Teams Can See Pomsets (Preliminary Version)

Gordon Plotkin and Vaughan Pratt*
University of Edinburgh and Stanford University
gdp@dcs.edinburgh.ac.uk, pratt@cs.stanford.edu

July 24, 1996

Abstract

The sequentiality postulate assumes that events occur in a definite order. We explore some of the boundary of applicability of this postulate for the case of sequential observers, varying number of observers, duration of events, and variability of events. When there is one observer or events are atomic, the sequentiality postulate holds, making linear orders a fully abstract model of concurrent behavior. With more than one observer and with structured events it fails. We show that unlimited observers and variable events make pomsets a fully abstract model. Putting duration in place of variability yields an intermediate situation in which the sequentiality postulate does not hold but pomsets are not a fully abstract model.

1 Overview

It is widely believed that trace or interleaving semantics, which assigns a definite order of occurrence to every pair of events, is sufficient for all practical purposes. In support of this belief, Jonsson [Jon89] and Russell [Rus89] show that trace semantics is fully abstract for parallel computation, at least of the kind represented by Kahn networks.

However these full abstractness results suffer from an overly constrained notion of observer. In this paper we consider a wider range of observational behaviors or *testing scenarios*, and give a detailed picture of just where full abstractness for trace semantics becomes unsound for the eight scenarios obtained by varying three basic parameters of computation, namely duration D, variability V, and multiplicity M of observers ("teams").

Duration expresses the notion of an ongoing action, one that can be analyzed as a sequence of subactions. Duration is naturally modeled as a string. An

*This work was supported by ONR under grant number N00014-92-J-1974.

action a may be analyzed as say the string a_1a_2 indicating that a decomposes into two consecutively performed actions, a_1 then a_2.

Variability expresses choice, naturally modeled as a set of alternatives. An action a may be analyzed as say the set $\{a_1, a_2\}$ indicating that for a to occur means that exactly one of a_1 or a_2 occurs.

Multiplicity expresses the notion of two or more observers both observing the same run of a computation, but from different vantage points. We shall assume that when two observers see the same events from different viewpoints, they agree on all choices that have been made, including those associated with variability, but may disagree on the relative order of events. We understand choice as absolute, in that it is unambiguous which of two alternatives has been chosen. However we view time as relative in that two events not occurring in each other's light cone do not have a well-defined order of occurrence. This asymmetry of choice and time, while certainly questionable, is consistent with physics as standardly taught.

Our results in the case of computational behaviors consisting of single pomsets (labeled partial orders) is summarized by the following cube.

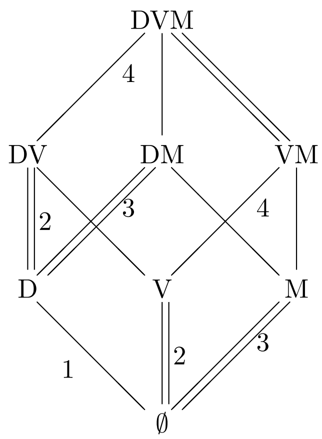

Figure 1. Eight testing scenarios

Edges are labeled with the number of the relevant proposition, while the double lines indicate equivalence, with respect to distinguishing power, of two kinds of observational behavior, with the remaining lines then indicating strict inequalities. Thus Proposition 1 shows that Duration on its own makes a difference while Propositions 2 and 3 show that neither Variability nor Multiplicity make any difference, neither on their own nor as an addition to Duration. Proposition 4 shows that in the presence of Variability, Multiplicity does make a difference. Moreover an unlimited supply of observers leads to full abstractness for pomsets even at VM, whence DVM cannot be any bigger and so must equal VM. This then has the side effect of removing Duration as a contributing factor.

The identifications reduce the classes to three, namely $\emptyset = V = M$, $D = DV = DM$, and $VM = DVM$, while Propositions 1 and 4 show that these three classes are distinct.

As a refinement of these all-or-nothing results, Proposition 5 extends Proposition 4 to a hierarchy theorem: $n+1$ observers can observe distinctions invisible to n observers.

We also consider processes as *sets* of pomsets, and show that the identifications of VM with DVM, and of $\emptyset$ with V, continue to hold. (Rob van Glabbeek has pointed out to us that this cannot be improved, via examples separating D from DV and from DM, and $\emptyset$ from M.)

2 Background

Linearly ordered multisets (labelled chains up to isomorphism) are strings. Pomsets as partially ordered multisets therefore constitute a generalization of strings to partial orders. This model as an extension of formal language theory is due to Grabowski [Gra81] who called it a partial word, the characterization as a partially ordered multiset being due to the second author [Pra82]. Pomsets with a conflict relation are called event structures, introduced by Nielsen, Plotkin, and Winskel [NPW81]. Prior related notions are Mazurkiewicz's partial monoids [Maz77, Maz84] and Greif's treatment of actors [Gre75]. A list of more recent papers on the topic [MS80, Gis88, Pra86, AH87, Win88] would be bound to be incomplete.

We shall identify observation with linearization. That is, at least in the case of atomic events, an observer of a pomset sees its events in some linear order consistent with the order of the pomset.

To a zeroth order approximation, two pomsets should be observationally equivalent when they have the same set of linearizations.

The familiar theorem that (the graph of) a poset is the intersection of the set of (graphs of) its linearizations is due to Szpilrajn [Szp30]. In our framework posets are pomsets with no repeated elements, i.e. the function assigning labels to poset elements is injective. Thus in our application Szpilrajn's theorem states that distinct posets are not observationally equivalent.

At the other extreme from posets are pomsets over a one-letter alphabet, say the alphabet $\{a\}$. In our framework these amount to posets up to isomorphism. (So pomsets span a spectrum from posets-up-to-isomorphism to posets.) There are just two two-element pomsets over $\{a\}$, which we write as aa (linearly ordered) and $a|a$ (discretely ordered). These have the same set of linearizations and hence are observationally equivalent. So whereas Szpilrajn's theorem applies to posets this example shows that it does not apply to posets up to isomorphism.

The meaning of $a|a$ is that we have two copies of an activity a that are running in parallel. If a is an instantaneous event, as we have been assuming up to now, and the possibility of exact simultaneity is neglected, then there would seem to be no basis for distinguishing between aa and $a|a$ in either theory or practice.

If however a has duration we have the possibility of overlap for the case $a|a$, but not for aa. We may represent duration by taking a to be a pomset of size two or more, e.g. the string 01. Then the only linearization of aa is 0101, whereas $a|a$ has for its linearizations both 0101 and 0011. Hence in the presence of events with duration it becomes possible to observe a difference between aa and $a|a$. A similar difference is observable if we take a to be $0|1$. In this case the linearizations of aa are 0101, 0110, 1001, and 1010, while those of $a|a$ are those four together with 0011 and 1100.

Gischer [Gis88] shows that any two pomsets that are observationally equivalent for strings of length two are observationally equivalent for strings of any length, whence there is no duration hierarchy for strings. Gischer conjectured, and Tschantz has shown [Tsc94], that duration suffices to distinguish any two series-parallel (N-free) pomsets. (A series-parallel pomset is a pomset constructible using only the operations of concatenation ab and concurrence $a|b$.) Hence series-parallel pomsets are extensional in the presence of duration. (Another striking corollary of this result is that the equational theory of concatenation and interleaving of languages is completely axiomatized by the equations for commutativity of interleaving and associativity of both.)

Gischer gives as an example of pomsets indistinguishable even with duration the two pomsets $N(a, a, b, b)$ and $ab|ab$, where $N(1, 2, 3, 4)$ is the 4-vertex pomset ordered so that $1 < 3$, $2 < 4$, and $1 < 4$, these constraints constituting respectively the two verticals and the diagonal of the letter N, so that $N(a, a, b, b)$ is $ab|ab$ plus the diagonal. If they could be distinguished it would have to be by a string of $ab|ab$ not allowed by $N(a, a, b, b)$, possible only by violating the diagonal $1 < 4$ of the N. Hence 1 and 4 overlap; where they do, 2 cannot have started but 3 must have finished, so the other diagonal $2 < 3$ is satisfied. But that diagonal belongs to an isomorphic copy of $N(a, a, b, b)$, whence that string must be allowed after all.

We may further take a to be not just a single string but a set of strings, that is, a language. This provides a notion of variety for a: we have a variety of choices of behaviors of a. When all strings of a are of unit length we have variety without duration. Variety provides those little unpredictable hints that can allow observers to reach consensus as to the identities of entities without them being a part of the observation language. In some observations the observers may be unlucky and not get enough such hints; it only matters that there exist observations that do provide sufficient hints.

Gischer's argument above remains valid in the presence of variety, giving a pair of pomsets which variety does not help distinguish.

Two minor results concerning refinements of observational equivalence in this setting are as follows.

(i) For a single observer, duration helps but variety does not.

(ii) For multiple observers to make a difference, variety without duration helps but duration without variety does not.

Our main result is:

(iii) With enough variety and observers any two finite pomsets can be distinguished, even without duration.

Results (i) and (ii) assign very different roles to duration and variety. Duration is a loner that can help, though not always, as evidenced by Gischer's example above of $N(a,a,b,b) = ab|ab$. Variety on the other hand is useless by itself but in collaboration with multiple observers is able not only to outperform duration but, as (iii) shows, to make pomsets fully visible, i.e. extensional. The proof of (iii) is via a straightforward reduction to the poset case, allowing us to apply Szpilrajn's theorem.

A refinement of (iii) is that with enough variety, the number of observers needed to distinguish two pomsets is at most the larger of the dimensions of their underlying posets.[1] This shows that the hierarchy of observational equivalences with n observers is strict: $n+1$ observers can resolve more detail than n. Although our proof of this result is not long, neither is it at all obvious!

3 Definitions

The following notions are essentially as in [Gis84]. We start out by defining labelled partial orders and their maps.

Definition 1. A *labelled partial order* or *lpo* over a set Σ is a structure $(V, \leq, \sigma, \Sigma)$ where $\leq$ partially orders V and $\sigma : V \to \Sigma$ assigns to each element of V an element of Σ. When necessary we write the components of lpo p as $(V_p, \leq_p, \sigma_p, \Sigma_p)$.

We think of Σ as an alphabet of *actions* and V as instances of that alphabet, or *events* forming a word, with the order of occurrences of letters in the word given by $\leq$. The usual formal language theoretic notion of a word obtains for $\leq$ linear. An atomic lpo is one with $|V| = 1$.

Definition 2. A *map* of lpo's $(f,t) : (V, \leq, \sigma, \Sigma) \to (V', \leq', \sigma', \Sigma')$ consists of a monotone map $f : (V, \leq) \to (V', \leq')$ of posets together with an alphabet map (function) $t : \Sigma \to \Sigma'$ such that for all v in V, $\sigma'(f(v)) = t(\sigma(v))$.

Certain maps of lpo's are of special interest here. An *isomorphism* of lpo's is a map (f,t) for which f is an isomorphism of posets and t is the identity map on Σ (so isomorphic lpo's have a common alphabet). An *augmentation* of lpo's is a map (f,t) for which t is the identity function and f is the identity function on the elements of the poset (but not necessarily an isomorphism of posets, i.e. the order may increase); an augmentation yields an *augment* of its argument. We write $p \alpha q$ to indicate that q is an augment of p; this is the converse of Gischer's *subsumption* relation $q \succ p$ [Gis84].

Definition 3. A *pomset* is the isomorphism class of an lpo.

More intuitively a pomset is an lpo in which we pay no attention to the choice of the set V, other than its cardinality, but retain all other details. Thus if we replace $V = \{0,1,2\}$ by $V = \{5,6,7\}$ without otherwise disturbing either $\leq$ or

[1] The dimension of a poset is the least number of linearizations of that poset whose intersection is that poset. The notion is due to Dushnik and Miller [DM41], see Kelly and Trotter [KT82] for a survey.

σ the pomset does not change. With our definition of observation, isomorphic lpo's will be seen to be observationally equivalent, whence the most we can hope to resolve even with multiple observers is pomsets.

We shall understand a map between two pomsets to be a map between representative lpo's of the respective pomsets.

Definition 4. A *process* P is a set of finite pomsets. A process is *augment closed* when for all $p\alpha q$, $p \in P$ implies $q \in P$. The *augment closure* $\alpha(P)$ of P is the least augment closed process containing P.

We wish to define observation in terms of the notions of *linearization* and *substitution*, which we now define.

Definition 5. A *linearization* of a pomset p is a linear augment of p. We write $\lambda(p)$ for the set of all linearizations of p. This extends to $\lambda(P)$ for P a set of pomsets, namely as $\lambda(P) = \bigcup_{p \in P} \lambda(p)$.

Formal language theory has the notions of homomorphism and substitution [HU79]. These both generalize immediately from strings to pomsets. (This notion of homomorphism is quite different from that of map between two pomsets: the former goes between sets of pomsets, the latter between single pomsets.)

Definition 6. A *pomset homomorphism* is a function mapping pomsets on Σ to pomsets on Σ'. It is determined by a function f assigning a pomset on Σ' to each letter of Σ. It maps p to the pomset whose set of events is the disjoint sum of the events of the $f(\sigma(u))$'s over all $u \in V_p$, definable as $\{(u, v) | u \in V_p, v \in V_{f(\sigma(u))}\}$. Each (u, v) is labelled with $\sigma_{f(\sigma(u))}(v)$, i.e. just as v was labelled in $f(\sigma(u))$, and ordered so that $(u, v) \leq (u', v')$ just when $u <_p u'$ (i.e. $u \leq_p u'$ and $u \neq u'$) or ($u = u'$ and $v \leq_{f(u)} v'$), that is, lexicographic ordering.

Intuitively this is what is obtained by substituting a pomset for each label of p and flattening the resulting nested structure in the obvious way. For example the homomorphism taking a to bc takes aa to $bcbc$ and $a|a$ to $bc|bc$, while the homomorphism taking a to $b|c$ takes aa to $(b|c)(b|c)$ and $a|a$ to $b|b|c|c$.

This generalizes to *substitutions* of sets of pomsets exactly analogously to the generalization of homomorphisms of strings to substitutions of sets of strings [HU79], in which the result of substituting a set of strings for a letter is the set of all strings obtainable by choosing any string from each substitution instance of such a set. In lieu of a formal definition we offer the example of substituting the set $\{b, c\}$ for a in $a|a$, having two substitution instances of $\{b, c\}$ and so yielding the set of three pomsets $b|b$, $b|c$, $c|c$ ($c|b$ being isomorphic to $b|c$ as an lpo and hence equal as a pomset). Just as for formal languages, a homomorphism can be viewed as the special case of a substitution of singletons.

We may now regard pomsets as expressions, with the labels acting as variables. Evaluation is then just substitution: values for the variables determine the value of the expression. Thus the pomset aba is an expression with variables a and b, and if the value of a is cd and that of b is $\{e, f\}$ then the value of aba is $\{cdecd, cdfcd\}$. With this interpretation of substitution in mind we write $p(s)$ for the value of p under the substitution s. By $P(s)$ for a set P of pomsets we understand the union over the elements $p \in P$ of $p(s)$.

We might say that two pomsets are equivalent when their values are the same for all substitutions. But merely taking the value of each variable to be itself already suffices to distinguish distinct pomsets, so this equivalence is trivially the identity relation.

The notion of observation as linearization, reflecting the sequential life of an individual observer, leads to more interesting equivalences. We tentatively define an observation of a pomset to be a linearization of it. Thus the set of all observations of p is $\lambda(p)$, and the set of all observations of a set P of pomsets is $\lambda(P)$. Pomsets p and q are *equivalent* when $\lambda(p(s)) = \lambda(q(s))$ for all substitutions s.

We now extend this notion of observation to multiple observers. The idea is that n observers see n possibly different linearizations of the one observed pomset.

Definition 7. An *n-observation* of a pomset p is an n-tuple of linearizations of p. We write $\lambda_n(p)$ for the set consisting of all n-observations of p, a set of n-tuples of strings. For a process P we take $\lambda_n(P) = \bigcup_{p \in P} \lambda_n(p)$.

Definition 8. Pomsets p and q are *n-equivalent*, written $p \equiv_n q$, when $\lambda_n(p) = \lambda_n(q)$. Likewise for processes, $P \equiv_n Q$ when $\lambda_n(P) = \lambda_n(P)$.

Our tentative definitions of observation and equivalence are now subsumed as 1-observation and 1-equivalence.

Implicit in our definition of n-equivalence is a consensus between the observers as to which pomset of P to linearize, when constructing an n-observation in $\lambda_n(P)$. This reflects our intuition that the observers agreed on what happened but not when.

Finally we need the notion of dimension [KT82] in order to show the strictness of the hierarchy of n-equivalence in the presence of variety.

Definition 9. The *dimension* of a poset is the minimum number of its linearizations such that the intersection of those linearizations is that poset. We take the dimension of a pomset p to be the dimension of the underlying poset of a representative lpo of p.

4 Observation of Single Pomsets

In order to capture duration, variety, etc. we need a parametrized notion of n-equivalence, parametrized by the permitted substitutions. If substitutions are restricted so that the assignment to any variable must come from a class C of sets of pomsets, e.g. singletons, sets of one-element pomsets, languages (sets of linear pomsets), we say that two pomsets are n-equivalent for C when they have the same n-observations of their values for all substitutions where the assignments to the variables are drawn from C.

In the following we are interested in substitutions that have variety without duration, and duration without variety. We denote these respective classes of substitutions by **Var** and **Dur** respectively. A substitution from **Var** can replace each label by a set of labels. A substitution from **Dur** can replace each label by a pomset. The class of substitutions permitting neither duration nor variety, corresponding to mere renamings of labels, we call **Atm** for atomic substitutions.

None of our results make essential use of nonlinearity in the substructure of events. For example if **Dur** is taken instead to consist of those substitutions that replace labels by strings rather than pomsets, no modifications are required to either the following propositions or their proofs.

The first two propositions are simple, but give some insight into the respective roles played by duration and variety.

We first show that for a single observer, duration without variety helps but variety without duration does not.

Proposition 1. 1-equivalence for **Dur** is strictly finer than 1-equivalence for **Atm**.

Proof. It is finer because **Dur** includes **Atm**. The example of aa and $a|a$ shows strictness. ∎

Proposition 2. 1-equivalence for **Var** coincides with 1-equivalence for **Atm**.

Proof. This follows from $\lambda(p(s)) = (\lambda(p))(s)$. That is, we can substitute sets for variables in p and then linearize, or linearize p first (yielding a language) and then substitute, with the same result in either case. Hence $\lambda(p(s)) = (\lambda(p))(s) = (\lambda(q))(s) = \lambda(q(s))$. ∎

Proposition 3. For all $n \geq 1$, 1-equivalence for **Dur** coincides with n-equivalence for **Dur**.

Proof. In this case $p(s)$ is a singleton, substitutions being homomorphisms, for which $\lambda_n(p(s))$ is the set of all n-tuples of linearizations of the pomset $p(s)$. Hence $\lambda_n(p(s))$ can be computed from $\lambda(p(s))$. Thus if $\lambda(p(s)) = \lambda(q(s))$, we must have $\lambda_n(p(s)) = \lambda_n(q(s))$ as well. ∎

Corollary. For all $n \geq 1$, 1-equivalence for **Atm** coincides with n-equivalence for **Atm**.

We now come to the main results. The next two propositions show that for multiple observers to make a difference, variety without duration helps but duration without variety does not. The former, proposition 3, is the main result in that it shows that any two pomsets can be distinguished by n observers for sufficiently large n. It is noteworthy that duration plays no role in this result! Since our first explorations in this area focused on the role of duration in distinguishing pomsets we did not at first expect such a result. In retrospect it is not so surprising, nor particularly deep, being a straightforward reduction to Szpilrajn's theorem..

Proposition 4. For any pomset p there exists n such that p is not n-equivalent for **Var** to any other pomset.

Proof. We use variety to distinguish the otherwise indistinguishable events of a pomset. Let m be the size of p. We take n to be $m!$. Consider the substitution s mapping each letter a of Σ to the m-element set $\{(a,i)|0 \leq i < m\}$. This is enough variety for $p(s)$ to include at least one poset, call it q. Then $\lambda(q)$ has at most $m!$ members, whence some $m!$-tuple of $\lambda_{m!}(q)$ will contain all of them. This gives us a procedure for recovering p from $\lambda_{m!}(p(s))$. Discard $m!$-tuples of $\lambda_{m!}(q)$ not corresponding to posets (repeated letters). From the remainder select any

$m!$-tuple with a maximum number of different components, an $m!$-observation of some poset q. Use Szpilrajn's theorem to infer q from the $m!$-observation. Replace each label (a, i) by a in q, to yield p. This construction shows that the p so recovered will be independent of the choice of poset from $p(s)$. ∎

The argument for proposition 4 can be extended to show that, for any class including **Var**, n-equivalence for increasing n forms a strict hierarchy. Our particular witnesses to this hierarchy are independent of the class of substitutions.

Proposition 5. For every $n > 1$ there exist pomsets p and q such that for any class C of substitutions including **Var**, p and q are n-1-equivalent for C but not n-equivalent for C.

Proof. It suffices to consider pomsets over a one-letter alphabet, i.e. posets up to isomorphism. (Note that Szpilrajn's theorem separates even isomorphic posets, and cannot be applied directly here.) Given n we take for our counterexample a certain pair p, q of posets of dimension n. Using essentially the same argument as in Proposition 4 we show that as one-letter pomsets p and q cannot be n-equivalent for **Var**, and hence for any larger class. We then show that they are n-1-equivalent for any class.

We take p to be the *standard* poset S_n [KT82], having $2n$ elements $\{a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}\}$, ordered so that $a_i \leq b_j$ just when $i \neq j$. An equivalent description of S_n is as the lattice of atoms and coatoms of an n-atom Boolean algebra. S_n is known to have dimension n [KT82]. We take q to be S_n augmented with $a_0 \leq b_0$. (As pomsets, p and q are determined only up to isomorphism, so augmenting p with $a_i \leq b_i$ for any i yields the same pomset q.) Since q has $2n$ elements it is of dimension at most n [KT82]. Hence p and q are not n-equivalent for **Var**. The role of **Var** here is as for Proposition 4, namely allowing us to treat pomsets as posets.

For n-1-equivalence, suppose some linearization of an element of $p(s)$ violates $a_i \leq b_i$ for some i, necessary if we are to distinguish p and q. Then there is a point in that string where a_i has not yet finished (a_i could have duration in the general case) yet b_i has started. The constraints of p require that at that point all the other a_j's are done (for b_i to start) and none of the other b_j's have started (since a_i is not yet done). Hence for every $j \neq i$, $a_j \leq b_j$, that is, there can be at most one violation of $a_i \leq b_i$ for any i in any one linearization. But then any n-1-observation of $p(s)$ can collectively violate at most $n-1$ of the constraints of the form $a_i \leq b_i$. This always leaves one such constraint unviolated, which is consistent with observing q. Hence the n-1-observations of $p(s)$ must coincide with those of $q(s)$ for all s. ∎

5 Observation of Processes

A process is a set of pomsets, as per Definition 4. All our definitions of linearization, n-equivalence, etc. have been formulated to hold for processes in general, with single pomsets identified with singleton processes.

The following shows a basic limitation of all the testing scenarios considered in this paper when applied to processes.

Proposition 6. Observationally equivalent processes have equal augment closures.

Proof. Any pomset p of a process P must be visible to a team of size $\dim(P)$. If Q is observationally equivalent to P the same team must be able to observe p as an apparent behavior of Q. Hence Q must contain a behavior q of which p is an augment, whence $P \subseteq \alpha(Q)$. By symmetry of equivalence $Q \subseteq \alpha(P)$, whence $\alpha(P) = \alpha(Q)$.

Lemma 7. Let p be a pomset. Then there exists n such that for any family $\langle q_i \rangle_i$ of pomsets for which $\lambda_n(p) \subseteq \lambda_n(\bigcup_i q_i)$, there must exist q_j in the family such that p is an augment of q.

Proof. The only q_i's that can contribute to $\lambda_n(p)$ have the same number of vertices as p. Since each n-tuple in $\lambda_n(\bigcup_i q_i)$ arises from a choice of a particular q_i, and since $\lambda_n(p)$ includes a single n-tuple completely encoding p, it follows that some q_i must yield that n-tuple. But this is only possible for a q_i of which p is an augment. ∎

Proposition 8. For any two augment-closed processes P and Q there exists n such that P is not n-equivalent for **Var** to Q.

Proof. Assume without loss of generality that P contains a pomset p absent from Q. Then p is not an augment of any pomset of Q. Let n be the number associated to p by Lemma 7. Then $\lambda_n(p)$ cannot belong to $\lambda_n(Q)$, whence $\lambda_n(P)$ contains n-tuples not in $\lambda_n(Q)$. ∎

This generalizes Proposition 4 to full abstraction for processes. Hence VM for processes makes all possible distinctions between processes, whence DVM can only make the same distinctions. Thus for processes we retain the $VM = DVM$ edge of Figure 1.

Proposition 2 showed that variability alone makes no difference for single pomsets. But that proposition applies equally to pomsets and processes, whence variability also makes no difference for processes and we retain the $\emptyset = V$ edge of Figure 1.

References

[AH87] S. Anderson and P. Hudak. Pomset interpretations of parallel functional programs. In *Proc. 3rd International Conf. on Functional Programming Languages and Computer Architecture*, Portland, OR, September 1987.

[DM41] B. Dushnik and E.W. Miller. Partially ordered sets. *Amer. J. Math.*, 63:600–610, 1941.

[Gis84] J.L. Gischer. *Partial Orders and the Axiomatic Theory of Shuffle.* PhD thesis, Computer Science Dept., Stanford University, December 1984.

[Gis88] J.L. Gischer. The equational theory of pomsets. *Theoretical Computer Science*, 61:199–224, 1988.

[Gra81] J. Grabowski. On partial languages. *Fundamenta Informaticae*, IV.2:427–498, 1981.

[Gre75] I. Greif. *Semantics of Communicating Parallel Processes*. PhD thesis, Project MAC report TR-154, MIT, 1975.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[Jon89] B. Jonsson. A fully abstract trace model for dataflow networks. In *Proc. 16th Annual ACM Symposium on Principles of Programming Languages*, pages 155–165, January 1989.

[KT82] D. Kelly and W.T. Trotter. Dimension theory for ordered sets. In I. Rival, editor, *Ordered Sets*, pages 171–211. D. Reidel, 1982.

[Maz77] A. Mazurkiewicz. Concurrent program schemes and their interpretations. Technical Report DAIMI Report PB-78, Aarhus University, Aarhus, 1977.

[Maz84] A. Mazurkiewicz. Traces, histories, graphs: Instances of a process monoid. In *Proc. Conf. on Mathematical Foundations of Computer Science*, volume 176 of *Lecture Notes in Computer Science*. Springer-Verlag, 1984.

[MS80] U. Montanari and C. Simonelli. On distinguishing between concurrency and nondeterminism. In *Proc. Ecole de Printemps on Concurrency and Petri Nets*, Colleville, 1980.

[NPW81] M. Nielsen, G. Plotkin, and G. Winskel. Petri nets, event structures, and domains, part I. *Theoretical Computer Science*, 13:85–108, 1981.

[Pra82] V.R. Pratt. On the composition of processes. In *Proceedings of the Ninth Annual ACM Symposium on Principles of Programming Languages*, January 1982.

[Pra86] V.R. Pratt. Modeling concurrency with partial orders. *Int. J. of Parallel Programming*, 15(1):33–71, February 1986.

[Rus89] J. Russell. Full abstraction for nondeterministic dataflow networks. In *Proc. 30th IEEE Symposium on Foundations of Computer Science*, October 1989.

[Szp30] E. Szpilrajn. Sur l'extension de l'ordre partiel. *Fund. Math.*, 16:386–389, 1930.

[Tsc94] S.T. Tschantz. Languages under concatenation and shuffling. *Math. Structures in Comp. Sci.*, 4(4):505–511, December 1994.

[Win88] G. Winskel. An introduction to event structures. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, REX'88*, volume 354 of *Lecture Notes in Computer Science*, Noordwijkerhout, June 1988. Springer-Verlag.