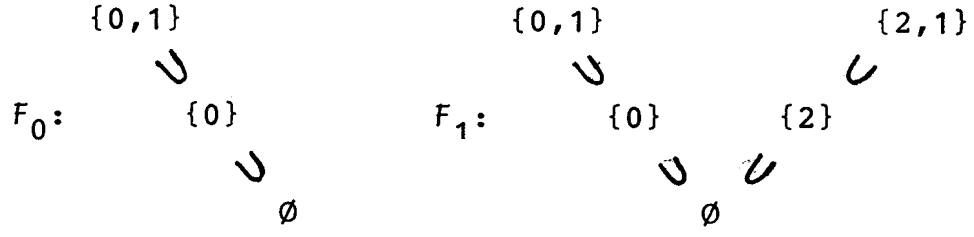
and $F_0 \subseteq F_1$

and $\forall x \subseteq E_0 . x \in F_1 \Rightarrow x \in F_0$.

2.4 Lemma Let (E_0,F_0), (E_1,F_1) be event structures. Then $(E_0,F_0) \trianglelefteq (E_1,F_1)$ iff $E_0 \subseteq E_1$ and $(E_0,F_0) = (E_1,F_1)\restriction E_0$.

Proof Directly from the definitions. ∎

2.5 Example Let (E_0,F_0), (E_1,F_1) be event structures with events $E_0 = \{0,1\}$, $E_1 = \{0,1,2\}$ and configurations as shown:



In (E_0,F_0) the event 1 can only occur after the event 0. In (E_1,F_1) the event 1 can occur in two ways, either after event 0 or after event 2. So (E_1,F_1) introduces a new way for the event 1 to occur even though $(E_0,F_0) \trianglelefteq (E_1,F_1)$.

The relation $\trianglelefteq$ specifies the sense in which one event structure approximates another. Our semantics for recursively defined processes is based on the relation $\trianglelefteq$. Event structures ordered by $\trianglelefteq$ almost form a cpo. The ordering is not a cpo merely because event structures form a class and not a set. (The same kind of situation occurs in [S] and [BC].)

2.6 Theorem (i) The relation $\trianglelefteq$ is a partial order on event structures with least event structure $(\emptyset,\{\emptyset\})$. Let $(E_0,F_0) \trianglelefteq \ldots \trianglelefteq (E_n,F_n) \trianglelefteq \ldots$ be an ω-chain of event structures. Then it has a lub with respect to $\trianglelefteq$. The lub is (E,F) where $E = \bigcup_{n\in\omega} E_n$ and

$$x \in F \text{ iff } (x \subseteq E \ \&\ (\forall n\in\omega.x_n \in F_n \ \&\ x = \bigcup_{n\in\omega} x_n,,$$

in which

$$x_n =_{def} \bigcup\{z\in F_n \mid z\subseteq x\}$$

(ii) Let A be a set. Define $\mathbb{E}_A$ to be the set of event structures (E,F) with $E \subseteq A$. Then $(\mathbb{E}_A,\trianglelefteq)$ is a c.p.o. with bottom element $(\emptyset,\{\emptyset\})$ and l.u.b.s of chains given as above in

Proof (i) Let $(E_0,F_0)\trianglelefteq\ldots\trianglelefteq(E_n,F_n)\trianglelefteq\ldots$ be an ω-chain of event structures. Take (E,F) as defined above. As above, for $x\subseteq E$ we take $x_n = \bigcup\{z\in F_n \mid z\subseteq x\}$. Note for $x\subseteq E$ we have $x_n\subseteq x_m$ if $n\leq m$. Firstly we check that (E,F) is an event structure.

Coherent Suppose $X\subseteq F$ and $\forall x,y\in X.x\uparrow^F y$. If $x,y\in X$ then $x,y\subseteq z$ for some $z\in F$. Thus $x_n,y_n\subseteq z_n$ where $x_n,y_n,z_n\in F_n$ for $n\in\omega$. Consequently $\{x_n \mid x\in X\}$ is pairwise compatible in $(F_n,\subseteq)$. Thus $\bigcup_{x\in X} x_n\in F_n$.

We show $(\bigcup X)_n = \bigcup_{x\in X} x_n$; then clearly $(\bigcup X)_n\in F_n$ and

$$\bigcup\{x_n \mid x\in X \ \&\ n\in\omega\} = \bigcup_n (\bigcup X)_n$$

so $\bigcup X\in F$ as required. The inclusion $\bigcup_{x\in X} x_n\subseteq(\bigcup X)_n$ is obvious. To show the converse inclusion suppose $e\in(\bigcup X)_n$. Then as (E_n,F_n) is finitary for some finite $z\in F_n$ we have $e\in z\subseteq\bigcup X$. For each $e'\in z$ there is some $x\in X$ with $e'\in x$. Thus as $x\in F$ there is some $m\in\omega$ for which $e'\in x_m$. However as z is finite we can choose some m, uniformly, so that $z\subseteq\bigcup_{x\in X} x_m$. Now by the definition of $\trianglelefteq$, $z\in F_m$ and, as above, $\bigcup_{x\in X} x_m\in F_m$. Thus for each $x\in X$, $z\uparrow^{F_m} x_m$ so $z\cap x_m\in F_m$. This implies $z\cap x_m\in F_n$ by the definition of $\trianglelefteq$. Therefore $e\in z = z\cap\bigcup_{x\in X} x_m = \bigcup_{x\in X}(x_m\cap z)\subseteq\bigcup_{x\in X} x_n$. We have shown the required converse inclusion $(\bigcup X)_n\subseteq\bigcup_{x\in X} x_n$.

Stable Suppose $\emptyset\neq X\subseteq F$ & $X\uparrow^F$. Clearly $(\bigcap X)_n = \bigcup_{x\in X} x_n$ for $n\in\omega$. As $\{x_n \mid x\in X\}\uparrow^{F_n}$ we get $(\bigcap X)_n\in F_n$. Noting $\bigcap X = \bigcap_{x\in X}\bigcup_{n\in\omega} x_n = \bigcup_n(\bigcap X)_n$ we have $\bigcap X\in F$.

Coincidence-free Let $e,e'\in x\in F$ and $e\neq e'$. Then $e,e'\in x_n$ for some $n\in\omega$. As (E_n,F_n) is coincidence-free $e\in z\Longleftrightarrow e'\notin z$ for some $z\in F_n$ s.t. $z\subseteq x_n$. But it is easily checked that $z\in F$ so (E,F) is coincidence-free.

Finitary Suppose $e\in x\in F$. Then $e\in x_n$ for some n. Then $e\in z\subseteq x_n$ for some finite $z\in F_n$. This gives $z\in F$ and $e\in z\subseteq x$ as required.

Therefore (E,F) is an event structure. We now show it is the lub of the chain $(E_0,F_0)\trianglelefteq\ldots\trianglelefteq(E_n,F_n)\trianglelefteq\ldots$.

For (E,F) to be an upper bound we require $(E_n,F_n)\trianglelefteq(E,F)$ for all $n\in\omega$. Clearly $E_n\subseteq E$. From the definition of $\trianglelefteq$ it follows that $F_n\subseteq F$. Suppose $x\subseteq E_n$ & $x\in F$. Then $x = \bigcup_{m\in\omega} x_m$ with $x_m\in F_m$ for all $m\in\omega$. However, $x_m\subseteq E_n$ so by the definition of $\trianglelefteq$ we get $x_m\in F_n$ for each $m\in\omega$. As $(E_n,F_n$ is coherent and $x_0\subseteq\ldots\subseteq x_m\subseteq\ldots$ is a chain in F_n we have $x=\bigcup_m x_m\in F_n$. Thus $(E_n,F_n)\trianglelefteq(E,F)$ for all $n\in\omega$ so (E,F) is an upper bound of the chain.

To see (E,F) is the least upper bound of the chain, let (E',F') be an event structure which is an upper bound of the chain. Then certainly $E\subseteq E'$ and $\bigcup_{n\in\omega} F_n\subseteq F'$. Let $x\in F$. Then the chain $x_0\subseteq\ldots\subseteq x_n\subseteq\ldots$ is included in F'. As (E',F') is coherent $x=\bigcup_n x_n\in F'$. Thus $F\subseteq F'$. Suppose now $y\in F'$ & $y\subseteq E$. We have $y_n = \bigcup\{z\in F_n \mid z\subseteq y\}\in F'$ so as $(E_n,F_n)\trianglelefteq(E',F')$ we get $y_n\in F_n$. Clearly $\bigcup_n y_n\subseteq y$. To show the converse inclusion, take $e\in y$. Then as (E',F') is finitary $e\in z\subseteq y$ for some finite $z\in F'$. As z is finite $z\subseteq E_n$ for some n. But $(E_n,F_n)\trianglelefteq(E',F')$ so $z\in F_n$. Evidently $z\subseteq y_n$. Thus $e\in\bigcup_n y_n$. Therefore $y=\bigcup_n y_n$. So $(E,F)\trianglelefteq(E',F')$ and (E,F) is the least upper bound of the chain of event structures.

ii Obvious, by (i). ∎

The naturalness of $\trianglelefteq$ and its lubs is easier to see on prime event structures because then the way an event can occur stays fixed in a $\trianglelefteq$-chain.

2.7 Proposition (i) Let (E_0,F_0), (E_1,F_1) be prime event structures with causal dependency relations $\leq_0$, $\leq_1$, and conflict relations $\#_0,\#_1$. For $e\in E_i$ write $[e]_i =_{\text{def}} \{e'\in E_i \mid e'\leq_i e\}$, for $i=0,1$. Then
$(E_0,F_0)\trianglelefteq(E_1,F_1)$ iff $E_0\subseteq E_1$ & $(\forall e\in E_0.[e]_0 = [e]_1)$ & $\#_0 = \#_1\restriction E_0$.
iff $F_0\subseteq F_1$, & $\forall x\in F_1.x\cap E_0\in F_0$.

Let $(E_0,F_0)\trianglelefteq\ldots\trianglelefteq(E_n,F_n)\trianglelefteq\ldots$ be an ω-chain of prime event structures. Let (E_n,F_n) have causal dependency and conflict relations $\leq_n,\#_n$.

Then the lub of the chain is (E,F) where $E = \bigcup_{n\in\omega} E_n$ and $x\in F$ iff $\forall n\in\omega.x\cap E_n\in F_n$; the lub (E,F) is prime with causal dependency relation $\leq = \bigcup_{n\in\omega}\leq_n$ and conflict relation $\# = \bigcup_{n\in\omega}\#_n$.

(ii) Let (A_0,B_0), (A_1,B_1) be trees. Then $(A_0,B_0)\trianglelefteq(A_1,B_1)$ iff $B_0\subseteq B_1$. Let $(A_0,B_0)\trianglelefteq\ldots\trianglelefteq(A_n,B_n)\trianglelefteq\ldots$ be an ω-chain of trees. Then its lub is a tree (A,B) where $A = \bigcup_{n\in\omega} A_n$ and $x\in B$ iff $\forall n\in\omega.x\cap A_n\in B_n$.

Proof (i) Let (E_0,F_0), (E_1,F_1) be prime event structures with causal dependency and conflict relations as above. Suppose $(E_0,F_0)\trianglelefteq(E_1,F_1)$. Then $E_0\subseteq E_1$. Let $e\in E_0$. Then $[e]_0\in F_1$ so $[e]_1\subseteq[e]_0$. However then by the definition of $\trianglelefteq$, $e\in[e]_1,\in F_0$. This implies $[e]_0 = [e]_1$. Now, let $e,e'\in E_0$. Then $e\#_0 e' \iff [e]_0 \not\uparrow^0 [e']_0 \iff [e]_0\cup[e']_0\in F_0 \iff [e]_1\cup[e']_1\in F_1 \iff [e]_1\not\uparrow[e']_1 \iff e\#_1 e'$ using properties of $\trianglelefteq$. Thus $(E_0,F_0)\trianglelefteq(E_1,F_1) \Rightarrow E_0\subseteq E_1$, & $(\forall e\in E_0.[e]_0 = [e]_1)$ & $\#_0 = \#_1\lceil E_0$. To show the converse implication assume the r.h.s.. Then $E_0\subseteq E_1$ and $x\in L(E_0,\leq_0,\#_0) \iff x\subseteq E_0$ & $x\in L(E_1,\trianglelefteq_1,\#_1)$. This gives $(E_0,F_0)\trianglelefteq(E_1,F_1)$.

We show the second equivalent. Suppose $(E_0,F_0)\trianglelefteq(E_1,F_1)$. Then $F_0\subseteq F_1$. Also if $x\in F_1$, then $x\in L(E_1,\trianglelefteq_1,\#_1)$. Thus by the first equivalent $x\cap E_0\in L(E_0,\leq_0,\#_0) = F_0$. Conversely suppose $F_0\subseteq F_1$ and $\forall x\in F_1.x\cap E_0\in F_0$. Then by fullness $E_0\subseteq E_1$. Also if $x\subseteq E_0$ and $x\in F_1$, then $x = x\cap E_0\in F_0$. This gives $(E_0,F_0)\trianglelefteq(E_1,F_1)$.

Now let $(E_0,F_0)\trianglelefteq\ldots\trianglelefteq(E_n,F_n)\trianglelefteq\ldots$ be a chain of prime event structures so (E_n,F_n) is associated with the relations $\leq_n,\#_n$. Take $\leq = \bigcup_n\leq_n$ and $\# = \bigcup_n\#_n$. Define (E,F) by $E = \bigcup_n E_n$ and $F = L(E,\leq,\#)$; it is a prime event structure. Then by the definition of $\trianglelefteq$ as $F_n = L(E_n,\leq,\#_n)$ we get $(E_n,F_n)\trianglelefteq(E,F)$. This means (E,F) is an upper bound of the chain. By Theorem 3.6 the lub of the chain is (E,F') for some set of configurations F'. Thus $(E,F')\trianglelefteq(E,F)$. However by the definition of $\trianglelefteq$ we then have $F' = F$. Thus (E,F) is the lub. Clearly $x\in L(E,\leq,\#) = F$ iff $x\cap E_n\in L(E_n,\leq_n,\#_n) = F_n$ for all n.

(ii) Let (A_0,B_0), (A_1,B_1) be trees. Obviously $(A_0,B_0)\trianglelefteq(A_1,B_1)$ implies $B_0\subseteq B_1$. Suppose conversely that $B_0\subseteq B_1$. Then $A_0\subseteq A_1$ by fullness.

Let $a \in A_0$. Then $[a]_1 \subseteq [a]_0$ where $[a]_i$ is the smallest configuration in B_i containing a. Let $a' \in [a]_0$ and $a' \neq a$. Then $a \notin [a']_0 \in B_1$ and $[a']_0 \uparrow^{B_1} [a]_1$ so $[a']_0 \subseteq [a]_1$ as (A_1, B_1) is a tree. Thus $[a]_0 \subseteq [a]_1$. Therefore $[a]_0 = [a]_1$.

Remembering for trees that compatible configurations are comparable we get $[a]_0 \uparrow^{B_0} [a']_0$ iff $[a]_1 \uparrow^{B_1} [a']_1$ for $a, a' \in A_0$. Thus $a \#_0 a' \iff a \#_1 a'$ for $a, a' \in A_0$, where $\#_0, \#_1$ are the conflict relations of (A_0, B_0), (A_1, B_1) respectively.

By i) we have $(A_0, B_0) \trianglelefteq (A_1, B_1$ ∎

The recursive definition of a process will be associated with an operation continuous w.r.t. $\trianglelefteq$. The denotation of the recursively defined process will be the least fixed point of the operation.

2.8 Definition Let op be an n-ary operation on the class of event structures.

Say op is monotonic iff when for event structures we have $E_1 \trianglelefteq E_1', \ldots E_n \trianglelefteq E_n'$ then

$$op(E_1, \ldots, E_n \trianglelefteq op(E_1', \ldots, E_n').$$

op is continuous iff for all countable chains

$$\begin{array}{l} E_{11} \trianglelefteq E_{12} \trianglelefteq \ldots \trianglelefteq E_{1i} \trianglelefteq \ldots \\ \quad \cdot \\ \quad \cdot \\ \quad \cdot \\ E_{n1} \trianglelefteq E_{n2} \trianglelefteq \ldots \trianglelefteq E_{ni} \trianglelefteq \ldots \end{array}$$

we have $op(\bigsqcup_i E_{1i}, \ldots \bigsqcup_i E_{ni}) = \bigsqcup_i op(E_{1i}, \ldots E_{ni})$

where $\bigsqcup$ denotes the lub with respect to $\trianglelefteq$.

As is wellknown (see[S]) an operation is continuous iff it is continuous in each argument separately. Given this the following lemma provides simple necessary and sufficient conditions for an ope-

ration to be continuous on event structures; it should be monotonic and act continuously on the component sets of events ordered by inclusion.

2.9 Lemma Let op be a unary operation on $\mathbb{E}$. Then op is continuous iff (i) op is monotonic and (ii) if $(E_0,F_0) \trianglelefteq \ldots \trianglelefteq (E_n,F_n) \trianglelefteq \ldots$ is a chain in $\mathbb{E}$ then each event of $op(\bigsqcup_n (E_n,F_n))$ is an event of $\bigsqcup_n op(E_n,F_n)$.

Proof "⇒" obvious.

"⇐" Suppose (i) and (ii) above. Let $(E_0,F_0) \trianglelefteq \ldots \trianglelefteq (E_n,F_n) \trianglelefteq \ldots$. Then as op is monotonic the event structure $\bigsqcup_n op(E_n,F_n)$ exists and $\bigsqcup_n op(E_n,F_n) \trianglelefteq op(\bigsqcup_n (E_n,F_n))$. Now by (ii), $\bigsqcup_n op(E_n,F_n)$ and $op(\bigsqcup_n (E_n,F_n))$ have the same events. From the definition of $\trianglelefteq$ they have the same configurations. Thus $\bigsqcup_n op(E_n,F_n) = op(\bigsqcup_n (E_n,F_n))$. Therefore op is continuous. ∎

As an example we show how the operation Pr is continuous. Recall from 1.15,16 that from an event structure Pr constructs a prime event structure with an isomorphic domain of configurations. This will mean Pr commutes with the operation of defining event structures recursively.

2.10 Theorem The operation Pr defined in 1.15 is $\trianglelefteq$-continuous.

Proof We use lemma 2.9.

We first show Pr is monotonic w.r.t. $\trianglelefteq$. Suppose $(E_0,F_0) \trianglelefteq (E_1,F_1)$ for event structures (E_0,F_0) and (E_1,F_1). We require $Pr(E_0,F_0) \trianglelefteq Pr(E_1,F_1)$. Let $Pr(E_i,F_i) = (P_i,\Pi_i)$ for $i = 0,1$, so P_i the set of complete primes of $(F_i,\subseteq)$. Suppose $p_0 \in P_0$. As $(E_0,F_0) \trianglelefteq (E_1,F_1)$ we have $p_0 \in F_1$. Assume $Y \subseteq F, Y\uparrow^{F_1}$ and $p_0 \subseteq \bigcup Y$. Then $p_0 = (\bigcup Y) \cap p_0 = \bigcup_{y \in Y} y \cap p_0$ where $y \cap p_0 \in F_1$ and $y \cap p_0 \subseteq E_0$ so $y \cap p_0 \in F_0$ for each $y \in Y$. Thus as p_0 is a complete prime of F_0, $p_0 \subseteq y$ for some $y \in Y$. Therefore p_0 is a complete prime of F_1. Consequently $P_0 \subseteq P_1$. Now from the definitions of

Π_0 and Π_1 - see 1.10 - as $(E_0,F_0) \trianglelefteq (E_1,F_1$ we get $z\in\Pi_1, \& z\subseteq P_0 \Leftrightarrow z\in\Pi_0$. This means $Pr(E_0,F_0) \trianglelefteq Pr(E_1,F_1)$.

We now show Pr is continuous on event sets. Let $(E_0,F_0) \trianglelefteq \ldots \trianglelefteq (E_n,F_n) \trianglelefteq \ldots$ be a chain of event structures with lub(E,F). Let p be an event of $Pr(E,F)$, so p is a complete prime of $(F,\subseteq)$. To use Lemma 2.9 we require that p is an event of $Pr(E_n,F_n)$ for some n. However p is finite so $p\subseteq E_n$ for some n (we have $E = \bigcup_n E_n$). Now $p\in F_n$ as $(E_n F_n) \trianglelefteq (E,F)$. Also as p is a complete prime of (E,F) it must be a complete prime of (E_n,F_n). Thus p is an event of $Pr(E_n,F_n)$ as required.

Applying 2.9 gives Pr is continuous. ∎

As a corollary we can give another characterisation of the lub of an ω-chain of event structures ordered by $\trianglelefteq$. The lub is simple to define for prime event structures in terms of their causal dependency and conflict relations. So, we first convert an ω-chain of arbitrary event structures to a chain of prime event structures using Pr, find its lub and then image back using Lemma 1.16 which shows the isomorphism between configurations of an event structure and its image under Pr.

2.11 Corollary Let $(E_0,F_0) \trianglelefteq \ldots \trianglelefteq (E_n, \ _n) \trianglelefteq \ldots$ be an ω-chain of event structures. It has lub (E,F) where $E = \bigcup_{n\in\omega} E_n$ and

$$x\in F \Leftrightarrow \exists z\in L(P,\subseteq,\natural).\ x = \bigcup z \quad \text{where}$$

$$P = \bigcup_{n\in\omega} \{[e]_x \mid e\in x\in F_n\}$$

and $\natural = \bigcup_{n\in\omega} \natural^{F_n}$

Proof From the $\trianglelefteq$-continuity of Pr we have $Pr(E,F) = \bigsqcup_{n\in\omega} Pr(E_n,F_n$

From Lemma 1.16 we know F is the image of the configurations of $Pr(E,F)$ under $\bigcup$. ∎

3. A CATEGORY OF EVENT STRUCTURES

We define a rather basic class of morphisms on event structures. They are partial functions between event-sets which respect events and configurations. An event is imagined to synchronise with its image event whenever this is defined. One notable example of morphism will be a projection from the compound process of an event structure put in parallel with another back to the original event structure - see the product of event structures 3.4. Refer to the appendix for our treatment of partial functions - we use * to represent undefined - and a formal definition of the $\hat{}$ operator which extends a function on events to a function on subsets.

3.1 Definition Let (E_0,F_0), $(E_1,F_1$ be event structures. A (partially synchronous) morhpism θ: $(E_0,F_0)\rightarrow(E_1,F_1)$ is a partial function $\theta:E_0\rightarrow_* E_1$ such that

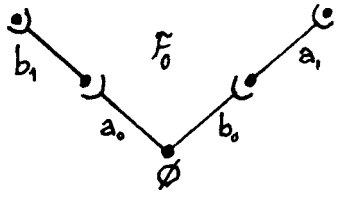
(i $\forall x\in F_0.\hat{\theta}(x)\in F_1$

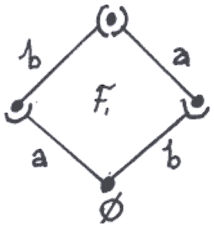
and (ii) $\forall x\in F_0 \forall e,e'\in x.\theta(e)=\theta(e' \neq * \Rightarrow e=e'$.

A morphism θ is synchronous iff θ is a total function

Note that condition (ii) above says no two distinct events are together synchronised with a common image event. Notice if we have $(E_0,F_0)\trianglelefteq(E_1,F_1)$, for two event structures (E_0,F_0) and (E_1,F_1), then the inclusion map i: $E_0\hookrightarrow E_1$ is a morphism, in fact a rather special one, so $\hat{i}$ is a rigid embedding in the sense of Kahn and Plotkin [KP].

3.2 Example Let (E_0,F_0), (E_1,F_1) be event structures with $E_0 = \{a_0,a_1,b_0,b_1\}$, $E_1 = \{a,b\}$ and configurations





Then θ defined so $\theta(a_0)=\theta(a_1)=a$ and $\theta(b_0)=\theta(b_1)=b$ is a (synchronous) morphism. (Incidentally this morphism, although total, cannot be induced on event structures by a net morphism on Petri nets - see [NT], [NPW1, 2].)

It is easy to check that the morphisms defined above give a category of event structures with the usual composition of partial functions and identity morphisms the identity functions on sets of events.

3.3 Definition Define $\mathbb{E}$ to consist of objects event structures and morphisms as defined in 3.1 with composition that of partial functions $\underline{\underline{\text{Set}}}_*$ defined in the appendix. Define $\mathbb{E}_{\text{syn}}$ to consist of event structures and synchronous morphisms with the usual composition of functions.

3.4 Proposition Both $\mathbb{E}$ and $\mathbb{E}_{\text{syn}}$ are categories with identity morphisms the identity functions. We have $\mathbb{E}_{\text{syn}}$ is a proper subcategory of $\mathbb{E}$. Both categories $\mathbb{E}$ and $\mathbb{E}_{\text{syn}}$ have the null event structure $(\emptyset,\{\emptyset\})$ as initial object. The null event structure is also the terminal object of $\mathbb{E}$ (but not $\mathbb{E}_{\text{syn}}$).

Let θ: $(E_0,F_0) \rightarrow (E_1,F_1)$ be a morphism in E. Then θ is an isomorphism iff θ is a total 1-1 and onto function such that $x \in F_0 \Leftrightarrow \hat{\theta}(x) \in F_1$.

θ is a monomorphism in $\mathbb{E}$ ($\mathbb{E}_{\text{syn}}$ iff θ is a monomorphism in $\underline{\underline{\text{Set}}}_*$ ($\underline{\underline{\text{Set}}}$).
θ is an epimorphism in $\mathbb{E}$ ($\mathbb{E}_{\text{syn}}$ iff θ is an epimorphism in $\underline{\underline{\text{Set}}}_*$ ($\underline{\underline{\text{Set}}}$).

The category E has products and coproducts characterised, to within isomorphism, by the following constructions. They provide a basis for defining, and proving relations between, different semantics of CCS and its variants.

The parallel composition of two processes will be denoted by a restriction of the product. The product corresponds to a very loose synchronisation discipline between processes; any event of one may or may not synchronise with an event of the other. A configuration of the product of two event structures E_0 and E_1 may contain events of synchronisation between E_0 and E_1 and must project to configurations of E_0 and E_1 by natural projection morphisms.

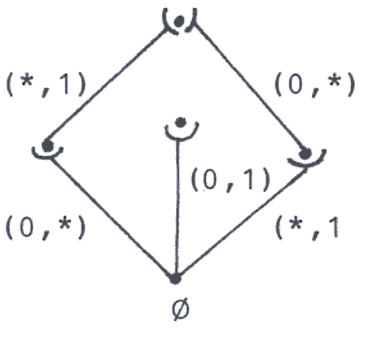
3.5 Definition (Partially synchronous) product

Let (E_0,F_0), (E_1,F_1) be event structures. Define their product $(E_0,F_0) \times (E_1,F_1)$ to be (E,F) where $E=E_0 \overset{\times}{*} E_1$, the product in $\underline{\underline{\mathrm{Set}}}_*$ with projections π_0,π_1, and F is given by:

$$
\begin{aligned}
& \text{iff } x \subseteq E_0 \overset{\times}{*} E_1 \\
\& \ (a) \quad & \hat{\pi}_0(x) \in F_0 \ \& \ \hat{\pi}_1(x) \in F_1 \\
\& \ (b) \quad & \forall e,e' \in x.\, \pi_0(e)=\pi_0(e') \neq * \text{ or } \pi_1(e)=\pi_1(e') \neq * \Rightarrow e=e' \\
\& \ (c) \quad & \forall e,e' \in x.\, e \neq e' \Rightarrow \exists y \subseteq x.\, \hat{\pi}_0(y) \in F_0 \ \& \ \hat{\pi}_1(y) \in F_1 \ \& \\
& \qquad\qquad (e \in y \ \& \ e' \notin y) \text{ or } (e \notin y \ \& \ e' \in y)) \\
\& \ (d) \quad & \forall e \in x \exists y \subseteq x.\, \hat{\pi}_0(y) \in F_0 \ \& \ \hat{\pi}_1(y) \in F_1 \ \& \ e \in y \ \& \ |y| < \infty
\end{aligned}
$$

Note how (a) and (b) express that the projections are morphisms while (c) and (d) say the structure (E,F) is coincidence-free and finitary respectively.

3.6 Example (product) Let (E_0,F_0) be $(\{0\},\{\emptyset,\{0\}\})$ and (E_1,F_1) be $(\{1\},\{\emptyset,\{1\}\})$. Then their product $(E_0,F_0) \times (E_1,F_1$ consists of events $E_0 \overset{\times}{*} E_1 = \{(0,*),(0,1),(*,1)\}$ with configurations



Intuitively (E_0,F_0), (E_1,F_1) can proceed asynchronously or alternatively communicate through synchronising events 0 and 1 to form the event $(0,1)$ (c.f. $(\alpha\,\mathrm{NIL}\,|\,\bar{\alpha}\mathrm{NIL})$ in Milner's CCS - see §5).

It is useful to also define a product in the category $\mathbb{E}_{syn}$ o event structures with synchronous morphisms, induced by just total functions.

3.7 Synchronous product Let (E_0,F_0), (E_1,F_1) be event structures. Define their synchronous proudct $(E_0,F_0) \otimes (E_1,F_1)$ to be (E,F) where $E = E_0 \times E_1$, the product in Set with projections π_0, π_1, and F is given by

$$
\begin{aligned}
x \in F \quad \text{iff} \quad & x \subseteq E_0 \times E_1 \\
& \& \text{(a)} \quad \hat{\pi}_0(x) \in F_0 \ \& \ \hat{\pi}_1(x) \in F_1 \\
& \& \text{(b)} \quad \forall e,e' \in x.\, \pi_0(e) = \pi_0(e') \text{ or } \pi_1(e) = \pi_1(e') \Rightarrow e = e' \\
& \& \text{(c)} \quad \forall e,e' \in x.\, e \neq e' \Rightarrow \exists y \subseteq x.\, \hat{\pi}_0(y) \in F_0 \ \& \ \hat{\pi}_1(y) \in F_1 \ \& \\
& \qquad\qquad ((e \in y \ \& \ e' \notin y) \text{ or } (e \notin y \ \& \ e' \in y)) \\
& \& \text{(d)} \quad \forall e \in x \exists y \subseteq x.\, \hat{\pi}_0(y) \in F_0 \ \& \ \hat{\pi}_1(y) \in F_1 \ \& \ e \in y \ \& \ |y| < \infty
\end{aligned}
$$

Note that the synchronous product is the restriction of the product to the events $E_0 \times E_1 \subseteq E_0 \overset{\times}{*} E_1$ i.e. $(E_0,F_0) \otimes (E_1,F_1) = (E_0,F_0) \times (E_1,F_1) \upharpoonright E_0 \times E_1$.

Notice how in the above definition an event of E_0 must synchronise with some event of E_1 if it is to occur. We use the synchronous product to define an interleaving operator on event structures. The operator synchronises occurrences of events one at a time with the ticking of a clock.

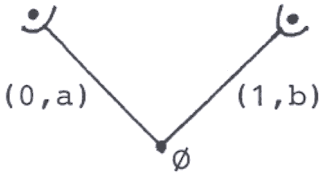
3.8 Proposition Let Ω be the event structure of example 1.3 - the "ticking clock". Let (E,F) be an event structure. The synchronous product $(E,F)\otimes\Omega$ is a pre-tree which consists of events $E\times\omega$ and configurations all finite or infinite sequences $\{(e_0,0),(e_1,1),\ldots,(e_n,n)\ \ldots\}$ such that $e_i = e_j \Rightarrow i = j$ and $\{e_0,e_1,\ldots,e_n\} \in F$ for all i,j, n at which the sequence is defined.

Proof Obviously from the definition of $\otimes$ the events of $(E,F)\otimes\Omega$ are $E\times\omega$. Let x be a configuration of $(E,F)\otimes\Omega$. Then $x\subseteq E\times\omega$ and by conditions (a) and (b) x is a "sequence", either null or of the form $\{(e_0,0),(e_1,1),\ldots,(e_n,n),\ldots\}$. Condition (c) now implies $\{e_0,\ldots,e_n\}\in F$ for any n at which e_n exists - if n marks the end of the sequence use (a), otherwise separate (e_n,n) and $(e_{n+1},n+1)$ using (c). Clearly any sequence satisfying the conditions stated in the proposition is a configuration of $(E,F)\otimes\Omega$. ∎

A simpler construction is that of coproduct which is essentially the disjoint union of event structures.

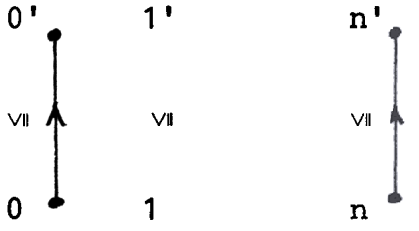
3.9 Definition Coproduct
Let (E_0,F_0), (E_1,F_1) be event structures. Define their coproduct $(E_0,F_0)+(E_1,F_1)$ to be (E,F) where $E=\{0\}\times E_0\cup\{1\}\times E_1$ and $F=\{\{0\}\times x \mid x\in F_0\}\cup\{\{1\}\times x \quad x\in F_1\}$. (Note the evident injections $i_0:E_0\to E$ and $i_1:E_1\to E$.)

3.10 Example (coproduct) Let $(E_0,F_0)=(\{a\},\{\emptyset,\{a\}\})$ a $(E_1,F_1)=(\{b\},\{\emptyset,\{b\}\})$. Then $(E_0,F_0)+(E_1,F_1)$ has events $\{(0,a),(1,b)\}$ and configurations



3.11 Example (the necessity of (c) in definitions 3.5 and 3.7). Let $(E_0,F_0)=(\{0,1\},\{\emptyset,\{0\},\{0,1\}\})$ and $(E_1,F_1)=(\{a,b\},\{\emptyset,\{a\},\{a,b\}\})$. Then without the restriction (c) in 3.5 and 3.7 both "products" would not be coincidence-free. They would have "configuration" $x=\{(0,b),(1,a)\}$ so that $(0,b)<_{\neq x}(1,a)<_{\neq x}(0,b)$ - a non-trivial loop in the local causality relation.

3.12 Example (the necessity of (d) in definitions 3.5 and 3.7) The necessity of (d) is best shown using the representation of [NPW1, 1] - see proposition 1.9. Without (d) the "product" of two (finitary) event structures need not be finitary. Let E_0 consist of events $0,0',1,1',\ldots,n,n',\ldots$ with no conflict, and causal dependency given by the partial order



Let E_1 be an isomorphic partial order with events $\bar{0},\bar{0}',\ldots,\bar{n},\bar{n}'$ Both partial orders determine prime event structures by taking their left closed subsets as configurations. However omitting (d) from the restrictions defining configurations of product would allow the "configuration" consisting of the synchronised events $x=\{(0,\bar{0}'),(\bar{0},1'),(1,\bar{1}'),\ldots,(n,\bar{n}'),(\bar{n},(n+1)'),\ldots\}$ which has an infinite descending chain with respect to the local causality relation $\leq_x$.

Now we verify that the constructions $\times$, $\otimes$, $+$ always give event structures characterising the categorical product, synchronous product and coproduct. To show $\times$ gives an event structure we need a lemma.

3.13 Lemma Let $\theta:E_0 \longrightarrow_* E_1$ be a partial function between sets E_0 and E_1. Let $X \subseteq P(E_0)$. Then if $(\forall e,e' \in \bigcup X.\theta(e)=\theta(e')\neq * \Rightarrow e=e')$ then $\hat{\theta}(\bigcap X)=\bigcap \hat{\theta} X$.

Proof Suppose $\theta(e)=\theta(e')\neq *$ implies $e=e'$ for every $e,e' \in \bigcup X$. Clearly $\hat{\theta}$ is monotonic w.r.t. $\subseteq$ so $\hat{\theta}(\bigcap X) \subseteq \bigcap \hat{\theta} X$. Take $e \in \bigcap \hat{\theta} X$ and $x \in X$. For some $e' \in x$ we have $\theta(e')=e$. Take $y \in X$. Then for some $e_y \in y$ we have $\theta(e_y)=e$. However $e_y, e \in \bigcup X$ and $\theta(e_y)=\theta(e')$. Thus by hypothesis $e_y=e'$. Therefore $e' \in \bigcap X$ so $e \in \hat{\theta}(\bigcap X)$. This establishes the converse inclusion; so $\hat{\theta}(\bigcap X)=\bigcap \hat{\theta} X$ as required. ∎

The following theorem shows the above constructions were already determined to within isomorphism by our choice of morphism. However our rather concrete constructions do give continuous operations on event structures ordered by $\trianglelefteq$, so they can be used in recursive definitions.

3.14 Theorem Let (E_0,F_0), (E_1,F_1) be event structures. Then

(i) $(E_0,F_0)\times(E_1,F_1),\pi_0,\pi_1$ as defined in 3.5 is their categorical product in $\mathbb{E}$.

ii) $(E_0,F_0)\otimes(E_1,F_1),\pi_0,\pi_1$ as defined in 3.7 is their categorical product in $\mathbb{E}_{syn}$.

(iii $(E_0,F_0)+(E_1,F_1)$ as defined in 3.9 is their categorical coproduct in $\mathbb{E}$ and $\mathbb{E}_{syn}$.

Further, each operation $\times$, $\otimes$ and $+$ is continuous w.r.t. $\trianglelefteq$.

Proof i Let (E,F) be $(E_0,F_0)\times(E_1,F_1$ and π_0, π_1 be as defined in 3.5.

Suppose $x\subseteq E_0\overset{x}{*}E_1$ and $e,e'\in x$. We shall say "y is a separating set for e,e' in x" when $y\subseteq x$ & $\hat{\pi}_i(y)\in F_i$ for i=0,1 & ((e$\in$y & e'$\notin$y) or (e$\notin$y & e'$\in$y)).

We first check (E,F) is an event structure.

Coherent Suppose $X\subseteq F$ & $\forall x,y\in X.x\uparrow y$. We require $\bigcup X$ satisfies (a)-(d) of 3.5.

(a) Clearly $\hat{\pi}_i(\bigcup X)=\bigcup\hat{\pi}_i X$. As X is pairwise compatible in F so is $\hat{\pi}_i X$ in F_i. Thus $\hat{\pi}_i(\bigcup X)\in F_i$.

(b) By the pairwise compatibility of X, if $e,e'\in\bigcup X$ and $\pi_i(e)=\pi_i(e')\neq *$ for i=0 or 1 then e=e'.

(c) Suppose $e,e'\in\bigcup X$ and $e\neq e'$. Then $\exists x,y\in X.e\in x$ & $e'\in y$. If either e$\notin$y or e'$\notin$x we have respectively either y or x is a separating set for e,e' in $\bigcup X$. Otherwise e,e'$\in$x or e,e'$\in$y. Then as both x and y satisfy (c) we obtain the required separating set.

(d) is obvious as $e\in\bigcup X$ means $e\in x$ for some $x\in X$ where x satisfies (d).

Stable Suppose $\emptyset \neq X \subseteq F$ & $X\uparrow$. We require X satisfies (a)-(d) of 3.5.

(a) By lemma 3.13, $\hat{\pi}_i(\bigcap X) = \bigcap \hat{\pi}_i X$. But $\bigcap \hat{\pi}_i X \in F_i$ as $\hat{\pi}_i X$ is a compatible set in F_i we have $\hat{\pi}_i(\bigcap X) \in F_i$.

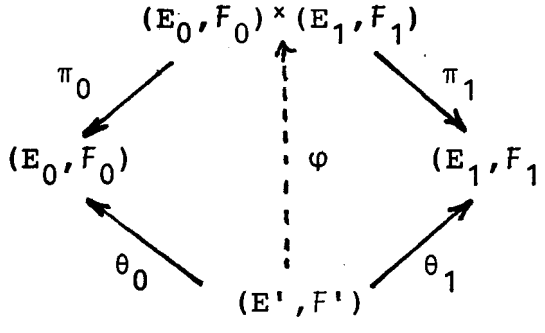
(b) As any $x \in X$ satisfies (b) and $\bigcap X \subseteq x$ certainly $\bigcap X$ satisfies (b).

(c) Suppose $e,e' \in \bigcap X$ and $e \neq e'$. Choose $x \in X$. Because $x \in F$ there is a separating set y for e,e' in x. Take $v = y \cap \bigcap X$. Clearly y, $\bigcap X \subseteq x$ so because (E_i, F_i) is stable, by lemma 3.13 $\hat{\pi}_i(v) = \hat{\pi}_i(y) \cap \hat{\pi}_i(\bigcap X) \in F_i$. This makes v a separating set for e,e' in $\bigcap X$.

(d) is like (c) above.

Coincidence-free Suppose $e,e' \in x \in F$ and $e \neq e'$. As x satisfies (c) there is a separating set y for e,e' in x. We further require $y \in F$. Clearly y satisfies (a), (b). To show y satisfies (c), assume $\varepsilon,\varepsilon' \in y$ & $\varepsilon \neq \varepsilon'$. Take a separating set v for ε,ε' in x. Take $u = v \cap y$. Then, just as in the proof of stability part (c), we get u is a separating set for ε,ε' in x. Property (d) for y follows from property (d) holding for x, using lemma 3.13.

Thus we have shown $(E_0,F_0) \times (E_1,F_1)$ is an event structure. It remains to show that with projections π_0, π_1 it forms the categorical product in $\mathbb{E}$. First note π_0 and π_1 are morphisms by (a), (b) of 3.5. Suppose there are morphisms $\theta_i : (E',F') \to (E_i,F_i)$ in $\mathbb{E}$ for i=0,1. We require a unique morphism φ such that the following diagram commutes:



Take $\varphi=\theta_0 \overset{\times}{*} \theta_1$ i.e. $\varphi(e) = \begin{cases} (\theta_0(e),\theta_1(e) & \text{if } \theta_0(e)\neq * \text{ or } \theta_1(e)\neq * \\ * & \text{otherwise} \end{cases}$

Obviously $\pi_i \circ \varphi = \theta_i$ in $\underline{\underline{\text{Set}}}_*$ for i=0,1 so provided φ is a morphism in $\mathbb{E}$ it is unique so the diagram commutes. To show φ is a morphism we need:

I $\quad \forall x\in F'.\ \hat{\varphi}(x)\in F$

II $\quad \forall x\in F' \forall e,e'\in x.\ \varphi(e)=\varphi(e')\neq * \Rightarrow e=e'$

We prove II first:

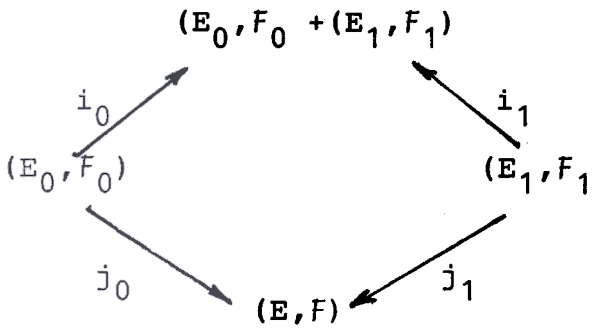
Suppose $e,e'\in x\in F'$. Then if $\varphi(e)=\varphi(e')\neq *$ then $\theta_i(e)=\theta_i(e')\neq *$ for either i=0 or i=1. As each θ_i is a morphism e=e' as required to prove II.

Now we prove I. Let $x\in F'$. We need $\hat{\varphi}(x)$ satisfies (a)-(d) of 3.5 Both (a) and (b) follow from the commutations $\pi_i \circ \varphi = \theta_i$ using the morphism properties of θ_0 and θ_1. To prove (c), suppose $e,e'\in\hat{\varphi}(x)$ and $e\neq e'$. Then $e=\varphi(\varepsilon)$ and $e'=\varphi(\varepsilon')$ for some $\varepsilon,\varepsilon'\in x$. We must have $\varepsilon\neq\varepsilon'$. Thus as (E',F') is coincidence-free we have some $y\in F'$ such that $y\subseteq x$ & $((\varepsilon\in y$ & $\varepsilon'\notin y)$ or $(\varepsilon\notin y$ & $\varepsilon'\in y))$. As we know φ satisfies II above it follows that one and only one of e,e' is in $\hat{\varphi}(y)$. The commutations $\pi_i \circ \varphi = \theta_i$ give $\pi_i\hat{\varphi}(y)\in F_i$. Thus $\hat{\varphi}(y)$ separates e,e' in x. Property (d) follows as (E',F') is finitary.

Thus finally we have shown $(E_0,F_0)\times(E_1,F_1$ is a categorical product in $\mathbb{E}$ with projections π_0,π_1.

(ii) Clearly $(E_0,F_0)\otimes(E_1,F_1)$ is the restriction $(E_0,F_0)\times(E_1,F_1)\restriction E_0\times E_1$. Thus by lemma 2.2 it is an event structure. In this case the projections π_0,π_1 are total so synchronous and the mediating morphism (φ above) stays in the category $\mathbb{E}_{syn}$. This means $(E_0,F_0)\otimes(E_1,F_1)\pi_0,\pi_1$ is a product in $\mathbb{E}_{syn}$.

(iii) It is easily checked that $(E_0,F_0)+(E_1,F_1)$ is an event structure. The injections are clearly (rigid) morphisms. Suppose in $\mathbb{E}$ (or $\mathbb{E}_{syn}$) we had:



Then define θ: $E_0 \dot{\cup} E_1 \to E$ by $\theta(e) = \begin{cases} j_0(e_0) & \text{if } e=(0,e_0)\in 0\times E_0 \\ j_1(e_1) & \text{if } e=(1,e_1)\in 1\times E_1 \end{cases}$

Then θ is the unique morphism in $\mathbb{E}$ ($\mathbb{E}_{syn}$ respectively) such that the diagram commutes. This means $(E_0,F_0)+(E_1,F_1),i_0,i_1$ is a coproduct in $\mathbb{E}$ (and $\mathbb{E}_{syn}$).

Now we show the operations product $\times$, synchronous product $\otimes$ and coproduct $+$ are continuous operations on event structures with respect to $\trianglelefteq$.

Recall an operation is continuous iff it is continuous in each argument separately. If $(E_0,F_0)\trianglelefteq(E_1,F_1)$ and (E,F) are event structures then by inspecting definition 3.5 it is clear that $(E_0,F_0)\times(E,F)=(E_1,F_1)\times(E,F)\upharpoonright E_0^{\times}E$ so $\times$ is monotonic in its first argument. Thus property (i) of lemma 2.9 holds and property (ii) is obvious. Therefore $\times$ is continuous in its first and, by symmetry, its second argument. Therefore $\times$ is continuous. Similarly so are $\otimes$ and $+$. ∎

Similarly one can define infinite products, synchronous products and coproducts - left to the reader.

4. TWO SUBCATEGORIES, PRIME EVENT STRUCTURES AND TREES

Importantly our work transfers over to the two subcategories of $\mathbb{E}$ with objects the prime event structures and trees. In particular this means we can relate event structure semantics to semantics based on trees using interleaving.

4.1 Definition Define $\mathbb{P}$ to be the full subcategory of $\mathbb{E}$ with objects the prime event structures. Define $\mathbb{T}r$ to be the full subcategory of $\mathbb{E}$ with trees as objects.

We characterise morphisms in the two categories prime event structures $\mathbb{P}$ and trees $\mathbb{T}r$.

4.2 Proposition (i) Let (E_0,F_0), (E_1,F_1) be prime event structures with causal dependency relations $\leq_0,\leq_1$ and conflict relations $\#_0,\#_1$. For $e\in E_i$ write $[e]_i =_{def} \{e'\in E_i \mid e'\leq e\}$ for $i = 0,1$. Write $W_i = \#_i \cup 1_{E_i}$. Let $\theta: E_0 \rightarrow_* E_1$ be a partial function. Then θ is a morphism iff $\forall e\in E_0.\theta(e)\neq * \Rightarrow [\theta(e)]_1 \subseteq \hat{\theta}([e]_0)$ and $\forall e,e'\in E_0.\theta(e)\neq * \ \&\ \theta(e')\neq * \ \&\ \theta(e)W_1\theta(e') \Rightarrow eW_0e'$.

(ii) Let (A_0,B_0), (A_1,B_1) be trees. Let $\prec_i$ be the covering relation in $(B_i,\subseteq)$. Write $\preceq_i = \prec_i \cup 1_{B_i}$. Let $f: (B_0,\subseteq) \rightarrow (B_1,\subseteq)$ be a continuous function. Then there is a unique morphism $\theta: (A_0,B_0) \rightarrow (A_1,B_1)$ with $f = \hat{\theta}$ iff $f(\emptyset) = \emptyset$ & $\forall b,b'\in B_0.\ b \prec_0 b' \Rightarrow f(b) \preceq_1 f(b')$.

Proof (i) "=>" Suppose θ is a morphism. Assume $\theta(e)\neq *$ for $e\in E_0$. Then $\hat{\theta}[e]\in F_1$. Therefore $\hat{\theta}[e]$ is leftclosed. Therefore $[\theta(e)]_1 \subseteq \hat{\theta}[e]_0$.

Let $e,e'\in E_0$. Assume $\theta(e),\theta(e')\neq *$ and $\theta(e)W_1\theta(e')$. Suppose $\neg e\#_0e'$. Then $e,e'\in x$ for some $x\in F_0$. As θ is a morphism $\hat{\theta}(x)\in F_1$. Thus $\neg\theta(e)\#_1\theta(e')$. But then as θ is a morphism $e=e'$.

"<==" Suppose θ satisfies the r.h.s. conditions of i above. We require that θ is a morphism.

Let $x \in F_0$. Assume $e_1' \leq_1 e_1 \in \hat{\theta}(x)$. Then $e_1 = \theta(e)$ for some $e \in E_0$. By assumption $e_1' \in [\theta(e)]_1 \subseteq \hat{\theta}[e]_0 \subseteq \hat{\theta}(x)$. Thus $\hat{\theta}(x)$ is left-closed. Assume $e_1, e_1' \in \hat{\theta}(x)$. Suppose $e_1 \#_1 e_1'$. Then $e_1 = \theta(e)$ and $e_1' = \theta(e')$ for some $e, e' \in x$. However then by assumption $e W_0 e'$. This yields a contradiction as neither $e \#_0 e'$ (they are both in x) nor $e = e'$ (as $e_1 \neq e_1'$ by supposition). Thus $\neg(e_1 \#_1 e_1')$. Consequently $\hat{\theta}(x)$ is consistent. Therefore $x \in F_0 \Rightarrow \hat{\theta}(x) \in F_1$.

We also need $e, e' \in x$ & $\theta(e) = \theta(e') \neq * \Rightarrow e = e'$ for θ to be a morphism. Assume $e, e' \in x$ & $\theta(e) = \theta(e') \neq *$. Then by assumption $e W_0 e'$. However as $e, e' \in x$ we have $\neg(e \#_0 e')$ so $e = e'$ as required.

(ii) Let $\theta: (A_0, B_0) \to (A_1, B_1)$ be a morphism between trees. Then as $\hat{\theta}$ is additive it is continuous and $\hat{\theta}(\emptyset) = \emptyset$. If $b \prec_\theta b'$ for $b, b' \in B_0$, then there is a unique event $a \in A_0$ s.t. $a \in b' \setminus b$. Then clearly $\hat{\theta}(b) \prec_1 \hat{\theta}(b')$ if $\theta(a) \neq *$ and $\hat{\theta}(b) = \hat{\theta}(b')$ if $\theta(a) = *$. Conversely given a function $f: B_0 \to B_1$ satisfying the conditions above we define $\theta: A_0 \to_* A_1$ as follows. For $a \in A_0$ there are unique $b, b' \in B_0$ s.t. $b' \setminus b = \{a\}$. Then $b \prec_0 b'$. If $f(b) \prec_1 f(b')$ take $\theta(a)$ to be the unique event in $f(b') \setminus f(b)$. Otherwise $f(b) = f(b')$ so take $\theta(a) = *$. The partial function θ checks to be a morphism so $f = \hat{\theta}$. ∎

The inclusion function $\mathbb{P} \hookrightarrow \mathbb{E}$ has as right adjoint Pr (see 1.16) which to an event structure associates a prime event structure with an isomorphic domain of configurations. Intuitively the operation Pr renames events of a process so each event has a unique causal history. Similarly the inclusion functor $\mathbb{T}r \hookrightarrow \mathbb{E}$ has a right adjoint I which is an interleaving operation defined with the synchronous product $\otimes$ and "ticking clock" of 3.8. These adjunctions determine the form of products and coproducts in $\mathbb{P}$ and $\mathbb{T}r$ (see [Mac]). Both operations Pr and I are ω-continuous so a fixed point semantics based on event structures will image under Pr to a semantics based directly on prime event structures, or under I to one based directly on trees.

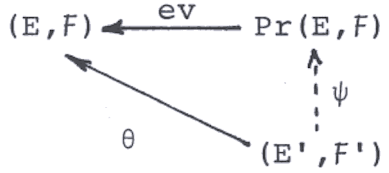
4.3 Theorem Let (E,F) be an event structure.
(i) Define $Pr(E,F)$ to consist of events $P=\{[e]_x \mid e\in x\in F\}$ and configurations F_p where $z\in F_p$ iff $\exists x\in F.z=\{[e]_x \mid e\in x\}$. Then $Pr(E,F)$ is a prime event structure. There is a morphism $ev_{E,F}:Pr(E,F)\to(E,F)$ given by $ev_{E,F}([e]_x)=e$ for $e\in x\in F$. In fact $Pr(E,F),ev_{E,F}$ is cofree over (E,F) i.e. for any morphism $\theta:(E',F')\to(E,F)$ with (E',F') a prime event structure, there is a unique morphism $\psi:(E',F')\to Pr(E,F)$ such that $\theta=ev_{E,F}\psi$.

(ii) Define $I(E,F)=Pr((E,F)\otimes\Omega)$. Then $I(E,F)$ is a tree. There is a morphism $\pi_{E,F}:I(E,F)\to(E,F)$ given by $\pi_{E,F}=\pi_0 ev_{(E,F)\otimes\Omega}$ where $\pi_0:(E,F)\otimes\Omega\to(E,F)$ is the projection morphism. In fact $I(E,F)$, $\pi_{E,F}$ is cofree over (E,F).

Further, both operations Pr and I are $\trianglelefteq$-continuous.

Proof Let (E,F) be an event structure.
(i) By lemma 1.16 $Pr(E,F)$ is a prime event structure. We require that $ev_{E,F}:Pr(E,F)\to(E,F)$ above is a morphism. First we need ev is well-defined as a function $ev:P\to E$, where $P=\{[e]_x \mid e\in x\in F\}$. Suppose $[e]_x=[e']_y$ for $e\in x$ & $x\in F$ and $e'\in y$ & $y\in F$. Then by the coincidence-freeness of (E,F) we have $e=e'$, giving ev well-defined as a (total) function. From the definition if z is a configuration of $Pr(E,F)$ then $z=\{[e]_x \mid e\in x\}$ for some $x\in F$; thus $\hat{ev}(z)=\bigcup z=x\in F$. Let z be a configuration of $Pr(E,F)$ so $p,p'\in z$ and $ev(p)=ev(p')=e$ say. Then $p=p'=[e]_{\bigcup z}$. Thus ev is a morphism.

We show $Pr(E,F)$, $ev_{E,F}$ is cofree over (E,F). Let $\theta:(E',F')\to(E,F)$ be a morphism from a prime event structure (E',F'). We require a unique morphism $\psi:(E',F')\to Pr(E,F)$ s.t. the following diagram commutes:



Define $\psi: E \to P$ by $\psi(e) = \begin{cases} [\theta(e)]_{\hat{\theta}([e])} & \text{if } \theta(e) \neq * \\ * & \text{otherwise} \end{cases}$

where $[e]$ is the smallest configuration of F' containing e (possible because (E',F') is prime).

Let $x \in F'$.

Then $\hat{\psi}(x) = \{[\theta(e)]_{\hat{\theta}([e])} \mid e \in x \ \& \ \theta(e) \neq * \}$

$= \{[e']_{\hat{\theta}(x)} \mid e' \in \hat{\theta}(x)\}$ so $\hat{\psi}(x)$ is a configuration of $Pr(E,F)$. If $e,e' \in x$ and $\psi(e)=\psi(e') \neq *$ then $\theta(e)=\theta(e') \neq *$ so $e=e'$, as θ is a morphism. Thus ψ is a morphism. Clearly $ev\psi = \theta$ so ψ makes the diagram commute.

Let $\emptyset$: $(E',F') \to Pr(E,F)$ be a morphism such that the diagram commutes i.e. $ev\emptyset=\theta$. We require $\emptyset=\psi$. Let $e \in E'$. Firstly note if $\theta(e)=*$ then because ev is a total function we must have $\emptyset(e)=*$ which agrees with ψ. So suppose that $\theta(e) \neq *$. Then $\emptyset(e)$ is a complete prime of $(F,\subseteq)$ s.t $ev(\emptyset(e))=\theta(e)$. Now $\hat{ev}$ is just union so using the assumed commutation we get

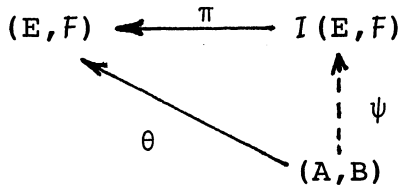
$$\emptyset(e) \subseteq \bigcup \hat{\emptyset}([e]) = \hat{ev}\ \hat{\emptyset}([e]) = \hat{\theta}([e])$$

As $\emptyset(e)$ is a complete prime in $\hat{\theta}([e])$ and $ev(\emptyset(e))=\theta(e)$ we have $\emptyset(e)=[\theta(e)]_{\hat{\theta}([e])}$, i.e. $\emptyset(e)=\psi(e)$.

Consequently ψ is the unique morphism making the diagram commute.

(ii) We have already shown $\otimes$ is an operation on event structures and by (i) so is Pr. Thus $I(E,F) = Pr((E,F)\otimes\Omega)$ is an event structure. Clearly also $\pi_{E,F}: I(E,F) \to (E,F)$ is a morphism.

We show $I(E,F),\pi$ is cofree over (E,F). Let $\theta:(A,B) \to (E,F)$ be a morphism from a tree (A,B). We require a unique morphism $\psi:(A,B) \to I(E,F)$ s.t. the following diagram commutes:



Recall from proposition 3.8 that the configurations of $(E,F)\otimes\Omega$ are sequences $\{(e_0,0),(e_1,1),\ldots,(e_n,n),\ldots\}$ for distinct e_n's s.t. $\{e_0,e_1,\ldots,e_n\}\in F$ for each n at which e_n is defined. It is convenient to write $(e_0,e_1,\ldots,e_n,\ldots)$ for $\{(e_0,0),(e_1,1),\ldots,(e_n,n),\ldots\}$, a configuration of $(E,F)\otimes\Omega$. The complete primes of such configurations are finite non-null sequences $(e_0,e_1,\ldots,e_n$ The map π acts as $\pi((e_0,e_1,\ldots,e_n))=e_n$.

As (A,B) is a tree each event $a\in A$ corresponds 1-1 to a unique finite non-null sequence $(a_0,a_1,\ldots,a_n)$ s.t. $a=a_n$ and $\{a_0\},\ldots,\{a_0,\ldots,a_m\},\ldots,\{a_0,\ldots,a_n\}\in B$. Thus it is sufficient to define ψ on such sequences by the following induction:

If n=0 and $\theta(a_0)\neq *$ define $\psi((a_0)=(\theta(a_0))$ and otherwise *

For n>0, define $\psi((a_0,\ldots,a_{n-1},a_n)) = \begin{cases} \psi((a_0,\ldots,a_{n-1}))^\wedge\theta(a_n \\ \qquad\qquad \text{if } \theta(a_n)\neq * \\ * \text{ otherwise} \end{cases}$

(We use ^ to represent concatenation of a value to the end of a sequence.)

We write ψ for the function determined on A, too. Clearly ψ is the unique partial function s.t. $\pi\psi=\theta$. It is a morphism by the above identifications of configurations as particular kinds of sequences

Finally we note Pr and I are continuous by Theorems 2.10 and 3.14 ∎

We observe some intuitive properties of Pr and I.

4.4 Lemma (i If (E,F) is a prime event structure then $Pr(E,F)\cong(E,F)$.

(ii) Let (E_0,F_0), (E_1,F_1) be event structures. Then $Pr(E_0,F_0) \cong Pr(E_1,F_1) \Leftrightarrow (F_0,\subseteq) \cong (F_1,\subseteq)$.

(iii) Let (E,F) be an event structure and $Pr(E,F)=(P,F_p)$. Then $(F_p,\subseteq) \cong (F,\subseteq)$.

(iv) If (A,B) is a tree then $I(A,B) \cong (A,B)$.

Proof Details are left to the reader

(i) follows because for a prime event structure events correspond to primes.

(ii) follows because $Pr(E_0,F_0)$, $Pr(E_1,F_1)$ are built from the complete primes of $(F_0,\subseteq)$, $(F_1,\subseteq)$ respectively, just using their order theoretic properties.

(iii) is just 1.16.

(iv) Events a of (A,B) correspond to finite non-null sequences $\{(a_0,0),(a_1,1),\ldots,(a_n,n)\}$ s.t. $a=a_n$ and $\{a_0\},\{a_0,a_1\},\ldots,$ $\{a_0,\ldots,a_n\}\in B$. ∎

4.5 Corollary (i) Let (E_0,F_0), $(E_1,F_1)\in \mathbb{P}$. Their product in $\mathbb{P}$ is $Pr((E_0,F_0) \times (E_1,F_1)$. Their coproduct in $\mathbb{P}$ is $(E_0,F_0)+(E_1,F_1)$

(ii) Let (A_0,B_0), $(A_1,B_1)\in \mathbb{T}r$. Their product in $\mathbb{T}r$ is $I((A_0,B_0)\times(A_1,B_1))$. The coproduct in $\mathbb{T}r$ is $(A_0,B_0)+(A_1,B_1$ (Note $\times$ and $+$ stand for product and coproduct in $\mathbb{E}$.)

Proof The right adjoints Pr and I preserve limits [Mac], in particular products giving the form of products in $\mathbb{P}$ and $\mathbb{T}r$ by 4.4 (i), (iv). The inclusion functors are left-adjoints so preserve coproducts. Thus coproducts in $\mathbb{P}$ and $\mathbb{T}r$ coincide with those of $\mathbb{E}$. ∎

Another characterisation of product $\overset{x}{}_{Tr}$ in Tr relates it to Milner's parallel combinator on synchronisation trees [M1]. When labels are introduced his combinator is just a restriction of the product of trees.

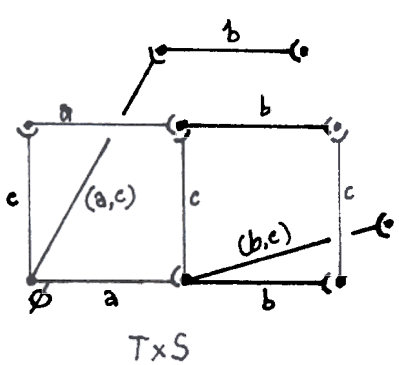
4.6 Definition Let $(E,F)\in\mathbb{E}$ and $\varepsilon\notin E$. Define $\varepsilon\hat{}(E,F)=(E\cup\{\varepsilon\},F_\varepsilon$ where $z\in F_\varepsilon$ iff $z=\emptyset$ or $(\varepsilon\in z \;\&\; z\setminus\{\varepsilon\}\in F)$.

4.7 Proposition Let the trees T, S be coproducts $T=\sum_{a\in A} a\hat{}T_a$ and $S=\sum_{b\in B} b\hat{}S_b$. Then

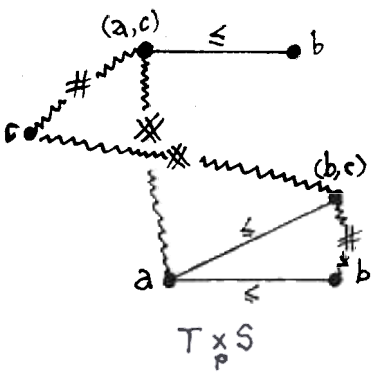
$$T \underset{Tr}{\times} S \cong \sum_{a\in A} (a,*)\hat{}(T_a \underset{Tr}{\times} S) + \sum_{(a,b)\in A\times B} (a,b)\hat{}(T_a \underset{Tr}{\times} S_b) + \sum_{b\in B} (*,b)\hat{}(T \underset{Tr}{\times} S_b)$$

Proof We give the idea. Using proposition 3.8 and 3.5, characterising product in $\mathbb{E}$ one shows the configurations of both the l.h.s. and r.h.s. are isomorphic when ordered by inclusion to sequences of events of T×S, ordered by extension, of the form $(e_0,e_1,\ldots,e_n,\ldots)$ s.t. $\hat{\pi}_0(\{e_0,\ldots,e_n\})$ is a configuration of T and $\hat{\pi}_1(\{e_0,\ldots,e_n\})$ is a configuration of S for all n at which e_n is defined. As both the l.h.s. and r.h.s. are trees, so prime, isomorphism of configurations implies isomorphism of the event structures l.h.s. and r.h.s. ∎

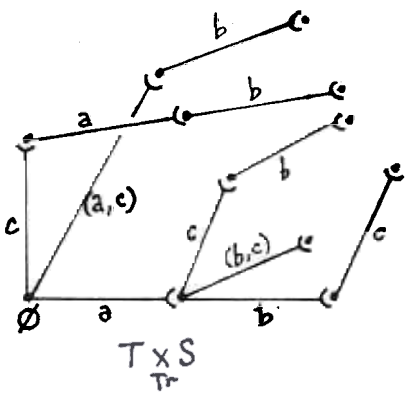
4.8 Example Let a, b, c be distinct events. Let T be the tree $ab(\emptyset,\emptyset)$ and S the tree $c(\emptyset,\emptyset)$. We show their products in $\mathbb{E}$, $\mathbb{P}$ and $\mathbb{T}$r. We label coverings and events to show how they project to T and S.



T×S



$T\times_P S$



$T\times_{Tr} S$

Note how the events of $T\times_P S$ are the complete primes of the configurations of T×S. See how interleaving makes branches out of $\prec$-chains of the original configurations.

5. A SEMANTICS FOR COMMUNICATING PROCESSES

Now we label the events of processes. Possible synchronisations between two processes set in parallel are determined by a synchronisation algebra (S.A.). An S.A. specifies how, depending on their labels, pairs of events are combined to form synchronised events and what labels such combinations carry. We adopt an idea from [M2] and present an S.A. as a binary operation on labels. Unlike [M2] our algebra is not necessarily a monoid (it may not have 1) and has two distinguished constants * and a zero 0.

The constant * still represents undefined, exactly as it does for morphisms and is important for handling asynchrony. No real event is ever labelled *. However when two processes are set in parallel, an event of one process may be left to occur asynchronously, unsynchronised with any event of the other. Then it is enormously convenient to pretend, mathematically, that the event is synchronised with the unreal "event"* labelled by * - just as we did in the product 3.5.

The constant 0 is another ficticious label; no real event is labelled 0. We have $\lambda\cdot\lambda'=0$, for two labels λ,λ', when two events labelled λ and λ' cannot be synchronised. The introduction of 0 saves us from a partial operation on labels.

5.1 Definition A synchronisation algebra (S.A.) is a quadruple $(L,*,0,\cdot)$ where L is a set of labels, containing * and 0 with $L\setminus\{*,0\}\neq\emptyset$ and $\cdot$ is a binary associative, commutative operation on L which satisfies:

(i) $\forall\lambda\in L.\quad \lambda\cdot 0=0$

(ii) $*\cdot *=*$ and $\forall\lambda,\lambda'\in L.\,\lambda\cdot\lambda'=*\Rightarrow\lambda=\lambda'=*$

An S.A. determines a "divides" relation as follows. It says when one label is a divisor, or factor, of another.

5.2 Definition Let $(L,*,0,\cdot)$ be an S.A. For $\alpha,\beta\in L$ define $\alpha \text{ div } \beta \Leftrightarrow (\alpha=\beta \text{ or } \exists\gamma\in L.\alpha\cdot\gamma = \beta)$.

Thus condition (ii) in the definition of an S.A. says * is the unique divisor of *

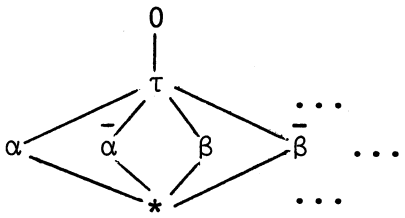
5.3 Lemma Let $(L,*,0,\cdot$ be an S.A. Then

(i) The relation div is reflexive and transitive.
(ii) For $\lambda\in L$, if λdiv* then $\lambda = *$.
(iii) For $\lambda\in L$, if 0divλ then $\lambda = 0$.
(iv) Let $\alpha_0,\alpha_1,\beta_0,\beta_1 \in L$. If $\alpha_0 \text{div} \beta_0$ and $\alpha_1 \text{div} \beta_1$ then $\alpha_0\cdot\alpha_1 \text{div} \beta_0\cdot\beta_1$.

Proof (i) by associativity.
(ii) by property (ii) in the definition of an S.A.
(iii) as 0 is a zero.
(iv) by commutativity and associativity. ∎

5.4 Example The S.A. for CCS
Without value passing
Recall that in CCS [M1] there are three kinds of (non*,0) labels; labels α,β, ..., their complementary labels $\bar{\alpha},\bar{\beta},\ldots$ and the label τ. Only pairs of events with complementary labels can synchronize to produce a τ-labelled event. Thus we get the following S.A. table and division relation for CCS. In this case * behaves like an identity - this is not true in general (see ex. 5.6).

·	*	τ	α	$\bar{\alpha}$	β	...	0
*	*	τ	α	$\bar{\alpha}$	β		0
τ	τ	0	0	0	0		0
α	α	0	0	τ	0		
$\bar{\alpha}$	$\bar{\alpha}$	0	τ	0	0		
β	β	0	0	0	0		
⋮							
0	0	0	.....				



With value passing

Suppose values $v \in V$ are passed during synchronisation. Take labels of the form $*, 0, \alpha v$ (receiving the values v labelled by α), $\bar{\alpha} v$ (sending of value v labelled by α), with an S.A. like above but now with $\bar{\alpha} v$ the complement of αv.

An S.A. determines a category of labelled event structures. Morphisms are event structure morphisms such that the label of the image of an event divides the event's label.

5.5 Definition Let $(L, *, 0, \cdot)$ be an S.A. Define the category $\mathbb{E}_L$ to consist of objects (E, F, l) where $(E, F) \in \mathbb{E}$ and $l: E \to L \setminus \{*, 0\}$, and morphisms $\theta: (E_0, F_0, l_0) \to (E_1, F_1, l_1)$ where $\theta: (E_0, F_0) \to (E_1, F_1)$ is a morphism of $\mathbb{E}$ and $\forall e \in E_0 . l_1 \theta(e) \operatorname{div} l_0(e)$; composition is that of $\mathbb{E}$. Define $\mathbb{P}_L$ and $\mathbb{T}r_L$ to be the full subcategories of labelled prime event structures and trees respectively.

Note in the above definition that the composition $l\theta$ is understood to be in $\underline{\underline{Set}}_*$. If $\theta(e) = *$ for some e in E_0 then $l\theta(e) = *$. Then for θ to be a morphism in $\mathbb{E}$ we would require $* \operatorname{div} l_0(e)$. Thus an S.A. can specify whether morphisms are partial or total functions. For example the categories $\mathbb{E}$ and $\mathbb{E}_{syn}$ arise from very simple S.A.s.

5.6 Example The S.A.'s for $\mathbb{E}$ and $\mathbb{E}_{syn}$

Take the S.A.'s A, S to be given by

A

	*	T	0
*	*	T	0
T	T	T	0
0	0	0	0

S

	*	T	0
*	*	0	0
T	0	T	0
0	0	0	0

Notice how morphisms in $\mathbb{E}_A$ may be partial functions as *divT. We get $\mathbb{E}_A \cong \mathbb{E}$. However morphisms in $\mathbb{E}_S$ must be total functions as * does not divide T. We get $\mathbb{E}_S \cong \mathbb{E}_{syn}$.

We now define the parallel composition of two labelled event structures as a restriction of the product in $\mathbb{E}$. Only pairs of events (one of which may be the fictitious event *) whose labels have a non-zero composition can be synchronised. (See definition 3.5 of product; π_0, π_1 below are the projection morphisms.).

5.7 Definition Let L be an S.A. Let (E_0,F_0,l_0), $(E_1,F_1,l_1) \in \mathbb{E}_L$. Define their parallel composition $(E_0,F_0,l_0) \textcircled{L} (E_1,F_1,l_1) = ((E_0,F_0) \times (E_1,F_1) \restriction E, l)$ where $E=\{e \in E_0 \overset{*}{\times} E_1 \mid l_0\pi_0(e) \cdot l_1\pi_1(e) \neq 0\}$ and $l(e)=l_0\pi_0(e) \cdot l_1\pi_1(e)$.

5.8 Example Let L be the S.A. for CCS without value passing - refer to 5.5. Suppose (E_0,F_0l_0), $(E_1,F_1,l_1) \in \mathbb{E}_L$. Then their parallel composition is their product in $\mathbb{E}$ restricted to the events $\{(e_0,*) \mid e_0 \in E\} \cup \{(*,e_1) \mid e_1 \in E_1\} \cup \{(e_0,e_1) \in E_0 \times E_1 \mid l_0(e_0), l_1(e_1)$ are complementary$\}$ with a subsequent labelling $l(e_0,*) = l_0(e_0)$, $l(*,e_1) = l_1(e_1)$, $l(e_0,e_1) = \tau$.

5.9 Example "Broadcasting" Let L be the S.A. with labels $*,0,\alpha,\tau$ satisfying laws of the form $\alpha \cdot \alpha = \alpha$, $\alpha \cdot * = \alpha \cdot \tau = 0$ and $\tau \cdot * = \tau$. Then the parallel composition of several processes must synchronise on α while τ-labelled events occur asynchronously (such multiway synchronisation is used in [H], [Mi], [LST] - see [M2] too).

	*	α	τ	0
*	*	0	τ	0
α	0	α	0	0
τ	τ	0	0	0
0	0	0	0	0

5.10 Example The following S.A. ensures all events occur asynchronously in a parallel composition (cf. S of 5.6 which ensures all events occur synchronously)

	*	T	0
*	*	T	0
T	T	0	0
0	0	0	0

There are obvious projection functions for a parallel composition which suggests $\textcircled{L}$ is a product. Although, in fact, the operation $\textcircled{L}$ is associative and does extend to a functor it is not always a product. It is however when the operation $\cdot$ behaves like the operation of least common multiple (l.c.m).

5.11 Proposition Let L be an S.A.

(i) The operation $\textcircled{L}$ extends to a functor $\mathbb{E}_L{}^2 \to \mathbb{E}_L$: For morphisms θ_0, θ_1 in $\mathbb{E}_L$ define $\theta_0 \textcircled{L} \theta_1 = \theta_0 * \theta_1$. The functor is associative i.e. for $E_0, E_1, E_2 \in \mathbb{E}_L$ there is a natural isomorphism $E_0 \textcircled{L} (E_1 \textcircled{L} E_2) \cong (E_0 \textcircled{L} E_1) \textcircled{L} E_2$.

(ii) Let (E_0, F_0, l_0), $(E_1, F_1, l_1) \in \mathbb{E}_L$. Then $(E_0, F_0, l_0) \textcircled{L} (E_1, F_1, l_1$ with the obvious projections, is their categorical product in $\mathbb{E}_L$ iff

$$\forall \gamma \in L \ \forall \alpha \in l_0 E_0, \ \beta \in l_1 E_1 . \ \alpha \text{div} \gamma \text{ and } \beta \text{div} \gamma \Rightarrow (\alpha \cdot \beta) \text{div} \gamma .$$

(iii) The parallel composition $\textcircled{L}$ with the evident projections always gives a categorical product in $\mathbb{E}_L$ iff

$$\forall \alpha, \beta, \gamma \in L . \ \alpha \text{div} \gamma \text{ and } \beta \text{div} \gamma \Rightarrow (\alpha \cdot \beta) \text{div} \gamma .$$

Proof (i) Let θ_i: $(E_i,F_i,l_i) \to (E_i',F_i',l_i')$ be morphisms in $\mathbb{E}_L$ for i=0,1. Let e be an event of (E_0F_0,l_0) (L) (E_1,F_1,l_1) i.e. $e \in E_0 {}_*\!\times E_1$ s.t. $l_0\pi_0(e)\cdot l_1\pi_1(e) \neq 0$. As θ_0,θ_1 are morphisms $l_i'\theta_i\pi_i(e) \text{ div } l_i\pi(e)$ for i = 0,1. Then by Lemma 5.3 (iv), $(l_0'\theta_0\pi_0(e)\cdot l_1'\theta_1\pi_1(e)) \text{ div } (l_0\pi_0(e)\cdot l_1\pi_1(e))$. By 5.3 (iii), $l_0'\theta_0\pi_0(e)\cdot l_1'\theta_1\pi_1(e)\neq 0$ so $(\theta_0\pi_0(e),\theta_1\pi_1(e))$ is an event of $(E_0',F_0'l_0')$ (L) (E_1',F_1',l_1').

The functor laws and associativity property of (L) follow as it is a restriction of $\times$, the product functor on $\mathbb{E}$.

(ii The "if" part follows as the condition stated above ensures the mediating morphism for $\times$ exists and, because (L) is a restriction of the product in $\mathbb{E}$. The "only if" part relies on event structures not having to be full: Take $\alpha \in l_0E_0$, $\beta \in l_1E_1$ so $\alpha \text{ div } \gamma$ and $\beta \text{ div } \gamma$. Take $(\{\varepsilon\},\emptyset,\{(\varepsilon,\gamma)\})$ and morphisms $\theta_0:\varepsilon \mapsto e_0$ where $l_0(e_0)=\alpha$ and $\theta_1:\varepsilon \mapsto l_1$ where $l_1(e_1)=\beta$. Assuming (L) is the product there is a mediating morphism $\varepsilon \to (e_0,e_1)$ where the event (e_0,e_1) must be labelled $\alpha\cdot\beta$ and must divide γ.

(iii) The "if" part follows as in (ii). The "only if" part would be true even if event structures had to be full. Suppose α and β divide γ. Take $E_0=(\{\varepsilon_0\},\{\emptyset,\{\varepsilon_0\}\},\{(\varepsilon_0,\alpha)\})$ and $E_1 = (\{\varepsilon_1\},\{\emptyset,\{\varepsilon_1\}\}, \{(\varepsilon_1,\beta)\})$. The product is E_0 (L) E_1 with the obvious projections, by assumption. Let $E_2 = (\{\varepsilon_2\},\{\emptyset,\{\varepsilon_2\}\}, \{(\varepsilon_2,\gamma)\})$ and $\theta_0:\varepsilon_2 \to \varepsilon_0$ and $\theta_1:\varepsilon_2 \to \varepsilon_1$ — both θ_0,θ_1 are morphisms. Because the mediating morphism exist $\alpha\cdot\beta \text{ div } \gamma$. ∎

5.12 Examples

(i) Let L be the S.A. for CCS. Then (L) does not coincide with product: We may have $\alpha \text{ div } \tau$ and $\beta \text{ div } \tau$ and $\alpha\cdot\beta = 0$ so $\alpha\cdot\beta$ cannot divide $\tau (\neq 0)$.

(ii For the S.A.'s A and S of $\mathbb{E}$ and $\mathbb{E}_{syn}$ it may be checked that $\cdot$ does satisfy the condition in proposition 5.11 (ii). As we know they do have products given by (L_0) and (L_1).

More generally for SA's L of the following forms (L) coincides with products:

	*	α	β	...	0
*	*	α	β	...	0
α	α	α	0	...	0
β	β	0	β	...	0
⋮	·	·	·		
0	0	0	0	...	0

	*	α	β	...	0
*	*	0	0	...	0
α	0	α	0	...	0
β	0	0	β		
⋮	·	·	·		
0	0	0	0	...	0

Before giving the programming language based on an S.A. we present a few extra much simpler operations on labelled event structures based on [M1, 2].

Definitions Let L be an S.A. Define the following operations on $\mathbb{E}_L$. Let $(E,F,1)$, $(E_i,F_i,l_i) \in \mathbb{E}_L$ for $i = 0,1$.

5.13 Lifting Suppose $\lambda \in L \setminus \{*,0\}$. Define $\lambda(E,F,1 = (E',F',1'$ where $E' = \{0\} \cup (\{1\} \times E)$ and

$$x \in F' \Leftrightarrow x \subseteq E' \text{ and } x = \emptyset \text{ or } (0 \in x \text{ and } \{e \mid 1,e) \in x\} \in F)$$

$$1(e' = \lambda \text{ if } e = 0,\ 1(e) \text{ if } e' = (1,e) \text{ for } e' \in E'$$

5.14 Sum Define $(E_0,F_0,l_0) + (E_1,F_1,l_1) = ((E_0,F_0) + (E_1,F_1,1)$ where + is the coproduct of 3.8 and $1((0,e)) = l_0(e)$ and $1((1,e)) = l_1(e)$.

5.15 Restriction Let $\lambda \in L \setminus \{*,0\}$. Define $(E,F,1) \setminus \lambda = (E,F) \restriction E',1'$ where $E' = \{e \in E \mid 1(e) \neq \lambda\}$ and $1' = 1 \restriction E'$.

5.16 Relabelling Let S be an endomorphism on L (i.e. S preserves *, 0 and · and $\forall \lambda \in L.\ S(\lambda) = 0 \Rightarrow \lambda = 0\ \&\ S(\lambda) = * \Rightarrow \lambda = *$). Define $(E,F,1) \langle S \rangle = (E,F,S1)$.

Apart from restriction, the above operations extend to functors on $\mathbb{E}_L$ in an obvious way. Sum is coproduct in $\mathbb{E}_L$. They are all continuous with respect to $\trianglelefteq_L$ the labelled version of $\trianglelefteq$. Thus we can take fixed points of them and their compositions.

5.17 Proposition Let L be an S.A. For $(E_0,F_0,l_0),(E_1,F_1,l_1)\in\mathbb{E}_L$ define $(E_0,F_0,l_0)\trianglelefteq_L(E_1,F_1,l_1)$ iff $(E_0,F_0)\trianglelefteq(E_1,F_1)$ and $l_0 = l_1\restriction E_0$. Then

(i $\mathbb{E}_L$ has lubs of all ω-chains ordered by $\trianglelefteq_L$.

(ii Each operation above is continuous with respect to $\trianglelefteq_L$ i.e. they preserve lubs of ω-chains ordered by $\trianglelefteq_L$.

(iii Let Γ be a continuous operation on $\mathbb{E}_L^r\to\mathbb{E}_L$. Let $\bot = ((\emptyset,\{\emptyset\},\emptyset),\ldots,(\emptyset,\{\emptyset\},\emptyset))\in\mathbb{E}_L^r$. Define $\mathrm{fix}\Gamma$ to be the lub of $\bot\trianglelefteq_L\Gamma\bot\trianglelefteq_L\ldots\trianglelefteq_L\Gamma^n\bot\trianglelefteq_L\ldots$. Then $\Gamma(\mathrm{fix}\Gamma) = \mathrm{fix}\Gamma$.

Proof (i) follows from the corresponding property of $\trianglelefteq$. (ii) In particular $\textcircled{L}$, + are continuous because $\times$,+ are. For the remaining operations use Lemma 2.9. (iii) is well known see [S]. ∎

Given L, an S.A., we define a language for communicating processes called Proc_L. Each term of Proc_L denotes an event structure in $\mathbb{E}_L$.

5.18 The syntax of Proc_L Assume an infinite set of process-variables $x\in X$. Define a term of Proc_L by:

$t ::= \underline{\mathrm{NIL}} \mid x \mid \lambda t \mid t+t \mid t\setminus\lambda \mid t\langle S\rangle \mid t\,\textcircled{L}\,t \mid x \text{ } \underline{\mathrm{isrec}} \text{ } t$, where $x\in X$, $\lambda\in L\setminus\{*,0\}$ and S is an endomorphism of L.

5.19 The semantics of Proc_L Define an environment to be a function $\rho: X\to\mathbb{E}_L$ from process-variables to labelled event structures. For a term t and an environment ρ, define $[\![t]\!]\rho$, the event structure t denotes with respect to ρ, by the following structural induction. (Note, that syntactic operators occur on the left and their semantic counterparts, operations on $\mathbb{E}_L$ occur on the right.)

$$
\begin{aligned}
[[\mathrm{NIL}]] &= (\emptyset,\{\emptyset\},\emptyset)\\
[[x]]\rho &= \rho(x)\\
[[\lambda t]]\rho &= \lambda([[t]]\rho)\\
[[t_1+t_2]]\rho &= [[t_1]]\rho + [[t_2]]\rho\\
[[t\backslash\lambda]]\rho &= [[t]]\rho \backslash \lambda\\
[[t\langle S\rangle]]\rho &= ([[t]]\rho)\langle S\rangle\\
[[t_1 \textcircled{L} t_2]]\rho &= ([[t_1]]\rho)\textcircled{L}([[t_2]]\rho)\\
[[x\ \underline{\text{isrect}}\, t]]\rho &= \mathrm{fix}\,\Gamma \text{ where } \Gamma: \mathbb{E}_L \to \mathbb{E}_L \text{ is given by}\\
&\qquad \Gamma(E) = [[t]]\rho[x\leftarrow E]
\end{aligned}
$$

A structural induction shows that Γ is indeed continuous so the above definition is justified by Proposition 5.17.

In a similar way one can obtain semantics in $\mathbb{P}_L$ and $\mathbb{T}r_L$; define parallel composition in either category as a restriction of $\times$ the product of § 3 and take environment into the categories. Equivalently one obtains semantics in $\mathbb{P}_L$ and $\mathbb{T}r_L$ by composing the above semantics with Pr and I extended to cope with labels.

5.20 Definition

(i Define $Pr_L: \mathbb{E}_L \to \mathbb{P}_L$ by $Pr_L(E,F,l) = (Pr(E,F), lev_{E,F})$ - refer to 4.3.
For $\rho: X \to \mathbb{P}_L$ and $t \in Proc_L$ define $[[t]]_P\,\rho = Pr_L([[t]]\rho)$.

(ii Define $I_L: \mathbb{E}_L \to \mathbb{T}r_L$ by $I_L(E,F,l) = (I(E,F), l\pi_{E,F})$ - refer to 4.3.
For $\rho: X \to \mathbb{T}r_L$ and $t \in Proc_L$ define $[[t]]_{Tr}\rho = I_L([[t]]\rho)$.

When L is the S.A. for CCS our interleaved semantics in $\mathbb{T}r_L$ agrees with Milner's synchronisation/communication tree semantics because of the following fact. (Our treatment of recursion is more general than Milner's so our denotations as trees may be χ_0-branching when recursion is not "guardedly well-defined".)

5.21 Proposition Let L be an S.A. Write the parallel composition operation in $\mathbb{T}r_L$ as $\underset{Tr}{\textcircled{L}}$ so $T \underset{Tr}{\textcircled{L}} S = I_L(T \textcircled{L} S)$. Suppose $T, S \in \mathbb{T}r_L$ are sums of the form $T \cong \sum_i \lambda_i T_i$ and $S \cong \sum_j \mu_j S_j$ for labels λ_i, $\mu_j \in L \setminus \{*, 0\}$ indexed by i and j. Then $T \underset{Tr}{\textcircled{L}} S$ is given recursively by

$$T \textcircled{L} S \cong \sum_{\lambda_i \cdot * \neq 0} (\lambda_i \cdot *) T_i \underset{Tr}{\textcircled{L}} S + \sum_{\lambda_i \cdot \mu_j \neq 0} (\lambda_i \cdot \mu_j) T_i \underset{Tr}{\textcircled{L}} S_j + \sum_{* \cdot \mu_j \neq 0} (* \cdot \mu_j) T \underset{Tr}{\textcircled{L}} S_j$$

Proof From 4.7 by restricting the product of trees. ∎

Isomorphism in each category $\mathbb{E}_L$, $\mathbb{P}_L$, $\mathbb{T}r_L$ induces a congruence on closed terms of $Proc_L$, where L is an S.A

5.22 Definition Let L be an S.A. For closed terms t, $t' \in Proc_L$ and any environment ρ define

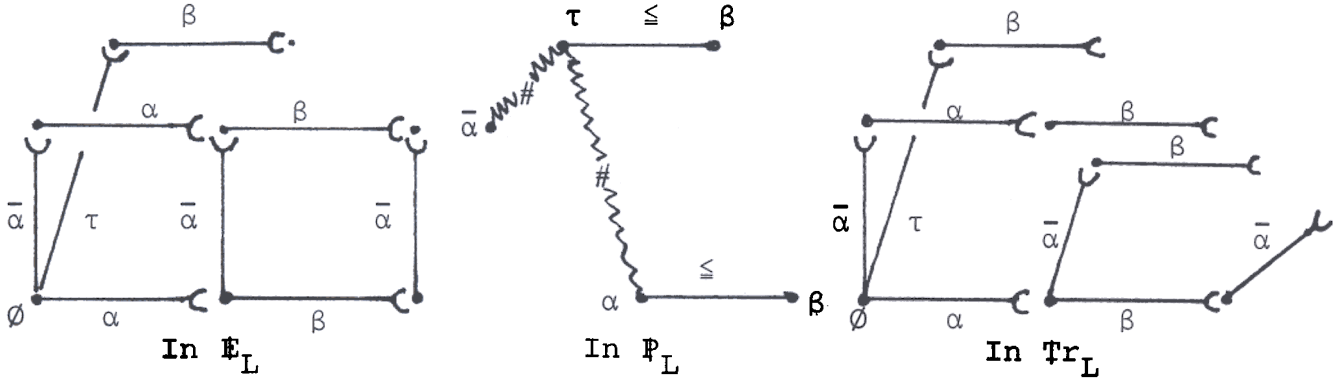
$$t \sim t' \text{ iff } [\![t]\!]\rho \cong [\![t']\!]\rho$$
$$t \sim_p t' \text{ iff } [\![t]\!]_p \rho \cong [\![t']\!]_p \rho.$$
$$t \underset{Tr}{\sim} t' \text{ iff } [\![t]\!]_{Tr} \rho \cong [\![t']\!]_{Tr} \rho.$$

5.23 Proposition The relations $\sim$, $\sim_p$, $\underset{Tr}{\sim}$ define congruences on the closed terms $Proc_L$ w.r.t. $\textcircled{L}$, +, λ-, $-\setminus\lambda$, $-\langle S \rangle$. We have $\sim \subseteq \sim_p \subseteq \underset{Tr}{\sim}$

Proof Each operation on $\mathbb{E}_L$ respects isomorphism. ∎

Generally because the event structures of $\mathbb{E}_L$ and $\mathbb{P}_L$ reflect concurrency their congruences are strictly included in that for $\mathbb{T}r_L$.

5.24 Example Let L be the S.A. for CCS. We look at denotations of the terms $\alpha\beta NIL \textcircled{L} \bar{\alpha} NIL$ in the categories $\mathbb{E}_L$, $\mathbb{P}_L$ and $\mathbb{T}r_L$ (obtained by restricting the products drawn in Example 4.8).



We have labelled events and coverings. Clearly in $\mathbb{T}r_L$ we have $\alpha\beta\,\mathrm{NIL}\,\textcircled{L}\,\bar{\alpha}\mathrm{NIL} \sim_{Tr} \bar{\alpha}\alpha\beta\mathrm{NIL} + \tau\beta\mathrm{NIL} + \alpha(\bar{\alpha}\beta\mathrm{NIL} + \beta\bar{\alpha}\mathrm{NIL})$ which does not hold for the other two congruences.

This strict inclusion fails in an interesting special case where communication is purely synchronous, when no asynchrony is allowed because L satisfies a strict synchronous law:

5.25 Definition Let L be an S.A. Say L is synchronous iff $\forall\lambda\in L\setminus\{*\}.\,\lambda\cdot * = 0$.

When an S.A. is synchronous parallel composition is purely synchronous, if an event is to occur in a parallel composition it must synchronise, no event can occur asynchronously. Then parallel composition is a restriction of the synchronous product $\otimes$. The synchronous product $\otimes$ is based on Set so parallel composition inherits some nice properties from product in Set.

5.26 Proposition Let L be an S.A. Then the following are equivalent

(i) L is synchronous

(ii) NIL is an $\textcircled{L}$-zero i.e. $t\,\textcircled{L}\,\mathrm{NIL} \sim \mathrm{NIL}$ for all terms $t \in \mathrm{Proc}_L$.

(iii) Parallel composition $\textcircled{L}$ distributes over sum i.e. $t_0\,\textcircled{L}\,(t_1+t_2) \sim t_0\,\textcircled{L}\,t_1 + t_0\,\textcircled{L}\,t_2$ for all terms $t_0,t_1,t_2 \in \mathrm{Proc}_L$

When L is synchronous denotations of closed terms in $\mathbb{P}_L$ are isomorphic to those in $\mathbb{T}r_L$, so $\sim_P = \sim_{Tr}$.

This indicates how assumptions on L determine laws and proof rules for congruences on terms.

Milner's synchronous calculi [M2] can be based on synchronous S.A.s as the following proposition shows:

5.27 Proposition (The synchronous calculi of [M2])
Any Abelian monoid $(M,\cdot,1)$ extends to an S.A. $(L,*,0,\cdot)$ simply by adjoining elements * and 0 to M and extending composition so $*\cdot* = *$ and $*\cdot\lambda = 0$ for all $\lambda \in L$. The language $Proc_L$ includes the synchronous calculus associated with the monoid M in [M2]. In the parallel composition $E_0 \textcircled{L} E_1$ every event e_0 of E_0 is synchronised with an event e_1 of E_1; the event may be labelled by 1 when it represents a delay or idle action. Denotations of closed terms of $Proc_L$ are pre-trees in Φ_L and trees in Φ_L.

CONCLUSION

Thus we have a framework in which to give denotational semantics to a wide range of parallel programming languages. But more, the framework makes connections between different kinds of semantics and different approaches. Milner's synchronisation trees are a special kind of labelled event structure. Thus we link up to the work in [M1]. (Notice incidentally that Milner's idea of "sequential observer" is embodied in the interleaving operator.) Then the synchronous calculi of Milner [M2] arise once synchronisation algebras satisfy a strict law, which essentially bans all synchrony. Unlike Milner in [M2] we do not model asynchrony in a synchronous framework but rather allow a free-mix of synchrony and asynchrony, depending on the synchronisation algebra. Prime event structures correspond to intuitive and simple structures of events with a causal dependency and conflict relation. We automatically get semantics in terms of these structures. This is important because for example Mogens Nielsen and Torben Fogh of Aarhus [F] and Ugo Montanari and coworkers of Pisa (see e.g. [MS]) have given semantics in terms of such structures, and also because the works [NPW1, 2], [W] establish links between such structures and Petri nets. Net semantics like that in [LTS] translates to prime event structure semantics by the techniques of [NPW1, 2] and [W]. (In fact there is a more direct connection between Petri nets and event structures. A condition event system [NT] with initial marking, which is contact-free and such that every condition occurs at most once in playing the token game, determines an event structure as follows: Take the configurations to be those sets of events which have occurred by some stage possibly infinite in playing the token game.) Then, event structures represent Scott domains and partially synchronous and synchronous morphisms induce rather special continuous functions between domains - see appendix B.

Clearly we have left several loose ends like:

- How to go from our semantics to proof rules.

- How to go from our rather basic semantics to more abstract semantics.

- How to give an operational semantics which justifies denotations which are sensitive to concurrency (the most philosophical and probably the most difficult loose end to tidy up);

- How to generalise event structures and still keep a useful category (for example are there more general event structures which model continuous processes or express "fairness"in some way? Then our present definition of morphisms should still be useful.).

- How to define homomorphisms of **synchronisation** algebras and use the attendant algebraic constructions.

APPENDIX A

Sets and partial functions

We take $\underline{\underline{\text{Set}}}$ to be the category of sets with usual function-composition. To cope with partial functions, we take $\underline{\underline{\text{Set}}}_*$ to have sets as objects but morphisms are now functions which may take the value * (representing "undefined"). A morphism in $\underline{\underline{\text{Set}}}_*$ is drawn as $\theta: X \to_* Y$. The morphisms $X \overset{\theta}{\underset{*}{\to}} Y$ and $Y \overset{\varphi}{\underset{*}{\to}} Z$ compose to $\varphi\theta(x) = (\theta(x))$ if $\theta(x) \neq *$, and * otherwise. Morphisms in $\underline{\underline{\text{Set}}}$ (total functions) correspond to those morphisms of $\underline{\underline{\text{Set}}}_*$ which never yield *. For $\theta: X \overset{\to}{\underset{*}{}} Y$ and $A \subseteq X$ define $\hat{\theta}(A) = \{\theta(e) \mid e \in A \ \& \ \theta(e) \neq *\}$.

For us, a notable fact about $\underline{\underline{\text{Set}}}_*$ is the nature of its products. If X and Y are sets their categorical product in $\underline{\underline{\text{Set}}}_*$ takes the form $X \underset{*}{\times} Y =_{def} \{(x,*) \mid x \in X\} \cup \{(*,y) \mid y \in Y\} \cup \{(x,y) \quad x \in X \ \& \ y \in Y\}$ with the obvious projections.

APPENDIX B

Domains of configurations

Here we show the relation between our categories of event structures and categories of Scott-domains.

B1 Some basic definitions: Let $(D,\sqsubseteq)$ be a partial order.

A directed subset of D is a non-null subset $S\subseteq D$ such that $\forall s,t\in S\ \exists u\in S.\ s\sqsubseteq u\ \&\ t\sqsubseteq u$. The p.o. D is a complete partial order (cpo) iff there is a least element $\perp\in D$ and all directed subsets S have a least upper bound (lub) $\bigsqcup S$.

If D is a cpo, an element $x\in D$ is isolated (= finite = compact) iff for all directed subsets S, if $x\sqsubseteq\bigsqcup S$ then $x\sqsubseteq s$ for some $s\in S$. A cpo D is said to be algebraic iff for each $x\in D$ the set S of isolated elements below x is directed and $x = \bigsqcup S$. (An algebraic cpo is generally called a domain though some authors insist it also be consistently complete - see below.)

Let $X\subseteq D$. Then X is said to be pairwise compatible iff $\forall x,y\in X\ \exists d\in D.x\sqsubseteq d\ \&\ y\sqsubseteq d$. The po $(D,\sqsubseteq)$ is coherent iff every pairwise compatible subset has a lub. (Clearly every coherent po is a cpo.) Similarly a subset X is said to be finitely compatible iff every finite subset of X has an upper bound in D. Then a po $(D,\sqsubseteq)$ is consistently complete iff every finitely compatible subset has a lub. (Clearly coherence implies consistent completeness.)

By proposition 1.8 an event structure $(E,\vdash)$ represents a domain $(\vdash,\subseteq)$ of configurations satisfying rather special properties. Such domains are coherent, prime algebraic and so that every isolated element dominates only a finite number of elements. Conversely any such domain is represented, to within isomorphism, by an event structure, in fact a prime event structure, in the following way: Take the complete primes as events and all the sets of complete primes below some element as configurations.

B2 Definition Let $(D,\sqsubseteq)$ be a prime algebraic coherent partial order satisfying the property that every isolated element dominates only a finite number of elements. Then define $p(D) =_{def} (P,F)$ where P = complete primes of D and $x \in F$ iff $\exists z \in D.\ x = \{p \in P \mid p \sqsubseteq z\}$.

B3 Lemma In the above definition $p(D)$ is a prime event structure so $(D,\sqsubseteq) \cong (F,\subseteq)$ under $x \longmapsto \{p \in P \mid p \sqsubseteq x\}$ with inverse $X \longmapsto \bigsqcup X$.

Proof Directly from the definitions. ∎

The concept of prime algebraicity was introduced in [NPW 1]. There Petri net concepts were related to Scott-domain concepts. In particular an event occurrence in a net showed itself as a complete prime in a domain of event configurations associated with the net. The complete primes formed a subbasis giving rise to the concept of prime algebraic domain. Now it turns out that, saying a domain is prime algebraic is just the same as saying it is completely distributive and algebraic, so really the concept is well known (see [CL]). We present the proof for lattices and its corollaries for domains.

It will follow that domains of configurations are, to within isomorphism, precisely the distributive, algebraic coherent partial orders which satisfy the finiteness property that every isolated element dominates only a finite number of elements.

B4 Definition Let $(D,\sqsubseteq)$ be a partial order with meets of all non-null subsets. Say $(D,\sqsubseteq)$ is distributive iff $x \sqcap (y \sqcup z) = (x \sqcap y) \sqcup (x \sqcap z)$ for all $x,y,z \in D$ so $y \sqcup z$ exists. Say $(D,\sqsubseteq)$ is completely distributive iff $(\bigsqcap X) \sqcup y = \bigsqcap_{x \in X} x \sqcup y$ for subsets $X \subseteq D$ s.t. $x \sqcup y$ exists for all $x \in X$ and $(\bigsqcup X) \sqcap y = \bigsqcup_{x \in X} x \sqcap y$ for non-null subsets $X \subseteq D$ s.t. $\bigsqcup X$ exists.

B5 Theorem Let $(D,\sqsubseteq)$ be a complete lattice. Then $(D,\sqsubseteq)$ is prime algebraic iff $(D,\sqsubseteq)$ is completely distributive and algebraic.

Proof It is easy to show a prime algebraic lattice is completely distributive and algebraic (or see [NPW1, 2]). Conversely suppose $(D,\sqsubseteq)$ is completely distributive and algebraic. Algebraicity expresses a kind of discreteness, it will mean:

(i) $\forall x,y\in D.x\not\sqsupseteq y \Rightarrow \exists z,z'\in D.x\sqsubseteq z \prec z'\sqsubseteq y$ where $\prec$ is the covering relation.

Complete distributivity will mean that each covering interval determines a complete prime, so:

(ii) Let $x \prec x'$ in D. Then $pr[x,x']=_{def}\bigsqcap\{y\in D \mid x'\sqsubseteq x\sqcup y\}$ is a complete prime of D.

To show (i), suppose $x,y\in D$ and $x\not\sqsupseteq y$. By algebraicity there is an isolated element a s.t. $a\not\sqsubseteq x$ and $a\sqsubseteq y$. By Zorn's lemma there is a maximal chain C of elements above x and strictly below $x\sqcup a$. As a is algebraic from the construction of C we must have $x\sqsubseteq \bigsqcup C \prec x\sqcup a\sqsubseteq y$.

To show (ii), let $x,x'\in D$ and $x \prec x'$. Suppose $p =_{def}\bigsqcap\{y\in D \mid x'\sqsubseteq x\sqcup y\}$. Note first that $x\sqcup p =\bigsqcap\{x\sqcup y \mid x'\sqsubseteq x\sqcup y\} = x'$ using complete distributivity. Now suppose $p\sqsubseteq \bigsqcup Z$ for some subset $Z\subseteq D$. Then $x' = x\sqcup p\sqsubseteq x\sqcup(\bigsqcup Z) = \bigsqcup_{z\in Z}(x\sqcup z)$. However as $x \prec x'$ we must then have $x'\sqsubseteq x\sqcup z$ for some $z\in Z$. But then $p\sqsubseteq z$. Thus p is a complete prime of D.

Let $z\in D$. Then we require $z =\bigsqcup\{pr[x,x'] \mid x \prec x'\sqsubseteq z\}$ in order to make D prime algebraic. Write $w =\bigsqcup\{pr[x,x'] \mid x \prec x'\sqsubseteq z\}$. Clearly $z\sqsupseteq w$. Suppose $z\not\sqsupseteq w$. Then $w\sqsubseteq x \prec x'\sqsubseteq z$ for some x,x' in D. Write $p = pr[x,x']$. Then $p\sqsubseteq w$ making $x\sqcup p = x$, a contradiction as $x\sqcup p = x'$. Thus D is prime algebraic as required. ∎

It is easy to see that a more general version of the above theorem also holds. The proof would work if the partial order $(D,\sqsubseteq)$ were coherent or consistently complete (sets with upper bounds have least upper bounds), and not necessarily a lattice.

As a corollary we obtain a representation theorem for completely distributive algebraic lattices; a completely distributive algebraic lattice is isomorphic to the left-closed subsets ordered by $\subseteq$ of some partial order. The converse is clear. (Surely this result exists in the lattice theory literature somewhere, but where?)

B6 Corollary (i) Let $(P,\leqq)$ be a partial order. Then the left closed subsets $(\mathcal{L}(P,\leqq),\subseteq)$ form a completely distributive algebraic lattice with complete primes of the form $[p] =_{def} \{p' \in P \mid p' \leqq p\}$ for $p \in P$.

(ii) Let $(D,\sqsubseteq)$ be a completely distributive algebraic lattice Let $(P,\leqq)$ be the complete primes ordered by $\leqq = \sqsubseteq \restriction P$. Then $(D,\sqsubseteq) \cong \mathcal{L}(P,\leqq)$ under $x \longmapsto \{p \in P \mid p \sqsubseteq x\}$.

Proof By the above theorem and the properties of prime algebraic lattices spelt out in [NPW1, 2] or [W]. ∎

Proposition 1.8 shows configurations of event structures give coherent prime algebraic domains satisfying the finiteness restriction that every isolated element dominates only a finite number of elements. In the presence of algebraicity and the finiteness restriction, complete distributivity is equivalent to the generally more humble distributivity. This gives the following characterisation of the domains of configurations.

B7 Proposition Ordered by inclusion the configurations of an event structure form a distributive, algebraic coherent partial order in which every isolated element dominates only a finite number of elements. Moreover any such partial order can be represented, to within isomorphism, by the configurations of a prime event structure.

Proof Domains of configurations clearly satisfy the above properties. To show the converse, we need only show that a distributive algebraic coherent partial order satisfying the above finiteness restriction is necessarily completely distributive. Let $(D,\sqsubseteq)$ be such a p.o. Then this distributive law follows: $x \sqcup (y \sqcap z) = (x \sqcup y) \sqcap (x \sqcup z)$ for $x,y,z \in D$ in which $x \sqcup y$ and $x \sqcup z$ exist. (See [Bir] or [KP] for details). Incidentally, because we do not work with lattices the two distributive laws are not equivalent. Now we show the two infinite distributivities hold.

(a) Let $X \subseteq D$ s.t. $\bigsqcup X$ exists and $y \in D$. Clearly then $\bigsqcup_{x \in X} (x \sqcap y) \sqsubseteq (\bigsqcup X) \sqcap y$. To show the converse inequality, suppose a is isolated and $a \sqsubseteq (\bigsqcup X) \sqcap y$. Then as $a \sqsubseteq \bigsqcup X$ and as a is isolated for some finite $X' \subseteq X$ we have $a \sqsubseteq \bigsqcup X'$. Then $a \sqsubseteq (\bigsqcup X) \sqcap y \Rightarrow a \sqsubseteq (\bigsqcup X') \sqcap y \Rightarrow a \sqsubseteq \bigsqcup_{x \in X'} (x \sqcap y)$ (by distributivity) $\Rightarrow a \sqsubseteq \bigsqcup_{x \in X} (x \sqcap y)$.

Thus as D is algebraic we have the converse inequality so $\bigsqcup_{x \in X} (x \sqcap y) = (\bigsqcup X) \sqcap y$.

(b) We require in addition that $(\bigsqcap X) \sqcup y = \bigsqcap_{x \in X} (x \sqcup y)$ for $y \in D$ and $\emptyset \neq X \subseteq D$ s.t. $x \sqcup y$ exists for all $x \in X$. Clearly $(\bigsqcap X) \sqcup y \sqsubseteq \bigsqcap_{x \in X} x \sqcup y$. Suppose a is isolated and $a \sqsubseteq \bigsqcap_{x \in X} x \sqcup y$. Then $a = \bigsqcap_{x \in X} (x \sqcup y)) \sqcap a = \bigsqcap_{x \in X} ((x \sqcup y) \sqcap a)$.

Now a dominates only a finite number of elements. Thus for some finite $X' \subseteq X$ we must have $a = \bigsqcap_{x \in X'} (x \sqcup y) \sqcap a)$.

Then by distributivity $a = ((\bigsqcap X') \sqcap a) \sqcup (y \sqcap a) \sqsubseteq (\bigsqcap X) \sqcup y$. By algebraicity we have $\bigsqcap_{x \in X} (x \sqcup y) \sqsubseteq (\bigsqcap X) \sqcup y$ and so the required equality. ∎

Thus event structures represent a natural class of domains. Similarly morphisms on event structures induce morphisms on domains.

are the properties they satisfy.

<u>B8 Definition</u> Let $(D_0, \sqsubseteq_0), (D_1, \sqsubseteq_1$ be partial orders. Let f be a function $f: D_0 \to D_1$. Say f is

<u>conditionally additive</u> (c.a.) iff $\forall X \subseteq D_0 . X\uparrow \Rightarrow f(\bigsqcup X) = \bigsqcup fX$

<u>conditionally multiplicative</u> (c.m.) iff $\forall X \subseteq D_0 . X \neq \emptyset$ & $X\uparrow \Rightarrow f(\bigsqcap X) = \bigsqcap fX$

(iii (a) <u>$\Subset$-preserving</u> iff $\forall x, x' \in D_0 . x \Subset x' \Rightarrow f(x) \Subset f(x')$

(b) <u>$\prec$-preserving</u> iff $\forall x, x' \in D_0 . x \prec x' \Rightarrow f(x) \prec f(x')$

(We use $\Subset$ to mean $\prec \cup Id$.)

B9 Lemma (i) Let θ: $(E_0,F_0) \to (E_1,F_1)$ be a morphism of event structures. Then $\hat{\theta}$: $F_0 \to F_1$ is c.a., c.m. and $\frown\!\!<$-preserving. If θ is synchronous then $\hat{\theta}$ is $\frown\!\!<$-preserving.
(ii) Let (E_0,F_0), (E_1,F_1) be prime event structures. Let f: $(F_0,\subseteq) \to (F_1,\subseteq)$ be c.a., c.m. and $\frown\!\!<$-preserving. Then there is a unique event structure morphism $\theta:(E_0,F_0) \to (E_1,F_1)$ s.t. $f = \hat{\theta}$. If further f is $\frown\!\!<$-preserving then θ is synchronous.

Proof (i) needs a routine verification.
(ii) Let (E_0,F_0), (E_1,F_1) be prime event structures. Let $f:F_0 \to F_1$ be a c.a., c.m. and $\frown\!\!<$-preserving function with respect to the inclusion ordering on configurations. We show how f is induced by an event structure morphism θ: $E_0 \to_* E_1$ so $f = \hat{\theta}$.

Recall some basic facts explained more fully in [NPW1, 2]:
A prime interval is a pair $[x,x']$ where $x \frown\!\!< x'$. Define a relation $\lessdot$ between prime intervals by $[x,x'] \lessdot [y,y']$ iff $x = x' \cap y$ and $y' = x' \cup y$. Form an equivalence relation $\sim$ as the symmetric transitive closure $\sim = (\lessdot \cup \lessdot^{-1})^*$. In fact, for a prime event structure, events are in 1-1 correspondence with $\sim$-equivalence classes of prime intervals in the configurations because there $[x,x'] \sim [y,y'$ iff $x' \setminus x = y' \setminus y$.

These facts make it easy to define the required event structure morphism θ. Let $x \frown\!\!< x'$ and $y \frown\!\!< y'$ in $(F_0,\subseteq)$ and $[x,x'] \lessdot [y,y']$ Then because f is c.a. and c.m. we get

$$f(x) = f(x' \cap y) = f(x' \sqcap f(y)$$
$$f(y' = f(x' \cup y) = f(x' \sqcup f(y).$$

Because f is $\frown\!\!<$-preserving too the above equations make $f(x) \frown\!\!< f(x'$ iff $f(y) \frown\!\!< f(y')$.

It follows that if $[x,x'] \sim [y,y']$ and $f(x) \frown\!\!< f(x'$ then $[f(x),f(x')] \sim [f(y),f(y'$]

Thus the following definition of $\theta: E_0 \to_* E_1$ is well-defined: for $e \in E_0$ take $x,x' \in F_0$ s.t. $x' \setminus x = \{e\}$; then if $f(x) \prec\!\!\!\!\cdot\; f(x')$ take $\theta(e)$ to be the unique event of $f(x') \setminus f(x)$, and otherwise set $\theta(e) = *$.

A simple induction on the size of x shows that for all finite $x \in F_0$ we have $\hat{\theta}(x) = f(x)$. Thus as $\hat{\theta}$ and f are c.a. we have $\hat{\theta} = f$. From the fact that f is c.m. it follows that $\theta(e) = \theta(e') \neq *$ for $e,e' \in x$, a configuration in F_0, implies $e=e'$. Thus θ is an event structure morphism inducing f. Any other event structure morphism inducing f must act like θ on prime intervals and so on events, making θ unique.

Clearly event structure morphisms which are total correspond to $\prec\!\!\!\!\cdot$-preserving functions as (i) and (ii) specialise to synchronous morphisms. ∎

As a corollary we can exhibit a natural equivalence between a category of domains and the category of prime event structures

B10 Definition Let $\mathbb{D}$ be the category of coherent distributive algebraic domains with morphism functions which are c.a., c.m. and $\prec\!\!\!\!\cdot$-preserving. Let $\mathbb{D}_{syn}$ be the subcategory with morphisms which are $\prec\!\!\!\!\cdot$-preserving.

B11 Proposition Taking $\hat{\cdot}: \mathbb{P} \to \mathbb{D}$ to act on objects by $(E,F) \mapsto (F,\subseteq)$ and on morphisms by

$$\begin{array}{ccc} (E_0,F_0) & & (F_0,\subseteq \\ \theta & & \hat{\theta} \;\big\downarrow \\ 1 & & (F_1,\subseteq \end{array}$$

defines a functor which is a natural equivalence of categories. It restricts to a natural equivalence $\hat{\cdot}: \mathbb{P}_{syn} \to \mathbb{D}_{syn}$.

Proof It is easy to check that $\hat{\cdot}$ is a functor. In [Mac] (theorem 1, page 91) it is shown that a functor is an equivalence of categories iff it is full, faithful and dense (i.e. every object in the codomain category is isomorphic to an object in the range of the functor) - all of which hold for $\hat{\cdot}$. ∎

This means that all the categorical properties of $\mathbb{P}$, $\mathbb{P}_{syn}$ transfer to $\mathbb{D}$ and $\mathbb{D}_{syn}$ respectively. For example we know there are products in $\mathbb{D}$ and $\mathbb{D}_{syn}$ and what form they take. It is hard to see how one could confirm the existence of products in $\mathbb{D}$ and $\mathbb{D}_{syn}$ without the aid of an event structure representation.

Viewed abstractly in the category $\mathbb{D}$ the approximation relation $\trianglelefteq$ corresponds to a special type of morphism on domains - the rigid embeddings of Kahn and Plotkin (see [KP]).

B12 Definition Let D_0, D_1 be two cpos. Let $f:D_0 \to D_1$ be a continuous function. Say f is an embedding iff there is a continuous function $g:D_1 \to D_0$ called a projection such that

$$g(f(x) = x \text{ for all } x \in D_0$$
$$\text{and} \quad f(g(y) \sqsubseteq y \text{ for all } y \in D_1.$$

Say f is a rigid embedding iff it is an embedding with projection g such that $y \sqsubseteq f(x) \Rightarrow fg(y) = y$ for all $x \in D_0$, $y \in D_1$.

B13 Theorem (i) Let $(E_0, F_0) \trianglelefteq (E_1, F_1)$ for two event structures. Let ι be the inclusion $\iota:E_0 \to E_1$. Then ι is a morphism of event structures such that $\hat{\iota}:(F_0, \subseteq) \to (F_1, \subseteq)$ is a rigid embedding.
(ii) The categories $\mathbb{D}$ and $\mathbb{D}_{syn}$ have colimits of ω-chains of rigid embeddings.

Proof (i) routine verification.
(ii) Let $D_0 \overset{f_0}{\to} D_1 \to \ldots D_n \overset{f_n}{\to} \ldots$ be an ω-chain of rigid embeddings By induction, there is a chain $(E_0, F_0) \trianglelefteq (E_1, F_1) \trianglelefteq \ldots \trianglelefteq (E_n, F_n) \trianglelefteq \ldots$ of (prime) event structures with $(F_n, \subseteq) \cong D_n$ so

$$\begin{array}{ccc} D_n & \xrightarrow{f_n} & D_{n+1} \\ \uparrow & & \| \\ (F_n, \subseteq & \xrightarrow{\iota_n} & (F_{n+1}, \subseteq \end{array}$$

commutes for each n, where ι_n is the inclusion map $\iota_n : E_n \hookrightarrow E_{n+1}$. Let (E,F) be its $\mathcal{S}$-lub. Then $(F,\subseteq)$ is easily seen to be the colimit of the chain $(F_0,\subseteq) \xrightarrow{\iota_0} (F_1,\subseteq) \xrightarrow{\iota_1} \cdots \longrightarrow (F_n,\subseteq) \xrightarrow{\iota_n} \cdots$
Thus $(F,\subseteq)$ is a colimit of $D_0 \xrightarrow{f_0} \cdots \longrightarrow D_n \xrightarrow{f_n} \cdots$. ∎

Thus we see event structure concepts in a domain setting. The categories $\mathbb{E}$ and $\mathbb{P}$ of event structures represent the category $\mathbb{D}$ of domains.

One can base semantics for synchronised communication directly on $\mathbb{D}$, though this will be essentially the same as a semantics based on prime event structures $\mathbb{P}$ because $\mathbb{D}$ and $\mathbb{P}$ are equivalent categories. One obtains labelled domains $\mathbb{D}_L$ by labelling prime intervals by elements of a synchronisation algebra L but in a $\sim$-respecting way (the same relation $\sim$ as above). We leave the detailed definition of $\mathbb{D}_L$ to the reader; it should be equivalent to $\mathbb{P}_L$. The treatment of recursion can be based on rigid embeddings. One expects a recursive definition to correspond to an ω-cocontinuous functor, for which we shall seek the "least fixed point". Starting with the null event structure repeated application of the functor will yield an ω-chain of rigid embeddings. The least fixed point will be its colimit (see [KP], [F], [P1]). There may be some lessons to be learnt from the categories $\mathbb{D}_L$ because they are closely akin to labelled transition systems often used to give operational semantics (used in e.g. [M1, 2]).

By the way the categories $\mathbb{D}$ and $\mathbb{D}_{syn}$ do not have exponentiations so are not cartesian closed; however a larger category in which morphisms are merely c.m. is cartesian closed and in fact is a full subcategory of Gérard Berry's dI-domains with stable functions (see [Ber]

Acknowledgements

I am grateful for discussions with Mogens Nielsen. Mogens has previously given a prime event structure semantics to CCS. Thanks to Gordon Plotkin for encouraging morphisms even when they were quarter-baked. The stability axiom is essentially Gérard Berry's "deterministic" condition [BC]. Related ideas appear in [F]and [MS]. Many thanks to Karen Møller and Angelika Paysen for typing a symbol-laden paper. The work was supported in part by an SRC grant directed by Robin Milner and Gordon Plotkin and in part by the Royal Society.

References

[B] G. Berry, "Modèles complement adéquats et stables des λ-calculs typés", Thèse de Doctorat d'Etat, Universit´ Paris VII, 1979.

G. Berry, P.L. Curien, "Sequential algorithms on concrete data structures", to appear in TCS.

[Bi G. Birkhoff, "Lattice theory", Coll. Pub., vol, 25, Amer. Math. Soc., Providence, R.I., 3rd edition, 1967.

G. Gierz, K.H. Hoffman, K. Keimal, J.D. Lawson, M. Mislove D. Scott, "A Compendium of Continuous Lattices", Springer-Verlag (1980).

T. Fogh, "En semantik for synkroniserede parallelle processer", Master's Thesis, Aarhus University, 1981.

C.A.R. Hoare, "A Model for Communicating Sequential Processes", Programming Research Group, Oxford University, 1978.

[KP] G. Kahn, G. Plotkin, "Structures de Données concrètes", IRIA-Laboria Report 336, 1979.

[LTS] P.E. Lauer, P.R. Torrigiani, M.W. Shields, "COSY: A System Specification Language based on Paths and Processes", Acta Informatica 12, 1979.

[Mac S. Maclane, "Categories for the working mathematician" Springer-Verlag, 1971.

G.J. Milne, "Synchronised Behaviour Algebras: a model for interacting systems", Dept, of Comp. Sci., University of Southern California, 1979.

[M1 R. Milner, "A Calculus for Communicating Systems", LNCS 92, Springer-Verlag, 1980.

[M2] R. Milner, "On relating Synchrony and Asynchrony", Dept. of Comp. Sci., University of Edinburgh, 1980.

U. Montanari, C. Simonelli, "On distinguishing between concurrency and nondeterminism", Proc. Ecole de Printemps on Concurrency and Petri Nets, Colleville, 1980 (to appear)

[NPW1 M. Nielsen, G. Plotkin, G. Winskel, "Petri nets, event structures and domains", Proc. Conf. on Semantics of Concurrent Computation, Evian, LNCS 70, Springer-Verlag, 1979

[NPW2] M. Nielsen, G. Plotkin, G. Winskel, "Petri nets, event structures and domains, part I", TCS 13, 1981.

W. Brauer (ed.), "Net Theory and Applications", LNCS 84, Springer-Verlag, 1980.

G.D. Plotkin, "A Structural View of Operational Semantics", Lecture Notes, Computer Science Department, Aarhus University, 1981.

G.D. Plotkin, "Lectures on Domains", Summerschool, Pisa, 1978.

D.S. Scott, "Lectures on a Mathematical Theory of Computation", Lecture notes in mathematics, University of Oxford, 1980.

[W] G. Winskel, "Events in Computation", Ph.D. Thesis, Dept. of Comp. Sci., University of Edinburgh, 1980.