Reconciling Event Structures and Higher Dimensional Automata

Vaughan Pratt*
Dept. of Computer Science
Stanford University
Stanford, CA 94305-2140
pratt@cs.stanford.edu

July 13, 1996

Abstract

Addressing the problem in the title, we diagnose the essential difficulty as the implicit assumption of two-valued logic in the event structure view of event occurrence as a true-or-false proposition. Taking the 3-valued logic of before-during-after as the common underlying logic of both event structures and higher dimensional automata (HDA's) unifies them into a single model having the same process algebra, linear logic, and schedule-automaton duality as had previously been achieved for event structures, but not previously for HDA's. The crucial tool is Chu spaces, whose application to concurrency has previously limited itself to Chu spaces over 2, and which here is taken over 3.

This model is simplified by the absence from the definition of Chu spaces of any a priori logical structure on the *set* 3, allowing a process on a set A of events to be defined as simply any subset of the set 3^A. The structure required for our model of HDA's is a specific 3-element quantale, which however is not built into the basic model but made an after-the-fact interpretation of it; remarkably, Chu transforms respect whatever structure that interpretation gives rise to, and no more.

1 The Problem

At question time after my presentation of higher dimensional automata (HDA) [Pra91], Boris Trakhtenbrot asked, what is the relationship between HDA's and event structures?[1] Despite having addressed the issue in section 7, focusing on persistence of conflict, I had no good answer to the question, a conflict that persisted for me until the solution presented here struck me late last year.

The general problem here is to reconcile the respective motivating intuitions and supporting formalisms underlying the event-based and geometry-based approaches to concurrency. Event structures replace the traditional state-based view of computation by the view of $a||b$ as a set $\{a, b\}$ representing two events unconstrained as to order. In contrast the geometry-based approach retains the state-based view and models $a||b$ as the evident four-state "square" automaton accepting $ab+ba$ but with its square interior filled in as a two-dimensional surface representing the overlapped execution of a and b.

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

[1] More precisely, he asked about the relationship with event spaces [Pra92], but that distinction is a negligible one compared with the difference with HDA's.

The question then becomes, can these be understood as simply the one notion of concurrent computation seen from two compatible perspectives, or is there some inconsistency preventing the reconcilation of these two models of concurrency?

There are two main reasons for pursuing Trakhtenbrot's question. First, HDA's and event structures are both attractive models of concurrency, raising the question of whether their respective intuitions about the nature of concurrency are compatible. Second, HDA's and event structures draw on complementary areas of mathematics: much of the HDA work since its introduction [vG96, GJ92, GC93, Gou93, Gun94, Gou95b, Gou95a, Gou96a, Gou96b, BJ96, SC96, Tak96] has drawn on methods in algebraic topology, in particular homotopy and homology, whereas event structures tend to depend more on methods from domain theory and logic, especially those involving duality. It is therefore both intrinsically interesting in principle and important in practice for concurrent programming language design to understand how the one notion of concurrency can shift its emphasis in this way between these substantially different areas of mathematics.

What are some essential differences between HDA's and event structures?

Two basic differences involve conflict and duality. HDA's can express not only persistent but also transient conflict, whereas event structures cater better to schedule-automaton duality.

Persistent conflict, notated $a\#b$ (in the binary case), means that at most one of a or b can happen. Transient conflict, as in mutual exclusion, means that a precludes b (again for the binary case) only temporarily, e.g. while a is executing. Whereas HDA's can express both kinds of conflict equally directly, event structures express only persistent conflict directly, and deal only clumsily with transient conflict. An unlabeled event structure with two events a and b can express their persistent conflict directly as $a\#b$, by design. To express a *mutex* b however, it is rendered as $ab + ba$, which instead of two unlabeled events requires four labeled events, two labeled a and two labeled b, with each of the two events of ab in persistent conflict with each of those of ba. HDA's do much better here, expressing the same concept simply by omitting the interior of the square.

On the other hand, event structures dualize easily to acyclic automata, or families of configurations as they are called in the event structure literature. Dualization is accomplished simply by taking the configurations to be those sets of events satisfying the constraints defining the event structure. Identifying sets of events with their characteristic functions to 2, dualization can be understood as "homming into" an object k (here the set 2), in general $\mathrm{Hom}(-, k)$ as the usual notion of representable contravariant functor. In contrast, HDA's as typically defined have no obvious dualization to higher dimensional schedules, or to any plausible notion of schedule.

Both transient conflict and schedule-automaton duality are too important to neglect. Transient conflicts of various kinds pervade the theory of concurrency control of databases [Pap86]. And it is overly restrictive to limit the representation of computation to just one of automata as states connected by transitions or schedules as events separated by temporal constraints: the features of interest of a given computational process are seen more clearly sometimes as a schedule, sometimes as an automaton, just as physical interactions are seen best in terms of waves or particles depending on the circumstances, with no sharp boundary at the crossover.

Since both models have worthwhile advantages absent from the other, we cannot simply discard one. We therefore propose the following tightening of Trakhenbrot's question. Can the definitions of HDA's and event structures be adjusted in a way that would overcome the respective limitations of each relative to the other, yet without compromising in any way either the formal capabilities or underlying intuitions of either model?

Anyone who has contemplated the capabilities of both frameworks for more than a week is likely to agree that this is a tall order. Nevertheless there exists a simple solution.

2 A Solution

Our proposed solution can be stated succinctly in terms of the following modifications to both frameworks.

For HDA's, their extant formalizations have been based, *inter alia*, on n-categories [Pra91, BJ96], homology [GJ92], and a mildly intricate cubical construction [vG96]. We propose here to model an n-event HDA instead simply as an arbitrary subset of the cartesian power $\{0,1,2\}^n = 3^n$. In fact section 2 of [Pra91] introducing the HDA concept considered that model (with $\{0,1,2\}$ denoted $\{0,T,1\}$) briefly as a stepping-stone to building intuition about HDA's. But at the time 3^n seemed, both to us and apparently to subsequent authors, too primitive and devoid of useful structure to take seriously. As we will see shortly, this hasty dismissal of this easily grasped model was premature, as it admits far more useful structure than one would guess from its simple form.

For event structures, we make two basic changes. First, the logic of the passage to their dual families of configurations has been assumed to date to be two-valued: an event either has or has not happened. Ordinarily one would treat the two truth values of that logic as forming a Boolean algebra. In preparation for the following passage, we treat them instead as forming the unique 2-element quantale. Our first modification to event structures is to extend that logic to a certain 3-valued quantale, namely the one we have studied elsewhere [CCMP91, §3.2] under the rubric $3'$ representing strict-nonstrict or prosset [GP87] order.

Second, we follow [VGP95] in defining event structures so as to allow arbitrary configuration structures and not just those arising from the notions of enabling and conflict. What was not changed there was the 2-valued logic of configuration structures, the net effect then being to make their notion of general configuration structure equivalent to the extensional Chu spaces over 2 studied by Gupta and Pratt as a model of concurrency [Gup93, GP93, Gup94]. Our use of the above three-element quantale will turn out to be equivalent to using Chu spaces over 3, unexpectedly with no mention made by the latter concerning that choice of quantale.

This solves the basic problem in full, at least to the extent defined by its formulation above. For HDA's, all the useful geometric structure can be found in $\{0,1,2\}^n$ in a form that supports process algebra as well or better than any competing HDA model. For event structures, the usual passage to families of configurations works essentially the same way for configurations represented as "3-fuzzy" sets of events having three degrees of membership as it does for ordinary "2-fuzzy" sets. The question of what additional concepts besides enabling and conflict would be useful for this generalized notion of event structure is taken care of automatically by the straightforward yet structurally fruitful approach of permitting arbitrary configuration structures [GP93, VGP95].

3 Definitions

This section defines quantales and Chu spaces as the two main tools for this paper. Chu spaces are objects of type $A \times X \to K$ separating out the respective concerns of *concurrency* as a conjunctive or covariant set A of events, *choice* as a disjunctive or contravariant set X of states, and *temporal logic* as an ambient or stationary set K of "qualia" understood schizophrenically as both delays and truth values. Quantales provide an organizing framework for this schizophrenia: for schedules they organize time as a monoid and logic as a lattice, for automata the converse.

Definition 1 A *Chu space* $(A, =\!\!|, X)$ *over* a set K consists of sets A and X and a binary operation $=\!\!| : A \times X \to K$. We write $=\!\!|(a,x)$ as either $a =\!\!| x$ or $x \models a$. ∎

We read $a =\!\!| x$ as "event a *informs* state x", synonymous with $x \models a$ as "state x *witnesses* event

a."

Chu spaces transform from $(A, \dashv, X)$ to $(A', \dashv', X')$ via a pair (f, g) of functions $f : A \to A'$, $g : X' \to X$ satisfying $f(a) \dashv' x = a \dashv g(x)$ for all $a \in A$ and $x \in X'$, called the *adjointness condition.* Composition is defined by $(f', g')(f, g) = (f'f, gg')$, itself a Chu transform since $f'f(a) \dashv'' x = f(a) \dashv' g'(x) = a \dashv gg'(x)$. The category of Chu spaces over K and their transforms is denoted $\mathbf{Chu}_K$. The role of g is to force f to behave as a homomorphism.

We may identify each element $x \in X$ of an extensional Chu space with the function $\lambda a.a \dashv x$, called a *column* of the space viewed as an $A \times X$ matrix. An *extensional* Chu space is one with no repeated columns. A Chu transform (f, g) between extensional Chu spaces can then be understood equivalently as a function $f : A \to A'$ such that for all $x \in X'$, $x \circ f \in X$. This is because the adjointness condition is equivalent to the requirement $g(x) = x \circ f$. When $K = 2$ (the case for ordinary event structures), and x is understood as (the characteristic function of) a subset of A, $x \circ f$ is the inverse image $f^{-1}(x)$ of x under f. For Chu spaces over 2 that meet the conditions for being a topological space with X consisting of its open sets, Chu transforms are exactly continuous functions. More generally, Chu transforms can be understood as exactly the homomorphisms preserving whatever structure is implicit in the selection of X as a subset of K^A [Pra95, §4]: the smaller X is, the more structure there is to preserve.

Definition 2 A *quantale* [Mul86, Ros90] is a complete join-semilattice Q (and hence a complete lattice) with an associative binary operation $p \cdot q$ that distributes over arbitrary sups on both sides ($p \cdot \bigvee_i q_i = \bigvee_i (p \cdot q_i)$ and $(\bigvee_i p_i) \cdot q = \bigvee_i (p_i \cdot q)$, including the empty sup $p \cdot 0 = 0 = 0 \cdot p$). A quantale *with unit* contains an element 1 satisfying $1 \cdot p = p = p \cdot 1$. A *commutative* quantale satisfies $p \cdot q = q \cdot p$ for all p, q. ∎

In the sequel we take quantale to mean quantale with unit.

The number of quantales with 1, 2, and 3 elements can be readily calculated as respectively 1, 1, and 3. As semilattices, all are chains, and just one of each cardinality is a Heyting algebra. The 3-element quantale we want is not the Heyting algebra but the chain $0 < 1 < 2$ (where 0 and 1 are as in the definition of quantale), which necessarily satisfies $2 \cdot 2 = 2$ (multiply both sides of $1 < 2$ by 2). Every quantale admits a unique implication or "distance measure" $p \to q$, defined as the maximal r satisfying $p \cdot r \leq q$ (and a second dually defined implication $q \leftarrow p$ when not commutative, but our quantale is commutative). Implication for our quantale can be straightforwardly calculated as satisfying $0 \to p = 2 = p \to 2$ for all p, $1 \to 1 = 1$, and otherwise $p \to q = 0$.

Why quantales, and why this quantale? Although Heyting algebras are more general than Boolean algebras, quantales are more general still, making quantale-based logics more broadly applicable than those based on Heyting or Boolean algebras, especially important for logics of time where $p \cdot q$ denotes the accumulated delay of the consecutive delays p and q, which might be truth values, integers, reals, etc. Yet quantales have (just) enough structure to support the constructions of this paper. These two considerations together make quantales the right level of generality. The interest in this particular quantale is that it is the one corresponding to the intuitions underlying HDA's.

4 Geometry of 3^A

Take A to be the set of dimensions of 3^A, which consists of all functions $x : A \to 3$, and let $|A| = n$. The geometry of 3^A is that of a solid cube. Its n-dimensional interior is the constant A-tuple $(1, \ldots, 1)$, its $2n$ $(n-1)$-dimensional faces are those A-tuples exactly one of whose components is not 1. More generally, every A-tuple of 3^A is called a cell, of dimension the number of 1's in it. A cell of dimension d is called a d-cell. For $d > 0$, replacing an occurrence of 1 in a d-cell by 0 or 2 turns that cell into one of its $2d$ *immediate faces*, each a $(d-1)$-cell.

The structure $(3, \sqsubseteq)$ equips 3 with a partial order $\sqsubseteq$ distinct from its quantale order $0 \leq 1 \leq 2$. This defines the *face order* on 3, satisfying $0 \sqsubset 1$, $2 \sqsubset 1$, that is, a $\wedge$-shaped join-semilattice with 1 at the top and no bottom. This join-semilattice structure extends pointwise to 3^A; adjoining a fictitious face at the bottom then completes this to a complete semilattice and hence a complete (but not in general distributive) lattice, called the *face lattice*. (Equivalently, form the distributive lattice 4 by equipping 3 with a bottom, exponentiate to form the distributive lattice 4^A, then identify ("smash") all A-tuples containing at least one bottom, which for $n > 1$ destroys distributivity. The analog of singular homology here is to skip the smashing step, yielding a better-behaved face lattice.)

The HDA interpretation of an A-cube as a concurrent process having n events interprets the 3^A cells of an A-cube as the possible states of an automaton. Each state, as a function $x : A \to 3$, indicates the progress of each event $a \in A$: $x(a) = 0$, 1, or 2 means respectively that a has not started, is running, or has finished.

Having established the basic geometry of the solid A-cube, we turn now to more general shapes. There are two directions, characterizable as cubical complexes and constraints. A cubical complex is a set of cubes of various dimensions and sharing some faces so as to give the set nontrivial connectivity, possibly containing cycles. The constraint approach simply imposes constraints on one A-cube (with n possibly infinite) which are interpreted as forbidding certain of its faces (including in all likelihood its interior). No cycles can arise with the latter.

The former is the standard approach in algebraic topology. Cubical complexes have an attractive definition as presheaves (functors to **Set**) on the category C° where C has as objects all A-cubes (one for each A, for A as large as required, infinite or uncountable if desired), and as morphisms the embeddings of smaller A-cubes as faces of larger ones. Instead of describing C however, it is simpler to describe C°, which has an especially simple description as the category of bipointed sets, sets each equipped with two constants, preserved by homomorphisms. The constants can be understood as formal processes one of which in every state has not yet started, the other of which in every state has already ended. States are homomorphisms to the 3-element bipointed set $\{0, 1, 2\}$ (our quantale 3 in disguise) where 0 and 2 are the two constants, processes are functors from C° to **Set**, and process homomorphisms are natural transformations, all as usual for presheaves. This works beautifully, but unfortunately for it our solution uses the constraint approach, so in the interests of space we must reluctantly abandon it.

The latter is essentially how configuration structures [VGP95] work, generalized from 2-valued logic to 3. A constraint may be understood as a rule for eliminating certain states. For example the constraint that event a precede event b means, for ordinary event structures, that no state may contain b without a, while the constraint that a and b are in conflict means that no state may contain both a and b.

HDA's admit a more expressive constraint language. To begin with, we may distinguish strict precedence $a < b$ from nonstrict precedence $a \leq b$, the basis for [GP87]. The meaning of $a \leq b$ is that in every state $x : A \to 3$, $x(b) \leq x(a)$ in the quantale 3, that is, b may not get ahead of a though it may be as far along.

The meaning of $a < b$ is the same with the added requirement that $x(a)$ and $x(b)$ may not both be 1.

In terms of 3's implication $p \to q$ defined earlier, $a \leq b$ is the requirement that $x(b) \to x(a) \geq 1$ in 3, while $a < b$ strengthens this to $x(b) \to x(a) \geq 2$.

What we would like to do now is to obtain these ostensibly ad hoc rules more uniformly as consequences of x being not just any function but a homomorphism from a set of constraints understood concretely as a quantale-like structure to 3 equipped with the same type of structure.

5 Q-metric Spaces

Definition 3 Let Q be a quantale. Along the lines of [Law73], a pair (A, d), A a set and $d : A^2 \to |Q|$ a function, is called a *Q-metric space* when $1 \leq d(a, a)$ and $d(a, b) \cdot d(b, c) \leq d(a, c)$. ∎

For $Q = 2$, the unique two-element quantale, (A, d) becomes an ordinary binary relation, and these conditions then express respectively reflexivity and transitivity, making (A, d) a preordered set.

For more general Q, an immediate example of a Q-metric space can be seen to be provided by Q itself with $d(p, q)$ taken to be $p \to q$. In particular our quantale 3 can now be understood as also a 3-metric space.

Definition 4 A function $f : A \to A'$ between Q-metric spaces is an *expansion* when for all $a, b \in A$, $d(a, b) \leq d(f(a), f(b))$. ∎

For $Q = 2$, the category of 2-metric spaces and their expansions is just that of preordered sets and their monotone functions. For our quantale 3, it becomes the category of prossets (preorder specification sets) introduced (less morphisms) in [GP87] and defined more in this style in [CCMP91]. Actually prossets impose the stronger condition that $d(a, a) = 1$, the natural requirement that there be no cycles $a < b < a$, so what we have defined here should properly be called preprossets.

We may now define a *simple schedule* to be a 3-metric space or preprosset (A, d). It is simple in that it does not provide for such nonprecedence constraints as permanent or transient conflict.

A *state* of a simple schedule (A, d) is an expansion from it to the 3-metric space 3 defined above.

It may now be verified from 3's implication (defining distance in 3 as a 3-metric space) that simple schedules can express collections of constraints of the forms $a < b$ and $a \leq b$. The former is represented as $d(b, a) = 2$ and the latter as $d(b, a) = 1$.

For any simple schedule (A, d), the set X of all expansions from (A, d) to 3 may be understood as the automaton dual to that schedule. But this define a Chu space $(A, \models, X)$ where $a \models x$ is taken to be $x(a)$. This leads naturally into the next section.

6 A and X as Partial Quantales

We now set up the machinery for reading structure into the rows and columns of Chu spaces. The motivating example is the quantale 2, a distributive lattice. A Chu space $(A, \models, X)$ over 2 can be understood as a partial distributive lattice [GP93] in either of two ways: give all equations that hold between lattice terms in A with meet and join interpreted as intersection (bitwise and) and union (bitwise or), or do the same for X. It can be shown (*op.cit.*) that the Chu space can be recovered up to isomorphism from either equational theory. The theory on A is interpreted as a schedule consisting of necessary temporal constraints on events, while that on X is interpreted as a generalized notion of possible transitions between states.

This is the basis for the Chu space theory of schedule-automaton duality in the case $K = 2$. It extends straightforwardly to any K equipped with any quantale structure, as follows.

Definition 5 A *partial quantale* (A, T) is a set A together with an equational theory T in the language of quantales with A regarded as a set of named constants, such that distinct elements of A are not equated. A *total* quantale is a partial quantale that is *saturated* in the sense that adding one more equation in the same language necessarily identifies two elements of A. ∎

A total quantale is thus a quantale whose theory completely specifies its quantale structure. A partial quantale may be incomplete for want of either equations fully specifying its structure or for want of elements needed to complete it to a quantale. For example $A = \{a, b\}$ with the equations $a = 1$, $a \vee b = b$, defines a two-element partial quantale which with the addition of a third bottom element becomes a total quantale isomorphic to our quantale 3. Note that the language of quantales is not finitary since it permits sups of any cardinality. Hence a theory may include infinite equations.

Definition 6 A function $f : A \to A'$ between partial quantales (A, T), (A', T') is a *homomorphism* just when, acting as a substitution of constants from A' for constants from A, f transforms T into a subset of T'. ∎

That is, a homomorphism preserves every given equation.

Definition 7 Let $\mathcal{A} = (A, \dashv, X)$ be a Chu space over the underlying set of a quantale. The *induced partial quantale* on A is defined as the partial quantale (A, T) where T consists of all quantale equations that hold pointwise for the rows of $\mathcal{A}$. ∎

Theorem 8 *Given extensional Chu spaces $(A, \dashv, X)$ and $(A', \dashv', X')$ and a function $f : A \to A'$, f is a Chu transform if and only if it is a homomorphism of the induced partial quantales on A and A'.*

This is a special case of Theorem 2 of [Pra95] identifying continuity with homomorphism, namely for those properties expressible in the equational language of quantales. These are only a small subset of the set 2^{K^A} of all possible properties of a Chu space A, all of which are covered by the general theorem.

What is shocking about this theorem is that the notion of Chu transform is defined independently of the choice of quantale structure on K, in fact independently of the notion of quantale altogether. This means that no matter which quantale structure is assigned to K, the same set of Chu transforms between the Chu spaces $(A, \dashv, X)$ and $(A', \dashv', X')$ will be all and only the homomorphisms on the induced partial quantale structures on A and A'. This is an unexpected eye in the middle of the hurricane of possible induced quantale structures, which can vary considerably.

7 Process Algebra

Chapter 6 of Gupta's thesis develops the algebra of Chu spaces over 2, both as a process algebra and as a model of linear logic. As far as the latter is concerned, the choice of 2 or 3 is essentially immaterial, and the operations and associated inference rules of linear logic are the same for 3 as for 2. The full paper will make this more concrete with examples of the effect of *plus* $A \oplus B$, *tensor* or orthocurrence [Pra86] $A \otimes B$, and *linear implication* $A \multimap B$ in the HDA setting (but it follows pretty automatically from the definitions).

One small example for now: consider $A \otimes A'$ where A and A' are two copies of a *mutex* b. The events are pairs of events one from each of A and A', four pairs altogether constituting $A \times A'$. The states are those maps $x : A \times A' \to 3$ satisfying the following bilinearity condition: for every $a \in A$

there exists a state $y : A' \to 3$ of A' such that for all $a' \in A'$ $x(a, a') = y(a')$, and for every $a' \in A$ there exists a state $z : A \to 3$ of A such that for all $a \in A$, $x(a, a') = z(a)$. Since $y(a') = y(b') = 1$ and $z(a) = z(b) = 1$ are both outlawed, this means that in every state of $A \otimes A'$, we may not have both $x(a, a') = 1$ and $x(b, a')$ for any $a' \in A'$, and moreover we may not have both $x(a, a') = 1$ and $x(a, b')$ for any $a \in A$. In other words, tensor product ensures in this orthocurrence of the two mutexes that the mutexes are preserved independently for each of A and A', as one would hope.

8 Conclusion

The HDA model 3^A, previously understood only as a stepping stone in introducing the HDA concept, can be equipped with a richer structure than had been previously realized by way of the machinery of Chu spaces over 3. This structure provides the basis for modifications to both HDA's and event structures that bring them together as two views of the one structure, namely processes as subsets of the cube 3^A. This reconciliation preserves the essential intuitions of both HDA's and event structures, along with their algebra and logic.

References

[BJ96] R. Buckland and M. Johnson. Echidna: A system for manipulating explicit choice higher dimensional automata. In *AMAST'96: Fifth Int. Conf. on Algebraic Methodology and Software Technology*, Munich, 1996.

[CCMP91] R.T Casley, R.F. Crew, J. Meseguer, and V.R. Pratt. Temporal structures. *Math. Structures in Comp. Sci.*, 1(2):179–213, July 1991.

[GC93] E. Goubault and R. Cridlig. Semantics and analysis of linda-based languages. In *Proc. 3rd Int. Workshop on Static Analysis*, volume 724 of *Lecture Notes in Computer Science*, pages 72–86, Padova, 1993. Springer-Verlag.

[GJ92] E. Goubault and T.P. Jensen. Homology of higher dimensional automata. In *Proc. of CONCUR'92*, volume 630 of *Lecture Notes in Computer Science*, pages 254–268, Stonybrook, New York, August 1992. Springer-Verlag.

[Gou93] E. Goubault. Homology of higher-dimensional automata. In *Proc. of CONCUR'93*, volume 630 of *Lecture Notes in Computer Science*, pages 254–268, Stonybrook, New York, August 1993. Springer-Verlag.

[Gou95a] E. Goubault. *The Geometry of Concurrency.* PhD thesis, Ecole Normale Superieure, 1995.

[Gou95b] E. Goubault. Schedulers as abstract interpretations of hda. In *Proc. of PEPM'95*, La Jolla, June 1995. ACM Press.

[Gou96a] E. Goubault. Durations for truly-concurrent actions. In *Proceedings of ESOP'96*, number 1058, pages 173–187. Springer-Verlag, 1996.

[Gou96b] E. Goubault. A semantic view on distributed computability and complexity. In *Proceedings of the 3rd Theory and Formal Methods Section Workshop.* Imperial College Press, 1996.

[GP87] H. Gaifman and V.R. Pratt. Partial order models of concurrency and the computation of functions. In *Proc. 2nd Annual IEEE Symp. on Logic in Computer Science*, pages 72–85, Ithaca, NY, June 1987.

[GP93] V. Gupta and V.R. Pratt. Gates accept concurrent behavior. In *Proc. 34th Ann. IEEE Symp. on Foundations of Comp. Sci.*, pages 62–71, November 1993.

[Gun94] J. Gunawardena. Homotopy and concurrency. *EATCS Bulletin 54*, pages 184–193, October 1994.

[Gup93] V. Gupta. Concurrent Kripke structures. In *Proceedings of the North American Process Algebra Workshop, Cornell CS-TR-93-1369*, August 1993.

[Gup94] V. Gupta. *Chu Spaces: A Model of Concurrency.* PhD thesis, Stanford University, September 1994. Tech. Report, available as ftp://boole.stanford.edu/pub/gupthes.ps.Z.

[Law73] W. Lawvere. Metric spaces, generalized logic, and closed categories. In *Rendiconti del Seminario Matematico e Fisico di Milano, XLIII.* Tipografia Fusi, Pavia, 1973.

[Mul86] C.J. Mulvey. &. In *Second Topology Conference*, Rendiconti del Circolo Matematico di Palermo, ser.2, supplement no. 12, pages 99–104, 1986.

[Pap86] C. Papadimitriou. *The Theory of Database Control.* Computer Science Press, 1986.

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

[Pra91] V.R. Pratt. Modeling concurrency with geometry. In *Proc. 18th Ann. ACM Symposium on Principles of Programming Languages*, pages 311–322, January 1991.

[Pra92] V.R. Pratt. Event spaces and their linear logic. In *AMAST'91: Algebraic Methodology and Software Technology*, Workshops in Computing, pages 1–23, Iowa City, 1992. Springer-Verlag.

[Pra95] V.R. Pratt. The Stone gamut: A coordinatization of mathematics. In *Logic in Computer Science*, pages 444–454. IEEE Computer Society, June 1995.

[Ros90] K.I. Rosenthal. *Quantales and their applications.* Longman Scientific and Technical, 1990.

[SC96] V. Sassone and G. L. Cattani. Higher-dimensional transition systems. In *Proceedings of LICS'96*, 1996.

[Tak96] Y. Takayama. Extraction of concurrent processes from higher-dimensional automata. In *Proceedings of CAAP'96*, pages 72–85, 1996.

[vG96] R. van Glabbeek. Bisimulations for higher dimensional automata. Manuscript available electronically as http://theory.stanford.edu/r̃vg/hda, June 1996.

[VGP95] R. Van Glabbeek and G. Plotkin. Configuration structures. In *Logic in Computer Science*, pages 199–209. IEEE Computer Society, June 1995.