REVISTA DE LA
UNIÓN MATEMÁTICA ARGENTINA
Volumen 49, Número 1, 2008, Páginas 83–98

THE DUALITY BETWEEN ALGEBRAIC POSETS AND BIALGEBRAIC FRAMES: A LATTICE THEORETIC PERSPECTIVE

JAMES B. HART AND CONSTANTINE TSINAKIS

Abstract. This paper sets two goals. The first is to present algebraists with a purely order-theoretic derivation of the adjunction between the category $\underline{\mathtt{DCPO}}$ of DCPOs (directed complete posets) and the category $\underline{\mathtt{Frm}}$ of frames. This adjunction restricts to several Stone-type dualities which are well-known and of considerable interest to computer scientists. The second goal is to describe the object classes of these subdualities in terms familiar to algebraists, thereby making a large body of literature about them more accessible.

1. Introduction

Since their introduction in the pioneering work of D. S. Scott and C. Strachey, the appealing properties of the posets used in denotational semantics have been expanded by logicians and theoretical computer scientists into a rapidly growing discipline called *domain theory.* The richness of this new field is attracting increasing numbers of mathematicians (present authors included) whose interests have previously lain outside of theoretical computer science. This paper is written with these people in mind. There are few survey papers or monographs devoted exclusively to the mathematical theory of domains, the recent monograph by Abramsky and Jung [2] and the text by Vickers [28] being exceptions. The beginner not versed in computer science who attempts to understand the theory is often confronted by a maze of interconnected, alternative approaches made more complicated by seemingly foreign literature and folklore. Our aim in this paper is to provide the reader, the algebraist in particular, with some familiar ground from which the primary concepts of the field may be studied and appreciated.

We accomplish this aim in three steps. First, in Section 2, we provide the reader with a detailed account of the motivations behind the use of the most general posets appearing in domain theory and then, in Section 3, present a purely order-theoretic derivation of one of the field's key features: the contravariant adjunction between the categories $\underline{\mathtt{DCPO}}$ and $\underline{\mathtt{Frm}}$. Second, in Section 4, we restrict the primary adjunction and use mostly lattice-theoretic techniques to obtain the important Stone-type duality between algebraic posets and bialgebraic frames and, in Section 5, further restrict this duality to categories of particular interest to denotational semantics. Third, in Section 6, we provide an example of the relevance to algebraists of the objects studied in domain theory by showing that the object assignments of the dualities in Section 5 yield new insights into the structure of the prime spectrum of distributive lattices.

Most of the results of this paper have appeared elsewhere. The novelty lies in the approach taken to their proof. The categorical correspondences considered are usually derived from a topological viewpoint (see, for example, Abransky and Jung [2], Abramsky [1], or Mislove [15]) and trace their lineage to the approach in Lawson [12]. The standard approach masks the underlying order-theoretic nature of the results; in this paper we make this nature foremost in consideration whenever possible. In keeping with this strategy, our order-theoretic notation will be standard, with the exception of the use of $\perp$ and $\top$ to denote the least and greatest element, respectively, of a poset (when such exist).

2. What is a domain?

A *program* is a syntactic description of a computational process. As Abramsky observed in [1], there are three activities involved in program development:

(1) *Program specification*, the task of defining a family of formulas each providing a syntactic description of a property of computations.
(2) *Program synthesis*, the task of finding a program for a given family of formulas.
(3) *Program verification*, the task of proving that a program satisfies a given family of formulas.

The third task provides us with the fundamental logical relationship in program development: $P \models \phi$, where P is a program and ϕ is a formula. This paper deals with the mathematics behind two approaches to the semantics of the triune process of program development. In the next few paragraphs, we introduce these approaches, then conclude the section with some motivation for those participating entities least likely to be familiar to algebraists. Throughout this section, we refer the reader to Abramsky [1], Abramsky and Jung [2], and Vickers [28] for details.

The *denotational* approach to programming languages, pioneered by D. S. Scott and C. Strachey, seeks to develop a semantics of computation. Each syntactic category of a programming language is assigned a *type* specifying which operations of the language may be performed upon it. A *domain* for a programming language is the underlying set of data objects for an admissible type, equipped with an information-based partial ordering. (Domains are usually algebraic posets under this ordering, for reasons discussed below.) In the denotational approach, the elements of domains represent programs. The domains are assigned a topology based upon their ordering, the open sets of which represent the formulas specifying programs. The fundamental relation is interpreted as set-inclusion: $P \models \phi \iff P \in \phi$.

The *axiomatic* approach to programming languages employs formal systems for reasoning about properties of programs. In the approach proposed by Abramsky [1], formulas specifying programs are viewed as elements of an abstract bialgebraic frame L. Programs are viewed as those maps from L to the two-element chain

$\{\perp, \top\}$ which preserve finite meets and arbitrary joins (frame homomorphisms). The fundamental relation is given by $P \models \phi \iff P(\phi) = \top$.

The connection between these approaches is provided by a Stone-type duality. On one hand, the set ptL of frame homomorphisms from L to the two-chain forms an algebraic poset whose lattice of open sets is order isomorphic to L. On the other hand, the open set lattice $\Sigma(D)$ of a algebraic poset D is bialgebraic and distributive, and $pt\Sigma(D)$ is order isomorphic to D. This correspondence extends to a categorical duality which assures that axiomatic semantics is compatible with denotational semantics in the sense that the denotation of a program is identified with the set of specifications true of it. (See Abramsky [1] and Robinson [19]; see also Vickers [28]).

Bialgebraic (algebraic and dually algebraic), distributive lattices should be familiar enough to algebraists; however, domains probably are not. We therefore pause to consider the question "What exactly *is* a domain?" Domains were introduced by Dana Scott for modeling computation. Since the concept evolved to meet specific needs of computer scientists, to answer the question requires a look at these needs.

As mentioned above, data objects are usually assigned types specifying which operations of the language may be performed upon them. In denotational semantics, the mathematical models for types are called *domains*. What properties do these models have?

A computation may be considered to be an algorithm acting successively upon a set of data objects to obtain increasingly refined approximations to a desired result. In this sense, computations are processes acting on types; a domain should therefore possess sufficient structure to allow mathematical meaning to be given to these processes. A computational process may be described by identifying each of its stages with a subset of the type whose total information content contains that of the desired result. The idea behind this scheme is simple: with each application of the algorithm, the data objects produced should provide better approximations to the desired result. The sets associated with the stages of the process are to represent refinements in the approximations of the desired result. In this scheme, the end result of the process is viewed as a "limit" of the approximating sets of data.

The structure of domains therefore depends on how we choose to formalize the loose notion that the end result of a computational process is a limit of the data given by its stages. To begin, models for types are viewed abstractly as posets, in which the relation $x \leq y$ implies that x is "less defined" than y, or that the information content of y is a "refinement" of that of x. Let T be a type endowed with such a partial ordering and suppose that P is a computational process acting on T. The nature of computers is such that, at any stage of P, the computer can act upon only finitely many data objects at a time. If subsequent stages are to represent refinements of the information content of a stage S, then for each finite $F \subseteq S$, there should exist a subsequent stage S' of P and a data object $d' \in S'$ such that d' is an upper bound for F. Consequently, the stages of a computational process should, together, form a directed subset of T. Recall that a subset D of a poset is *directed* provided every finite subset of D has an upper bound in D. It is natural, then, to consider computations in T to be joins of directed sets in T.

A poset P is said to be *directed complete* (a DCPO) provided the join of every directed subset of P exists in P. In view of the previous discussion, it is reasonable to consider domains to be DCPOs.

Given DCPOs P and Q, a function $f : P \longrightarrow Q$ is *Scott continuous* provided f preserves directed joins. A program in a language $\mathcal{L}$ can be thought of as a map between types of $\mathcal{L}$ which preserve computational processes. In the abstract, a program in $\mathcal{L}$ is a Scott continuous map between domains of $\mathcal{L}$. The class of all DCPOs with Scott continuous maps forms a category under function composition. We will call this category $\underline{\mathtt{DCPO}}$. Further, we will use $\underline{\mathtt{pDCPO}}$ to denote the full subcategory of $\underline{\mathtt{DCPO}}$ whose object class consists of all *pointed* DCPOs, that is DCPOs with least element.

In Abramsky [1], a category of domains is defined to be any full subcategory of $\underline{\mathtt{pDCPO}}$. In most applications, however, categories of domains are required to be *cartesian closed*. (A category is cartesian closed provided it has finite limits and a self-functor adjoint to the formation of finite products.) The reason for this lies in the fact that finite products in subcategories of $\underline{\mathtt{DCPO}}$, when they exist, are usually cartesian products; and, when it exists, the object assignment of the adjoint to the finite cartesian product of DCPOs is the *function space*. (Given DCPOs C and D, the function space $[C \to D]$ is the set of all Scott continuous maps from C to D under the pointwise ordering.) The formation of finite cartesian products and function spaces both have natural meanings as *type constructors* — processes by which new types may be constructed from existing ones. (For details on type constructors, see Abramsky [1], Abramsky and Jung [2], Plotkin [18], or Vickers [28].)

The need for domains to be pointed stems from the special demands of modeling recursively defined types and is not required for development of the general theory. For this reason, we will not make the blanket assumption in this paper that our DCPOs are pointed.

In many applications, it is also important to be able to express computations as limits (directed joins) of approximating "essential" or "explicitly computable" data objects. This need is reflected in the common requirement that domains be continuous or even algebraic as posets. The first DCPOs to be considered as models for computation were continuous lattices and were introduced by Scott in his fundamental works [20], [21], [22], and [23]. Today, continuous lattices form an important component of order theory in their own right (see in particular Gierz et al. [7]), but their value to denotational semantics has been limited by the fact that, as complete lattices, they contain elements which cannot be given natural meanings as computations. However, continuous lattices do possess a structural property that is highly desirable in most models of computation.

A type may contain data objects with finite or infinite information content. For purposes of this discussion, we will refer to these objects as *finite* and *infinite* elements, respectively, of the type. We look upon the finite elements of a poset P as representing data objects whose information content may be obtained by computation in a finite time. Due to the finite nature of computers, infinite data objects in T can be given a natural meaning only as the suprema of directed sets of approximating finite data objects. Consequently, if we want to distinguish between finite and infinite data objects, the domain model for T should be a DCPO in which every element is the join of a directed set of "finite" elements.

To understand what this is to mean, we must devise a formal definition for "finite data object". We want a finite approximation to a data object x to be essential to the computation of x and to any object which might refine x in the sense that, if $k \leq x$ is finite and D is any directed subset of T such that $x \leq \bigvee D$, then there exist $d \in D$ such that $k \leq d$. Under this assumption, the end result of any computational process which is approximated by x must be *at least as accurate* as k; and, in this sense, k is "essential" to the computation. In a DCPO P, such an element k is said to be *way-below* x. A DCPO P is said to be *continuous* if, for all $p \in P$, the set W_p of all elements way-below p is directed, and $p = \bigvee W_p$. Consequently, if we wish to distinguish between finite and infinite data objects in a type, then its domain model should be a continuous poset. (See Geirz et al. [7] for details concerning continuous lattices, including the motivation for the term "continuous".)

Strictly speaking, the previous discussion is inaccurate. The elements k we have defined really should be called *relatively* finite. A finite element should represent a data object computable in finite time. This means that a finite element k cannot itself be the supremum of a directed set D in which $d < k$ for all $d \in D$. Consequently, a finite element actually should be a relatively finite element which is wey-below itself. Such members of a DCPO P are said to be *compact*. To be precise, an element k of a DCPO P is compact if, for all directed $D \subseteq P$ such that $k \leq \bigvee D$, there exist $d \in D$ such that $k \leq d$. A DCPO P is *algebraic* if, for all $p \in P$, the set K_p of all compact elements below p is directed, and $p = \bigvee K_p$.

Unfortunately, the full subcategory $\underline{\texttt{AlgPos}}$ of $\underline{\texttt{DCPO}}$ whose object class consists of all algebraic posets fails to be cartesian-closed. (For example, the function space $[\mathbf{Z}^- \to \mathbf{Z}^-]$ fails to be algebraic, where $\mathbf{Z}^-$ denotes the negative integers under their natural ordering.) We are therefore led to seek subcategories of $\underline{\texttt{AlgPos}}$ which are cartesian-closed. The largest subcategory of $\underline{\texttt{AlgPos}}$ normally referred to as a category of domains has as its objects the so-called SFP domains (see Abramsky [1], Gunter [9], [10], Plotkin [18], and Smyth [26]). We will introduce the object class of this category and those of some of its most important subcategories in Section 5.

We now turn to a more systematic development of the topics discussed above.

3. The categories $\underline{\texttt{DCPO}}$ and $\underline{\texttt{Frm}}$

A function between DCPOs is said to be *computable* provided it preserves directed joins. That is, if P and Q are DCPOs and $f : P \longrightarrow Q$ is a function, then f is computable provided $f(\bigvee_P D) = \bigvee_Q \{f(d) : d \in D\}$ for all directed $D \subseteq P$. Note that this condition is equivalent to the requirement $f(\bigvee_P D) \leq \bigvee_Q \{f(d) : d \in D\}$. Consequently, computability expresses the requirement that, at least in the setting of algebraic posets, every finite amount of information about the resulting value $f(\bigvee_P D)$ requires only a finite amount of information about the element $\bigvee_P D$.

Computable functions between DCPOs are usually called Scott-continuous (or simply continuous) functions. The motivation for this lies with a natural topology which may be associated with any DCPO. We now describe this topology.

Let P be a poset. For a subset X of P, define the *lower set generated by* X in P to be the subset $\downarrow X = \{p \in P : p \leq x$, for some $x \in X\}$. The *upper set* of P generated by X is the subset $\uparrow X = \{p \in P : x \leq p$ for some $x \in X\}$. A subset I of P is a lower set of P if and only if $I = \downarrow I$. We write $\downarrow x$ in place of $\downarrow \{x\}$ and refer to this lower set as the *principal* lower set of P generated by x. The principal upper set of P generated by x is $\uparrow x = \uparrow \{x\}$.

A subset U of a DCPO P is *Scott-open* in P provided U is an upper set of P and, whenever $D \subseteq P$ is directed and such that $\bigvee D \in U$, then $D \cap U \neq \emptyset$. It is routine to prove that the collection of all Scott-open subsets of a DCPO P forms a topology (of open sets) for P. We call this topology the *Scott* topology for P and use $\Sigma(P)$ to denote the lattice of Scott-open subsets of P.

A *Scott-closed* subset of a DCPO P is simply a subset of P closed with respect to the Scott topology on P. It is easy to see that $C \subseteq P$ is Scott-closed if and only if C is a lower set of P which contains the join of each of its directed subsets.

It is routine to prove that a function $f : P \longrightarrow Q$ between DCPOs P and Q is computable if and only if it is continuous with respect to the Scott topology. Herein lies the motivation behind labeling computable functions as continuous. We will adhere to this convention for the remainder of this paper.

The class of all DCPOs together with continuous functions forms a category under function composition. We will use DCPO to denote this category and will use AlgPos to denote the full subcategory of DCPO whose objects are algebraic posets.

We hasten to point out that objects in these categories are not required to possess least elements. Such a restriction is vital for certain aspects of denotational semantics (such as fixpoint theory) but will not be needed in our considerations.

A function $f : L \longrightarrow M$ between frames L and M is called a *frame homomorphism* provided f preserves finite meets and arbitrary joins. In symbols, f is a frame homomorphism if and only if, for all finite $F \subseteq L$ and all $X \subseteq L$,

- $f(\bigwedge_L F) = \bigwedge_M \{f(a) : a \in F\}$, and
- $f(\bigvee_L X) = \bigvee_M \{f(a) : a \in X\}$.

We note that frame homomorphisms preserve least and greatest elements. The class of all frames together with frame homomorphisms forms a category under function composition. We will use Frm to denote this category.

Given any set X and topology of open sets Ω on X, it is easy to see that the open set lattice of Ω is a frame; in particular, $\Sigma(P)$ is a frame for any DCPO P.

Though it will not play a role in our considerations, we note that free frames exist. The free frame in countably many generators is the Lindenbaum algebra of propositional geometric logic (see Vickers [28]). The significance of the categorical duality between the full subcategory $\underline{\mathtt{AlgPos}}$ of $\underline{\mathtt{DCPO}}$ and the full subcategory $\underline{\mathtt{BiAlgFrm}}$ of $\underline{\mathtt{Frm}}$ described in Section 4 below lies with the power it provides for developing the axiomatic semantics of programming languages using the techniques of denotational semantics. For details, the reader is referred to Abramsky [1] and Robinson [19]; see also Vickers [28].

We have seen that every DCPO P may be associated with a frame, namely its lattice $\Sigma(P)$ of Scott-open sets. This correspondence extends to a contravariant functor $\Sigma : \underline{\mathtt{DCPO}} \longrightarrow \underline{\mathtt{Frm}}$. The morphism assignment $f \longmapsto \Sigma(f)$ maps a continuous function $f : P \longrightarrow Q$ between DCPOs P and Q to the frame homomorphism $\Sigma(f) : \Sigma(Q) \longrightarrow \Sigma(P)$ defined by $\Sigma(f)(V) = f^{-1}(V)$. This contravariant functor has associated with it a companion functor $pt : \underline{\mathtt{Frm}} \longrightarrow \underline{\mathtt{DCPO}}$, and together these functors provide a contravariant adjunction between these categories. The description of the companion functor requires some additional definitions and results.

Let L and M be frames and let $\underline{\mathtt{Frm}}[L, M]$ denote the set of frame homomorphisms from L to M under the *pointwise order*: $f \sqsubseteq g$ ifand only if $f(a) \leq g(a)$, for all $a \in L$. The following straightforward result shows that $\underline{\mathtt{Frm}}[L, M]$ is a DCPO (see Lemma 1.11, p. 47 of Johnstone [11]).

Lemma 3.1. *If L and M are frames, and $D \subseteq \underline{\mathtt{Frm}}[L, M]$ is a directed set, then the map $\bigsqcup D : L \longrightarrow M$ defined by $\bigsqcup D(a) = \bigvee_M \{f(a) : f \in D\}$ is the join of D in $\underline{\mathtt{Frm}}[L, M]$. In particular, $\underline{\mathtt{Frm}}[L, M]$ is a DCPO.*

In all that follows, we will use $\mathbf{2}$ to denote the two element chain $\{\bot, \top\}$. By Lemma 3.1, $\underline{\mathtt{Frm}}[L, \mathbf{2}]$ is a DCPO. We will call the elements of this poset the *points* of L and write ptL in place of $\underline{\mathtt{Frm}}[L, \mathbf{2}]$.

We can now describe the companion functor to the contravariant functor Σ. For every frame homomorphism $f : L \longrightarrow M$ between frames L and M, let $pt(f) : ptM \longrightarrow ptL$ be defined by $pt(f)(x) = x \circ f$ for all $x \in ptL$. The object assignment $L \longmapsto ptL$ coupled with the morphism assignment $f \longmapsto pt(f)$ constitutes a contravariant functor $pt : \underline{\mathtt{Frm}} \longrightarrow \underline{\mathtt{DCPO}}$.

Let $\lambda : Id_{\underline{\mathtt{DCPO}}} \longrightarrow pt \circ \Sigma$ be the following class of maps. For every DCPO P, the member $\lambda_P : P \longrightarrow pt\Sigma(P)$ of the class λ is the map defined by $\lambda_P(q)(U) = \top$ if and only if $q \in U$.

Let $\rho : Id_{\underline{\mathtt{Frm}}} \longrightarrow \Sigma \circ pt$ be the following class of maps. For every frame L, the member $\rho_L : L \longrightarrow \Sigma(ptL)$ of the class ρ is defined by $\rho_L(a) = \{x \in \mathrm{pt}L : x(a) = \top\}$.

The straightforward, albeit tedious, proof of the next result is included for completeness. It makes use of the well-known connection between adjunctions and free pairs. We refer the reader to MacLane [14] (Theorem 2, p. 81) for more information and note, as a word of caution, that the general result in that book is stated for covariant adjunctions.

Theorem 3.2. *The functors Σ and pt form a contravariant adjunction between $\underline{\mathtt{DCPO}}$ and $\underline{\mathtt{Frm}}$. Moreover, the classes λ and ρ are the units of the adjunction.*

Proof To prove that $(\Sigma, pt, \lambda, \rho)$ is a contravariant adjunction, it will suffice to show that, for every DCPO P, the pair $(\Sigma(P), \lambda_P)$ is free over P with respect to the functor pt. The proof that ρ is the other unit of the adjunction is left to the reader.

Let P be a DCPO. We begin by showing that λ_P is continuous. Let $D \subseteq P$ be directed and let $U \subseteq P$ be Scott-open. By definition of λ_P,

$$\begin{aligned}
\lambda_P(\bigvee D)(U) = \top \quad &\iff \quad \bigvee D \in U \\
&\iff \quad d \in U, \exists d \in U \\
&\iff \quad \lambda_P(d)(U) = \top, \exists d \in U \\
&\iff \quad \bigsqcup \{\lambda_P(d) : d \in D\}(U) = \top.
\end{aligned}$$

Thus, $\lambda_P(\bigvee D) = \bigsqcup\{\lambda_P(d) : d \in D\}$; and λ_P is continuous.

To prove the universal property, we must show that, for every frame L and continuous function $f : P \longrightarrow ptL$, there exists a unique frame homomorphism $\varphi : L \longrightarrow \Sigma(P)$ such that $pt(\varphi) \circ \lambda_P = f$.

For all $a \in L$, set $\varphi(a) = \{p \in P : f(p)(a) = \top\}$. Since f is isotone, $\varphi(a)$ is clearly an upper set of P for all $a \in L$. To see that $\varphi(a)$ is Scott-open, suppose that $D \subseteq P$ is directed and such that $\bigvee_P D \in \varphi(a)$. It then follows that $f(\bigvee_P D)(a) = \top$. Since f is isotone, $\{f(d) : d \in D\}$ is directed in ptL; thus, since f is continuous, $f(\bigvee_P D) = \bigsqcup\{f(d) : d \in D\}$. It follows that $f(d)(a) = \top$ for some $d \in D$; in particular, $d \in \varphi(a)$. This completes the proof that $\varphi(a)$ is Scott-open in P. The proof that φ is a frame homomorphism is similar and will be left to the reader.

To establish that $pt(\varphi) \circ \lambda_P = f$, observe that, for all $p \in P$, the morphism assignment of pt stipulates that $(pt(\varphi) \circ \lambda_P)(p) = \lambda_P(p) \circ \varphi$. Consequently, for all $a \in L$, we have

$$\begin{aligned}
f(p)(a) = \top \quad &\iff \quad p \in \varphi(a) \\
&\iff \quad \lambda(p)(\varphi(a)) = \top.
\end{aligned}$$

Thus, $pt(\varphi \circ \lambda_P) = f$.

It remains to prove that φ is unique. To this end, suppose that $\psi : L \longrightarrow \Sigma(P)$ is a frame homomorphism such that $f = pt\psi \circ \lambda_P$. Then, for all $a \in L$,

$$\begin{aligned}
p \in \psi(a) = \top \quad &\iff \quad \lambda_P(p)(\psi(a)) = \top \\
&\iff \quad f(p)(a) = \top \\
&\iff \quad \lambda_P(p)(\varphi(a)) = \top \\
&\iff \quad p \in \varphi(a).
\end{aligned}$$

□

4. Algebraic Posets and Bialgebraic Frames

In the paragraphs to follow, we will prove that the contravariant adjunction $(\Sigma, pt, \lambda, \rho)$ described in Section 3 restricts to a dual equivalence between the full subcategory of DCPO consisting of algebraic posets with continuous maps (which we will call $\underline{\texttt{AlgPos}}$) and an important full subcategory of $\underline{\texttt{Frm}}$ whose object class we now introduce.

A complete lattice L is *bialgebraic* provided both L and its dual are algebraic posets. Every algebraic, distributive lattice is a frame; we will use $\underline{\texttt{BiAlgFrm}}$ to denote the full subcategory of $\underline{\texttt{Frm}}$ whose object class consists of all bialgebraic, distributive lattices (bialgebraic frames).

In the work to follow, we will prove that $\underline{\texttt{AlgPos}}$ is dually equivalent to $\underline{\texttt{BiAlgFrm}}$. This result is known to computer scientists (see Vickers [28]); however, our proof will be based upon purely lattice-theoretic ideas.

Let L be a lattice. An element p of L is *meet-prime* (MP) if, whenever $F \subseteq L$ is finite, then $\bigwedge F \leq p$ always implies $x \leq p$ for some $x \in F$. We say p is *meet-irreducible* (MI) if, whenever $F \subseteq L$ is finite and $p = \bigwedge F$, then $p = x$ for some $x \in F$. Every meet-prime element of L is meet- irreducible; the converse is true if L is distributive. Note that, if L has a greatest element, $\top$, then the fact that $\top = \bigwedge \emptyset$ precludes $\top$ from being meet-irreducible. An element j is L is *join-prime* (JP) or *join-irreducible* (JI) in L provided it is meet-prime or meet-irreducible, respectively, in L^{op}.

Let L be a complete lattice. An element p of L is *completely* meet-prime (CMP) if $X \subseteq L$ and $\bigwedge X \leq p$ always implies that $x \leq p$ for some $x \in X$. By similarly extending the definitions of MI, JI, and JP elements to include arbitrary meets and joins, we obtain the definitions for *completely* meet-irreducible (CMI), *completely* join-irreducible (CJI), and *completely* join-prime (CJP) elements of L. Observe that an element of L is CJP if and only if it is compact and join-prime in L.

In all that follows, we will use $\texttt{MP}(L)$, $\texttt{MI}(L)$, $\texttt{JP}(L)$, and $\texttt{JI}(L)$ to denote the subposets of meet-prime, meet-irreducible, join-prime, and join-irreducible elements, respectively, of a lattice L. Likewise, we will use $\texttt{CMP}(L)$, $\texttt{CMI}(L)$, $\texttt{CJP}(L)$, and $\texttt{CJI}(L)$ to denote the subposets of completely meet-prime, completely meet-irreducible, completely join-prime, and completely join-irreducible elements, respectively, of a complete lattice L.

The meet-prime elements of a frame L are in bijective, order reversing correspondence with the points of L. Indeed, if $p \in \texttt{MP}(L)$, then the function $x_p : L \longrightarrow \mathbf{2}$ defined by $x_p(a) = \perp \iff a \in {\downarrow}p$ is a point of L with $x_p^{-1}(\perp) = {\downarrow}p$. On the other hand, if x is a point of L, then, since x is a frame homomorphism, $x^{-1}(\perp) = {\downarrow}p_x$ for some meet-prime $p_x \in L$. It is routine to prove that the assignments $p \longmapsto x_p$ and $x \longmapsto p_x$ are mutually inverse and order reversing. For future reference, we summarize this fact in the following lemma.

Lemma 4.1. *If L is a frame, then $\texttt{MP}(L)$ is dually order isomorphic to ptL. The dual isomorphism is implemented via the assignments $p \longmapsto x_p$ and $x \longmapsto p_x$ described above.*

It is easy to see that, given any poset P, the set $\mathcal{L}(P)$ of all lower sets of P, ordered by set-inclusion, is a bialgebraic, distributive lattice under the operations of set-union and set-intersection. It is well-known that every element of an algebraic lattice is the meet of a set of CMI elements. Using this fact, the following result provides several characterizations of bialgebraic frames, including the fact that every such frame is of the form $\mathcal{L}(P)$ for some poset P. For a proof of this result, the reader is referred to Crawley and Dilworth [4](p. 82).

Theorem 4.2. *For a complete lattice L, the following are equivalent:*

(1) *L is a bialgebraic frame;*
(2) *L is algebraic and L^{op} is a frame;*
(3) *L is an algebraic frame, and $\mathtt{CMP}(L) = \mathtt{CMI}(L)$;*
(4) *Every element of L is the join of a set of CJP elements;*
(5) *L is isomorphic to the frame of lower sets of $\mathtt{CJP}(L)$; and*
(6) *L is isomorphic to the frame of lower sets of some poset P.*

Let L be a complete lattice, and let $a, b \in L$. We say the ordered pair (a, b) *splits* L provided $\downarrow a \cap \uparrow b = \emptyset$ and $\downarrow a \cup \uparrow b = L$.

Lemma 4.3. *Let L be a complete lattice. If (a, b) splits L, then a is CMP and b is CJP in L.*

Proof We show that a is CMP in L. The fact that b is CJP follows from this and the observation that (a, b) splits L if and only if (b, a) splits L^{op}. To see that a is CMP, let $X \subseteq L$ be such that $x \not\leq a$ for all $x \in X$. Since (a, b) splits L and no element of X is contained in $\downarrow a$, we must have $X \subseteq \uparrow b$. Consequently, $b \leq \bigwedge X$; the fact that $\downarrow a \cap \uparrow b = \emptyset$ now implies that $\bigwedge X \not\leq a$. □

Given a complete lattice L and $a, b \in L$, set $p_b = \bigvee\{x \in L : b \not\leq x\}$ and set $j_a = \bigwedge\{y \in L : y \not\leq a\}$.

Lemma 4.4. *Let L be a complete lattice and let $a, b \in L$.*

(1) *If a is CMP in L, then j_a is CJP in L and (a, j_a) splits L.*
(2) *If b is CJP in L, then p_b is CMP in L and (p_b, b) splits L.*

Proof We prove Claim (1) and observe that Claim (2) follows by duality. Since a is CMP, it follows from the definition of j_a that $j_a \not\leq a$ and that, for each $x \in L$, we have $x \not\leq a$ if and only if $j_a \leq x$. Therefore, (a, j_a) splits L. The element j_a is CJP by Lemma 4.3. □

Lemmas 4.3 and 4.4 imply that, for every complete lattice L, the assignments $p \longmapsto j_p$ and $j \longmapsto p_j$ constitute mutually inverse isotone maps bewteen $\mathtt{CMP}(L)$ and $\mathtt{CJP}(L)$. Consequently, we have the following result, recorded as a lemma for future reference.

Lemma 4.5. *If L is a complete lattice, then $\mathtt{CJP}(L)$ is order isomorphic to $\mathtt{CMP}(L)$. The isomorphism is implemented via the mutually inverse maps $j \longmapsto p_j$ and $p \longmapsto j_p$.*

The following result describes when a point of a frame L is compact in ptL. Its simple proof relies on Lemmas 4.1 and 4.5 and is left to the reader.

Lemma 4.6. *Let L be a frame.*

(1) *If p is CMP in L, then the point $x_p : L \longrightarrow \mathbf{2}$ defined by $x_p(a) = \bot \iff a \leq p$ is a complete lattice homomorphism; that is, x_p preserves arbitrary joins and meets.*

(2) *If a point x of L is a complete lattice homomorphism, then the element $p_x = \bigvee x^{-1}(\bot)$ is CMP in L.*
  - *Suppose, in addition, that every element in L is a meet of meet-prime elements. Then the following hold.*

(3) *A point of L is compact in ptL if and only if it is a complete lattice homomorphism.*

(4) *The poset $K(ptL)$ of compact elements of ptL is dually isomorphic to* $\mathtt{CMP}(L)$*.*

We are now ready to prove that the contravariant adjunction $(\Sigma, \mathrm{pt}, \lambda, \rho)$ restics to a dual equivalence between $\underline{\mathtt{AlgPos}}$ and $\underline{\mathtt{BiAlgFrm}}$.

Let P be a DCPO and let $x \in P$. It is easy to see that $\uparrow x$ is Scott-open in P if and only if $x \in K(P)$. Furthermore, it is easy to see that whenever $\uparrow x$ is Scott-open for $x \in P$, then $\uparrow x$ is CJP in $\Sigma(P)$. With these facts in mind, we have the following result.

Lemma 4.7. *Let P be an algebraic poset.*

(1) *If $U \in \Sigma(P)$, then $U = \bigcup\{\uparrow x : x \in K(P) \cap U\}$.*

(2) *$\Sigma(P)$ is a bialgebraic, distributive lattice.*

(3) *An element of $\Sigma(P)$ is CJP if and only if it is of the form $\uparrow x$ for some $x \in K(P)$.*

(4) *The map $x \longmapsto \uparrow x$ is a dual order isomorphism between $K(P)$ and* $\mathtt{CJP}(\Sigma(P))$*.*

Proof Let P be an algebraic poset. Note first that for an element $x \in P$ the following are equivalent: (i) $x \in K(P)$, (ii) $\uparrow x \in \Sigma(P)$ and (iii) $\uparrow x$ is CJP in $\Sigma(P)$. In view of these facts, Claims (2), (3) and (4) are immediate consequences of Claim (1) and Theorem 4.4. We therefore prove Claim (1).

Let P be an algebraic poset, let $U \in \Sigma(P)$, and let V be the set $V = \bigcup\{\uparrow x : x \in K(P) \cap U\}$. It is clear that $V \subseteq U$. To obtain the reverse inclusion, suppose that $p \in U$. Since P is an algebraic poset, $K_p = \downarrow p \cap K(P)$ is directed and $p = \bigvee K_p$. Since U is Scott-open, it follows that there must exist $x \in K_p$ such that $x \in U$. Since $p \in \uparrow x$; we see that $U \subseteq V$. □

Lemma 4.8. *If L is a bialgebraic frame, then ptL is an algebraic poset.*

Proof In light of Lemma 4.1, it will suffice to show that $\mathtt{MP}(L)$ is a dually algebraic poset. In an algebraic lattice, every element is the meet of a set of CMI elements. It follows from Theorem 4.2 that every MP element of L is the meet of a set of CMP elements. Thus, in light of Lemma 4.6, it will suffice to prove that, for all $p \in \mathtt{MP}(L)$, the set $\uparrow p \cap \mathtt{CMP}(L)$ is down-directed in L.

To this end, let p be MP in L and suppose that p is a lower bound in L for a finite set F of CMP elements. We will find a CMP lower bound for F which exceeds p. We know that $p \leq \bigwedge F$. If $p = \bigwedge F$, then $p \in F$ by virtue of the fact that p is MP. Consequently, we may assume that $p < \bigwedge F$. For each $q \in F$, let j_q be the CJP element of L corresponding to q and let $J = \{j_q : q \in F\}$. Since $p < \bigwedge F$, we know that $j_q \not\leq p$ for all $q \in F$; hence, we know $\bigwedge J \not\leq p$. By Theorem 4.2, there exists a CJP element $c \in L$ such that $c \leq \bigwedge J$ but $c \not\leq p$. By Lemma 4.4, the element p_c is CMP and the pair (p_c, c) splits L. Since $c \not\leq p$, it must be true that $p \leq p_c$. Since $c \leq \bigwedge J$, it follows that $p_c \leq \bigwedge F$. Thus, p_c is the CMP lower bound for F that we seek. □

Lemma 4.9. *If L is a bialgebraic frame, then the map $\rho_L : L \longrightarrow \Sigma(ptL)$ is an isomorphism. Also, if P is an algebraic poset, then the map $\lambda_P : P \longrightarrow pt(\Sigma(P))$ is an isomorphism.*

Proof By Lemmas 4.7 and 4.8, $\Sigma(ptL)$ is a bialgebraic frame. We first establish that $\rho_L : L \longrightarrow \Sigma(ptL)$ is an isomorphism. To begin, it is easy to see that ρ_L is an injection. Indeed, let $a, b \in L$ be distinct elements. We may assume that $a \not\leq b$. There is a CMP element $p \in L$ such that $a \not\leq p$ but $b \leq p$. Consequently, if x_p is the point of L corresponding to p, then $x_p(a) = \top$ and $x_p(b) = \bot$. Thus, $\rho_L(a) \neq \rho_L(b)$.

It remains to prove that ρ_L is a surjection. By Lemmas 4.7 and 4.8, we know that if $U \in \Sigma(\mathrm{pt}L)$, then $U = \bigcup\{\uparrow x : x \in K(ptL)\}$. Since ρ_L is a frame homomorphism, to prove that ρ_L is a surjection, it will thus suffice to prove that $\uparrow x$ is in the image of L under ρ_L for all $x \in K(ptL)$. To this end, let $x \in K(ptL)$ and let p_x be the CMP element of L associated with x (see Lemma 4.6). Let $j = j_{p_x}$ be the CJP element of L associated with p_x (see Lemma 4.5). Observe that $y \in \rho_L(j) \iff y(j) = \top \iff j \not\leq p_y \iff p_y \leq p_x \iff x \leq y \iff y \in \uparrow x$. Hence, $\rho_L(j) = \uparrow x$.

We next prove that $\lambda_P : P \longrightarrow pt(\Sigma(P))$ is an order isomorphism. Recall that, for all $q \in P$, the map $\lambda_P(q) : \Sigma(P) \longrightarrow \mathbf{2}$ is defined by $\lambda_P(q)(U) = \top \iff q \in U$.

To see that λ_P is an order embedding, note first that λ_P is order preserving, since it is continuous. Let now $x, y \in P$ such that $\lambda_P(x) \leq \lambda_P(y)$. Then for all $U \in \Sigma(P)$, $y \in U$ whenever $x \in U$. In particular, the choice $U = \uparrow x$ gives $y \in \uparrow x$, that is, $x \leq y$.

To see that λ_P is a surjection, suppose first that x is a compact point of $\Sigma(P)$. By Lemma 4.6, x is a complete lattice homomorphism; hence, $x^{-1}(\top) = \uparrow U$ for some $U \in \Sigma(P)$. It is clear that U is a CJP element of $\Sigma(P)$; consequently, $U = \uparrow k$ for some $k \in K(P)$. Observe that $\lambda_P(k)(V) = \top \iff k \in V \iff U \subseteq V \iff x(V) = \top$. Consequently, $x = \lambda_P(k)$.

We have shown that the restriction of λ_P to $K(P)$ provides an order isomorphism between $K(P)$ and $K(pt\Sigma(P))$. Now suppose that y is an arbitrary point of $\Sigma(P)$. Since $pt\Sigma(P)$ is an algebraic poset, the set $K_y = \{x \in K(pt\Sigma(P)) : x \sqsubseteq y\}$ is directed and $y = \bigsqcup K_y$. By previous arguments, the set $\lambda_P^{-1}(K_y)$ is directed in $K(P)$; since λ_P is continuous, it now follows that $y = \lambda_P(\bigvee_P \lambda_P^{-1}(K_y))$. Thus, λ_P is a surjection. □

Combining Lemmas 4.7, 4.8, and 4.9, we obtain the main result of this section.

Theorem 4.10. *The contravariant adjunction $(\Sigma, pt, \lambda, \rho)$ restricts to a dual equivalence between* <u>`AlgPos`</u> *and* <u>`BiAlgFrm`</u>.

5. Further Refinements

The category AlgPos is one of the most interesting categories of ordered structures; but, unfortunately, this category fails to be closed under formation of function spaces (thereby failing to be cartesian-closed), a requirement critical to the semantics of recursion. Two categories which provide attractive remedies to this problem are the full subcategories of AlgPos whose objects are SFP domains and Scott domains. We will discuss these categories in this section.

We begin by considering the class of lattices whose members are isomorphic to ideal completions of lower bounded, distributive lattices. A poset is a member of this class if and only if it is an algebraic frame whose compact elements form a sublattice. We shall use the term *precoherent* frame for any member of this class. In this context, *coherent* frame is a precoherent frame whose greatest element is compact. Note that a poset is a coherent frame if and only if it is isomorphic to the ideal completion of a *bounded*, distributive lattice.

As an aside, we note that the logical significance of coherent frames lies in the fact that they possess presentations involving only finite disjunctions (see Johnstone [11]).

Let P be an algebraic poset and let $S \subseteq K(P)$ be finite. A set $M_S \subseteq K(P)$ is a *complete set of minimal upper bounds* for S provided

- M_S is a finite set of upper bounds for S;
- If k is any compact upper bound for S, then there exist $m \in M_S$ such that $m \leq k$.

The concept of a complete set of maximal lower bounds for S is defined dually. It is clear that, when M_S exists, we may assume it is an antichain. Furthermore, it is clear that, under this assumption, M_S is unique. We will reserve the symbol $\mathrm{MUB}(S)$ to denote the unique, pairwise incomparable complete set of minimal upper bounds for S (when it exists). It is clear that $\mathrm{MUB}(\emptyset)$ exists in P if and only if P has a finite set $M = \{m_1, \ldots, m_k\}$ of minimal elements and $P = \bigcup_{j=1}^{k} \uparrow m_j$. Note also that the empty set is itself a complete set of minimal upper bounds for a finite set $S \subseteq K(P)$ if and only if S has no upper bounds in P.

We wish to note that the original sources for the coherent cases of Lemmas 5.1 - 5.4 below are Abramsky [1], Gunter [9], [10], and Plotkin [17]. The reader is also referred to Vickers [28] for additional information.

Lemma 5.1. *Let P be an algebraic poset and let S be a finite nonempty subset of $K(P)$.*

(1) *The set S has a complete set of minimal upper bounds if and only if the upper set $\bigcap\{\uparrow s : s \in S\}$ is compact in $\Sigma(P)$.*
(2) *The empty set has a complete set of minimal upper bounds if and only if P is compact in $\Sigma(P)$.*

Proof

We prove Claim (1). First, suppose that $U = \bigcap\{\uparrow s : S \in S\}$ is compact in $\Sigma(P)$. By Lemma 4.7 and the fact that U is compact, there exists a finite set

$M_S \subseteq K(P)$ such that $U = \bigcup\{\uparrow m : m \in M_S\}$. It is routine to prove that M_S is a complete set of minimal upper bounds for S in $K(P)$.

Conversely, suppose that S has a complete set of minimal upper bounds. Let M_S be such a set. If M_S is empty, $\bigcap\{\uparrow s : s \in S\} = \emptyset$ and is therefore compact in $\Sigma(P)$. If M_S is not empty, let $U = \bigcap\{\uparrow s : s \in S\}$ and let $V = \bigcup\{\uparrow m : m \in M_S\}$. By Lemma 4.7, $\uparrow m$ is compact (indeed, CJP) in $\Sigma(P)$; hence, V is compact in $\Sigma(P)$. We will prove U is compact by showing that $U = V$.

Observe that, by construction, $p \in U$ if and only if p is an upper bound for S. Since P is an algebraic poset, it follows that $p \in U$ if and only if there exist compact $k \in P$ such that k is an upper bound for S and $k \leq p$. Since M_S is a complete set of minimal upper bounds for S, we see that $p \in U \iff p \in \uparrow m$ for some $m \in M_S \iff p \in V$. Consequently, $U = V$, as desired. □

An algebraic poset P in which every finite, nonempty subset of compact elements has a complete set of minimal upper bounds in $K(P)$ will be called a *precoherent* algebraic poset. We call such a poset *coherent* if, in addition, the empty set has a complete set of minimal upper bounds in $K(P)$. The reason for this terminology becomes apparent in the next few lemmas.

In a bialgebraic frame, every compact element is a finite join of CJP elements. As a result, the following is an immediate consequence of the distributive law.

Lemma 5.2. *In a bialgebraic frame, the meet of every nonempty, finite set of compact elements is compact if and only if the meet of every finite, nonempty set of CJP elements is compact.*

The next three results are direct consequences of Lemmas 4.7, 4.9, 5.1, and 5.2.

Lemma 5.3. *Let L be a frame.*

(1) *L is a precoherent, bialgebraic frame if and only if ptL is a precoherent algebraic poset.*
(2) *L is a coherent, bialgebraic frame if and only if ptL is a coherent algebraic poset.*

Lemma 5.4. *Let P be an algebraic poset.*

(1) *P is precoherent if and only if $\Sigma(P)$ is a precoherent, bialgebraic frame.*
(2) *P is coherent if and only if $\Sigma(P)$ is a coherent, bialgebraic frame.*

We wish to advise the reader that a coherent frame is often called a *spectral* frame, especially in more topological approaches to the subject. This terminology may be traced to the well-known fact that the open-set lattice for the prime spectrum of a bounded, distributive lattice is a coherent frame (see, for example, Johnstone [11]). The term "coherent" is widely used in domain theory and is therefore the one we adopt. Coherent, bialgebraic frames are frequently called *spectral algebraic* frames.

6. The Poset of Prime Ideals of a Distributive Lattice

In what follows, <u>`JPLat`</u> denotes the class of all lower-bounded (necessarily distributive) lattices which are finitely generated by their join-prime elements, and

MJPLat denotes the class of all members of JPLat having the additional property that the meet of any finite nonempty subset of join-prime elements is either join-prime or the least element of the lattice. Other pertinent results can be found in Balbes [3], Davey [5], and Speed [27].

Theorem 6.1.

(1) *A poset is isomorphic to the poset of prime ideals of a member of the class* JPLat *if and only if it is a dually algebraic poset in which every nonempty finite subset of dually compact elements has a complete set of lower bounds.*
(2) *A poset is isomorphic to the poset of prime ideals of a bounded member of the class* JPLat *if and only if it is a dually algebraic poset in which every finite subset of dually compact elements has a complete set of lower bounds.*

Theorem 6.2.

(1) *A poset is isomorphic to the poset of prime ideals of a member of the class* MJPLat *if and only if it is a dual predomain.*
(2) *A poset is isomorphic to the poset of prime ideals of a bounded member of the class* MJPLat *whose greatest element is join-prime if and only if it is a dual domain.*

Corollary 6.3. *Any dually algebraic lattice is isomorphic to the poset of prime ideals of a distributive lattice (in fact, of a member of the class* MJPLat*).*

References

[1] S. Abramsky, *Domain theory in logical form*, Annals of Pure and Applied Logic 51 (1991), 1-77. 1, 2, 2, 3, 5

[2] S. Abramsky and A. Jung, *Domain Theory*, in *Handbook of Logic in Computer Science*, S, Abramksy, D. Gabbay, and T.S.E. Maibaum eds., Oxford University Press, 1994, pp.1-168. 1, 2

[3] R. Balbes, *On the partially ordered set of prime ideals of a distributive lattice*, Can. J. Math. 23 (1971), 866-874. 6

[4] P. Crawley and R.P. Dilworth, *Algebraic Theory of Lattices*, Prentice-Hall, Englewood Cliffs, New Jersey, 1973. 4

[5] B.A. Davey, *A note on representable posets*, Algebra Universalis 3 (1973), 345-347. 6

[6] B.A. Davey and H.A. Priestley, *Introduction to Lattices and Order*, Cambridge University Press, Cambridge, 1990.

[7] G. Gierz, K.H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, and D.S. Scott, *A Compendium of Continuous Lattices*, Springer-Verlag, Berlin, 1980. 2

[8] G. Grätzer, *General Lattice Theory*, Academic Press, New York, 1978.

[9] C. Gunter, *Profinite solutions for recursive domain equations*, Technical Report CMU-CS-85-107, Carnegie-Melon Univ., 1985. 2, 5

[10] C. Gunter, *Universal profinite domains*, Inform. and Comput. 72(1) (1985), 1-30. 2, 5

[11] P.T. Johnstone, *Stone Spaces*, Cambridge University Press, Cambridge, 1982. 3, 5, 5

[12] J.D. Lawson, *The duality of continuous posets*, Houston J. Math. 5(3) (1979), 357-386. 1

[13] R. McKenzie, G. McNulty, and W. Taylor, *Algebras, Lattices, Varieties*, Vol. 1, Wadsworth and Brooks/Cole, Monterey, California., 1987

[14] S. MacLane, *Categories for the Working Mathematician*, Springer-Verlag, Berlin, 1971. 3

[15] M.W. Mislove, *Algebraic posets, algebraic CPO's and Models of Concurrency*, in *Topology and Category Theory in Computer Science*, G.M. Reed, A.W. Roscoe, and R.F. Wachter (eds), Oxford University Press (1991), 75-111. 1

[16] M.W. Mislove, and F.J. Oles, *A topological algebra for angelic nondeterminism*, Research Report, IBM Res. Div., 1991.

[17] G.D. Plotkin, *A Powerdomain construction*, SIAM J. Computing 5 (1976), 452- 488. 5

[18] G.D. Plotkin, *Post-graduate Lecture Notes in Advanced Domain Theory*, J. of Computing Science, Univ. of Edinburgh, 1981. 2

[19] E. Robinson, *Logical aspects of denotational semantics*, in *Category Theory*, D.H. Pitt, A. Poigné, and D. E. Rydeheard (Eds), Lecture Notes in Computer Science 283, Springer-Verlag, (1987), 238-253. 2, 3

[20] D.S. Scott, *Outline of a mathematical theory of computation*, Proceedings of the 4th Annual Princeton Conference on Information Sciences and Systems (1970), 169-176. 2

[21] D.S. Scott, *The lattice of flow diagrams*, in *Symposium on Semantics of Algorithmic Languages*, E. Engeler (Ed), Lecture Notes in Mathematics 188, Springer-Verlag (1971), 311-366. 2

[22] D.S. Scott, *Continuous lattices*, in *Toposes, algebraic geometry and logic*, F. W. Lawvere (Ed), Lecture Notes in Mathematics 274, Springer-Verlag (1972), 97-136. 2

[23] D.S. Scott, *Lattice theory, data types, and semantics*, in *Formal Semantics of Programming Languages*, R. Rustin (Ed), Prentice-Hall (1972), 65-106. 2

[24] D.S. Scott, *Data Types as lattices*, SIAM J. Computing 5 (1976), 522-587.

[25] D.S. Scott and C. Strachey, *Towards a Mathematical Semantics of Computer Languages*, Technical Report PRG-6, Programming Research Group, University of Oxford (1971).

[26] M. Smyth, *The largest cartesian-closed category of domains*, Theoret. Comput. Sci. 5(3) (1977), 257-274. 2

[27] T.P. Speed, *On the order of prime ideals*, Algebra Universalis 2 (1972), 85-87. 6

[28] S. Vickers, *Topology via Logic*, Cambridge University Press, Cambridge, 1989. 1, 2, 3, 4, 5

James B. Hart
Middle Tennessee State University,
Murfreesboro TN 37132, USA.

Constantine Tsinakis
Vanderbilt University,
Nashville TN 37240, USA.

Recibido: 10 de abril de 2008
Aceptado: 22 de abril de 2008