A LOGICAL INTERPRETATION OF POWERDOMAINS

Carl A. Gunter

MS-CIS-88-31
LINC LAB 111

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104

May 1988

Acknowledgements: This research was supported in part by DARPA grant NOOO14-85-K-0018, NSF grants MCS-8219196-CER, IRI84-10413-AO2 and U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027.

A Logical Interpretation of Powerdomains

Carl A. Gunter*
University of Pennsylvania

June 10, 1988

Abstract

This paper characterizes the powerdomain constructions which have been used in the semantics of programming languages in terms of formulas of first order logic under a pre-ordering of provable implication. The goal is to reveal the basic logical significance of the powerdomains by casting them in the right setting. Such a treatment may contribute to a better understanding of their potential uses in areas which deal with concepts of sets and partial information such as databases and artificial intelligence. Extended examples relating powerdomains to databases are provided. A new powerdomain is introducted and discussed in comparison with a similar operator from database theory. The new powerdomain is motivated by the logical characterizations of the three well-known powerdomains and is itself characterized by formulas of first order logic.

1 Introduction.

A *powerdomain* is a "computable" analogue of the powerset operator. They were introduced in the 1970's as a tool for providing semantics for programming languages with non-determinism. For such applications, the powerset operator was unsatisfactory for basically the same reasons that the full function space was unusable for the sematics of certain features of sequential programming languages (such as higher-order procedures and dynamic scoping). In the full powerset, there are *too many* sets and this causes problems for the solution of recursive domain equations. Hence, such applications call for a more parsimonious theory of subsets, based on a concept of non-deterministic computability.

The study of powerdomains has revealed many interesting connections between the semantics of programming languages and traditional topics of mathematical research in topology and category theory. Moreover, there is a widening awareness of the *logical* properties of powerdomains. It is the goal of this paper to prove several results intended to deepen our understanding of the logic of powerdomains. It is demonstrated that each of the best known powerdomains can be characterized by considering appropriate families of first order propositions under the pre-ordering of *provable implication.* These families provide a simple logical characterization of the information-theoretic content of the elements of the powerdomains. Such a view suggests methods for relating the known theory of powerdomains to work on similar structures which are the subject of investigations in other areas such as databases and artificial intelligence. A construction from database theory which is particularly similar to the ones studied in the semantics of programming languages will be discussed at the end.

The seminal work on powerdomains and their application in programming language semantics was G. Plotkin's paper [Plo76] on what is often called the *Plotkin powerdomain.* Subsequent research by

*Department of Computer and Information Sciences, Philadelphia, PA 19104 U.S.A. Supported in part by U.S. Army Research Office Grant DAAG29-84-K-0061.

M. B. Smyth [Smy78] led to the discovery of two similar constructions often called the *Smyth* and *Hoare* powerdomains. These three powerdomains have been used widely in programming language theory, and they have also sparked a body of theoretical research into their properties and relationships to similar constructions in Mathematics. Smyth [Smy83] demonstrated a close connection between the Smyth and Hoare powerdomains and the concepts of upper and lower semicontinuity respectively. He also found that the Plotkin powerdomain was related to what is known as the *Vietoris construction* from topology. This research led Smyth to suggest the names for the three powerdomains which I will use below: upper (Smyth), lower (Hoare) and convex (Plotkin). The categorical significance of the powerdomains was demonstrated by Hennessy and Plotkin [HP79] who proved that each of the three can be seen as a left adjoints to appropriate forgetful functors.

There has also been progress on understanding the powerdomains from the point of view of logic. Recent work by Abramsky [Abr87] has highlighted connections between domains, topology and logic. It will be illuminating to understand how powerdomains fit into this framework. The work most similar to what will be proved below is that of G. Winskel [Win85], who showed how each of the three powerdomains can be characterized using *modal* formulas under an interpretation in terms of non-deterministic computations. Winskel's results have a slightly different intuition from the ones proved below since I will generally be viewing powerdomains as partially described *sets* rather than partially described *computations.*

The paper is divided into four sections. The powerdomains are defined in the second section and an extended example using sets of records is discussed. In the third section the intuitions about information discussed in the second section are characterized using first order logic. Theorems establishing a precise relationship for the upper and lower powerdomains are proved. In the fourth section, the convex powerdomain is also characterized in terms of first order logic and a new powerdomain, the *mixed* powerdomain, is defined. Relationships between the convex and mixed powerdomains and the *sandwich* powerdomain from database theory are discussed. The mixed powerdomain is also characterized with first order formulas.

2 Sets of data.

This section begins by providing precise definitions for the upper, lower and convex powerdomains. As a guide to intuition, we will then look at several examples of sets from the the powerdomains of a simple datatype of records. Viewing things in such a concrete fashion aids one in seeing powerdomains as diverse theories of partially described sets and not just as a theories of the outcomes of non-determinitic computations.

Rather than follow the usual treatment which one can find in many places in the literature (see, for example, [Smy78] or [GS88]), I will reduce the domain-theoretic pre-requisites by working only with the action of the powerdomain operator on the *bases* of domains.[1] In this way, we may restrict our attention to the following simple class of directed graphs:

Definition: A *pre-order* is a set A together with a binary relation $\geq$ which is reflexive and transitive.

A pre-order is like a partial order except the anti-symmetry axiom need not hold. Intuitively, the elements of a pre-order A may be thought of as propositions (of first order logic, say) under the pre-ordering of provable implication. If we have propositions ϕ and ψ in A, then we may have $\phi \rightarrow \psi$ and $\psi \rightarrow \phi$ without it being the case that ϕ and ψ are the same proposition (although their truth values must be the same). For this reason and another (more important) reason mentioned below, it is more convenient to work with pre-orders than partial orders.

Let $\langle A, \geq \rangle$ be a pre-order and suppose $\mathcal{P}_f^* A$ is the collection of non-empty finite subsets of A. We define

[1] This way of doing things has been discussed in numerous references. The information systems of Scott [Sco82] are a popular tool; pre-orders and domains are discussed in some detail in [Gun87].

three pre-orderings on $\mathcal{P}^*_f A$ as follows. Suppose $u, v \in \mathcal{P}^*_f A$, then

- $u \geq^\sharp v$ iff for every $x \in u$ there is a $y \in v$ such that $x \geq y$,
- $u \geq^\flat v$ iff for every $y \in v$ there is a $x \in u$ such that $x \geq y$,
- $u \geq^\natural v$ iff $u \geq^\sharp v$ and $u \geq^\flat v$

It is easy to check that each of these relations is, in fact, a pre-ordering. The pre-order $\langle \mathcal{P}^*_f A, \geq^\sharp \rangle$ is called the *upper powerdomain* of $\langle A, \geq \rangle$ and it is denoted $\langle A^\sharp, \geq^\sharp \rangle$ (or just $A^\sharp$ when the pre-ordering is clear). The pre-order $\langle \mathcal{P}^*_f A, \geq^\flat \rangle$ is called the *lower powerdomain* of $\langle A, \geq \rangle$ and it is denoted $\langle A^\flat, \geq^\flat \rangle$. Finally, the pre-order $\langle \mathcal{P}^*_f A, \geq^\natural \rangle$ is called the *convex powerdomain* of $\langle A, \geq \rangle$ and it is denoted $\langle A^\natural, \geq^\natural \rangle$.

To get a few examples, let us look at the powerdomains of a simple pre-order of records. Our records will have between zero and four fields. the available fields are `name`, `age`, `socsec` and `married?`. The `age` and `socsec` fields may be filled with integers and the `married?` field may be filled with a boolean. The name field is a record with two fields: `first` and `second`. Each of these fields may be filled with a string. The type can be named by the following expression:

```
{ name = { first = string,
           last = string },
  age = int,
  socsec = int,
  married? = bool }
```

Here is a sample record r_1:

```
{ name = { first = "John",
          last = "Smith" },
  age = 28,
  socsec = 439048302,
  married? = true }
```

We will assume that records may have missing fields as in the following record r_2:

```
{ name = { first = "John" },
  age = 28 }
```

The record r_1 is *more informative* than r_2 because it provides more facts about the described individual "John". This concept of one record being more informative than another is basic to the discussion which follows. Records may have other relationships as well. In particular, there is an *inconsistency* between r_1, r_2 and the following record r_3:

```
{ name = { first = "John",
           last = "Smith" },
  socsec = 229068403,
  age = 2,
  married? = false }
```

We may model this collection of records and its associated information ordering as follows. First, we assume that we are given the types *string*, *int* and *bool* as *flat domains*. For example, the type of integers should contain the ordinary integers 1, -2, 0 and so on, together with a special *bottom* element $\perp$ which is intended to represent "no information". The ordering on these elements is given by taking $m \geq n$ if and only if $n = \perp$ or $m = n$. For example, we *do not* have $28 \geq 2$. This is what one would expect, after all; a record about a two year old John Smith is not less informative than a record about a 28 year old John Smith, these records are simply incompatible. The interpretation of strings is similar. The booleans are also a flat domain, but there are only three elements *true*, *false* and $\perp$. Now, the space of records is the product space

$$(\mathit{string} \times \mathit{string}) \times \mathit{int} \times \mathit{int} \times \mathit{bool}.$$

Of course, a record is interpreted in this space without regard to the order of its fields according to some convention (*e.g.* the first two strings are for the first and last names respectively; the first integer is the age and the second is the social security number). Missing record fields are interpreted as $\perp$. Records are ordered coordinate-wise. A pair of records r, r' is consistent if there is a record r'' such that $r'' \geq r$ and $r'' \geq r'$. Otherwise r and r' are inconsistent. Many of the sets in the powerdomain of our space of records will contain pairs of inconsistent records.

Our family of records is the raw material out of which we can build collections of data about some "real world entity". Some of our records probably make no real sense under any circumstances. For example:

```
{ name = { first = "John",
           last = "Smith" },
  age = 2,
  married? = true }
```

will probably not find its way into any useful database of records. There will also be pairs of records which are unlikely to be found together in the same database:

```
{ socsec = 229068403,
  age = 2 }

{ socsec = 229068403,
  age = 28 }
```

And most data items will be only partial descriptions (as is the case with most of the examples above). The question we need to answer is the following: how does a *set* of records provide a partial description of a real world entity?

Consider the following set s of records

```
{ name = { first = "Mary" },
  age = 2 }

{ name = { first = "Todd" },
  age = 2 }

{ name = { first = "John" },
  age = 2 }
```

which might be the database for a small nursery. When should we say of another set of records that it is *more informative* than the set of records above? Here is a first possibility s_1:

```
{ name = { first = "Mary" },
  age = 2 }

{ name = { first = "Todd" },
  age = 2 }

{ name = { first = "John",
           last = "Smith" },
  age = 2 }

{ name = { first = "Beth" }
  age = 3 }
```

This set seems more informative because it lists more of the children in the nursery and provides slightly more information about those who are enrolled (since we now have John's last name). In the lower powerdomain (pre)-ordering, $\geq^{\flat}$, the set s_1 is greater (more informative) than s. But consider the following set s_2 of records:

```
{ name = { first = "Mary" },
  socsec = 4392349703,
  age = 2 }

{ name = { first = "John",
           last = "Smith" },
  socsec = 429238406,
  age = 2 }

{ name = { first = "John",
           last = "Smith" },
  socsec = 229068403,
  age = 2 }
```

This seems more informative than s because it provides more information about the children in the class and eliminates the name of a child (Todd) who will not actually be attending. In the upper powerdomain ordering,

$\geq^\natural$, the set s_2 is greater than s. However, it is *not* greater than s in the lower powerdomain ordering. Conversely s_2 is not greater than s in the upper powerdomain ordering.

These two alternative extensions should point out how the ordering of partial information suggests the intuitive significance of the set of records s. In the first case, under the lower ordering, s might be a list of children who have been enrolled in the nursery; more may enroll later. In the second case (under the upper ordering) s might be the list of all children who are on a waiting list; some children may drop off of the list but no new ones may enter (since the deadline for such entries has passed). In either case, a further refinement of the individual records through the addition of new fields results in a more informative set of records.

It is important to note that powerdomains are only pre-orderings and not partial orderings. If the record

```
{ name = { last = "Smith" }
  age = 2 }
```

is added to s_1, there is *no change* in the intended meaning of the set of records with respect to the lower pre-ordering. In other words, if s_1' is the larger set, then $s_1 \geq^\flat s_1'$ and also $s_1' \geq^\flat s_1$. This is not true of the upper pre-ordering. In that pre-ordering, $s_1 \geq s_1'$, but $s_1' \not\geq s_1$. The following set of records

```
{ name = { first = "John",
           last = "Smith" },
  socsec = 229068403,
  age = 2 }

{ name = { first = "John" }
  age = 2 }
```

would not change, under either powerdomain ordering, if the following record were added:

```
{ name = { first = "John",
           last = "Smith" }
  age = 2 }
```

It may seem odd that we would allow in s_2 the possibility that a single record might split into two records as the record for John did. This seems more reasonable in other cases, however. For example, the singleton set of records containing only the record

```
{ age = 2 }
```

would indicate under the upper ordering that we are talking about a nursery of two year olds (whose names we do not yet know). In the lower ordering, this database would indicate only that there will be *some* two year old in the nursery (but there may also be some children of other ages). It is also possible for two data items to *merge* to form a new data item. For example, the following set of records:

```
{ name = { first = "Mary" } }

{ name = { first = "John" },
  age = 2 }

{ name = { last = "Smith" }
  socsec = 229068403 }

{ socsec = 429238406 }
```

is less descriptive (in either lower or upper ordering) than the set of records s_2 above.

We will look at some more examples of this kind when we get to the discussion of the convex ordering in a later section.

3 Powerdomains and logic.

Let us now try to relate the intuitions and pre-orderings discussed in the previous section to formulas of an appropriate logic. For this discussion first order preidcate logic will be used because it is simple, well-known and seems to be sufficient for the job at hand. After some motivation, the upper and lower powerdomain operators on pre-orders will be precisely related to certain operations on collections of first order formulas.

In the examples provided in the previous section, we thought of sets of records as describing a real world entity partially. However, one may dually think of a set of records as describing a *set* of "worlds" compatible with the set of records. Each record can be treated as a predicate over a collection of *individuals*. For example, the record

```
{ name = { first = "John" } }
```

is satisfied by individuals whose first name is "John". More concretely, we might think of individuals as *total records* (*i.e.* records with all fields filled in) for the example of the previous section.[2] If we view things this way, can we think of sets of records as predicates too? First of all, we must ask what is being predicated by a set of records. The answer seems clear: sets of individuals. Hence, a set of records should be considered a predicate over sets of individuals or, put succinctly, a second order predicate. [3]

This seems to justify a leap into second order logic for a description of powerdomains. We expect to find that the different powerdomain orderings give rise to different second order predicates. However, a *first* order formula may be considered a second order predicate if it contains a unary predicate symbol. Suppose we are given a distinguished unary predicate symbol W and a collection of predicate symbols U. In a given model, a formula like $U(x)$ might be asserting that x is a two year old. With this interpretation, a first order formula such as

$$\phi \equiv \forall x.\ W(x) \rightarrow U(x)$$

asserts that everyone in the interpretation of W is a two year old. Hence ϕ itself becomes a predicate of W. Of course, there will be many predicates defined by first order formulas in this way, but which of them (if any) correspond to the elements of the powerdomains?

Let us attempt to work out an example similar to those in the previous section. Recall the set s of records:

```
{ name = { first = "Mary" },
  age = 2 }

{ name = { first = "Todd" },
  age = 2 }

{ name = { first = "John" },
  age = 2 }
```

Let M, T and J be unary predicate symbols for having first name "Mary", "Todd" and "John" respectively. Under the lower powerdomain ordering, what is this collection of records telling us about the set of children in our hypothetical nursery? The first record of s seems to assert that *there is a child named "Mary" in the nursery.* If W is a predicate symbol which we are interpreting as the children in the nursery, this can be represented by the formula

$$\exists x.\ W(x) \wedge M(x)$$

which we may express more succinctly as $W \cap M \neq \emptyset$. Actually, the first record expresses a bit more than this. Let O be a predicate which is being interpreted as the set of all two year olds. Then the first record says: $W \cap M \cap O \neq \emptyset$. In summary, s corresponds to the following proposition:

$$\begin{array}{l} W \cap (M \cap O) \neq \emptyset \wedge \\ W \cap (T \cap O) \neq \emptyset \wedge \\ W \cap (J \cap O) \neq \emptyset \end{array}$$

As an exercise, the reader may express s_1 in this way and show that the resulting proposition implies the one above.

Now, what about the upper powerdomain ordering? Under this ordering, each record expresses a range of possiblities. The three records together assert that the children of the nursery (or those on its waiting list if that is prefered interpretation) are all named "Mary", "Todd" or "John". More specifically, a child on the waiting list must be a two year old "Mary", a two year

[2] It will not always be intuitively reasonable to view things in this way, although it works well for the example at hand.

[3] If necessary, the reader should consult a basic logic book for the elementary facts and definitions which will be used in this paper. Some good sources include [Bar77] and [BJ80].

old "Todd" or a two year old "John". However, this does not perclude the possibility that there is no "Todd" who is acually waiting for entry. If W is a new unary predicate symbol to be interpreted as the individuals in the nursery, then this assertion may be summarized as

$$(1) \qquad \forall x.\ W(x) \to \theta$$

where θ is the disjunction

$$(M(x) \wedge O(x)) \vee (T(x) \wedge O(x)) \vee (J(x) \wedge O(x)).$$

The formula (1) may also be expressed with set-theoretic notation:

$$W \subseteq (M \cap O) \cup (T \cap O) \cup (J \cap O).$$

Again, the reader may find it instructive to espress s_2 in this way and check that the resulting proposition implies this one.

It is tempting, at this point, to "think semantically" and try to view the powerdomains in terms of sets of individuals. This can be misleading, however. Given a predicate symbol U, let $[\![U]\!]$ be the interpretation of U in a fixed model. In particular, for the upper ordering, we may have

$$[\![U_1]\!] \cup \cdots \cup [\![U_m]\!] = [\![V_1]\!] \cup \cdots \cup [\![V_n]\!]$$

without it being the case that the $[\![U_i]\!] \subseteq [\![V_j]\!]$ or $[\![V_j]\!] \subseteq [\![U_i]\!]$ for *any* pair of predicate symbols U_i and V_j. It seems, therefore, that although the formulas

$$\phi \equiv W \subseteq U_1 \cup \cdots \cup U_n$$

and

$$\psi \equiv W \subseteq V_1 \cup \cdots \cup V_m$$

define the same family of predicates, this does not follow from the ordering under inclusion of the sets $[\![U]\!]$ for unary predicate symbols U of the language. For a *fixed* model, the interpretations of the predicates $\phi(W)$ and $\psi(W)$ seem to have more relationships than one can "obtain" from the ordering of the sets $[\![U]\!]$. One may place some *ad hoc* assumptions on the model to make things work out better. However, I claim that the use of *non-standard models* will make it possible to capture the idea more naturally.

To crystalize this discussion by proving some theorems, it is necessary to be somewhat more formal about the ground rules. Some notation is helpful. Fix a first order language $\mathcal{L}$ of unary predicate symbols and a set T of formulas of the form $U \subseteq V$ where U and V are unary predicates in the language. Given a set of formulas Φ, the theory T *induces* a pre-ordering on the formulas of Φ by provable implication. In other words, the induced pre-order has, as its elements, formulas $\phi \in \Phi$ and it is pre-ordered by taking $\phi \geq \phi'$ iff $T \vdash \phi \to \phi'$. For the remainder of this paper, fix the theory T and assume that W is a new unary predicate symbol not in the language of T. It will simplify matters to assume that $U \subseteq V$ is in T whenever $T \vdash U \subseteq V$. Let A be the pre-order which T induces on formulas of the form $U(x)$ where U is a unary predicate symbol of $\mathcal{L}$. Then we have the following:

Theorem 1 *The pre-order which T induces on formulas of the form*

$$W \subseteq U_1 \cup \cdots \cup U_n$$

is exactly the upper powerdomain of A.

Proof: Suppose we are given formulas

$$\phi \equiv W \subseteq U_1 \cup \cdots \cup U_n$$
$$\psi \equiv W \subseteq V_1 \cup \cdots \cup V_m$$

It is not at all difficult to see that if, for each predicate U_i, there is predicate V_j such that $U_i \subseteq V_j$ is in the theory T, then

$$T \vdash \phi \to \psi.$$

What is less obvious is the fact that this is *the only way* such an implication can be proved. By the Completeness Theorem for first order logic, it suffices to show that if

$$(2) \qquad T \models \phi \to \psi$$

then, for each predicate U_i, there is predicate V_j such that $U_i \subseteq V_j$ is in the set T. Suppose that (2) holds, but there is a predicate U_i such that $U_i \subseteq V_j$ is not in

T for any V_j. We demonstrate a contradiction. Define a model $\mathcal{A}$ of $T \cup \{\phi\}$ as follows. The universe of $\mathcal{A}$ is the set of predicate symbols of $\mathcal{L}$ (this does not include W). If U is a predicate symbol of $\mathcal{L}$, it is interpreted in $\mathcal{A}$ as the set of predicate symbols $V \in \mathcal{L}$ such that $U \subseteq V$ is in T. The predicate symbol W is interpreted as the set $\{U_1, \ldots, U_n\}$. Let $\llbracket U \rrbracket$ be our notation for the interpretation of a predicate symbol U. I claim that $\mathcal{A} \models T \cup \{\phi\}$. If $U \subseteq V$ is in T and $U' \in \llbracket U \rrbracket$, then $U' \subseteq U$ is in T so $U' \subseteq V$ is in T. Thus $U' \in \llbracket V \rrbracket$ and it follows that $\llbracket U \rrbracket \subseteq \llbracket V \rrbracket$ as desired. That $\mathcal{A} \models \phi$ follows immediately from the interpretation of W. On the other hand, I also claim that $\mathcal{A} \not\models \psi$. Since there is no V_j such that $U_i \subseteq V_j$ is in T, the element U_i is not in $\llbracket V_1 \rrbracket \cup \cdots \cup \llbracket V_m \rrbracket$ and therefore $W \not\subseteq \llbracket V_1 \rrbracket \cup \cdots \cup \llbracket V_m \rrbracket$. ∎

Theorem 2 *The pre-order which T induces on formulas of the form*

$$(W \cap U_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U_n \neq \emptyset)$$

is exactly the lower powerdomain of A.

Proof: Define formulas

$$\begin{aligned} \phi' &\equiv (W \cap U_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U_n \neq \emptyset) \\ \psi' &\equiv (W \cap V_1 \neq \emptyset) \wedge \cdots \wedge (W \cap V_m \neq \emptyset) \end{aligned}$$

If, for each V_j there is a predicate U_i such that $U_i \subseteq V_j$ is in T, then it is easy to show that

$$T \vdash \phi' \rightarrow \psi'$$

As before, the harder part is showing that the converse of this assertion holds. By the Completeness Theorem for first order logic, it suffices to show that if

$$(3) \qquad T \models \phi' \rightarrow \psi'$$

then, for each predicate V_j, there is a predicate U_i such that $U_i \subseteq V_j$ is in the set T. Suppose that (3) holds, but there is a predicate V_j such that $U_i \subseteq V_j$ is not in T for any U_i. I will demonstrate a contradiction. Let $\mathcal{A}$ be the model of T given in the proof of Theorem 1. Obviously $\mathcal{A} \models \phi'$. However, $\llbracket V_j \rrbracket \cap \llbracket W \rrbracket$ is the emptyset since there is no U_i in $\llbracket V_j \rrbracket$. ∎

4 Other powerdomains?

In this section I will look at a few more second order predicates such as the ones which were used to characterize the upper and lower powerdomains in the previous section. I begin by discussing the convex ordering and its information-theoretic significance using sets of records. A logical characterization of the convex powerdomain is then provided and a correspondence theorem similar to Theorems 1 and 2 will be given. I will then define a close relative of the *sandwich powerdomain* of Buneman, Davidson, Ohori and Watters [BDW88,BO86,BO88] which has been used used for the semantics of databases.

Under the convex ordering, *none* of the three sets of records s, s_1, s_2 given earlier are related. The following set s_3 satisfies $s_3 \geq^{\natural} s$:

```
{ name = { first = "Mary" },
  socsec = 4392349703,
  age = 2 }

{ name = { first = "Todd",
           last = "Smith" },
  socsec = 923799210,
  age = 2 }

{ name = { first = "John",
           last = "Smith" },
  socsec = 429238406,
  age = 2 }

{ name = { first = "John",
           last = "Smith" },
  socsec = 229068403,
  age = 2 }
```

Note that no new names were added in s_3 as we added the name “Beth” in s_1 (although the two John Smith’s were disambiguated), and no names were removed from s as we removed “Todd” in s_2. On the other hand, the records of s_3 are considerably more specific than those in s. For example, if we assume that now two children have the same social security number, then no further refinement of s_3 will have more or less than four children. (However, sets with multiple names associated with the same social security number are permited in the convex powerdomain.) As with the other powerdomains, it is easy to produce examples which show that the convex powerdomain of a partial order may not satisfy anti-symmetry. The following can be proved by combining the proofs of Theorems 1 and 2:

Theorem 3 *The pre-order which T induces on formulas of the form*

$$\begin{array}{l}(W \subseteq U_1 \cup \cdots \cup U_n) \wedge \\ (W \cap U_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U_n \neq \emptyset)\end{array}$$

is exactly the convex powerdomain of A. ▮

The convex powerdomain is generally considered to be more "natural" than the upper and lower powerdomains; this view is supported, for example, by the categorical characterizations of the three powerdomains [HP79,GS88] as well as considerations from the semantics of concurrency. However, when one views the three powerdomains from the standpoint of this paper, the convex powerdomain seems to entail a peculiar assumption. Each of the records in a database under the convex ordering must convey *both* upper and lower information; or, to put it another way, the upper and lower information conveyed by the database must be conveyed by the *same* set of predicates. We are permited to use formulas of the form

$$\begin{array}{l}(W \subseteq U_1 \cup \cdots \cup U_n) \wedge \\ (W \cap U_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U_n \neq \emptyset)\end{array} \tag{4}$$

but not formulas of the the more general form

$$\begin{array}{l}(W \subseteq U_1 \cup \cdots \cup U_m) \wedge \\ (W \cap U'_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U'_n \neq \emptyset)\end{array} \tag{5}$$

While it makes perfectly good sense to make a restriction to formulas as in (4), it also seems reasonable, in some circumstances, *not* to make this restriction. The use of formulas such as those in (5) in the theory of databases has been discussed in several publications [BDW88,BO86,BO88] using an operator known as the *sandwiches powerdomain.* Although questions about the categorical and topological significance of sandwiches are only beginning to be investigated, their information-theoretic significance and potential applications suggest interesting lines of investigation. I now define an operator which has a strong kinship to the sandwiches domain and demonstrate a logical characterization for it.

Definition: Suppose $\langle A, \geq \rangle$ be a pre-order. Let

$$A^{(\sharp,\flat)} = \{(u,v) \in \mathcal{P}^*_f A \times \mathcal{P}^*_f A \mid v \geq^\sharp u\}$$

and define $(u,v) \geq^{(\sharp,\flat)} (u',v')$ iff $u \geq^\sharp u'$ and $v \geq^\flat v'$. Let us refer to $\langle A^{(\sharp,\flat)}, \geq^{(\sharp,\flat)} \rangle$ as the *mixed powerdomain.* ▮

The choice of pre-ordering on the pairs $(u,v) \in A^{(\sharp,\flat)}$ is unsurprising. It is slightly less clear why only pairs (u,v) with $v \geq^\sharp u$ are used. To understand this restriction and get a feeling for the mixed powerdomain, it is best to look at some examples. Rather than representing elements of the mixed powerdomain with a pair of sets of records it is convenient to write a set of records which are *tagged* to indicate whether they belong in the first or second coordinate of the pair. I will use a tag # for the records in the first coordinate (since this looks like the ♯ sign) and a tag b for records in the second coordinate (since this looks like a ♭ sign). Forget, for the moment, about the condition that $v \geq^\sharp u$ and consider the following set of (tagged) records t:

```
b{ name = { first = "Mary" } }

b{ name = { first = "Todd" } }

b{ name = { first = "John" } }

#{ age = 2 }
```

This is very similar in information content to the set of records s which were considered earlier. It describes a group of two year olds which must include a "Mary", a "Todd" and a "John". Here is another set of records t_1 similar to s_1:

```
b{ name = { first = "Mary" },
   age = 2 }

b{ name = { first = "Todd" },
   age = 2 }

b{ name = { first = "John",
            last = "Smith" },
   age = 2 }

b{ name = { first = "Beth" }
   age = 3 }

#{ age = 2 }

#{ age = 3 }
```

which allows that the nursery is now enrolling three year olds as well as two year olds. However, the following set of records is *nonsense:*

```
b{ name = { first = "Mary" },
   age = 2 }

b{ name = { first = "Todd" },
   age = 2 }

b{ name = { first = "John",
            last = "Smith" },
   age = 2 }

b{ name = { first = "Beth" }
   age = 3 }

#{ age = 2 }
```

because Beth is incorrectly recorded as a three year old or the new admissions policy has not be properly entered. In order for a set of mixed records such as these to make sense, it is essential that, for each ♭-record, there is a ♯-record which applies to it. Otherwise, the set of mixed records is "insecure." As another example, a dating service may have a database d:

```
b{ name = { first = "Sharon" },
   age = 26,
   married? = false}

b{ name = { first = "David" },
   age = 28,
   married? = false}

b{ name = { first = "Mabel" },
   age = 58,
   married? = false}

b{ name = { first = "Lee" },
   age = 55,
   married? = false}

#{ married? = false }
```

but trouble may arise from adding a record such as

```
b{ name = { first = "John" }
   age = 30,
   married? = true }
```

The sandwiches powerdomain is defined to include records like t above; t is not in the mixed powerdomain because the ♭-records are missing their age fields. A *sandwich* is a pair

$$(u,v) \in \mathcal{P}^*_f A \times \mathcal{P}^*_f A$$

such that there is a set $w \in \mathcal{P}^*_f A$ such that $w \geq^\sharp u$ and $w \geq^\flat v$. Obviously, any element of the mixed powerdomain is a sandwich. Unfortunately, the logical interpretation of the sandwich powerdomain in the sense of this paper does not seem to be straight-forward.

To characterize the mixed powerdomain logically, it is necessary to generalize from formulas such as (4) to formulas such as (5) and add an assumption about insecurity. Recall that T is a set of formulas of the form $U \subseteq V$ where U and V are unary predicates in a fixed first order language $\mathcal{L}$. A is the pre-order which T induces on formulas of the form $U(x)$ where U is a unary predicate symbol of $\mathcal{L}$.

Definition: A formula of the form

$$\begin{array}{l}(W \subseteq U_1 \cup \cdots \cup U_m) \wedge \\ (W \cap U'_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U'_n \neq \emptyset)\end{array}$$

is *secure* (with respect to T) if, for every predicate symbol U'_i, there is a predicate symbol U_j such that $U'_i \subseteq U_j$ is in T. ▌

Theorem 4 *The pre-order which T induces on secure formulas is exactly the mixed powerdomain of A.*

Proof: Suppose we have formulas

$$\begin{array}{l}\phi \equiv W \subseteq U_1 \cup \cdots \cup U_m \\ \phi' \equiv (W \cap U'_1 \neq \emptyset) \wedge \cdots \wedge (W \cap U'_n \neq \emptyset) \\ \psi \equiv W \subseteq V_1 \cup \cdots \cup V_p \\ \psi' \equiv (W \cap V'_1 \neq \emptyset) \wedge \cdots \wedge (W \cap V'_q \neq \emptyset)\end{array}$$

We must show that

$$T \vdash (\phi \wedge \phi') \rightarrow (\psi \wedge \psi') \tag{6}$$

if and only if

1. for each U_i, there is a V_j such that $U_i \subseteq V_j$ is in T, and
2. for each V'_j, there is a U'_i such that $U'_i \subseteq V'_j$ is in T.

As with the earlier proofs of this kind, the harder part of the proof is showing that (6) implies items (1) and (2). As before, we utilize the Completeness Theorem to prove each of these items by contradiction. Define a model $\mathcal{A}$ of $T \cup \{\phi, \phi'\}$ as follows. The universe of $\mathcal{A}$ is the set of predicate symbols of $\mathcal{L}$ (this does not include the distinguished predicate symbol W). If U is a predicate symbol of $\mathcal{L}$, it is interpreted in $\mathcal{A}$ as the set of predicate symbols $V \in \mathcal{L}$ such that $U \subseteq V$ is in T. The predicate symbol W is interpreted as the set $\{U_1, \ldots, U_m, U'_1, \ldots, U'_n\}$. That $\mathcal{A}$ is a model of $T \cup \{\phi, \phi'\}$ follows from the fact that $\phi \wedge \phi'$ is secure. Now, suppose that (1) *fails*. Then there is some U_i such that $U_i \notin \llbracket V_1 \rrbracket \cup \ldots \cup \llbracket V_p \rrbracket$. Since $U_i \in \llbracket W \rrbracket$, it follows that $\llbracket W \rrbracket \not\subseteq \llbracket V_1 \rrbracket \cup \ldots \cup \llbracket V_p \rrbracket$ and therefore $\mathcal{A}$ does not satisfy ψ. Suppose that (2) *fails*. Then there is some V'_j such that $U'_i \notin \llbracket V'_j \rrbracket$ for each V'_j. To get the desired contradiction, we want to use a new model $\mathcal{A}'$ which is the same as $\mathcal{A}$ except $\llbracket W \rrbracket = U'_1, \ldots, U'_n$. Since the formula $\phi \wedge \phi'$ is secure, $\mathcal{A}' \models T \cup \{\phi, \phi'\}$. But $\llbracket V'_j \rrbracket \cap \llbracket W \rrbracket = \emptyset$ so $\mathcal{A}' \not\models \psi'$. ▌

The existence of sensible and potentially useful operators such as the mixed powerdomain and the sandwich powerdomain compel one to ask what are the ground rules and limits of the game. When does an operator qualify as a "powerdomain"? The upper, lower and convex powerdomains have been deeply related to known mathematical theories and they have played an important role in the semantics of programming languages. Are the mixed and sandwich powerdomains also deep concepts or are they an *ad hoc* inventions which supports a few shallow examples? It is possible to show that the mixed and sandwich powerdomains have many of the basic mathematical properties which make the other powerdomains useful in semantics. For example, it is possible to use the sandwich powerdomain in recursive domain equations and it could be used to provide a semantics for concurrency (although it does not seem to work as well as the convex powerdomain for this purpose). In some ways it is even nicer than the convex powerdomain. Assume for simplicity that pre-orders are quotiented so they become partial orders. It is well-known that the convex powerdomain of a lower semi-lattice (*i.e.* a poset with finite meets and a least element) may not be a lower semi-lattice. However, the sandwich powerdomain of a lower semi-lattice *is a lower semi-lattice!* To see this, note that both the upper and lower powerdomains preserve lower semi-lattices and the coordinate-wise meet in the mixed powerdomain preserves security. Since obtaining this property is a primary goal of research into the powerdomain of Hrbacek [Hrb85], it may provide an alternative theory.

A logical theory of powerdomains may help us to relate some of the diverse constructions that are being considered in the current literature. I hope that the results in this paper will provide some hints for the development of such a theory.

References

[Abr87] S. Abramsky. Domains in logical form. In D. Gries, editor, *Symposium on Logic in Computer Science*, pages 47–53, IEEE Computer Society Press, Ithaca, New York, June 1987.

[Bar77] J. K. Barwise. *Handbook of Mathematical Logic*. Volume 90 of *Studies in Logic and the Foundations of Mathematics*, North-Holland, 1977.

[BDW88] P. Buneman, S. Davidson, and A. Watters. A semantics for complex objects and approximate queries. In *Principles of Database Systems*, ACM, March 1988.

[BJ80] G. S. Boolos and R. C. Jeffrey. *Computability and Logic*. Cambridge University Press, second edition, 1980.

[BO86] P. Buneman and A. Ohori. A domain theoretic approach to higher-order relations. In *International Conference on Database Theory*, Springer-Verlag, 1986.

[BO88] P. Buneman and A. Ohori. Using powerdomains to generalize relational databases. *Theoretical Computer Science*, 1988. To appear.

[GS88] C. A. Gunter and D. S. Scott. Semantic domains. In J. van Leeuwen, editor, *Handbook ofTheoretical Computer Science*, North Holland, To appear in 1988.

[Gun87] C. A. Gunter. Universal profinite domains. *Information and Computation*, 72:1–30, 1987.

[HP79] M. Hennessy and G. D. Plotkin. Full abstraction for a simple parallel programming language. In J. Bečvář, editor, *Mathematical Foundations of Computer Science*, pages 108–120, *Lecture Notes in Computer Science vol. 74*, Springer, 1979.

[Hrb85] K. Hrbacek. Powerdmains as algebraic lattices. In W. Brauer, editor, *International Colloquium on Automata, Languages and Programs*, pages 281–289, *Lecture Notes in Computer Science vol. 194*, Springer, June 1985.

[Plo76] G. D. Plotkin. A powerdomain construction. *SIAM Journal of Computing*, 5:452–487, 1976.

[Sco82] D. S. Scott. Domains for denotational semantics. In M. Nielsen and E. M. Schmidt, editors, *International Colloquium on Automata, Languages and Programs*, pages 577–613, *Lecture Notes in Computer Science vol. 140*, Springer, 1982.

[Smy78] M. Smyth. Power domains. *Journal of Computer System Sciences*, 16:23–36, 1978.

[Smy83] M. Smyth. Power domains and predicate transformers: a topological view. In J. Diaz, editor, *International Colloquium on Automata, Languages and Programs*, pages 662–676, *Lecture Notes in Computer Science vol. 154*, Springer, 1983.

[Win85] G. Winskel. On powerdomains and modality. *Theoretical Computer Science*, 36:127–137, 1985.