Chu spaces:
Complementarity and Uncertainty
in
Rational Mechanics

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

July 3, 1994

1 Introduction to Chu spaces

1.1 Basic notions

A *Boolean Chu space* $\mathcal{A} = (X, \models, A)$ consists of two sets X and A and a binary relation $\models \ \subseteq X \times A$ from X to A. We call the elements $x, y, \ldots$ of X *states* or *opens*, and the elements $a, b, \ldots$ of A *points*, *propositions*, or *events*. We read $x \models a$ as the Boolean-valued assertion "state x satisfies point (proposition, event) a." Viewed as an event, a is understood as the proposition "event a has happened."

A Chu space can be depicted naturally as a matrix. Figure 1 gives some illustrative examples that we shall refer to in the sequel.

We define $\mathrm{row}_A(x) = \{a \mid x \models a\}$, the set of those column indices a containing a 1 at row x, and dually $\mathrm{col}_A(a) = \{x \mid x \models a\}$. When row_A is an injective function (no repeated rows) we call $\mathcal{A}$ *extensional*, and when col_A is injective (no repeated columns) we call $\mathcal{A}$ T_0 by analogy with topological spaces. When row_A is the identity function on X, X must be a subset of 2^A; we call such a Chu space *normal*, and write it as simply (X, A), $\models$ then being inferrable as the converse $\in\breve{}$ of set membership. A normal space is automatically extensional but need not be T_0. The Chu spaces of Figure 1 are all extensional and T_0 but

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

$$\begin{array}{c|ccc} & a & b & c \\ s & 0 & 0 & 0 \\ t & 0 & 0 & 1 \\ u & 0 & 1 & 0 \\ v & 0 & 1 & 1 \\ w & 1 & 0 & 0 \\ x & 1 & 0 & 1 \\ y & 1 & 1 & 0 \\ z & 1 & 1 & 1 \end{array}$$

(a)

$$\begin{array}{c|ccc} & a & b & c \\ s & 0 & 0 & 0 \\ w & 1 & 0 & 0 \\ x & 1 & 0 & 1 \\ y & 1 & 1 & 0 \\ z & 1 & 1 & 1 \end{array}$$

(b)

$$\begin{array}{c|ccc} & a & b & c \\ s & 0 & 0 & 0 \\ w & 1 & 0 & 0 \\ y & 1 & 1 & 0 \\ z & 1 & 1 & 1 \end{array}$$

(c)

$$\begin{array}{c|ccc} & a & b & c \\ s & 0 & 0 & 0 \\ x & 1 & 0 & 1 \\ y & 1 & 1 & 0 \\ z & 1 & 1 & 1 \end{array}$$

(d)

$$\begin{array}{c|cccc} & a & b & c & d \\ w & 0 & 0 & 0 & 0 \\ x & 0 & 0 & 1 & 1 \\ y & 0 & 1 & 0 & 1 \\ z & 0 & 1 & 1 & 0 \end{array}$$

(e)

$$\begin{array}{c|cccc} & a & b & c & d \\ x & 0 & 0 & 1 & 1 \\ y & 0 & 1 & 0 & 1 \end{array}$$

(f)

Figure 1. Representative Chu spaces presented as matrices.

not normal unless v is identified with $\{b, c\}$ etc.

The *dual* $\mathcal{A}^{\perp}$ of a Chu space $\mathcal{A} = (X, \models, A)$ is simply its transpose $(A, \models^{\smile}, X)$ as a matrix, with points turned into states and vice versa. We denote the converse of a binary relation $R \subseteq X \times Y$ as $R^{\smile}$, defined as the subset $\{(y, x) \mid xRy\}$ of $Y \times X$, The dual of a normal space is T_0 but never normal (assuming set membership is well-founded) even if extensional; dualizing a second time however restores it to a normal space since dualizing merely interchanges the two index sets, whatever they are.

Boolean Chu spaces are closely related to S. Vickers' notion of a *topological system* [Vic89, p.52], the definition of which Vickers begins thus.

> Let A be a frame; we call its elements *opens*. Let X be a set; we call its elements *points*. Finally, let $\models$ be a subset of $X \times A$. If $(x, a) \in \models$ we write $x \models a$ and say x *satisfies* a.

With Chu spaces, the two differences from a topological system are that A is not a frame but merely a set, and that points and opens are interchanged. The first difference is essential and will be seen to make the category of Chu spaces not only a substantial and very useful extension of that of topological systems but a self-dual one at that. The second difference is a mathematically inessential matter of interpretation: whereas the propositions of a topological system transform contravariantly, those of a Chu space transform covariantly. While there is no formal reason to prefer one orientation over the other, our choice for Chu spaces will be seen to be a natural consequence of thinking of states as *possible* worlds (or paintings of individuals) accumulating disjunctively and propositions as *necessary* constraints (or obstacles, or individuals) accumulating conjunctively.

A *Chu space over* a set K is a triple $(X, \models, A)$ where $\models: X \times A \to K$ is a K-valued function of states and points. Chu spaces over $K = \mathbf{2} = \{0,1\}$ are thus the Boolean Chu spaces defined above. We generalize row_A by replacing "subset of" by "function to K;" thus $\mathrm{row}_A(x) : A \to K$ is the function whose value at $a \in A$ is $x \models a$, and dually for col_A. For $K = \mathbf{2}$ this amounts to replacing subsets of A by their characteristic functions (functions from A to $\mathbf{2}$). We may understand $\mathrm{row}_A(x)$ as a painting of the points of the space $\mathcal{A}$ with colors from the "palette" K. It is not so natural to view $\mathrm{col}_A(a)$ as a painting of states since in the real world we perceive all the points at once but the states only one at a time. This perspective is relative however; it is appropriate for programs when they are running, but the perspective is reversed when writing the program: its states all exist at the same time in the text of the program. More generally, in game-oriented views of situations, certain contexts demand seeing things from one player's viewpoint, others require taking the opposing side, and yet anothers call for the neutrality of an umpire.

Chu transforms. For any fixed K, a *Chu transform* $(f, g) : \mathcal{A} \to \mathcal{B}$ between Chu spaces $\mathcal{A} = (X, \models_A, A)$, $\mathcal{B} = (Y, \models_B, B)$ over K consists of two functions $f : A \to B$ and $g : Y \to X$ satisfying the following *adjointness condition*: for all $a \in A$ and $y \in Y$, $g(y) \models_A a \;=\; y \models_B f(a)$. Chu transforms compose as $(f', g')(f, g) = (f'f, gg')$; this composition is evidently associative and has $(1_A, 1_X)$ as its identity at each Chu space $(X, \models, A)$. Chu spaces over K thereby form a category which we denote $\mathbf{Chu}_K$; we abbreviate $\mathbf{Chu}_2$ to $\mathbf{Chu}$.

The adjointness condition may be rephrased in terms of rows, namely $\mathrm{row}_A(g(y)) = \mathrm{row}_B(y) \circ f$, verified by the calculation $\mathrm{row}_A(g(y))(a) = g(y) \models_A a = y \models_B f(a) = (\mathrm{row}_B(y) \circ f)(a)$. That is, every row of B when composed with f must be some row of A, with g a function selecting a suitable row index. It follows that *when the source of a Chu transform (f, g) is extensional, g is determined by f.* We will often restrict attention to extensional Chu spaces, where it suffices to specify only the f component of a Chu transform. In this respect Chu transforms behave like homomorphisms of relational structures, consisting of just a function between their carriers meeting a certain condition. The dual form of this rephrasing is $\mathrm{col}_A \circ g = \mathrm{col}_B(f(a))$, rarely used; it shows that when the target of a Chu transform is T_0, f is determined by g.

Duality. Transposition of the objects of $\mathbf{Chu}_K$ extends in the obvious way to its morphisms, sending (f, g) to (g, f). This makes transposition a functor from $\mathbf{Chu}_K$ to $\mathbf{Chu}_K{}^{\mathrm{op}}$. Transposition is of course an involution, $\mathcal{A}^{\perp\perp} = \mathcal{A}$, and hence an isomorphism of $\mathbf{Chu}_K$ and $\mathbf{Chu}_K{}^{\mathrm{op}}$. This makes transposition a *duality*[1] from $\mathbf{Chu}_K$ to itself, making $\mathbf{Chu}_K$ a *self-dual* category.

[1] A duality is a contravariant equivalence between two categories C and D, meaning an

Carrier and cocarrier. Whereas an algebra or relational structure has only the one underlying set or *carrier*, a Chu space has both a carrier A and a *cocarrier* X. The cardinality of a Chu space is that of its carrier as usual; we refer to the cardinality of the cocarrier as the cocardinality of $\mathcal{A}$. We define the underlying-set functor $U_K : \mathbf{Chu}_K \to \mathbf{Set}$ as $U_K(X, \models, A) = A$, $U_K(f, g) = f$, and the underlying-antiset functor $V_K : \mathbf{Chu}_K \to \mathbf{Set}^{\mathrm{op}}$ as $V_K(X, \models, A) = X$, $V_K(f, g) = g$. We may understand $\mathbf{Set}^{\mathrm{op}}$ as the category of sets X and *antifunctions*, defined as binary relations $R \subseteq X \times Y$ whose converse $R^{\smile} \subseteq Y \times X$ is a function $g : Y \to X$. We shall sometimes refer to sets that transform by antifunctions as *menus* to emphasize their disjunctive quality; thus a Kripke structure is a menu of possible worlds.

Seen in this light, a pair $(f, g) : (X, \models_A, A) \to (Y, \models_B, B)$ of functions is a Chu transform just when the following square whose edges are binary relations, and which compose standardly as such, commutes.

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "\models_A{}^{\smile}"'] & B \arrow[d, "\models_B{}^{\smile}"] \\
X \arrow[r, "g^{\smile}"] & Y
\end{tikzcd}$$

Let $\mathbf{Rel}$ denote the category of sets and their binary relations, and let $\mathbf{Rel}^2$ denote its "arrow category," whose objects are the morphisms of $\mathbf{Rel}$ and whose morphisms are the commuting squares of $\mathbf{Rel}$ (e.g. the above diagram). Then $\mathbf{Chu}$ may be understood as the subcategory (not full) of $\mathbf{Rel}^2$ such that each morphism (square) has a function for its upper edge and an antifunction for its lower, as in the diagram.

We regard X as a *disjunctive* set of alternatives ("possible worlds"), and A as a *conjunctive* set of entities all simultaneously present. This is the distinction a player of a game draws between her own moves, which form a menu of *opportunities* one of which she chooses, and those of her opponent, forming a set of *risks* all of which she must guard against.

We shall also take this as an abstraction of mind-body duality, with a set understood as a pure body, a menu as a pure mind, and a Chu space as a binary relation from a mind to a body.

Whereas functions may *identify* (when not injective) and *adjoin* (when not surjective), the corresponding operations performed by antifunctions are respectively *copy* and *delete*. Viewed as editing operations between sets of memory locations, the latter come more naturally to computer scientists than the former. In identifying two cells, how does one combine their contents? And what should a newly adjoined cell contain? No such awkward questions arise when copying and deleting cells. Exercise: noting that "contents-of" is a function from cells to values, relate all this to the op in Hom : $\mathbf{Set}^{\mathrm{op}} \times \mathbf{Set} \to \mathbf{Set}$.

equivalence between C and D^{op}. The duality here is the more intimate relationship of actual isomorphism of categories.

An attractive feature of the category **Set** is that it is both complete and cocomplete (has all limits and colimits). Hence so does $\mathbf{Set}^{\mathrm{op}}$, making it equally attractive. One might think to work in the larger category **Rel**, a self-dual category offering not only both functions and antifunctions in the one category but all binary relations. However although **Rel** has all products and coproducts it does not have other limits or colimits (equalizers, coequalizers, pullbacks, pushouts, etc.).

Chu as a self-dual extension of both **Set** and $\mathbf{Set}^{\mathrm{op}}$ (comonadic in the former, the latter monadic in **Chu**) inherits their bicompleteness. In choosing a self-dual category to do mathematics in, this constitutes a significant advantage for **Chu** over either **Rel** or $\mathbf{Rel}^2$. It is our thesis that Tarski's vision of mathematics founded on binary relations would be more effective if founded on Chu spaces.

We will show later that the category of κ-ary relational structures and their homomorphisms (standardly understood) embeds fully and concretely in $\mathbf{Chu}_{2^\kappa}$. This allows us to think of $\mathbf{Chu}_K$ as a combined logic-and-algebra of relations of arity up to κ. In particular Boolean Chu spaces correspond in this way to the monadic predicate calculus. This makes Chu spaces a rich mathematical playground.

There are only 2^n unary relations on an n-element set, but 2^{2^n} Boolean operations and hence extensional Chu spaces. This indicates that the lower bound of the previous paragraph on the utility of Chu spaces is a very conservative one: most Chu spaces over 2^κ do not correspond to κ-ary relational structures. Yet inspection of T_0 extensional Chu spaces of cardinality up to 3 reveals all of them to be useful for something. (A two-day computer search found 2,4,8,64,3828,37320288 isomorphism classes of extensional T_0 Boolean Chu spaces of cardinality respectively 0 through 5.)

The next two subsections present Boolean Chu spaces from respectively a logical and an algebraic viewpoint.

1.2 Boolean Chu spaces as Boolean operations

The following logical view of *extensional* Chu spaces allows us to identify them with their properties expressed as a theory. An important application is to the realization, or full concrete embedding, of categories of various order-theoretic kinds—posets, topological spaces, semilattices, distributive lattices, etc.—in the single self-dual category **Chu**.

The Chu space $\mathcal{A} = (X, \models, A)$ may be understood as the Boolean operation (abstract formula) φ_A in set A of propositional variables whose satisfying assignments are those determined by X: the operation is true whenever the variables are assigned truth values according to some row of $\mathcal{A}$, and false for all other truth assignments. The Chu spaces of Figure 1 determine in this way the respective operations *true* (in variable set $A = \{a, b, c\}$), $b \vee c \;\rightarrow\; a$, $(c \rightarrow b) \wedge (b \rightarrow a)$,

$a \equiv (b \vee c)$, $\neg a \wedge (d \equiv (b \oplus c))$, and $\neg a \wedge (b \oplus c) \wedge d$ (where $a \oplus b = \neg(a \equiv b)$, i.e. exclusive-or). Operations are abstract in the sense that logical equivalence is identity: $a \wedge b$ and $b \wedge a$ are the same operation.

Conversely every Boolean operation in set A of variables uniquely determines the extensional Chu space $(X, \models, A)$ where X is the set of satisfying assignments and $x \models a$ is the truth of a in assignment x. A concrete formula realizing this abstract operation may be formed as the disjunctive normal form formula having one disjunct per row x; each such disjunct is the conjunction of literals, one per $a \in A$, with a appearing positively or negatively according to whether $x \models a$ or not.

By composing these two translations we see that this operation representation of an arbitrary Chu space $\mathcal{A} = (X, \models, A)$ captures it as $(\mathrm{row}_A(X), \models, A)$. The original column index set is lost, being replaced by its image $\mathrm{row}_A(X) = \{\mathrm{row}_A(x) \mid x \in X\}$ under row_A.

The operation φ_A may be understood as a complete description of the space $\mathcal{A}$. We define a *property* of $\mathcal{A}$ to be any Boolean consequence of φ_A; we think of the properties of $\mathcal{A}$ as the *theory* of $\mathcal{A}$, with φ_A as its single axiom and strongest property. Since φ_A is the (infinite if necessary) conjunction of the properties of $\mathcal{A}$, φ_A and the properties of $\mathcal{A}$ determine each other.

A property of a Chu space $\mathcal{A}$ itself determines a Chu space, obtainable from $\mathcal{A}$ by adjoining additional rows. The set of all properties of a Chu space therefore itself forms a power set, and for normal spaces can be given explicitly as $2^{2^A - X}$. In particular *true* has just one property since $X = 2^A$, while *false* has the set 2^{2^A} of properties, namely all Boolean operations in variables from A.

An *axiomatization* of $\mathcal{A}$ is any set of properties whose conjunction is equivalent to φ_A. While the theory of $\mathcal{A}$ axiomatizes $\mathcal{A}$, we can always do better, starting by omitting the property *true* which holds of all Chu spaces. If only axiom count matters then the single axiom φ_A will always suffice. However φ_A is typically constituted as a conjunction of properties of one or another particular kind, various instances of which give rise to various familiar subcategories of **Chu**.

Given two operations φ_A and φ_B, a function $f : A \to B$ defines a *renaming* of the variables of $\mathcal{A}$ to those of $\mathcal{B}$. Renaming is the special case of substitution in which the substituted expressions are merely variables. We write $f(\varphi_A)$ for the result of so renaming the variables of φ_A. For example when $f(a) = d$ and $f(b) = f(c) = e$, $f(a \vee (b \wedge c))$ is $d \vee (e \wedge e)$ which is $d \vee e$ (logical equivalence is identity here).

We say that f *preserves properties* when every property of $\mathcal{A}$ (or equivalently just the property φ_A) renames under f to a property of $\mathcal{B}$. Thus if φ_A is $a \vee b$ and φ_B is $c \oplus d$ (exclusive-or) then $f(a) = d, f(b) = c$ preserves properties (since $f(a \vee b) = f(a) \vee f(b) = d \vee c$ is a consequence of $c \oplus d$) but $f(a) = f(b) = c$ does not (since $c \vee c = c$ is not a consequence of $c \oplus d$).

Theorem 1 *Given normal spaces $\mathcal{A} = (X, A)$, $\mathcal{B} = (Y, B)$, a pair of functions $f : A \to B$, $g : Y \to X$ is a Chu transform $(f, g) : \mathcal{A} \to \mathcal{B}$ if and only if f preserves properties.*

Proof: (If) Given $f : A \to B$, by extensionality of $\mathcal{A}$ there is at most one possible $g : Y \to X$ making (f, g) is a Chu transform. We have $\varphi_B \to f(\varphi_A)$, whence every satisfying assignment $row_B(y)$ of φ_B is a satisfying assignment of $f(\varphi_A)$. Hence the assignment $\mathrm{row}_B(y) \circ f$ to variables of A must be a satisfying assignment of φ_A. Hence there must exist $x \in X$ such that $\mathrm{row}_A(x) = \mathrm{row}_B(y) \circ f$; we may therefore satisfy the adjointness condition by setting $g(y) = x$. Doing this for all $y \in Y$ determines $g : Y \to X$ such that (f, g) is a Chu transform.

(Only if) Let (f, g) be a Chu transform. Each satisfying assignment of φ_B must be $\mathrm{row}_B(y)$ for some $y \in Y$. Now $\mathrm{row}_A(g(y))$ is a satisfying assignment of φ_A. But by adjointness $\mathrm{row}_A(g(y)) = \mathrm{row}_B(y) \circ f$, making $\mathrm{row}_B(y)$ a satisfying assignment of $f(\varphi_A)$. Hence every assignment satisfying φ_B satisfies $f(\varphi_A)$, whence $\varphi_B \to f(\varphi_A)$, whence f preserves properties. ∎

An equivalent condition is that f preserve axioms, since the set of properties preserved by a renaming is closed under arbitrary Boolean operations including arbitrary conjunction.

This view of extensional Boolean Chu spaces as the collection of its properties yields an easy demonstration that extensional Chu spaces can *realize* a great variety of ordered structures: sets, preordered sets, partially ordered sets, Stone spaces, topological spaces, locales, semilattices, complete semilattices, distributive lattices (but not general lattices), algebraic lattices, frames, profinite (Stone) distributive lattices, Boolean algebras, and complete atomic Boolean algebras, to name just the more prominent full, faithful, and concrete subcategories of **Chu**.

Normally these structures stick to their own kind in that each forms its own category, with morphisms staying inside individual categories. Chu spaces bring all these objects into the one self-dual category **Chu**, permitting meaningful morphisms between say semilattices and algebraic lattices, while revealing various Stone dualities such as that between Boolean algebras and Stone spaces, frames and locales, sets and complete atomic Boolean algebras, etc. to be all fragments of the same self-duality.

The notion of realization we intend here is the strong one defined by Pultr and Trnková [PT80]. Informally, one structure *represents* another when they transform in the same way, and *realizes* it when in addition they have the same carrier. Formally, a functor $F : C \to D$ is a *representation* of the objects of C by objects of D when F is a full embedding[2] (a full embedding). A representation F is a *realization* when in addition $U_D(F(A)) = U_C(A)$, where $U_C : C \to \mathbf{Set}$, $U_D : D \to \mathbf{Set}$ are the respective underlying-set functors.

[2] An embedding is a faithful functor $F : C_A \to C_B$, i.e. for distinct morphisms $f \neq g$ of C_A, $F(f) \neq F(g)$, and is *full* when for all pairs a, b of objects of C_A and all morphisms $g : F(a) \to F(b)$ of C_B, there exists $f : a \to b$ in C_A such that $g = F(f)$.

Pultr and Trnková give hardly any realizations in their book, concentrating on mere representations. In contrast all the representations of this paper will be realizations.

The category of Boolean operations and their property-preserving renamings is not self-dual since non-T_0 Chu spaces transpose to nonextensional ones. By the same reasoning the full subcategory consisting of T_0 operations, those with no properties $a \equiv b$ for distinct variables a, b, is self-dual. This is a very important fact: it means that to every full subcategory C of this self-dual category we may associate its dual as the image of C under the self-duality. This associates sets to complete atomic Boolean algebras, Boolean algebras to Stone spaces, distributive lattices to Stone-Priestley posets, semilattices to algebraic lattices, complete semilattices to themselves, and so on for many other familiar [Joh82] and not so familiar (self-duality of finite-dimensional vector spaces over $GF(2)$) instances of Stone duality

We now illustrate the general idea with some examples.

Sets. A set is a Chu space axiomatizable with no axioms; equivalently, an extensional T_0 Chu space whose rows form a complete atomic Boolean algebra or CABA, that is, are closed under complement and arbitrary union. That is, sets are dual to CABA's, argued later. The normal Chu space representing the set A is $(2^A, A)$. Every function $f : A \to B$ between sets $(2^A, A)$ and $(2^B, B)$ is a Chu transform because $(2^A, A)$ has all possible rows whence we can always find g making (f, g) a Chu transform. A better way to see this however is to use the fact that the Chu transforms from $\mathcal{A}$ to $\mathcal{B}$ are those functions $f : A \to B$ that preserve the axioms of $\mathcal{A}$, which must be all functions when $\mathcal{A}$ has the empty set of axioms.

Pointed Sets. A pointed set is a Chu space with the one axiom $a = 0$ (or any other constant from K), this element being the "point." Equivalently it is the result of adjoining a constant column to the Chu realization of a set. (Thus a constant is quite literally a constant column.) For $K = \mathbf{2}$ bipointed sets are also possible, axiomatized as $a = 0$, $b = 1$; in general up to K points are possible. Chu transforms between pointed sets preserve the point: $f(0) = 0$.

Preorders. A preorder is a Chu space axiomatized by "atomic implications," namely propositions of the form $a \to b$ where a and b are variables. A partial order is a T_0 preorder.

Theorem 2 *A Chu space realizes a preorder if and only if it is extensional and its rows form a complete lattice under arbitrary (including empty and infinite) union and intersection.*

Proof: (Only if) Fix a set Γ of atomic implications defining the given preorder. Suppose that the intersection of some set Z of assignments each satisfying all implications of Γ fails to satisfy some $a \to b$ in Γ. Then it must assign 1 to a and 0 to b. But in that case every assignment in Z must assign 1 to a, whence every such assignment must also assign 1 to b, so the intersection cannot have assigned 0 to b after all. Dually, if the union of Z assigns 1 to a and 0 to b, it must assign 0 to b in every assignment of Z and hence can assign 1 to a in no assignment of Z, whence the union cannot have assigned 1 to a after all. So the satisfying assignments of any set of atomic implications is closed under arbitrary union and disjunction.

(If) Assume the rows of $\mathcal{A}$ under union and intersection form a complete lattice.[3] It suffices to show that the set Γ of atomic implications holding in $\mathcal{A}$ axiomatizes $\mathcal{A}$, i.e. that $\mathcal{A}$ contains all satisfying assignments of Γ. Let $x \subseteq A$ be any such assignment. For each $a \in A$ form the intersection of all rows of $\mathcal{A}$ containing a, itself a row of $\mathcal{A}$ containing a, call it y_a. Now form the union $\bigcup_{a\in x} y_a$ to yield a row z of $\mathcal{A}$, which must be a superset of row x. Now suppose $b \in y - x$. Then there exists $a \in x$ such that $b \in y_a$, whence b is in every row of $\mathcal{A}$ containing a, whence $a \rightarrow b$ is in Γ. But x contains a and not b, contradicting the assumption that x satisfies Γ. Hence $b \in y - x$ cannot exist, i.e. $y = x$. However y was constructed from rows of $\mathcal{A}$ by arbitrary union and intersection and therefore is itself a row of $\mathcal{A}$, whence so is x. ∎

This is the essence of the argument showing that posets are dual to profinite distributive lattices [Joh82, p.249], with rows playing the role of ultrafilters or maps to $\perp$, cf. the isomorphism $A \multimap \perp \cong A^{\perp}$ in section 3.3. A normal T_0 Chu space whose columns are closed under arbitrary union and intersection realizes a profinite distributive lattice, which for our purposes suffices for a definition of this notion; consult Johnstone (op.cit.) for an alternative definition.[4] That this is a categorical duality follows immediately from the self-duality of **Chu**.

This result makes it easy to demonstrate the duality of sets and CABA's we promised earlier. One direction is clear: sets contain all possible rows and hence form a CABA by set theory. For the other direction, a CABA is a profinite distributive lattice, whence the theory of its dual is axiomatizable by atomic implications. When $a \leq b$ in a poset for distinct a, b there must exist a satisfying assignment making $a = 0$ and $b = 1$; the complementary assignment, which exists in a CABA, then contradicts $a \leq b$, showing that the theory of the dual of a CABA cannot contain any atomic implications $a \rightarrow b$ for distinct a, b, and hence is axiomatizable with the empty set of axioms.

To complete the argument that this is a realization we need the Chu transforms between posets realized in this way as Chu spaces to be exactly the monotone functions. Now monotonicity is the condition that if $a \leq b$ holds in (is a property of) the source then $f(a) \leq f(b)$ holds in the target. Since the only axioms are atomic implications, monotonicity is equivalent to being axiom-preserving, equivalently property-preserving, hence a Chu transform, and we are

[3] It is worth mentioning that this is a stronger assumption than that the rows of $\mathcal{A}$ partially ordered by inclusion form a complete lattice, since the meets and joins thereof then need not coincide with intersection and union.

[4] Here is a more conventionally abstract but simple and novel definition for lattices. A distributive lattice is *profinite* when it is complete and its maximal chains are *nowhere dense*, that is, every proper interval (pair $a < b$ and all elements between) includes a *gap*, meaning a proper interval containing only its two endpoints.

done.

The following fact about posets will prove useful when we come to locales.

Theorem 3 *A Chu space realizing a poset is column-maximal with respect to the requirement that its rows be closed under arbitrary union and intersection.*

Proof: Adjoin a new element c, and let Γ be the set of atomic implications that are properties of the result. Form the intersection y of those rows containing c, itself a row containing c (even if no rows contain c, in which case we get a new row $A \cup \{c\}$ and we are done). This must be the least assignment satisfying Γ such that c holds. Dropping c from that row then yields a new row that still satisfies Γ, so by the previous theorem the row set was not closed under intersection. ∎

Topological spaces. A topological space is an extensional Chu space whose rows are closed under arbitrary union and *finite* (including empty) intersection. The Chu transforms between topological spaces are exactly the continuous functions. This is most easily seen using the form $\mathrm{row}_A(g(y)) = \mathrm{row}_B(y) \circ f$ of the adjointness condition, with composition (of functions $B \to \mathbf{2}$ as the open sets of $\mathcal{B}$) with f (yielding functions $A \to \mathbf{2}$) being exactly the inverse image function $f^{-1} : 2^B \to 2^A$. Lafont and Streicher [LS91] mention in passing this realization along with that of Girard's coherent spaces [Gir87], also in $\mathbf{Chu}_2$, and the realization of vector spaces over the field K in $\mathbf{Chu}_{U(K)}$.

Locales. A *spatial locale* [Isb72, Joh82] is a *column-maximal* T_0 topological space, one to which no point can be added without defeating the requisite closure properties of the rows. A *locale* is the same less the requirement of extensionality that is imposed automatically for topological spaces. That is, a locale is a column-maximal T_0 Chu space whose rows are closed under arbitrary union and finite intersection. In this case we do not attempt to understand the opens of a locale as sets of points: on the contrary, we prefer to understand points of locales as sets of opens, locales being T_0.

A *frame* (op.cit.) is the dual of a locale; unlike the dual of a topological space in general, a frame is always *algebraic* in the sense that the Chu transforms from it are *all* functions preserving its arbitrary joins and finite meets. In contrast the dual of any infinite poset (a kind of topological space) must have all infinite meets, which Chu transforms from it must preserve. Hence no infinite poset (and *a fortiori* no infinite set) can be a locale.

The only difference between a T_0 topological space and a poset is the possibility that the rows (open sets) may not be closed under certain *infinite* intersections. This difference creates a loop-hole whereby T_0 topological spaces unlike posets (see preceding theorem) need not be column-maximal. For finite topological spaces this distinction disappears, so any example of a topological space not a locale *must* be infinite. As remarked above, any infinite poset provides an example of a nonlocale. A popular example is the chain of natural numbers standardly ordered. Its rows are the order filters or up-sets of $\mathbf{N}$, which are closed under arbitrary intersection; in particular all infinite intersections yield the empty row. This example is not column-maximal: we can add a new point which appears in all rows except the empty one without violating closure under either arbitrary union or *finite* intersection (though the result is no longer closed under intersection of any infinite set of rows that omits the empty row, these intersections being the singleton containing the new point).

Semilattices. The semilattice $(A, \vee)$ is axiomatized by all equivalences $a \vee b \equiv c$ holding in $(A, \vee)$, one for each pair a, b in A. For f to preserve these axioms is to have $f(a) \vee f(b) \equiv f(c)$ hold in the target. But this is just the condition for f to be a semilattice homomorphism, giving us a realization in **Chu** of the category of semilattices.

Equivalently a semilattice is a T_0 extensional Chu space whose columns are closed under binary union and which is row-maximal subject to the other conditions. Row-maximality merely ensures that all rows satisfying the axioms are put in.

The dual of a semilattice is an algebraic lattice [Joh82, p.252].

Complete semilattices. The complete semilattice $(A, \bigvee)$ has all joins, including the empty join and infinite joins. It is axiomatized as for a semilattice, but the left hand sides of its equivalences may be either infinite joins or 0. The dual of a complete semilattice is itself a complete semilattice; thus the subcategory of **Chu** consisting of these complete semilattices is a self-dual subcategory (the same duality).

Distributive Lattices. The idea for the semilattice $(A, \vee)$ is extended to the lattice $(A, \vee, \wedge)$ by adding to the semilattice equations for $\vee$ all equations $a \wedge b = c$ holding in $(A, \vee, \wedge)$ for each a, b in A. Distributivity being a Boolean tautology, it follows that all lattices so represented are distributive. The second (equivalent) formulation of semilattices also extends to distributive lattices along the same lines.

Boolean algebras. A Boolean algebra is a complemented distributive lattice, hence as a Chu space it suffices to add the requirement that the set of rows be closed under complement. Equivalently a Boolean algebra is a T_0 extensional Chu space whose columns form a Boolean algebra under pointwise Boolean combinations (complement and binary union suffice) and which is row-maximal subject to these conditions.

The dual of a Boolean algebra can be obtained as always by transposition. What we get however need not have its set of rows closed under arbitrary union, in which case this dual will not be a topological space. But M. Stone's theorem [Sto36] is that the dual of a Boolean algebra is a totally disconnected compact Hausdorff space. We therefore have to explain how the dual of a Boolean algebra may be taken to be either a topological space or an object which does not obviously behave like a topological space.

There is a straightforward explanation, which at the same time yields a slick proof of Stone's theorem stated as a categorical duality.

The transpose of a Boolean algebra may be made a topological space by closing its rows under arbitrary union. The remarkable fact is that when this adjustment is made to a pair of transposed Boolean algebras, *the set of Chu transforms between them does not change.* (Actually their g components may change, but since these spaces are extensional g is determined by f, which is therefore all that we care about; the f's do not change.)

We prove this fact by first closing the source, then the target, and observing that neither adjustment changes the set of Chu transforms.

Closing the rows of the source under arbitrary union can only add to the possible Chu transforms, since this makes it easier to find a counterpart for a target row in the source. Let $f : A \to B$ be a function that was not a Chu transform but became one after closing the source. Now the target is still a transposed Boolean algebra so its rows are closed under complement, whence so is the set of their compositions with f. But no new source row has a complement in the new source, whence no new source row can be responsible for making f a Chu transform, so f must have been a Chu transform before closing the source.

Closing the rows of the target under arbitary union can only delete Chu transforms, since we now have new target rows to find counterparts for. But since the new target rows are arbitrary unions of old ones, and all Boolean combinations of rows commute with composition with f (a simple way of seeing that f^{-1} is a CABA homomorphism), the necessary source rows will also be arbitrary unions of old ones, which exist because we previously so closed the source rows. Hence Chu transforms between transposed Boolean algebras are the same thing as Chu transforms, and hence continuous functions, between the topological spaces they generate.

This accounts for the fact that the Chu dual of a Boolean algebra is not a topological space. To complete this to a proof of Stone's theorem it suffices to show that the generated topological spaces are totally disconnected, compact, and Hausdorff, omitted here.

An interesting aspect of this proof of Stone's theorem is that usually a duality is defined as a contravariant *equivalence*. Here, all categorical equivalences appearing in the argument that are not actual isomorphisms are covariant. The one contravariant equivalence derives from the self-duality of **Chu**, which is an *isomorphism* of **Chu** with $\mathbf{Chu}^{\mathrm{op}}$. Those equivalences on either side of this duality that fail to be isomorphisms do so on account of variations in the choice of carrier and cocarrier. We pass through the duality with the aid of two independent sets A and X. But when defining Boolean algebras and Stone spaces, in each case we take X to consist of subsets of A, and it is on account of those conflicting representational details that we must settle for less than isomorphism on at least one side of the duality.

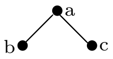
Vector spaces over $GF(2)$. An unexpected entry in this long list of full concrete subcategories of **Chu** is that of vector spaces over $GF(2)$. These are T_0

extensional Chu spaces containing the constantly zero column, with columns closed under binary exclusive-or, and row-maximal subject to these conditions. In the finite case row-maximality is not needed and we obtain symmetric matrices of size a power of two in each dimension. Hence the finite (same as finite-dimensional since the field is finite) vector spaces are self-dual as usual for a finite-dimensional vector space over any field. This follows as the case $k = GF(2)$ of Lafont and Streicher's observation that the category of vector spaces over any field k is realizable in $\mathbf{Chu}_{U(k)}$. There is one finite field of cardinality each power of each prime.

1.3 Chu spaces as partial distributive lattices

In this section we present Chu spaces as two-toned Hasse diagrams or *partial distributive lattices*, abbreviated *pdlat.* This leads to a natural view of a Chu space as a schedule defining a process. The corresponding diagram for its dual gives the corresponding unfolded automaton for the same process.

The columns of Figure 1(b) when ordered pointwise in the obvious way form a partial order which we may depict with the following *Hasse diagram.*



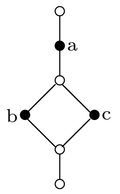
From Fig. 1(b)

But Figure 1(d), which is not isomorphic to Figure 1(b), yields the same partial order. We seek a method of diagramming Chu spaces that distinguishes them up to isomorphism.

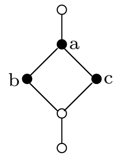
When two columns are equal (contain the same bit pattern) we can at best preorder them. Hasse diagrams are intended for partial orders, but can be adapted to preorders by depicting each maximal clique as a single element labeled with the cardinality of the clique. We therefore address the T_0 case (contrast this with the previous subsection, which assumed extensionality).

Our approach is to close the columns under arbitrary join (disjunction or pointwise OR) and meet (conjunction or pointwise AND), and to depict the resulting poset of columns as a two-toned Hasse diagram whose black elements denote the original columns and whose white ones are those formed by taking the closure. Diagrams we can actually draw will be finite; the "arbitrary" is so that the method will work even for infinite Hasse diagrams, had we only the patience and space to draw them.

When we close up the columns of Figure 1(d) in this fashion, we obtain three new columns: the meet 0001 of b and c, the empty join 0000, and the empty meet 1111. Similarly closing Figure 1(b) yields these columns but in the form 00001, 00000, and 11111, together with a fourth new column, the join of b and c, namely 00111. The corresponding Hasse diagrams are then as follows.



From Fig. 1(b)



From Fig. 1(d)

We can read off from each of these pdlats their common partial ordering of a, b, and c. But we can also read off that the join of b and c is a in 1(d) but not in 1(b). We say that the join of b and c is undefined or does not exist in (b). Their meet exists in neither. We refer to such a structure as a *partial distributive lattice.*

To see that the Chu space can be recovered from the pdlat we identify three sets: the set A of black points, the set 2^X of order filters of X ordered by inclusion (which will be the points of the pdlat without regard for color), and the set X. Ignoring the color scheme for the moment, we start from 2^X as the elements of the given pdlat, a profinite distributive lattice (cf. Theorem 2). We take X to consist of the *complete primes* of 2^X, namely those elements not the join of the set of elements strictly below them. Each element X' of 2^X is then represented as the set of complete primes $Y \subseteq X'$; this representation distinguishes all elements because the lattice is a profinite distributive lattice. The set A of black elements of this lattice then define the columns of the desired Chu space $(X, \leq, A)$ where $\leq$ is the lattice order on 2^X.

1.4 Examples

Up to isomorphism, there are 78 = 2+4+8+64 T_0 extensional Chu spaces of respective cardinalities 0,1,2,3, which we display in Figure 2 below. (The powers of two are pure coincidence; the next two numbers are 3828 and 37320288.) We show each in all three presentation forms: pdlat, equation, and matrix in that order. There are actually two bit patterns shown, the one on the left being the Chu matrix and the one on the right being a representation of the dual of the pdlat we will explain shortly.

The top row shows the 0, 1, and 2 element spaces in three groups, (a)-(b), (c)-(f), (g)-(h). We show all 6 of those having at most one element, but take advantage of certain symmetries to economize the display of the 72 2- and 3-element spaces.

Note that the four one-element pdlats (c)-(f) differ only in whether their top and/or bottom is extended or retracted. These four cases obtain for all nonempty pdlats, so for the pdlats with more than one element we only bother to display those with both bounds retracted. We express the retraction of 1 equationally as $\bigvee A = 1$, and dually $\bigwedge A = 0$ retracts 0. The corresponding matrices are obtained by deleting the columns of all 0's and all 1's respectively. One final economy: 12 of the 16 3-element top-and-bottom-retracted pdlats are not isomorphic to their order duals, so we have omitted half of them leaving just (j)-(l) and (o)-(q) (the starred ones), along with the four that are isomorphic to their order duals, namely (i), (m), (n), and (r).

The discrete pdlats have for their lattices the free distributive lattice $\mathcal{F}_i$ on $i = 0$ to 3 generators[5] (The top and bottom of $\mathcal{F}_2$ and $\mathcal{F}_3$ need to be extended to make them the free *bounded* lattice, whence the quotation marks around "discrete" in those two cases.) All other pdlats are obtainable as quotients, viewable as retracts (quotients), of the free ones, with the retractions being specified by the equations. The free lattice on n generators can be seen to have 2^n "dimensions" along which retractions are permitted. Each retraction is expressible as a series of projections each projecting out one dimension. For example (r) retracts to (q) along dimension q, which then disappears as a distinguishable dimension. In any such retraction, if an edge of a square contracts then so does the opposite edge, thereby identifying the square's other two edges.

The diagram to the right of the matrix for A is the pdlat for $A^\perp$, an n-cube in standard position (cf. pdlat (o)) with its edges deleted to reduce clutter (note that (i)-(k) need only 6 of the eight vertices). The solid points denote the real elements of the dual, the circles its lattice-imaginary elements, and the periods its remaining Boolean-imaginary elements.

The real (black) elements of $A^\perp$ correspond to the dimensions of A and to the columns of the matrix representation, e.g. (r) has six dimensions, its matrix six columns, and its dual the six reals p-u. Retracting along a dimension of A corresponds to deleting the corresponding matrix column and the corresponding real of $A^\perp$; the imaginary elements are then whatever can be generated from the remaining reals. (The duals of smaller pdlats are represented similarly: for (g)-(h) the nonbounds are p and q, while for (c)-(f) the bounds themselves are named p and q.)

This is best understood by starting with (r) and regarding the remaining three-element pdlats as quotients of (r), and their corresponding duals as subpdlats of the dual of (r), whose six elements p-u can be tracked as they disappear, e.g. (m) is obtained by retracting along dimension u, the dual of deleting element u. Note that the correspondence between matrix columns and their labels is maintained during this process, the column labelled u for example always being 001.

[5] Sloane [Slo73] gives the size of the free distributive lattices on n generators as sequence 309, "Motonone Boolean Functions," namely 2, 3, 6, 20, 168, 7581, 7828354, 2414682040998, ...

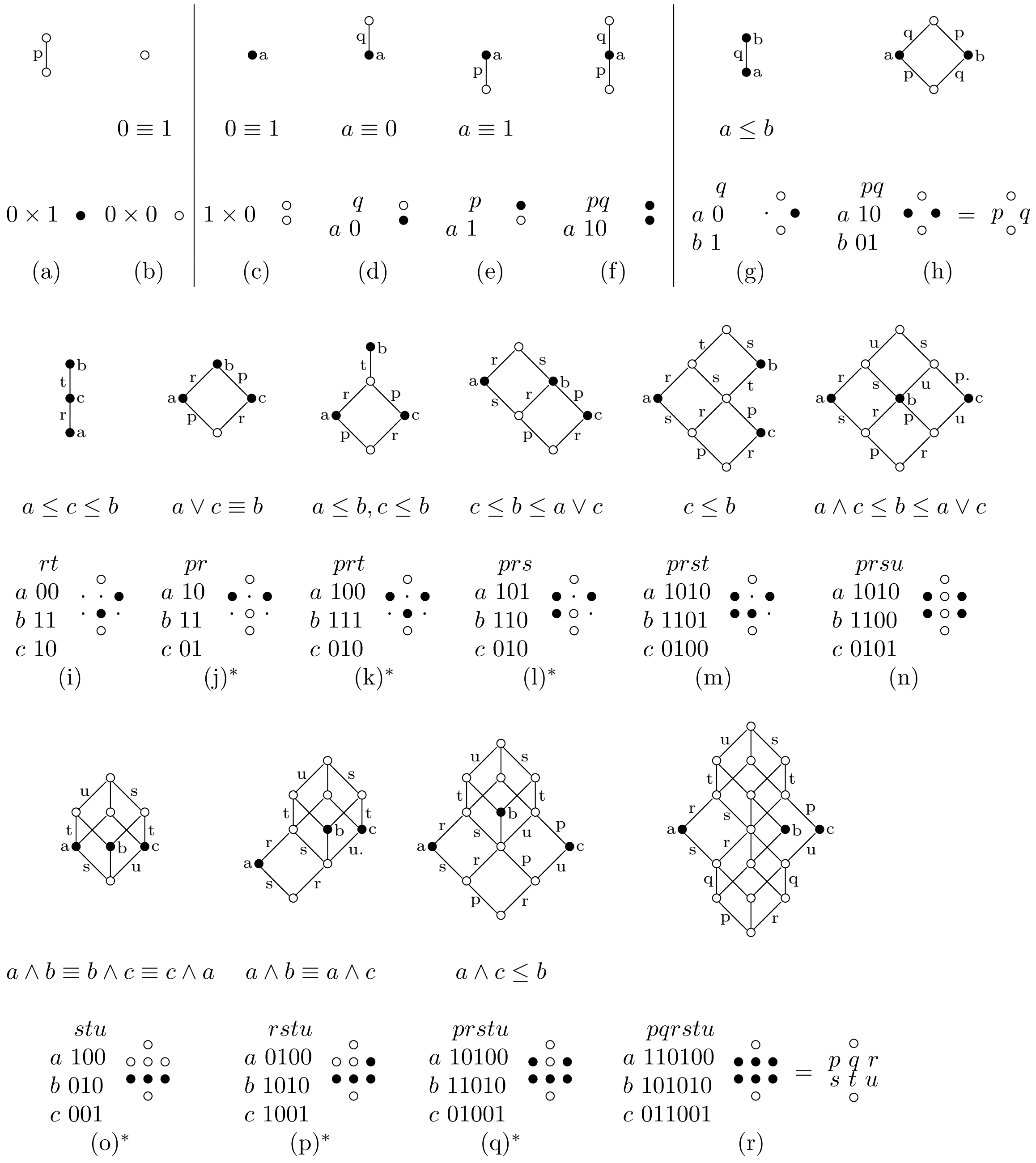


Figure 2. The T_0 extensional Chu spaces with up to 3 points.

2 Behavior: from event structures to rational mechanics

The classical 1970's conception of an automaton was as a device for accepting a formal language defined as a set of strings, possibly infinite in the case of so-called ω-automata. This conception made two automata equivalent when they accepted the same language. As models of behavior, each string of the accepted language was considered as one of the alternative or *possible* behaviors or *runs* of that automaton, and the symbols in that string all occurred during that run, in the order of occurrence in that string. In this context strings are usually referred to as *traces*.

We shall understand automata formally as follows. A *transition system* (X, Σ, δ) consists of a set X of *states* $x, y, z, \ldots$, a set Σ (or *Act*) of *actions* $a, b, c, \ldots$ (the alphabet), and a set $\delta \subseteq X \times \Sigma \times X$ of *transitions* (x, a, y), the *transition relation.* An *automaton* $(X, \Sigma, \delta, x_0, F)$ is a transition system with a distinguished state x_0, the *initial* state, and a set F of *final* states.

Instead of treating automata and languages as separate notions, it will clarify the sequel if we regard both strings and languages as just special kinds of automata. The string *aba* is taken to be the four-state straight-line automaton $0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 3$ having state set $X = \{0, 1, 2, 3\}$, initial state 0, and set $\{3\}$ of final states. We allow infinite strings, whose state set may then be taken to be the set of natural numbers, with initial state 0 and with no final state. The language L is taken to be the automaton whose state set is formed as the disjoint union of the state sets of the strings of L viewed as automata, with their initial states identified to form the single initial state of L, and with the set of final states of L being those of the individual strings.

Formally, a *language* L is an automaton $(X, \Sigma, \delta, x_0, F)$ with the following properties. (i) Every state in X has zero or one transition leading to it according respectively to whether it is or is not the initial state x_0. (ii) Every state is reachable from the initial state by a finite path of transitions. (iii) Every noninitial state x has zero or one transition leaving it according respectively to whether x is or is not a final state ($x \in F$ or $x \notin F$).

It follows that a language is a rooted tree, with only the root (the initial state) allowed more than one descendant. Further the initial state may or may not belong to the set F of final states, corresponding respectively to whether the empty string is or is not a member of L.

Motivated by such concerns as fair scheduling of concurrent infinite behaviors, a variety of more general acceptance criteria all involving repeated visits to final states have been considered, giving rise to the automata of Büchi, Rabin, and Streett. In these kinds of automata acceptance of an infinite string can only be determined given the whole string. Reflecting the scope to date of our work on applying Chu spaces to the modeling of behavior, the emphasis of this paper will be on finitely observable properties of automata and we shall therefore not consider these more general acceptance criteria here.

The new automata theory raised two objections to this conception, which in due course came to be called respectively branching time and true concurrency.

2.1 Branching Time.

The first objection was raised by Robin Milner in his book on CCS, a Calculus of Communicating Systems [Mil80]. The standard model appears to condense all choices about behavior into a single selection of a string from a language made at the start of the behavior. Real behavior however makes informed decisions on the fly as information comes to hand. Milner proposed a logic that took deferred branching into account by abandoning the equation $a(b+c) = ab+ac$, and introducing a model, synchronization trees, to serve as counterexamples for this equation. This equation was taken as characteristic of *linear time*, its antonym being *branching time.*

A *synchronization tree* is an automaton satisfying conditions (i) and (ii) for languages. The omission of condition (iii) extends to all states the branching privilege that languages enjoy only at the initial state. (Less significantly it also removes the requirement that a noninitial *leaf* state, one with no transition leaving it, be a final state; one may impose this latter requirement as a weakened form of (iii), but we shall omit this as a complication irrelevant to our purposes.) Whereas the evident synchronization tree $ab + ac$ is a language having five states, the four-state synchronization tree $a(b+c)$ is not a language because the noninitial state to which the a transition leads has two transitions leaving it.

We regard the distinction that synchronization trees draw between $ab + ac$ and $a(b+c)$ as one of *timing.* Both trees entail the decision whether to perform action b or action c. But whereas $ab + ac$ makes this decision at the initial state, before a has been performed, $a(b + c)$ makes it at the state following the a transition. We understand the difference to be the additional information obtained by performing a: $a(b+c)$ makes a *more informed* decision than $ab+ac$ thanks to the information brought to light by a.

Thus a synchronization tree is intermediate in abstractness between automata and languages. While synchronization trees can draw distinctions based on timing of decisions, they cannot draw other distinctions available to general automata related to looping structure and confluence. For example a synchronization tree cannot distinguish a^* (short loop) from $(aa)^*(\epsilon + a)$ (length-two loop), nor $(a + b)c$ (confluence or rejoining) from $ac + bc$ (nonconfluence).

Since cyclic structure entails confluence (a loop must return to either the initial state as a degenerate case of confluence, or to a state reachable by some other path from the initial state), we may regard a synchronization tree as a confluence-free automaton.

Every automaton can be approximated by a synchronization tree in a canonical way, namely the tree whose states are the paths leading from the root and whose transitions are the singleton path extensions (p, a, q) where q is a path extending path p with an a transition. The initial state is the empty path while the final states are those paths terminating in a final state of the given automaton.

In turn, every synchronization tree can be approximated by a language in a canonical way, namely the language whose strings are the root-to-final-state paths and all[6] infinite paths in that tree. The passage from tree to language may be understood as the "teasing apart" of the paths of the tree, moving all branching points up to the root.

2.2 True concurrency

The second objection to the language interpretation of automata, raised sporadically by various people over a long period [Pet62, Gre75, Maz77, Gra81, NPW81, Pra82], was that the standard model assigned a well-defined order to every pair of events (symbol occurrences) in the same string. Besides contradicting relativity, this assumption also contradicts practical engineering issues at all scales, from "data skew" on parallel signal lines within a single chip to detecting when a husband and wife are simultaneously making withdrawals from the same account at remote automatic teller machines. At the 1988 Königswinter conference on true concurrency, Robin Milner with tongue in cheek referred to global ordered time as "false concurrency;" a more informative term would be *atomic mutual exclusion* which postulates that atomic events are mutually exclusive in the sense of not being permitted to overlap in time.

Our notion of automaton appears to enforce this linear ordering of the events that actually occur during a run of an automaton. A straightforward way to relax this requirement is to generalize the notion of string to that of partial string, called a *partial word* by Grabowski [Gra81] and a *partially ordered multiset* by Pratt [Pra82], later shortened to *pomset* [Pra84, Gis88].

A *pomset* $(A, \leq, \Sigma, \lambda)$ is a set A of *events*[7] partially ordered by a binary relation $\leq$, the *temporal precedence relation*, and a *labeling function* $\lambda : A \to \Sigma$ assigning to each event $a \in A$ its *label* $\lambda(a) \in \Sigma$ denoting the action performed by a. A string is then taken to be the special case where the pomset is linearly ordered.

Event structures [NPW81] generalize pomsets to incorporate a notion of alternative behavior expressed as conflict information. An *unlabeled event structure* $(A, \leq, \#)$ consists of a set A of events partially ordered by $\leq$, together with

[6] Or the accepted infinite paths when treating automata endowed with infinitary acceptance criteria.

[7] We shall henceforth use $A = \{a, b, c, \ldots\}$ to denote events or *action occurrences*, and use $\alpha, \beta, \ldots$ instead of $a, b, \ldots$ for the actions themselves, which we shall understand as labels on events.

a symmetric irreflexive binary relation #, denoting conflict, such that if $a\#b$ and $b \leq c$, then $a\#c$. When $a\#b$ holds, this indicates that the events a and b cannot both occur in the same run. A *labeled event structure* $(A, \leq, \#, \Sigma, \lambda)$ adds an action alphabet Σ and a labeling function λ as for pomsets.

Just as a synchronization tree may be understood as a confluence-free automaton, a pomset may be understood as a conflict-free event structure.

2.3 Relating Automata and Event Structures

Automata and labeled event structures may represent each other in the following canonical ways.

Event structures cannot represent any of the information lost in the passage from an automaton to its canonical approximation by a synchronization tree. Hence we translate automata to event structures in two stages via synchronization trees. We already have the first stage, we now give the second.

The synchronization tree $(X, \Sigma, \delta, x_0, F)$ is represented by the labeled event structure $(\delta, \leq, \#, \Sigma, \lambda)$ such that (i) $(x, \alpha, y) \leq (x', \beta, y')$ just when these two transitions occur in that order on some path of the tree, (ii) $(x, \alpha, y)\#(x', \beta, y')$ just when these two transitions do not both appear on the same path in the tree, and (iii) $\lambda(x, \alpha, y) = \alpha$.

In the other direction, the labeled event structure $(A, \leq, \#, \Sigma, \lambda)$ is represented by the *automaton* $(X, \Sigma, \delta, x_0, F)$ where (i) $X \subseteq 2^A$ consists of the conflict-free order ideals of the event structure, namely those subsets $x \subseteq A$ such that $a \in x$ and $b \leq a$ implies $b \in x$, and such that $a\#b$ implies not both $a \in x$ and $b \in x$; (ii) $(x, \alpha, y) \in \delta$ just when $y - x = \{a\}$ where $\lambda(a) = \alpha$; (iii) $x_0 = \{\}$; and (iv) F consists of the maximal elements of X.

For example the discretely ordered conflict-free event structure

$$(\{a, b\}, \{(a, a), (b, b)\}, \{\}, \{a, b\}, \lambda x.x)$$

is approximated by the automaton whose state set X is the power set of $\{a, b\}$; whose transition relation consists of the four transitions $(\{\}, a, \{a\})$, $(\{\}, b, \{b\})$, $(\{a\}, b, \{a, b\})$, and $(\{b\}, a, \{a, b\})$; whose initial state is empty; and whose final state set is $\{\{a, b\}\}$. Modifying this event structure by ordering it as $a \leq b$ has the effect of deleting the state $\{b\}$ and its two incident transitions. If instead the two events are put in conflict, $a\#b$, this has the effect of deleting the state $\{a, b\}$ from X, and its two incident transitions from δ, and the final state set then becomes $\{\{a\}, \{b\}\}$, these now being the two maximal states in X. Making both modifications deletes both those states; the result is equivalent, with respect to this translation, to the event structure having just the one event a.

This translation then yields another, namely from event structure to synchronization tree: just compose the above translation from event structure to automaton with the canonical approximation of an automaton by a synchronization tree treated earlier.

Now when the synchronization tree of the above example is translated to an event structure we find that all pairs a, b of events are related by either $\leq$ or $\#$. On the one hand this event structure is different from the one we began with; on the other, it then translates back to the same synchronization tree. Thus only the first of these translations has lost any information, namely concerning whether or not a and b are mutually exclusive in the sense of not being permitted to happen concurrently.

But note that the critical true-concurrency information, namely the independence of a and b, was lost in the passage from the automaton to the synchronization tree. Since the automaton is acyclic, this passage only "unfolds" confluences, there are no loops to unwind. Those confluences that do arise may be associated with the absence of both conflict and order between the confluent events. In this way the translation to the automaton loses less information; in particular the independence of two events may be recovered from their appearance in the automaton as a confluent pair. Nevertheless some information is lost, as witnessed by the event structure $a \leq b$, $a\#b$ which yields the same automaton as the event structure with just the one event a. The Chu account of automata cleanly resolves this and related issues.

Now event structures and synchronization trees can both exhibit branching structure. However there is an important difference. Whereas any one run of a synchronization tree takes only *one* path out of each branch, a run of a pomset performs *all* events, and a run of an event structure performs all the events of some maximal non-conflicting set of events.

We shall understand this difference by viewing the set X of states of an automaton as a *disjunctive* set, and the set A of events of an event structure as a *conjunctive* set. This distinction is implicit in the transformation of A and X by respectively functions and antifunctions. The following gives some additional insight into this distinction.

Both automata and event structures are graphs. However their edges have the following dual character. An edge (x, a, y) of an automaton *enables* behavior in that it permits an a transition from x to y. This gives automaton edges the character of the modal operator $\Diamond$ expressing the possibility of a transition to a state. An edge of either form $a \leq b$ or $a\#b$ of an event structure *constrains* behavior in that it limits the possible states of the corresponding automaton. This associates event structure edges with the modal operator $\Box$ expressing a *necessary constraint.*

2.4 Behavioral Interpretation of Chu Spaces

We interpret the Chu space $(X, \models, A)$ as a process with state set X, event set A, and *occurrence relation* $x \models a$ indicating that in state x event a has already happened. The "has already happened" in this interpretation is where time enters the Chu space picture of computation.

As an $X \times A$ Boolean matrix, a Chu space is a two-dimensional object. We take the vertical axis to be the information axis and the horizontal as the temporal axis. This corresponds to states being distributed in an information space and events in a temporal space. This is in complete agreement with the dimensions of the logics of *sequential* behavior we have collected under the rubric of "two-dimensional logic" [Pra94, p.156], where we said:

> We shall visualize the dimensions as oriented respectively vertically and horizontally. (This will be recognized as in agreement with 2-category usage, where 1-cell composition is horizontal and 2-cell vertical.) It is natural to associate information or static logical strength with the vertical axis and time or dynamic progress with the horizontal.

In this viewpoint a two-dimensional logic is a set of propositions-cum-actions ordered somewhat independently in these two dimensions, with the vertical ordering imparting the static character of propositions and the horizontal ordering (actually a monoid) conferring the dynamic character of actions. This is an unsorted framework; the program-proposition sort distinction drawn by dynamic logic is viewed from this perspective not as fundamental but merely as a minor syntactic distinction leading to decidability and finite axiomatizability of the propositional case [Pra90]. What Chu spaces do is put the propositions (*including* the programs) of the *language* of dynamic logic, action logic, etc. and the states of their *semantics* on an equal footing as forming the dual sets A and X of respectively events and states.

Recall that a property of a Chu space is a superset of its state set, equivalently, a Boolean consequence of the space viewed as a Boolean proposition. The following succinctly expressed properties correspond naturally to various aspects of concurrency. Any property expressed using states rather than events in the following (e.g. transitions) is to be understood as a property of the dual space.

Transition: A state x can evolve into a state y just when $x \to y$. These are just inclusions between states, all of which we regard as legitimate state transitions.

Temporal order: a occurs before b if $b \to a$. Winskel writes this as $a \vdash b$, called *prime enabling* [Win88]. If $b \to a$, then every state with $b = 1$ must also have $a = 1$, which means that a must have been set to 1 no later than b.

Enabling: General (nonprime) enabling has the form $a, b \vdash e; \quad c, d \vdash e$, an enablement of e equivalent to the Boolean formula $e \to (a \wedge b) \vee (c \wedge d)$ (either one of (a with b) or (c with d) suffices to enable e), i.e. any number of pre-events before the $\vdash$, and any number of $\vdash$'s.

Conflict: $a\#b$ is $\neg(a \wedge b)$ (binary or coherent conflict[NPW81]), and means that it is illegal to set both a and b to 1, i.e. they are in conflict. More generally, $\#x$ may be defined as $(\bigwedge_{a \in x} a) \equiv 0$ (no state contains all events in x, i.e. no superset of x is a state of X.)

Internal choice: $x \equiv y \wedge z$ and $\#(y \vee z)$ expresses the choice of conflicting states y or z, made in the state x, so this choice is "internal". This corresponds to the branching construct of programming languages, where a choice is made based on the information accumulated in the current state. We can have a choice between several states, like a `case` statement in C.

Causality: $a \wedge b \equiv c$ asserts that a and b jointly cause c as their *immediate* effect, as it is impossible to have done both a and b without doing c also. On the other hand $c \to a \wedge b$ is mere prime enabling, $a, b \vdash c$, that is it is OK to wait for a while before doing c. This distinction is absent from all other models of concurrency we are aware of.

Nondeterminism: $a \equiv b \vee c$ and $b\#c$ asserts choice of conflicting events b or c. Since the choice is made at the same time as doing a, any information gathered by doing a could not have been used to choose, however, the choice was not available before a. So this choice is made by the environment, and is "external". This contrasts with $b \vee c \to a$, which is mere prime enabling $a \vdash b,\ a \vdash c$.

Synchronization: $a \equiv b$ asserts that a and b must happen simultaneously. A T_0 space has only identity synchronizations $a \equiv a$. Conditional synchronization, $a \to b \equiv c$ however may hold even for T_0 spaces: it holds for causality $(a \wedge b) \equiv c$.

3 Algebra: from linear logic to process algebra

In this section we first treat the linear logic of Chu spaces in its own right. We then give a process algebra interpretation of the connectives of linear logic. In this interpretation linear logic is by no means a complete process algebra; we accordingly round out the language to a more comprehensive process algebra by providing several additional connectives. Whereas the linear logic connectives are naturally defined categorically as functors on **Chu** with suitable universal properties, these additional connectives are defined set theoretically, and some are not even functorial.

3.1 Linear logic

The language of linear logic, LL, closely parallels that of relation algebras, RA. The latter amounts to two copies of the logical connectives *or*, *false*, *and*, *true*, *not*, and *implies*, distinguished as the *logical* and *relative* (relational) forms of those connectives, due to Peirce but anticipated to some extent by by De Morgan [DM60]. To these Schröder [Sch95] added reflexive transitive closure a_0, nowadays a^*, and its De Morgan dual a_1.

Combining the separate involutory logical and relative duals, a^- and $a^{\smile}$, as a single involutory ($a^{\perp\perp} = a$) dual $a^{\smile -} = a^{\perp}$ [Pra92a, p.252] weakens the Boolean structure of RA to that of a De Morgan lattice [Dun86, p.184,p.193], since neither $a + a^{\perp} = 1$ nor $aa^{\perp} = 0$ hold of binary relations. This seems in practice to leave the utility of RA largely unimpaired, whose operations are then as follows.

	Logical :	$a{+}b$	0	ab	1
Relation	*Relative* :	$a \dagger b$	0’	$a;b$	1’
Algebra:	*Nonmonotone* :	$a^{\perp}$	$a\backslash b$	b/a	$a \rightarrow b$
	Closure :		a^*	a_1	

Interpreted standardly for binary relations over a fixed set, the logical connectives are union, empty, intersection, and the complete relation. The relative connectives are $a \dagger b = (a^-;b^-)^- = (b^{\perp};a^{\perp})^{\perp}$ (the De Morgan dual of composition), $\neq$, composition, and $=$. The nonmonotone connectives are complement-of-converse $a^{\perp} = a^{-\smile}$, right residuation $a\backslash b = a^{\perp} \dagger b$, left residuation $b/a = b \dagger a^{\perp}$, and “static implication” $a \rightarrow b = a^- + b$. The closure operations are reflexive transitive closure and its De Morgan dual $a_1 = ((a^{\perp})^*)^{\perp}$.

These operations are not independent, and a suitable basis is $a{+}b$, 0, $a;b$, 1’, $a^{\perp}$, and a^*.

The language of linear logic can be closely matched to this as follows. We use Barr and Seely’s notation [Bar91, See89] in preference to Girard’s more idiosyncratic notation [Gir87].

	Additives :	$A{+}B$	0	$A{\times}B$	1
Linear	*Multiplicatives* :	$A \oplus B$	$\perp$	$A \otimes B$	$\top$
Logic:	*Nonmonotone* :	$A^{\perp}$		$A{\multimap}B$	$A{\Rightarrow}B$
	Exponentials :		$!A$	$?A$	

Additives. Just as the Boolean or static connectives of RA can be inferred solely from the inclusion order among relations, ignoring the monoid structure $a;b$, so can the additives of linear logic be inferred solely from the categorical (morphism) structure on the category of Chu spaces, ignoring the monoidal structure $A \otimes B$. The additives are respectively coproduct, the initial object, product, and the final object.

The coproduct $\mathcal{A} + \mathcal{B}$, where $\mathcal{A} = (X, \models_A, A)$, $\mathcal{B} = (Y, \models_B, B)$, is defined as $(X \times Y, \models, A + B)$ where $X \times Y$ is the cartesian product of the state sets, $A + B$ is the disjoint union $A \times \{0\} \cup B \times \{1\}$ of the event sets, and $\models$ satisfies $(x, y) \models (a, 0) \;=\; x \models a$, $(x, y) \models (b, 1) \;=\; y \models b$. The associated inclusion $i_A : A \rightarrow A + B$ is defined as (i_A, p_X) where $i_A : A \rightarrow A + B$ is the inclusion in **Set** and $p_X : X \times Y \rightarrow X$ is the projection in **Set**; and analogously for $i_B : B \rightarrow A{+}B$. We verify that (i_A, p_X) is a Chu transform with the calculation $p_X(x, y) \models_A a = x \models_A a = (x, y) \models (a, 0) = (x, y) \models i_A(a)$.

To see that $A{+}B$ with these two inclusions is coproduct, it suffices to exhibit a natural bijection between pairs of Chu transforms $A \xrightarrow{f} C \xleftarrow{g} B$ and those Chu transforms $A+B \xrightarrow{h} C$ such that $hi_A = f$ and $hi_B = g$. The trick is to establish this bijection separately for the covariant and contravariant components and then verify that each pair of maps making up some $A + B \xrightarrow{h} C$ is a Chu transform, and (immediate) that all Chu transforms satisfying $hi_A = f$ and $hi_B = g$ arise in this way.

The initial object 0 is $(1, !, 0)$, meaning the 1-state 0-event Chu space. Initiality is immediate.

We obtain the product $A \times B$ and the final object as the De Morgan duals (with respect to $A^\perp$) of coproduct and the initial object. That is, $A \times B$, in full $(X, \models_A, A) \times (Y, \models_B, B)$, is $(X + Y, \models, A \times B)$ where $\models$ satisfies $(x, 0) \models (a, b) = x \models a$, $(y, 1) \models (a, b) = y \models b$, with associated projections $p_A : A \times B \to A$ defined as (p_A, i_X) where $p_A : A \times B \to A$ is the projection in **Set** and $i_X : X \to X + Y$ is the inclusion in **Set**.

Nonmonotone Connectives. (We treat the monotone connectives first as being easier.) The *dual* $(X, \models, A)^\perp$ of a Chu space is simply its transpose $(A, \breve{\models}, X)$. The *linear implication* $(X, \models_A, A) \multimap (Y, \models_B, B)$ is $(A \times Y, \models, A \to B)$ where $A \to B$ denotes the *set* of Chu transforms (f, g) from Chu space A to Chu space B and $\models$ satisfies $(f, g) \models (a, y) = g(y) \models_A a = y \models_B f(a)$ (the second equation is just the adjointness condition, but points up the symmetry that the first equation by itself might otherwise conceal).

The *intuitionistic implication* $(X, \models_A, A) \Rightarrow (Y, \models_B, B)$ is $(A \times Y, \models, A \to B)$ where this time $A \to B$ denotes the set of *functions* $f : A \to B$ from event set A to event set B, and $\models$ satisfies $f \models (a, y) = y \models_B f(a)$.

Multiplicatives. We may define $A \oplus B = A^\perp \multimap B$, $\perp = (\{0\}, \in, \{\{\}, \{0\}\})$, $A \otimes B = (A^\perp \oplus B^\perp)^\perp = (A \multimap B^\perp)^\perp$, and $\top = \perp^\perp = (\{\{\}, \{0\}\}, \breve{\in}, \{0\})$. Thus $\perp$ is a 1×2 matrix while $\top$ as its transpose is 2×1; in both cases the two entries are respectively 0 and 1, corresponding respectively to 0 being a nonmember or member of the given set.

Exponentials. We take $!A = A \Rightarrow \perp$ and $?A = (!A^\perp)^\perp$. We could with isomorphic effect have defined $!(X, \models, A)$ as the normal Chu space $(2^A, A)$ (realizing the set A) and defined $A \Rightarrow B$ in terms of $!A$, namely via $A \Rightarrow B = !A \otimes B$.

Except when defining these operations, we shall often write their operands as A and B instead of $\mathcal{A}$ and $\mathcal{B}$, provided no confusion is likely.

3.2 Definition by Circuits

The Boolean operation view of Chu spaces leads to a natural way to define certain operations on Chu spaces, namely with Boolean circuits that combine those operations.

The units for sum and tensor product, namely 0 and $\top$, and the exponential $!\mathcal{A}$, are all defined by circuits whose output is constantly 1 and hence whose inputs are ignored. Zero has no inputs, the tensor unit $\top$ has one input, and $!\mathcal{A}$ has for its inputs those of $\mathcal{A}$. (This definition of $!\mathcal{A}$ satisfies the Girard axioms but also satisfies $!!\mathcal{A} \cong !\mathcal{A}$, leaving open the possibility of a less constrained alternative Chu interpretation for $!\mathcal{A}$.)

The sum $\mathcal{A} + \mathcal{B}$ is implemented as a circuit with components $\mathcal{A}$ and $\mathcal{B}$, by forming the conjunction of the outputs of $\mathcal{A}$ and $\mathcal{B}$, with the set of inputs of $\mathcal{A}+\mathcal{B}$ then being the disjoint union $A+B$ of those of $\mathcal{A}$ and $\mathcal{B}$. This construction is illustrated in Figure 3, for which $|A| = 3$ and $|B| = 2$.

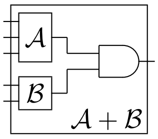


Figure 3. Circuit for Sum

The space thus implemented can be seen to partition its inputs into two disjoint blocks, one judged by $\mathcal{A}$, the other by $\mathcal{B}$, with the space as a whole registering its approval just when all its components approve of their respective blocks.

As it turns out, the categorical product $A \times B = (A^\perp + B^\perp)^\perp$, i.e. the dual of coproduct, is circuit-definable, as follows.

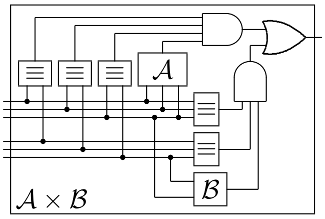


Figure 4. The product of two gates.

Exercise: decipher this.

Tensor product $\mathcal{A} \otimes \mathcal{B}$ is the only operation requiring some work, and is where the circuit approach to defining operations really helps. As one might guess from the fact that the tensor unit has output 1, tensor product is a form of conjunction. But whereas sum is a noninteracting conjunction, tensor product behaves like a generic logical inference, in which the constraints in the components can entail new constraints involving the variables of both components.

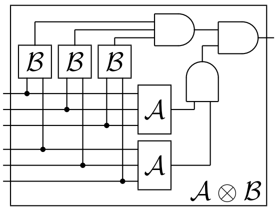


Figure 5. Circuit for Tensor Product

Now as with sum, tensor product assumes no *a priori* relationship between the inputs of its arguments, which are therefore made disjoint. Nevertheless a connection is established between the two sets of inputs, namely by taking the set of inputs of $\mathcal{A}\otimes\mathcal{B}$ to be the cartesian product $A\times B$ of those of $\mathcal{A}$ and $\mathcal{B}$ instead of their sum. Visualizing this product as a rectangular array of inputs, we define *bilinearity* to be the condition that each column of this rectangle (what one obtains by fixing a particular $b\in B$ of $\mathcal{B}$'s inputs) independently satisfies $\mathcal{A}$ while each row satisfies $\mathcal{B}$. This is realized by using $|B|$ distinct copies of $\mathcal{A}$ to monitor each column of $A\times B$, and likewise $|A|$ copies of $\mathcal{B}$ to monitor rows, as illustrated in Figure 5 for the case where the rectangle is 3×2.

3.3 Equational Logic

We have the *equation* $a{+}b = b{+}a$ for relation algebras, and it is natural to expect this for linear logic as well. However on closer inspection we notice that each a in the carrier A of $\mathcal{A}$ becomes $(a,0)$ in $A{+}B$ but $(a,1)$ in $B{+}A$. We may however claim the *isomorphism* $A{+}B\cong B{+}A$, in which $(a,0)$ in $A{+}B$ is matched up with $(a,1)$ in $B{+}A$. (A is isomorphic to B when there exist bijections $X_A\cong X_B$, $Y_A\cong Y_B$ of their index sets making their corresponding entries equal.) This applies to the other laws of linear logic as well, with the exception of $A^{\perp\perp}=A$ and definitions (e.g. $A\otimes B = (A\multimap B^{\perp})^{\perp}$. The full list of isomorphisms (and equalities where possible) we know to hold for extensional T_0 Chu spaces is as follows.

$$\begin{array}{rcl@{\qquad}rcl@{\qquad}rcl}
A{+}(B{+}C) & \cong & (A{+}B){+}C & A{+}0 & \cong & A & A{+}B & \cong & B{+}A\\
A\otimes(B\otimes C) & \cong & (A\otimes B)\otimes C & A\otimes\top & \cong & A & A\otimes B & \cong & B\otimes A\\
A\otimes(B{+}C) & \cong & (A\otimes B){+}(A\otimes C) & A\otimes 0 & \cong & 0 & A^{\perp\perp} & = & A
\end{array}$$

From these laws and the definitions of abbreviations we can derive for example $(A\otimes B)\multimap C = (C^{\perp}\otimes(A\otimes B))^{\perp}\cong((C^{\perp}\otimes A)\otimes B)^{\perp} = B\multimap(C^{\perp}\otimes A)^{\perp} = B\multimap(A\multimap C)$. We also have $A\multimap B = (A\otimes B^{\perp})^{\perp}\cong(B^{\perp}\otimes A^{\perp\perp})^{\perp} = B^{\perp}\multimap A^{\perp}$, and $A\multimap\perp\,\cong(A\otimes\top)^{\perp}\cong A^{\perp}$. We leave $(A{\times}B)\Rightarrow C\cong A\Rightarrow(B\Rightarrow C)$ as an exercise. We are not aware of any completeness results for the isomorphism theory of Chu spaces.

Note that $A^{\dagger}$, $A^{\dagger\dagger}$, $A^{\dagger\dagger\dagger}\ldots$ is $Y_A, K^{Y_A}, K^{K^{Y_A}},\ldots$, in contrast to $!!A = !A$.

3.4 Process algebra

The linear logic operations $A + B$ and $A \otimes B$ for Chu spaces realize the process algebra operations we have previously called in a series of papers respectively *concurrence* and *orthocurrence* [Pra85, Pra86, CCMP91].

Basic process algebra operations not provided by any linear logic connectives include *choice* $\mathcal{A} \sqcup \mathcal{B}$ and *sequence* $\mathcal{A};\mathcal{B}$.

We define the *choice* $\mathcal{A} \sqcup \mathcal{B}$ of normal Chu spaces $\mathcal{A} = (X, A)$ and $\mathcal{B} = (Y, B)$, assumed to have disjoint event sets A, B, as $(X \cup Y, A \cup B)$. Disjointness of event sets ensures that the empty state will be the only state the arguments have in common, which this construction therefore identifies. $\mathcal{A} \sqcup \mathcal{B}$ chooses which of $\mathcal{A}$ or $\mathcal{B}$ it is going to do as soon as it performs its first event from one of the arguments, since no state of the other argument contains that event.

The circuit representation of this definition is as follows.

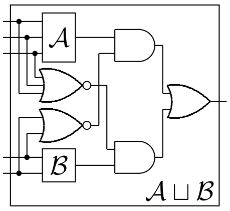


Figure 6. Choice of two Chu spaces.

We define the *sequence* $A; B$ by deleting from the states of $A + B$ those states x which contain an event of B and for which there exists a state $y \supseteq x$ in $A + B$ such that $y - x$ contains an event of A. A simple example of this is when A and B each have one event (respectively a and b) and two states. Then $A + B$ has two events a, b and as states the four subsets of $\{a, b\}$. $A; B$ deletes state $\{b\}$ from this because it contains an event of B and there exists a state $\{a, b\}$ such that $\{a, b\} - \{b\}$ contains an event of A.

We have omitted *iteration* $\mathcal{A}^*$, and more generally recursion, from this treatment.

4 Relational structures

We have seen that **Chu** is a sort of universal category for lattice theory. This section extends the universality of Chu spaces to arbitrary relational structures and their homomorphisms, by passing to $\mathbf{Chu}_K$ for larger K. This is the Chu space version of the passage from propositional logic to first-order logic.

We have already mentioned Lafont and Streicher's observation [LS91, p.45] that the category of vector spaces over a field $\mathbf{K}$ is a full subcategory of $\mathbf{Chu}_K$, and that the category **Top** of topological spaces is a full subcategory of $\mathbf{Chu}_2$.

We improve on these observations by showing that *every* κ-ary relational structure is realizable as an object of $\mathbf{Chu}_{2^\kappa}$, giving a strong sense in which Chu spaces form a universal category.

The earliest instance of a universal category is due to Trnková [Trn66]. The universality of the category of semigroups was established by Hedrlín and Lambek [HL69]. These and a number of other such embeddings all took the form of a full and faithful functor that did not preserve underlying sets, for example representing some finite objects as infinite ones. The advantages accruing from the unifying framework of semigroups are then more than offset by the radically different discipline required to do mathematics in the absence of the expected underlying set.

Definition 4 For any ordinal κ, a κ-ary *relational structure* (X, ρ) consists of a set X, the *carrier*, and a κ-ary relation $\rho \subseteq X^\kappa$ on X. A *homomorphism* $f : (X, \rho) \to (Y, \sigma)$ between two such structures is a function $f : X \to Y$ between their underlying sets for which $f\rho \subseteq \sigma$. Here $f\rho$ denotes $\{f\mathbf{a} \mid \mathbf{a} \in \rho\}$, where $\mathbf{a}$ denotes $(a_0, \ldots, a_{\kappa-1})$ and $f\mathbf{a}$ denotes $(fa_0, \ldots, fa_{\kappa-1})$. We denote by $\mathbf{Str}_\kappa$ the category formed by the κ-ary relational structures and their homomorphisms. ∎

It suffices to treat structures with a single carrier and relation, since k carriers can be combined as their disjoint union, kept track of with k unary relations ($\lceil \log_2 k \rceil$ is enough information-theoretically, but not enough to ensure that homomorphisms respect type). Multiple nonempty relations on a set can be joined to form a single relation on the same set, of arity at most the sum of the arities of its constituent relations. For algebras, structures all of whose $(n+1)$-ary relations are n-ary operations, the join may share the input coordinates of the operations, reducing the total arity to the maximum of the input arities plus the number of operations (including constants).

This notion of homomorphism is standard in the strong sense that *any* class of n-ary relational structures and their homomorphisms constitutes a full subcategory of $\mathbf{Str}_\kappa$. Familiar examples of such categories and their arities include those of semigroups (3), monoids (4), groups (3), rings (4), rings with a multiplicative unit (5), fields (4), lattices (3), lattices with top and bottom (5), Boolean algebras (3), vector spaces (4),[8] directed graphs or binary relations (2), multigraphs (4), posets (2), and categories (4).

Many of these numbers benefit from group structure, for which homomorphisms preserve inverses and identities even when these operations are not given explicitly as part of the relation. Units of monoids, including tops and bottoms of lattices, are not so fortunate and each requires its own unary relation in order to be recognized and preserved by homomorphisms.

[8] Treat as partial rings, with uv defined just when u is on a specified axis. This works equally well for homogeneous vector spaces (all over the one field) and heterogeneous, the only nontrivial field endomorphisms being automorphisms.

The universality achieved here is of a different kind from that achieved by say ZF set theory. Externally a model of ZF is a single object of $\mathbf{Str}_2$ of some cardinality, with membership as its only relation, "internally" coding objects larger than any fixed cardinal including its own. Our universality has no separate notion of an internal world; instead we code our objects purely externally.

We now define the promised functor $F : \mathbf{Str}_\kappa \to \mathbf{Chu}_{\mathbf{2}^\kappa}$, namely in definitions 6 and 10, and prove that it is full, faithful, *and concrete.*

The complementarity of constraints and states indicates ρ and $\overline{\rho}$ as the appropriate respective sources of each. We shall define a state to be essentially an element of $\overline{\rho}$, with however a small but essential refinement. The following lemma obtains from the standard constraint-based definition of homomorphism an equivalent state-based characterization.

Lemma 5 $f\rho \subseteq \sigma \;\Leftrightarrow\; f^{-1}\overline{\sigma} \subseteq \overline{\rho}$. Here $\overline{\rho} = A^\kappa - \rho$ and $\overline{\sigma} = B^\kappa - \sigma$.

Proof:

$$\begin{array}{rcll} f\rho \subseteq \sigma & \Leftrightarrow & \rho \subseteq f^{-1}\sigma & \text{(Definition of } f^{-1}\text{)} \\ & \Leftrightarrow & \overline{f^{-1}\sigma} \subseteq \overline{\rho} & \text{(Complement)} \\ & \Leftrightarrow & f^{-1}\overline{\sigma} \subseteq \overline{\rho} & (f^{-1} \text{ preserves Boolean operations)} \end{array}$$

■

Definition 6 (F on objects). Let $\mathbf{2}^\kappa$ denote the set of κ-bit bit vectors, that is, κ-tuples over $\mathbf{2}$. We define the object part of the functor $F : \mathbf{Str}_\kappa \to \mathbf{Chu}_{\mathbf{2}^\kappa}$ as taking the κ-ary relational structure (A, ρ) to the Chu space $(X, \models, A)$ defined as follows. Take X to consist of those κ-tuples $x \in (2^A)^\kappa$ of subsets of A for which $\prod_{i<\kappa} x_i \subseteq \overline{\rho}$. Let $\models : X \times A \to 2^\kappa$ satisfy $\models (x,a)_i = 1$ if $a \in x_i$, and 0 otherwise. ■

Noting the isomorphism of $(2^\kappa)^A$ with $2^{\kappa\times A}$, we may equivalently think of X as follows. A κ-tuple over A is a function from κ to A. Define a κ-ruple (for *relational* tuple) over A to be a binary relation from κ to A. This makes the κ-tuple t the κ-ruple $\{(i,a) \mid t_i = a\}$. Then X consists of those κ-ruples over A extending no κ-ruple of ρ.

It might seem that X could be represented more naturally and conveniently as just the power set of $\overline{\rho}$. But observe that a state x as defined here can be recovered from the set $\prod_i x_i$ of its κ-tuples just when no component x_i is empty. The definition of $f^{-1} : Y \to X$ in Definition 10 below requires each x_i to be available independently even when some are empty.

The crucial test of whether $(X, \models, A)$ faithfully represents (A, ρ) is whether ρ can be recovered from it. We show this constructively as follows.

Lemma 7 *For all* $\mathbf{a} \in A^\kappa$, $\;\mathbf{a} \in \rho \;\Leftrightarrow\; \forall x{\in}X\; \exists i{<}\kappa : \models (x, a_i)_i = 0$.

Proof:

$$\begin{array}{llll} \mathbf{a} \in \rho & \Leftrightarrow & \forall x{\in}X : \mathbf{a} \notin \prod_i x_i & \text{(Construction of } X\text{)} \\ & \Leftrightarrow & \forall x{\in}X\ \exists i{<}\kappa : a_i \notin x_i & \text{(Definition of product)} \\ & \Leftrightarrow & \forall x{\in}X\ \exists i{<}\kappa : \models (x, a_i)_i = 0 & \text{(Construction of } \models\text{)} \end{array}$$

■

Corollary 8 *F is injective on objects.*

Lemma 9 *$(X, \models, A)$ is extensional.*

Proof: If row(x) = row(y) then $\forall i < \kappa[a \in x_i \Leftrightarrow a \in y_i]$, so $\forall i : x_i = y_i$, whence $x = y$. ■

Definition 10 (F on maps). Let $f : (A, \rho) \to (B, \sigma)$ be a homomorphism, with $F(A, \rho) = (X, \models, A)$ and $F(B, \sigma) = (Y, \models', B)$ as per Definition 6. Define $f^{-1} : (2^\kappa)^B \to (2^\kappa)^A$ to take $g : B \to 2^\kappa$ to $gf : A \to 2^\kappa$. Now for all $y \in Y$, $\prod_i y_i \subseteq \overline{\sigma}$ by construction of Y. Hence $\prod_i f^{-1} y_i \subseteq \overline{\rho}$, by Lemma 5. Thus $f^{-1}y \in X$ by construction of X. We may therefore define $F(f)$ as (f, f^{-1}) where $f^{-1} : Y \to X$. ■

Theorem 11 *The functor F of Definitions 6 and 10 is concrete, faithful, and full.*

Proof: F is concrete by construction, and *a fortiori* faithful.

For fullness consider any Chu transform $(f, g) : F(A, \rho) \to F(B, \sigma)$ where $F(A, \rho) = (A, X)$ and $F(B, \sigma) = (B, Y)$. If $\mathbf{a} \in \rho$, then for every $y \in Y$ there exists $i < \kappa$ such that

$$\begin{array}{rcll} \models (gy, a_i)_i & = & 0 & \text{(Lemma 7 with } x = gy\text{),} \\ \text{whence} \quad \models' (y, fa_i)_i & = & 0 & \text{((}f, g\text{) is a Chu transform).} \end{array}$$

Hence by Lemma 7, $f\mathbf{a} \in \sigma$, establishing that f is a homomorphism. And since $(X, \models, A)$ is extensional, by Lemma 9, g is determined by f. Hence $F(f) = (f, g)$. ■

Remarks. (i) Where size matters, X need contain only those states representable as the inverse image of a tuple of singletons. These can be characterized explicitly as those states x with the property that either $x_i = x_j$ or $x_i \cap x_j = \emptyset$ for all $i, j < \kappa$, observing that f^{-1} preserves this property. (ii) Lemma 9 is an inessential bonus. Had Definition 6 produced a nonextensional $(X, \models, A)$, we would simply have enforced extensionality, needed for fullness, by identifying those states having the same extension.

5 Heisenberg uncertainty in Chu spaces

We first discuss ways in which events and states interfere with each other from a lattice theoretic perspective, concentrating on $K = \mathbf{2}$. Passing from lattices to numbers, we treat uncertainty in Chu spaces as a purely quantitative phenomenon devoid of information-theoretic signicance; this holds for arbitrary K. We then relate the phenomenon to a natural model of information flow between Chu spaces.

5.1 Event-State Interference

Events as columns are made of the same bits as states as rows, so it stands to reason the two would interfere with other.

A simple case of interference is given by a pointed set, one with a constant column. The only possible constant row must be for the same constant. In particular for $K = \mathbf{2}$, if A has constant 0, $A^\perp$ cannot have constant 1.

A slightly more complex example is given by proper meets and joins. A meet or join is proper just when the result is not among the arguments.

Theorem 12 *A proper meet in $\mathcal{A}$ precludes some proper join in $\mathcal{A}^\perp$.*

Proof: Let $a \vee b$ be proper. Then there must exist rows x, y such that $x \models a$ and $y \models b$ but not $x \models b$ or $y \models a$. Hence $x \models (a \vee b)$ and $y \models (a \vee b)$. Now suppose $x \wedge y$ exists. Then neither $(x \wedge y) \models a$ or $(x \wedge y) \models b$ hold, and hence neither does $(x \wedge y) \models (a \vee b)$. But this contradicts $x \models (a \vee b)$ and $y \models (a \vee b)$. ■

Corollary 13 *If $\mathcal{A}$ has all meets then $\mathcal{A}^\perp$ has no proper joins.*

Proof: For if $\mathcal{A}^\perp$ had a proper join it would preclude some proper meet of $\mathcal{A}$. ■

Exercise: study this interference for infinite joins and meets of various cardinalities.

5.2 Quantitative aspects of uncertainty

Heisenberg's uncertainty principle is $\Delta p.\Delta q \geq \hbar$, the product of the uncertainties in momentum and position exceeds Planck's constant over 2π. This principle is sometimes explained in introductory lectures in terms of the Fourier transform. Here the natural units for a signal and its Fourier transform, namely seconds and Hertz for a signal distributed in time, force the constant $\hbar$ in this inequality to unity, a scaling that is also popular in high-energy particle physics.

The inequality appearing in the principle results from an application of the Schwartz inequality in its proof. Thus the intrinsic uncertainty in momentum of a particle cannot be inferred exactly from uncertainty in its position, which yields only a lower bound on momentum uncertainty.

For Chu spaces this inequality becomes an equality. Here however the reasoning is much simpler. We treat only finite Chu spaces. We define *absolute* uncertainty in state, or *information uncertainty*, to be the precision to which states can be identified, namely $1/|X|$. That is, a Chu space with only four states can specify its "exact" state only to within an uncertainty of $1/4$, i.e. two bits of precision. Likewise the absolute imprecision associated with knowledge of any event, or *temporal uncertainty*, is $1/|A|$. (There are no actual "exact states" distinct from the states in X, this is merely a convenient fiction for making sense of the concept that the elements of finite sets are specified only up to some precision.)

The bits in the matrix of a Chu space of a given size can be chosen independently. For a space with $n = |X| \times |A|$ bits one might guess that $\hbar$ should be 2^{-n}. However the product of information uncertainty $1/|X|$ with temporal uncertainty $1/|A|$ equals $1/n$. Our uncertainty principle must therefore take $\hbar$ to be $1/n$.

Now we customarily think of $\hbar$ as a universal constant, and it is disconcerting to have it depend on the size of the agents in an interaction. The following reasoning plausibly justifies the universal value $\hbar = 1$.

A Chu space having more events naturally has more states, with

$$\log_2(|X|) \leq |A| \leq 2^{|X|}$$

in the case of extensional T_0 spaces (no repeated rows or columns). "Balanced" Chu spaces have $|X| \approx |A|$, this being the situation when there are interesting and fruitful interactions between states and events. With this in mind we normalize to put all Chu spaces on the one comparable scale independent of their size, by defining the *relative* number of states to be $|X|/|A|$, and the relative number of events to be its reciprocal, $|A|/|X|$. This makes the *relative uncertainty* of a state $|A|/|X|$, and of an event $|X|/|A|$. The product of these uncertainties is 1, the uncertainty principle for Chu spaces.

From this point of view uncertainty resides only in the "form factor" of the Chu space and not its absolute size. Long low ones have more precise events, tall skinny ones more precise states.

5.3 Observation

We now justify the word "uncertainty" for these quantities in terms of the following model of information flow between Chu spaces. We briefly sketched this general idea in [Pra92b], but without the benefit of Chu spaces as a concrete model of it yielding the simple calculations of the previous section, a simplicity we had not expected to be possible in 1992. As then, the idea is to define a "message" between spaces A and B. In the present framework such a message becomes a Chu transform $f : A \to B$, as the basic notion of measurement or observation of A in the language of an abstract observer B. In the simplest nontrivial case, the observer is the two-element Boolean algebra $\perp$, which can record only a binary distinction, which it does for each point (event) of A, corresponding to a monochrome (black-and-white) image. This is the "picture" of A that B "gets."

This is a natural enough notion of message on its own, assuming we accept the idea of identifying the observer with the observation language. The following idea of "structure-induced veil," that smarter observers reveal less about themselves to any given observer, makes it an even more attractive notion by equipping the abstract calculations of the preceding section with an intuitively plausible meaning in terms of the information content of observations. The supporting principle for this connection is the identity $|A \multimap \perp| = |X_A|$, that is, the number of messages Chu space A can send to the canonical observer $\perp$ equals the number of states of A.

If A is a set (i.e. no structure), say of pixels on a computer screen, the messages it can send to $\perp$ are all possible black-and-white images. If however A has some structure, e.g. a linear ordering imposed on those pixels, the variety of possible messages can drop sharply; we then think of this additional structure as creating a sort of veil that defocuses the screen, making it less distinct. This gives a primitive model of the intuitively plausible idea that while one can see straight through an idiot, deeper thinkers are harder to understand.

As explained in the introduction, we have treated Chu spaces as blank canvases devoid of paint, analogous to the role of vector spaces in computer graphics as blank spaces ready to receive images. That is, Chu spaces as objects give the underlying geometry of behavior. A Chu transform as an observation amounts to a painting of the observed space with "colors" drawn from the points (events) of the observing space viewed as a palette. Using linear transformations to paint vector spaces does not work as well: vector spaces do not make good palettes, whereas Chu spaces provide a wide range of quality of palettes, from much worse than vector spaces (e.g. sets) to much better (e.g. Boolean algebras). Vector spaces fall exactly in the middle of this spectrum; indeed the n-dimensional vector space over $GF(2)$ is representable as a square (whence "in the middle") Chu space [LS91] with 2^n points and 2^n states (the dual points or functionals in the usual sense of vector spaces), with Chu transforms between vector spaces so represented being exactly their linear transformations.

The canonical choice of nontrivial palette B is the two-point one-state Boolean algebra $\perp = (\{0\}, \models, \{0,1\})$ for which $x \models a$ (where x is necessarily 0) is taken to be a itself. This has both a constant 0 and a constant 1, and hence permits *some* painting of every consistent or empty Chu space A (the inconsistent one-point space $(\emptyset, !, \{0\})$ is the only exception: it insists on painting its one point both 0 and 1, only possible with itself as the palette). Dually the canonical choice of canvas A is the one-element set, which permits some painting by every nonempty palette B (the empty palette is the dual of the inconsistent one-point space). (The choice of cardinals here is only to make these choices canonical, and does not affect the other claims.)

It follows from the isomorphism $A \multimap \perp \cong A^\perp$ (section 3.3) that the states of any Chu space are in 1-1 correspondence with its Chu transforms to $\perp$. This is analogous to the corresponding situation for topological spaces, where the open sets of a space are in 1-1 correspondence with its continuous functions to the two-point Sierpinski space defined as having exactly one open set that is a singleton (along with the empty set and whole space), which performs for topological spaces the function performed by $\perp$ for Chu spaces. In general, for any class requiring a given constant row, e.g. the empty set (constantly 0 row) and whole space (constantly 1 row) in the case of sets, posets, and topological spaces, the role of $\perp$ as a dualizing element is not impaired *for that class* when those constant rows are added to it; this is the Chu account of schizophrenia of dualizing objects. Exercise: further generalize what modifications to $\perp$ are possible for the purpose of dualizing, and use the generalization to explain why the two-element set is a dualizer for **Set**.

We can thus read off the opacity of $(X, \models, A)$ directly from X alone, at least when $\perp$ is the observer. This accounts for the relevance of $|X|$ to observation of A: it gives the number of observations of A that the canonical observer $\perp$ can distinguish between.

The relevance of $|A|$ is that this gives the number of observations of the conjugate of A, $A^\perp$, possible by $\perp$. Observing both A and $A\perp$ is analogous to observing the conjugate properties of position and momentum of a particle. Thus the abstract uncertainty principle derived in the preceding section is made real by identifying $|X|$ and $|A|$ with the precision with which canonical observer $\perp$ can observe respectively A and its conjugate $A^\perp$.

For other observers we do not as yet have an interesting story to tell. One problem that can arise is that some observers may be handicapped by an inadequate observation alphabet. We analyze general observers via the identity $A \multimap B \cong (A \otimes B^\perp)^\perp$. Here the number of messages observer B can receive from A is the number of states of the tensor product $A \otimes B^\perp$. For $B = \perp$ we have $\perp^\perp = \top$, the tensor unit in the sense that $A \otimes \top \cong A$, confirming the special case proved above.

But while there is a simple rule for the number of *points* in the tensor product $A \otimes B^\perp$, namely the product of the numbers of points in the arguments, there is no simple general rule for the number of states. One extreme is that when A is a set, i.e. $A = (2^A, A, \breve{\in})$, then $A \multimap B$ has $|B|^{|A|}$ elements. This expresses the fact that any Chu space can be used as the observation language, independently of its structure, when observing anything as naive as a set. (The same (large) extreme is reached by the dual situation in which $B = (Y, \models, B)$ is a Boolean algebra, where $A \multimap B \cong B^\perp \multimap A^\perp$, whence $A \multimap B$ has X^Y elements since $B^\perp =$

$(B, \models^{\smile}, Y)$ is a set.) A simple example of the other extreme is given by A having a constant 0 (or 1) and B not, in which case $A \multimap B$ is empty since Chu transforms must preserve constants. This is the situation where no measurement is possible because one of the variables (points) being measured always has a value that is missing from the observation alphabet (the empty alphabet is a degenerate case of this).

The general picture of communication between arbitrary Chu spaces then is that the amount of communicable information is highly dependent on the ability of the observer to understand the observed object.

6 Notes and Acknowledgements

Historical notes. Chu spaces are the case $V = \mathbf{Set}$ of the construction described by Po-Hsiang Chu in the appendix of Barr's book on *-autonomous (i.e. self-dual closed) categories [Bar79]. Chu's construction takes a closed monoidal category V with pullbacks and completes it to a self-dual category $\mathbf{Chu}(V, k)$. De Paiva [dP89a, dP89b] and Brown and Gurr [BG90, BGdP91] apply the Chu construction to respectively a version of Gödel's Dialectica and Petri nets. Lafont and Streicher study Chu spaces over K, which they call games [LS91]; the term "Chu construction" had been around previously, and the name "Chu space" for $V = \mathbf{Set}$ was suggested to us by Barr in email. Since Barr gave the basic construction to Chu in the first place it would be fair to be call them Chu-Barr spaces.

Acknowledgements. I must acknowledge first and foremost two years of productive collaboration with my student Vineet Gupta on the development of Chu spaces. Second, nearly a megabyte of email correspondence with Mike Barr on this and related topics has proved immensely insightful. More recently I have been collaborating with Gordon Plotkin, and with more time would have included some of his excellent insights in this paper. A two-month visit by Carolyn Brown, also working on Chu spaces, was very helpful. I also acknowledge useful conversations on the subject with Peter Freyd, Valeria De Paiva, Andreas Blass, Samson Abramsky, Rick Blute, and Robert Seely.

References

[Bar79] M. Barr. *$*$-Autonomous categories, LNM 752.* Springer-Verlag, 1979.

[Bar91] M. Barr. $*$-Autonomous categories and linear logic. *Math Structures in Comp. Sci.*, 1(2), 1991.

[BG90] C. Brown and D. Gurr. A categorical linear framework for Petri nets. In J. Mitchell, editor, *Logic in Computer Science*, pages 208–218. IEEE Computer Society, June 1990.

[BGdP91] C. Brown, D. Gurr, and V. de Paiva. A linear specification language for Petri nets. Technical Report DAIMI PB-363, Computer Science Department, Aarhus University, October 1991.

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

[DM60] A. De Morgan. On the syllogism, no. IV, and on the logic of relations. *Trans. Cambridge Phil. Soc.*, 10:331–358, 1860.

[dP89a] V. de Paiva. The dialectica categories. In *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*, pages 47–62, held June 1987, Boulder, Colorado, 1989.

[dP89b] V. de Paiva. A dialectica-like model of linear logic. In *Proc. Conf. on Category Theory and Computer Science, LNCS 389*, pages 341–356, Manchester, September 1989. Springer-Verlag.

[Dun86] J.M. Dunn. Relevant logic and entailment. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume III, pages 117–224. Reidel, Dordrecht, 1986.

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

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

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

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

[HL69] Z. Hedrlín and J. Lambek. How comprehensive is the category of semigroups. *J. Algebra*, 11:195–212, 1969.

[Isb72] J.R. Isbell. Atomless parts of spaces. *Math. Scand.*, 31:5–32, 1972.

[Joh82] P.T. Johnstone. *Stone Spaces.* Cambridge University Press, 1982.

[LS91] Y. Lafont and T. Streicher. Games semantics for linear logic. In *Proc. 6th Annual IEEE Symp. on Logic in Computer Science*, pages 43–49, Amsterdam, July 1991.

[Maz77] A. Mazurkiewicz. Concurrent program schemas and their interpretation. In *Proc. Aarhus Workshop on Verification of Parallel Programs*, 1977.

[Mil80] R. Milner. *A Calculus of Communicating Systems, LNCS 92.* Springer-Verlag, 1980.

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

[Pet62] C.A. Petri. Fundamentals of a theory of asynchronous information flow. In *Proc. IFIP Congress 62*, pages 386–390, Munich, 1962. North-Holland, Amsterdam.

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

[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*, pages 180–196, Pittsburgh, 1984. Springer-Verlag.

[Pra85] V.R. Pratt. Some constructions for order-theoretic models of concurrency. In *Proc. Conf. on Logics of Programs, LNCS 193*, pages 269–283, Brooklyn, 1985. Springer-Verlag.

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

[Pra90] V.R. Pratt. Dynamic algebras as a well-behaved fragment of relation algebras. In *Algebraic Logic and Universal Algebra in Computer Science, LNCS 425*, pages 77–110, Ames, Iowa, June 1988, 1990. Springer-Verlag.

[Pra92a] V.R. Pratt. Origins of the calculus of binary relations. In *Proc. 7th Annual IEEE Symp. on Logic in Computer Science*, pages 248–254, Santa Cruz, CA, June 1992.

[Pra92b] V.R. Pratt. Quantum logic, linear logic, and constructivity. In *Computation of Physics workshop: collected abstracts*, Dallas, October 1992. Superseded by published IEEE proceedings version, retitled "Linear Logic for Generalized Quantum Mechanics".

[Pra94] V.R. Pratt. A roadmap of some two-dimensional logics. In J. Van Eijck and A. Visser, editors, *Logic and Information Flow (Amsterdam 1992)*, pages 149–162, Cambridge, MA, 1994. MIT Press.

[PT80] A. Pultr and V. Trnková. *Combinatorial, Algebraic and Topological Representations of Groups, Semigroups, and Categories.* North-Holland, 1980.

[Sch95] E. Schröder. *Vorlesungen über die Algebra der Logik (Exakte Logik). Dritter Band: Algebra und Logik der Relative.* B.G. Teubner, Leipzig, 1895.

[See89] R.A.G Seely. Linear logic, $*$-autonomous categories and cofree algebras. In *Categories in Computer Science and Logic*, volume 92 of *Contemporary Mathematics*, pages 371–382, held June 1987, Boulder, Colorado, 1989.

[Slo73] N.J.A. Sloane. *A Handbook of Integer Sequences.* Academic Press, 1973.

[Sto36] M. Stone. The theory of representations for Boolean algebras. *Trans. Amer. Math. Soc.*, 40:37–111, 1936.

[Trn66] V. Trnková. Universal categories. *Comment. Math. Univ.Carolinae*, 7:143–206, 1966.

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

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