Minimal Representation of Type-Hierarchies

Richard Elling Moe
University of Bergen
Department of Information Science and Media Studies

Abstract: We discuss the interpretation and representation of type-hierarchies in the form of bounded complete partial orders, with typing-functions. Within the framework of representation theory we present a technique for cheap representation which is optimal under certain conditions.

Key–Words: Type-hierarchies, Representation theory

1 Introduction

There is a widespread use of type-hierarchies in computer- and information sciences, both in theory and applications. In addition comes a rich selection of applications owing to the great number of other disciplines that make use of such structures. Accordingly, the representation and processing of hierarchical order has received much attention [13; 1; 5; 7; 10; 12; 14].

The literature varies in terms of what constitutes a hierarchy. When specified, we find that they sometimes amount to trees [8] or even total orders [14], but typically they are something between preorders [10] and lattices [9].

We shall concentrate on the structure used in [1; 2; 3; 6]: The bounded complete partial order is more restricted than pre-orders, but not quite a lattice. The main operations on such structures include the usual Join and Meet and sometimes typing functions which associate each entity with a unique type.

A type-hierarchy could be represented in a number of ways, all depending on its intended use. We base our discussion on representation theory [4] where types are interpreted as sets such that the hierarchical structure can be observed through the subset relation.

We aim to present a criterion for economical representation of type-hierarchies and identify conditions under which it is optimal.

1.1 Preliminaries

Assuming that the reader is familiar with the basic concepts concerning partially ordered sets [4] and the standard terms that accompany them, we only summarize some notation and terminology.

We write \( (A, \leq) \) to express that the set \( A \) is partially ordered by the relation \( \leq \subseteq A \times A \). The relation \( Min_\leq(X, a) \) holds iff \( a \) is minimal in \( X \). Similarly, \( Max_\leq(X, a) \) iff \( a \) is maximal in \( X \). If \( X \) has a unique minimal element we refer to it as the least element, and denote it by \( min_\leq(X) \). Similarly, when a greatest element of \( X \) exists it is denoted by \( max_\leq(X) \).

We say that \( a' \) is an upper cover for \( a \) if \( a < a' \) and there is no \( a'' \in A \) such that \( a < a'' < a' \). Likewise, \( a \) is a lower cover of \( a' \). Two elements \( a \) and \( a' \) are said to be comparable iff either \( a \leq a' \) or \( a' \leq a \). A subset \( S \) of \( A \) is said to be consistent if every finite subset of \( S \) has an upper bound in \( A \). We adopt the convention of letting \( \sqcup S \) denote the join of \( S \), i.e. the least upper bound (provided, of course, it exists). Similarly, the meet-function \( \sqcap \) returns the greatest lower bound.

Partially ordered sets can be compared using the notions of monotone maps and isomorphisms. A monotone map preserves the order, while an isomorphism reflects an exact correspondence between partial orders. Suppose \( \langle A, \leq \rangle \) and \( \langle B, \succeq \rangle \) are partially ordered sets: A monotone map from \( A \) to \( B \) is a total function \( h : A \rightarrow B \) such that for all \( a, a' \in A \), \( a \leq_A a' \) implies \( h(a) \leq_B h(a') \). An isomorphism between \( A \) and \( B \) is a total surjective function \( h : A \rightarrow B \) such that for all \( a, a' \in A \), \( a \leq_A a' \) iff \( h(a) \leq_B h(a') \).

We adopt Carpenter’s definition [1] of a type-hierarchy.

Definition 1

- A bounded complete partial order (BCPO) is a partial order \( \langle A, \leq \rangle \) in which every consistent subset of \( A \) has at least upper bound. In particular, the empty set is regarded as consistent, having a bottom element \( \bot \) as its least upper bound.

- A type-hierarchy is a finite BCPO. That is, the set \( A \) is finite. We then refer to \( \leq \) as the subsumption relation.

Compare the BCPOs with the so called consistently complete CPOs [4] and you will find that they amount...
to the same thing. Sometimes they are referred to as meet semi-lattices.

2 Typed domains

In this section we address the question of how to impose the structure of the hierarchy onto a domain using the techniques of representation theory. Similar approaches have been taken in connection with a variety of ordering-relations [4; 10; 11; 9]. Carpenter [1], however, does not explain his type-hierarchies in these terms and some variations found in the literature is left unclear in his exposition. For instance, should inconsistent types necessarily be associated with disjoint sets? [9] has this requirement, whereas [10; 11] do not. We shall discuss this distinction.

In the following we consider, unless otherwise specified, an arbitrary type-hierarchy \( H = \langle T Y P E, \sqsubseteq \rangle \).

We now develop a representation theory for type-hierarchies. We begin with the basic idea of interpretation in terms of sets.

**Definition 2** Let \( D \) be a set. A \( D \)-interpretation of a type-hierarchy \( H \) is a function \( f : T Y P E \to 2^D \) assigning a subset of \( D \) to every type. Let \( D^f_\sigma \) denote the set assigned to a type \( \sigma \) by \( f \).

It can be convenient to illustrate interpretations of simple type-hierarchies by projecting the Euler-diagram of the interpreting sets onto the Hasse-diagram representing the type-hierarchy, as shown in figure 1.

Given a \( D \)-interpretation \( f \) of a type-hierarchy \( H \) we regard the partially ordered set \( \langle \{ D^f_\sigma \mid \sigma \in T Y P E \}, \sqsubseteq \rangle \) as a representation of \( H \). However, interpretation does not in itself provide good representation. To obtain close correspondence between the type-hierarchy and its interpretation we need additional requirements. We then refer to the representation as a typed domain.

**Definition 3** A \( D \)-interpretation \( f \) defines an \( H \)-typed domain iff the following conditions are satisfied:

- \( D^f_\bot = D \)
- \( \sqcap \Sigma = \tau \) implies \( \bigcap_{\sigma \in \Sigma} D^f_\sigma = D^f_{\tau} \)

Figure 2 illustrates a typed domain whereas figures 1 and 4 show three different typed domains over the same hierarchy.

On this approach, \( \sigma \sqsubseteq \tau \) implies \( D^f_\sigma \subseteq D^f_{\tau} \). That is, a \( D \)-interpretation \( f \) defining an \( H \)-typed domain will be a monotone map from \( H \) to its representation. Hence, the ordering-relation is preserved.

On the other hand, distinct types may be indistinguishable and there is no guarantee that the operations of the original type-hierarchy are preserved. The representation need not even be a BCPO. Under these circumstances we can not expect a representation to be isomorphic to the type-hierarchy. For some applications such underdeveloped representation can be appropriate since ‘small’ structures, that does not involve the entire hierarchy, may be built from too few elements to form distinct interpretations for all the types [6]. Even so, a typed domain may reflect the concept of types poorly. In order to remedy this, we tighten definition 3 so that a representation necessarily forms a CCPO isomorphic to the given type-hierarchy.

**Definition 4** A \( D \)-interpretation \( f \) defines a faithfully \( H \)-typed domain iff the following conditions are satisfied:

- \( D^f_\bot = D \)
- \( \sqcup \Sigma = \tau \) iff \( \bigcap_{\sigma \in \Sigma} D^f_\sigma = D^f_{\tau} \)

Figures 1(b) and 4 illustrates faithfully typed domains, whereas 1(a) does not. As a special case of definition 4 we get \( \sigma \sqsubseteq \tau \) iff \( D^f_\sigma \subseteq D^f_{\tau} \). Then \( f \) is necessarily injective: if \( D^f_\sigma = D^f_{\tau} \) then both \( \sigma \subseteq \tau \) and \( \tau \subseteq \sigma \) and hence \( \tau = \sigma \).

Inconsistency of types appears as a non-existing join in a type-hierarchy \( H \), whereas in a faithfully \( H \)-typed domain defined by \( f \), this inconsistency is reflected by their intersection not being assigned to a type by \( f \). Note also that in order to satisfy definition 4, \( f \) can assign the empty set to no type other than a top element of \( H \).
A faithfully typed domain is isomorphic to the hierarchy it represents: If \( f \) is a \( D \)-interpretation defining a faithfully \( H \)-typed domain then \( \langle \text{TYPE}, \sqsubseteq \rangle \) and \( \langle \{ D_\sigma \mid \sigma \in \text{TYPE} \}, \sqsupseteq \rangle \) are isomorphic. Furthermore, in the representation join corresponds to intersection whereas the meet of a set \( X \) is

\[
\min_{\subseteq}\{ Y \mid Y \in \{ D_\sigma \mid \sigma \in \text{TYPE} \}, Y \supseteq \bigcup_{x \in X} x \}.
\]

Having established the particulars of the CCPO induced on a domain by an interpretation satisfying definition 4, we present the technical vocabulary for talking about parts of a typed domain or the addition of elements: Specifically, we need the dual concepts of expansion/subinterpretation.

**Definition 5** Let \( f \) be a \( D \)-interpretation of \( H \). An expansion of \( f \) is a \( D' \)-interpretation \( f' \) such that

- \( D \subseteq D' \)
- \( d \in D_\sigma \) iff \( d \in D'_\sigma \) for all \( d \in D \) and \( \sigma \in \text{TYPE} \).

We say that \( f \) is a subinterpretation of \( f' \).

It is easily verified that being a typed domain is a uniform property in the sense that if \( f \) defines a faithfully \( H \)-typed domain then its subinterpretations define \( H \)-typed domains.

### 3 Typing functions

In addition to the basic operations of meet and join, type-hierarchies are often associated with a typing-function \([1; 3]\). In this section we develop this idea for typed domains. In principle, any function that maps nodes to types may pass as a typing function, but most of these are of little use if the typed domains are to reflect the concepts of types and inheritance.

Obviously, an appropriate typing function should map a node to a type whose interpretation contains the node. Typically, typing functions adhere to the following modelling convention: 'According to this convention, the nodes ... are taken to represent objects, and we assume that every node is labelled with a type symbol which represents the most specific conceptual class to which the object is known to belong' [1]. However, as long as an element may belong to the interpretations of incomparable types there need not be a unique 'most specific conceptual class'. Then two options seems to be open: either some kind of overriding choosing between multiple maximal types or, alternatively, opting for a less specific compromise. Since override strategies will typically depend on the application, we concentrate on the second approach: For any element \( x \), the typing function picks a single designated type, referred to as the type of \( x \). Specifically, the greatest type comparable with every other type whose interpretation contains \( x \).

**Definition 6** Let \( f \) be a \( D \)-interpretation that defines an \( H \)-typed domain. The typing function \( \theta : D \rightarrow \text{TYPE} \) is defined by

\[
\theta(x) = \max_{\subseteq}\{ \sigma \mid x \in D_\sigma \}, \text{for every type } \tau:
\]

\[
x \in D_\tau \text{ implies } \sigma \sqsubseteq \tau \text{ or } \tau \sqsubseteq \sigma
\]

It is easily verified that the typing function is total.

The typing function is illustrated in figure 2, where the interpretation of \( \sigma_5 \) is the intersection between the interpretations of \( \sigma_2 \) and \( \sigma_3 \). Note that \( d_2 \) is contained in the interpretations of both \( \sigma_2 \) and \( \sigma_3 \), neither of which is more suited than the other to be its designated type. As a 'compromise', \( \theta \) assigns \( d_2 \) to \( \sigma_1 \)

![Figure 2: \( \theta(d_1) = \sigma_2 \), \( \theta(d_2) = \sigma_1 \), \( \theta(d_3) = \sigma_4 \), \( \theta(d_4) = \sigma_3 \)](image)

An element \( x \) is said to be characteristic of a type \( \sigma \) iff \( \theta(x) = \sigma \), but there is no guarantee that a type has any characteristic elements at all.

**Definition 7** Let \( \sigma \in \text{TYPE} \) and let \( f \) define an \( H \)-typed domain with corresponding typing function \( \theta \).

- \( \mathcal{C}^f(\sigma) = \{ x \mid \theta(x) = \sigma \} \) is the set of characteristic elements of \( \sigma \).
- \( \mathcal{S}^f(\sigma) = \{ \tau \mid \sigma \sqsubseteq \tau \text{ and } \mathcal{C}^f(\tau) \text{ is nonempty} \} \) is the set of characteristic subtypes of \( \sigma \).

### 4 Saturated typed domains

In this section we consider how to obtain faithfulness by expansion while preserving the elements’ memberships in type-interpretations. That is, the transformation consists entirely in adding elements. We
introduce a simple criterion for faithfulness, taking care to restrict the number of added elements for the purpose of discussing minimal representation of type-hierarchies.

**Definition 8** A typed domain defined by $f$ is said to be saturated iff $C^f(\sigma) \neq \emptyset$ whenever $\sigma$ has exactly one upper cover or Max$_\subseteq$(TY PE, $\sigma$), except possibly when $\sigma = \top$.

In other words, a typed domain is saturated when every type with less than two upper covers, except possibly $\top$, has characteristic elements.

Now our goal is to show that a saturated typed domain is necessarily a faithfully typed domain. The crux of the argument lies in the following lemma.

**Lemma 9** Let $f$ define a saturated $H$-typed domain. If $\tau' \nsubseteq \tau$ then there is $\sigma \in S^f(\tau)$ such that $\tau' \nsubseteq \sigma$.

**Proof** Suppose $\tau' \nsubseteq \tau$. In order to establish a contradiction we assume that the lemma does not hold:

For every $\sigma \in S^f(\tau)$: $\tau' \nsubseteq \sigma$.  

This assumption implies the following proposition:

For every positive integer $n$ there is a set $\Pi_n \subseteq TY PE$ such that $|\Pi_n| \geq n$ and for all $\pi_i \in \Pi_n$:

a) $\tau \subseteq \pi_i$

b) $C^f(\pi_i) = \emptyset$

c) $\tau' \nsubseteq \pi_i$

The proof is by induction on $n$.

**Basis** $n = 1$: Let $\Pi_1 = \{\tau\}$. a) holds since $\tau \subseteq \tau$. By (1) $\tau'$ subsumes all types in $S^f(\tau)$. Then $\tau$ can hold have no characteristic elements, since $\tau' \nsubseteq \tau$, i.e. $C^f(\tau) = \emptyset$, so b) holds. Finally, c) holds by our initial assumption.

**Step (see figure 3 for an illustration):** Assume as the induction hypothesis that the proposition is valid for $n = k$ and let $\Pi_k$ be the set of types it prescribes.

Let $\pi_k$ be such that Max$_\subseteq(\Pi_k, \pi_k)$. Note that $\pi_k$ is not a top element because the induction hypotheses asserts that $\tau' \nsubseteq \pi_k$.

Since $C^f(\pi_k) = \emptyset$, it follows from definition 8 that $\pi_k$ has (at least) two upper covers $\pi$ and $\pi'$. Then there are $\sigma$ and $\sigma'$ such that Min$_\subseteq(S^f(\pi), \sigma)$ and Min$_\subseteq(S^f(\pi'), \sigma')$, and hence $\{\sigma, \sigma'\} \subseteq S^f(\tau)$.

It can not be the case that both $\tau' \subseteq \pi$ and $\tau' \subseteq \pi'$. To see this, there are two cases to consider. First, if $\pi_k \subseteq \tau'$ then we must have $\pi_k \subseteq \tau'$ since we have assumed that $\tau' \nsubseteq \pi_k$. Then, $\tau'$ is subsumed by some upper cover of $\pi_k$ and thereby $\tau'$ cannot subsume any of the other upper covers of $\pi_k$. Hence, either $\tau' \nsubseteq \pi$ or $\tau' \nsubseteq \pi'$. Secondly, if $\pi_k \nsubseteq \tau'$, i.e. $\pi_k$ and $\tau'$ are incomparable, then we cannot have both $\tau' \subseteq \pi$ and $\tau' \subseteq \pi'$ because then both $\pi_k$ and $\tau'$ would be lower bounds of $\{\pi, \pi'\}$ and since neither $\tau' \subseteq \pi_k$ nor $\pi_k \subseteq \tau'$ there would have to be a greatest lower bound, $\gamma$, of $\{\pi, \pi'\}$ such that $\gamma \subseteq \pi$, $\gamma \subseteq \pi'$ and $\pi_k \supseteq \gamma$, which is in conflict with $\pi$ and $\pi'$ being upper covers of $\pi_k$. Consequently, either $\tau' \nsubseteq \pi$ or $\tau' \nsubseteq \pi'$.

As both cases are similar, we can assume that $\tau' \nsubseteq \pi$. By assumption (1), $\tau' \subseteq \pi$, so $\pi \neq \pi'$ and there must be an upper cover $\pi''$ of $\pi'$ such that $\pi'' \subseteq \pi$. Therefore, since $\min_{\subseteq}(S^f(\pi), \sigma)$, $\pi$ can have no characteristic elements, that is, $\pi \not\subseteq S^f(\pi)$. Now, by the induction hypothesis, $\tau \subseteq \pi_k$. Furthermore, $\pi$ is an upper cover of $\pi_k$ so $\tau \subseteq \pi$ and hence, $C^f(\pi) = \emptyset$.

![Figure 3: the induction step](image)

By now we know that:

- $\tau \subseteq \pi$, since $\tau \subseteq \pi_k$, by the induction hypotheses, and $\pi$ is an upper cover of $\pi_k$.

- $\tau' \nsubseteq \pi$ and

- $C^f(\pi) = \emptyset$.

Moreover $\pi \not\in \Pi_k$ since Max$_\subseteq(\Pi_k, \pi_k)$ and $\pi_k \subseteq \pi$. Hence, $\Pi_{k+1} = \Pi_k \cup \{\pi\}$ satisfies the requirements a), b) and c) above. This concludes the proof of proposition (2).

The proposition states that there is a set $\Pi_n$ for every positive integer $n$, even for $n$ greater than the size of $TY PE$. But this is impossible. $\Pi_n$ contains at least $n$ distinct types and hence requires more than the available number of types. We have obtained a contradiction. Consequently we reject assumption (1) and conclude that the lemma holds.

Armed with lemma 9 it is easily verified that for any $f$ defining a saturated $H$-typed domain: If $\tau$ and $\sigma$ are incomparable then neither $D^f_{\tau} \subseteq D^f_\sigma$ nor $D^f_\sigma \subseteq D^f_{\tau}$.
\(D_f\). Furthermore, if \(\sigma \preceq \tau\) then \(D_f^\sigma \subset D_f^\tau\). It follows that \(f\) is injective.

We are ready to verify that a saturated typed domain is also a faithfully typed domain.

**Lemma 10** A \(D\)-interpretation \(f\) that defines a saturated \(H\)-typed domain also defines a faithfully \(H\)-typed domain.

**proof** Assuming that \(\bigcap_{\sigma \in \Sigma} D_f^\sigma = D_f^\tau\) it will suffice to show that \(\cup \Sigma = \tau\). Then \(\cup \Sigma\) must be defined. Suppose first, for the purpose of contradiction, that \(\cup \Sigma\) is undefined. Let \(\tau'\) be a maximal element such that \(\tau \subseteq \tau'\). As \(\cup \Sigma\) is undefined there can be no top element, so by definition 8, there should be characteristic elements of \(\tau'\). But, by definition 3, \(D_f^\tau' \subseteq D_f^\tau\) so \(D_f^\tau' \subseteq D_f^\sigma\) for all \(\sigma \in \Sigma\). Furthermore, since \(\tau'\) is maximal and \(\Sigma\) has no upper bound, \(\tau'\) can not be comparable with every \(\sigma \in \Sigma\). Hence, there can be no \(d \in D_f^\tau\) such that \(\theta(d) = \tau'\). This contradicts that \(f\) defines a saturated typed domain and we may conclude that \(\cup \Sigma\) is defined.

Now, by definition 3, \(D_f^{\cup \Sigma} = \bigcap_{\sigma \in \Sigma} D_f^\sigma\). Then, by assumption, \(D_f^{\cup \Sigma} = D_f^\tau\). Since \(f\) is injective, \(\cup \Sigma = \tau\).

If the type-hierarchy \(H\) happens to be a total order then any faithfully \(H\)-typed domain is saturated, but the converse of lemma 10 does not hold in general. Figure 4 shows a faithfully typed domain that is not saturated since the maximal type \(\tau_2\) lacks a characteristic element.

The transformation of a typed domain into a faithfully typed domain is now a straightforward task. One simply adds the elements it takes to make it saturated.

**Definition 11** Let \(f\) be a \(D\)-interpretation that defines an \(H\)-typed domain, \(\{\sigma_1, \ldots, \sigma_n\} = \{\sigma \mid \sigma \neq \top, C_f(\sigma) = \emptyset, \sigma\) has less than two upper covers\} and let \(D' = D \cup \{d_1, \ldots, d_n\}\) where \(d_i \notin D\).

A \(D'\)-interpretation \(f'\) is a saturation of \(f\) iff the following holds:

- For all \(d \in D\) and \(\sigma \in TYPE\): \(d \in D_f'^\sigma\) iff \(d \in D_f^\sigma\).

- For all \(i \in \{1, \ldots, n\}\): \(d_i \in D_f'^\sigma\) iff \(\sigma \subseteq \sigma_i\).

By construction, the saturation \(f'\) is clearly an expansion of \(f\) defining a saturated typed domain. This, together with lemma 10, shows that every typed domain is a subinterpretation of a faithfully typed domain. Actually, the converse is easily shown to be true [6], i.e. \(f\) defines an \(H\)-typed domain iff it is a subinterpretation of an \(f'\) defining a faithfully \(H\)-typed domain.

Suppose that \(f\) defines an \(H\)-typed domain, then there are certainly expansions of \(f\) that define a saturated typed domain without being a saturation of \(f\). However, since saturation according to definition 11 adds no more elements than necessary it is clear that no other saturated expansion involves fewer new elements:

**Lemma 12** Consider \(f\) and \(f'\) of definition 11 and let \(f'' : TYPE \rightarrow 2^{|D''|}\) be any expansion of \(f\) defining a saturated \(H\)-typed domain. Then \(|D''| \leq |D'|\).

Saturation is fairly economical with regard to the number of new elements added in order to obtain a faithfully typed domain. However it is not optimal. Let \(f\) be the \(D\)-interpretation that defines the typed domain of figure 1(a). Note that \(D_f^{\tau_2} = \emptyset\). The saturation of \(f\) is shown in figure 1(b), but it has strictly more elements than the faithful expansion of \(f\) shown in figure 4.

![Figure 4: A faithfully, but not saturated, typed domains.](image)

Figure 4: A faithfully, but not saturated, typed domains.

## 5 Consistently typed domains

In this section we investigate typed domains under the requirement that that the interpretations of inconsistent types should be disjoint. (Like for instance Smolka [9] does.) We then refer to domains as being consistently typed.

**Definition 13** Let \(f\) be a \(D\)-interpretation that defines an \(H\)-typed domain. \(f\) is said to define a consistently \(H\)-typed domain iff for every inconsistent \(\Sigma \subseteq TYPE, \bigcap_{\sigma \in \Sigma} D_f^\sigma = \emptyset\).

Figure 1(a) and (b) illustrate consistently typed domains whereas and figures 2 and 4 do not.
In consistently typed domains, an element can not be associated with inconsistent types, hence the typing function need not resort to overriding and ‘compromise’ types, as explained in section 3. In fact, it is easily verified that the typing function of definition 6 boils down to $\theta(x) = \max_{\pi} \{\sigma \mid x \in D^\pi_{\sigma}\}$ [6] which corresponds precisely to the standard modelling convention.

Confinement to consistently typed domains does not affect our use of saturation. This is because saturation of a typed domain according to definition 11 by construction preserves consistency. That is, the saturation of a consistently typed domain is also consistently typed.

In section 4, we showed that a saturated typed domain was necessarily a faithfully typed domain. We also illustrated, in figure 4, that a faithfully typed domain need not be saturated. However, the example relies on intersecting interpretations of inconsistent types. When only consistently typed domains are considered, faithfulness and saturatedness becomes the same thing.

**Theorem 14** A consistently typed domain is faithful iff it is saturated.

**proof** The $If$-part is shown directly by lemma 10.

In order to prove the Only $If$-part, we let $f$ be a $D$-interpretation defining a consistently and faithfully $H$-typed domain. We must verify that for each $\sigma \neq \top$ with less than two upper covers, there is $d \in D$ such that $\theta(d) = \sigma$.

For the purpose of contradiction we assume that the typed domain defined by $f$ is not saturated:

$$\text{There is a type } \pi \neq \top \text{ with less than two upper covers such that for all } d \in D: \quad \theta(d) \neq \pi.$$

There are two cases to consider:

1. Suppose that $\pi$ has no upper covers, that is $\max_{\pi} \{TY PE, \pi\}$. As only a top element can be interpreted as the empty set, $D^\pi_{\top} \neq \emptyset$. Let $d \in D^\pi_{\top}$. By (1), maximality of $\pi$ and the definition of $\theta$ there must be a type $\tau$ incomparable with $\pi$ such that $d \in D^\pi_{\tau}$. In fact, $\tau$ and $\pi$ must be inconsistent since $\pi$ is maximal. But this is impossible since $f$ defines a consistently typed domain.

2. Suppose that $\pi$ has exactly one upper cover $\tau$. Let $d \in D^\pi_{\tau}$ and recall that under consistent typing $\theta(d) = \max_{\pi} \{\sigma \mid d \in D^\pi_{\sigma}\}$. By (1), $\theta(d) \neq \pi$ so there must be a type $\tau'$ such that $\pi \sqsubseteq \tau'$ and $d \in D^\pi_{\tau'}$. Since $\tau$ is the only upper cover of $\pi$ we know that $\tau \sqsubseteq \tau'$. Then, $d \in D^\pi_{\tau'}$, since $f$ defines a faithfully typed domain. Because $d$ was chosen arbitrarily, it follows that $D^\pi_{\tau} \subseteq D^\pi_{\tau'}$. Moreover, since $\pi \subseteq \tau$ and $f$ defines a faithfully typed domain it follows that $D^\pi_{\tau} \subseteq D^\pi_{\tau'}$. Hence, $D^\pi_{\tau} = D^\pi_{\tau'}$ which contradicts the injectivity of $f$.

The assumption (1) leads to a contradiction in both cases and therefore we must reject it. Consequently, the theorem holds.

We now address the question of how many elements it actually takes to obtain a faithful representation.

In section 4, we briefly touched this issue when we suggested that saturation was an economical way to obtain a faithfully typed domain. Then, we showed that saturation was not an optimal approach, as we illustrated with figures 1(b) and 4. However, when only consistently typed domains are under consideration the situation is different. Given an interpretation that defines a consistently typed domain, no faithful expansion has fewer elements than its saturation has.

**Theorem 15** Let $f$ be a $D$-interpretation defining a consistently $H$-typed domain. $f' : TY PE \rightarrow 2^{D^\prime}$ a saturation of $f$ and let $f'' : TY PE \rightarrow 2^{D''}$ be any faithful expansion of $f$ defining a consistently $H$-typed domain. Then $|D'| \leq |D''|$.

**proof** By theorem 14, $f''$ defines a saturated typed domain. Then it follows from lemma 12 that $|D'| \leq |D''|$. 

So, saturation is the optimal expansion into faithfulness for consistently typed domains. This is also the key to finding minimal consistently and faithfully typed domains. One simply saturates the smallest interpretation there is, namely the one that contains no elements.

**Corollary 16** Let $f : TY PE \rightarrow 2^D$ be a saturation of the $\emptyset$-interpretation of $H$ and let $f' : D' \rightarrow$ any $D'$-interpretation defining a consistently and faithfully $H$-typed domain. Then $|D| \leq |D'|$.

**6 Conclusion**

We have discussed how to interpret and represent type-hierarchies and studied how to obtain isomorphic representation given one that carries a reduced version of the hierarchy.

The effects of confinement to consistently typed domains is substantial. The typing function is simplified, corresponding precisely to the standard modelling convention which some applications are based on [6]. Furthermore, saturatedness and faithfulness becomes the same thing and saturation is known to be the cheapest possible transformation of a typed domain into a faithfully typed domain. On the other
hand, when interpretations of inconsistent types are allowed to intersect, the transformation can be even cheaper. Anyhow the discussion reveals upper bounds on the number of elements necessary to obtain a faithfully typed domain.

References: