AUTOMATA-THEORETIC MODELS OF PARALLELISM

Mike Shields
Electronic Engineering Laboratories
The University of Kent at Canterbury

Preface

The following notes describe the main formal results of theory of parallel systems based on generalised automata. The ultimate aim of the theory is to provide *non-interleaving operations semantics* in a uniform manner to existing and not-yet existing models of parallel systems.

Part 1 describes the basic behavioural model. Systems are conceived of as generating behaviour: a behaviour is essentially a pre-order· whose elements represent occurrences of events and are labelled by the names of the events of which they are occurrences. A collection of such sets is called a behavioural presentation. Behavioural presentations are capable of exhibiting the full range of parallel behaviour; sequentiality, non-determinism, concurrency and simultaneity.

The link between behavioural presentations and automata is provided by the trace languages of Mazurkiewicz. A trace, an element of a semi-commutative monoid, is capable of representing a certain kind of asynchronous behaviour. In part 2, we develop the basic order theoretic properties of trace languages and show how they determine behavioural presentations belonging to a certain class - the so-called *linguistic* behavioural presentations. We also present some results about languages of finite and infinite traces.

Conventional automata accept string languages. By extending the structure, they may be made to accept trace languages and hence linguistic behavioural presentations. By extending the automata further, they become powerful eneough to accept so-called *discrete* behavioural presentations.

The notes are not heavily mathematical - or at least, not as heavily mathematical as I could make them. The reader is expected to know set theory and a little bit about partial orders and semigroups, but not much.

Mike Shields
Electronic Engineering Laboratories
The UNiversity of Kent at Canterbury
December 1988

1. Behavioural Presentations

1.1. Basic Definitions

We begin by considering the formal description of behaviour in a system. The following model is an adaption of that of *event structures* [1].

DEFINITION 1.1.1. (Behavioural Presentations [2]) A behavioural presentation is a quadruple $B = (O, P, E, \lambda)$ where

(1) O is a set of occurrences;

(2) $P \subseteq 2^O$ is a non-empty set of points satisfying $\bigcup_{p \in P} p = O$;

(3) E is a set of events;

(4) $\lambda: O \rightarrow E$ is a function.

Intuitively, B describes the entire repertoire of activity of some system. Things happen when a system executes; O is the set of all such things. A happening is the occurrence of some events. $\lambda(o) = e$ is to be read 'o is an occurrence of the event e'. Finally, there are certain points in the notional space-time of the system at which assertions may be made as to what has occurred. $p \in P$ is the point at which it may be asserted that precisely those occurrences which are elements of p have occurred.

EXAMPLE 1.1.2. (Waveforms) Behavioural presentations may be used to describe continuous or analogue systems. Consider an electronic black box with two output lines. The function of the box is to generate a signal f at one of its output lines and a signal g at its other output line. We consider the behaviour of the system between times t_1 and t_2. We shall let the occurrences of this system be the attaining of a given voltage at a given time by a given signal, so that $O = [t_1, t_2) \times \{f, g\}$.

Points correspond to instants in the time interval, so that $P = \{p_t \mid t \in [t_1, t_2]\}$.

and the set of all things that have happened prior to a point t is therefore $p_t = [t_1, t) \times \{f, g\}$.

Finally, $\lambda(t, f) = f(t)$ and $\lambda(t, g) = g(t)$.

EXAMPLE 1.1.3. (Special Relativity) This example is based on the famous thought-experiment of Einstein in [3] Two trains are travelling at a constant speed in opposite directions along a pair of straight parallel tracks. Observers O_1 and O_2 are sitting at the middle of the two trains. At a given instant, the two observers are on a line at right angles to the side of the train with a third observer, O_3, sitting on the embankment, and at that instant two forks of lightning strike the ends of the first train in such a way that O_3 sees them strike simultaneously. Observer O_2, travelling towards the light coming from the first strike, sees that before he sees the light coming from the second. Observer O_1 travelling towards the light coming from the second strike, sees that before he sees the light coming from the first.

Let o_1 denote the occurrence of the first bolt of lightning striking and let o_2 denote the occurrence of the second bolt of lightning striking, so that $O = \{o_1, o_2\}$.

Now, from the point of view of observer O_3, there are two distinct time points; $p_0 = \emptyset$ when nothing has happened yet and $p_{both} = \{o_1, o_2\}$, when both have. O_3 never sees one without the other; O_3 sees the lightning bolts strike simultaneously. O_2 sees the first bolt strike before the second, so that from his point of view, there are three points; p_0 and p_{both} and a third point $p_{first} = \{o_1\}$, when o_1 has occurred but not o_2. Likewise, O_1 has three points; p_0, p_{both} and $p_{second} = \{o_2\}$.

Thus, $P = \{p_0, p_{both}, p_{first}, p_{second}\}$. We may take $A = \{flash\}$ and $\lambda(o_1) = \lambda(o_2) = flash$.

EXAMPLE 1.1.4. (Coin Tossing) The preceding examples have all been of *determinate* systems; there has been no element of choice or indeterminacy in them. The next example is a description of a system consisting of a coin being tossed and coming down either head or tail. Let $A = \{H, T\}$.

If X is a set, then X^* denotes the set of all finite sequences (or words or strings) of elements of X. Let Ω denote the null or empty string. Let $X^+ = X^* - \{\Omega\}$. If $x, y \in X^*$, then we write $x.y$ for the string concatenation of x and y and we define $x \leq y \iff \exists\, u \in X^*: x.u = y$.

We suppose that two occurrences of the same action are the same $\Longleftrightarrow$ they have been preceded by the same sequences of events. The third 'tail' in the sequence *HTHHTT* is not the same occurrence as the third 'tail' in the sequence *HHTHHTT*; they take place in different 'possible worlds'. An event may be specified by giving the sequence of which it is the last occurrence. Thus, $O = \{o_x \mid x \in A^+\}$ and $P = \{p_x \mid x \in A^*\}$, with $p_x = \{o_y \mid y \leq x\}$. If $x = u.a$, with $a \in A$, then set $\lambda(o_x) = a$.

Now, let us return to the general case. So far we have not considered relationships between occurrences. We introduce two fundamental relations.

DEFINITION 1.1.5. Let B be a behavioural presentation and suppose $o_1, o_2 \in O$. We define.

(1) $o_1 \# o_2 \Longleftrightarrow \forall p \in P : o_2 \in p \Rightarrow o_1 \notin p$:

(2) $o_1 \to o_2 \Longleftrightarrow \forall p \in P : o_2 \in p \Rightarrow o_1 \in p$.

These two definitions introduce concepts of *mutual exclusion* and *time ordering*. If $o_1 \# o_2$, then an occurrence o_2 excludes the future occurrence o_1 - and vice versa. It is this relation that allows us to introduce notions of *non-determinism* into the theory. If $o_1 \to o_2$, on the other hand, then if o_2 has occurred, then so must o_1. We may read $o_1 \to o_2$ as meaning, 'occurrence o_1 either preceded or was at the same time as occurrence o_2'.

The following remark gives the basic properties of these two relations. First of all, recall that a *pre-order* is a reflexive, transitive relation. An *independence relation* is an irreflexive,[1] symmetric relation.

REMARK 1.1.6. Let B be a behavioural presentation.

(1) $\#$ is an independence relation on O satisfying the following property $o_1 \# o_2 \;\&\; o_1 \to o'_1 \;\&\; o_2 \to o'_2 \Rightarrow o'_1 \# o'_2$;

(2) $\to$ is a pre-order on O.

□

Other temporal relationships may be constructed from $\#$ and $\to$.

DEFINITION 1.1.7. Let B be a behavioural presentation and suppose $o_1, o_2 \in O$.

(1) o_1 *strictly preceeds* o_2 iff $o_1 < o_2$, where $o_1 < o_2 \Longleftrightarrow (o_1 \to o_2) \;\&\; \neg(o_2 \to o_1)$

(2) o_1 and o_2 are *simultaneous* iff $o_1 \approx o_2$, where $o_1 \approx o_2 \Longleftrightarrow o_1 \to o_2 \;\&\; o_2 \to o_1$

(3) o_1 and o_2 are *concurrent* if $o_1 \; co \; o_2$, where $o_1 \; co \; o_2 \Longleftrightarrow \neg(o_1 \# o_2) \;\&\; \neg(o_1 \to o_2) \;\&\; \neg(o_2 \to o_1)$.

PROPOSITION 1.1.8. Let B be a behavioural presentation, then

(1) $<$ is a strict pre-order, that is a transitive relation which is irreflexive and asymmetric[2] relation.

(2) $\approx$ is an equivalence relation which is a congruence with respect to the pre-order $\to$ and the mutual exclusion relation $\#$.[3]

(3) co is an independence relation.

PROPOSITION 1.1.9. Let B be a behavioural presentation and let $o_1, o_2 \in O$, then precisely one of the following is true: $o_1 < o_2$, $o_2 < o_1$, $o_1 \approx o_2$, $o_1 \; co \; o_2$ or $o_1 \# o_2$.

□

Behavioural presentations may be classified according to the nature of their co, $\approx$ and $\#$ relations. First, if X is a set, then let Id_X denote its identity relation.

DEFINITION 1.1.10. (Types of Behavioural Presentation) Let B be a behavioural presentation, then

---

[1] $\forall o \in O : \neg o < o$.

[2] $\forall o, o' \in O : o < o' \Rightarrow \neg o' < o$.

[3] If we denote the $\approx$ class of o by $[o]_{\approx}$, and the set of all $\approx$ classes of O by $O/\approx$, then (a) the relation $\to_{\approx}$ given by $[o]_{\approx} \to_{\approx} [o']_{\approx} \Longleftrightarrow o \to o'$ is well defined and a partial order on $O/\approx$ and (b) the relation $\#_{\approx}$ defined by $[o]_{\approx} \#_{\approx} [o']_{\approx} \Longleftrightarrow o \# o'$ is well defined and is an independence relation on $O/\approx$ satisfying (1) of 1.1.6.

(1) B is *sequential* $\Longleftrightarrow co = \varnothing$ and $\approx = id_O$.

(2) B is (non-sequentially) *synchronous* $\Longleftrightarrow co = \varnothing$ and $\approx \neq id_O$.

(3) B is (non-sequentially) *asynchronous* $\Longleftrightarrow co \neq \varnothing$ and $\approx = id_O$.

(4) B is (non-sequentially) *hybrid* $\Longleftrightarrow co \neq \varnothing$ and $\approx \neq id_O$.

By synchronous (respectively asynchronous, hybrid), we usually mean non-sequentially synchronous or sequential (respectively non-sequentially asynchronous or sequential, non-sequentially hybrid or sequential).

DEFINITION 1.1.11. Let B be a behavioural presentation, then

(1) B is *determinate* $\Longleftrightarrow \# = \varnothing$.

(2) B is *non-determinate* $\Longleftrightarrow \# \neq \varnothing$.

REMARK 1.1.12. (Classification of Behavioural Presentations) Let B be a behavioural presentation, then precisely one of the following holds.

(1) B is sequential and determinate.

(2) B is sequential and non-determinate.

(3) B is non-sequentially synchronous and determinate.

(4) B is non-sequentially synchronous and non-determinate.

(5) B is non-sequentially asynchronous and determinate.

(6) B is non-sequentially asynchronous and non-determinate.

(7) B is non-sequentially hybrid and determinate.

(8) B is non-sequentially hybrid and non-determinate.

□

EXERCISE Determine # and $\rightarrow$ for examples 1.1.2, 1.1.3 and 1.1.4. Hence, classify them according to the taxonomy of remark 1.1.12.

1.2. Discrete Behavioural Presentations

We begin this section with an examination of two of Zeno's paradoxes.

In his *arrow paradox*, he argues that an arrow can never reach its target, since before it can get there it must first travel half the distance, but before that it must travel the first quarter and before that it must travel the first eighth and so on.

First of all, notice that we are given a set of points - points at which assertions are made about the current state of affairs - for example, that the arrow has travelled one-eighth of the distance towards the target. We thus have a set of events, $E_{arrow} = \{e_n \mid n \in N\}$, where e_n is shorthand for 'The arrow travels reaches 2^{-n} of the distance' (so that e_0 represents the arrow arriving at the target). We define a set of occurrences $O_{arrow} = \{o_n \mid n \in N\}$ and define $\lambda_{arrow}(o_n) = e_n$. Finally, let $p_i = \{o_n \in O \mid n \geq i\}$ and let $P_{arrow} = \{p_n \in O \mid n \in N\}$.

If we now examine B_{arrow}, it does indeed seem that nothing can happen. How does the system start? Each point is preceded by (is a superset of) another one and there is no initial point (which would have to be $\varnothing$) at which nothing yet has happened yet. Even if we add $\varnothing$ into P, nothing can happen from it, for, no occurrence o_n is prior to all the others; we may easily check that $\ldots \rightarrow o_{n+1} \rightarrow o_n \rightarrow \ldots \rightarrow o_2 \rightarrow o_1$.

A similar paradox concerns Achilles and a tortoise. The two agree to have a race and in acknowledgement of Achilles' greater speed, the tortoise is given a head start. The argument goes that Achilles will never overtake the tortoise, for by the time that he reaches a place where the tortoise was at some previous time, the latter will have moved on. In this paradox, Zeno constructs a series of points extending into the future towards, but never attaining, a hypothetical point at which Achilles and the tortoise are dead level.

In this case also, we may exhibit a behavioural presentation $B_{achilles}$ with occurrences $O_{achilles} = O_{arrow}$, where o_1 denotes the occurrence of Achilles reaching the starting position of the tortoise and o_n represent him reaching the position that the tortoise had reached at occurrence o_{n-1}. The points are $p_n = \{o_i \mid i \leq n\}$ together with a futuremost point, which we shall call p_∞, which we can think of as the point when Achilles catches up with the tortoise, so that $p_\infty = O$. It is easy to see that $o_1 \rightarrow o_2 \rightarrow \ldots \rightarrow o_n \rightarrow o_{n+1} \rightarrow \ldots$. This system *can* get started, but the futuremost point is never reached.

Zeno's argument may be applied to any system which can be described by a piecewise continuous function of time. It may be used, for example, to 'prove' that a logic inverter never works (think of the 'arrow' of time crossing the interval of its propagation delay).[4] However, we know that logic inverters *do* work (or so, at least, my colleagues assure me), that arrows *do* reach their targets - or at least reach *something* - and that unless he is hit by a stray arrow, Achilles *will* catch up with and pass the tortoise. If we look at Zeno's argument, we see that it has the form

If time is discrete then motion is impossible

and since we know motion to be all-too possible, it follows that time is *not* discrete.

Computer Science is largely concerned with discrete systems, systems which proceed (or are considered to proceed, whether they do or not) in isolated hiccups. We would like to use our model to describe such systems and so would like to discover a subclass of behavioural presentations that we may deem to represent discrete behaviour. Our discussion leads us to exclude infinite, strictly ascending and descending chains of points.

DEFINITION 1.2.1. (Chain Conditions) Let B be a behavioural presentation.

(1) B will be said to satisfy the *descending chain condition* (DCC) $\Longleftrightarrow$

$$\forall p, p_1, p_2, \ldots \in P: p_1 \supseteq p_2 \supseteq \cdots \supseteq p \Rightarrow \exists N: n \geq N \Rightarrow p_n = p_N$$

(2) B will be said to satisfy the *ascending chain condition* (ACC) $\Longleftrightarrow$

$$\forall p, p_1, p_2, \ldots \in P: p_1 \subseteq p_2 \subseteq \cdots \subseteq p \Rightarrow \exists N: n \geq N \Rightarrow p_n = p_N$$

However, behavioural presentations, even without such infinite chains, still lack an essential element of discrete behaviour. We illustrate with an example.

Consider a behavioural presentation with $O = \{c, h, t\}$, $P = \{p_0, p_1, p_2\}$ and $p_0 = \emptyset$, $p_1 = \{c, h\}$ and $p_2 = \{c, t\}$ (ignore λ and E). We can think of it as representing a simple nondeterministic system is which a coin is tossed (occurrence c) and then either lands with its head uppermost (occurrence h) or its tail uppermost (occurrence t). We may check that $c < h$, $c < t$ and $h \# t$.

However, there is something missing. The coin has been tossed; it glitters as it spins through the air and for a heartbeat or two the two opponents wait apprehensively for the outcome - who will be forced to give the FORTRAN for Social Scientists course? This is certainly a point in the space-time of the system, which would be represented by a set $\{c\}$ - but this point is not present in P.

As well as the chain conditions, we shall insist that a discrete presentation contains enough time points to separate events which are strictly ordered or non-simultaneous. This is the repletion property, defined below.

DEFINITION 1.2.2. (Repletion) Let B be a behavioural presentation, then B is *replete* $\Longleftrightarrow$

$$\forall p_1, p_2 \in P: p_1 \subseteq p_2 \, \forall o_1, o_2 \in p_2 - p_1:$$
$$\neg\, o_2 \rightarrow o_1 \Rightarrow \exists p_3 \in P: p_1 \subseteq p_3 \subseteq p_2 \;\&\; o_1 \in p_3 \, o_2 \notin p_3 \qquad [1]$$

[1] is a bit of a mouthful, so let us go explain it in words. Given points p_1 and p_2 such that p_1 is before p_2, and given that o_1 and o_2 occurred between p_1 and p_2, then if o_2 did not occur before or at the same time as o_1, then there is a point in time, p_3, after p_1 and before p_2, at which it is legitimate to assert that o_1 has happened but that o_2 hasn't.

[4] Rumour has it that F. X. Reid's latest programming language, *Zeno*, operates according to some such principles. All *Zeno* programs are partially correct as a consequence.

For example, $c, h \in p_1 - p_0$ and $\neg (h \to c)$. In order for the behavioural presentation to be replete, we would need a point p_3 such that $p_0 \subseteq p_3 \subseteq p_1$ such that $c \in p_3$ and $h \notin p_3$. It follows that $p_3 = \{c\}$ which, as we have observed, does not belong to P. Hence the example is not replete.

We have not quite dealt with the problem with which the introduction of the notion of repletion was intended to deal. Consider the example of the tossed coin again, but with the 'initial state', p_o, removed. $<$ and # are unaffected by this modification, but something is again missing. Not only do we not have a point at which the coin has not yet been tossed, but in this new behavioural presentation, the repletion property is invoked in vain; there are no points p, p' such that $p \subset p'$. One further condition will do the trick.

DEFINITION 1.2.3. (Bottom Element) Let B be a behavioural presentation, then B has a *bottom element* $\Longleftrightarrow \emptyset \in P$.

We choose the properties described in definitions 1.2.1 to 1.2.3 as characterising the behaviours of discrete systems.

DEFINITION 1.2.4. (Discreteness) Let B be a behavioural presentation, then B will be said to be *discrete* $\Longleftrightarrow B$ satisfies the DCC and the ACC, is replete and has a bottom element.

The rest of this section is devoted to elucidating the properties of discrete behavioural presentations. We are chiefly concerned with three aspects.

Firstly, we must reassure ourselves that discrete systems do proceed in an orderly way in discrete steps without giving rise to embarrassing anomalies. In doing this, we need to explain what a step actually *is* in terms of our model. This will be of great importance later when we come to examine the relationships between behavioural presentations and automata-like objects (see chapter 3).

Secondly, we shall give an alternative characterisation of the discreteness property. This will be in terms of a certain finiteness condition (to cope with the chain conditions) and a closure property. This closure property is related to an ordering on subsets of O.

Finally, if we are to use discrete behavioural presentations to describe and reason about the behaviour of systems, then the more we know about their properties the better. It turns out that what is significant is the closure property. Behavioural presentations with this property have a characteristic order structure.

So, let us look at steps.

DEFINITION 1.2.5. (Steps or Derivations) Let B be a behavioural presentation. A derivation or step in B is a triple (p, X, p') where $p, p' \in P$ and $X \in O/\approx$ such that $p \subseteq p'$ and $p' - p = X$.

We shall write $p \vdash^X p'$ to indicate that (p, X, p') is a step and we shall refer to it as *a step from p to p' via (the occurrences in) X.* If $X = \{o\}$, for $o \in O$, then we write $p \vdash^o p'$.

Looking back at the Zeno examples, we see that in B_{arrow}, $p_n \vdash^{o_{n-1}} p_{n-1}$ for each n, but that $\emptyset \vdash^o p$ for no p or o - there is no first step. In the athletics example, $\emptyset \vdash^{o_1} p_1 \vdash^{o_2} p_2 \vdash \ldots$, but for no p or o do we have $p \vdash^o p_\infty$ - there is no final step. However:

LEMMA 1.2.6. (Existence of Steps) Let B be a replete behavioural presentation satisfying the DCC, then

$$\forall p, p' \in P: p \subset p' \Rightarrow \exists p'' \in P \;\; \exists X \in O/\approx: p \vdash^X p'' \subseteq p'$$

□

LEMMA 1.2.7. (Finiteness of Chains) Let B be a replete behavioural presentation satisfying the DCC and the ACC, then

$$\forall p, p' \in P: p \subset p' \Rightarrow \exists p_1, p_2, \ldots, p_{n-1} \in P \; \exists X_1, X_2, \ldots, X_n \in O/\approx:$$
$$p \vdash^{X_1} p_1 \vdash^{X_2} p_2 \vdash \ldots p_{n-1} \vdash^{X_n} p'$$

□

DEFINITION 1.2.8. ($\approx$-Finitary Property) For a behavioural presentation B and $p \in P$, define $p/\approx = \{X \in O/\approx \mid X \subseteq p\}$. A behavioural presentation will be said to be *$\approx$-finitary* $\Longleftrightarrow \forall$

$p \in P: |p/\approx| < \infty$.

COROLLARY 1.2.9. Let B be a discrete behavioural presentation, then B is $\approx$-finitary.

□

LEMMA 1.2.10. Let B be a $\approx$-finitary behavioural presentation, then B satisfies the DCC and the ACC.

□

The $\approx$-finitary property corresponds to the chain conditions. It is not unreasonable to wonder whether there is anything corresponding to repletion.

DEFINITION 1.1.11. (Left-Closure Relation) Let $X, Y \subseteq O$, then X is *left closed* in Y and we write $X \leq Y \iff X \subseteq Y$ and

$\forall o_1 \in X \; \forall o_2 \in Y: o_2 \to o_1 \Rightarrow o_2 \in X$.

For instance, in the coin-tossing example, $\{c\} \leq \{c, h\}$ and $\{c\} \leq \{c, t\}$.

It is easy to verify that $\leq$ is a partial order on 2^O and that for $p_1, p_2 \in P, p_1 \subseteq p_2 \iff p_1 \leq p_2$.

DEFINITION 1.2.12. (Left-Closed Behavioural Presentations) A behavioural presentation will be said to be *left closed* $\iff$

$\forall p \in P \; \forall X \subseteq O: X \leq p \Rightarrow X \in P$

The following related notions will be useful later.

DEFINITION 1.2.13. (Left-Closure Operator) Let $X \subseteq O$, we define its left-closure, written $\downarrow X$ to be the set $\{o \in O \mid o \to o' \text{ some } o' \in X\}$.

If $X = \{o\}, o \in O$, then we write $\downarrow o$ for $\downarrow X$.

If $\mathbf{U} \subseteq 2^O$, then we define $\downarrow \mathbf{U} = \{\downarrow X \mid X \in \mathbf{U}\}$.

REMARK 1.2.14. (Properties of $\downarrow$) Let $X, Y \subseteq O, \mathbf{U} \subseteq 2^O$ and $p \in P$, then

(1) $X \subseteq \downarrow X$;

(2) $X \subseteq Y \Rightarrow \downarrow X \leq \downarrow Y$ and in particular, $\downarrow X \leq O$;

(3) $\downarrow \bigcup_{X \in \mathbf{U}} X = \bigcup_{X \in \mathbf{U}} \downarrow X$;

(4) $\downarrow \bigcap_{X \in \mathbf{U}} X = \bigcap_{X \in \mathbf{U}} \downarrow X$;

(5) $p = \downarrow p$.

□

THEOREM 1.2.15. (Characterisation of Discrete Behavioural Presentations) Let B be a behavioural presentation, then B is a $\approx$-finitary and left-closed behavioural presentation $\iff$ B is discrete.

□

This result is useful, not only because it gives a more tractable characterisation of discreteness than that given in definition 1.2.4, but also because it emphasises the importance of the left-closure property. The principal order-theoretic properties of discrete behavioural presentations that we shall now establish depend almost without exception on left-closure. The two characteristic properties of left-closed behavioural presentations are the properties of being *consistently complete* and *prime algebraic*. We shall deal with them in that order.

DEFINITION 1.2.16. (Consistent Completeness) Let $(X, \leq)$ be a partial order, then $(X, \leq)$ is consistently complete $\iff$

$\forall Y \subseteq X: Y \neq \varnothing \Rightarrow glb(Y)$ exists.

Here $glb(Y)$ denotes the greatest lower bound of the set Y.

DEFINITION 1.2.17. (Primes and Prime Algebraic Posets) Let $(D, \leq)$ be a poset. An element $x \in D$ is a *complete prime* of D $\iff$

$\forall X \subseteq D : x \leq lub(X) \Rightarrow \exists y \in X : x \leq y.$ [2]

x is a prime if [2] is guaranteed to hold for *finite* X. Since we shall not be needing to talk about primes, as opposed to complete primes, we shall drop the adjective 'complete' from now on.

$(D, \leq)$ is *prime algebraic* $\Leftrightarrow$

$\forall x \in D$: $lub(\{y \in Pr(D) \mid y \leq x\})$ exists and equals x.

THEOREM 1.2.18. Suppose B is a left closed behavioural presentation, then $(P, \subseteq)$ is prime algebraic and consistently complete. The complete primes of $(P, \subseteq)$ are the elements $\downarrow o$, $o \in O$. Furthermore if $U \neq \emptyset$, then

(1) $glb(U) = \bigcap_{p \in U} p$;

(2) $lub(U) = \bigcup_{p \in U} p$, if the former exists.

□

We have a converse. It relies on noticing that there is a 1-1 correspondance between elements of $O/\approx$ and primes. If we take any prime algebraic, consistently complete partial order D, then let O be the set of primes of D, $P = \{Pr(d) \mid d \in D\}$, where $Pr(d)$ is the set of primes below D, and E and λ anything reasonable (say $E = O$ and $\lambda = Id_O$). This actually gives a behavioural presentation. The construction is one that we shall meet again several times.

THEOREM 1.2.19. Suppose $(D, \leq)$ is prime algebraic and consistently complete, then there exists a left-closed behavioural presentation B_D such that $(P_D, \subseteq)$ is isomorphic to $(D, \leq)$.

□

1.3. Event Structures and Domains

An alternative approach to modelling concurrent behaviour is taken by the authors of [4]. The objects in question are known as *event structures*. In this section, we describe the connection between them and behavioural presentations. Our reasons twofold. Firstly, event structures are an important model in their own right and constituted an important source of inspiration for the work reported here. Secondly, the relationship between the two involves a form of closure - analogous to the closure of a string language.

We shall not be concerned with E or λ in the first part of this section; our results concern only the order structure of P. Let us therefore define an *unlabelled behavioural presentation* to be a pair $B = (O, P)$ where O is a set of occurrences and $P \subseteq 2^O$ satisfies $\bigcup_{p \in P} p = O$. All results obtained heretofore apply to unlabelled behavioural presentations.

We present a slightly extended notion of event structure. In [4] the ordering relation is a partial order rather than a pre-order.

DEFINITION 1.3.1. (Event Structures) A (hybrid) event structure is a triple $S = (O, \rightarrow, \#)$, where

(1) O is a set of *occurrences*;

(2) $\rightarrow$ is a preorder on O;

(3) $\#$ is an irreflexive, symmetric relation on O satisfying:

$\forall o_1, o_2, o'_1, o'_2 \in O : o_1 \rightarrow o'_1 \; \& \; o_2 \rightarrow o'_2 \; \& \; o_1 \# o_2 \Rightarrow o'_1 \# o'_2$

By remark 1.1.6, if B is a behavioural presentation, then $(O, \rightarrow, \#)$ is an event structure. Define $\Sigma(B) = (O, \rightarrow, \#)$.

Conversely, given an event structure, and hence relations $\rightarrow$ and $\#$, we may construct a set of points, as we now define.

DEFINITION 1.3.2. (Processes of Event Structures) Let $S = (O, \rightarrow, \#)$ be an event structure then $X \subseteq O$ is said to be *conflict-free* $\Leftrightarrow$

$\forall o_1, o_2 \in X : \neg\, o_1 \# o_2.$

$X \subseteq O$ is said to be a *process* of $S \iff X$ is conflict-free and $X \leq O$. We denote the set of all processes of S by $P(S)$.

As in the case of behavioural presentations, we define

$\downarrow X = \{o \in O \mid \exists o' \in X: o \rightarrow o'\}$,

for all non-empty sets $X \subseteq O$. It is easy enough to show that if $o \in O$, then $\downarrow o \in P(S)$. This entails that $O \subseteq \bigcup_{p \in P(S)} p$, from which it follows that $(O, P(S))$ is an unlabelled behavioural presentation.

Define $\Lambda(S) = (O, P(S))$. Summing up the foregoing discussion, we have.

REMARK 1.3.3. If B is a behavioural presentation, then $\Sigma(B)$ is an event structure. If S is an event structure then $\Lambda(S)$ is an unlabelled behavioural presentation.

□

Σ and Λ establish a relationship between the objects in the two models, unlabelled behavioural presentations and hybrid event structures. Let us now look at this relationship in more detail. First we need a definition.

DEFINITION 1.3.4. Let B_1, B_2 be unlabelled behavioural presentations. Define $B_1 \subseteq B_2 \iff O_1 \subseteq O_2$ & $P_1 \subseteq P_2$. Of course, $\subseteq$ is reflexive, asymmetrical and transitive.

REMARK 1.3.5

(1) Let B be an unlabelled behavioural presentation, then $B \subseteq \Lambda(\Sigma(B))$;

(2) Let S be an event structure then $S = \Sigma(\Lambda(S))$.

□

Let **ES** denote the class of all event structures. Let **BP** denote the class of all unlabelled behavioural presentations and let **CBP** denote the class of all unlabelled behavioural presentations of the form $\Lambda(S)$ for $S \in$ **ES**. We may first observe that Λ is a bijection between ES and CBP. For, it is certainly onto, by definition, while if $\Lambda(S_1) = \Lambda(S_2)$ then by 1.3.5 (2),

$S_1 = \Sigma(\Lambda(S_1)) = \Sigma(\Lambda(S_2)) = S_2$

Furthermore, Σ restricted to **CBP** is the inverse to Λ. That $S = \Sigma(\Lambda(S))$ is given in 1.3.5 (2). If $B \in$ **CBP**, then $B = \Lambda(S)$, some $S \in$ **ES**, and so

$\Lambda(\Sigma(B)) = \Lambda(\Sigma(\Lambda(S))) = \Lambda(S) = B$

Therefore, hybrid event structures correspond precisely to behavioural presentations belonging to the class **CBP**. Standard event structures (in which the $\rightarrow$ relation is a partial order) correspond precisely to the asynchronous behavioural presentations belonging to the class **CBP**.

There still remain a number of questions, notably, what are the elements of **CBP** like and how are they related to those of **BP**? We note that $\Lambda \circ \Sigma$ determines a map from **BP** to **CBP**; what is the relationship between $B \in$ **BP** and $\Lambda(\Sigma(B))$? Let us first introduce some notation.

DEFINITION 1.3.6. Suppose $B \in$ **BP**. Define $\overline{B} = \Lambda(\Sigma(B))$.

REMARK 1.3.7.

(1) $B \in$ **CBP** $\iff B = \overline{B}$;

(2) $\forall B \in$ **CBP**: $\overline{B} = \overline{\overline{B}}$.

□

DEFINITION 1.3.8. (Coherent Posets) Let $(D, \leq)$ be a poset.

$x, y \in D$ are *compatible* $\iff$ the set $\{x, y\}$ possesses a least upper bound.

$X \subseteq D$ is *pairwise compatible* $\iff$ every $x, y \in X$ are compatible.

$(D, \leq)$ is *coherent* $\iff$ every pairwise compatible subset of D has a least upper bound.

THEOREM 1.3.9. [4] Suppose S is an event structure, then $(P(S), \subseteq)$ is prime algebraic and coherent, with as primes the elements $\downarrow o$, each $o \in O$.

□

It follows that the elements of CBP are prime algebraic and coherent. This property turns out to characterise them.

PROPOSITION 1.3.10. Let $B = (O, P)$ be an unlabelled behavioural presentation, then $B \in$ CBP $\Leftrightarrow$ $(P, \subseteq)$ is left closed and coherent.

□

We have now identified event structures as corresponding precisely to those behavioural presentations which are left-closed and coherent. We end this section by describing the relationship between behavioural presentations B and $\bar{B}$.

PROPOSITION 1.3.11. Suppose B is a behavioural presentation, then $\bar{B}$ is the smallest left-closed, coherent behavioural presentation containing B. That is, if B' is left closed and coherent, then $B \subseteq B' \Rightarrow \bar{B} \subseteq B'$. We shall call $\bar{B}$ the *closure* of B. If $B = \bar{B}$ (equivalently, by 1.3.7 (1), $B \in$ CBP), then we shall say that B is *closed.*

□

We now turn to a second version of the idea of event structure. This is taken from [1], in which Winskel is concerned with providing behavioural semantics for algebraic languages such as CCS. First, we need to following notion.

Let E be a set and let $\mathbf{F} \subseteq 2^E$. Suppose $X \subseteq \mathbf{F}$, then X is compatible in $\mathbf{F}$ - and we write $X \uparrow^{\mathbf{F}}$ - $\Leftrightarrow \exists y \in \mathbf{F} \, \forall x \in X: x \subseteq y$. If $x, y \in \mathbf{F}$, then we write $x \uparrow^{\mathbf{F}} y$ for $\{x, y\} \uparrow^{\mathbf{F}}$.

DEFINITION 1.3.12. An *event structure* in the sense of [1] (hereafter just 'event structure' providing there is no possibility of confusion) is a pair $S = (E, \mathbf{F})$ where E is a set of *events* and $\mathbf{F} \subseteq 2^E$ is a set of *configurations*, satisfying.

(1) $\mathbf{F}$ is *coherent*: if $X \subseteq \mathbf{F}$ is pairwise compatible, then $\bigcup_{x \in X} x \subseteq \mathbf{F}$;

(2) $\mathbf{F}$ is *stable*: $\forall X \subseteq \mathbf{F}: X \neq \emptyset \;\&\; X \uparrow^{\mathbf{F}} \Rightarrow \bigcap_{x \in X} x \subseteq \mathbf{F}$;

(3) $\mathbf{F}$ is *coincidence free*:
$\forall x \in \mathbf{F} \, \forall e, e' \in x: e \neq e' \Rightarrow \exists y \subseteq x: (e \in y \;\&\; e' \notin y)$ *or* $(e' \in y \;\&\; e \notin y)$;

(4) $\mathbf{F}$ is *finitary*: $\forall x \in \mathbf{F} \, \forall e \in x \; \exists y \subseteq x: e \in y \;\&\; |y| < \infty$.

Furthermore, if $E = \bigcup \mathbf{F}$, then S is *full.*

Thus, an event structure is rather like an unlabelled behavioural presentation. Indeed, *full* event structures *are* unlabelled behavioural presentations (though not all unlabelled behavioural presentations satisfy all the conditions of definition 1.3.12). However, there is a great deal of difference between the ways in which behavioural presentations and event structures are to be understood in defining system behaviour.

Technically, this difference manifests itself in the way time-ordering of events is defined.

DEFINITION 1.3.13. Let $S = (E, \mathbf{F})$ be an event structure and let $x \in \mathbf{F}$. Define $e \leq_x e' \Leftrightarrow \forall y \in \mathbf{F}: y \subseteq x \;\&\; e' \in y \Rightarrow e \in y$.

$\leq_x$ is clearly reflexive and transitive, while condition 1.3.12 (3) ensures that it is asymmetrical. Thus $(x, \leq_x)$ is a partial order for each $x \in \mathbf{F}$.

$\leq_x$ is defined *locally* on the configurations of S, as opposed to our relation $\rightarrow$ which is defined *globally* for a behavioural presentation. It is possible that $e_1 \leq_x e_2$ for some $x \in \mathbf{F}$, while $\neg\, e_1 \leq_y e_2$ for some $y \in \mathbf{F}$.

Let us illustrate this with an example taken from [1]. Let $S = (E, \mathbf{F})$, where

$E = \{0, 1, 2\}$
$\mathbf{F} = \{\emptyset, \{1\}, \{2\}, \{1, 0\}, \{2, 0\}\}$

S is an unlabelled behavioural presentation in which 1 *co* 0, 2 *co* 0 and 1 # 2.

Regarded as an event structure, however, we have $1 \leq_{\{1,0\}} 0$ and $2 \leq_{\{2,0\}} 0$. Furthermore, 1 and 2 are in conflict - they never appear in the same configuration. So the event structure describes a system in which either 1 or 2 occur, after which an 0 occurs. However, the two 0s are distinct. In fact, in the terminology of chapter 2, the elements of E stand revealed as events rather than occurrences.

S considered as a behavioural presentation does not represent the same system as S regarded as an event structure. It is relatively easy to see how we *could* represent S by a behavioural presentation. Set

$O = \{o_0, o'_0, o_1, o_2\}$,
$P = \{\emptyset, \{o_1\}, \{o_2\}, \{o_1, o_0\}, \{o_2, o'_0\}\}$,
$E = \{0, 1, 2\}$ and
$\lambda(o_1) = 1, \lambda(o_2) = 2, \lambda(o_0) = \lambda(o'_0) = 0.$

As we shall see, any event structure may be represented by a behavioural presentation, in the sense of the following definition.

DEFINITION 1.3.17. Suppose $S = (E, \mathbf{F})$ is an event structure and B' is a behavioural presentation, then B' will be said to represent $S \iff$ there exists a bijection $\pi\colon \mathbf{F} \to P'$ such that:

(1) $\forall x, y \in \mathbf{F}\colon x \subseteq y \iff \pi(x) \subseteq \pi(y)$;

(2) $E' = E$;

(3) For each $x \in \mathbf{F}$, $\lambda(\pi(x)) = x$ and the restriction of λ to $\pi(x)$ is a bijection satisfying

$\forall o_1, o_2 \in \pi(x)\colon o_1 \to o_2 \iff \lambda(o_1) \leq_x \lambda(o_2)$.

Thus, B' represents S providing there is a correspondence between the points of B' and the configurations of S such that corresponding points and configurations describe the same partial orders.

We would like to know whether any event structure may be represented by a behavioural presentation - that is, whether the class of behavioural presentations contains that of event structures in any sense. To show that this is indeed the case, we shall give a construction that turns event structures into behavioural presentations.

The discussion leading to the statement of theorem 3.23 suggests that if S is prime algebraic, then we may construct the points of a behavioural presentation from the complete primes of S. S is indeed prime algebraic, as the next result shows.

THEOREM 1.3.15. [1] Suppose $S = (E, \mathbf{F})$ is an event structure, then $(\mathbf{F}, \subseteq)$ is prime algebraic and coherent. The complete primes of S are the elements $\downarrow_x e$, where

$\downarrow_x e = \{e' \in x \mid e' \leq_x e\}$

Write $Pr(S) = \{\downarrow_x e \mid x \in \mathbf{F} \;\&\; e \in x\}$.

□

Since coherence entails consistent completeness - any set which is bounded above is easily seen to be pairwise consistent - we may use theorem 1.3.15 and theorem 1.2.19 to construct a behavioural presentation from S. Recall that in the construction of theorem 1.2.19, occurrences are constructed from the primes and points are the primes below given members of - in this case $\mathbf{F}$.

DEFINITION 1.3.16. Let $S = (E, \mathbf{F})$ be an event structure. Define $B_S = (O_S, P_S, E_S, \lambda_S)$, where

(1) $O_S = Pr(S)$;

(2) $P_S = \{p_x \mid x \in \mathbf{F}\}$, where $p_x = \{\downarrow_x e \mid e \in x\}$;

(3) $E_S = E$

(4) $\lambda_S\colon O_S \to E_S$ is defined $\lambda(\downarrow_x e) = e$.

PROPOSITION 1.3.17. Suppose $S = (E, \mathbf{F})$ is an event structure, then B_S is a behavioural presentation and B_S represents S via the map π_S defined by $\pi_S(x) = p_x$.

□

Not all behavioural presentations represent event structures. Let us suppose that B represents $S = (E, \mathrm{F})$ via a function π. What can we conclude about B?

Firstly, π establishes a poset isomorphism between $(P, \subseteq)$ and $(\mathrm{F}, \subseteq)$. Hence, by theorem 1.3.18, $(P, \subseteq)$ is prime algebraic and coherent. It follows, by theorem 1.3.10 that

- B is closed.

Secondly, we note that (3) of 1.3.15 entails that S contains no non-trivial simultaneity. This is true of B also.

- B is asynchronous.

Next, we note that according to the finitary condition, 1.3.15 (4), each event of S belongs to a finite set. Let us say that B is *prime-finitary* $\Leftrightarrow \forall o \in O: |\downarrow o| < \infty$.

- B is prime-finitary.

The final peculiarity of behavioural presentations which represent event structures is to do with labelling. We note that π is a bijection and that $\lambda(\pi(x)) = x$, so that it follows that the sets $\lambda(p)$ be all distinct. We also note that λ must be injective on each p. There is one final property, which is not as easily motivated, that is, that

$$\forall p, p' \in P \; \forall X \subseteq p: \lambda(X) = \lambda(p') \Rightarrow X = p' \qquad [3]$$

A labelling function λ having these three properties - namely that the sets $\lambda(p)$ are all distinct, that λ is injective on each p and that [3] holds - will be called *W-labelled.*

- B is W-labelled.

We shall say that B is *Winskellian* $\Leftrightarrow$ B is closed, B is asynchronous, B is prime-finitary and B is W-labelled.

THEOREM 1.3.18 B is Winskellian $\Leftrightarrow$ there exists an event structure S such that B represents S. Furthermore, S is full $\Leftrightarrow$ λ is surjective.

□

2. Trace and Vector Languages

2.1. Basic Definitions and Properties

Trace languages were introduced by A. Mazurkiewicz in [5] to give a firing-sequence-type semantics to Petri Nets, in which concurrency could be represented. Technically, a trace language is a subset of a semi-commutative monoid, that is to say, a monoid generated by some set A subject to relations of the form $a.b = b.a$, for certain pairs $a, b \in A$. (Of course, if this holds for *all* pairs, then the monoid will be commutative). The idea is that if $a.b = b.a$, then the trace $a.b$ represents a behaviour in which a and b have happened, but not in that order - or in *any* order. Their occurrences are concurrent.

Trace languages thus generalise string languages - which provide one means for describing sequential behaviour (although an imperfect one) - by providing a language theoretic means for representing concurrent behaviour (though not all concurrent behaviour - we shall see how expressive they are later).

They also provide the vital like between automata and behavioural presentations. Automata accept languages - and we can generalise them to accept trace languages - and trace languages describe behaviour - so we should be able to map an appropriate type of trace language to an appropriate type of behavioural presentation.

DEFINITION 2.1.1. (Traces and Trace Languages) [5]) Let A be a set and let ι be an independence relation on A. We define a relation $\equiv_\iota^{(1)}$ on A^* by:

$$x \equiv_\iota^{(1)} y \iff \exists\, u, v \in A^*\ \exists\, a, b \in A: x = u.a.b.v \ \&\ y = u.b.a.v \ \&\ a\ \iota\ b$$

Let $\equiv_\iota$ be the reflexive, transitive closure of $\equiv_\iota^{(1)}$. By definition, $\equiv_\iota$ is an equivalence relation on A^*. We denote the equivalence class of $x \in A$ by $\langle x\rangle_\iota$.

Let $A_\iota^* = \{\langle x\rangle_\iota \mid x \in A^*\}$. A_ι^* is the set of *traces* of A with independence relation ι. Any subset L of A_ι^* is called a *trace language.*

PROPOSITION 2.1.2. Suppose ι is an independence relation on a set S, then

(1) A_ι^* is a monoid with respect to a composition defined by $\langle x\rangle_\iota.\langle y\rangle_\iota = \langle x.y\rangle_\iota$. The identity of the monoid is $\langle\Omega\rangle_\iota$.

(2) A_ι^* is a partial order with bottom $\langle\Omega\rangle_\iota$ with respect to the relation $\leq$ defined by

$$\langle x\rangle_\iota \leq \langle y\rangle_\iota \iff \exists\, z \in A^*: \langle x.z\rangle_\iota = \langle y\rangle_\iota$$

□

We shall now describe a representation of trace languages using vectors of strings. These were first appeared in [6] as an asynchronous semantics for path expressions. The intuition behind them is that there is a collection of sets of events, $\alpha(i)$, $i \in I$ such that any two elements $a, b \in A = \bigcup_{i \in I} \alpha(i)$ may never occur concurrently. Thus, any behaviour of the system determines a set of sequences $x_i \in \alpha(i)^*$ describing the order in which occurrences of elements from $\alpha(i)$ have taken place during that behaviour. The behaviour may be described by the *vector* of strings whose i corrdinate is x_i. Let us set up the machinery.

DEFINITION 2.1.3. (Indexed Covers) An indexed cover for a set A, is a map $\alpha: I \to \underline{P}(A)$ such that $\bigcup_{i \in I} \alpha(i) = A$

LEMMA 2.1.4. ([7]). Every indexed cover α determines an independence relation ι_α by

$$a\ \iota_\alpha\ b \iff \forall\, i \in I: \neg\, \{a, b\} \subseteq \alpha(i)$$

Conversely, if ι is an independence relation, then there exists a indexed cover α such that $\iota_\alpha = \iota$.

□

From lemma 2.1.4, we see that an indexed cover α of A determines a trace monoid $A_{\iota_\alpha}^*$. α also determines a set of string-vectors and we have noted that these may be concatenated together. It transpires that vectors form a monoid, which we shall call A_α^*, under this concatenation. Theorem 2.1.14 explains the connection between the two. First, we need some definitions.

DEFINITION 2.1.5. (α-Vectors) An α-vector is a mapping $x: I \to A^*$ such that

$\forall i \in I: x(i) \in \alpha(i)^*$

Ω_α is defined to be the vector v satisfying $\forall i \in I: v(i) = \Omega$.

If $I = \{1, \ldots, n\}$, then we shall shall sometimes write α-vectors x in the form $(x_1, \ldots, x_n)$, where for each i, $x(i) = x_i$.

We shall denote the set of all α-vectors by M_α. We may define a componentwise concatenation and ordering on M_α as follows.

DEFINITION 2.1.6. (Concatenation and Ordering of α-vectors) Let $x, y \in M_\alpha$, then:

(1) x.y is defined to be the α-vector z such that $\forall i \in I: z(i) = x(i).y(i)$;

(2) $x \le y \iff \forall i \in I: x(i) \le y(i)$.

REMARK 2.1.7. Suppose that α is an indexed cover, then:

(1) M_α is a monoid with identity Ω_α;

(2) M_α is a poset w.r.t. $\le$, with bottom Ω_α.

A particular submonoid of M_α is of interest to us, namely that generated by the vectors corresponding to individual actions.

DEFINITION 2.1.8. Let $a \in A$, then a_α is the α-vector such that

$$a_\alpha(i) = \begin{cases} a \text{ if } a \in \alpha(i) \\ \Omega \textit{ otherwise} \end{cases}$$

Define $A_\alpha = \{a_\alpha \mid a \in A\}$ and let A_α^* denote the submonoid of M_α generated by A_α.

The most important elementary fact about A_α^* is that it has the same commutativity as A_ι^*, where $\iota = \iota_\alpha$.

LEMMA 2.1.9. Suppose $a, b \in A$, then $a_\alpha.b_\alpha = b_\alpha.a_\alpha \iff a \; \iota_\alpha \; b$ *or* $a = b$.

□

It will follow from lemma 2.1.9 that A_ι^* is isomorphic to A_α^*. This is the main point of introducing the vectors, after all; they are 'concrete' representations of traces.

We shall need to argue by induction on the size of a vector. Intuitively, each vector has a 'length', namely the number of elements of A_α that go to make it up. Let us make this precise. First, if $a \in A$ and $x \in A^*$, then define $\#_a(x)$ to be the number of a's occurring in x. Formally, $\#_a(\Omega) = 0$ and

$$\#_a(x) = \begin{cases} 0 & \text{if } x = \Omega \\ \#_a(y) + 1 & \text{if } x = a.y \\ \#_a(y) & \text{if } x = b.y \;\&\; a \ne b \end{cases}$$

If $a \in A$, $x \in A_\alpha^*$, then $\forall i, j \in I: a \in \alpha(i) \cap \alpha(j) \Rightarrow \#_a(x(i)) = \#_a(x(j))$, so we may define $\#_a(x)$ to be $\#_a(x(i))$, for any $i \in I$ such that $a \in \alpha(i)$, and to be 0 if no such i exists. Finally, define $lnth(x) = \sum_{a \in A} \#_a(x)$. The following are easy consequences of the definition.

REMARK 2.1.9. (Properties of $lnth$) Suppose $a \in A$, $x, y \in A_\alpha^*$, then

(1) $lnth(x) = 0 \iff x = \Omega_\alpha$;

(2) $lnth(a_\alpha) = 1$;

(3) $lnth(x.y) = lnth(x) + lnth(y)$.

□

We shall find it convenient to extend the notion of independence from individual actions to vectors.

DEFINITION 2.1.11. (Independence of Vectors) Let $x, y \in A_\alpha^*$, then x and y are *independent*, and we write $x \ ind_\alpha \ y \iff$

$\forall i \in I: x(i) > \Omega \Rightarrow y(i) = \Omega$

The following are easy consequences of the definition.

REMARK 2.1.12. (Properties of ind_α) Suppose $a, b \in A$, $x, y \in A_\alpha^*$, then

(1) $x \ ind_\alpha \ y \iff y \ ind_\alpha \ x$;

(2) $a_\alpha \ ind_\alpha \ b_\alpha \iff a \ \iota_\alpha \ b$;

(3) $x \ ind_\alpha \ y \Rightarrow x.y = y.x$.

(4) $x.y \ ind_\alpha \ z \iff x \ ind_\alpha \ z \ \& \ y \ ind_\alpha \ z$.

□

Finally, we introduce a 'left-cancellation' operator.

PROPOSITION 2.1.13. (Left-cancellation) If $x, y \in A_\alpha^*$ with $x \leq y$, then there exists a unique $z \in A_\alpha^*$ such that $x.z = y$. We denote this vector by y/x. Furthermore:

$\forall i \in I: (y/x)(i) = y(i)/x(i)$

where for $x, y \in A$ with $x \leq y$, y/x is defined to be the unique z such that $x.z = y$.

□

THEOREM 2.1.14. (Vector Representation of Traces) Suppose $\iota_\alpha = \iota$, then the mapping $\phi_\alpha: A_\iota \rightarrow A_\alpha$ defined by $\phi_\alpha(\langle a \rangle_\iota) = a_\alpha$ extends to a unique mapping $\phi_\alpha^*: A_\iota^* \rightarrow A_\alpha^*$ which is both a poset and monoid isomorphism.

□

Let us now turn to order theoretic properties of A_α^*. We begin by considering greatest lower upper bounds. First, if $X \subseteq A_\alpha^*$ is non empty, define

$x \leq X \iff \forall y \in X: x \leq y$ and $X \leq x \iff \forall y \in X: y \leq x$

and if $x \leq X$ define $X/x = \{y/x \mid y \in X\}$.

It is possible to show that if $a_1, \ldots, a_n \in A_\alpha^*$ such that $a_1 \leq X$ and $a_{i+1} \leq X/(a_1 \ldots a_i)$ and $lub(X/(a_1 \ldots a_n) = \Omega$, then $lub(X) = a_1 \ldots a_n$. The proof involves an induction on $lnth(X) = min(\{lnth(x) \mid x \in X\})$. The induction step is not quite as striaghtforward as it looks at first and uses the following lemma, which we include for completeness.

LEMMA 2.1.15. Suppose $y, z \in A_\alpha^*$ and $a \in A$ such that $a \leq z$, then

(1) $y \leq z \Rightarrow (a \nleq y \iff a \ ind \ y)$;

(2) $a \ ind \ y \Rightarrow (y \leq z \iff y \leq z/a)$.

□

Hence:

PROPOSITION 2.1.16. (Existence of *glbs*) Let $X \subseteq A_\alpha^*$, then if $X \neq \emptyset$, then $glb(X)$ exists.

□

COROLLARY 2.1.17. A_α^* is consistently complete.

□

By a standard result in the theory of partial orders, if $X \leq y$ for some y, then $lub(X) = glb(\{ y \in A_\alpha^* \mid X \leq y\})$. Thus, we assured in certain circumstances of the existence of least upper bounds. Let us now look at ways to compute them. First, some notation. We shall write $x \cup y$ for $lub(\{x, y\})$ and $x \cap y$ for $glb(\{x, y\})$.

PROPOSITION 2.1.18. (Existence of *lubs*) Let x, y ∈ A_α*, then $x \cup y$ exists ⟺ x′ *ind* y′, where $x' = x/(x \cap y)$ and $y' = y/(x \cap y)$. Furthermore,

$$x \cup y = (x \cap y).x'.y' = x.y' = y.x' \qquad [4]$$

□

The proof of proposition 2.1.18 depends on the facts that $x/z \cap y/z = (x \cap y)/z$, $x/z \cup y/z = (x \cup y)/z$, and on the following lemma.

LEMMA 2.1.19. Suppose x, y, z ∈ A_α* with x, y ≤ z, then x *ind* y ⟺ $x \cap y = \Omega$.

□

It follows from equation [4] that if $x \cup y$ is defined, then for each i, $(x \cup y)(i) = x(i) \cup y(i)$. This turns out to hold generally, as we see in the next lemma, and shows one advantage of vectors over traces: lubs may be calculated in a straightforward manner.

LEMMA 2.1.20. Let $X \subseteq A_\alpha$*, then *lub*(X) exists and equals x ⟺

$\forall i \in I$: $x(i) = lub(\{y(i) \mid y \in X\})$

□

Note that the analogous statement for greatest lower bounds is not true in general. For example, the coordinatewise *glb* of $(a.b, b.d)$ and $(a.b, c.b.d)$ is $(a.b, \Omega)$, but $(a.b, b.d) \cap (a.b, c.b.d) = (a, \Omega)$.

Before we enter the next stage, the examination of prime algebraicity, we present the following useful result.

LEMMA 2.1.21. (Factorisation Lemma) Suppose that x, y, z ∈ A_α*, then

$x \le y.z \Rightarrow \exists$ u, v ∈ A_α*: u.v = x & u ≤ y & v ≤ z & v *ind* y/u.

Consequently, y.z = x.(y/u).(z/v).

□

We now wish to show that left-closed vector languages are prime algebraic and consistently complete. Obviously, it would be a great help if we could say what the complete primes were. We can get a clue about this by considering what the complete primes of behavioural presentations and string languages are. The former, you will remember, are points $\downarrow o$, where $o \in O$, that is, points which are preceded by a unique final simultaneity set $[o]_{=}$. Now, we don't have any simultaneity in vector or trace languages, so that it is a question of behaviours with a unique final action. This prompts the following definition. A prime in a vector with a unique last action.

DEFINITION 2.1.22. (Primes) We define the set of primes of A_α*, written $Pr(A_\alpha^*)$ as the set of all vectors x such that

$\forall y', y'' \in A_\alpha^* \ \forall a', a'' \in A: y'.a' = x = y''.a'' \Rightarrow y' = y'' \ \& \ a' = a''$

If $x \in A_\alpha^*$ then define $Pr(x) = \{y \in Pr(A_\alpha^*) \mid y \le x\}$. If $L \subseteq A_\alpha^*$, then define $Pr(L) = Pr(A_\alpha^*) \cap L$.

Before we explain why the elements of $Pr(A_\alpha^*)$ *are* the complete primes of A_α*, we give a useful construction, which, among other things, allows the elements of $Pr(x)$ to be computed.

LEMMA 2.1.23. Suppose $x \in A_\alpha^*$ and $a \in A$, then there exists a unique u ≤ x such that a *ind* x/u and $u.a \in Pr(A_\alpha^*)$. Define $pr(x, a) = u.a$.

Consequently, $Pr(x) = \{pr(v, a) \mid v.a \le x\}$.

□

Essentially, the proof that the elements of $Pr(A_\alpha^*)$ are prime is by induction. If *lub*(X) exists, then X must be finite, by 2.1.17, and we prove $u \le lub(X) \Rightarrow u \le x$, some $x \in X$ by induction on $|X|$ using the following lemma.

LEMMA 2.1.24. Suppose $\mathbf{u} \in Pr(A_\alpha{}^*)$ and $\mathbf{z}_1, \mathbf{z}_2 \in A_\alpha{}^*$ then $\mathbf{u} \le \mathbf{z}_1 \cup \mathbf{z}_2 \Rightarrow \mathbf{u} \le \mathbf{z}_1$ *or* $\mathbf{u} \le \mathbf{z}_2$.

□

COROLLARY 2.1.25. Each element of $Pr(A_\alpha{}^*)$ is prime in $(A_\alpha{}^*, \le)$.

□

We next wish to show that the elements of $Pr(A_\alpha{}^*)$ are the *only* complete primes of $A_\alpha{}^*$. The key to this is

$$\forall \mathbf{x} \in A_\alpha{}^*: \mathbf{x} = lub(Pr(\mathbf{x})) \qquad [5]$$

for we may then argue that if $\mathbf{x}$ is prime, then from $\mathbf{x} \le lub(Pr(\mathbf{x}))$ we may infer that $\mathbf{x} \le \mathbf{u}$ for some $\mathbf{u} \in Pr(\mathbf{x})$. But $\mathbf{u} \le \mathbf{x}$, by definition, and so $\mathbf{x} = \mathbf{u}$. Thus, $\mathbf{x} \in Pr(A_\alpha{}^*)$. [5] now allows us to conclude that $(A_\alpha{}^*, \le)$ is prime algebraic. We accordingly state:

THEOREM 2.1.26. $(A_\alpha{}^*, \le)$ is prime algebraic and consistently complete, with $Pr(A_\alpha{}^*)$ as its complete primes.

□

Of course, we shall generally be concerned with *subsets* of $A_\alpha{}^*$ rather than the whole monoid. We need the following notions.

THEOREM 2.1.27. If L is left-closed in $A_\alpha{}^*$ with respect to $\le$, then $(L, \le)$ is prime algebraic and consistently complete. The primes of L are the elements of $Pr(L) = Pr(A_\alpha{}^*) \cap L$.

□

2.2. Linguistic Behavioural Presentations

Theorem 2.1.27 gives us the authority to construct behavioural presentations from left-closed trace languages. We have indeed seen (in theorem 1.2.19) that any prime algebraic and consistently complete poset is isomorphic to some behavioural presentation, where the events of the presentation are in bijection with the complete primes of the poset. Furthermore, since each prime in a left closed trace language has a unique last element, we have a means of associating each prime with an event of the language. It is therefore fairly natural to make the following construction, which, as we shall see shortly, gives rise to an asynchronous discrete behavoural presentation. For the rest of this section, L denotes a left-closed subset of $A_\alpha{}^*$.

DEFINITION 2.2.1. Define $B_L = (O_L, P_L, E_L, \lambda_L)$ where:

(1) $O_L = Pr(L)$;

(2) $P_L = \{Pr(\mathbf{x}) \mid \mathbf{x} \in L\}$;

(3) $E_L = A$;

(4) $\lambda_L(\mathbf{x}) = a$, where $a \in A$ is unique with respect to $\mathbf{x} = \mathbf{y}.a$, some $\mathbf{y}$.

If $\mathbf{u} \in O_L$, then $\mathbf{u} \in Pr(\mathbf{u})$. It follows that $O_L \subseteq \bigcup_{\mathbf{x} \in L} Pr(\mathbf{x})$ and so B_L is a behavioural presentation. Indeed

PROPOSITION 2.2.2. B_L is a discrete, asynchronous behavioural presentation.

□

Of course, we would like to know what sort of discrete, asynchronous behavioural presentations B_L is. Is every discrete asynchronous behavioural presentation isomorphic to B_L, some L? Unfortunately, life isn't as simple as that; the independence relation ι_α has some part in the construction of B_L, and one would expect that this would manefest itself in the structure. In particular, one would imagine that ι_α determines which occurrences may be concurrent. This is in fact the case.

LEMMA 2.2.3. $\forall \mathbf{u}_1, \mathbf{u}_2 \in O_L: \mathbf{u}_1\ co_L\ \mathbf{u}_2 \Rightarrow \lambda_L(\mathbf{u}_1)\ \iota_\alpha\ \lambda_L(\mathbf{u}_2)$.

□

So, ι_α does have some relationship to the structure of B_L; two occurrences may be concurrent only if their labels are independent. Is the converse true? Not necessarily. It is possible, for example, that $\lambda_L(\mathbf{u}_1)\ \iota_\alpha\ \lambda_L(\mathbf{u}_2)$, but that some third party gets in the way in the sense that $\exists\ \mathbf{u}' \in O_L: (\mathbf{u}_1 < \mathbf{u}' < \mathbf{u}_2\ or\ \mathbf{u}_2 < \mathbf{u}' < \mathbf{u}_1)$. Let us pause to name such situations, in which tiresome creatures like $\mathbf{u}'$ are not present.

DEFINITION 2.2.4. Let B be a behavioural presentation and let $o_1, o_2 \in O$. We shall say that they are *unseparated*, and write o_1 *unsep* o_2, if

$\neg\, o_1 \# o_2\ \&\ \neg \exists o'' \in O: (o < o'' < o'\ OR\ o' < o'' < o)$

LEMMA 2.2.5. $\forall\, \mathbf{u}_1, \mathbf{u}_2 \in O_L: \mathbf{u}_1\ unsep\ \mathbf{u}_2 \Rightarrow \lambda_L(\mathbf{u}_1)\ \iota_\alpha\ \lambda_L(\mathbf{u}_2) \Rightarrow \mathbf{u}_1\ co_L\ \mathbf{u}_2$.

□

Lemmas 2.2.3 and 2.2.5 characterise behavioural presentations which derive from trace or vector languages as far as constraints on labellings are concerned. There is one further characteristic property we need to consider.

DEFINITION 2.2.6. Let B be a behavioural presentation and suppose $p_1, p_2 \in P$, then p_1 and p_2 are *isomorphic*, and we write $p_1 \equiv p_2$, if there exists a bijection $\phi: p_1 \rightarrow p_2$ such that

(1) $\forall\, o_1, o_2 \in p_1: o_1 \rightarrow o_2 \Longleftrightarrow \phi(o_1) \rightarrow \phi(o_2)$

(2) $\forall\, o \in p_1: \lambda(o) = \lambda(\phi(o))$

B will be said to be λ-reduced $\Longleftrightarrow \forall\, p_1, p_2 \in P: p_1 \equiv p_2 \Rightarrow p_1 = p_2$.

LEMMA 2.2.7. B_L is λ-reduced.

□

This prompts the following definition

DEFINITION 2.2.8. (ι-linguistic Behavioural Presentations) Let B be a behavioural presentation and ι an independence relation on E, then B will be said to be ι-*linguistic* $\Longleftrightarrow$

(1) B is discrete and asynchronous;

(2) If $\forall\, o_1, o_2 \in O: o_1\ co\ o_2 \Rightarrow \lambda(o_1)\ \iota\ \lambda(o_2)$;

(3) $\forall\, o_1, o_2 \in O: o_1\ unsep\ o_2 \Rightarrow (\lambda(o_1)\ \iota\ \lambda(o_2) \Rightarrow o_1\ co\ o_2)$;

(4) B is λ reduced.

We thus have the following characterisation of behavioural presentations which derive from left-closed trace languages.

THEOREM 2.2.9. Suppose L is left-closed in $A_\alpha{}^*$, then B_L is an ι-linguistic behavioural presentation.

Proof By proposition 2.2.2 and lemmas 2.2.3, 2.2.5 and 2.2.7.

□

To complete the section, we explain how left-closed trace or vector languages are images of ι-linguistic behavioural presentations.

Suppose B is discrete and asynchronous, and let $p \in P$. Let $\rho(p)$ be defined to be the set of strings of E^* as follows. $x \in \rho(p) \Longleftrightarrow$ there exist $o_1, \ldots, o_n \in O$ such that

(1) $x = \lambda(o_1) \ldots \lambda(o_n)$;

(2) $p = \{o_1, \ldots, o_n\}$;

(1) $\forall\, i, j: o_i < o_j \Rightarrow i < j$.

Thus, $\rho(p)$ is the set of all sequential interleavings of p.

PROPOSITION 2.2.10. If B is ι-linguistic, then $\rho(P) \subseteq E_\iota{}^*$ and $L = \rho(P)$ is a left closed trace language. Furthermore, considered as a map $\rho: P \rightarrow L$, ρ is a poset isomorphism.

□

Thus, one may move in both directions between models. Up to isomorphism, all information is preserved.

PROPOSITION 2.2.11.

(1) Suppose B is an ι-linguistic behavioural presentation, then $B \equiv B_{\rho(P)}$.

(2) Suppose L is left closed in $A_\iota{}^*$, then $\forall x \in L$: $\rho(Pr(x)) = x$. Consequently, $\rho(P_L) = L$.

□

We have finally accummulated all the material we need for the last main result of this chapter, which sums up the relationships that we have ascertained between trace/vector languages and behavioural presentations. First, let us say that B is *linguistic* $\Longleftrightarrow$ B is ι linguistic for some independence relation ι. A linguistic behavioural presentation could be ι linguistic for several ι. However, we note that $\rho(P)$ does not, in fact, depend on the ι in question, since $\rho(p) = \overline{\lambda}(\rho(p))$ and both functions are defined independently from ι.

THEOREM 2.2.12. (Representation Theorem) Let **IBP** denote the class whose elements are all the $\equiv$ equivalence classes of inguistic behavioural presentations. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let **LCTL** denote the class of all left-closed trace languages.

Then there exists a bijection ρ: **IBP** $\rightarrow$ **LCTL** such that $\rho([\![B]\!]) = L \Longleftrightarrow \rho(P) = L$.

□

2.3. Infinite Trace and Vector Lanaguges

In section 1.3 we examined the relationships between behavioural presentations and events structures and discovered that event structures of the kind described in [4] and [1] correspond to behavioural presentations which are closed. In this chapter we extend the connection between discrete behavioural presentations and vector languages to deal with closures of discrete behavioural presentations.

Such infinite traces or vectors are useful for the treatment of *liveness properties* of systems [8].

Concepts of liveness and safety find their most elegant treatment in temporal logic and some temporal logics use models involving infinite sequences. The phrase 'eventually property P will hold', which is typical of liveness properties, may be formally interpreted as saying that in all maximal (and in some cases therefore infinite) behaviours, the property P holds at some point.

First, we shall examine the business of completing trace and vector languages. This is largely analogous to the case of string languages (see, for example [9]). In completing a string language, one adjoins to it its *adherence*, the set of limits of monotonics ascending chains. We shall do much the same with our asynchronous languages, except that we also need to add in lubs of finite sets of elements. (This is not necessary with string languages; if $X \subseteq A^*$ is a finite set and $lub(X)$ exists, then $lub(X) \in X$).

We shall find it convenient to use vectors rather than traces because we shall be very much concerned with order-theoretic properties and operations, which for the most part work coordinate-wise for vectors. This is simpler than using equivalence classes of infinite strings, where things can be messy.

Let us set out the basic terminology for infinite string languages.

DEFINITION 2.3.1. (Infinite Strings) Let A be a set. The set of infinite strings over A, denoted A^ω is the set of all functions $x: N^+ \rightarrow A$, where N^+ denotes the set of all non-zero natural numbers. If $x \in A^\omega$, define $lnth(x) = \infty$, where ∞ has its usual meaning. We define $A^\infty = A^* \cup A^\omega$. If $x \in A^\omega$ and $i \in N^+$, then we write x_i for $x(i)$ and use the notation $x_1x_2\ldots$ to represent x.

We now extend the order relation on finite strings to elements of A^∞.

DEFINITION 2.3.2. (Ordering A^∞) Let $x, y \in A^\infty$, then we define

$x \le y \Longleftrightarrow lnth(x) \le lnth(y)$ & $\forall i \le lnth(x)$: $x_i = y_i$

$\le$ is obviously a partial order on A^∞. If $x, y \in A^*$, then $\le$ has its usual meaning. The following theorem shows that $(A^\infty, \le)$ belongs to a class of partial orders with which we are familiar.

THEOREM 2.3.3. $(A^{\infty}, \leq)$ is prime algebraic and coherent poset. The complete primes of A^{∞} are the elements of $A^* - \{\Omega\}$.

□

We shall now construct the vector equivalent of A^{∞}. To do this, we define mappings $\pi_i: A^* \to \alpha(i)^*$ and a mapping $\pi_\alpha: A^* \to M_\alpha{}^*$ such that for $a \in A, x, y \in A^*$ and $i \in I$,

$$\pi_i(a) = \begin{cases} a \text{ if } a \in \alpha(i) \\ \Omega \; otherwise \end{cases} \quad \pi_i(x.y) = \pi_i(x).\pi_i(y) \quad \pi_\alpha(x)(i) = \pi_i(x)$$

It is possible to show that $\pi_\alpha^{-1}(x) \in A_\iota{}^*$, for each $x \in A_\alpha{}^*$. (This is at the heart of the proof of theorem 2.1.14). We shall construct infinite vectors by extending π_α to a mapping $\pi_\alpha^{\infty}: A^{\infty} \to \times_{i \in I} \alpha(i)^{\infty}$.

First we note that the functions $\pi_i: A^* \to \alpha(i)^*$ are all monotonic. Secondly, observe that if $X \subseteq A^*$ is pairwise compatible in A^*, then $\pi_i(X)$ is pairwise compatible in $\alpha(i)^*$. We can use these two facts to define functions $\pi_i^{\infty}: A^{\infty} \to \alpha(i)^{\infty}$. Let $x \in A^{\infty}$. By 2.3.3, $Pr(x) = \{u \in A^* - \{\Omega\} \mid u \leq x\}$. $Pr(x)$ is a total order and is thus clearly pairwise compatible. Hence, $\pi_i(Pr(x))$ is pairwise compatible. Since $\alpha(i)^{\infty}$ is coherent by theorem 2.3.3, it follows that $lub(\pi_i(Pr(x)))$ exists. Define

$$\pi_i^{\infty}(x) = lub(\pi_i(Pr(x))). \qquad [6]$$

Note that this definition entails that π_i^{∞} coincides with π_i on A^*. Another useful property of π_i^{∞} is the following.

LEMMA 2.3.4. (Continuity of π_i^{∞}) π_i^{∞} is continuous, that is

$$\forall X \subseteq A^{\infty}: lub(X) \in A^{\infty} \Rightarrow \pi_i^{\infty}(lub(X)) = lub(\pi_i^{\infty}(X))$$

□

We are now in a position to define infinite vectors. The ones that extend our vector monoids $A_\alpha{}^*$ will be defined to be the vector of π_i^{∞} projections of infinite strings.

DEFINITION 2.3.5. (Infinite Vectors) Define M_α^{∞} to be the set of all functions $x: I \to \bigcup_{i \in I} \alpha(i)^{\infty}$ such that for all $i \in I$, $x(i) \in \alpha(i)^{\infty}$. Define $\pi_\alpha^{\infty}: A^{\infty} \to M_\alpha^{\infty}$ by $\pi_\alpha^{\infty}(x)(i) = \pi_i^{\infty}(x)$. Finally, define $A_\alpha^{\infty} = \pi_\alpha^{\infty}(A^{\infty})$.

Note that $A_\alpha{}^* \subseteq A_\alpha^{\infty}$.

A_α^{∞} is the set of vectors that we shall use. Let us now investigate its order theoretic properties. First, we define the order relation.

DEFINITION 2.3.6. (Ordering A_α^{∞}) Suppose $x, y \in A_\alpha^{\infty}$. Define $x \leq y \Leftrightarrow \forall i \in I: x(i) \leq y(i)$.

Note that $\leq$ on A_α^{∞} agrees with $\leq$ on $A_\alpha{}^*$.

For instance, $a.b^{\infty} \leq (a.b)^{\infty}$ in the example above. Since $a.b^{\infty} \neq (a.b)^{\infty}$, we have $a.b^{\infty} < (a.b)^{\infty}$. This shows that in contrast with strings, 'infinite' behaviours need not be maximal. It also shows that in moving from non-interleaving to interleaving representations of behaviour, we lose information as to which infinite behaviours are maximal and which are not.

As usual, we are interested in the existence of least upper bounds. Our first result shows that these always exist for sets which constitute monotonic increasing chains of vectors of finite length.

LEMMA 2.3.7. Suppose $(x_n)_{n \in N^+}$ is a monotonic increasing chain in $A_\alpha{}^*$, then $x = lub(x_n)_{n \in N^+}$ exists in A_α^{∞} and $\forall i \in I: x(i) = lub(x_n(i))_{n \in N^+}$.

□

Let us now elucidate the order structure of A_α^{∞}. We begin by showing that if I is countable, then A_α^{∞} is a complete partial order (cpo). For those few readers who have managed to escape exposure to denotational semantics, here are the pertinent definitions.

DEFINITION 2.3.8. (Complete Partial Orders) Suppose $(D, \leq)$ is a poset and $X \subseteq D$, then D is *directed* $\Leftrightarrow X \neq \emptyset$ and $\forall x, y \in X \; \exists \in X: x \leq \; \& \; y \leq z$.

$(D, \leq)$ is a *complete partial order* (cpo) $\Longleftrightarrow D$ has a bottom element and every directed $X \subseteq D$ has a lub.

Since directed sets are pairwise compatible, every coherent poset is a cpo.

PROPOSITION 2.3.9. If I is countable, then A_α^∞ is a cpo.

□

Observe that the hypothesis that I is countable cannot be dispensed with in general. However, if I is countable, A_α^∞ is not only a cpo; we shall show that it is coherent. The key to the proof is the observation that if we take a pairwise compatible set and adjoin all least upper bounds of its finite subsets, then the resulting set is directed and has the same lub. We call this construction 'finite lub closure'.

DEFINITION 2.3.10. (Finite *lub* Closure) Suppose $X \in A_\alpha^\infty$. Its *finite lub closure*, denoted $ficl(X)$ is defined

$ficl(X) = \{\mathbf{x} \in A_\alpha^\infty \mid \exists Y \subseteq X : |Y| < \infty \ \&\ lub(X) = \mathbf{x}\}$

It is clear that $lub(X)$ exists $\Longleftrightarrow lub(ficl(X))$ exists and that in either case the two are equal.

As promised, we have the following lemma, which will allow us to use proposition 2.3.9 to show that $(A_\alpha^\infty, \leq)$ is coherent.

LEMMA 2.3.11. Suppose $X \subseteq A_\alpha^\infty$ is nonempty, then X is pairwise compatible $\Rightarrow ficl(X)$ is directed.

□

It follows that if $X \subseteq A_{\alpha_\infty}$ is pairwise compatible, then $ficl(X)$ is directed and therefore possesses a lub, by proposition 2.3.9. But $lub(ficl(X)) = lub(X)$. Thus pairwise compatible sets have lubs, that is:

COROLLARY 2.3.12.

If I is countable, then A_α^∞ is coherent.

□

By this time, the reader should be used to meeting coherence or consistent completeness in the company of another property. Sure enough, we have:-

PROPOSITION 2.3.13. Suppose I is countable, then A_α^∞ is prime algebraic. The complete primes are the elements of $Pr(A_\alpha{}^*)$.

□

To sum up, from corollary 2.3.12 and proposition 2.3.13, we conclude that our vectors have the characteristic properties of closed behavioural presentations. Accordingly, we have the following generalisation of theorem 2.3.3.

THEOREM 2.3.14. (Order Theoretic Properties of A_α^∞)

Suppose I is countable, then A_α^∞ is prime algebraic and coherent. The complete primes are the elements of $Pr(A_\alpha{}^*)$.

□

We now have the environment for closures of vector languages.

DEFINITION 2.3.15. (*moncl*(L) and Closure of Vector Languages) Suppose I is countable. Let $L \subseteq A_\alpha^\infty$. Define *moncl*$(L)$ to be the set of all $\bigcup_{n=1}^{\infty} \mathbf{x}_n$, where $\mathbf{x}_n \subseteq L$ is a monotonic ascending chain. (These lubs exist, since A_α^∞ is coherent and $\{\mathbf{x}_n \mid n = 1, \ldots, \infty\}$ is pairwise compatible).

Let $L \subseteq A_\alpha^\infty$. Define the *closure* of L, $\overline{L}$, to be the set *moncl*$(ficl(L))$.

The language $L \subseteq A_\alpha$ is *closed* $\Longleftrightarrow L = \overline{L}$.

The closure of a language ought to be closed itself.

PROPOSITION 2.3.16. Suppose I is countable and let $L \subseteq A_\alpha^\infty$, then $\overline{\overline{L}} = \overline{L}$.

□

By definition, a closed language is the closure of some language. Actually, we can show that every closed language is the closure of a *finite* language.

PROPOSITION 2.3.17. Suppose I is countable and let L be left closed in A_α^∞, then L is closed $\Leftrightarrow$ $L = \overline{L^{fin}}$, where $L^{fin} = L \cap A_\alpha{}^*$.

□

The next proposition shows that a left-closed and closed language has the characteristic property of an event structure or closed behavioural presentation.

PROPOSITION 2.3.18. Suppose I is countable and let L be left closed in $A_\alpha{}^*$, then $\overline{L}$ is prime algebraic and coherent. The complete primes are the elements of $Pr(L) = Pr(A_\alpha{}^*) \cap L$.

□

Given that a closed, left-closed L is prime algebraic and coherent (and hence consistently complete), it ought, by now, to be almost a reflex action to construct a behavioural presentation from it. The construction is quite straightforward. Suppose I is countable and let L be left closed in A_α^∞. We define $B_L = (O_L, P_L, E_L, \lambda_L)$, where, as usual,

$O_L = Pr(L), P_L = \{Pr(\mathrm{x}) \mid \mathrm{x} \in L\}, E_L = A$ and $\lambda_L(\mathrm{x.a}) = a$.

Since $\mathbf{u} \in Pr(\mathrm{u})$ for $\mathbf{u} \in Pr(A_\alpha{}^*)$, we may conclude that B_L satisfies condition (2) of definition 1.1.1 and is hence a behavioural presentation. What else can we find out about it?

One observation we may make about B_L is that it has a discrete sub behavioural presentation. Indeed, let $L^{fin} = L \cap A_\alpha{}^*$. L^{fin} is easily seen to be left closed (since L is) and thus determines an ι_α-linguistic behavioural presentation $B_{L^{fin}}$. Checking through the definitions, it may be seen that $B_{L^{fin}} \subseteq B_L$. We can say more than this, however. Define L to be *finite lub closed* $\Leftrightarrow$ $L = ficl(L)$.

PROPOSITION 2.3.19. Suppose I is countable and let L be left closed and finite lub closed in $A_\alpha{}^*$, then $\overline{B_L} = B_{\overline{L}}$. Consequently, if L' is closed and left closed in A_α^∞, then $\overline{B_{L'^{fin}}} = B_{L'}$.

□

One question that naturally arises is whether we can generalise the Representation Theorem (2.2.17) to the infinite case. In view of proposition 2.3.19, we need behavioural presentations B such that the 'finite' part of B is ι-linguistic - so that we have our mapping ρ - and so that B is the closure of its finite part - so that each element of B will be the lub of finite primes. We also need the hypothesis of I countable, so that the results we already have may be applied. Let us collect the definitions together.

DEFINITION 2.3.20. ($\omega\iota$-Linguistic Behavioural Presentations) Suppose B is a behavioural presentation. We define $B^{fin} = (O^{fin}, P^{fin}, E^{fin}, \lambda^{fin})$, where

$P^{fin} = \{p \in P \mid |p| < \infty\}, O^{fin} = \bigcup_{p \in P^{fin}} p$ and $E^{fin} = E$ and $\lambda^{fin} = \lambda|^{O^{fin}}$

Let ι be an independence relation on E. We will say that ι has *countable dimension* $\Leftrightarrow$ $\iota = \iota_\alpha$ for some $\alpha: I \to 2^E$ such that $|I|$ is countable.

Let B be a behavioural presentation, then B will be said to be $\omega\iota$-*linguistic* $\Leftrightarrow$ $B = \overline{B^{fin}}$ and B^{fin} is an ι-linguistic behavioural presentation B' such that ι has countable dimension. From proposition 3.2.19, we obtain:

COROLLARY 2.3.21. Suppose I is countable and let L be closed and left closed in A_α^∞, then B_L is $\omega\iota_\alpha$- linguistic.

□

The easiest way to extend ρ from P^{fin} to P is to define $\rho^\infty(p) = lub(\rho(Pr(p)))$. Fortunately, the right lubs exist.

LEMMA 2.3.22. Let B be an ι_α-linguistic behavioural presentation and suppose $p \in \overline{P}$, then $\rho(Pr(p))$ is pairwise compatible in E_α^∞.

□

Lemma 2.3.22 and corollary 3.2.12 allow us to define ρ^{∞} as we wished and guarantee that it is a map $\rho^{\infty}: P \to E_{\alpha}^{\infty}$.

We may now state a generalisation of proposition 2.2.10.

PROPOSITION 2.3.23. Let B be an ι_{α}-linguistic behavioural presentation with α having countable dimension, then the poset isomorphism $\rho: P \to L = \rho(P) \subseteq E_{\alpha}{}^{*}$ extends to a unique poset isomorphism $\rho^{\infty}: \overline{P} \to \overline{L} \subseteq E_{\alpha}^{\infty}$.

□

We also have the following generalisation of proposition 2.2.11.

PROPOSITION 2.3.24. Suppose I is countable and let L be closed and left closed in A_{α}^{∞}, then B_L is $\omega\iota$- linguistic and $\rho^{\infty}(P_L) = L$.

□

Let us now tackle closures of trace languages. We shall approach the definition of $\equiv_{\iota}$ for infinite strings via a preordering relation. First, if $x, y \in A^*$, then we define $x \lesssim_{\iota} y \iff \langle x\rangle_{\iota} \leq \langle y\rangle_{\iota}$.

We note that $\lesssim_{\iota}$ is a preorder on A^* which determines $\equiv_{\iota}$ in the sense that $x \equiv_{\iota} y \iff x \lesssim_{\iota} y$ and $y \lesssim_{\iota} x$. We shall extend $\lesssim_{\iota}$ to A^{∞}. It determines an equivalence relation, which extends $\equiv_{\iota}$, which we shall take as our infinite trace congruence. To reassure us that this decision is a reasonable one, we shall see that the representation of traces by vectors (theorem 2.1.14) generalises to the infinite case. Let us define the pre-order.

DEFINITION 2.3.25. ($\lesssim_{\iota}$ on A^{∞}) Suppose $x, y \in A^{\infty}$. Define $x \lesssim_{\iota}^{\infty} y \iff$

$\forall u \in A^*: u \leq x \Rightarrow \exists v \in A^*: v \leq y \;\&\; u \lesssim_{\iota} v$

We note that if $x, y \in A^*$, then $x \lesssim_{\iota}^{\infty} y \iff x \lesssim_{\iota} y$ (take $v = u$). From now on, we shall abbreviate $\lesssim_{\iota}^{\infty}$ to $\lesssim_{\iota}$.

LEMMA 2.3.26. $\lesssim_{\iota}$ is a preorder on A^{∞} and determines an equivalence relation $\equiv_{\iota}$ on A^{∞}. Let $\langle x\rangle_{\iota}$ denote the $\equiv_{\iota}$ class of $x \in A^{\infty}$ and let $A_{\iota}^{\infty} = \{\langle x\rangle_{\iota} \mid x \in A^{\infty}\}$. We have $A_{\iota}{}^{*} \subseteq A_{\iota}^{\infty}$. The elements of A_{ι}^{∞} will be called *infinite traces*.

Also, $\equiv_{\iota}$ is a congruence for $\lesssim_{\iota}$, that is, there is a partial order on A_{ι}^{∞}, which we shall also call $\lesssim_{\iota}$, such that $\forall x, y \in A^{\infty}: \langle x\rangle_{\iota} \lesssim_{\iota} \langle y\rangle_{\iota} \iff x \lesssim_{\iota} y$ and this order agrees with that on $A_{\iota}{}^{*}$.

□

The reader has probably guessed what *kind* of partial order $\lesssim_{\iota}$ is - prime algebraic and coherent, like all the others in this chapter. The simplest way to prove this is establish a vector representation theorem like theorem 2.1.14. Then we can simply appeal to the order theoretic properties of A_{α}^{∞} that we know about already.

Given $\langle x\rangle_{\iota} \in A_{\iota}^{\infty}$, we would like to be able to associate it with an element of A_{α}^{∞}. Which one? If x were a finite string, then we know the answer from theorem 2.1.14, it would be $\phi_{\alpha}(\langle x\rangle_{\iota}) = \pi_{\alpha}(x)$. Now, we just happen to have an extension of π_{α} hanging around, namely π_{α}^{∞} (equation [6]). Accordingly, we would *like* to define $\phi_{\alpha}^{\infty}(\langle x\rangle_{\iota}) = \pi_{\alpha}^{\infty}(x)$. Of course, we need to show that $\phi_{\alpha}^{\infty}(\langle x\rangle_{\iota})$ doesn't depend on the choice of x in $\langle x\rangle_{\iota}$. This will follow from lemma 2.3.27.

LEMMA 2.3.27. Suppose α is a countable cover of A, then

$\forall x, y \in A^{\infty}: x \lesssim_{\iota_{\alpha}} y \iff \pi_{\alpha}^{\infty}(x) \leq \pi_{\alpha}^{\infty}(y)$

□

Thus, $x \equiv_{\iota_{\alpha}} y \iff x \lesssim_{\iota_{\alpha}} y \;\&\; y \lesssim_{\iota_{\alpha}} x \iff \pi_{\alpha}^{\infty}(x) \leq \pi_{\alpha}^{\infty}(y) \;\&\; \pi_{\alpha}^{\infty}(y) \leq \pi_{\alpha}^{\infty}(x) \iff \pi_{\alpha}^{\infty}(x) = \pi_{\alpha}^{\infty}(y)$. Hence, our tentative definition of an embedding function does work. Not only that, but lemma 2.3.27 shows that the function in question is a poset isomorphism.

THEOREM 2.3.28. (Representation of Infinite Traces by Vectors) Suppose $\iota = \iota_{\alpha}$, where α has countable dimension, then the map $\pi_{\alpha}^{\infty}: A^{\infty} \to A_{\alpha}^{\infty}$ determines a bijective mapping $\phi_{\alpha}^{\infty}: A_{\iota}^{\infty} \to A_{\alpha}^{\infty}$ by

$\phi_\alpha^\infty(<x>_\iota) = \pi_\alpha^\infty(x)$. ϕ_α^∞ is a poset isomorphism which agrees with ϕ_α on $A_\iota{}^*$.

□

COROLLARY 2.3.29. Suppose ι has countable dimension, then $(A_\iota^\infty, \leq_\iota)$ is prime algebraic and coherent. The complete primes are the elements of $Pr(A_\iota{}^*)$.

□

One advantage of traces over vectors is that there will be only one trace language corresponding to each $\equiv$ class of behavioural presentations. There are a plethora of vector languages for the same behavioural presentation - one for each appropriate α.

Suppose B is $\omega\iota$ linguistic and let α be countable, with $\iota = \iota_\alpha$. We may construct a trace language $(\phi_\alpha^\infty)^{-1}(\rho^\infty(P))$. The mapping $(\phi_\alpha^\infty)^{-1}\varphi^\infty$ thus maps behavioural presentations to trace languages.

Of course, this particular mapping seems to depend on the choice of α. If α' is another cover, then we can also construct a mapping of P into E_ι^∞, $(\phi_{\alpha'}^\infty)^{-1}\varphi_{\alpha'^\infty}$. There would be little point in this discussion, however, if the two mappings were different.

LEMMA 2.3.30. Suppose α, α' are covers of countable dimension such that $\iota_\alpha = \iota_{\alpha'}$, then $(\phi_\alpha^\infty)^{-1}\varphi^\infty = (\phi_{\alpha'}^\infty)^{-1}\varphi_{\alpha'}^\infty$.

□

Define $\rho_\iota^\infty = (\phi_\alpha^\infty)^{-1}\varphi_\alpha^\infty$, where α is a cover of countable dimension such that $\iota = \iota_\alpha$. By lemma 2.3.30, ρ_ι^∞ is well defined. We may now state a generalisation of theorem 2.2.17.

THEOREM 2.3.31. (Representation of ω Linguistic Behavioural Presentations by Traces) Let ωLBP denote the class whose elements are all $\equiv$ equivalence classes of behavioural presentations which are $\iota\omega$-linguistic for some ι. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let ωLCTL denote the class of all left-closed and closed trace languages with finite dimensional independence relations.

Then there exists a bijection ρ^∞: ωLBP $\rightarrow$ ωLCTL such that $\rho^\infty([\![B]\!]) = L \iff \rho_\iota^\infty(P) = L$.

□

We have concentrated on order theoretic properties of A_α^∞, mainly because they seem to predominate over the monoid aspects when we go to the limit. The coordinatewise definition would not work in general, so concatenation is a partial operation. We shall get at its definition via an extension of the left cancellation operator.

DEFINITION 2.3.32. (Left Cancellation in A_α^∞) Suppose $x, y \in A^\infty$ with $x \leq y$. If $x = y$, then we define $y/x = \Omega$. Otherwise $x < y$ and so x is finite, say $lnth(x) = n$. Define $(y/x)(m) = y(m+n)$.

If $x, y \in A_\alpha^\infty$ with $x \leq y$, then define y/x by

$\forall i \in I$: $(y/x)(i) = y(i)/x(i)$

We observe, as a consequence of proposition 2.1.13, that the definition of / on A_α^∞ agrees with the definition of / on $A_\alpha{}^*$.

THEOREM 2.3.40. Suppose I is countable, then

$\forall x, y \in A_\alpha^\infty : x \leq y \Rightarrow y/x \in A_\alpha^\infty$.

□

We may now treat concatentation.

DEFINITION 2.3.41. (Concatenation in A_α^∞) Suppose $x, y \in A_\alpha^\infty$, then x.y is defined and equal to z $\iff$ $z \in A_\alpha^\infty$ and $x \leq z$ and $y = z/x$.

By a coordinatewise argument, it may be shown that if $x \leq y$ and $y = z_1/x$ and $y = z_2/x$ then $z_1 = z_2$. Thus the operation is well-defined. We also note that this definition agrees with the definition of concatenation on $A_\alpha{}^*$.

We conclude this section remarking that concatenation is associative where defined.

PROPOSITION 2.3.43. Suppose x, y, z $\in A_{\alpha}^{\infty}$, then if either of x.(y.z), (x.y).z is defined, then so is the other, and they are equal.

□

3. Generalised Automata

3.1. Sequential Behavioural Presentations and Transition Systems

In this chapter we examine three classes of automata, transition systems, asynchronous transition systems and hybrid transition systems. Each of these classes corresponds to a class of behavioural presentations, in the sense that to each automaton of a given class, there exists a behavioural presentation of the corresponding class which describes the 'behaviour' of the automaton from a given initial state. We begin with sequential systems.

DEFINITION 3.1.1. (Transition Systems [10]) A *transition system* is a triple $T = (Q, A, \rightarrow)$, where Q is a set of (global) *states*, A is a set of *actions* and $\rightarrow$ is a relation $\rightarrow \subseteq Q \times A \times Q$ called the *transition relation*. Write $q_1 \rightarrow^a q_2$ for $(q_1, a, q_2) \in \rightarrow$.

A transition system describes, in a very low-level way, the actions that some system may perform and how the performance of these actions transforms the system's (global) internal state. Formally, we have the notion of a set of *execution sequences* of the transition system from some initial state.

DEFINITION 3.1.2. (Execution Sequences) Let T be a transition system and let $q \in Q$. The set of execution sequences of T from q, denoted $L(T, q)$, is defined to be

$$L(T, q) = \{x \in A^* \mid \exists q' \in Q: q \rightarrow^x q'\}$$

where if $a \in A$ and $x \in A^*$, and $q, q' \in Q$, then

(1) $q \rightarrow^{\Omega} q$

(2) $q \rightarrow^{x.a} q' \Leftrightarrow \exists q'' \in Q: q \rightarrow^x q'' \;\&\; q'' \rightarrow^a q'$.

Define the set of states reachable from $q \in Q$, denoted $R(T, q)$, by

$$R(T, q) = \{q' \in Q \mid \exists x \in L(T, q): q \rightarrow^x q'\}$$

Any behavioural presentation determines a transition system - the states of the system are the elements of P and transitions are steps, as given in definition 1.2.5. Formally, we have the following construction.

DEFINITION 3.1.3. (Transition Systems from Behavioural Presentations) Let B be any asynchronous behavioural presentation. Define $T_B = (Q_B, A_B, \rightarrow_B)$, where $Q_B = P$, $A_B = E$ and

$$p_1 \rightarrow_B{}^e p_2 \Leftrightarrow \exists o \in O: p_1 \vdash^o p_2 \;\&\; \lambda(o) = e.$$ [5]

Of course, T_B is a transition system.

A given behavioural presentation may not have any steps at all. For example, the analogue system described in example 1.1.3 is of this nature. However, this problem does not arise with discrete behavioural presentations. Indeed, from lemma 1.2.6 we may infer that any replete behavioural presentation satisfying the DCC possesses steps.

As we have observed, for a discrete, sequential behavioural presentation B, T_B has its own behavioural semantics; from a given initial state, p, we may construct the language of execution sequences $L(T_B, p)$. B itself has an obvious 'initial state', namely $\emptyset$, and B describes the behaviours possible to some system from that initial state. What, then, is the relationship between B and $L(T_B, \emptyset)$?

In one direction, it is relatively straightforward. Let $p \in B$, then since B is discrete and sequential, p will be finite and totally ordered by $<$, so that $p = \{o_1, \ldots, o_n\}$ for some n, with $o_1 < \ldots < o_n$. The sequence of events performed before this time point may be represented by the string $\rho(p) = \lambda(o_1) \ldots \lambda(o_n)$. (If $p = \emptyset$, then $\rho(p) = \Omega$, the null string).

[5] Since B is asynchronous, if $p_1 \vdash^X p_2$, then $X = \{o\}$, some $o \in O$.

PROPOSITION 3.1.4. Suppose B is a discrete sequential behavioural presentation then

$\forall p \in P: \varnothing \to_B^x p \Longleftrightarrow x = \rho(p)$

and hence $L(T_B, \varnothing) = \rho(P)$.

□

We shall find it useful to cite some additional properties of ρ.

LEMMA 3.1.5.

(1) $\forall p_1, p_2 \in P: p_1 \subseteq p_2 \Rightarrow \rho(p_1) \leq \rho(p_2)$;

(2) If, in addition ρ is injective then $\forall p_1, p_2 \in P: p_1 \subseteq p_2 \Longleftrightarrow \rho(p_1) \leq \rho(p_2)$.

□

This establishes a relationship between B and $L(T_B, \varnothing)$, but what kind of relationship is it? We may obtain the latter from the former, but not the former from the latter. In fact, behavioural presentations may not be distinguished simply by the languages associated with them. The problem is that the mapping ρ need not be injective because the transition system may possess a property akin to *ambiguity* in phrase structure grammars.

This marks a limitation of formal languages as representatives of sequential behaviour.

Initially, we shall consider the class of behavioural presentations for which such problems do not arise. *This is precisely the class of behavioural presentations/systems for which a representation of behaviour by a string language is adequate.*

DEFINITION 3.1.7. (Unambiguity) A transition system T is unambiguous $\Longleftrightarrow$

$\forall q, q_1, q_2 \in Q \ \forall a \in A: q \to^a q_1 \ \& \ q \to^a q_2 \Rightarrow q_1 = q_2$

If B is *any* behavioural presentation, then B is unambiguous $\Longleftrightarrow$

$\forall p, p_1, p_2 \in Q \ \forall o_1, o_2 \in O: p \vdash^{o_1} p_1 \ \& \ p \vdash^{o_2} p_2 \ \& \ \lambda(o_1) = \lambda(o_2) \Rightarrow p_1 = p_2$

Clearly, if B is a discrete, sequential behavioural presentation, then B is unambiguous $\Longleftrightarrow$ T_B is unambiguous.

We also note (by a simple induction argument) that if T is an unambiguous transition system and $q \in Q$, then

$\forall q, q_1, q_2 \in Q \ \forall x \in L(T, q): q \to^x q_1 \ \& \ q \to^x q_2 \Rightarrow q_1 = q_2$

so that each execution sequence x from q determines a unique final state. In particular, if B is unambiguous, then each $x \in L(T_B, \varnothing)$ determines a unique $p \in P$ such that $\varnothing \to^x p$. Since $\rho(p) = x$ (by proposition 3.1.4), it follows that ρ is injective. We may thus appeal to lemma 3.1.5 to conclude that:

PROPOSITION 3.1.8. Suppose that B is unambiguous, then the map ρ is an isomorphism of the posets $(P, \subseteq)$ and $(\rho(P), \leq) = (L(T_B, \varnothing), \leq)$.

□

From this we may argue that an unambiguous behavioural presentation is determined up to isomorphism by an appropriate string language. (We still have to say what we mean by 'isomorphism', of course), and hence that if a transition system determines an unambiguous behavioural presentation, then that behavioural presentation is unique up to isomorphism.

DEFINITION 3.1.9. (Isomorphisms) Let B_1, B_2 be *any* behavioural presentations, then they are *label isomorphic*, and we write $B_1 \equiv_\varepsilon B_2$ if there exists bijective mappings $\omega: O_1 \to O_2$, $\pi: P_1 \to P_2$ and $\varepsilon: E_1 \to E_2$ such that

(1) $\forall o \in P_1: \varepsilon(\lambda_1(o)) = \lambda_2(\omega(o))$;

(2) $\forall p \in P_1: \pi(p) = \{\omega(o) \mid o \in p\}$.

If ε is the identity function, then we shall just call B_1 and B_2 isomorphic and write $B_1 \equiv B_2$. Note, however, that this property includes the requirement that $E_1 = E_2$.

Two behavioural presentations are label isomorphic if they are identical except for the names of occurrences and their events. Two behavioural presentations are isomorphic if they are identical except for the names of occurrences. If $\pi(p_1) = p_2$, then p_1 and p_2 describe *exactly the same orderings of event occurrences.*

It is also clear that

$$B_1 \equiv B_2 \Rightarrow \forall o_1, o_2 \in P_1: o_1 \to_1 o_2 \Longleftrightarrow \omega(o_1) \to_2 \omega(o_2)$$

$$B_1 \equiv B_2 \Rightarrow \forall o_1, o_2 \in P_1: o_1 \#_1 o_2 \Longleftrightarrow \omega(o_1) \#_2 \omega(o_2)$$

From this, we may easily deduce that $B_1 \equiv B_2 \Rightarrow \rho(P_1) = \rho(P_2)$. In fact:

PROPOSITION 3.1.10. Suppose B_1 and B_2 are unambiguous discrete sequential behavioural presentations then $\rho(P_1) = \rho(P_2) \Longleftrightarrow B_1 \equiv B_2$.

□

Proposition 3.1.10 tells us that if B is an unambiguous discrete sequential behavioural presentation and $L(T, q) = L(T_B, \varnothing)$, then B is unique - up to isomorphism - among unambiguous discrete sequential behavioural presentations. For, if $L(T, q) = L(T_{B'}, \varnothing)$ for a second unambiguous discrete sequential behavioural presentation B', then by proposition 3.1.4, $\rho(P) = L(T_B, \varnothing) = L(T_{B'}, \varnothing) = \rho(P')$ and so $B \equiv B'$, by proposition 3.1.10.

The question remain whether, given a transition system T and $q \in Q$ there actually exists a behavioural presentation B such that $L(T, q) = L(T_B, \varnothing)$. Since the latter equals $\rho(P)$, by 3.1.4, it all boils down to finding, for a given language L, a behavioural presentation B such that $\rho(P) = L$. Note that if $L = L(T, q)$ then L must be *left-closed* in E^* in the sense that

$$\forall x \in L \; \forall y \in E^*: y \leq x \Rightarrow y \in L$$

The following remark gives us the clue we need.

REMARK 3.1.11.

Suppose L is left-closed in E^*, then $(L, \leq)$ is prime algebraic and consistently complete. The complete primes are the elements of $L - \{\Omega\}$.

□

Referring back to section 1.2, we see that L determines an unlabelled behavioural presentation, with its primes as occurrences and sets $Pr(x)$, $x \in L$, as points. The labelling function is constructed to reflect the idea that primes have unique last elements. These label the occurrence corresponding to the prime.

DEFINITION 3.1.12. Suppose $L \subseteq E^*$ is left closed in E^*. Define B_L by

(1) $P_L = \{p_x \mid x \in L\}$, where $p_x = \{u \mid \Omega < u \leq x\}$;

(2) $O_L = L - \{\Omega\}$;

(3) $E_L = E$;

(4) Let $u \in L - \{\Omega\}$, then $u = u'.e$, some $e \in E$. Define $\lambda_L(u) = e$.

Since $x \in p_x$ for all $x \in O_L$, it follows that $O_L \subseteq \bigcup_{p_x \in P_L} p_x$ and hence that B_L is a behavioural presentation. The reader might care to refer back to 1.1.4 to see that the example given there is $B_{\{H,T\}^*}$.

B_L is the behavioural presentation we are loking for.

PROPOSITION 3.1.13. Suppose $L \subseteq E^*$ is left closed in E^*, then B_L is a discrete, unambiguous, sequential behavioural presentation and $\rho(P_L) = L$.

□

There remains the question: what is the relationship between transition systems plus initial states that 'accept' the same behavioural presentations? The answer is that they are bisimilar.

DEFINITION 3.1.14. (Bisimulation) Suppose T_1 and T_2 are transition systems and that $q_1 \in Q_1$ and $q_2 \in Q_2$. A *strong bisimulation* between (T_1, q_1) and (T_2, q_2) is a relation $R \subseteq Q_1 \times Q_2$ such that $q_1 \, R \, q_2$ and

$$\forall (q'_1, q'_2) \in R \; \forall q''_1 \in Q_1 \forall e \in E_1: q'_1 \to^e q''_1 \Rightarrow \exists q''_2 \in Q_2: q'_2 \to^e q''_2 \; \& \; q''_1 \, R \, q''_2$$

$$\forall (q'_1, q'_2) \in R \; \forall q''_2 \in Q_2 \forall e \in E_2: q'_2 \to^e q''_2 \Rightarrow \exists q''_1 \in Q_1: q'_1 \to^e q''_1 \; \& \; q''_1 \, R \, q''_2$$

We shall say that (T_1, q_1) and (T_2, q_2) are *strongly congruent* and write $(T_1, q_1) \sim (T_2, q_2) \Leftrightarrow$ there exists a strong bisimulation R between (T_1, q_1) and (T_2, q_2).

Clearly, $\sim$ is reflexive, symmetric and transitive.

LEMMA 3.1.15 Suppose that T_1 and T_2 are transition systems and that $q_1 \in Q_1$ and $q_2 \in Q_2$, then

(1) $(T_1, q_1) \sim (T_2, q_2) \Rightarrow L(T_1, q_1) = L(T_2, q_2)$

(2) $L(T_1, q_1) = L(T_2, q_2) \Rightarrow (T_1, q_1) \sim (T_2, q_2)$ if T_1 and T_2 are unambiguous.

□

We may sum up our findings about unambiguous sequential systems as follows:

THEOREM 3.1.16 Let USB denote the class whose elements are all the $\equiv$ equivalence classes of unambiguous, sequential, discrete behavioural presentations. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let UTS denote the class whose elements are all the $\sim$ equivalence classes of unambiguous transition systems with initial states. Write $[\![(T, q)]\!]$ for the $\sim$ class of (T, q).

Then there is a bijective mapping

Beh_U: UTS $\to$ USB

such that $Beh_U([\![(T, q)]\!]) = [\![B]\!] \Leftrightarrow \rho(P) = L(T, q)$.

Furthermore, $Beh_U([\![(T, q)]\!]) = [\![B]\!] \Leftrightarrow (T_B, \varnothing) \in [\![(T, q)]\!] \Leftrightarrow B \equiv B_{L(T, q)}$.

□

Let us now turn to the general case. We have seen, in definition 3.1.3, that behavioural presentations determine transition systems. However, to obtain an isomorphic copy of the behavioural presentation back from the transition system via its language of execution sequences is not possible in general. The problem is that there may be an execution sequence which corresponds to more than one route through the transition system.

This ambiguity is removed, however, if we include information about states. By attaching the name of the state to the name of the event, we remove the ambiguity. We therefore make the following construction.

DEFINITION 3.1.17. Let $T = (Q, A, \to)$ be a transition system. Define T° to be the transition system $(Q^\circ, A^\circ, \to^\circ)$, where

(1) $Q^\circ = Q$;

(2) $A^\circ = \{aq' \mid \exists q \in Q: q \to^a q'\}$;

(3) $\to^\circ = \{(q, aq', q') \mid q \to^a q'\}$.

It is immediate that T° is unambiguous and that

$$a_1 q_1 \ldots a_n q_n \in L(T^\circ, q) \Leftrightarrow q \to^{a_1} q_1 \ldots q_{n-1} \to^{a_n} q_n \text{ in } T$$

Thus, $L(T^\circ, q)$ contains the additional information about the route through the transition system taken by an execution sequence. This allows us to generalise our earlier construction.

DEFINITION 3.1.18. Let $T = (Q, A, \to)$ be a transition system and let $q \in Q$. We define $B_{(T, q)} = (O_{(T, q)}, P_{(T, q)}, E_{(T, q)}, \lambda_{(T, q)})$, where

(1) $O_{(T, q)} = L(T^\circ, q) - \{\Omega\}$;

(2) $P_{(T, q)} = \{p_x \mid x \in L(T^\circ, q)\}$, where for all $x \in L(T^\circ, q)$, $p_x = \{u \in O_{(T, q)} \mid u \le x\}$;

(3) $E_{(T,q)} = A$;

(4) $\lambda_{(T,q)}(u) = a$ if $u = u'.a.q'$ for some $q' \in Q$.

LEMMA 3.1.19. $B_{(T,q)}$ is a discrete, sequential behavioural presentation.

□

If T happens to be unambiguous, then it may be shown that $B_{(T,q)} \equiv B_{L(T,q)}$. Thus, out new construction coincides - up to isomorphism - with our construction in the unambiguous case, as it certainly should.

DEFINITION 3.1.20. Let T be a transition system and let $q \in Q$. Let B be a discrete, sequential behavioural presentation, then (T, q) will be said to *accept* $B \iff B \equiv B_{(T,q)}$.

We should expect that any sequential behavioural presentation should be accepted by at least one pair (T, q), namely the pair $(T_B, \varnothing)$.

PROPOSITION 3.1.21. Let B be a discrete, sequential behavioural presentation, then $(T_B, \varnothing)$ accepts B.

□

To establish the generalisation of theorem 3.1.16, we need to know the circumstances under which two transition systems with initial state accept the same behavioural presentations. Alas, it turns out that strong congruence is not strong enough to distinguish between non-isomorphic behavioural presentations in some cases.

At this point, it seems that there is a choice of positions.

- Since the definition of isomorphism is clearly the 'natural' one, the notion of bisimulation is too weak to capture structural identity of system behaviour and we therefore need an alternative and stronger version.
- The notion of isomorphism may be 'natural' from a purely formal point of view, but it is too strong. For example, it would distinguish between the behavioural presentations corresponding to the transition systems T_1 and T_2 below.

$$\bullet \xrightarrow{a} \bullet \qquad\qquad \bullet \xleftarrow{a} \bullet \xrightarrow{a} \bullet$$

$$T_1 \qquad\qquad\qquad T_2$$

whereas both of them represent a system which may only perform one action, a.

We feel reluctant to arbitrate between these two positions and will therefore present the theory relating to both of them. The reader may then use whichever of the two he/she prefers.

Let us deal with a relation between transition systems corresponding to strong congruence between behavioural presentations.

DEFINITION 3.1.22. (Bisimulation of Behavioural Presentations) Suppose B_1 and B_2 are sequential behavioural presentations. A *strong bisimulation* between B_1 and B_2 is a relation $R \subseteq P_1 \times P_2$ such that $\varnothing \, R \, \varnothing$ and

$$\forall (p_1, p_2) \in R \; \forall p'_1 \in P_1 \forall e \in E_1 : p_1 \rightarrow^e p'_1 \Rightarrow \exists p'_2 \in P_2 : p_2 \rightarrow^e p'_2 \; \& \; p'_1 \, R \, p'_2$$

$$\forall (p_1, p_2) \in R \; \forall p'_2 \in P_2 \forall e \in E_2 : p_2 \rightarrow^e p'_2 \Rightarrow \exists p'_1 \in P_1 : p_1 \rightarrow^e p'_1 \; \& \; p'_1 \, R \, p'_2$$

We shall say that B_1 and B_2 are *strongly congruent* and write $B_1 \sim B_2 \iff$ there exists a strong bisimulation R between B_1 and B_2.

Clearly, $\sim$ is reflexive, symmetric and transitive. As expected:

LEMMA 3.1.23. Suppose B, B' are discrete sequential behavioural presentations then $(T_B, \varnothing) \sim (T_{B'}, \varnothing) \iff B \sim B'$.

□

THEOREM 3.1.24

Let $SB_\sim$ denote the class whose elements are all the ~ equivalence classes of sequential, discrete behavioural presentations. Write $[\![B]\!]_\sim$ for the ~ class of B.

Let $TS_\sim$ denote the class whose elements are all the ~ equivalence classes of transition systems with initial states. Write $[\![(T, q)]\!]_\sim$ for the ~ class of (T, q).

Then there is a bijective mapping $Beh_T: TS_\sim \rightarrow SB_\sim$ such that $Beh_T([\![(T, q)]\!]_\sim) = [\![B]\!]_\sim \Leftrightarrow (T, q)$ accepts B.

□

Now let us turn to the matter of a relation between transition systems corresponding to isomorphism between behavioural presentations.

First, for $q \in Q$, define $[q\rangle = \{aq' \in A.Q \mid q \rightarrow^a q'\}$.

DEFINITION 3.1.25. Let T_1, T_2 be transition systems and let $q_1 \in Q_1$ and $q_2 \in Q_2$. A *strict equivalence* on (T_1, q_1) and (T_2, q_2) is a relation $R \subseteq Q_1 \times Q_2$ such that

(1) $q_1 R q_2$;

(2) If $q'_1 R q'_2$, then there exists a bijection $f_{q'_1}: [q'_1\rangle \rightarrow [q'_2\rangle$ such that $f_{q'_1}(aq''_1) = bq''_2 \Rightarrow a = b \;\&\; q''_1 R q''_2$

We will say that (T_1, q_1) and (T_2, q_2) are *strictly equivalent* $\Leftrightarrow$ there is a strict equivalence on (T_1, q_1) and (T_2, q_2). Write $(T_1, q_1) \equiv (T_2, q_2)$. It is elementary that $\equiv$ is an equivalence relation on the class of all transition systems with initial state.

Also observe that if T_1, T_2 are unambiguous, then $(T_1, q_1) \equiv (T_2, q_2) \Leftrightarrow (T_1, q_1) \sim (T_2, q_2)$.

LEMMA 3.1.26. Suppose B, B' are discrete sequential behavioural presentations then $(T_B, \varnothing) \equiv (T_{B'}, \varnothing) \Leftrightarrow B \equiv B'$.

□

Finally, we have a result analogous to theorem 3.1.24 for strong equivalence.

THEOREM 3.1.27 Let SB denote the class whose elements are all the $\equiv$ equivalence classes of sequential, discrete behavioural presentations. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let TS denote the class whose elements are all the $\equiv$ equivalence classes of transition systems with initial states. Write $[\![(T, q)]\!]$ for the $\equiv$ class of (T, q).

Then there is a bijective mapping Beh_T: TS $\rightarrow$ SB such that $Beh_T([\![(T, q)]\!]) = [\![B]\!] \Leftrightarrow (T, q)$ accepts B.

3.2. Asynchronous Transition Systems

We would like to extend the material of section 3.1 to deal with non-sequential behavioural presentations. There are basically two additional phenomena to account for, namely concurrency and simultaneity. We shall tackle the former in this section and the entire[6] class of discrete behavioural presentations in section 3.3. In fact, we shall only consider linguistic behavioural presentations in this section.

Suppose B is ι-linguistic. Definition 3.1.3 allows us to construct a transition system from it, T_B, but all that transition systems accept are sequential behavioural presentations. What extra structure do we need? Obviously ι has to come into it somewhere. Where precisely it comes in is shown by the next lemma.

[6] For technical reasons, we restrict ourselves to behavioural presentations which are not only ≈ finite but in which all points are finite.

LEMMA 3.2.1. Suppose B is a discrete asynchronous behavioural presentation, then

(1) T_B is unambiguous;

(2) If $p_1, p_2, p_3 \in P$ and $e_1, e_2 \in E$ such that $p_1 \to^{e_1} p_2 \to^{e_2} p_3$ and $e_1 \, \iota \, e_2$, then there exists $p'_2 \in P$ such that $p_1 \to^{e_2} p'_2 \to^{e_1} p_3$.

□

This prompts the following definition.

DEFINITION 3.2.2. (ι Asynchronous Transition Systems) An ι asynchronous transition system is quadruple $C = (Q, A, \to, \iota)$ where

(1) $T(C) = (Q, A, \to)$ is an unambiguous transition system;

(2) ι is an independence relation;

(3) For all $q, q_1, q_2 \in Q$ and $a, b \in A$, if $a \, \iota \, b$, $q \to^a q_1$ and $q_1 \to^b q_2$, then there exists $q'_1 \in Q$ such that $q \to^b q'_1$ and $q'_1 \to^a q_2$.

From lemma 3.2.1, we deduce

PROPOSITION 3.2.3. Let B be ι linguistic and define $C_B = (Q, A, \to, \iota)$, where $T(C_B) = (Q, A, \to) = T_B$. Then C_B is an ι-asynchronous transition system.

□

The main thing about asynchronous transition systems is that they accept trace languages. This is the point of the 'lozenge rule'.

LEMMA 3.2.4. Suppose C is ι-asynchronous and let $x, y \in A^*$ such that $x \equiv_\iota y$, then for all $q, q' \in Q$, $q \to^x q' \iff q \to^y q'$, (where we are considering execution sequences in $T(C)$).

□

Thus, we may unambiguously define $q \to^{<x>_\iota} q' \iff q \to^x q'$ and obtain a trace language for C from each of its states.

$$L(C, q) = \{<x>_\iota \in A_\iota^* \mid \exists q' \in Q : q \to^{<x>_\iota} q'\}$$

REMARK 3.2.5. $L(C, q)$ is a left-closed trace language.

□

Since $L(C, q)$ is left closed, we may construct a behavioural presentation $B_{(C,q)}$ from it, following the recipe of definition 2.2.1. Now, the obvious question is, if $C = C_B$, what is the relationship between B and $B_{(C,q)}$?

LEMMA 3.2.6. Let B be an unlabelled, discrete, asynchronous behavioural presentation and let $p \in P$, then $x \in \rho(p) \iff \emptyset \to^x p$ in the transition system $T(C(H_B))$.

□

It follows from lemma 3.2.4 and lemma 3.2.6 that $\rho(P) = L(C_B, \emptyset)$. But now we may appeal to proposition 2.2.11 to deduce that:

PROPOSITION 3.2.7. Suppose B is an ι linguistic behavioural presentation then $B \equiv B_{L(C_B, \emptyset)}$.

□

DEFINITION 3.2.8. Let C be an ι-asynchronous transition system and $q \in Q$, then C accepts the behavioural presentation $B \iff B \equiv B_{L(C,q)}$.

Proposition 3.2.7 assures us that $(C_B, \emptyset)$ accepts B - so every ι-linguistic behavioural presentation is accepted by *some* asynchronous transition system. Conversely, $B_{(C,q)}$ wiill be ι-linguistic. Thus we have a relation between the two classes of object.

As in section 3.1, we would like to know the relationship between asynchronous transition systems which accept the same isomorphism class of behavioural presentations. Since the underlying transition systems don't suffer from ambiguity, we don't need two notions of equivalence. First, define

$a \; \iota/q \; b \iff a \, \iota \, b \; \& \; \exists q', q'' \in R(T(C), q): q' \rightarrow^a q'' \rightarrow^b$

DEFINITION 3.2.9. Let $C_1 = (Q_1, E_1, \rightarrow_1, \iota_1)$ and $C_2 = (Q_2, E_2, \rightarrow_2, \iota_2)$ be asynchronous transition systems and suppose that $q_1 \in Q_1$ and $q_2 \in Q_2$. A *strict equivalence* on (C_1, q_1) and (C_2, q_2) is a relation $R \subseteq Q_1 \times Q_2$ such that

(1) R is a strict equivalence between $(T(C_1), q_1)$ and $(T(C_2), q_2)$;

(2) $\iota_1/q_1 = \iota_2/q_2$;

Write $(C_1, q_1) \equiv (C_2, q_2)$ to indicate that there exists a strict equivalence on (C_1, q_1) and (C_2, q_2).

It is clear that $\equiv$ is an equivalence relation on the class of all asynchronous transition systems with initial state.

The next lemma shows that our equivalence has the right consequences.

LEMMA 3.2.10. Let $C_1 = (Q_1, E_1, \rightarrow_1, \iota_1)$ and $C_2 = (Q_2, E_2, \rightarrow_2, \iota_2)$ be asynchronous transition systems and suppose that $q_1 \in Q_1$ and $q_2 \in Q_2$, then the following are equivalent.

(1) $(C_1, q_1) \equiv (C_2, q_2)$;

(2) $L(C_1, q_1) = L(C_2, q_2)$;

(3) $B_{(C_1, q_1)} = B_{(C_2, q_2)}$.

□

We may now state the asynchronous analogue of theorems 3.1.16 and 3.1.24.

THEOREM 3.2.11. Let **LBP** denote the class whose elements are all the $\equiv$ equivalence classes of ι-linguistic, discrete behavioural presentations. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let ATS denote the class whose elements are all the $\equiv$ equivalence classes of ι-asynchronous transition systems with initial states. Write $[\![(C, q)]\!]$ for the $\equiv$ class of (C, q).

Then there is a bijective mapping

Beh_A: ATS $\rightarrow$ **LBP**

such that $Beh_A([\![(C, q)]\!]) = [\![B]\!] \iff (C, q)$ accepts B.

□

3.3. Hybrid Transition Systems

Finally, let us consider discrete behavioural presentations in general. How may we associate these with automata? We certainly have a transition structure, by lemma 1.2.6, but steps are generally of the form $p \vdash^X p'$, where X is a set.

If X is *finite*, then we may generalise the construction of definition 3.1.3 by defining $p \rightarrow^s p'$, where s is something that represents the events of which X is the set of occurrences together with their multiplicities. Following [11], we take s to be a particular member of the free commutative semigroup on E, which we shall denote by $S(E)$.

For our purposes it is convenient to regard $S(E)$ as the set of all functions $s: E \rightarrow N$ (the natural numbers)[7] such that

$|\{e \in E \mid s(e) > 0\}| < \infty$

with composition defined by

$(s_1 \bullet s_2)(e) = s_1(e) + s_2(e)$

The elements of $S(E)$ may be written as expressions of the form $e_1^{n_1} \ldots e_m^{n_m}$ which represents the function which sends e_i to n_i, $i = 1, \ldots, m$, and everything else to 0.

[7] $S(E)$ may also be regarded as the set of finite multisets (bags) over E.

The set $X \subseteq O$ may now be mapped to $\mu(X)$, defined by

$\mu(X)(e) = |\{o \in X \mid \lambda(o) = e\}|$

which counts the number of occurrences of each event associated with X.

Thus, if B is *synchronous* and discrete, and each of its $\approx$ class is finite - which is the same thing as saying that all its points are finite - then we may associate it with a transition system $(P, S(E), \rightarrow)$, where $\rightarrow$ is defined by:

$p \rightarrow^s p' \Longleftrightarrow \exists X \subseteq O: p \vdash^X p' \;\&\; \mu(X) = s$

It may be shown that this class of automata do accept precisely the discrete synchronous behavioural presentations with finite points. We are interested in general discrete behavioural presentations, however.

Since there will be simultaneity in the most general type of discrete behavioural presentation, we shall need something like the μ function to record the events associated with a step. This means, unfortunately, that we shall have to constrain ourselves to behavioural presentations with finite points.

From now on, by **discrete** *we mean a left closed behavioural presentation such that* $|p| < \infty$ *for each* $p \in P$.

But how do we introduce concurrency? It holds between occurrences rather than events, so we should be looking at them. Recall from proposition 1.1.8 that $\approx$ is a congruence relation with respect to $\rightarrow$ and $\#$. From this, it is easy to show that if $X, Y \in Q/\approx$, then

$(\exists o \in X \; \exists o' \in Y: o \; co \; o') \Longleftrightarrow (\forall o \in X \; \forall o' \in Y: o \; co \; o')$

This means that we may unambiguously define an independence relation on $O/\approx$, which we shall call co, by

$X \; co \; Y \Longleftrightarrow \exists o \in X \; \exists o' \in Y: o \; co \; o'$

We now have all our ingredients. Define $H_B = (P, O/\approx, \vdash, co, E, \mu)$. H_B is an example of an hybrid transition system, as we now define.

DEFINITION 3.3.1. (Hybrid Transition Systems) An hybrid transition system is sextuple $H = (Q, A, \rightarrow, \iota, E, \mu)$ where

(1) $C(H) = (Q, A, \rightarrow, \iota)$ is an ι asynchronous transition system;

(2) $\mu: A \rightarrow S(E)$.

PROPOSITION 3.3.2. Let B be a discrete behavioural presentation, then H_B is an hybrid transition system.

□

How do hybrid transition systems accept behavioural presentations? Given H and $q \in Q$, how can we construct a behavioural presentation representing the behaviour of H from q?

First, note that H determines an asynchronous transition system, namely $C(H) = (Q, A, \rightarrow, \iota)$ and so we may construct a behavioural presentation $B_{(C(H), q)}$. Now $B_{(C(H), q)}$ will be of the form (O, P, A, λ). But each $a \in A$ is a transition in H and detemines an element of $S(E)$, namely $\mu(a)$. Thus an occurrence $o \in O$ determines an element of $S(E)$, namely $\mu(\lambda(o))$. Now, recall that $S(E)$ was hauled in to represent $\approx$ classes. We may obtain a behavioural presentation with a simultaneity class, appropriately labelled, for each $o \in O_{(C(H), q)}$, by replacing each o with a set $\varepsilon(o)$ such that $\mu(\varepsilon(o)) = \mu(\lambda(o))$. We have the following construction.

DEFINITION 3.3.3. (ε Operator)

Let B be a behavioural presentation of the form $(O, P, S(E), \mu)$. For $o \in O$ and $X \subseteq O$, we define

(1) $\varepsilon(o) = \{(o, e, i) \mid 1 \leq i \leq \exp_e(\mu(o))\}$;

(2).2 $\varepsilon(X) = \bigcup_{o \in X} \varepsilon(o)$;

and we define $\varepsilon(B)$ to be the quadruple $(\hat{O}, \hat{P}, \hat{E}, \hat{\lambda})$, where $\hat{O} = \varepsilon(O)$, $\hat{P} = \{\varepsilon(p) \mid p \in P\}$, $\hat{E} = E$ and $\hat{\lambda}(o, e, i) = e$.

PROPOSITION 3.3.4. Suppose H is an hybrid transition system and $q \in Q$, then $B_{(H,q)}$, defined $B_{(H,q)} = \varepsilon(B_{(C(H),q)})$ is a discrete behavioural presentation.

□

DEFINITION 3.3.5. We shall say that (H, q) *accepts* $B \Longleftrightarrow B \equiv B_{(H,q)}$.

Now, if *any* initialised hybrid transition system accepts B, then $(H_B, \varnothing)$ ought to (otherwise, we have been rather misleading with our notation).

LEMMA 3.3.6. Suppose B is discrete, then H_B is hybrid and $(H_B, \varnothing)$ accepts B.

□

Thus, every initialised hybrid transition system accepts some discrete behavioural presentation and every discrete behavioural presentation is accepted by some initialised hybrid transition system. As in sections 3.1 and 3.2, we would like to discover the relationship between initialised hybrid transition systems which accept the same $\approx$ class of discrete behavioural presentation.

DEFINITION 3.3.7. Let H_1, H_2 be hybrid transition systems and let $q_1 \in Q_1$ and $q_2 \in Q_2$. A *strict equivalence* on (H_1, q_1) and (H_2, q_2) is a relation $R \subseteq Q_1 \times Q_2$ such that

(1) $q_1 \, R \, q_2$;

(2) If $q'_1 \, R \, q'_2$, then there exists a bijection $f_{q'_1}: [q'_1\rangle \to [q'_2\rangle$ such that

(2a) $f_{q'_1}(aq''_1) = bq''_2 \Rightarrow \mu_1(a) = \mu_2(b) \;\&\; q''_1 \, R \, q''_2$

(2b) $f_{q'_1}(aq''_1) = bq''_2 \;\&\; f_{q''_1}(a'q'''_1) = b'q'''_2 \Rightarrow (a \,\iota_1\, a' \Longleftrightarrow b \,\iota_2\, b')$.

We will say that (H_1, q_1) and (H_2, q_2) are *strictly equivalent* $\Longleftrightarrow$ there is a strict equivalence on (H_1, q_1) and (H_2, q_2). Write $(H_1, q_1) \equiv (H_2, q_2)$.

It is clear that $\equiv$ is an equivalence relation on the class of all hybrid transition systems with initial state. The next result shows that we are on the right track.

LEMMA 3.3.8. $(H_1, q_1) \equiv (H_2, q_2) \Longleftrightarrow B_{(H_1, q_1)} \equiv B_{(H_2, q_2)}$.

□

From this, we may deduce the last of our classification results.

THEOREM 3.3.9. Let **DBP** denote the class whose elements are all the $\equiv$ equivalence classes of discrete behavioural presentations. Write $[\![B]\!]$ for the $\equiv$ class of B.

Let **HTS** denote the class whose elements are all the $\equiv$ equivalence classes of hybrid transition systems with initial states. Write $[\![(C, q)]\!]$ for the $\equiv$ class of (C, q).

Then there is a bijective mapping

Beh_{H}: **HTS** $\to$ **DBP**

such that $Beh_{\mathrm{H}}([\![(H, q)]\!]) = [\![B]\!] \Longleftrightarrow (H, q)$ accepts B.

□

References

1. G. Winskel, "Event Structure Semantics for CCS and Related Languages.," Tech. Rep. DAIMIPB-159, Computer Science Department, University of Aarhus, April, 1983.
2. M. W. Shields, "Behavioural Presentations," in *Procedings REX Workshop on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency.*, Springer Verlag, 1988.
3. A. Einstein, *Relativity: The Special and General Theory*, Methuen.
4. M. Neilsen, G. Plotkin, and G. Winskel, "Petri Nets, Event Structures and Domains," in *Proceedings, Symposium on the Semantics of Concurrent Computation*, Lecture Notes in

Computer Science, vol. 70, pp. 266-283, Springer Verlag, 1979.

5. A. Mazurkiewicz, "Concurrent Program Schema and their Interpretations," in *Procedings Aarhus Workshop on Verification of Parallel Programs*, 1977.
6. M. W. Shields, "Adequate Path Expressions," in *Proceedings, Symposium on the Semantics of Concurrent Computation*, Lecture Notes in Computer Science, vol. 70, pp. 249-265, Springer Verlag, 1979.
7. R. Janicki, "A Characterisation of Concurrency-like Relations," in *Proceedings, Symposium on the Semantics of Concurrent Computation*, Lecture Notes in Computer Science, vol. 70, pp. 102-122, Springer Verlag, 1979.
8. L. Lamport, "Proving the Correctness of Multiprocessor Programs," *IEEE Trans. Software Engineering*, vol. SE-3,2, pp. 125-143, 1977.
9. L. Boasson and M. Nivat, "Adherences of Languages," *Journal or Computer and Systems Software*, vol. 20, pp. 285-308, 1980.
10. R. M. Keller, "Formal Verification of Parallel Programs," *CACM*, vol. 19(7), 1975.
11. A. R. J. G. Milner, "Calculi for Synchrony and Asynchrony," Tech. Rep. CSR-104-82, Computer Science Department, University of Edinburgh, 1982.