

Logic of Fixed Points and Scott Topology*

Michael A. Bukatin^{1,2}

¹ Department of Computer Science, Brandeis University, Waltham, MA 02254, USA;

bukatin@cs.brandeis.edu; <http://www.cs.brandeis.edu/~bukatin/papers.html>

² MetaCarta, Inc., 126 Prospect St., Suite 5, Cambridge, MA 02139

Abstract. Domains with Scott topologies are frequently defined as sets of fixed points of retractions or, more generally, arbitrary Scott continuous transformations of domains. For the case of retractions, the logical approach to this technique was developed by Hoofman under the name of *continuous information systems*. We present this approach for general Scott continuous transformations, together with its underlying intuition, a number of applications to closure operations and finitary retractions in the algebraic case, and a spectrum of outstanding open problems.

1 Introduction

The elementary covariant logical approach to domains is known under the name of *information systems*. This approach was introduced by Dana Scott in 1982 [18] for algebraic bounded complete dcpo's. We only consider bounded complete dcpo's in the present paper. Under this approach domain elements are thought of as *theories* in a logical calculus, and continuous functions are thought of as *inference engines*, deducing information about $f(x)$ from information about x . These inference engines are known under the name of *approximable mappings*. This means that whenever one has a *Scott continuous transformation*, one can think of it as a *non-standard logic*.

A more complicated and less elementary contravariant logical approach to domains uses the ideas of *Stone duality* and is beyond the scope of the present paper (see [1] and references therein).

In Section 3 we present the usual framework of information systems for algebraic Scott domains. We reformulate the notion of consistency in a more technically convenient way compared to the standard framework [18].

Section 4 is the key section of this paper. There we show how to take an arbitrary Scott continuous transformation (of a powerset or, more generally, of an algebraic bounded complete dcpo) and to use the corresponding approximable mapping as a non-standard logic describing the domain of the fixed points of this transformation. This allows us to explain the intuition underlying the general-

*Supported in part by NSF Grant CCR-9216185 and Office of Naval Research Grant ONR N00014-93-1-1015 and in part by Applied Continuity in Computations Project.

ization of the information systems to the case of continuous Scott domains via non-reflexive logic made by R. Hoofman [9]. Hoofman essentially considered inference engines corresponding to retractions and used the resulting non-reflexive logic to describe continuous domains.

Based on this intuition we further generalize the information systems to domains of fixed points of Scott continuous transformations of powersets. This allows us to go beyond the realm of continuous domains, and if the conjecture of Section 4.3.1 is correct, this approach would allow to describe all bounded complete dcpo's.

We also discuss the significance of the algebraic case from the logical point of view: algebraic Scott domains correspond to the standard logic, while some of the more general classes of bounded complete domains result from either Hoofman's non-reflexive logic or Sazonov's non-finitary logic [16].

The subsequent sections, which represent our earlier results, may be also viewed as applications of Section 4. Section 5 studies the notion of *subdomain* and the *domain of subdomains* for the algebraic Scott domains. We show that the closure operations play an exclusive role in the formation of subdomains in the algebraic case, as opposed to retractions or projections, even when those are finitary (i.e. their sets of fixed points are algebraic). This exclusive role can be explained by the intuition presented in Section 4. Namely, we'll see that it is the closure operations, and not the finitary retractions in general, that correspond to the standard logic under our approach.

Section 6 studies finitary retractions and provides the following novel simple criterion for their finitariness. Consider the approximable mapping r corresponding to a retraction $|r|$. The relation urv holds, if whenever u is true about x , v is true about $|r|(x)$. Here x is a domain element, and u and v are finite conjunctions of elementary statements from the underlying logic. Then the retraction $|r|$ is finitary, if and only if whenever urw , there is such finite conjunction v , that urv, vrv, vrw . Again, the results can be viewed as an illustration for the intuition of Section 4. Basically, the "reflexive" conjunctions v form the skeleton of the new standard logic describing the domain isomorphic to the $\text{Fix}(|r|)$. Some of the results of Sections 5 and 6 might be known in folklore, as Carl Gunter suggested to me. However, it turned out to be impossible to trace any written evidence of that.

2 Related Work and Notation

The canonical reference for the theory of domains equipped with Scott topology is [2].

The origins of using inverse transitivity to describe non-algebraic continuous domains should probably be traced to R-structures introduced by Smyth [20].

Hoofman gave this idea its truly logical form in [9].

Among the related papers of interest one should mention the results obtained by Vickers [21], Edalat and Smyth [8], and more recently Jung, Kegelmann, and Moshier [11].

One should also mention the thesis by Rothe [15] extensively studying retractions in continuous domains.

We assume that all dcpo's in this paper have the least element, \perp .

Recall that if an algebraic dcpo is bounded complete, it is called an *algebraic Scott domain* or, simply, a *Scott domain*, and that if a continuous dcpo is bounded complete, it is called a *continuous Scott domain*.

To be consistent with these definitions, if a dcpo is bounded complete, we call it a *complete Scott domain*.

3 Algebraic Information Systems and Domains

This section outlines the approach of information systems in the case of algebraic Scott domains as can be found, e.g. in [18, 12, 13].

The algebraic information systems and domains described in this reflect the standard notions of inference and theories in traditional formal theories.

3.1 Information Systems and Domains

The approach of information systems describes an approximation domain as a set of *theories* in a logical calculus. There are two ways to follow this approach. One could consider a given approximation domain, which obeys certain specific axioms, and then try to build such a logical calculus, that its theories form an isomorphic domain. This approach is very useful, when one needs to investigate which class of domains is covered by a specific variant of information systems, or when one starts with a given domain to begin with.

However, for didactic purposes, a different approach has proven much more valuable. This approach was used in the first key paper on information systems by Dana Scott [18] and works as follows. We presume that there is some approximation domain on the background, but we do not specify it precisely. Then, having this hypothetical domain in mind, we reason about the desired properties of our logical systems and impose appropriate axioms describing the behavior of these systems. Then we define domains as sets of theories and establish their properties as theorems rather than postulating them as axioms.

3.1.1 UNKNOWN as a Truth Value

An information system is a *logical calculus* of *elementary statements* and their *finite conjunctions*. These elementary statements and finite conjunctions should be viewed as *continuous predicates* of a special kind on the elements of the domain in question. These predicates map domain elements to a two-element set of truth values, however these truth values are not ordinary **true** and **false**, but **true** and **unknown**. This crucial feature is usually not emphasized enough, but one needs to keep it in mind.

The reason for this distinction is that we view a domain element, x , as a dynamic object, about which only some of the information is known at any given

time of the computational process, but which later can be supplemented with more information, and thus replaced with y , $x \sqsubseteq y$. The information, which was **unknown** about x , can become **true** about y . This dynamic process is, however, viewed as *monotonic* with respect to time — once a piece of information becomes **true** about an element, it remains this way further on.

It is possible to talk about **false** pieces of information about x — these are such pieces of information which are not **true** about any y , such that $x \sqsubseteq y$, that is, the pieces which cannot become **true** about x during its arbitrary monotonic evolution. The **false** truth values will be treated via *consistency* mechanism, however they will always remain auxiliary, and usually can be easily eliminated from the scene if necessary, while **true** and **unknown** truth values are essential.

3.1.2 Example: Formal Theories

The most natural example one should keep in mind is any traditional formal theory, where elementary statements are all formulas, and the domain in question is the domain of all theories ordered by ordinary set-theoretic inclusion.

If a statement belongs to a theory, we will say that it is **true** in (or, if you wish, “about”) this theory, otherwise it is **unknown** in this theory. If the theory cannot be refined to a larger non-contradictory theory to include a specific statement, we might wish to say that this statement is **false** in (“about”) this theory.

In the traditional formal theories any deduction is thought of as a formal text of finite length. Hence any entailment of a statement or a contradiction from a set of statements is made on the basis of a finite subset of this set. This *finitarity* property will be reflected in the definitions below.

3.1.3 Consistency and Entailment

Let us denote the set of *elementary statements* as D , and the set of all its finite subsets as $\mathcal{P}_{\text{fin}}(D)$. We interpret a finite subset, $u \subseteq D$, $u = \{d_1, \dots, d_n\}$, as a *finite conjunction* of statements d_1, \dots, d_n .

Keeping the hypothetical approximation domain at the background, we would like to call a finite conjunction, u , of elementary statements *consistent*, if there is a domain element, x , about which all statements of u are true.

It is traditional to assume that \emptyset is consistent, that is, that the domain in question is non-empty [18]. We do follow this tradition here. It is also traditional to assume that any single elementary statement is true about some domain element, that is, $\{d\}$ is consistent for any elementary statement d . This means that one considers contradictory elementary statements to be “junk” and wants to exclude them from an information system.

We would like to depart from this convention, as it seems to gain nothing, and makes it more difficult to talk about some natural examples, like the one considered in the previous subsection, and also makes it impossible to talk about effective structures on domains in full generality (the point of view, advocated, in particular, by V.Yu.Sazonov; our style of information system is, in fact,

intermediate between the traditional one and the style of Sazonov [16] and seems to be the most convenient). Our choice will also make the discourse in Sections 4 and 5 more technically convenient. In particular, it enables the systematic proof of Theorem 5.2 based on Section 5.3 instead of an *ad hoc* proof given in [3].

We would like to say, that a finite conjunction, u , of elementary statements *entails* an elementary statement, d , if whenever u is true about a domain element, x , statement d is also true about x .

Again, it is traditional to consider only consistent conjunctions, u , in the context of entailment. However, it is much more convenient to assume that a conjunction, which is not consistent, entails everything. It is technically convenient to introduce a special **false** statement, ∇ , which entails everything and follows from all inconsistent conjunctions.

Taking all this into account, the following definition becomes quite natural.

Definition 3.1 The tuple $A = (D_A, \nabla_A, \vdash_A)$, where D_A is a set of distinctive *tokens (elementary statements)*, $\nabla_A \in D_A$ (the *false statement*), $\vdash_A \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A)$ (the *entailment relation*) is called an *algebraic information system* if

1. $\emptyset \not\vdash_A \{\nabla_A\}$ (non-degeneracy; a calculus must admit at least one non-contradictory theory);
2. $\forall u, v \in \mathcal{P}_{\text{fin}}(D_A). v \subseteq u \Rightarrow u \vdash_A v$ (reflexivity of entailment and conjunction elimination);
3. $\forall u, v_1, \dots, v_n, w \in \mathcal{P}_{\text{fin}}(D_A). u \vdash_A v_1, \dots, u \vdash_A v_n, v_1 \cup \dots \cup v_n \vdash_A w \Rightarrow u \vdash_A w$ (transitivity of entailment and conjunction introduction);
4. $\forall u \in \mathcal{P}_{\text{fin}}(D_A). \{\nabla_A\} \vdash_A u$ (contradiction entails everything).

The equivalence between this and the classical definition of information system will be shown in Section 3.1.5.

3.1.4 Theories as Domain Elements

Definition 3.2 *Domain* $|A|$ associated with an algebraic information system A is the set of theories,

$$\{x \subseteq D_A \mid$$

1. $u \subseteq x, u \vdash_A v \Rightarrow v \subseteq x$ (x is deductively closed);
2. $\nabla_A \notin x$ (deductively closed x is consistent) }.

Informally, we will say that if $d \in x$, then d is **true** about x , otherwise d is **unknown** about x .

The following two theorems can be proven along the lines given in [18].

Theorem 3.1 Domain $|A|$ associated with algebraic information system A is an algebraic Scott domain.

Theorem 3.2 For any algebraic Scott domain (X, \sqsubseteq_X) , there is an algebraic information system A , such that the partial order $(|A|, \sqsubseteq_A)$ is isomorphic to partial order (X, \sqsubseteq_X) .

3.1.5 Equivalence between Classical Information Systems and Our Definition

Definition 3.3 $A = (D'_A, Con'_A, \vdash'_A)$, where D'_A is a set of distinctive *tokens* (*assertions*), $Con'_A \subseteq \mathcal{P}_{\text{fin}}(D'_A)$ (the *consistent* finite conjunctions of assertions), $\vdash'_A \subseteq Con'_A \times D'_A$ (the *entailment* relation), is called a *classical information system* if

- (i) $\emptyset \in Con'_A$ (non-degeneracy; empty domains are barred);
- (ii) $\forall d \in D'_A. \{d\} \in Con'_A$ (no “junk”);
- (iii) $\forall u \subseteq v \in Con'_A. u \in Con'_A$;
- (iv) $\forall d \in u \in Con'_A. u \vdash'_A d$ (reflexivity of entailment);
- (v) $\forall u \in Con'_A, d \in D'_A. u \vdash'_A d \Rightarrow u \cup \{d\} \in Con'_A$ (entailment preserves consistency);
- (vi) $\forall u, v \in Con'_A, d \in D'_A. (\forall d' \in v. u \vdash'_A d') \Rightarrow v \vdash'_A d \Rightarrow u \vdash'_A d$ (transitivity of entailment).

The domain $|A|$ associated with a classical information system A is defined as a set of $x \in D'_A$ satisfying the consistency condition, $\forall u \subseteq x. u$ is finite $\Rightarrow u \in Con'_A$, and the condition of deductive closeness, $\forall u \subseteq x, d \in D'_A. u \vdash'_A d \Rightarrow d \in x$. Of course, $\sqsubseteq_{|A|}$ is defined to equal \subseteq .

All such domains are algebraic Scott domains, and for any algebraic Scott domain X it is possible to build a classical information system A , such that $|A|$ is isomorphic to X . So everything is very similar to our version of algebraic information systems.

Now we are going to establish an even stronger equivalence between these two versions. We are going to build two translations as follows. Given an algebraic information system $A = (D_A, \nabla_A, \vdash_A)$, we will build a classical information system, $B = C(A)$, and given a classical information system, $B = (D'_B, Con'_B, \vdash'_B)$, we will build an algebraic information system $A = S(B)$, such that the following properties hold.

For domains, $|C(A)| = |A|$ and $|S(B)| = |B|$, where the set equalities take place, as opposed to mere isomorphisms. For information systems, $C(S(B)) = B$ and if we consider $A_c = S(C(A))$, then $D_{A_c} = D_A \setminus \{d \in D_A \mid d \vdash_A \nabla_A \& d \neq \nabla_A\}$ and \vdash_{A_c} is obtained by restricting \vdash_A on $\mathcal{P}_{\text{fin}}(D_{A_c}) \times \mathcal{P}_{\text{fin}}(D_{A_c})$. Basically, the only disturbance that $S \circ C$ cannot avoid is that other “junk” tokens equivalent to ∇_A die.

The translations are defined by the following formulas.

$D'_{C(A)} = \{d \in D_A \mid \{d\} \not\vdash_A \nabla_A\}$; $Con'_{C(A)} = \{u \in \mathcal{P}_{\text{fin}}(D_A) \mid u \not\vdash_A \nabla_A\}$; if $u \in Con'_{C(A)}$, $d \in D'_{C(A)}$, then $u \vdash'_{C(A)} d \Leftrightarrow u \vdash_A \{d\}$.

$D_{S(B)} = D'_B \cup \{\nabla_{S(B)}\}$ (under assumption $\nabla_{S(B)} \notin D'_B$). Consider $u, v \in \mathcal{P}_{\text{fin}}(D_{S(B)})$, $d_1, \dots, d_n \in D_{S(B)}$. If $u \notin Con'_A$ then $u \vdash_{S(B)} v$. If $u \in Con'_A$ then $u \vdash_{S(B)} \{d_1, \dots, d_n\} \Leftrightarrow u \vdash'_B d_1, \dots, u \vdash'_B d_n$.

It is easy to check that if A is an algebraic information system and B is a classical information system, then $C(A)$ is a classical information system, $S(B)$ is an algebraic information system, and our claims above hold.

3.2 Approximable Mappings and Continuous Functions

Information systems allow to think about domain elements as theories. Likewise, this approach allows to think about Scott continuous functions as special inference engines, which infer information about output, $f(x)$, from information about input, x .

Unfortunately, these *input-output inference engines* are called *approximable mappings* for their property to approximate one another. While this is an important property, Scott continuous functions thought of as graphs, or, in fact, any functions to domains approximate one another. Moreover, the “mappings” in question are not even functions. However, at this point of the development of the field one does not have much of a choice, but to follow the accepted terminology.

The following definition takes into account the intuition, that consistent information about input should produce consistent information about output, and that “native” inference relations of input and output domains can be used by an input-output inference engine.

Definition 3.4 Given two information systems, A and B , relation $f \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_B)$ is called an *input-output inference relation* or an *approximable mapping* between A and B , if the following axioms hold:

1. $\emptyset f \emptyset$ (non-triviality; minimalistic version of $uf\emptyset$);
2. $\forall u \in \mathcal{P}_{\text{fin}}(D_A). u \not\vdash_A \{\nabla_A\} \Rightarrow \neg(uf\{\nabla_B\})$ (preservation of consistency);
3. $\{\nabla_A\} f \{\nabla_B\}$ (inconsistency about an input allows to infer anything about the corresponding output);
4. $\forall u \in \mathcal{P}_{\text{fin}}(D_A), v_1, \dots, v_n \in \mathcal{P}_{\text{fin}}(D_B). uf v_1, \dots, uf v_n \Rightarrow uf(v_1 \cup \dots \cup v_n)$ (accumulation of output information, i.e. conjunction introduction);
5. $\forall u', u \in \mathcal{P}_{\text{fin}}(D_A), v, v' \in \mathcal{P}_{\text{fin}}(D_B). u' \vdash_A u, uf v, v \vdash_B v' \Rightarrow u' f v'$ (transitivity with “native” inference relations in A and B).

3.2.1 Scott Continuous Functions

An approximable mapping, f , between information systems A and B naturally gives rise to a function, $|f| : |A| \rightarrow |B|$, where $|f|(x)$ is computed by inferring all information from x using f .

We will see that $|f|$ is always Scott continuous, and that for any Scott continuous function $g : |A| \rightarrow |B|$ one can find an approximable mapping \widehat{g} between A and B , such that $|\widehat{g}| = g$, $|\widehat{f}| = f$. Together with the results about identity maps and composition this yields an equivalence between the category of algebraic information systems and approximable mappings and the category of algebraic Scott domains and Scott continuous functions.

Definition 3.5 Given an approximable mapping f between A and B , define the associated function $|f| : |A| \rightarrow |B|$ by formula $|f|(x) = \{d \in D_B \mid \exists u \subseteq x. uf\{d\}\}$.

Theorem 3.3 Function $|f|$ is correctly defined and Scott continuous.

Definition 3.6 Given a Scott continuous function $g : |A| \rightarrow |B|$, define the associated relation $\widehat{g} \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_B)$ by formula $u\widehat{g}v \Leftrightarrow \forall x \in |A|. u \subseteq x \Rightarrow v \subseteq g(x)$.

It is easy to check that \widehat{g} is an approximable mapping.

If $u \in \mathcal{P}_{\text{fin}}(D_A)$ is consistent, then $u\widehat{g}v \Leftrightarrow v \subseteq g(\bar{u})$, otherwise $u\widehat{g}v$ for all $v \in \mathcal{P}_{\text{fin}}(D_B)$.

This and the Scott continuity of g allow to establish easily, that $|\widehat{g}| = g$, $|\widehat{f}| = f$.

Moreover, $f \subseteq g$ iff $|f| \sqsubseteq_{[A \rightarrow B]} |g|$.

3.2.2 Identity and Composition

It is easy to see that $|\vdash_A| = id_A$, where $\forall x \in |A|. id_A(x) = x$.

Define the *composition* of approximable mappings $f \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_B)$ and $g \subseteq \mathcal{P}_{\text{fin}}(D_B) \times \mathcal{P}_{\text{fin}}(D_C)$ as $g \circ f = h \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_C)$, such that $uhw \Leftrightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_B). uf\{v\}, vgw$.

One can think of $g \circ f$ as an input-output inference engine obtained by hooking the input of engine g to the output of engine f : $\left(\leftarrow g \leftarrow f \leftarrow \right)$.

It is easy to check that $|g \circ f| = |g| \circ |f|$.

3.3 Functional Spaces

Consider algebraic information systems A and B . We are going to define an information system, $[A \rightarrow B]$, such that $|[A \rightarrow B]|$ would consist precisely of all approximable mappings between A and B .

Since there is an isomorphism between approximable mappings and Scott continuous functions, and since the set-theoretical inclusion of approximable mappings corresponds to the partial order on the domain of Scott continuous functions, $[|A| \rightarrow |B|]$, we will use the information system $[A \rightarrow B]$ to represent this functional space.

Take $D_{[A \rightarrow B]} = \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_B)$. Take $\nabla_{[A \rightarrow B]} = \langle \emptyset, \{\nabla_B\} \rangle$.

Say, that $\{\langle u_1, v_1 \rangle, \dots, \langle u_n, v_n \rangle\} \vdash_{[A \rightarrow B]} \{\langle u'_1, v'_1 \rangle, \dots, \langle u'_m, v'_m \rangle\}$, where $n, m \geq 0$, if

1. for any $i \in \{1, \dots, m\}$, $\bigcup_{\{j \in \{1, \dots, n\} \mid u'_i \vdash_A u_j\}} v_j \vdash_B v'_i$ or $u'_i \vdash_A \nabla_A$; or
2. $\exists I \subseteq \{1, \dots, n\}$. $\bigcup_{i \in I} u_i \not\vdash_A \nabla_A$, $\bigcup_{i \in I} v_i \vdash_B \nabla_B$.

We leave the necessary correctness checks to the reader.

3.4 Effective Domains and Computations

We say that an algebraic information system A is *effective*, if D_A and \vdash_A are recursively enumerable. The corresponding domain $|A|$ is called *effective* too.

An important example is produced by the *effective domain of arithmetic theories* in any of the usual systems of arithmetic. This example shows why our degree of generality is the right one.

Usually people give a more restrictive definition, which is equivalent to D_A and \vdash_A being recursive. One of the reasons for this is the inconvenience of the definition of a classical information system. Indeed, if A is an effective algebraic information system, then $\text{Con}'_{C(A)}$ is co-recursively enumerable, and $\vdash'_{C(A)}$ has to be defined effectively on a co-recursively enumerable domain of definition, which is not a trivial undertaking, and, in any case, the result would be awkward.

An element $x \in |A|$ is called *computable* if it is a recursively enumerable set. This condition is equivalent to the recursive enumerability of the set of compact elements approximating x .

If f is a computable element of a functional domain, $[|A \rightarrow B|]$, we call it a *computable function*. Observe that computable functions map computable elements to computable elements and, moreover, transform a recursive enumeration of x into the recursive enumeration of $|f|(x)$.

We should note here that the problems of the correctness of definition for effective domains and computable elements are not decidable in general. For example, it is often impossible to develop a procedure deciding whether a given effective domain is not empty, or whether a given recursive enumeration of a computable element does not contain ∇ .

An implementation of a computable element is some computational device, which recursively enumerates the tokens of this element. The issues of more effective implementation of some classes of computable elements (defined by some restricted classes of formulas) are quite important from the practical viewpoint.

4 Non-reflexive Logics for Non-algebraic Domains

In the previous section we saw that information systems based on the ordinary logic correspond to algebraic Scott domains. Hence, in order to generalize the logical approach to larger classes of dcpo's, one has to modify the logic of inference and/or the notion of theory.

In this section we concern ourselves with bounded complete dcpo's. Significant progress for some classes of domains which are not bounded complete was achieved by Abramsky [1].

This section develops our ideas from [4, 5] and Appendix 1 of [6]

4.1 Non-reflexive Logic

The seminal paper [9] by R. Hoofman generalized the logical approach to non-algebraic continuous domains. Hoofman replaced the ordinary reflexivity rule, $A \vdash A$, with a weaker property of *inverse modus ponens*: $A \vdash C \Rightarrow \exists B. A \vdash B, B \vdash C$. His paper is a well-written one, but at the same time the actual and simple reasons for his construction to work are hidden in its later sections. Thus, the average reader gets an impression that it is a miracle construction, and the real intuition behind this paper is lost.

In particular, another important feature, which allows to exploit non-reflexivity, is underemphasized. This feature is the change in the notion of a theory (i.e. domain element). A theory, x , is traditionally a consistent, deductively closed set of statements. Hoofman adds an additional requirement of *inverse deductive closeness*, which is trivial for a reflexive situation: $\forall d \in D_A. d \in x \Rightarrow \exists u \subseteq_{fin} x. u \vdash_A d$.

We will soon see, that this feature is essential for his approach, while the *inverse modus ponens rule* is important only in order to maintain continuity of the resulting domain. In fact, we generalize the results of [9] to arbitrary *spaces of fixed points of Scott continuous transformations of algebraic Scott domains*, by omitting both the inverse and the standard rule of *modus ponens*.

This will allow us to go beyond continuous domains, and potentially (modulo the conjecture in Section 4.3.1) to all bounded complete dcpo's.

4.1.1 Correspondence between Properties of Scott Continuous Functions and Inference Rules

Scott continuous functions are equivalent to input-output inference engines (approximable mappings). In particular, given an algebraic information system A , Scott continuous transformations $|f|$ of the corresponding domain $|A|$ are equivalent to the generalized inferences $f \subseteq \mathcal{P}_{fin}(D_A) \times \mathcal{P}_{fin}(D_A)$, which are transitive with the standard \vdash_A and respect consistency and inconsistency.

Reflexivity. Consider the reflexive $f, v \subseteq u \Rightarrow ufv$. Since $ufv, v \vdash_A w \Rightarrow ufw$, if f is reflexive, then from $\forall u \in \mathcal{P}_{fin}(D_A). ufu$ we can infer $\vdash_A \subseteq f$. Hence if f is reflexive, then $\forall x \in |A|. x \sqsubseteq_A |f|(x)$.

Conversely, $\forall x \in |A|. x \sqsubseteq_A |f|(x)$ implies that f is reflexive.

Transitivity. It is easy to see, that $\forall u, v, w \in \mathcal{P}_{\text{fin}}(D_A). ufv, vfw \Rightarrow ufw$ is equivalent to the condition $|f| \circ |f| \sqsubseteq_{[A \rightarrow A]} |f|$.

Inverse Transitivity. It is easy to see, that the Hoofman rule of inverse transitivity, $\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). ufw \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). ufv, vfw$ is equivalent to $|f| \sqsubseteq_{[A \rightarrow A]} |f| \circ |f|$.

Retractions. The previous two paragraphs imply that f is transitive and inversely transitive if and only if $|f|$ is a retraction: $|f| = |f| \circ |f|$.

Inverse Reflexivity. The rule $\forall x \in |A|. |f|(x) \sqsubseteq_A x$ is equivalent to the following rule of "inverse reflexivity": $ufv \Rightarrow u \vdash v$, which will be transformed into $ufv \Rightarrow v \subseteq u$ in some simple cases.

Closures and Projections. It is easy to see, that reflexivity implies inverse transitivity, and that inverse reflexivity implies transitivity.

Hence, the combination of reflexivity and transitivity yields precisely *closures* (such retractions $|f|$, that $x \sqsubseteq_A |f|(x)$), and the combination of inverse reflexivity and inverse transitivity yields precisely *projections* (such retractions $|f|$, that $|f|(x) \sqsubseteq_A x$).

4.1.2 Fixed Points as Theories

Consider the definition of $|f|$: $|f|(x) = \{d \mid \exists u \in x. uf\{d\}\}$. So $|f|(x)$ consists of tokens which are inferrable from x via inference engine f .

Hence, the deductive closeness of x with respect to inference f , $\forall u, v \in \mathcal{P}_{\text{fin}}(D_A). u \in x, ufv \Rightarrow v \in x$ is equivalent to $|f|(x) \sqsubseteq_A x$.

Similarly, the *inverse deductive closeness* of x with respect to inference f , namely $\forall v \in \mathcal{P}_{\text{fin}}(D_A). v \in x \Rightarrow \exists u \in \mathcal{P}_{\text{fin}}(D_A). u \in x, ufv$, is equivalent to $x \sqsubseteq |f|(x)$.

Hence, together the deductive closeness and the inverse deductive closeness of x with respect to inference f is equivalent to x being a fixed point of $|f|$: $|f|(x) = x$.

4.1.3 Domains of Fixed Points

The previous paragraph suggests the following procedure. Consider an algebraic information system A and replace its entailment relation with an approximable mapping f between A and A .

Then replace the notion of theory with the consistent subset $x \subseteq D_A$, such that x is deductively closed and inversely deductively closed with respect to f . The result is the domain $|A_f|$ of fixed points of continuous transformation $|f|$, $|A_f| = \text{Fix}(f) \subseteq |A|$.

We can call $|A_f|$ a *fixed-point subdomain* of $|A|$.

However, we want to introduce a more general notion of a fixed-point subdomain.

4.1.4 Adding the Top Element

Consider an algebraic information system $A = (D_A, \nabla_A, \vdash_A)$ and the corresponding domain $|A|$. Consider an element $\nabla_{A_\top} \notin D_A$ and define the algebraic information system A_\top as follows.

Take $D_{A_\top} = D_A \cup \{\nabla_{A_\top}\}$. Of course, we take ∇_{A_\top} as the ∇ of the new system. We set $u \vdash_{A_\top} v$ iff either $u \vdash_A v$ or $\nabla_{A_\top} \in u$.

Then $|A_\top| = |A| \cup \{\top\}$, where $\top = D_{A_\top} = \{\nabla_{A_\top}\}$ is the new top element.

4.1.5 General Notion of Fixed-Point Subdomain

In this subsection we consider approximable mappings as generalized entailment relations. Sometimes we would like an approximable mapping f to entail the contradiction from some of finite conjunctions of statements from D_A , noncontradictory under \vdash_A . In order to formalize such a situation we have to consider approximable mappings f from A to A_\top .

The general notion of a fixed-point subdomain of A is the set of fixed points of Scott continuous function $|f| : |A| \rightarrow |A_\top|$, $\text{Fix}(f) = \{x \in A \mid x = |f|(x)\}$. Equivalently, one can consider fixed points of Scott continuous transformations $|f|$ of domain A_\top , such that $|f|(\top) = \top$.

If, for the sake of tradition, one wants to impose the requirement that the subdomains are non-empty, one has to add the requirement $|f|(\perp) \neq \top$.

4.1.6 Application: Removing the Compact Top Element

Assume that an algebraic Scott domain $|B|$ has the compact top element $\top_B \neq \perp_B$ (in particular, $|A_\top|$ has the compact top element, $\{\nabla_{A_\top}\}$). Then consider $|f| : |B| \rightarrow |B_\top|$, such that $|f|(\top_B) = \top_{B_\top}$ and $|f|(x) = x$ for all other $x \in |B|$.

Then $\text{Fix}(f) = B \setminus \{\top_B\}$.

The compactness of \top_B is important, since a Scott continuous function on an algebraic Scott domain is completely defined by its values on compact elements.

In terms of entailment, $u f v$ iff either $u \vdash_B v$ or $\top_B \subseteq \bar{u}$. It is easy to see, that since $\top_B \neq \perp_B$, the resulting system $B_\top = (D_B, \nabla_B, f \cap (\mathcal{P}_{\text{fin}}(D_B) \times \mathcal{P}_{\text{fin}}(D_B)))$ is an algebraic information system, and $|B_\top| = B \setminus \{\top_B\}$.

4.1.7 Powersets and Qualitative Domains

Consider set D , such that $\nabla_A \notin D$, and define $D_A = D \cup \{\nabla_A\}$ and $u \vdash_A^m v$ iff $v \subseteq u$ or $\nabla_A \in u$. The resulting *minimal* algebraic information system $A^m = (D_A, \nabla_A, \vdash_A^m)$ defines the powerset of D as its domain $|A^m|$.

If we consider $W \subset \mathcal{P}_{\text{fin}}(D_A)$, such that $\emptyset \notin W$, $\{\nabla_A\} \in W$, and modify the previous construction so that $u \vdash_A v$ iff $v \subseteq u$ or $\exists w \in W. w \subseteq u$, then we obtain a *qualitative domain* (see [9]). We can modify the construction of

cutting the compact top element above to obtain any qualitative domain from the powerset domain.

4.1.8 General Notion of Finitary Information System And Finitary Scott Domain

Consider an algebraic information system $A = (D_A, \nabla_A, \vdash_A)$ and an approximable mapping f from A to A_\top . We agreed that f describes a fixed-point subdomain of $|A|$. Observe that f also describes a fixed-point subdomain of $|A^m|$. Moreover, the sets of fixed points of corresponding Scott continuous functions $|f| : |A| \rightarrow |A_\top|$ and $|f| : |A^m| \rightarrow |A_\top^m|$ coincide.

This leads us to a general definition of what we call a *finitary information system* and the definition of the corresponding *finitary Scott domain*. We will respect the rule $|f|(\perp) \neq \top$ meaning the non-emptiness of the resulting domains. These definitions will be respectively a streamlined equivalent of A^m together with an approximable mapping f between A^m and A_\top^m and a description of the set of fixed points of the corresponding function $|f|$.

Definition 4.1 The tuple $A = (D_A, \nabla_A, \vdash'_A)$, where D_A is a set of distinctive *tokens* (*elementary statements*), $\nabla_A \in D_A$ (the *false statement*), $\vdash'_A \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A)$ (the *entailment* relation) is called a *finitary information system* if

1. $\emptyset \not\vdash'_A \{\nabla_A\}$ (non-degeneracy; a calculus must admit at least one non-contradictory theory);
2. $\emptyset \vdash'_A \emptyset$ (non-triviality);
3. $\forall u, v_1, \dots, v_n \in \mathcal{P}_{\text{fin}}(D_A). u \vdash'_A v_1, \dots, u \vdash'_A v_n \Rightarrow u \vdash'_A (v_1 \cup \dots \cup v_n)$ (accumulation of output information);
4. $\forall u', u, v, v' \in \mathcal{P}_{\text{fin}}(D_A). u' \subseteq u, u' \vdash'_A v', v \subseteq v' \Rightarrow u \vdash'_A v$;
5. $\forall u, v \in \mathcal{P}_{\text{fin}}(D_A). u \vdash'_A \{\nabla_A\} \Rightarrow u \vdash'_A v$;
6. $\forall u, v \in \mathcal{P}_{\text{fin}}(D_A). \nabla_A \in u \Rightarrow u \vdash'_A v$.

Definition 4.2 The *finitary Scott domain* $|A|$ associated with a finitary information system A is the set of theories,

$$\{x \subseteq D_A \mid$$

1. $u \subseteq x, u \vdash'_A v \Rightarrow v \subseteq x$ (x is deductively closed);
2. $v \subseteq x \Rightarrow \exists u \in \mathcal{P}_{\text{fin}}(D_A). u \vdash'_A v, u \subseteq x$ (x is inversely deductively closed);
3. $\nabla_A \notin x$ (deductively closed x is consistent) }.

In order to check the correctness of our discourse, one has to consider information systems $A^m = (D_A, \nabla_A, \vdash_A^m)$ and the corresponding system A_\top^m . Given a finitary information system $A = (D_A, \nabla_A, \vdash_A')$, one should define $f \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A \cup \{\nabla_{A_\top^m}\})$ via $ufv \Leftrightarrow u \vdash_A' v$ or $\nabla_A \in u$ and establish that f is an approximable mapping, such that $\neg(\emptyset f \{\nabla_{A_\top^m}\})$. Then given an approximable mapping f between A^m and A_\top^m , such that $\neg(\emptyset f \{\nabla_{A_\top^m}\})$, one should define \vdash_A' via $u \vdash_A' v \Leftrightarrow ufv$ and $\nabla_{A_\top^m} \notin v$ and establish that $A = (D_A, \nabla_A, \vdash_A')$ is a finitary information system.

Finally one should observe that we have just defined one-to-one correspondence between all possible entailment relations in finitary information systems $(D_A, \nabla_A, \vdash_A')$ and all approximable mappings between A^m and A_\top^m , such that $\neg(\emptyset f \{\nabla_{A_\top^m}\})$, and that under that correspondence finitary Scott domains are exactly the sets of fixed points of the respective functions $|f|$.

4.1.9 Domains of Fixed-Point Subdomains

In the subsections 4.1.5 and 4.1.8 we defined a finitary information system and the corresponding finitary Scott domain of fixed points of $|f|$ for any approximable mapping f , such that $\neg(\emptyset f \{\nabla_{A_\top}\})$, between any algebraic information system $A = (D_A, \nabla_A, \vdash_A)$ and the corresponding system A_\top .

The approximable mapping f , such that $\emptyset f \{\nabla_{A_\top}\}$, corresponding to Scott continuous function $|f|$, such that $|f|(\perp) = \top_{A_\top}$, is the top compact element of the domain $[A \rightarrow A_\top]$, and, hence, it can be removed by the technique described above, yielding the algebraic Scott domain $[[A \rightarrow A_\top]_{\mathcal{F}}]$.

Because these mappings f are in one-to-one correspondence with fixed-point subdomains of $|A|$, we can take the domain $[[A \rightarrow A_\top]_{\mathcal{F}}]$ as representing fixed-point subdomains of $|A|$. Of course, different f might have the same set of fixed points, and hence the corresponding fixed-point domains will coincide as sets, but their underlying entailment relations will differ, so they should be considered different subdomains.

Hence we call $[[A \rightarrow A_\top]_{\mathcal{F}}]$ the *domain of fixed-point subdomains* of $|A|$.

4.1.10 Discussion

One should observe that f is not an arbitrary new entailment relation, but is closely related to the original \vdash_A , namely f is transitive with respect to \vdash_A due to the axioms of approximable mappings. This property is responsible for the fact that all elements of the resulting fixed-point subdomain belong to the original domain.

We can think about a fixed-point subdomain as the result of some elements of the original domain being destroyed. There are three mechanisms of such a destruction. An element can lose its deductive closeness or consistency. When these two mechanisms are involved, we still remain within the realm of reflexive logic and algebraic subdomains result. This important case is covered in details in Section 5.

The third mechanism of destruction of elements of the original domain is the loss of inverse deductive closeness. When this happens we, in general, go beyond ordinary reflexive logic. However, there are cases of non-reflexive f 's, where we still can remain within the realm of reflexive logic. The price for this is the distortion of the resulting subdomain: instead of the proper fixed-point subdomain we only obtain a domain isomorphic to this fixed-point subdomain. One such case, namely the case of finitary retractions, is considered in details in Section 6.

We could have considered a different definition of a fixed-point subdomain of $|A|$, namely any finitary information system determined by arbitrary approximable mapping f from A^m to A^m , such that $\text{Fix}(|f|)$ is a non-empty subset of $|A|$. Then, however, it is unlikely that we could form a domain of such subdomains, although I did not try to prove such a result. Even if we could do so, such domains of subdomains would not have to be isomorphic for isomorphic domains $|A|$ and $|B|$ if the cardinalities of D_A and D_B differ. Hence this alternative definition is unsatisfactory.

4.1.11 Results for Continuous Scott Domains

Now the results on continuous information systems by Hoofman can be easily explained. He retained transitivity of f and replaced its reflexivity by inverse transitivity. This resulted in $|f|$ being a retraction.

It is well known that retractions of continuous lattices are continuous lattices, and that all continuous lattices can be obtained as retractions of powersets, and these results can be generalized for continuous Scott domains, since the only thing which distinguishes them from continuous lattices is that continuous Scott domains do not have to possess the top element.

These facts explain the success of Hoofman's program for continuous Scott domains.

4.2 Infinitary Logic

In [16] Sazonov and Sviridenko introduced another generalization of logic. They kept reflexivity and transitivity, but removed the finitariness requirement that an infinite set of statements x only infers those statements, which the finite subsets $u \subseteq x$ infer.

Their approach describes exactly all bounded complete partial orders. Directed completeness is equivalent to the following requirement of *partial finitariness*: a set of statements x cannot infer the contradiction, unless one of its finite subsets $u \subseteq x$ infers the contradiction.

Hence partially finitary systems of Sazonov and Sviridenko exactly correspond to complete Scott domains.

Sazonov and Sviridenko also gave characterizations of some subclasses of complete Scott domains in their system, in particular, they described a class of their logic corresponding to continuous Scott domains.

4.3 Open Issues

4.3.1 The Class of Finitary Scott Domains

What is the class of domains described as fixed-point domains is an important open question. It is well known that the set of fixed points of a Scott continuous transformation of a complete lattice is a complete lattice. It was traditionally thought that the converse is also true, namely that any complete lattice can be obtained as a set of fixed points of a Scott continuous transformation of a sufficiently large powerset. For example, Exercise 18.4.3(ii)(1) on page 491 of the famous textbook on lambda calculus by Barendregt [7] asks to establish this for the case of countable bases of the respective Scott topologies.

Hence we expected that our approach would allow us to obtain all complete Scott domains. We thus expected to follow the terminology of continuous information systems by Hoofman and call the fixed-point information systems developed in this section *complete information systems*.

However, our analysis of literature and numerous conversations with experts in the field convinced us that the problem stated in the textbook by Barendregt is, in fact, open. Our attempts to solve it were, so far, unsuccessful. Hence we opted for the less committing terminology of *finitary information systems* and *finitary Scott domains*.

4.3.2 Important Subclasses

What subclasses of finitary domains will be obtained, if we require f to be transitive or, more strongly, inversely reflexive? The conjecture is that we still obtain all finitary Scott domains as sets of fixed points of the corresponding functions $|f|$.

Transitivity is, of course, a very desirable property of a logical system, so it is of interest to check whether it restricts generality. These questions come in at least two flavors: for f defining a continuous transformation of $|A^m|$ and for f defining a continuous transformation of an arbitrary algebraic Scott domain $|A|$.

4.3.3 Approximable Mappings And Other Issues for Finitary Information Systems

Hoofman successfully introduced the notion of continuous approximable mapping for continuous information systems based on the fact, that given spaces A and B and their retractions $r_A : A \rightarrow A$ and $r_B : B \rightarrow B$, one can build a retraction $(A \rightarrow B) \rightarrow (A \rightarrow B)$, namely $f \mapsto r_B \circ f \circ r_A$. Scott continuous functions $f : A \rightarrow B$, such that $f = r_B \circ f \circ r_A$, are in one-to-one correspondence with Scott continuous functions $\text{Fix}(r_A) \rightarrow \text{Fix}(r_B)$.

Unfortunately, no such simple solution seems possible for finitary information systems. Yet, a satisfactory notion of finitary approximable mapping seems to be necessary for further successful studies of finitary information systems.

There is a possibly simpler variant of this problem if f is required to be transitive.

4.3.4 Possible Translation between Nonreflexive and Nonfinitary Logics

In [16] Sazonov and Sviridenko gave a translation between their logic and the logic of Hoofman for continuous Scott domains.

Naturally there is a question whether such a translation is possible for a larger class of domains.

It also might be of interest to combine the features of nonreflexive and nonfinitary logic in one system.

5 Subdomains for the Algebraic Case

This section represents our results obtained in 1986-1988 and presented in [3]. With the introduction of our new version of the definition of algebraic information system and our ideas in nonreflexive logic the presentation is greatly simplified.

For the duration of this section “domain” means algebraic Scott domain, “subdomain” means algebraic Scott subdomain, and “information system” means algebraic information system.

5.1 The Brief History of the Question

In this section, we study the question “Which subsets of a domain should be considered subdomains?” for the domains described by algebraic information systems. One criterion for a good answer is a requirement that all subdomains of domain $|A|$ should themselves form a domain, $|Sub(A)|$. We would also like to be able to solve domain equations such as $|X| \cong |Sub(X)| + \dots$. It will be shown in the next few pages that the framework of domains as abstract cpo’s does not provide sufficient hints for the solution, but information systems do.

The systematic use of the operation of taking a certain subset of a domain to form another domain and the systematic consideration of domains, whose elements are domains, first appear in the mid-’70’s [17, 19]. It is notable that nobody has tried to consider arbitrary subsets of domain $|A|$ that would satisfy a particular version of domain axioms, although at the first glance this would seem to be the most intuitive and general version of a subdomain definition in the framework of domains as abstract cpo’s. The reason why nobody has considered such a definition is that such arbitrary subsets might satisfy the domain axioms for random causes, and their collection would be totally unmanageable; in particular, one cannot hope to form a domain, $|Sub(A)|$, with these subsets as elements.

The next idea, which originated by Dana Scott in [17], is to define a subdomain as a set of fixed points of a *retraction* belonging to a certain class. There

are two problems here: a) What are the reasons for using retractions? b) What is the appropriate class of retractions? None of these problems receives a clear answer in the framework of domains as abstract cpo's. On the use of retractions Scott writes: "it seems almost an accident that the idea [to use the retractions] can be applied" (see [17], p.540).

For the algebraic case, Scott suggests using *closure operations*, that is, retractions $|r|$, such that $|r| \sqsupseteq id$. This suggestion can be motivated by the theorem (see [17], Theorem 5.1) saying that the fixed points of a closure operation [on an algebraic lattice] form an algebraic lattice. Although this is not true for arbitrary retractions or projections (i.e. retractions $|r|$, such that $|r| \sqsubseteq id$), if we consider *finitary* retractions or projections (*finitary* means algebraicity of the set of fixed points), then this argument no longer favors closure operations.

In the paper entitled "Data Types as Objects" [19], Shamir and Wadge suggest that to describe the systems of polymorphic types, one should consider data types that incorporate their own subtypes as elements, and allow an element of a data type to belong simultaneously to different subtypes of this data type.

Let us briefly present the main features of subtypes in [19]. The subtype $|Y| = \{x \in |D| \mid x \sqsubseteq y\}$ is associated with each element y of a data type $|D|$. We can view such subtype as the set of fixed points of the (finitary, in the algebraic case) projection $x \mapsto x \sqcap y$. Subtypes are ordered simply by inclusion. Thus, this simple approach yields the result that $|D| \cong |Sub(D)|$ for every domain $|D|$. Each element represents the subtype of its approximations, and an element x belongs to all subtypes $|Y|$ of $|D|$ that are represented by y 's, such that $x \sqsubseteq y$.

At the end of the Introduction to [3], we discussed the rather adverse relations between retraction-based approaches to subtyping ([17, 19], and this section are based on retractions) and the dominant form of modern theories of typing that deals with types of intensional objects (the type polymorphism in ML is, probably, the most widely known example).

5.2 Algebraic Subdomains Correspond to Closures

When we considered this problem in [3], we were using classical information systems. We wrote: "Let us consider informally, what should we do to an information system A to obtain a subdomain $|B| \subseteq |A|$. We would like to destroy some elements of $|A|$. The elements are consistent and deductively closed sets of assertions in A . Therefore, some elements of $|A|$ must lose their consistency or deductive closure to be destroyed. Also notice, that the loss of deductive closure means intensification of entailment."

Then we figured out, that inconsistency can be treated as entailment of contradiction in A_{\top} , and hence the loss of consistency can be also treated as strengthening of entailment. The need to keep considering separately entailment and consistency in a number of technical situations, nevertheless, complicated our whole discourse.

With the definition of algebraic information system introduced in this Thesis, the matters become much simpler.

Definition 5.1 We say that for algebraic information systems A and B , domain $|B|$ is an (*algebraic*) *subdomain* of domain $|A|$ if $|B| \subseteq |A|$.

Lemma 5.1 (Contravariance) For algebraic information systems A and B , such that $D_A = D_B$, $|B| \subseteq |A|$ iff $\vdash_A \subseteq \vdash_B$.

It is easy to see that the case when $D_A \neq D_B$ does not lead to new domains $|B|$, so we do not consider this case further.

Since every algebraic information system is a finitary information system, the discourse from the previous section applies. Namely, we can see that if $\vdash_A \subseteq \vdash_B$, the approximable mapping f from A to A_\top , which corresponds to \vdash_B , is reflexive.

Since B is an algebraic information system, the restriction of f to $A \rightarrow A$ approximable mapping (or its extension to $A_\top \rightarrow A_\top$) is also transitive, so the corresponding restriction or extension of $|f|$ would be a closure. Also the non-triviality condition $|f|(\perp_A) \neq \top_{A_\top}$ holds.

Definition 5.2 An approximable mapping f from an algebraic information system A to the algebraic information system A_\top is called a *generalized non-trivial closure*, if

1. $\forall u, v \in_{fin} \mathcal{P}_{fin}(D_A). u \vdash_A v \Rightarrow uf v$ (reflexivity of the restriction on A);
2. $\forall u, v, w \in_{fin} \mathcal{P}_{fin}(D_A). uf v, vfw \Rightarrow ufw$ (transitivity of the restriction on A);
3. $\neg(\emptyset f \{\nabla_{A_\top}\})$ (non-triviality);

Lemma 5.2 Such generalized non-trivial closures from $[[A \rightarrow A_\top]]$ are in one-to-one correspondence with algebraic subdomains of $|A|$. Moreover, for closures f and g and the corresponding subdomains $|F|$ and $|G|$, $|F| \subseteq |G|$ iff $g \subseteq f$.

5.3 A Metatheorem on Reflexive Transitive Closure

We are about to build the domain of algebraic subdomains of an algebraic Scott domain $|A|$ as an algebraic subdomain of the algebraic Scott domain $[[A \rightarrow A_\top]]$. Our new setting allows us to do it in a systematic way compared to [3].

Definition 5.3 Consider the minimal algebraic information system $A^m = (D_A, \nabla_A, \vdash_A^m)$ and any binary relation $R \subseteq \mathcal{P}_{fin}(D_A) \times \mathcal{P}_{fin}(D_A)$. Then define the *reflexive transitive closure* $\vdash_A^R \subseteq \mathcal{P}_{fin}(D_A) \times \mathcal{P}_{fin}(D_A)$ as follows. For any $u, v \in \mathcal{P}_{fin}(D_A)$, we say that $u \vdash_A^R v$, if there is a finite sequence $v_1, \dots, v_n \in \mathcal{P}_{fin}(D_A)$, such that $v = v_n$ and the following condition holds:

Denote $u_0 = u$ and, inductively, $u_i = u_{i-1} \bigcup v_i$ for all $i \in \{1, \dots, n\}$. Then for any $i \in \{1, \dots, n\}$, we require that either $\nabla_A \in u_{i-1}$, or $v_i \subseteq u_{i-1}$, or there is $u' \subseteq u_{i-1}$, such that $u' R v_i$.

Definition 5.4 Consider the minimal algebraic information system $A^m = (D_A, \nabla_A, \vdash_A^m)$ and any binary relation $R \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A)$. We say that a set $x \subseteq D_A$ is a *theory with respect to R* , if x is consistent, that is $\nabla_A \notin x$, and closed under R , that is if $u \subseteq x$ and uRv , then $v \subseteq x$.

Theorem 5.1 Consider the minimal algebraic information system $A^m = (D_A, \nabla_A, \vdash_A^m)$ and any binary relation $R \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A)$. If there is at least one closed theory x with respect to R , then $A^R = (D_A, \nabla_A, \vdash_A^R)$ is an algebraic information system and domain $|A^R|$ is equal to the set of all theories closed with respect to R .

Corollary 5.1 Consider an algebraic information system $A = (D_A, \nabla_A, \vdash_A)$ and any binary relation $R \subseteq \mathcal{P}_{\text{fin}}(D_A) \times \mathcal{P}_{\text{fin}}(D_A)$. Define binary relation $R' = R \cup \vdash_A$. Then if domain $|A|$ contains at least one element which is a theory closed with respect to R , then domain $|A^{R'}|$ consists of those elements of the domains $|A|$, which are theories closed with respect to R .

5.4 The Domain of Algebraic Subdomains

We are now ready to study the domain of algebraic subdomains of an algebraic Scott domain $|A|$. We denote this domain $|Sub(A)|$. This domain will consist of all generalized non-trivial closures from $[[A \rightarrow A_\top]]$. These closures will represent the corresponding algebraic subdomains of $|A|$.

Theorem 5.2 The set $|Sub(A)|$ of generalized non-trivial closures from $[[A \rightarrow A_\top]]$ is an algebraic subdomain of the algebraic Scott domain $[[A \rightarrow A_\top]]$. Hence, $|Sub(A)|$ is an algebraic Scott domain itself.

Proof. Consider the algebraic information system $F = [A \rightarrow A_\top]$. We need to define the relation R , which would axiomatize properties of generalized non-trivial closures.

Consider the following three sets:

$$R_r = \{\langle \emptyset, \{\langle u, v \rangle\} \rangle \mid u \vdash_A v, u, v \in \mathcal{P}_{\text{fin}}(D_A)\}.$$

$$R_t = \{\langle \{\langle u, v \rangle, \langle v, w \rangle\}, \{\langle u, w \rangle\} \rangle \mid u, v, w \in \mathcal{P}_{\text{fin}}(D_A)\}.$$

$$R_n = \{\langle \emptyset, \nabla_{A_\top} \rangle, \langle \nabla_F \rangle\}.$$

They axiomatize, respectively, reflexivity, transitivity, and non-triviality.

Take $R = R_r \cup R_t \cup R_n$.

The set \vdash_A is an approximable mapping from $|F| = [[A \rightarrow A_\top]]$. (Remark: $|\vdash_A|$ is the embedding $\lambda x.x$). \vdash_A is also a theory with respect to R . Hence, the corollary from the previous section implies, that domain $|F^{R'}|$ consists of those elements of the domains $|F|$, which are theories closed with respect to R . It is easy to see that these elements are precisely all generalized non-trivial closures.

Hence, the information system $F^{R'}$ is exactly the information system $Sub(A)$ we have been looking for.

□

Theorem 5.3 For any $b, c \in |Sub(A)|$, $b \sqsubseteq_{Sub(A)} c$ iff for the corresponding algebraic subdomains $|B|$ and $|C|$, $|C| \subseteq |B|$.

The idea behind this result is simple: the stronger the entailment, i.e. the larger the generalized closure is, the fewer elements of $|A|$ remain intact (consistent and deductively closed).

This differs favorably from [19] and from the other approaches to subdomains as sets of fixed points of projections, where subdomains are ordered by inclusion. Indeed, the smaller a subdomain is as a set, the more information we have about each of its elements, and hence, the larger this subdomain should be informationally (*integers* \sqsupseteq *reals* is desirable).

Theorem 5.4 The continuous functions $|i| : |A| \rightarrow |Sub(A)|$ and $|j| : |Sub(A)| \rightarrow |A|$ defined below yield a projection of $|Sub(A)|$ onto $|A|$. In terms of closure operations

$$|i|(y) = \lambda x : |A|.x \sqcup_{A_\top} y;$$

$$|j|(r) = |r|(\perp).$$

In terms of subdomains themselves

$$|i|(y) = \{x \in |A| \mid x \sqsupseteq y\};$$

$$|j|(S) = \min_{|A|} S, \text{ where } S \text{ is a subdomain of } |A|.$$

Notice that some light non-triviality is involved in the construction of the embedding $|i| : |A| \rightarrow |Sub(A)|$. All one-element subsets of $|A|$ form subdomains (the corresponding generalized closure is $|r_y| = \lambda x : |A_\top|.if\ x \sqsubseteq y\ then\ y\ else\ \top_{A_\top}$). But due to Theorem 5.3, they are all incomparable (in fact, they are total elements in $|Sub(A)|$). Therefore, the function $x \mapsto \{x\}$ is not monotonic. This implies that this function is not Scott continuous and cannot be used as an embedding.

5.5 Open Issues

5.5.1 Reflexive Domain of Algebraic Subdomains

Because $|A|$ and $|Sub(A)|$ form an embedding-projection pair, we can use the inverse limit construction to obtain solutions of certain domain equations involving *Sub* [12, 13, 18], e.g. $|X| \cong |Sub(X)|$, $|X| \cong |Sub(X)| + |A|$, etc. Because *Sub*(A) describes theories of all possible ways to intensify our means of entailment in A , it might be of interest to study *explicitly* the iterations of this construction, $Sub(Sub(A))$, $Sub^3(A)$, etc., and their limit D , where $|D| \cong |Sub(D)|$.

5.5.2 Relationship between Subdomains and Subtypes

In [3] we expressed the opinion that compile-time and inheritance-oriented approaches to types, like subclasses in object-oriented programming and variants of strongly normalizing typed lambda-calculi, are incompatible with retraction-based subtyping in domain theory. This opinion was further confirmed by our reading of Section 10.4.4 on partial equivalence relations as types for systems

with subtyping in the textbook by Mitchell [14]. Mitchell explains that in such systems entailment relation for the supertype should, in general, be stronger, than entailment for the subtype. This and next sections show that this is impossible for retractions yielding algebraic domains, at least as long as we stay with the reflexive transitive finitary logic.

Thus the issue of developing practical programming language with types conforming to “data types as objects” paradigm [19], where types are runtime first-class objects, and any compile-time type checking and inference is considered to be an optimizing partial evaluation, remains an important open issue. As long as denotational semantics of such a language is expressible via algebraic domains, the notion of subtyping should probably correspond to the notion of subdomain developed in this section.

6 Finitary Retractions and Projections

This section represents our results obtained in 1986-1988 and presented in [4]. With the introduction of our new version of the definition of algebraic information system and our ideas in nonreflexive logic this material became much more transparent.

In general, sets of fixed points of retractions of algebraic Scott domains form continuous Scott domains. In this section we study the classes of finitary retractions and projections.

Definition 6.1 A retraction or projection of an algebraic Scott domain $|r| : |A| \rightarrow |A|$ is called *finitary* if its set of fixed points is algebraic.

For the duration of this section “domain” means algebraic Scott domain, “subdomain” means algebraic Scott subdomain, and “information system” means algebraic information system.

6.1 Characterization of the Classes of Finitary Retractions and Projections

6.1.1 Finitary Retractions

Lemma 6.1 *A retraction $|r|$ of an algebraic Scott domain $|A|$ is finitary if and only if there is an algebraic Scott domain $|B|$ and Scott continuous functions $|i| : |B| \rightarrow |A|$ and $|j| : |A| \rightarrow |B|$, such that $\langle |i|, |j| \rangle$ is an embedding-retraction pair, such that $|r| = |i| \circ |j|$.*

Proof. It is enough to establish, that the embedding $\text{Fix}(|r|) \rightarrow |A|$ and the map $R : |A| \rightarrow \text{Fix}(|r|)$, where $R(x) = |r|(x)$ for all $x \in |A|$, are Scott continuous. This, in turn, follows from the fact that given a directed set S of fixed points of $|r|$, the least upper bound of S in $|A|$ is also a fixed point of $|r|$.

□

Definition 6.2 We say that a retraction $|r|$ of an algebraic Scott domain $|A|$ possesses the *intermediate reflexive property* if

$$\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). \text{urw} \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). \text{urv}, \text{vrw}, \text{vrw}.$$

Lemma 6.2 Consider a retraction $|r|$ of an algebraic Scott domain $|A|$ possessing the intermediate reflexive property. Define $A^r = (D_A^r, \nabla_A^r, \vdash_A^r)$ as follows. Let $D_A^r = \{u \in \mathcal{P}_{\text{fin}}(D_A) \mid \text{uru}\}$. Let $\nabla_A^r = \{\nabla_A\}$. Let $\{u_1, \dots, u_n\} \vdash_A^r \{v_1, \dots, v_m\}$ iff $(u_1 \cup \dots \cup u_n)r(v_1 \cup \dots \cup v_m)$ for all $u_1, \dots, u_n, v_1, \dots, v_m \in D_A^r$. Then A^r is an algebraic information system and $|A^r| \cong \text{Fix}(|r|)$.

Proof. Checking the axioms of algebraic information system for A^r is straightforward. Observe that reflexivity of \vdash_A^r follows from our selection of only such elements $u \in \mathcal{P}_{\text{fin}}(D_A)$ as members of D_A^r , that uru .

Now we use the intermediate reflexive property of r . For any $x \in \text{Fix}(|r|)$, the set $\{u \in D_A^r \mid u \subseteq x\}$ belongs to $|A^r|$. For any $y \in |A^r|$, the set $\bigcup\{u \mid u \in y\}$ is a fixed point of $|r|$. These are monotonic injective maps. This establishes the desired isomorphism. □

Lemma 6.3 Given an algebraic information system A and an approximable mapping r from A to A defining a retraction $|r|$, the following conditions are equivalent:

1. $|r|$ is finitary.
2. $\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). \text{urw} \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). \text{urv}, \text{vrw}, \text{vrw}.$
3. $\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). \text{urw} \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). \text{urv}, \text{vrw}, v \vdash_A w.$

Proof. $3 \Rightarrow 2$ follows from r being an approximable mapping, and $2 \Rightarrow 3$ follows from considering $v' = v \cup w$ and observing that $\text{urv}, \text{vrw}, \text{vrw}$ implies $\text{urv}', v'rv', v' \vdash_A w$.

$2 \Rightarrow 1$ follows from the previous lemma.

Here we prove $1 \Rightarrow 2$. Consider an algebraic information system B and approximable mappings i and j from the Lemma 6.1. Consider $u, w \in \mathcal{P}_{\text{fin}}(D_A)$, such that urw . Then there is $v' \in \mathcal{P}_{\text{fin}}(D_B)$, such that ujv' and $v'iw$, because $r = i \circ j$. Because $j \circ i = \vdash_B$, there is $v \in \mathcal{P}_{\text{fin}}(D_A)$, such that $v'iv$ and $v'v'$. Then urv, vrw , and vrw . □

6.1.2 Finitary Projections

Lemma 6.4 Given an algebraic information system A and an approximable mapping p from A to A , the following conditions are equivalent:

1. $|p|$ is a finitary projection.

2. $\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). upw \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). u \vdash_A v, vpv, v \vdash_A w.$

Proof. $1 \Rightarrow 2$ follows from the previous lemma and the fact that for a projection $upv \Rightarrow u \vdash_A v.$

To prove $2 \Rightarrow 1$, observe that in the condition 2 upv and vpw , so $|p|$ is a finitary retraction, and $u \vdash_A w$, so $|p|$ is a projection.

□

6.2 Domains of Fixed Points of Finitary Projections and Retractions from the Viewpoint of Logic of Fixed Points

We could have used our technique of fixed-point subdomains to describe the set of fixed points of an arbitrary or finitary retraction or projection. However, since arbitrary retractions and projections do not possess reflexivity property, this would move us outside of the realm of algebraic information system even in the finitary case.

Instead, we use the fact that finitary retractions are exactly those possessing the *intermediate reflexive property* that if $\forall u, w \in \mathcal{P}_{\text{fin}}(D_A). urw \Rightarrow \exists v \in \mathcal{P}_{\text{fin}}(D_A). urv, vrv, vrw.$ Informally, this means that there are sufficiently many finite conjunctions v , which are reflexive (vrv) with respect to the retraction r , considered as prospective entailment relation, so that any entailment done by r can be done *via* such a reflexive conjunction v . These reflexive conjunctions serve as a backbone of the new algebraic information system A^r describing the domain isomorphic to the domain of fixed points of $|r|$.

Then, in the case of r defining a projection, we build the domain $|A^r|$ via mechanism of *conjunctive completion* and of *forgetting* the statements, not reflexive with respect to r . In the case of r defining a general finitary retraction, we also have to replace the native entailment by the one induced by r (Lemma 6.2). It is precisely the *forgetting* step, which makes it impossible to consider sets of fixed points of finitary projections and retractions as algebraic subdomains, as we will see later in this section.

6.2.1 Conjunctive Completeness

We say that algebraic information system A is (*finitely*) *conjunctively complete*, if for any $u \in \mathcal{P}_{\text{fin}}(D_A)$, there is $d \in D_A$, such that $\{d\} \vdash_A u$ and $u \vdash \{d\}.$

The step $D_A^r = \{u \in \mathcal{P}_{\text{fin}}(D_A) \mid uru\}$ in Lemma 6.2 provides for *conjunctive completion*. Actually, $A_A^{\vdash_A}$ yield precisely the conjunctive completion of algebraic information system A , and $|A_A^{\vdash_A}| \cong |A|.$

It is actually enough to add conjunctions only for such $u \in \mathcal{P}_{\text{fin}}(D_A)$, that uru and there is no such $d \in D_A$, that $\{d\} \vdash_A u$ and $u \vdash \{d\},$ so if we start from a conjunctively complete system to begin with, the conjunctive completion step can be omitted.

6.2.2 Domains of Fixed Points of Finitary Projections

When p defines a finitary projection, the construction of Lemma 6.2 is rewritten as follows.

An algebraic information system $A^p = (D_A^p, \nabla_A^p, \vdash_A^p)$ is defined by $D_A^p = \{u \in \mathcal{P}_{\text{fin}}(D_A) \mid upu\}$, $\nabla_A^p = \{\nabla_A\}$, $\{u_1, \dots, u_n\} \vdash_A^p \{v_1, \dots, v_m\}$ iff $(u_1 \cup \dots \cup u_n) \vdash_A (v_1 \cup \dots \cup v_m)$ for all $u_1, \dots, u_n, v_1, \dots, v_m \in D_A^p$.

As before, $|A^p| \cong \text{Fix}(|p|)$.

If the information system A was conjunctively complete to begin with, the construction above could be modified as follows:

$D_A^p = \{d \in D_A \mid dpd\}$, $\nabla_A^p = \nabla_A$, $u \vdash_A^p v \Leftrightarrow u \vdash_A v$ for all $u, v \in \mathcal{P}_{\text{fin}}(D_A^p)$.

Then for any $x \in \text{Fix}(|p|)$, the set $x \cap D_A^p$ would belong to $|A^p|$. For any $y \in |A^p|$, the deductive closure of y in A is a fixed point of $|p|$. This establishes the desired isomorphism. In this case, for any $x \in |A|$, $|p|(x)$ is the deductive closure of $x \cap D_A^p$ in A .

6.2.3 Why Finitary Projections Do Not Form Algebraic Subdomains

We give two examples. The first example illustrates the need for conjunctive completeness or conjunctive completion. The second example actually shows, why forgetting does not allow us to consider sets of fixed points of finitary projections to be subdomains. In both examples we only list domain elements and sets of fixed points and leave it to the reader to restore the actual information systems and projections.

Consider the domain $|A| = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$. Consider the projection $|p|$ of $|A|$ onto $\{\emptyset, \{1, 2\}\}$. While there is an information system describing the domain $\{\emptyset, \{1, 2\}\}$, it cannot be obtained by a projection from our original domain, because the original domain is not conjunctively complete. The first version of our construction would allow us to obtain the domain isomorphic to $\text{Fix}(|p|)$ as $|A^p| = \{\emptyset, \{\{1, 2\}\}\}$ via a conjunctive completion.

On the other hand, if we consider an isomorphic conjunctively complete situation, $\{\emptyset, \{1\}, \{2\}, \{1, 2, t\}\}$ and its projection onto $\{\emptyset, \{1, 2, t\}\}$, the second version of our construction of A^p would yield $\{\emptyset, \{t\}\}$.

The set $\{\emptyset, \{1, 2\}\}$ in the previous example can still be obtained as a subdomain of $|A|$, although not via a projection. Now we consider a different example, namely $|A| = \{\emptyset, \{1\}, \{1, 2\}, \{1, 3\}\}$, and its projection $|p|$ onto $\{\emptyset, \{1, 2\}, \{1, 3\}\}$. The set $\{\emptyset, \{1, 2\}, \{1, 3\}\}$ is not a domain determined by any algebraic information system (consider, what would be $\overline{\{1\}}$). The set $\{\emptyset, \{1, 2\}, \{1, 3\}\}$ is not a subdomain of $|A|$ and cannot be obtained from $|A|$ via a closure operation (consider, where would $\{1\}$ go under a closure). This is a conjunctively closed situation, and the second version of our construction of A^p yields $|A^p| = \{\emptyset, \{2\}, \{3\}\}$.

The last example shows that sometimes a set of fixed points of a finitary projection is not equal to any domain described by an algebraic information system and, hence, cannot be considered an algebraic subdomain of the original domain. In other cases, like our first example, the set in question is an algebraic

subdomain, but the closure operation describing it does not have anything in common with the finitary projection in question.

6.2.4 Domains of Fixed Points of Generalized Nontrivial Finitary Retractions

In order to fully incorporate the discourse of the previous two sections into the framework of finitary retractions, one has to consider *generalized nontrivial retractions* $|r| : |A| \rightarrow |A_\top|$, such that $|r|(\perp) \neq \top_{A_\top}$ and $|r|(|r|(x)) = |r|(x)$, when $|r|(x) \neq \top_{A_\top}$.

In order to use the setup of Lemma 6.1, one has to consider retraction $|r'| : |A_\top| \rightarrow |A_\top|$, such that $|r'|(\top_{A_\top}) = \top_{A_\top}$ and $|r'|(|r|(x)) = |r|(x)$ for $x \in |A|$. Notice that $|B|$ has the compact top element, and that $\text{Fix}(|r|) \cong |B_\top|$.

Definition 6.2 can be extended to the generalized retraction $|r|$ without change.

Lemma 6.2 and its proof are applicable to the generalized nontrivial retraction $|r|$ without change as well.

Lemma 6.3 also holds for $|r|$, but in order to establish $1 \Rightarrow 2$ in its proof, one should consider $|r'|$.

Lemma 6.3 gives the criterion of finitariness of generalized nontrivial retractions, and Lemma 6.2 yields the construction of the domain isomorphic to the domain of fixed points of a generalized nontrivial finitary retraction.

6.2.5 Other Criteria of Finitarity

The following criterion of finitariness for projections was known before. A projection $|p| : |A| \rightarrow |A|$ is finitary iff $\forall x \in |A|. x = |p|(x) \Rightarrow x = \sqcup\{x_0 \in |A|_0 \mid x_0 = |p|(x_0), x_0 \sqsubseteq x\}$. This criterion can be obtained from Lemma 1(ii) of [10], which in effect says that the set of finite elements in $\text{Fix}(|p|)$ is $\{x_0 \in |A|_0 \mid x_0 = |p|(x_0)\}$.

The consideration of information systems easily produces a much nicer criterion based on the intermediate reflexive property (Lemma 6.4). It does not require consideration of arbitrary non-finite elements x and allows to select finitary projections from the space of all continuous functions rather than from the space of projections. It can be literally rewritten in terms of abstract cpo's: $\forall x_0, z_0 \in A_0. z_0 \sqsubseteq |p|(x_0) \Rightarrow \exists y_0 \in A_0. x_0 \sqsupseteq y_0 \sqsupseteq z_0, y_0 = |p|(y_0)$. Yet, one is unlikely to choose this criterion if he does not consider information systems because it involves inequality $z_0 \sqsubseteq |p|(x_0)$. Also notice that in this case we have to consider $|p|(x_0)$, which is generally not finite.

For retractions, the criterion based on the intermediate reflexive property is given by Lemma 6.3. In terms of abstract cpo's our criterion can be rewritten as follows: a retraction $|r| : |A| \rightarrow |A|$ is finitary iff $\forall x_0, z_0 \in A_0. z_0 \sqsubseteq |r|(x_0) \Rightarrow \exists y_0 \in A_0. y_0 \sqsubseteq |r|(x_0), y_0 \sqsubseteq |r|(y_0), z_0 \sqsubseteq |r|(y_0)$.

The finite elements of $\text{Fix}(|r|)$ are obtained as images of the finite elements of $|A|$ under $|r|$. Hence, in the style close to [10], another criterion can be written

as follows: a retraction $|r| : |A| \rightarrow |A|$ is finitary iff $\forall x \in |A|. x = |r|(x) \Rightarrow x = \sqcup\{|r|(x_0) \mid x_0 \in |A|_0, x_0 \sqsubseteq |r|(x_0) \sqsubseteq x\}$.

6.3 Domains of Finitary Projections and Generalized Finitary Retractions

6.3.1 Domain of Finitary Projections

The set of all finitary projections $|A| \rightarrow |A|$ can be obtained as the set of fixed points of the finitary projection $|Pr|$ of $|A \rightarrow A|$. Specifically, if $g \in |A \rightarrow A|$ define $f = |Pr|(g)$ by ufw iff $\exists v \in \mathcal{P}_{\text{fin}}(D_A). u \vdash_A v, vgv, v \vdash_A w$.

It is easy to check that f is a finitary projection, and that if g is a finitary projection then $g = |Pr|(g)$. To prove that $|Pr|$ itself is a continuous function and a finitary projection, one should notice that $\{(u_1, w_1), \dots, (u_n, w_n)\}Pr\{(u, w)\}$ iff $\exists v. u \vdash_A v, v \vdash_A w, \{(u_1, w_1), \dots, (u_n, w_n)\} \vdash_{A \rightarrow A} (v, v)$.

6.3.2 Domain of Generalized Finitary Retractions

In a similar fashion the set of all generalized nontrivial finitary retractions can be obtained as a set of fixed points of a generalized nontrivial finitary retraction $|Ret|$ of $|A \rightarrow A_{\top}|$. Hence this set is an algebraic Scott domain.

6.4 Omitted Issues

The details of construction of $|Ret|$ is omitted in the present text. This construction is the result of mating the construction of $|Pr|$ with the construction of the domain of subdomains.

6.4.1 Limits of Projective Sequences

It is possible to generalize the construction of the limits of some projective sequences of domains defined by algebraic information systems, $\{\langle |A_n|, |A_{n+1}|, |i_n|, |j_n| \rangle \mid |i_n| : |A_n| \rightarrow |A_{n+1}|; |j_n| : |A_{n+1}| \rightarrow |A_n|; \langle |i_n|, |j_n| \rangle \text{ is an embedding-projection pair; } n = 1, \dots\}$, from [12, 13] to arbitrary sequences of this kind. We do not give the details of the generalized construction here.

6.5 Open Issues

6.5.1 Issues Related to the Domain of Finitary Retractions

I do not know, whether $|Ret|$ is unique.

The question might be somewhat related to the properties of the space of all retractions, say, of a powerset. This space is a complete lattice, but it is not algebraic and not even continuous.

6.5.2 Other Subclasses of Finitary Retractions

Here we look at two special subclasses of the class of finitary retractions, which may deserve special attention.

Let us look at the criteria for finitariness of retractions and projections once more. For retractions it is $urw \Rightarrow \exists v. urv, vrv, vrw$, which can be rewritten as $urw \Rightarrow \exists v. urv, vrv, v \vdash w$. For projections it is $upw \Rightarrow \exists v. u \vdash v, vrv, v \vdash w$. There is an intermediate case — retractions satisfying the property that $urw \Rightarrow \exists v. u \vdash v, vrv, vrw$. This class of retractions includes both finitary projections and closure operations, and the retractions from this class possess a certain weak property of minimality, hence they can be called *quasi-minimal*.

Another class is obtained from the observation that while for finitary projections finite elements of sets of fixed points are finite in the original domain too, this does not generally hold for finitary retractions. We can call the retractions possessing this property *strongly finitary*. The class of strongly finitary retractions includes the class of finitary projections, but does not include the class of closure operations.

Both quasi-minimal retractions and strongly finitary retractions may be worth more detailed studies.

Acknowledgements

I am especially thankful to Alexander Saevsky and Vladimir Sazonov for extremely valuable discussions. I would also like to acknowledge helpful suggestions of Carl Gunter, Michael Huth, and other colleagues.

References

- [1] Abramsky S. Domain theory in logical form. *Annals of Pure and Applied Logic*, **51** (1991), 1–77.
- [2] Abramsky S. and Jung A. Domain Theory. In S. Abramsky, D. Gabbay, T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science Volume 3*, pages 1–168, Oxford University Press, 1994.
- [3] Bukatin M.A. Subdomains for Algebraic Information Systems, November, 1994.
Available via URL <http://www.cs.brandeis.edu/~bukatin/Nov94.ps.gz>
- [4] Bukatin M.A. Bukatin M.A. Information Systems and Retractions: an Elementary Treatment, December, 1994.
Available via URL <http://www.cs.brandeis.edu/~bukatin/Dec94.ps.gz>
- [5] Bukatin M.A. Information Systems and Complete Lattices. Abstract of the talk presented at the Eleventh Summer Conference on General Topology

and Applications, August 10 - 13, 1995, University of Southern Maine Gorham, ME, USA.

Available via URL <http://at.yorku.ca/cgi-bin/amca/caae-10>

- [6] Bukatin M.A. *Continuous Functions as Models for Programs: Mathematics and Applications*. PhD Thesis Proposal, Brandeis University, 1996.
Available via URL <http://www.cs.brandeis.edu/~bukatin/draft.ps.gz>
- [7] Barendregt H.P. *The Lambda Calculus : Its Syntax And Semantics*. North-Holland, 1984.
- [8] Edalat A. and Smyth M. I-Categories as a Framework for Solving Domain Equations. *Theoretical Computer Science*, **115** (1993), 77–106.
- [9] Hoofman R. Continuous information systems. *Information and Computation*, **105** (1993), 42–71.
- [10] Huth M. *Cartesian closed categories of domains and the space Proj(D)*. Lecture Notes in Computer science, Vol.598, pp.259-271, Springer-Verlag, 1991.
- [11] Jung A., Kegelmann M., Moshier M.A. Multi Lingual Sequent Calculus and Coherent Spaces. *Fundamenta Informaticae*, **37** (1999), 399–412.
- [12] K. G. Larsen and G. Winskel. Using information systems to solve recursive domain equations effectively. In D. B. MacQueen, G. Kahn, G. Plotkin, editors, *Semantics of Data Types*, pages 109–130, Berlin, 1984. Springer-Verlag. Lecture Notes in Computer Science Vol. 173.
- [13] Kim Guldstrand Larsen and Glynn Winskel. Using information systems to solve recursive domain equations. *Information and Computation*, 91(2):232–258, April 1991.
- [14] John C. Mitchell. *Foundations for Programming Languages*. MIT Press, 1996.
- [15] Rothe M. Retraktionen auf stetigen Bereichen. Diplomarbeit. Fachbereich Mathematik. Technische Universität Darmstadt. April 1991.
- [16] Vladimir Sazonov and Dmitri Sviridenko. Abstract deducibility and domain theory. DIMACS Technical Report 96-08, Rutgers University, 1996. Published as DIMACS Technical Report in 1996.
Available via URL
<ftp://dimacs.rutgers.edu/pub/dimacs/TechnicalReports/TechReports/1996/96-08.ps.gz>
- [17] Scott D. S. *Data types at lattices*. SIAM Journal on Computing, **5**, #3, pp.522-587, 1976.

- [18] Dana S. Scott. Domains for denotational semantics. In M. Nielsen and E. M. Schmidt, editors, *Automata, Languages and Programming*, volume 140 of *Lecture Notes in Computer Science*, pages 577–613. Springer-Verlag, 1982.
- [19] Shamir A., Wadge W.W. *Data types as objects*. Lecture Notes in Computer Science, Vol.52, pp.465-479, Springer-Verlag, 1977.
- [20] Smyth M. Effectively Given Domains. *Theoretical Computer Science*, **5** (1977), 257-274.
- [21] Vickers S. Information System for Continuous Posets. *Theoretical Computer Science*, **114** (1993), 201–229.