Temporal Structures

Ross Casley† Roger F. Crew† José Meseguer‡ Vaughan Pratt†

October 26, 1990

† Dept. of Computer Science, Stanford University, Stanford, CA 94305

† Supported by the National Science Foundation under grant CCR-8814921

‡ SRI International, Menlo Park, CA 94025 and
Center for the Study of Language and Information, Stanford University, Stanford, CA 94305

‡ Supported by the Office of Naval Research under contracts N00014-86-C-0450 and N00014-88-C-0618, and by the National Science Foundation under grant CCR-8707155.

This paper is a revision of a CTCS-89 paper [CCMP89].

Abstract

We combine the principles of the Floyd-Warshall-Kleene algorithm, enriched categories, and Birkhoff arithmetic, to yield a useful class of algebras of transitive vertex-labeled spaces. The motivating application is a uniform theory of abstract or parametrized time in which to any given notion of time there corresponds an algebra of concurrent behaviors and their operations, always the same operations but interpreted automatically and appropriately for that notion of time. An interesting side application is a language for succinctly naming a wide range of datatypes.

1 Introduction

Posets, metric spaces, "closed" automata, and categories have in common the notion of a *space* of points with distances between points. These distances are respectively truth values, reals, languages, and sets.

Distances have two facets, logical and metrical. The logical facet is expressed respectively via implications $p \rightarrow q$ between truth values, comparisons $x \geq y$ between reals, inclusions $L \subseteq M$ between languages, and functions $f: X \rightarrow Y$ between sets. The metrical facet is expressed via a suitable monotone associative operation, respectively conjunction $p \wedge q$, addition $x+y$, concatenation LM, and cartesian product $X \times Y$. These two facets confer on any such set of distances the attributes of an ordered monoid $(D, \leq, \otimes, I)$, having simultaneously the attributes of a poset $(D, \leq)$ and a monoid $(D, \otimes, I)$, with $\otimes$ monotone in each argument with respect to $\leq$. For such cases as our last example, sets as distances, the poset structure is generalized to category structure, so that rather than an ordered monoid we have a monoidal category, for which "monotone operation" must be correspondingly generalized to "functor."

The transitivity law $u \leq v \wedge v \leq w \rightarrow u \leq w$ for a poset, the triangle inequality $d(u,v)+d(v,w) \geq d(u,w)$ for a metric space, the requirement $L_{uv}L_{vw} \subseteq L_{uw}$ for an automaton to be considered closed, and the family of composition operations m_{uvw}: $\mathrm{Hom}(v,w) \times \mathrm{Hom}(u,v) \rightarrow \mathrm{Hom}(u,w)$ for a category, respectively combine these two facets via essentially the same basic triangle inequality.

Each of these classes of generalized metric spaces has a natural notion of map, respectively monotone functions, contractions, morphisms of automata, and functors, in each case making that class a category with those maps as its morphisms.

In computer science some of the common elements of these notions have been unified by organizing distances as an *idempotent semiring*, an ordered monoid whose underlying partial order contains all sups and whose monoidal operation preserves (i.e. distributes over) those sups. The basis for this unification is the $O(n^3)$ time procedure of Roy [Roy59] and independently that of S. Warshall [War62] for computing the transitive closure of a binary relation on n elements presented as an $n \times n$ Boolean (0- and 1-valued) matrix. The unification was hinted at by R.W. Floyd [Flo62] who observed that Warshall's procedure would compute shortest paths instead of transitive closure if Booleans were replaced by nonnegative reals, disjunction by **min**, and conjunction by addition. The uniform expression of this common algorithm in terms of semirings was first described by Robert and Ferland [RF68]. Synonyms for this notion of semiring include regular algebra [Con71], Kleene algebra [Koz80, Koz81], and quantale [Vic89].

Other instantiations of this abstract algorithm were subsequently found. Replacing Booleans in Warshall's algorithm by languages, conjunction by concatenation, and disjunction by union, yields Kleene's algorithm for "closing" an automaton M having n states, viewed as an $n \times n$ matrix of (finitely presented) regular languages, from which one may then easily read off a regular expression denoting the language $L(M)$ accepted by M [AHU74]. Even Gaussian elimination was found to be so describable in the (nonidempotent) ring of reals made a field via $1/(1-x) = 1 + x + x^2 + \ldots$.

Independently and working in a categorical setting, Eilenberg and Kelly [EK66] developed a more categorically sophisticated version of the same generalization of metric space under the name enriched category or V-category. In place of a semiring they used a monoidal category V. Distances became homobjects, a generalization of homset. The logical facet was expressed by the category structure of V: the morphisms of V permitted "comparisons" of homobjects. The metrical facet was expressed by the monoidal structure of V: "consecutive" homobjects were combined by the binary operation of the monoid.

Although Eilenberg and Kelly described enriched categories in their full generality, their motivating applications were confined to V's forming classes rather than sets, an emphasis continued in Kelly's subsequent book on enriched categories [Kel82, p22]. Yet in 1974 F.W. Lawvere [Law73] in an excellent advertisement for the utility of enriched categories emphasized V's that were semirings, hence sets rather than classes, and with category structure merely that of a poset. This brought enriched categories into close proximity to the parallel computer science development. Nevertheless this connection between enriched categories and shortest-path algorithms remained undetected for another 15 years [Pra89].

The present paper starts from the notion of a partial order as a behavior of a "truly concurrent" process, and uniformly extends it to other classes of spaces via the above correspondences. This extension was first proposed by Pratt [Pra84] with just the semiring view in mind; here we extend that proposal to take advantage of the enriched category perspective as well as additional basic operations whose utility were not at all apparent at the time, and develop the resulting framework in detail.

This approach achieves a considerable unification of ideas relevant to concurrency, as well as making connections with other areas to which the semiring and enrichment insights apply. In the concurrency application we characterize time abstractly as an ordered monoid, and more generally albeit speculatively as a monoidal category, whose objects are temporal quantities. From this model of time we construct via enrichment a category of behaviors. A behavior, or computation, is a space whose points represent events and whose distances are to be interpreted as delays between events.

Various natural operations on such spaces correspond to useful constructs for concurrent programming languages. These operations are functors, most of which prove to be definable via familiar categorical constructions. We treat only concurrency and not nondeterminism (choice), in that we work only with single behaviors rather than sets of them representing alternative behaviors.

An appealing feature of this approach is its abstractness. A single framework is developed independently of choice of ordered monoid or monoidal category. Instantiating the whole framework for a particular monoidal structure yields the corresponding model of concurrency incorporating that structure as its notion of time, with all the operations of the framework likewise instantiated. The development lends itself to the application of categorical methods.

A practical application of this perspective is to improving the organization of current theories of real time in concurrency modeling. A case in point is the recent work of H. Lewis [Lew90]. Lewis works with state diagrams each of whose transitions is labeled with a set of $O(n^2)$ intervals, with larger sets at later transitions, per his Figure 6. In our framework the essence of this information would be captured with one real labeling each edge of the *transitive closure* of the diagram, with the delay from u to v being a lower bound whose matching upper bound (to form an interval) is the negation of the delay from v to u.

The ordered monoids that have previously been found useful in this setting are all useful here for one view or another of time. In addition we identify a class of finite generalizations of the ordered monoid **2** of truth values which we call the *idempotent closed ordinals* or ico's. There are 2^{n-2} such closed or residuated ordered monoids with n elements, exactly one of which is cartesian closed, proved via a pretty representation theorem. We describe natural applications for the two three-element ico's $\mathbf{3}$ and $\mathbf{3}'$, and show where each has in effect been used in the concurrency literature.

The operations on spaces ordinarily considered in the context of shortest-path algorithms and their cousins can be collectively understood as Kleene's *regular* operations $L+M$, LM, and L^*, defined by Kleene only for languages but all equally meaningful for the other domains, even the one for Gaussian elimination (with $x^* = 1/(1-x)$). In terms of matrices these are the operations of pointwise sum of two $m \times n$ matrices M, N to yield an $m \times n$ matrix $M+N$, product of an $m \times k$ matrix M by a $k \times n$ matrix N to yield an $m \times n$ matrix MN, and reflexive and/or transitive closure of an $n \times n$ matrix M to yield an $n \times n$ matrix M^*. A reasonably close connection between such matrices and spaces can be made by regarding rectangular $m \times n$ matrices as complete bipartite graphs from m vertices to n vertices, and in the other direction ordinary (nonbipartite) directed graphs as square matrices, with distances entering as edge labels.

This paper adds to these regular operations a number of other operations such as disjoint union or juxtaposition, tensor product, concatenation, exponentiation, and useful variations on these obtained by generalizing products to pullbacks and coproducts to pushouts. These operations are generally better matched to the concurrency modeling application than the regular operations, both extrinsically and intrinsically. Extrinsically juxtaposition captures concurrence, tensor product captures orthocurrence, etc. And intrinsically these operations impose few if any constraints on relationships between vertex sets of their arguments, unlike the regular operations.

An early and striking example of these "nonregular" operations is provided by Birkhoff's arithmetic [Bir37, Bir42] of posets up to isomorphism under addition, multiplication, and exponentiation each of two kinds, cardinal and ordinal. Birkhoff's application was to the arithmetic of cardinals and ordinals, which he proposed to unify by regarding both as posets, with cardinals as discrete posets and ordinals as linear. In place of two sorts of data he then had two sorts of operations.

In relating Birkhoff arithmetic to concurrent programming we make the connections cardinal-concurrent and ordinal-sequential. Discrete posets model the purely concurrent behaviors (no sequentiality) while linearly ordered posets model the purely sequential. The cardinal operations map discrete sets to discrete, i.e. they preserve concurrency, while the ordinal operations map linear sets to linear, i.e. they preserve sequentiality.

Our framework can then be viewed as a generalization of Birkhoff arithmetic, in several directions: several additional operations, many other metrics besides **2**, provision of labels on points, and setting Birkhoff arithmetic in a suitable categorical framework. We have not however succeeded in finding the right categorical expression of either ordinal multiplication (i.e. lexicographic product) or ordinal exponentiation, which we therefore raise as an interesting problem.

There is a recursive aspect to the enrichment process that permits a further generalization of this framework. We introduce an operation we call $\mathcal{D}!$, the enriched category term for which is V-Cat,[1] which takes a symmetric monoidal category $\mathcal{D}$ and returns the symmetric monoidal category $\mathcal{D}!$ of all small $\mathcal{D}$-categories. For example if $\mathcal{D}$ is the monoidal category $\{0, 1\}$ of truth values then $\mathcal{D}!$ is the monoidal category of preordered sets. This construction can therefore be iterated to yield $\mathcal{D}!!$, $\mathcal{D}!!!$, etc.

Behaviors as sets of events require not only delay information between events but information describing each event. That is, we wish to label vertices independently of the labeling of edges. From a set theoretic perspective there is nothing to this. However from a category theoretic perspective, with some operations defined as limits or colimits the presence of labels is a nontrivial complication; consider for example coproducts in the category of vertex-labeled posets. We define the category of $\mathcal{E}$-labeled $\mathcal{D}$-spaces, each of whose objects is an object d of $\mathcal{D}$ (we call such a d a $\mathcal{D}$-space) paired with an object e of $\mathcal{E}$, along with a function $f: U_d \to V_e$ from the underlying set of d (the points of the space d) to the underlying set of e (typically the set e itself, construed as an alphabet of labels) serving as a vertex labeling function. The appropriate category of such

[1] We prefer $\mathcal{D}$ to V as connoting distance or delay.

$\mathcal{E}$-labeled $\mathcal{D}$-spaces is the comma category (U, V).

To make the comma-category construct iterable analogously to the iterability of enrichment we need to extend its arguments so as to carry both the structure we need for enrichment (i.e. a symmetric monoidal category) and that for the comma construction (hence we need a forgetful functor). We denote this extension of the comma construction to these extended arguments by $\mathcal{D} \rhd \mathcal{E}$.

With these two operations, along with certain constants, we now have a language with such expressions as $\mathbf{1}!$, $\mathbf{3}!! \rhd \mathbf{1}!$, $\mathbf{R}!$, etc. The succinctness of each expression belies its content. For example the expression $\mathbf{1}!!!$ does not merely hint at the category of all 2-categories but specifies it in full detail, complete with all internal features such as the interchange law, 2-functors between 2-categories, etc. Moreover it supplies some external features: it is a closed category, and is equipped with a forgetful functor to **Set** taking each 2-category to the set of its objects. However it is not a 3-category as it should be, or even a 2-category, and has no other useful forgetful functors such as to **Cat**. These restrictions reflect our particular recursive construction of categories, which yields only semiconcrete symmetric monoidal categories.

2 Operations

The motivating application of our framework is to define an algebra of concurrent behaviors (runs, computations), independently of any particular choice of notion of time. In this section we describe the desired operations of such an algebra, and illustrate them for *dual* metric spaces, spaces in which distance d from u to v indicates that v must follow u by *at least* d units (metric spaces would replace "at least" by "at most").

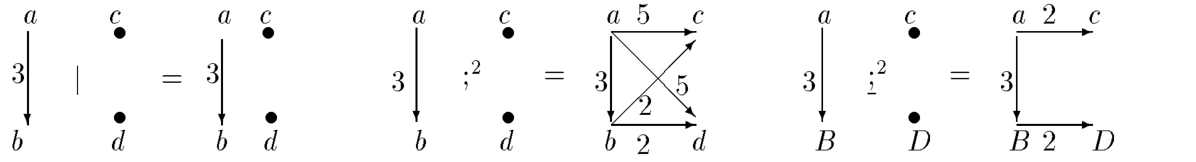
In formal language theory, concatenation is defined on individual strings as well as on languages (sets of strings) whereas union and Kleene star are defined only on languages. In the framework of the present paper strings are generalized to *behaviors*, defined as labeled spaces, with languages correspondingly generalized to *processes* as sets of behaviors. We shall define only operations on individual behaviors, hence including concatenation but excluding union and Kleene star. The operations we treat include all behavior operations of the process language of [Pra86], namely concurrence, orthocurrence, concatenation, and local concatenation, as well as new operations synchronized concurrence, exponentiation, product, and local product. The process operations of that paper are linearization, union, intersection, complement, star, augment closure, and prefix closure, whose definition we defer for now pending the appropriate integration of our framework with the current understanding of nondeterminism.

We now illustrate and define three forms of sum: concurrence $p|q$, concatenation $p;^d q$, and local concatenation $p\underline{;}^d q$. In these examples all edges are labeled with reals or ∞, with absence of an edge interpreted as distance $-\infty$. These examples may be taken as pomset examples by ignoring the edge labels (equivalent to taking $-\infty$ as 0 and everything else as 1), and as automata examples by treating distance d as the set of all strings of length at most d (making $-\infty$ the empty language).

In all of these forms of sum, the set of points of the sum is the disjoint union of those of p and q.

That is, the basic step in forming the sum is to juxtapose p and q.

Concurrence $p|q$ is the least constrained form of sum. Labels on points, and distances within p and q, remain unchanged, while the distance from an event of p to one of q is $-\infty$, and likewise from q to p. *Disjoint* concurrence differs from concurrence in that the labels from p and q are "marked" to distinguish them: the label a becomes a_0 or a_1 according to whether the point it labels is from p or q.



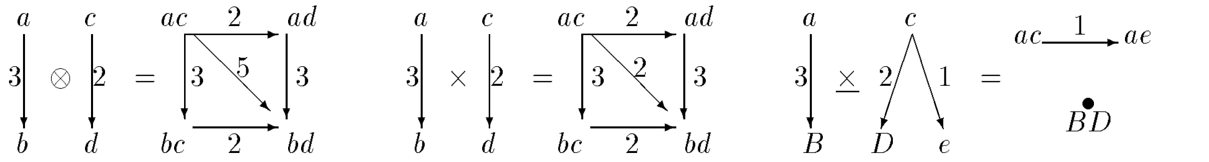
Concurrence $(a;^3 b)|(c|d)$ Concatenation $(a;^3 b);^2 (c|d)$ Local Concatenation $(a;^3 B)\underline{;}^2(c|D)$

Figure 1. Additive Operations.

Concatenation $p;^d q$ is like $p|q$, differing only with respect to the distances from p to q, which are the least possible distances no less than d. Disjoint concatenation is to concatenation as disjoint concurrence is to concurrence.

Local concatenation $p\underline{;}^d q$ is intermediate in strength between concurrence and concatenation: it only imposes the additional distance constraints of concatenation between colocated elements of p and q. For the above example we have identified location with case of label: lower case at one location, upper at the other.

We now illustrate and define three forms of product, namely orthocurrence $p \otimes q$, product $p \times q$, and local product $p\underline{\times}q$.



Orthocurrence $(a;^3 b) \otimes (c;^2 d)$ Product $(a;^3 b) \times (c;^2 d)$ Local Product $(a;^3 b)\underline{\times}c(;^2 D, ;^1 c)$

Figure 2. Multiplicative Operations.

The underlying set of each of these products is the cartesian product of the underlying sets of their arguments (or a subset thereof in the case of local product), while their labels are corresponding tuples: thus if μ assigns labels to points then the label $\mu(\langle u, v\rangle)$ in a product is $\langle \mu_p(u), \mu_q(v)\rangle$.

Orthocurrence $p|q$ is distinguished from other products by the distance from $\langle u, v\rangle$ to $\langle u', v'\rangle$ being $d(u, u') + d(v, v')$. *Product* $p \times q$ takes this distance to be $\min(d(u, u'), d(v, v'))$. *Local product* $p\underline{\times}q$ is obtained from product by deleting all points with mixed locations (again indicated by case in the example), and reducing the distance between tuples at distinct locations to $-\infty$.

For pomsets orthocurrence and product coincide, both combining the distances $d(u,u')$ and $d(v,v')$ via conjunction. For automata they differ: orthocurrence uses concatenation, product intersection.

Basic Constants. The empty schedule, denoted 0, has no events. It is the identity for all sums: concurrence, concatenation, and local concatenation. The unit schedule, denoted I, has one event with self-distance 0, and has the singleton alphabet $\{\bullet\}$, whence that one event is labeled $\bullet$. Up to isomorphism I is the identity for orthocurrence. The top schedule, denoted 1, differs from I in having self-distance ∞. Up to isomorphism it is the identity for product. (The isomorphism is not only between events but between labels, via the evident isomorphism between Σ and $\Sigma \times \{\bullet\}$.)

We have illustrated these operations and constants for (dual) metric spaces, and indicated how to derive the corresponding pomset and automata examples. However these operations and constants also admit of obvious interpretations for the metrics themselves. Concurrence is respectively max, disjunction, and union for each of the ordered monoids consisting of truth values, reals, and languages, and is the same operation as concatenation and local concatenation. Orthocurrence is respectively arithmetic sum, conjunction, and concatenation. Product is respectively min, conjunction, and intersection, and is the same operation as local product.

3 Monoidal Categories and their Functors

Monoidal categories and enrichment are less well established in the computing literature than such other aspects of category theory as adjunctions. We therefore recall here enough details of these notions to make this paper self-contained at least on a first reading by those familiar with at least adjunctions. In addition our treatment will serve to define the perspective on these topics that we will assume of the reader, and to coordinate this perspective with the rest of the paper. Considerably more information on these topics can be found in the books of Mac Lane [Mac71] and Kelly [Kel82].

3.1 Monoidal Categories

Informally, a monoidal category amounts to a structure that is both a monoid and a category. Formally a *strict monoidal category* $\mathcal{D} = (D, \otimes, I)$ is a category D together with a functor $\otimes\colon D^2 \to D$ called the *tensor product* [2] and an object I of D called the *unit*, such that the object part and the morphism part of D each form a monoid under $\otimes$ with respective identities I and 1_I. We refer to $(\otimes, I)$ as a *monoidal structure* for the category $\mathcal{D}$.

The structure $(\mathbf{Set}, \times, \{\bullet\})$ with $\times$ cartesian product is not strict monoidal because in general $X \times (Y \times Z)$ is not equal to $(X \times Y) \times Z$, they are merely isomorphic. Moreover there is a particular isomorphism we would like to be able to *assume* is the one meant when we say that they are isomorphic, namely $\alpha_{XYZ}\colon (X \times Y) \times Z \to X \times (Y \times Z)$ defined as $\alpha(\langle\langle x, y\rangle, z\rangle) = \langle x, \langle y, z\rangle\rangle$. Similarly particular isomorphisms take the place of the two identity laws for the unit $I = \{\bullet\}$.

[2] Mac Lane writes $\otimes$ as $\Box$.

We therefore define a more general notion. A *monoidal category* $\mathcal{D} = (D, \otimes, I, \alpha, \lambda, \rho)$ is as for strict monoidal categories but specifying natural isomorphisms $\alpha\colon (c \otimes d) \otimes e \cong c \otimes (d \otimes e)$, $\lambda\colon I \otimes d \cong d$ and $\rho\colon d \otimes I \cong d$ in place of the usual three-axiom basis for the theory of monoids. A *symmetric* monoidal category specifies an additional natural isomorphism $\gamma\colon d \otimes e \cong e \otimes d$. We shall confine ourselves to symmetric monoidal categories throughout, and understand "monoidal" to mean symmetric monoidal at all times.

The intent of these isomorphisms is that they be canonical: if an element of one set corresponds to an element of another set at all, this is a universal or global correspondence. In particular there may be at most one such isomorphism between two sets, and in the case of a set in isomorphism with itself that isomorphism must be the identity. This intuition is formalized via certain *coherence conditions*, whose effect is that if there are multiple ways to infer by transitivity two isomorphisms between two sets, those isomorphisms must be the same. A strict monoidal category amounts to a monoidal category whose such isomorphisms are all identities.

A monoidal category C is *left closed* (*right closed*) when $- \otimes b$ $(a \otimes -)$ has a right adjoint. When both exist it is called biclosed. When C is symmetric, either left closed or right closed implies biclosed, and in this case we simply call it closed. The right adjoint to $- \otimes b$ is notated $-^b\colon C \to C$ and defined by an isomorphism $\mathrm{Hom}(a \otimes b, c)$ and $\mathrm{Hom}(a, c^b)$ natural in a and c. Setting a to I yields via λ the natural isomorphism $b \to c \cong I \to c^b$, that is, Hom is naturally isomorphic to $\mathrm{Hom}'\colon C^{\mathrm{op}} \times C \xrightarrow{-^-} C \xrightarrow{\mathrm{Hom}(I,-)} \mathbf{Set}$. In this sense $-^-$ *represents* $\mathrm{Hom}(b, c)$, making it the *internal hom*(functor). As $\mathbf{R}^{\mathrm{op}}_{\geq}$ in example (3) below shows, I need not determine $\otimes$ and hence the internal hom even up to isomorphism, showing that it is possible for the internal hom to contain information not present in the external hom even together with I.

Since left adjoints preserve colimits, we have in any closed category $(A + B) \otimes C \cong A \otimes C + B \otimes C$ and $C \otimes (A + B) \cong C \otimes A + C \otimes B$ when the coproduct $A + B$ exists and $A \otimes 0 \cong 0 \cong 0 \otimes A$ when the initial object 0 exists.

A category D with all finite products determines the *cartesian* symmetric monoidal category $(D, \times, 1)$. When the latter is closed (common but not universal, e.g. **Top**), D is said to be *cartesian closed.*

3.2 Examples of Monoidal Categories

We illustrate the above definitions with the monoidal categories we will be using in KL. Most of these will be closed and bicomplete, the kind we are most interested in.

(1) Each successor ordinal $\mathbf{n{+}1}$, as an $n{+}1$-object category with $\binom{n+2}{2}$ morphisms $i \to j$ where $0 \leq i \leq j \leq n$, forms the cartesian closed category algebraic topologists call $[n]$. That is, the symmetric monoidal structure is $(n{+}1, \wedge, n)$, and the right adjoint of $\wedge$, the internal hom $-^-$, must be the largest a such that $a \wedge b \leq c$, namely n when $b \leq c$ and c otherwise. The product of two such cartesian closed ordinals or cco's is the cco $[m] \times [n] = [m + n]$.

But the category $\mathbf{n{+}1} = [n]$ admits other monoidal structures, all strict since $\mathbf{n{+}1}$ has no non-identity isomorphisms, and all bicomplete. Of these, 2^n are idempotent $(x \otimes x = x)$, since each is representable as a permutation of $\{0, 1, \ldots, n\}$ whose domain partitions into two blocks on one of which the permutation is monotone and on the other antimonotone.

Such a permutation can be written down by writing $0, 1, 2, \ldots, I$ (the monotone block) from left to right and then reversing direction so as to write $I+1, I+2, \ldots, n$ (the antimonotone block) from right to left interleaved arbitrarily with the monotone block, leaving I at the far right. For $n = 3$ this can be done in eight ways, namely 0123, $01_{3}2$, $0_{3}12$, $0_{32}1$, $_{3}012$, $_{3}0_{2}1$, $_{32}01$, $_{321}0$, here distinguishing the two blocks by writing the second as subscripts. Then $x \otimes y$ is taken to be the earlier of x and y in the permutation. Since I is always at the right end of the permutation it will always be the identity of $x \otimes y$. For $\otimes$ to be a functor it suffices that it be monotone, which the reader may verify.

For $n > 0$ exactly half of these are closed. For to be closed we require that for any b and c there be a largest a for which $a \otimes b \leq c$. Setting $c = 0$ shows that $0 \otimes b$ must be 0 for all b, whence a necessary condition is that the permutation start with 0. But this is also sufficient since there then exists a solution in a to $a \otimes b \leq c$, namely $a = 0$; finiteness then ensures a largest solution. These then are the *idempotent closed ordinals* or ico's.

There is therefore a unique ico **2**, namely [1]. Here $\otimes$ is $\wedge$ and $I = 1$. (The nonclosed one has $\otimes = \vee$ and $I = 0$.) This category provides the metric for preordered sets. This two-valued cartesian closed logic is that of ordinary precedence, where for any pair u, v of events there are two cases: either u is constrained to precede v, or not.

There are two closed categories on 3, each appearing implicitly in one of H. Gaifman's two papers on concurrency. In each, the elements of 3 represent strengths of temporal precedence constraint between two events, with 0 representing no constraint. In the noncartesian case, call it $\mathbf{3}'$, 1 represents nonstrict temporal precedence and 2 strict, with $1 \otimes 2 = 2$ $(u \leq v < w \rightarrow u < w)$. This structure is hidden in the two-relation "prosset" model [GP87] used in the proof of Kahn's principle relative to a pomset-based semantics of nets [Pra86].

For the cartesian closed **3**, 1 denotes "temporal" or accidental order and 2 causal [Gai89]. Here $1 \otimes 2 = 1$, that is, if u accidentally precedes v whereas v causes (necessarily precedes) w, then we may only infer that u accidentally precedes w. Thus one may identify the logic of causal and accidental precedence with the cartesian closed category **3**.

Of the four idempotent closed categories on 4, described by the first four permutations on the list five paragraphs above, the second and third have the same I, namely 2, but their internal homfunctors differ at 1^1. Composing $\mathrm{Hom}(I, -)$ with either yields the external homfunctor for 4, whence the category and I together are not sufficient to determine either $\otimes$ or the internal homfunctor. However neither of these are cartesian closed, and in fact for all n the cartesian closed structure is the unique one with $I = n$. We do not as yet have a natural application for any of the structures on 4.

(2) We have already discussed **Set** with $\otimes$ taken to be cartesian product, as a basic example of a nonstrict category. This monoidal structure for **Set** is cartesian closed, and will always be the one we have in mind when referring to **Set** as a monoidal category. In the case of **Set** the internal homfunctor is *identified with* its external one.

(3) Let $\mathbf{R}_\wedge$ denote the real numbers together with ∞ and $-\infty$ with the usual ordering and considered as a category in the usual way. Now $\mathbf{R}_\wedge$ is bicomplete, and furthermore is cartesian closed with $a \otimes b = a \wedge b$, $I = \infty$, and internal homfunctor $c^b = \infty$ for $b \leq c$ and c otherwise (cf. the cartesian closed ordinals above). $\mathbf{R}_\wedge$ is isomorphic to its opposite $\mathbf{R}_\vee^{\mathrm{op}}$, the reals with their order reversed and with $\otimes$ now $\vee$. $\mathbf{R}_\vee^{\mathrm{op}}$ is the metric used for so-called ultrametric spaces.

However a more useful closed structure for our purposes will be that obtained by taking tensor product to be not min but $+$, making the unit 0, and with $\infty + (-\infty) = -\infty$, necessary in order to be closed. We denote it by $\mathbf{R}$, or $\mathbf{R}_+$ when there might be confusion. To be closed requires c^b satisfying $a + b \leq c \cong a \leq c^b$, for which $c^b = c - b$ is the patently obvious solution. Hence $\mathbf{R}$ is bicomplete and closed, but of course not cartesian closed since, as we saw with the ordinals, a category admits at most one cartesian closed structure.

We use $\mathbf{R}$ to represent lower bounds on delay. A distance of 5 units from event u to event v means that v must wait at least 5 units after u. Thus a delay of 0 from u to v simply asserts that v follows u, not necessarily strictly.

Now a delay of -5 units from u to v indicates that v may precede u by at most 5 units. Hence we can express upper bounds as negative lower bounds in the opposite direction. So to indicate that v must follow u by 2 to 5 units we bound from below the delay from u to v by 2 and that from v to u by -5.

Combining these two directions, we may read the two oppositely oriented edges between u and v as together defining an interval.

Here as with $\mathbf{R}_\wedge$, $\mathbf{R}$ is isomorphic to its opposite $\mathbf{R}^{\mathrm{op}}$, the reals with their order reversed, however this time with the only change to the monoidal structure being $\infty + (-\infty) = \infty$, this being the one bit of asymmetry in $\mathbf{R}$. As we have seen, $\mathbf{R}^{\mathrm{op}}$ is the monoidal structure associated with ordinary metric spaces.

A sometimes useful closed subcategory of $\mathbf{R}^{\mathrm{op}}$, namely $\mathbf{R}_\geq^{\mathrm{op}}$, is obtained by omitting the negative reals and $-\infty$ [Law73]. The internal homfunctor then becomes *truncated* subtraction, in which negative differences are rounded up to 0. This is the category of generalized metric spaces; if the further restrictions are made that distances be symmetric and distinct points are a nonzero distance apart then we have the usual category of metric spaces and their contractions, modulo a detail about constant factors in the contractions. $\mathbf{R}_\vee^{\mathrm{op}}$ may be similarly truncated and its I set to 0 to yield Lawvere's basic example [Law73] of a cartesian and a noncartesian closed structure on the same category with the same I.

(4) Take $\mathbf{SR}$ to be the category with objects arbitrary sets of reals and with morphisms $X \to Y$ just when $X \subseteq Y$. This poset is cartesian closed when we take $\otimes = \cap$ and $I = \mathbf{R}$. As with $\mathbf{R}$ however we shall prefer a different, hence noncartesian, closed structure, namely $X \otimes Y = \{x + y \mid x \in X, y \in Y\}$ with $I = \{0\}$, and with internal homfunctor $Z^Y = \{x \in \mathbf{R} \mid \{x\} + Y \subseteq Z\}$.

The meaning of a set as a delay is that it consists of the disallowed actual delays. Thus $\emptyset$, like $-\infty$ in $\mathbf{R}$, is no constraint while $\mathbf{R}$, like ∞, disallows all delays.

We may now find $\mathbf{R}$ and $\mathbf{R}^{\mathrm{op}}$, as well as $\mathbf{R}^{\mathrm{op}}$ truncated at 0, as subcategories of $\mathbf{SR}$. We note in particular that $\mathbf{R} + \emptyset = \emptyset$, corresponding to $\infty + -\infty = -\infty$ in $\mathbf{R}$.

(5) Any monoid $(M, \cdot, \epsilon)$ automatically forms a strict monoidal category by taking the set M as a discrete category. Alternatively M may be taken as indiscrete (the maximal preorder on M). We refer to these as discrete and indiscrete monoids respectively.

For a discrete monoid to be closed means simply that for all b, c, $a + b = c$ has a solution in a, whence closed discrete monoids coincide with groups. No nontrivial discrete monoid has finite products or coproducts.

An indiscrete monoid on the other hand is trivially closed with all limits and colimits. Only the fact that it is strictly monoidal saves it from total anarchy. A nonstrict monoidal codiscrete category is no more than an arbitrary binary operation on a pointed but otherwise indiscrete set; nevertheless all such are bicomplete and closed.

(6) The category **Pos** of partially ordered sets forms a subcategory of the cartesian closed category **Pros** of preordered sets. The tensor product is direct or cardinal [Bir42] product of preorders. The cartesian closed structure of **Pros** is inherited from that of its underlying metric, namely the cartesian closed **2**. The following section on enrichment treats the passage from the metric to the spaces in this example and the next two.

By the same token Gaifman's computations with distinct accidental and causal orders [Gai89] form a subcategory of a cartesian closed category of such computations with cycles allowed. As with **Pros** the cartesian closed structure is inherited from that of its underlying metric, here **3**.

The Gaifman-Pratt "prossets" or preordered specification sets [GP87], each consisting of a set with an irreflexive partial order $<$ and a preorder $\leq$, with $u < v \leq w \to u < w$, form a subcategory of a noncartesian closed category of "preprossets" in which $<$ is itself just a preorder, but still meeting the condition $u < v \leq w \to u < w$.

3.3 Monoidal Functors

Monoidal functors, as the appropriate morphisms of monoidal categories, should preserve both the monoidal and category structure. Just as strict monoidal categories motivate monoidal categories, so do strict monoidal functors motivate monoidal functors. A *strict monoidal functor* $F: \mathcal{D} \to \mathcal{D}'$ between monoidal categories $\mathcal{D}, \mathcal{D}$ is a functor $F: D \to D'$ between their underlying categories satisfying $Fx \otimes' Fy = F(x \otimes y)$ and $I' = FI$ for objects and similarly for morphisms. (The unit morphism is often written 1 instead of I or 1_I.)

Definition A *lax monoidal functor* (F, τ, t): $(D, \otimes, I, \alpha, \lambda, \rho, \gamma) \to (D', \otimes', I', \alpha', \lambda', \rho', \gamma')$ consists of a functor $F: D \to D'$, a natural transformation $\tau_{xy}: Fx \otimes' Fy \to F(x \otimes y)$, and a morphism $t: I' \to FI$ of D', with certain coherence conditions requiring that natural transformations constructed from τ and t commute with $\alpha, \lambda, \rho, \gamma$ and all the other natural transformations corresponding to the equations of the theory of commutative monoids [EK66].

When τ and t are both isomorphisms or both identities we call F respectively *strong* or *strict*. For the remainder of this paper the default will be strong, that is, we take "monoidal functor" to mean "strong monoidal functor." In particular we take the morphisms of the category **SMON** of large (symmetric) monoidal categories to be the strong monoidal functors.

We wish to be able to refer to the "points" of spaces. This is accomplished by the next definition.

Definition. The category **COSMON** is the slice (comma) category **SMON/Set**. Its objects are pairs $(\mathcal{D}, U)$ called *semiconcrete monoidal categories*, where $\mathcal{D}$ is a (large) monoidal category and $U_{\mathcal{D}}: \mathcal{D} \to$ **Set** is a monoidal "forgetful functor" for $\mathcal{D}$. The morphisms $F: (\mathcal{D}, U) \to (\mathcal{D}', U')$ of **COSMON** are monoidal functors satisfying $U = U'F$, $\tau_U = \tau_{U'}\tau_F$, $t_U = t_{U'}t_F$.

Our interest in **COSMON** is that its members carry enough structure to support the definitions of the operations $\mathcal{D}!$ and $\mathcal{D} \rhd \mathcal{E}$ constructing categories of spaces and labeled spaces. While this adds some complexity to the development of enrichment, it does demonstrate the feasibility of adding such structure when needed. It would be particularly useful to have a characterization of what structures can be so added.

All the examples of monoidal categories in our list above belong to **COSMON**; of these, only the discrete monoids are not bicomplete. **Set**, **Pos**, etc. are large objects in **COSMON**. The ordinal **1** is the null object of **COSMON**.

Our insistence on the identity $U = U'F$, as opposed to just a natural transformation from U to $U'F$, makes for a clean division of labor between **COSMON**, where the kind language KL operates, and the objects of **COSMON**, where PSL operates. The significance of this identity is that, via functors of **COSMON**, KL may act on kinds without disturbing the underlying sets of objects of those kinds.

For example when we replace a numeric metric by a Boolean one we *identically* preserve the vertex sets of the affected behaviors; only the distances between their elements change, from numbers to truth values. In PSL on the other hand we work in a fixed object $\mathcal{D}$ of **COSMON**, where we may do violence to vertex sets and alphabets via morphisms of $\mathcal{D}$, but where we cannot change to another $\mathcal{D}$ and so have no control over the metric or the kind of alphabet.

As a pleasant fringe benefit the bidirectionality of the identity in $U = U'F$ permits the domain of $\rhd$ to consist of all pairs F, G of functors of **COSMON**, not just those for which there exists a natural transformation from $U'F$ to U. However this benefit would accrue even from a natural isomorphism $h: U \to U'F$ (assuming h commutes with both τ and t), whereas identity results in total separation of the domains of KL and PSL.

3.4 Examples of Monoidal Functors

(1) Consider functors $F, G: \mathbf{R} \to \mathbf{2}$, defined by $F(x) = (x \geq 0)$ and $G(x) = (x > -\infty)$. Each functor equips its target with an interpretation. F interprets $u \leq v$ as saying that u must precede v, while G makes it say that v can only precede u by a bounded amount. Only G however is strong monoidal, since $F(1 + -1) = F(0) = 1$ while $F(-1) \wedge F(1) = 0$. It is straightforward to show that

G and the constantly true functors are the *only* strong monoidal functors from $\mathbf{R}$ to $\mathbf{2}$. In fact with the obvious order on the functors from $\mathbf{R}$ to $\mathbf{2}$ to make $\mathbf{2}^{\mathbf{R}}$ a category, we have $\mathbf{2}^{\mathbf{R}} \cong \mathbf{2}$ in **COSMON**, with the constant functor in $\mathbf{2}^{\mathbf{R}}$ corresponding to 1 in $\mathbf{2}$.

Now consider any functor F from $\mathbf{2}$ to $\mathbf{R}$. F must send 1 to 0, so $F(0)$ cannot be positive. Moreover $F(0) = F(0 \wedge 0) = F(0) + F(0)$ whence $F(0)$ must be either $-\infty$ or 0. Hence $\mathbf{R}^{\mathbf{2}} \cong \mathbf{2}$ (again with $\mathbf{R}^{\mathbf{2}}$ made a category via the obvious order on its two functors), with the constant functor in $\mathbf{R}^{\mathbf{2}}$ corresponding to 1 in $\mathbf{2}$.

(2) The endofunctor $x/2$ from $\mathbf{R}$ to $\mathbf{R}$ that simply halves its argument is clearly monoidal. This is a "speedup" functor that allows one to pass to a world where everything happens twice as fast. In fact from $F(a+b) = F(a) + F(b)$ and $F(0) = 0$ we infer that the linear speedups and slowdowns and their negations (time reversals) are the only functors, and thus we have $\mathbf{R}^{\mathbf{R}} \cong \mathbf{R}$. The ∞ and $-\infty$ speedups are included, and their range in each case is isomorphic to the noncartesian closed ordinal $\mathbf{3}'$ with the convention that $0 \otimes \infty = 0 = 0 \otimes -\infty$.

(3) We leave it to the reader to verify that $\mathbf{2}^{\mathbf{2}} \cong \mathbf{2}$ with the constant functor corresponding to 1 as always.

4 Enriched Categories and the ! Operator

We now introduce the basic notions of enriched categories. Our hope is that these will be made more accessible via definitions that are not only close at hand but presented from the same perspective as the applications. Enriched categories or V-categories are treated briefly by Arbib and Manes [AM75] and exhaustively and precisely by Kelly [Kel82]. Section I-8 of Mac Lane also touches on them, without calling them such. As far as we are aware ours is the first proposed engineering application of Lawvere's vision [Law73] of enriched categories as a combined generalized metric and logic.

4.1 $\mathcal{D}$-categories

The essence of an enriched category may be found in an ordinary category C. Write $\delta(u, v)$ for the set of morphisms from u to v. Associated with each triple u, v, w of objects of C is a function $m_{uvw}\colon \delta(u, v) \times \delta(v, w) \to \delta(u, w)$ defining composition. And to each object u there is a function $j_u\colon \{\cdot\} \to \delta(u, u)$ picking out the identity element $1_u \in \delta(u, u)$.

We pass from this account of an ordinary category C to the notion of a category C *enriched in* $\mathcal{D}$, or $\mathcal{D}$-category, by requiring each $\delta(u, v)$ to be an object of $\mathcal{D}$ instead of a set, and each m_{uvw} and j_u be a morphism of $\mathcal{D}$ instead of a function. The class of objects of C do not participate in this passage: the objects of C remain a class. A functor $F\colon A \to B$ participates partially in this passage: when we view it as the $\mathcal{D}$-functor $F\colon A \to B$, its object part remains unchanged but its action on morphisms, viewed for each pair uv of objects of A as a function $F_{uv}\colon \delta^A(u, v) \to \delta^B(F(u), F(v))$, becomes a morphism of $\mathcal{D}$ between homobjects $\delta^A(u, v)$ and $\delta^B(F(u), F(v))$ of A, B respectively.

From this viewpoint, ordinary categories and functors between them are **Set**-categories and **Set**-functors, that is, categories and functors enriched in **Set**.

Definition. A *$\mathcal{D}$-category* $A = (V, \delta, m, j)$, or *category enriched in $\mathcal{D}$*, consists of a set V, a function $\delta\colon V^2 \to \mathrm{ob}(\mathcal{D})$, and families of morphisms of $\mathcal{D}$, namely compositions $m_{uvw}\colon \delta(u,v) \otimes \delta(v,w) \to \delta(u,w)$ and identities $j_u\colon I \to \delta(u,u)$, such that for all objects u, v, w in V the following diagrams commute. These diagrams express associativity of composition, and the left and right identity laws, explained below.

$$\begin{tikzcd}
(\delta(u,v) \otimes \delta(v,w)) \otimes \delta(w,x) \arrow[rr, "\alpha_{\delta(u,v)\delta(v,w)\delta(w,x)}"] \arrow[d, "m_{uvw} \otimes 1"] & & \delta(u,v) \otimes (\delta(v,w) \otimes \delta(w,x)) \arrow[d, "1 \otimes m_{vwx}"'] \\
\delta(u,w) \otimes \delta(w,x) \arrow[r, "m_{uwx}"] & \delta(u,x) & \delta(u,v) \otimes \delta(v,x) \arrow[l, "m_{uvx}"']
\end{tikzcd}$$

$$\begin{tikzcd}
I \otimes \delta(u,v) \arrow[r, "\lambda_{\delta(u,v)}"] \arrow[d, "j_u \otimes 1"] & \delta(u,v) \arrow[d, equal] & \delta(u,v) \otimes I \arrow[l, "\rho_{\delta(u,v)}"'] \arrow[d, "1 \otimes j_v"'] \\
\delta(u,u) \otimes \delta(u,v) \arrow[r, "m_{uuv}"] & \delta(u,v) & \delta(u,v) \otimes \delta(v,v) \arrow[l, "m_{uvv}"']
\end{tikzcd}$$

For $\mathcal{D} = \mathbf{2}$ the diagrams expressing the associativity and identity laws hold vacuously since $\mathcal{D}$ is an ordered set. Thus composition and identity become

$$\begin{aligned} m_{uvw}\colon &\quad u \leq v \ \wedge \ v \leq w \quad \to \quad u \leq w \\ j_u\colon &\quad u \leq u \end{aligned}$$

expressing respectively transitivity and reflexivity. Thus **2**-categories (categories enriched in **2**, as opposed to 2-categories which here are **Cat**-categories) are just preorders.

For $\mathcal{D} = \mathbf{R}^{\mathrm{op}}$, also an ordered set, we again may ignore the associativity and identity laws. Here we get

$$\begin{aligned} m_{uvw}\colon &\quad d(u,v) + d(v,w) \geq d(u,w) \\ j_u\colon &\quad 0 \geq d(u,u) \end{aligned}$$

For $\mathcal{D} = \mathbf{Cat}$ we obtain ordinary 2-categories, along the same lines as for $\mathcal{D} = \mathbf{Set}$ but with the addition of 2-cells between morphisms of the same homset.

4.2 $\mathcal{D}$-functors

Definition. A *$\mathcal{D}$-functor* $F\colon A \to B$ where A and B are $\mathcal{D}$-categories consists of a function $F\colon V_A \to V_B$ between object sets together with a family $F_{uv}\colon \delta_A(u,v) \to \delta_B(Fu, Fv)$ of morphisms between homobjects satisfying the following conditions stating that compositions and identities are preserved.

$$\begin{tikzcd}
\delta_A(u,v) \otimes \delta_A(v,w) \arrow[r, "m_{uvw}"] \arrow[d, "F_{uv} \otimes F_{vw}"'] & \delta_A(u,w) \arrow[d, "F_{uw}"'] \\
\delta_B(Fu,Fv) \otimes \delta_B(Fv,Fw) \arrow[r, "m_{Fu,Fv,Fw}"] & \delta_B(Fu,Fw)
\end{tikzcd}
\qquad
\begin{tikzcd}
I \arrow[r, "j_u"] \arrow[d, equal] & \delta_A(u,u) \arrow[d, "F_{uu}"'] \\
I \arrow[r, "j_{Fu}"] & \delta_B(Fu,Fu)
\end{tikzcd}$$

The elements of a $\mathcal{D}$-functor are depicted on the left of the following figure, and compose in the obvious way as shown on the right.

$$\begin{tikzcd}
u \arrow[rr, Rightarrow, "\delta_A(u,v)"] \arrow[d, mapsto, "F"'] & {} \arrow[d, "F_{uv}"'] & v \arrow[d, mapsto, "F"'] \\
Fu \arrow[rr, Rightarrow, "\delta_B(Fu,Fv)"] & {} & Fv
\end{tikzcd}
\qquad
\begin{tikzcd}
u \arrow[rr, Rightarrow, "\delta_A(u,v)"] \arrow[d, mapsto, "F"'] & {} \arrow[d, "F_{uv}"'] & v \arrow[d, mapsto, "F"'] \\
Fu \arrow[rr, Rightarrow, "\delta_B(Fu,Fv)"] \arrow[d, mapsto, "G"'] & {} \arrow[d, "G_{Fu,Fv}"'] & Fv \arrow[d, mapsto, "G"'] \\
GFu \arrow[rr, Rightarrow, "\delta_C(GFu,GFv)"] & {} & GFv
\end{tikzcd}$$

It can be seen that **2**-functors are monotone functions, **R**-functors and $\mathbf{R}^{\mathrm{op}}$-functors are expanding and contracting maps respectively, and **Set**-functors are ordinary functors. In our computational application a $\mathcal{D}$-functor is an event map which maintains all temporal constraints, though the result may have more constraints than the source.

4.3 Enrichment as a Function

We now define the object part of the functor $-!\colon \mathbf{COSMON} \to \mathbf{COSMON}$. Given a semiconcrete monoidal $\mathcal{D}$ define $\mathcal{D}!$ to be the category $\mathcal{D}$-**Cat** of small categories enriched in $\mathcal{D}$.

$\mathcal{D}!$ is symmetric monoidal as follows [EK66]. The tensor product of two $\mathcal{D}$-categories A and B is defined to have as objects the cartesian product $V_A \times V_B$, and homobjects

$$\delta((u,v),(u',v')) = \delta_A(u,u') \otimes_{\mathcal{D}} \delta_B(v,v').$$

The symmetry of $\mathcal{D}$ ensures that appropriate composition morphisms can be defined and that $\mathcal{D}!$ is symmetric to boot. The corresponding unit I is the one-object category with homobject $\delta_I(\cdot,\cdot) = I_{\mathcal{D}}$.

To make $\mathcal{D}!$ semiconcrete we define the forgetful functor $U_{\mathcal{D}!}: \mathcal{D}! \to \mathbf{Set}$ mapping each $\mathcal{D}$-category A to its underlying set V_A of objects and each $\mathcal{D}$-functor to its object map. $U_{\mathcal{D}!}$ is monoidal by the above definition of $\otimes$ for $\mathcal{D}!$. In our application U yields the event set $U(b)$ of a behavior b.

We summarize the above as follows.

Lemma 1 *If $\mathcal{D}$ is an object of* **COSMON** *then so is $\mathcal{D}!$.*

4.4 Enrichment as a Functor

We now define the action of ! on morphisms of **COSMON**.

Definition Given a monoidal functor $(F, \tau, t): \mathcal{D} \to \mathcal{E}$, define the functor $(F, \tau, t)! = (F!, \tau!, t!): \mathcal{D}! \to \mathcal{E}!$ as follows:

$F!$ takes a $\mathcal{D}$-category A to the $\mathcal{E}$-category $F!A = (V, \delta, m, j)$ defined by

$$\begin{array}{rl} V & = V_A \quad \text{(that is, } \; U_{\mathcal{D}!} = U_{\mathcal{D}'!}F!) \\ \delta(u,v) & = F(\delta_A(u,v)) \\ m_{uvw} & = F(m^A_{uvw})\tau_{\delta_A(u,v)\delta_A(v,w)} \\ j_u & = F(j^A_u)t \end{array}$$

$F!$ takes a $\mathcal{D}$-functor $G: A \to B$, to that $\mathcal{E}$-functor $F!G$ which has the same object function and takes $(F!G)_{uv}$ to $F(G_{uv})$. Thus $F!$ acts only on the distances within a behavior, and on the "distance comparison" part of a morphism of behaviors.

For $\mathcal{D}$-categories A and B and objects u and u' or A, and v and v' of B, define

$$(\tau!)_{(u,v)(u',v')} = \tau_{\delta_A(u,u'),\delta_B(v,v')}.$$

By the definition of $F!$, $F!I_{\mathcal{D}!}$ is an $\mathcal{E}$-category with a single object. Thus there is a unique possible object map for $t!: I_{\mathcal{E}!} \to F!I_{\mathcal{D}!}$. $t!$ on homobjects must be a map from $I_{\mathcal{E}} \to FI_{\mathcal{D}}$; define this map to be t.

Lemma 2 *If $(F, \tau, t): \mathcal{D} \to \mathcal{E}$ is a morphism of* **COSMON** *then so is $(F!, \tau!, t!): \mathcal{D}! \to \mathcal{E}!$.*

We omit the straightforward, though tedious, proof.

Finally, by checking that ! preserves composition and identities, we obtain the following.

Theorem 3 *! is an endofunctor on* **COSMON**.

4.5 Examples

(1) Section 2 contains a figure illustrating the example $(a;^3 b) \otimes (c;^2 d)$ of orthocurrence.

The triangle inequality requires the upper bounds on diagonals across squares and rectangles to be at most the sum of the bounds encountered along any path around the sides, e.g. 5 from ac to bd. This is exactly the effect obtained by defining tensor product of spaces (enriched categories) in terms of the addition (tensor product) of the underlying metric. Had we defined tensor product of spaces in terms of ordinary product in the metric (min), we obtain the next example, ordinary product of spaces.

(2) Consider the monoidal functor $F\colon \mathbf{R} \to \mathbf{2}$ which takes $-\infty$ to 0 and all else to 1. Application of $F!$ to each of the illustrated examples of Section 2 with vertex labels deleted to make them objects of $\mathbf{R}!$, produces the corresponding object of $\mathbf{2}!$, a poset, in effect by erasing the edge labels.

4.6 Continuity of !

We now show that ! enjoys certain continuity properties. These will be used in section 7 to show that all KL kinds are bicomplete and closed, which in turn ensures that PSL is well-defined.

Proposition 4 *If $\mathcal{D}$ has an initial object* 0 *and a natural isomorphism* $A \otimes 0 \cong 0$ *then $\mathcal{D}!$ has coproducts.*

The natural isomorphism $A \otimes 0 \cong 0$ always exists when $\mathcal{D}$ is closed.

A discrete category is a copower in **Cat** of 1, that is, the coproduct of a set of one-object categories, yielding the following similar result:

Proposition 5 *If $\mathcal{D}$ has an initial object* 0 *and* $0 \otimes A \cong 0 \cong A \otimes 0$, *then the forgetful functor* $U_{\mathcal{D}!}\colon \mathcal{D}! \to \mathbf{Set}$ *has a left adjoint* $D\colon \mathbf{Set} \to \mathcal{D}!$ *taking a set S to the corresponding* discrete $\mathcal{D}$-*category* ($\delta_{DS}(u,v) = I$ *if* $u = v$, *0 otherwise).*

The dual result to this, though not with a dual proof, is:

Proposition 6 *If $\mathcal{D}$ has a final object* 1, $U_{\mathcal{D}!}\colon \mathcal{D}! \to \mathbf{Set}$ *has a right adjoint* $E\colon \mathbf{Set} \to \mathcal{D}!$ *taking the set X to the corresponding chaotic (codiscrete) $\mathcal{D}$-category* ($\delta_{DS}(u,v) = 1$ *for all* u, v).

Meanwhile, for the case of general colimits we refer to a result of [BCSW83].

Theorem 7 *If $\mathcal{D}$ is cocomplete and $\mathcal{D}$ is closed then $\mathcal{D}!$ is cocomplete.*

There is also a similar but much easier result about limits. PSL needs limits for local product and hence local concatenation.

Theorem 8 *If $\mathcal{D}$ has limits of (small) type J, then so does $\mathcal{D}!$.*

If any one theorem could be considered at the heart of enriched categories it is the following. Kelly's proof ([Kel82] p.55, see also Lawvere [Law73] p.153) is dauntingly formal, to which we offer an informal counterpart.

Theorem 9 *If $\mathcal{D}$ is complete and closed then $\mathcal{D}!$ is closed.*

Proof: The objects of the $\mathcal{D}$-category C^B are of course just the $\mathcal{D}$-functors from B to C. What is less obvious is the appropriate homobject $\delta_{C^B}(U,V)$ abstracting the homset of natural transformations between any two $\mathcal{D}$-functors U,V of C^B. The natural *and unnatural* transformations are given by $\prod_{b\in B}\delta_C(Ub,Vb)$; we seek its subobject $\int_{b\in B}\delta_C(Ub,Vb)$ (an *end* [Mac71] p.219) consisting of just the natural ones.

Now for $\mathcal{D} = \mathbf{Set}$ such a transformation τ determines, for all b,b', two functions $\rho_\tau,\sigma_\tau\colon\delta_B(b,b')\to\delta_C(Ub,Vb')$, that is, two elements of $\delta_C(Ub,Vb')^{\delta_B(b,b')}$. Here ρ_τ takes $f\colon b\to b'$ to $V(f)\circ\tau_b$, while σ_τ takes the same f to $\tau_{b'}\circ U(f)$. But these are just the two sides of the square

$$\begin{tikzcd}
Ub \arrow[r, "\tau_b"] \arrow[d, "Uf"'] & Vb \arrow[d, "Vf"'] \\
Ub' \arrow[r, "\tau_{b'}"] & Vb'
\end{tikzcd}$$

which must be equal for τ to be natural. Encapsulating "for all b,b'" as a product over $b,b'\in B$ and generalizing to arbitrary $\mathcal{D}$ (formalized in Kelly's proof), we may therefore obtain the desired homobject $\delta_{C^B}(U,V)$ as the equalizer of

$$\begin{tikzcd}
\delta_{C^B}(U,V) \arrow[r] & \prod\limits_{b\in B}\delta_C(Ub,Vb) \arrow[r, shift left, "\rho"] \arrow[r, shift right, "\sigma"'] & \prod\limits_{b,b'\in B}\delta_C(Ub,Vb')^{\delta_B(b,b')}.
\end{tikzcd}$$

This construction required an exponential, two products, and an equalizer, for which it suffices that $\mathcal{D}$ be complete and closed. ∎

5 Comma Categories and the $\triangleright$ Operator

The ! operator creates categories of spaces with distances between pairs of points. We wish to interpret the points as events and the distances as temporal constraints on the events. To complete this interpretation we must have some way to say what type of event each point represents.

Given a labeling alphabet, a set Σ, we label the underlying set $U(p)$ of points of a space p with a function $\mu: U(p) \to \Sigma$. For p a poset this is the notion of pomset or partially ordered multiset as a labeled partial order.

Our framework can be simplified by dropping the assumption that Σ is a pure set and instead associating with the category supplying Σ a forgetful functor V that strips off any unwanted structure to reveal its underlying set $V(\Sigma)$. Our labeling function then becomes $\mu: U(p) \to V(\Sigma)$.

But this is what it means to be an object of the comma category $(U{\downarrow}V)$. Moreover the morphisms of this comma category from $\mu: U(p) \to V(\Sigma)$ to $\mu': U(p') \to V(\Sigma')$, namely pairs of maps $f: p \to p'$ and $t: \Sigma \to \Sigma'$ such that $V(t)\mu = \mu' U(f)$, turn out to be just what we need.

5.1 The Function $\triangleright$

Given categories $\mathcal{D}$, $\mathcal{E}$ in **COSMON** (i.e., both symmetric monoidal equipped with functors $U: \mathcal{D} \to$ **Set**, $V: \mathcal{E} \to$ **Set**, respectively), we define $\mathcal{D} \triangleright \mathcal{E}$ to be the comma category $U \downarrow V$. Recall that this is the category whose objects are triples $\langle x, \mu, u \rangle$ where $\mu: Ux \to Vu$, and whose morphisms are pairs $\langle f, g \rangle$ such that the following square commutes:

$$\begin{tikzcd}
Ux \arrow[r, "\mu"] \arrow[d, "Uf"'] & Vu \arrow[d, "Vg"'] \\
Ux' \arrow[r, "\mu'"] & Vu'.
\end{tikzcd}$$

$\mathcal{D} \triangleright \mathcal{E}$ can be made symmetric monoidal by defining the tensor product

$$\begin{tikzcd}
Ux \arrow[r, "\mu"] \arrow[d, "Uf"'] & Vu \arrow[d, "Vg"'] & & Uy \arrow[r, "\nu"] \arrow[d, "Uh"'] & Vv \arrow[d, "Vk"'] \\
Ux' \arrow[r, "\mu'"] & Vu' \arrow[r, phantom, "\otimes"] & {} & Uy' \arrow[r, "\nu'"] & Vv'
\end{tikzcd}$$

to be

$$\begin{tikzcd}
U(x \otimes y) \arrow[d, "U(f \otimes h)"] & Ux \times Uy \arrow[l, "\tau_U", "\cong"'] \arrow[r, "\mu \times \nu"] \arrow[d, "Uf \times Uh"'] & Vu \times Vv \arrow[r, "\tau_V", "\cong"'] \arrow[d, "Vg \times Vk"'] & V(u \otimes v) \arrow[d, "V(g \otimes k)"'] \\
U(x' \otimes y') & Ux' \times Uy' \arrow[l, "\tau_U", "\cong"'] \arrow[r, "\mu' \times \nu'"] & Vu \times Vv \arrow[r, "\tau_V", "\cong"'] & V(u \otimes v)
\end{tikzcd}$$

The unit object of $D \rhd \mathcal{E}$ is taken to be $I_{\mathcal{D} \rhd \mathcal{E}} = \langle I_{\mathcal{D}}, h, I_{\mathcal{E}} \rangle$, where $h = t_V t_U^{-1}$

$$\begin{tikzcd}
UI_{\mathcal{D}} & 1 \arrow[l, "t_U", "\cong"'] \arrow[r, "t_V", "\cong"'] & VI_{\mathcal{E}}
\end{tikzcd}$$

(In fact one may use these definitions for $\otimes$ and I in the case where V is only lax monoidal; U must still be strong monoidal).

Now we must show that this product is associative, symmetric, and $I_{\mathcal{D} \rhd \mathcal{E}}$ is in fact an identity. We outline briefly the definition of the associativity natural transformation $\alpha_{\mathcal{D} \rhd \mathcal{E}}$. (Other required natural transformations are defined similarly.)

We take $\alpha_{\mathcal{D} \rhd \mathcal{E}, \langle x,\mu,u \rangle \langle x',\mu',u' \rangle \langle x'',\mu'',u'' \rangle}$ to be $\langle \alpha_{\mathcal{D},xx'x''}, \alpha_{\mathcal{E},uu'u''} \rangle$. This is a well-defined morphism because $V(\alpha)(\mu \otimes (\mu' \otimes \mu''))$ is equal to $((\mu \otimes \mu') \otimes \mu'')U(\alpha)$. This requirement follows from properties of monoidal functors and natural transformations. The coherence conditions on $\alpha_{\mathcal{D} \rhd \mathcal{E}}$ are established similarly.

We make $\mathcal{D} \rhd \mathcal{E}$ semiconcrete by using the functor

$$\begin{tikzcd}
\mathcal{D} \rhd \mathcal{E} \arrow[r, "\pi_0"] & \mathcal{D} \arrow[r, "U"] & \mathbf{Set}
\end{tikzcd}$$

This makes the underlying sets of objects of $\mathcal{D} \rhd \mathcal{E}$ those of objects of $\mathcal{D}$. If the objects of $\mathcal{D}$ are spaces while those of $\mathcal{D} \rhd \mathcal{E}$ are labeled spaces, the underlying sets of the latter are the same as those of the former, i.e. the labels are ignored.

5.2 $\rhd$ as a Functor

To complete the description of the functor $\rhd$ we describe its action on morphisms (i.e., functors) F, G of **COSMON**. Recall that we impose the condition $U = U'F$ in the definition of the morphisms F of **COSMON** as well as the condition that F be strong monoidal. The functor

$F \rhd G$: **COSMON** $^2 \to$ **COSMON** acts thus:

$$\begin{tikzcd}
\mathcal{D} \arrow[r, "U"] \arrow[dd, "F"] & \mathbf{Set} \arrow[dd, equal] & \mathcal{E} \arrow[l, "V"'] \arrow[dd, "G"'] & & \mathcal{D} \rhd \mathcal{E} \arrow[r, "U\pi_0"] \arrow[dd, "F \rhd G"] & \mathbf{Set} \arrow[dd, equal] \\
& & \arrow[r, mapsto] & \rhd & & \\
\mathcal{D}' \arrow[r, "U'"] & \mathbf{Set} & \mathcal{E}' \arrow[l, "V'"'] & & \mathcal{D}' \rhd \mathcal{E}' \arrow[r, "U'\pi_0'"] & \mathbf{Set}
\end{tikzcd}$$

where π_0 projects onto $\mathcal{D}$ as before and $F \rhd G$ takes each morphism $\langle f, g\rangle$ of $\mathcal{D} \rhd \mathcal{E}$ to $\langle Ff, Gg\rangle = \langle U'Ff, V'Gg\rangle$.

$$\begin{tikzcd}
U'Fx \arrow[r, equal] \arrow[d, "U'Ff"'] & Ux \arrow[r, "\mu"] \arrow[d, "Uf"'] & Vu \arrow[r, equal] \arrow[d, "Vg"] & V'Gu \arrow[d, "V'Gg"] \\
U'Fx' \arrow[r, equal] & Ux' \arrow[r, "\mu'"] & Vu' \arrow[r, equal] & V'Gu'
\end{tikzcd}$$

We now need to verify that $F \rhd G$ is strong monoidal. This involves exhibiting a morphism $\tau_{F \,\rhd G}$: $(F \rhd G)\langle x, \mu, u\rangle \otimes (F \rhd G)\langle y, \nu, v\rangle \to (F \rhd G)(\langle x, \mu, u\rangle \otimes \langle y, \nu, v\rangle)$ of $\mathcal{D}' \rhd \mathcal{E}'$ for any pair $\langle x, \mu, u\rangle$, $\langle y, \nu, v\rangle$ both in $\mathcal{D} \rhd \mathcal{E}$, namely that given by pairing τ_F: $Fx \times Fy \to F(x \otimes y)$ with the corresponding τ_G. That this is indeed a morphism of $\mathcal{D}' \rhd \mathcal{E}'$ depends on the commutativity of the squares appearing in

$$\begin{tikzcd}
U'(Fx \otimes Fy) \arrow[dd, "U'\tau_F"', "\cong"] & U'Fx \times U'Fy \arrow[l, "\tau_{U'}"', "\cong"] \arrow[d, equal] & V'Gu \times V'Gv \arrow[r, "\tau_{V'}", "\cong"'] \arrow[d, equal] & V'(Gu \otimes Gv) \arrow[dd, "\cong"', "V\tau_G"] \\
& Ux \times Uy \arrow[d, "\tau_U"', "\cong"] \arrow[r, phantom, "\overset{\mu\times\nu}{\longrightarrow}"] & Vu \times Vv \arrow[d, "\cong"', "\tau_V"] & \\
U'F(x \otimes y) \arrow[r, equal] & U(x \otimes y) & V(u \otimes v) \arrow[r, equal] & V'G(u \otimes v)
\end{tikzcd}$$

which follows from the $\tau_U = (U'\tau_F)\tau_{U'}$ condition in the definition of **COSMON**. That the morphisms $\tau_{F \,\rhd G}$ constitute a natural transformation and that they satisfy the appropriate coherence conditions directly follows from the corresponding naturality and coherence conditions on τ_F and τ_G.

5.3 Continuity of $\triangleright$

We now wish to lift limits and colimits that exist in the component categories to comma categories defined from them. To do this we must deal with functors into comma categories.

The following lemma can be applied whenever functors into a comma category are to be defined. It states that defining a functor from a category J into $(U{\downarrow}V)$, where $U: \mathcal{A} \to \mathcal{C}$ and $V: \mathcal{B} \to \mathcal{C}$, is essentially equivalent to defining functors $F: J \to \mathcal{A}$ and $G: J \to \mathcal{B}$ and a natural transformation from UF to VG. Furthermore, natural transformations between such functors are essentially equivalent to a pair of natural transformations satisfying a certain commutativity condition.

For any functor $F: \mathcal{X} \to \mathcal{Y}$ denote by F^J the functor from the functor category $\mathcal{X}^J$ to $\mathcal{Y}^J$ defined by "left-composition with F". Then we have:

Lemma 10 $(U \downarrow V)^J \cong (U^J \downarrow V^J)$.

Outline of Proof: The comma category $U \downarrow V$ has the projections $\pi_0: U \downarrow V \to \mathcal{A}$ and $\pi_1: U \downarrow V \to \mathcal{B}$, and defines a natural transformation $\nu: U\pi_0 \to V\pi_1$. (In fact this is an alternative definition of the concept of a comma category.)

Hence, given a functor, $F: J \to U \downarrow V$, we can define functors $\pi_0 F: J \to \mathcal{A}$ and $\pi_1 F: J \to \mathcal{B}$, and a natural transformation $\nu \circ F: U\pi_0 F \to V\pi_1 F$. This defines an object of the comma category $(U^J \downarrow V^J)$. It is routine to extend this mapping to a functor $(U \downarrow V)^J \to (U^J \downarrow V^J)$ and show that it is an isomorphism of categories. ∎

We can use this lemma to prove a basic result about limits in comma categories. (Note: Although this result is purely categorical we are not aware of its having appeared in any textbook. A marginally weaker form was proven by Tarlecki in a rather different context [Tar85] and Mac Lane proved a special case as part of his proof of Freyd's Adjoint Functor Theorem [Mac71, page 123].)

Theorem 11 *Let $U: \mathcal{A} \to \mathcal{C}$ and $V: \mathcal{B} \to \mathcal{C}$ be functors, and J be a category. If $\mathcal{A}$ and $\mathcal{B}$ have all J-limits and V preserves J-limits then the comma category $(U \downarrow V)$ has all J-limits*

Proof: Let $F: J \to (U \downarrow V)$ be an J-diagram in $(U \downarrow V)$. Then by lemma 10, F can be considered as a triple, $\langle F_1, \mu, F_2 \rangle$ where

$$\begin{aligned} F_1&: J \to \mathcal{A} \\ F_2&: J \to \mathcal{B} \\ \mu&: UF_1 \to VF_2 \end{aligned}$$

Since $\mathcal{A}$ and $\mathcal{B}$ have J-limits, both F_1 and F_2 have limits. Denote their limiting cones by $\lambda^1: \lim F_1 \to F_1$ and $\lambda^2: \lim F_2 \to F_2$ respectively.

Since V preserves J-limits, $V(\lim F_2)$ is a limit of VF_2, with limiting cone $V\lambda^2$.

Now the composite natural transformation

$$U(\lim F_1) \to U\lambda^1 UF_1 \to \mu VF_2$$

is a cone from $U(\lim F_1)$ to VF_2, so by the universal property of limits there exists a unique arrow p such that the following diagram of functors and natural transformations commutes. (Note: in this and the following diagrams an object of C represents the constant functor to that object and an arrow of C represents the "constant" natural tranformation between such constant functors.)

$$\begin{tikzcd}
U(\lim F_1) \arrow[r, "U\lambda^1"] \arrow[d, "p"'] & UF_1 \arrow[d, "\mu"] \\
V(\lim F_2) \arrow[r, "V\lambda^2"] & VF_2
\end{tikzcd}$$

Now we claim that $\langle \lim F_1, p, \lim F_2 \rangle$ is the limit of F, and that the limiting cone is defined by the pair of natural transformations $\langle \lambda^1, \lambda^2 \rangle$.

For suppose we have an object, $\langle A, f, B \rangle$ and a cone $\kappa \colon \langle A, f, B \rangle \to F$. κ can be considered as a pair of natural transformations, $\kappa^1 \colon A \to F_1$ and $\kappa^2 \colon B \to F_2$. Since F_1 and F_2 have limits there must exist unique maps $a \colon A \to \lim F_1$ and $b \colon B \to \lim F_2$ which factor κ^1 and κ^2 through λ^1 and λ^2 respectively.

Consider the following diagram of natural transformations.

$$\begin{tikzcd}
UA \arrow[rrr, "f"] \arrow[dr, "Ua"] \arrow[dd, "U\kappa^1"'] & & & VB \arrow[dl, "Vb"'] \arrow[dd, "V\kappa^2"] \\
& U(\lim F_1) \arrow[r, "p"] \arrow[dl, "U\lambda^1"] & V(\lim F_2) \arrow[dr, "V\lambda^2"'] & \\
UF_1 \arrow[rrr, "\mu"] & & & VF_2
\end{tikzcd}$$

The outer square of this diagram commutes because κ is an arrow in the comma category. The two inner triangles commute by definition of a and b. The lower inner square commutes by definition of p. Now we wish to show that the upper inner square commutes. We do this by showing that the cone $(V\lambda^2).(Vb).f$ is identical to the cone $(V\lambda^2).p.(Ua)$. Since there must be a *unique* arrow which factors this cone through the limiting cone $V\lambda^2$, we can conclude that the arrows $(Vb).f$ and

$p.(Ua)$ must be identical. This is the required commutativity condition. But we have

$$\begin{array}{rcll} (V\lambda^2).(Vb).f &=& (V\kappa^2).f & \text{definition of } b \\ &=& \mu.(U\kappa^1) & \text{outer square commutes} \\ &=& \mu.(U\lambda^1).(Ua) & \text{definition of } a \\ &=& (V\lambda^2).p.(Ua) & \text{lower square commutes} \end{array}$$

This proves the required commutativity, which shows that $\langle a, b\rangle$ is an arrow in $U\downarrow V$ and that it factors κ. That $\langle a, b\rangle$ is the unique such arrow follows from the fact that a must factor κ^1 through the limiting cone λ^1, so a is the unique such arrow. Similarly, b must factor κ^2 through λ^2, and so is unique. ∎

Note that the proof gives an explicit construction of limits in a comma category when the conditions of the theorem are satisfied.

The same technique allows us to construct colimits:

Corollary 12 *If $\mathcal{A}$ and $\mathcal{B}$ have all J-colimits and U preserves J-colimits then $U\downarrow V$ has all J-colimits.*

Proof: For any functor $U: \mathcal{A} \to C$, let U^{op} be the corresponding functor $U^{\mathrm{op}}: \mathcal{A}^{\mathrm{op}} \to C^{\mathrm{op}}$. Observe that

$$(U\downarrow V)^{\mathrm{op}} = (V^{\mathrm{op}}\downarrow U^{\mathrm{op}}).$$

∎

With rather more work we may show that under appropriate conditions $U\downarrow V$ is closed.

Theorem 13 *Suppose that $\mathcal{A}$ and $\mathcal{B}$ are monoidal closed, and that $\mathcal{C}$ is monoidal. Suppose further that $F: \mathcal{A} \to \mathcal{C}$ is a strong monoidal functor, that $G: \mathcal{B} \to \mathcal{C}$ is (lax) monoidal, and that $\mathcal{A}$ has all pullbacks. If F admits a right adjoint, R, then $F\downarrow G$ is closed.*

Proof Outline: The proof is in three stages. Under the assumptions of the theorem we show:

1. R is monoidal and hence that RG is monoidal;
2. $F\downarrow G$ is isomorphic to $1_{\mathcal{A}}\downarrow RG$; and finally
3. For any monoidal functor $H: \mathcal{A} \to \mathcal{B}$, $1_{\mathcal{A}}\downarrow H$ is closed.

∎

We adopt the following notation. Suppose that $F \dashv R$. For any arrow $g: FA \to B$ denote the adjoint transpose by $g^{\sharp}: A \to RB$. Dually, denote the adjoint transpose of $f: A \to RB$ by $f^{\flat}: FA \to B$.

Observe that for appropriate arrows f, g and g' we have

$$\begin{aligned} (g'g)^{\sharp} &= (Rg')g^{\sharp} \\ (g(Ff))^{\sharp} &= g^{\sharp}f. \end{aligned}$$

The dual results, $(ff')^{\flat} = (f^{\flat})(Ff')$ and $((Rg')f)^{\flat} = g'f^{\flat}$ will also be useful.

We now proceed with the proofs of the three lemmas which prove the theorem.

Lemma 14 *Let $F: \mathcal{A} \to \mathcal{C}$ be strong monoidal and have a right adjoint, R. Then R is monoidal.*

Proof: Let $\phi: FA \otimes FB \to F(A \otimes B)$ and $\widehat{\phi}: I \to FI$ be the isomorphisms which make F monoidal.

Define $\psi: RX \otimes RY \to R(X \otimes Y)$ to be the adjoint transpose of the composite

$$F(RX \otimes RY) \xrightarrow{\phi^{-1}} FRX \otimes FRY \xrightarrow{\epsilon\otimes\epsilon} X \otimes Y;$$

and $\widehat{\psi}: I \to RI$ to be $(\widehat{\phi}^{-1})^{\sharp}$.

To show that ϕ and $\widehat{\phi}$ make R into a monoidal functor, we must show that they satisfy the coherence conditions. We give here the proof of the associativity condition. The proofs of the other two conditions are similar.

F is monoidal, so we know that the following commutes:

$$\begin{tikzcd}
(FRX \otimes FRY) \otimes FRZ \arrow[r, "\phi \otimes 1"] \arrow[d, "\alpha"'] & F(RX \otimes RY) \otimes FRZ \arrow[r, "\phi"] & F((RX \otimes RY) \otimes RZ) \arrow[d, "F\alpha"] \\
FRX \otimes (FRY \otimes FRZ) \arrow[r, "1 \otimes \phi"] & FRX \otimes F(RY \otimes RZ) \arrow[r, "\phi"] & F(RX \otimes (RY \otimes RZ))
\end{tikzcd}$$

Rearrange to obtain the equation

$$\alpha(\phi^{-1} \otimes 1)\phi^{-1} = (1 \otimes \phi^{-1})\phi^{-1}F\alpha.$$

Compose both sides of the equation with $\epsilon \otimes (\epsilon \otimes \epsilon)$. Then the left hand side of the new equation is:

$$\begin{aligned} &(\epsilon \otimes (\epsilon \otimes \epsilon))\alpha(\phi^{-1} \otimes 1)\phi^{-1} \\ &= \alpha(\psi^{\flat} \otimes \epsilon)\phi^{-1} \\ &= \alpha((\epsilon(F\psi)) \otimes \epsilon)\phi^{-1} \\ &= \alpha(\epsilon \otimes \epsilon)\phi^{-1}F(\psi \otimes 1) \\ &= \alpha\psi^{\flat}F(\psi \otimes 1) \end{aligned}$$

The right side of the new equation is

$$\begin{aligned}
&(\epsilon \otimes (\epsilon \otimes \epsilon))(1 \otimes \phi^{-1})\phi^{-1}(F\alpha) \\
&\quad = \quad (\epsilon \otimes \psi^{\flat})\phi^{-1}(F\alpha) \\
&\quad = \quad (\epsilon \otimes (\epsilon(F\psi)))\phi^{-1}(F\alpha) \\
&\quad = \quad (\epsilon \otimes \epsilon)\phi^{-1}F(1 \otimes \psi)(F\alpha) \\
&\quad = \quad \psi^{\flat}F(1 \otimes \psi)(F\alpha)
\end{aligned}$$

Apply $\sharp$ to both sides of the new equation and simplify using the properties of $\sharp$ to obtain the required condition

$$(R\alpha)\psi(\psi \otimes 1) = \psi(1 \otimes \psi)\alpha.$$

∎

Lemma 15 *Let $R\colon \mathcal{C} \to \mathcal{A}$ be a right adjoint of F, considered as monoidal functor as described in the previous lemma. Under the conditions of the theorem $F{\downarrow}G$ and $1_{\mathcal{A}}{\downarrow}RG$ are isomorphic monoidal categories.*

Proof: To each object $FA \xrightarrow{f} GB$ associate the corresponding object $A \xrightarrow{f^\sharp} RGB$. It is straightforward to show that this correspondence can be extended to an isomorphism of categories.

Now we show that the tensor product of the two categories coincide.

Let γ be the natural transformation $GB_1 \otimes GB_2 \to G(B_1 \otimes B_2)$. Let ϕ be the natural isomorphism $FA_1 \otimes FA_2 \to F(A_1 \otimes A_2)$. Denote the corresponding transformation for R by ψ, recalling that $\psi = ((\epsilon \otimes \epsilon)\phi^{-1})^\sharp$. Hence, the transformation for the composite functor RG is $(R\gamma)\psi$.

Consider two objects, $FA_1 \xrightarrow{f_1} GB_1$ and $FA_2 \xrightarrow{f_2} GB_2$. Applying $\sharp$ and then taking the tensor product in $1{\downarrow}RG$ results in the composite arrow

$$A_1 \otimes A_2 \xrightarrow{f_1^\sharp \otimes f_2^\sharp} RGB_1 \otimes RGB_2 \xrightarrow{(R\gamma)\psi} RG(B_1 \otimes B_2).$$

We want to show that this arrow is identical to the result of first taking tensor product in $F{\downarrow}G$ and then applying $\sharp$, namely

$$(\gamma(f_1 \otimes f_2)\phi^{-1})^\sharp.$$

But we have

$$\begin{aligned}
&(R\gamma)\psi(f_1^\sharp \otimes f_2^\sharp) && \\
&= (R\gamma)((\epsilon \otimes \epsilon)\phi^{-1})^\sharp(f_1^\sharp \otimes f_2^\sharp) && \text{definition of } \psi \\
&= (\gamma(\epsilon \otimes \epsilon)\phi^{-1}F(f_1^\sharp \otimes f_2^\sharp))^\sharp && \text{properties of } \sharp \\
&= (\gamma(\epsilon \otimes \epsilon)(F(f_1^\sharp) \otimes F(f_2^\sharp))\phi^{-1})^\sharp && \phi \text{ is natural} \\
&= (\gamma(\epsilon F(f_1^\sharp) \otimes \epsilon F(f_2^\sharp))\phi^{-1})^\sharp && \\
&= (\gamma(f_1 \otimes f_2)\phi^{-1})^\sharp, &&
\end{aligned}$$

as required. ∎

We now turn to the final lemma.

Lemma 16 *Let $\mathcal{A}$ and $\mathcal{B}$ be closed monoidal categories, and $H\colon\mathcal{A}\to\mathcal{B}$ be a monoidal functor. Suppose that $\mathcal{A}$ has pullbacks. Then the monoidal category $1_{\mathcal{A}}\!\downarrow\! H$ is closed.*

Proof: Denote the natural transformation $HB_1\otimes HB_2\to H(B_1\otimes B_2)$ by γ.

Let $\langle A_1,f_1,B_1\rangle$ and $\langle A_2,f_2,B_2\rangle$ be two objects of $1\!\downarrow\! H$. We show that tensor product has a right adjoint by constructing an object universal from the functor $\langle A_1,f_1,B_1\rangle\otimes -$ to $\langle A_2,f_2,B_2\rangle$.

Consider the pullback

$$\begin{tikzcd}
X \arrow[r, "\pi_2"] \arrow[dd, "\pi_1"'] & H[B_1,B_2] \arrow[d, "((H\epsilon)\gamma)^\sharp"] \\
& {[HB_1,HB_2]} \arrow[d, "{[f_1,1]}"] \\
{[A_1,A_2]} \arrow[r, "{[1,f_2]}"] & {[A_1,HB_2]}
\end{tikzcd}$$

We will show that the top line of this diagram is the required universal object. Firstly, we claim that the universal arrow to $\langle A_2,f_2,B_2\rangle$ is given by

$$\begin{tikzcd}
A_1\otimes X \arrow[r, "f_1\otimes\pi_2"] \arrow[d, "\pi_1^\flat"'] & HB_1\otimes H[B_1,B_2] \arrow[r, "\gamma"] & H(B_1\otimes[B_1,B_2]) \arrow[d, "H\epsilon"] \\
A_2 \arrow[rr, "f_2"] & & HB_2
\end{tikzcd}$$

To see that this square commutes, apply $\flat$ to both directions around the pullback square. It is immediate that $([1,f_2]\pi_1)^\flat$ equals $(f_2\pi_1^\flat)$. For the other direction, observe from standard properties of adjunctions that $[f_1,1]=(\epsilon(f_1\otimes 1))^\sharp$. Hence,

$$\begin{aligned}
&([f_1,1]((H\epsilon)\gamma)^\sharp\pi_2)^\flat \\
&\quad= [f_1,1]^\flat(1\otimes((H\epsilon)\gamma)^\sharp)(1\otimes\pi_2) \\
&\quad= \epsilon(1\otimes((H\epsilon)\gamma)^\sharp)(f_1\otimes\pi_2) \\
&\quad= (H\epsilon)\gamma(f_1\otimes\pi_2),
\end{aligned}$$

as required.

Suppose now that there exists a commuting square

$$\begin{tikzcd}
A_1 \otimes Y \arrow[r, "f_1 \otimes g"] \arrow[d, "p"'] & HB_1 \otimes HC \arrow[r, "\gamma"] & H(B_1 \otimes C) \arrow[d, "Hq"] \\
A_2 \arrow[rr, "f_2"] & & HB_2
\end{tikzcd}$$

$\mathcal{B}$ is closed, so q can be factored into $\epsilon(1 \otimes q^\sharp)$. By the naturality of γ, $(H(1 \otimes q^\sharp))\gamma$ is equal to $\gamma(1 \otimes Hq^\sharp)$, so we obtain

$$f_2 p = (H\epsilon)\gamma(f_1 \otimes (Hq^\sharp)g).$$

Applying $\sharp$ to both sides of this equation shows that the following square commutes.

$$\begin{tikzcd}
Y \arrow[r, "g"] \arrow[dd, "p^\sharp"'] & HC \arrow[r, "H(q^\sharp)"] & H[B_1, B_2] \arrow[d, "(f_1 \otimes 1)^\sharp"] \\
& & {[A_1, HB_1 \otimes H[B_1, B_2]} \arrow[d, "{[1, (H\epsilon)\gamma]}"] \\
{[A_1, A_2]} \arrow[rr, "{[1, f_2]}"] & & {[A_1, HB_2]}
\end{tikzcd}$$

Consider the right side of this square. Observe that

$$[1, (H\epsilon)\gamma](f_1 \otimes 1)^\sharp = ((H\epsilon)\gamma(f_1 \otimes 1))^\sharp.$$

Now consider the right side of the pullback square that defines X.

$$\begin{aligned}
[f_1, 1]((H\epsilon)\gamma)^\sharp &= (\epsilon(f_1 \otimes 1))^\sharp((H\epsilon)\gamma)^\sharp \\
&= (\epsilon(f_1 \otimes 1)(1 \otimes ((H\epsilon)\gamma)^\sharp))^\sharp \\
&= (\epsilon(1 \otimes ((H\epsilon)\gamma)^\sharp)(f_1 \otimes 1))^\sharp \\
&= ((H\epsilon)\gamma(f_1 \otimes 1))^\sharp
\end{aligned}$$

So we know that the right arrow of this square is the same as the right arrow of the pullback square defining X. But then there must exist a unique map $r: Y \to X$ such that $\pi_2 s = (Gq^\sharp)g$ and $\pi_1 s = p^\sharp$. The first equation is precisely the condition that $\langle s, q^\sharp \rangle$ be a well-defined map from $\langle Y, g, C \rangle$ to $\langle X, \pi_2, [B_1, B_2] \rangle$. The second shows that $\langle s, q^\sharp \rangle$ factors the original map $\langle p, q \rangle$ through the counit of the adjunction. Uniqueness follows from the unique factorization of q through ϵ and properties of pullbacks. ∎

A fine example related to this construction is provided by the partially ordered multisets motivating this work. A pomset $p = \langle P, \mu, \Sigma \rangle$ is just a poset P, labeled by a function $\mu: V(P) \to \Sigma$ for some set Σ. This makes it an object of $\mathbf{Pom} = \mathbf{Pos} \triangleright \mathbf{Set}$. A morphism of pomsets $f\colon \langle P, \mu, \Sigma\rangle \to \langle P', \mu', \Sigma'\rangle$ consists of a monotone event map $f\colon P \to P'$ on the underlying posets together with an alphabet map or *translation* $t\colon \Sigma \to \Sigma'$ from Σ to Σ', such that $t\mu = \mu' f$ (i.e., the event map and translation are consistent with respect to the labelings). We can view **Pom** as the full subcategory of $\mathbf{Prom} = 2! \triangleright \mathbf{Set}$, the latter generalizing the notion of pomsets by allowing P to be a preorder.

6 Abstract Specification of the Operations

Having established the categorical framework for our abstract treatment of time, we now define the operations of our algebras of behaviors. The sense in which time is abstract is that a fixed category $\mathcal{D}$ of spaces is assumed to be given as a parameter. The definitions may therefore refer to $\mathcal{D}$, but will not assume that $\mathcal{D}$ has any a priori structure such as labels on points or a metric between points. Each choice of $\mathcal{D}$ determines an algebra of behaviors, really a class of behaviors since we impose no cardinality bounds on the number of events in a behavior, other than the requirement that the events of a behavior form a set.

Most operations are of arity a small integer. On occasion however, e.g. when certain arguments are required to have the same domain or the same codomain, the arity will be a graph, meaning that the arguments will be organized as a functor from that graph. For example if the arity is the 3-vertex 2-edge graph of shape < (two edges pointing to the right) then the arguments are two morphisms with a common domain. This is made uniform by regarding each integer arity n as a discrete (edge-less) graph with vertices 1 through n.

Disjoint Concurrence, $\texttt{dconcur}(p, q)$, notation $p + q$. The *disjoint concurrence* of behaviors p and q performs both p and q *concurrently*, that is, unconstrained in time. It is defined as the coproduct $p + q$ in $\mathcal{D}$.

Disjoint Concatenation, $\texttt{dconcat}(p, d, q)$, notation $p;^d q$. The *disjoint concatenation* of behaviors p and q *with delay* d performs both p and q as in disjoint concurrence, but in the order p, then a delay $d\colon I + I \to d_0$, then q. It is defined by the pushout

$$\begin{tikzcd}
(I+I) \otimes (p \times q) \arrow[r, "\pi_p + \pi_q"] \arrow[d, "d \otimes 1"'] & p + q \arrow[d] \\
d_0 \otimes (p \times q) \arrow[r] & p;^d q
\end{tikzcd}$$

where the top morphism is the sum $\pi_p + \pi_q\colon p \times q + p \times q \to p + q$ of the two projections $\pi_p\colon p \times q \to p$ and $\pi_q\colon p \times q \to q$ from $p \times q$, and $(I + I) \otimes r \cong I \otimes r + I \otimes r \cong r + r$.

Concurrence, $\texttt{concur}(p \to k \leftarrow q)$, notation $p||_k q$, or $p||q$ when k is supplied by context. The concurrence of behaviors p and q *over* k performs both p and q concurrently, *coordinated by* k. It is defined as the coproduct of the morphisms $p \to k$ and $q \to k$ in $\mathcal{D}/k$, the comma or slice category of morphisms to k ("objects over" k), itself a morphism to k and thus retaining the coordination.

Concatenation, `concat`$((p, d, q) \to k)$, notation $p;^d_k q$. The *concatenation* of p with q performs both p and q concurrently, *coordinated by* k, but in the order p, then a delay d, then q. It is defined as for disjoint concatenation, with the pushout formed in $\mathcal{D}/k$.

Local disjoint concatenation, `ldconcat`$(d, p \to L \leftarrow q)$, notation $p\underline{;}^d_L q$. The *local disjoint concatenation* of behaviors p and q with delay d performs p and q concurrently except that events of p at a given location must be followed by a delay $d\colon (I + I) \to d_0$ before performing events of q at that location. It is defined analogously to disjoint concatenation via the pushout

$$\begin{tikzcd}
(I+I) \otimes (p \times_L q) \arrow[r, "(\pi_p + \pi_q) \circ L_="] \arrow[d, "d \otimes 1"'] & p + q \arrow[d] \\
d_0 \otimes (p \times_L q) \arrow[r] & p\underline{;}^d_L q
\end{tikzcd}$$

where $L_=\colon p \times_L q \to p \times q$ is the standard embedding, namely the equalizer $L_p \circ \pi_p$ and $L_q \circ \pi_q$, and the rest is as for disjoint concatenation.

Local concatenation, $p\underline{;}^d_{kL} q$, is the appropriate combination of concatenation and local disjoint concatenation.

Orthocurrence, `orthocur`(p, q), notation $p \otimes q$. The *orthocurrence* of behaviors p and q consists of the flow of p and q past each other. It is defined as their tensor product $p \otimes q$.

Unit, `unit`, notation I. The *unit* is a one-event behavior that flows by unnoticed. It is defined as the unit I of $\mathcal{D}$, being the unit for orthocurrence.

Observation, `observe`(q, r), notation r^q. The *observation* of behavior q by behavior r is the result of observing every stage of every event of q, where the stages of an event of q are defined as the events of r. It is defined by the adjunction $p \otimes q \to r \cong p \to r^q$, that is, $-^q$ is right adjoint to $- \otimes q$.

Product, `product`(p, q), notation $p \times q$. The *product* of behaviors p and q consists of all pairs (a, b) of constituents a of p and b of q, with all the structure of p and q preserved coordinatewise. It is defined as their ordinary (categorical) product $p \times q$.

Local product, `lproduct`$(p \to L \leftarrow q)$, notation $p \times_L q$. The *local product* of behaviors p and q consists of all pairs (a, b) such that a and b are L-colocated (conceptually, at the same location $l \in L$). It is defined as the pullback

$$\begin{tikzcd}
p \times_L q \arrow[r] \arrow[d] & q \arrow[d] \\
p \arrow[r] & L
\end{tikzcd}$$

7 The Kind Language KL

The kind language KL consists of terms naming certain kinds in **COSMON**. A KL term is either a constant denoting one of the ground kinds $\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{3}'$, $\mathbf{R}$, $\mathbf{R}^{\mathrm{op}}$, and $\mathbf{SR}$, or a compound term $\mathcal{D}!$ or $\mathcal{D} \rhd \mathcal{E}$ where $\mathcal{D}$ and $\mathcal{E}$ are KL terms.

We make each ground kind semiconcrete by setting its underlying set functor U to 0 at 0 and 1 elsewhere. The one exception is $\mathbf{1}$, where $U(0) = 1$, since $0 = 1$ in $\mathbf{1}$.

Some terms in this language are neither useful nor well-behaved, for example $\mathbf{2} \rhd \mathbf{1}!$, which denotes a category that is not closed. A *safe* term is defined by induction to be either a ground term, a term $\mathcal{D}!$ where $\mathcal{D}$ is safe, or a term $\mathcal{D}! \rhd \mathcal{E}$ where $\mathcal{D}$ and $\mathcal{E}$ are safe. We then take a KL *kind* to be the denotation in **COSMON** of a safe KL term.

Theorem 17 *Every safe KL kind is closed and bicomplete.*

Proof: This follows by induction on the structure of terms of KL. For the base case, ground kinds are closed and cocomplete by definition. For compound safe terms the theorem follows by induction from Theorems 7, 8, 9, and 11, Corollary 12, Theorem 13, and the fact that the forgetful functors of the basic kinds and those constructed by ! and $\rhd$ have the necessary continuity properties and adjoints. ∎

KL kinds include $\mathbf{1}!$ = sets, $\mathbf{1} \rhd \mathbf{1}!$ = pointed sets, $\mathbf{2}!$ = preordered sets, $\mathbf{1}! \rhd \mathbf{1}!$ = multisets, $\mathbf{2}! \rhd \mathbf{1}!$ = ordered multisets, $\mathbf{1}!!$ = categories, $\mathbf{2}!!$ = order-enriched categories, $\mathbf{1}!!!$ = 2-categories, $\mathbf{3}!$ = causal spaces, $\mathbf{3}'! \rhd \mathbf{1}!$ = prossets, and $\mathbf{R}^{\mathrm{op}}!$ = premetric spaces. It should be straightforward to verify these correspondences from what has been said thus far about the ground kinds and the operations.

This language provides a succinct system of names for a usefully broad range of categories of datatypes. It suffices to name such a category $\mathcal{D}$ to be assured of the presence of all operations definable by universal constructions (limits, colimits, adjunctions, etc.) from the monoidal category structure of $\mathcal{D}$.

8 Conclusion

8.1 Directions For Further Work

We mention briefly some projects we have in mind.

Working over **Cat**. Our techniques would appear to extend straightforwardly to monoidal 2-categories, where the forgetful functors are not to **Set** but to **Cat**. This turns the labeling function from the underlying set of a behavior to (the underlying set of) an alphabet into a labeling *functor*.

Its functoriality can be put to good use, as in conveniently naming the closed category of order ideals.

Extensions to KL. Currently, methods for specifying subkinds of dynamic kinds are somewhat ad hoc, e.g., posets are acyclic **2**-categories. It would be useful to have operators that would produce subkinds for a wide variety of $\mathcal{D}$, such as a single approach to producing partial orders (as opposed to preorders) and say symmetric metric spaces.

It would also be useful to have operators for constructing new metrics. Indeed the present supply of metrics has been constructed ad hoc. Some uniform way of constructing them would be useful.

Such operators would be additions to the kind language KL. The only operations we have admitted thus far to KL are ! and $\triangleright$. We do not however intend by any means that these be all. To fit in with ! and $\triangleright$ however these operations would need similar continuity properties.

Stochastic delay. Another possible choice for $\mathcal{D}$ is that of a category of probability distributions. In other words for each pair of events u, v we specify a probability density function $\rho_{uv}\colon \mathbf{R} \to [0,1]$ for the associated delay. Consecutive delays would be combined using convolution

$$\rho_{uv} \otimes \rho_{vw}(y) = \int_{-\infty}^{\infty} \rho_{uv}(x)\rho_{vw}(y-x)dxdy$$

The main problems include finding a suitable ordering on distributions and dealing with the fact that the various distributions on delays are not independent. There is also the question of the proper treatment of distributions which are not well behaved and continuous (e.g., the Dirac delta function).

8.2 Summary

We have described a model of concurrency consisting of two orthogonal parts, one dealing with kinds of behaviors, the other with the behaviors themselves.

Kinds are taken to be semiconcrete monoidal categories, objects of **COSMON**. A "kind language" KL is provided for naming a few basic kinds and combining them with operations ! and $\triangleright$ to form compound kinds. The ! operation turns a metric into a category of spaces with that metric. The $\triangleright$ operation combines two kinds by using one as a source of structures and the other as a source of alphabets with which those structures may be labeled. Kinds namable in this way are called KL kinds.

Behaviors are taken to be objects of a given kind $\mathcal{D}$ an object of **COSMON**. A "behavior language" PSL is provided for naming basic behaviors and combining them with a useful library of operations suitable for concurrent programming. Although typical behaviors contain much structure, including a metric and a labeling function, this is *only* typical and not required. In general nothing is assumed about the "internal" structure of a behavior, which may well be just a simple atomic value. We are however given operations for assembling behaviors to form larger behaviors, namely limits, colimits, and a tensor product. These operations belong to the language PSL.

PSL is thus a true algebra of behaviors. Even though the PSL operations are defined uniformly across the spectrum of KL kinds, their action on complex KL kinds would appear to be as useful as if PSL had been defined to operate explicitly on spaces with metrics and labeling functions. Yet the meaning of PSL is defined no differently for "atomic" behaviors than for behaviors with complex structures. Thus PSL acts as an algebra of behaviors in assuming no structure within its elements yet being able to recreate that internal structure "from outside."

The division of labor between KL and PSL is completely orthogonal. "Motion" via functors of **COSMON**, from which KL operations are drawn, has no influence on the point sets of spaces and individual alphabets making up any given object $\mathcal{D}$ of **COSMON**. This independence finds its expression in the identity $U = U'F$ relating underlying sets, which says that underlying sets of points and functions between them pass through F completely unscathed. Conversely, motion via functors of $\mathcal{D}$, from which PSL operations are drawn, takes place relative to a fixed $\mathcal{D}$ and hence has no influence on the metrics and alphabet kinds making up **COSMON**.

That PSL operates on the "individuals"—objects and arrows—of a fixed $\mathcal{D}$ gives it the character of a first order language. Since formation of $\mathcal{D}$-functor spaces is an operation of PSL, the proper analogy is with ZF as a first order theory, where sets of individuals are also individuals. That is, ZF and PSL are both "internally closed," both by virtue of working in a closed universe.

Unlike ZF however, which operates in the cartesian closed universe of sets, PSL operates in a closed universe that is not presumed to be cartesian closed, whence the need to distinguish between tensor product $\otimes$ and ordinary product $\times$ in PSL.

This distinction arises in a natural and inevitable way from an elementary and familiar observation: *the temporal accumulation operator need not be product.* That it is product (conjunction) in the two-valued metric logic underlying ordered time is not a necessary facet of the logic of time.

A recent noncartesian logic is Girard's linear logic [Gir87]. Like PSL, linear logic distinguishes ordinary and tensor product. Like Boolean logic but unlike PSL, Girard's linear logic is self-dual, giving rise by de Morgan's law to two binary operations dual to the two products. This prompts the following question.

Why should self-duality survive the diverging of the two products?

To bring the concepts of PSL together we have drawn heavily on some basic techniques that have evolved in category theory within the past three decades, most notably those of closed categories and enriched categories. These techniques in the dry context of a mathematics text can seem dauntingly abstract. But just as there are good and bad cholesterols so it is with abstractions: the bad daunt and the good clarify. In adapting these techniques to computing practice we have tried to leave the good abstractions intact, namely those that simplified matters or seemed to bear on computation.

One defect of our account is that it has put relatively little emphasis on the logical character of PSL. The adjunctions and compositions pervading our framework contain all the essential elements of a modern logic, yet we have not made this logical character as explicit as we might. This may just be a reflection of the algebraic perspective which nurtured this work. We hope that the proper balance of algebra and logic here will become clear in due course.

Acknowledgments

We are indebted to F.W. Lawvere both for his paper [Law73] clarifying the topic of enriched categories and for much helpful advice. We also appreciate the help we have received from the Sydney Category Seminar, especially A.J. Power and R.F.C. Walters. The paper was typeset using D. Knuth's TeX, L. Lamport's LaTeX macros, and P. Taylor's diagram macros.

References

[AHU74] A.V. Aho, J. E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Mass, 1974.

[AM75] M. Arbib and E. Manes. *Arrows, Structures, and Functors: The Categorical Imperative.* Academic Press, 1975.

[BCSW83] R. Betti, A. Carboni, R. Street, and R. Walters. Variation through enrichment. *Journal of Pure and Applied Algebra*, 29:109–127, 1983.

[Bir37] G. Birkhoff. An extended arithmetic. *Duke Mathematical Journal*, 3(2), June 1937.

[Bir42] G. Birkhoff. Generalized arithmetic. *Duke Mathematical Journal*, 9(2), June 1942.

[CCMP89] R.T Casley, R.F. Crew, J. Meseguer, and V.R. Pratt. Temporal structures. In *Proc. Conf. on Category Theory and Computer Science, LNCS*, Manchester, September 1989. Springer-Verlag.

[Con71] J.H. Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, London, 1971.

[EK66] Samuel Eilenberg and G. Max Kelly. Closed categories. In S. Eilenberg, D. K. Harrison, S. MacLane, and H. Röhrl, editors, *Proceedings of the Conference on Categorical Algebra, La Jolla, 1965*, pages 421–562. Springer-Verlag, 1966.

[Flo62] R.W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

[Gai89] H. Gaifman. Modeling concurrency by partial orders and nonlinear transition systems. In *Proc. REX School/Workshop on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, pages 467–488, Noordwijkerhout, The Netherlands, 1989. Springer-Verlag.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[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.

[Kel82] G.M. Kelly. *Basic Concepts of Enriched Category Theory: London Math. Soc. Lecture Notes.* 64. Cambridge University Press, 1982.

[Koz80] D. Kozen. A representation theorem for models of *-free PDL. In *Proc. 7th Colloq. on Automata, Languages, and Programming*, pages 351–362, July 1980.

[Koz81] D. Kozen. On induction vs. *-continuity. In D. Kozen, editor, *Proc. Workshop on Logics of Programs 1981, LNCS 131*, pages 167–176. Spring-Verlag, 1981.

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

[Lew90] H. Lewis. A logic of concrete time intervals. In *Proc. 5th Annual IEEE Symp. on Logic in Computer Science*, Philadelphia, July 1990.

[Mac71] S. Mac Lane. *Categories for the Working Mathematician.* Springer-Verlag, 1971.

[Pra84] V.R. Pratt. The pomset model of parallel processes: Unifying the temporal and the spatial. In *Proc. CMU/SERC Workshop on Analysis of Concurrency, LNCS 197*, Pittsburgh, 1984. Springer-Verlag.

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

[Pra89] V.R. Pratt. Enriched categories and the Floyd-Warshall connection. In *Proc. First International Conference on Algebraic Methods and Specification Techniques*, pages 177–180, Iowa City, May 1989.

[RF68] P. Robert and J. Ferland. Généralisation de l'algorithme de warshall. *Revue francaise d'Informatique et de Recherche Operationnelle*, 2(7):13–25, 1968.

[Roy59] B. Roy. Transitivité et connexité. *Comptes Rendues Acad. Sci.*, 249:216–218, 1959.

[Tar85] Andrzej Tarlecki. Bits and pieces of the theory of institutions. In *Category Theory and Computer Programming*, Lecture Notes in Computer Science 240, pages 334–363, Guildford, UK, 1985. Springer-Verlag.

[Vic89] S. Vickers. *Topology via Logic.* Cambridge University Press, 1989.

[War62] S. Warshall. A theorem on Boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.