A Model and Representation for Type Information and Its Use in Reasoning with Defaults

Lin Padgham
LINKÖPING UNIVERSITY
Department of Computer and Information Science
S–581 83 Linköping, Sweden

Abstract

Typing schemes which allow inheritance from super to sub types are a common way of representing information about the world. There are various systems and theories which use such representations plus some inferencing rules to deduce properties of objects, about which the system has only partial information. Many such systems have problems related to multiple inheritance, and have some difficulty in drawing conclusions which we as humans see as intuitively simple.

We present a model of typing based on a lattice of feature descriptors. A type is represented by two important points in the lattice representing core and default information. The use of two points allows some information to be monotonic whilst other information is nonmonotonic.

We give some operations which can be used in default reasoning about an object, based on knowledge about the relationships between the points in the lattice which are defined as types. We then work through some specific examples, showing the conclusions which we reach with this system. We compare the expressiveness of our system to some of the well known work in the area of default reasoning using inheritance.

1. Introduction

Classification into categories and sub–categories, along with default reasoning about the properties of objects within these categories is a common human activity. It allows for continuation of the reasoning process in situations where people could otherwise be paralyzed by lack of information, or by the overwhelming number of theoretical possibilities.

There have been a number of systems built which attempt to capture the notion of inheritance from super to sub–types, and to simulate the human reasoning regarding characteristics of objects known to belong to a certain type. Some well known examples are FRL [Roberts and Goldstein,77], NETL [Fahlman,79] and TMOIS [Touretsky,86]. These systems all have difficulties around issues to do with multiple inheritance combined with exceptions. They easily run into ambiguous situations where they cannot make a decision, or make an intuitively wrong decision. This is true even for questions which people would resolve easily and unambiguously.

We present a model of typing based on the splitting of a type into *type default* and *type core*, and on the formalism of a lattice, rather than the more usual notions of a tree or an acyclic directed graph ("tangled hierarchies" [Fahlman,79]). Our model allows for representation of information which we believe people typically use in reasoning about defaults, and which is not representable in the above mentioned systems. This model plus the inference mechanism we suggest gives clear solutions to many of the problems which have previously been difficult. It also provides solutions in a consistent manner, without resorting to more complex inference mechanisms to deal with special situations.

2. Cores and Defaults

We use the notion of a type being defined not as a node in a graph of super and subtypes, but rather as a collection of characteristics which we expect to see in an object of a particular type. It is our view that this is a natural description of what typing or classification is for humans. It is the association of certain characteristics together as a group.

If we then look at the characteristics defining a type, we see that we are far more willing to override some of the characteristics than others. For this reason we identify two clusterings representing a type, – the *type core* and the *type default*.

The *type core* includes those characteristics which we regard as always present in objects of this type. [1] Only type descriptions which are *above*[2] this type core are considered to be subtypes of this type. If we state that A is a subtype of B then we know unequivocally that the core descriptor for type A contains at least all the information that is contained in the core descriptor for B.

The *type default* contains the information for typical

1 Individual objects can always fall below even the type core. This is taken up further in an example. We are currently working on a suitable way of describing such objects in order to reason about them in the way intuitively desired.

2 Note that specialization or more information is ***higher*** in the lattice, whereas it is *lower* in inheritance graphs.

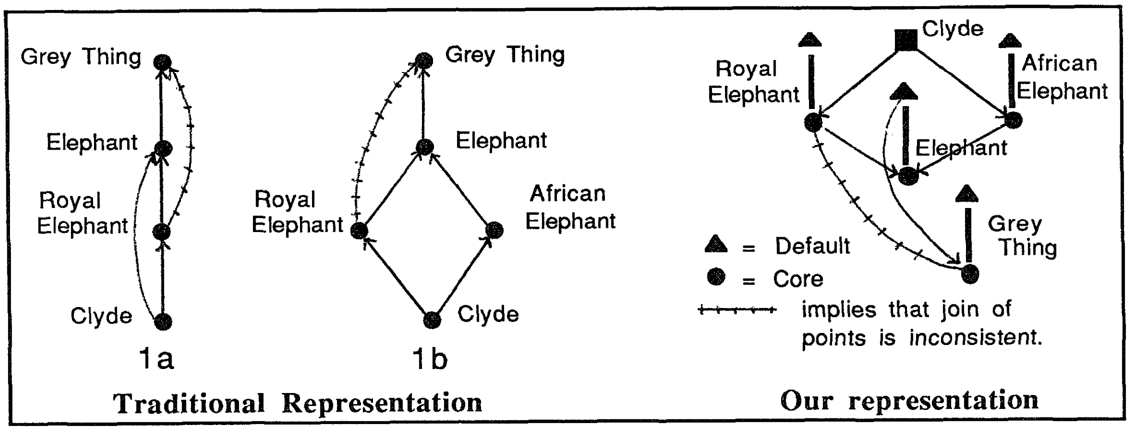

Figure 1

objects of that type. The type default must of course contain at least the same informatiom as is contained in the type core.

An individual object can also be seen as a clustering of characteristics which we can call the *descriptor* for that object.

Using this notion of two points to describe a type, let us look at a simple but classical problematic example.

2.1 Example 1

The network 1a with its redundant link (Clyde –> Elephant) is given by Touretsky [1987] as the sort of example that causes shortest path reasoning algorithms to fail, and is thus a justification for his shortest inferential distance (or onpath preemption) algorithm. Sandewall[1986] shows with network 1b that shortest inferential distance is also inadequate and proposes an algorithm which Touretsky[1987] refers to as offpath preemption. Our approach gives the desired result in both examples, and is appealing in that the addition of extra information (links to and from African Elephant) does not require any change in reasoning strategy.

We can write the following relationships from our representation:

Core(Royal Elephant) $\sqsupseteq$ *Core*(Elephant)
Core(African Elephant) $\sqsupseteq$ *Core*(Elephant)
Default(Elephant) $\sqsupseteq$ *Core*(Grey thing)
Core(Royal Elephant) $\sqcup$ *Core*(Grey thing) $\Rightarrow \mathcal{K}$
Core(Royal Elephant) $\in$ NOT(*Core*(Grey thing))
Core(Grey thing) $\in$ NOT(*Core*(Royal Elephant))[1]
Desc(Clyde) $\sqsupseteq$ *Core*(African Elephant)
Desc(Clyde) $\sqsupseteq$ *Core*(Royal Elephant)

Our diagram defines a set of *inheritance paths* having a specific object (e.g. Clyde) as their first element, and which characterize properties the object may inherit. The inheritance paths are written in the form:

$$n_0 \ op \ n_1 \ op \ n_2 \ op \ \ldots \ n_i \ op \ n_{i+1} \ldots$$

where the operation *op* in the link $n_i \ op \ n_{i+1}$, may be either of the following:

$\sqsupseteq$ $n_i \sqsupseteq n_{i+1}$ may occur in the path if the corresponding relationship occurs in the diagram.

$\mathcal{E}$ *Core*(A) $\mathcal{E}$ *Default*(A) may occur in the path, representing a default assumption that the object at the beginning of the path is not only an A, but is a typical A, (in so far as A's typicality does not conflict with already known information).

Notice that path steps using $\sqsupseteq$ proceed downwards in the graph, and steps using $\mathcal{E}$ proceed upwards, thus adding extra information to n_0 besides what is deductively available.

$\in$ The path may be terminated by $n_i \in$ NOT(A), indicating that n_i is a member of the set of nodes:

$$\mathrm{NOT}(A) \equiv \{ x \mid x \sqcup A \Rightarrow \mathcal{K} \}$$

($\mathcal{K} \equiv$ inconsistent in at least one feature value)

A set of paths is said to be inconsistent if it contains both the step $\sqsupseteq$ A, and the step $\in$ NOT(A), and consistent otherwise. The full set of paths obtained by using the $\mathcal{E}$ operator wherever possible is often inconsistent. Consequently we only allow adding in of information following an $\mathcal{E}$ operation provided it is not inconsistent with information obtained at an earlier step.

Abbreviating *Core* to *Cr*, *Desc* to *Ds*, and *Default* to *Df*, we can now infer the following about Clyde:

Ds(Clyde) $\sqsupseteq$ *Cr*(Af. Eleph.) $\sqsupseteq$ *Cr*(Eleph.)
...*Cr*(Af. Eleph.) $\sqsupseteq$ *Cr*(Eleph.) $\mathcal{E}$ *Df*(Eleph.)
[$\sqsupseteq$ *Cr*(Grey thing)] [1]
Ds(Clyde) $\sqsupseteq$ *Cr*(Roy. Eleph.) $\sqsupseteq$ *Cr*(Eleph.)
...*Cr*(Roy. Eleph.) $\in$ NOT(*Cr*(Grey thing))

1 Note that we have represented the information that royal elephants must be non-grey, whereas only typical elephants are grey.

2 We show in [ ] the information which potentially could have been added in following the extension operation, but which is not added due to its being contradictory.

We see here that we have a single extension, with no default assumptions, giving the result that Clyde, in addition to being a Royal Elephant and an African Elephant, is an Elephant and NOT a Grey thing.

If we had instead represented and reasoned about the information for the network 1a, we would simply have been missing the information:

$$Desc(\text{Clyde}) \sqsupseteq Core(\text{African Elephant})$$
$$Core(\text{African Elephant}) \sqsupseteq Core(\text{Elephant})$$

This would not have altered the inferred information. The missing initial information that Clyde is an African Elephant would of course also be missing in the conclusion.

3. A Lattice Based Model of Types

Taking the theoretical space of all possible combinations of all possible characteristics induces a lattice of descriptors. Any clustering of characteristics then belongs somewhere in this lattice. Typing can be seen as naming some of the points in this lattice as being relevant clusterings of characteristics.

This view allows for both descriptive and prescriptive typing. We can deduce the type(s) of an object by having information concerning its characteristics, and we can deduce information about an object's characteristics by having information about its types. Human reasoning also uses typing and classification in both these ways.

Reasoning about the relation between individual objects and type descriptors can also be applied to relations between subtypes and their supertypes. A particular type, A, can be observed to be a specialization or subtype of some other type, B, if the type descriptor for A contains all the information in the type descriptor for B plus some extra information.

Conclusions can be drawn about the type(s) of a given object by placing its descriptor in the lattice, and noting which types it falls above.

However we do not always have complete information regarding an object, and can therefore not place its descriptor directly in the lattice. We introduce the notion of partial information in the descriptors, plus relations between the descriptors which constrain their positioning in the lattice without fully defining it. If we say the descriptor for A is above the descriptor for B ($A \sqsupseteq B$), and we know that P is an A, then we can conclude (by transitivity of $\sqsupseteq$) that P is also a B, without knowing any of the actual characteristics of either descriptor.

By comparing two types with respect to cores and defaults, we can make the statement A's are B's with four different shades of meaning, ordered with respect to the strength of the statement.

$Core(A) \sqsupseteq Default(B)$
A's are always typical B's.

$Core(A) \sqsupseteq Core(B)$
A's are always B's – but not necessarily typical B's.

$Default(A) \sqsupseteq Default(B)$
A's are usually typical B's.

$Default(A) \sqsupseteq Core(B)$
A's are usually B's – but not necessarily typical B's.

The greater expressivity given by split into core and default is often valuable in reasoning about characteristics of objects. The information is intuitive for humans (at least within some fuzzy boundaries), and needs to be represented if a reasoning system is to draw conclusions we intuitively wish it to draw. This is demonstrated in examples.

4. Representation of negative information

Multiple inheritance reasoning systems often have negative links in the inheritance graph, which are important in the reasoning process. (E.g. NETL [Fahlman,79] and TMOIS[Touretsky,86]. On examination, these negative links are used to express two different things. These can be described as *incompatability* and *overriding.* We define incompatibility within the lattice framework, and show overriding to be unnecessary.

4.1 Incompatibility

By incompatibility between A and B we mean that there is at least one characteristic of A that is incompatible with a characteristic of B. Consequently no object can be both an A and a B. This can be described as

$$A \sqcup B \Rightarrow \mathcal{K}$$

where $A \sqcup B$ is the least upper bound (or join) of A and B, and $\mathcal{K}$ indicates that at least one feature value is inconsistent. Incompatibility can also be expressed as $A \in \text{NOT}(B)$.

As with positive relations between types we can use the core and default for the type in order to say "A's are not B's" with different shades of meaning.

$Core(\text{A}) \sqcup Core(\text{B}) \Rightarrow \mathcal{K}$
A's are never B's.

$Core(\text{A}) \sqcup Default(\text{B}) \Rightarrow \mathcal{K}$
A's are never typical B's.

$Default(\text{A}) \sqcup Core(\text{B}) \Rightarrow \mathcal{K}$
Typical A's are not B's.

$Default(\text{A}) \sqcup Default(\text{B}) \Rightarrow \mathcal{K}$
Typical A's are not typical B's.

4.2 Overriding

Overriding refers to the situation where in NETL[Fahlman,79] and TMOIS[Touretsky,86] negative links are used to override positive inherited information. The situation is not such that the negatively linked points are necessarily incompatible, but rather that it is necessary to block a possible chain of reasoning.

Within our framework this situation occurs when there exists some known type, B, which is above *Core*(A) but not above *Default*(A), and B is incompatible with *Default*(A). One then wants to ensure that the chain of reasoning from *Default*(A) is disallowed. Because we make a distinction between core and default and can identify subtypes that are known to lie between the core and default, we do not need any special override mechanism.

5. Reasoning Using The Lattice

Having defined how to express information regarding types and their relations to each other, we use the inference mechanism as described in example one, for reasoning with the information to state what types a given object may have.

The basic relation $\sqsupseteq$ between lattice points is transitive, so if we know that an object P has a descriptor such that *Desc*(P) $\sqsupseteq$ *Core*(A), and that *Core*(A) $\sqsupseteq$ *Core*(B), we can conclude that *Desc*(P) $\sqsupseteq$ *Core*(B). This method allows us to state all definite (monotonic) positive conclusions regarding *Desc*(P). We can of course always add relations of the form *Default*(x) $\sqsupseteq$ *Core*(x) (by definition).

Negative information of the form

'A $\sqcup$ B $\Rightarrow \mathcal{K}$'

gives us relations of the form

'X $\sqsupseteq$ A $->$ X $\in$ NOT(B)'

as defined previously.

The extension operation $\mathcal{E}$ defined in example one allows us to make the nonmonotonic jump from the core of a given type to its default,

Core(A) $\mathcal{E}$ *Default*(A)

The extension operation can be repeated as long as it is possible to obtain more information by doing so. Each such operation implies an assumption. If there is more than one such assumption in the reasoning process, the ordering of the extension operation gives the different valid extensions. Preference for making this assumption at the most specific point possible (in terms of the $\sqsupseteq$ lattice relation) gives the intuitively desirable preferences between extensions.

This preference for the most specific *Core* to *Default* specializations prefers the same extensions as those obtained by the shortest inferential distance algorithm of Touretsky, and the offpath preemption algorithm of Sandewall, but in a more clearly motivated and consistent manner.

6. Some Classical Examples

6.1 Example 2

Figure 2 shows the same network as figure 1 but with different labeling on the nodes. Touretsky argues that with the changed labeling it is less intuitively clear what conclusions we wish to draw. The lattice diagram shows the suggested representation within our model, which differs from the lattice representation given for example 1. This captures the differing strength in the information that Marines (and Chaplains) are Men, compared with African Elephants (and Royal Elephants) are Elephants.

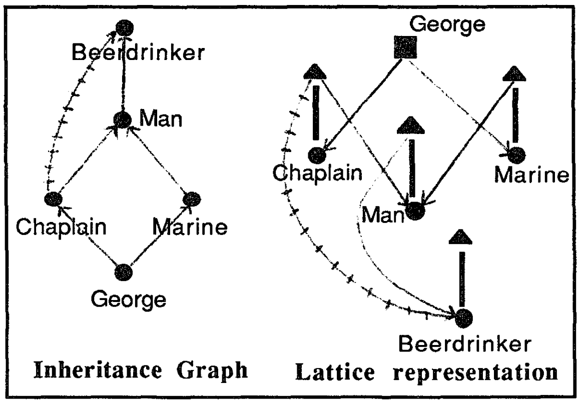

Figure 2

Extracting the formulas similarly to example 1 we reason as follows:

Ds(George) $\sqsupseteq$ *Cr*(Mar.) $\mathcal{E}$ *Df*(Mar.) $\sqsupseteq$ *Cr*(Man)
$\mathcal{E}$ *Df*(Man) $\sqsupseteq$ *Cr*(Brdr.)
Ds(George) $\sqsupseteq$ *Cr*(Chap.) $\mathcal{E}$ *Df*(Chap.) $\sqsupseteq$ *Cr*(Man)
...*Df*(Chap.) $\in$ NOT(*Cr*(Brdr.))

The extension to *Default*(Man), giving *Core*(Beerdrinker) conflicts with the extension to *Default*(Chaplain) which gives $\in$ NOT(Beerdrinker). Ordering of the conflicting extension operations gives two extensions – one in which George is a Beerdrinker and one in which he is NOT a Beerdrinker. The previously discussed preference for extension operations from the highest lattice points, when there is a conflict, gives preference to the extension in which George is NOT a Beerdrinker. (*Core*(Chaplain) $\sqsupseteq$ *Core*(Man))

The representation that allows us to differentiate between 'Typical Chaplains are Men' vs 'All Royal Elephants are Elephants' enables us to then use this information to reason in a more natural (and correct?) way, than the IS–A network representation does.

6.2 Example 3

Figure 3 shows the canonical ambiguous net ('the Nixon diamond') concatenated with a net giving further ambiguities (from TOU87). We work through this example with two different choices of representation, in order to bring out some properties of our model.

6.2.1 Quakers are always Pacifists (Figure 3b)

In the following representation we represent all Quakers as being Pacifists, but only default Republicans as being incompatible with Pacifists. All other choices between default and core of the types have no influence on the resulting conclusions. (They simply influence the certainty of the extensions. We have made choices here to reduce the number of extension operations needed)

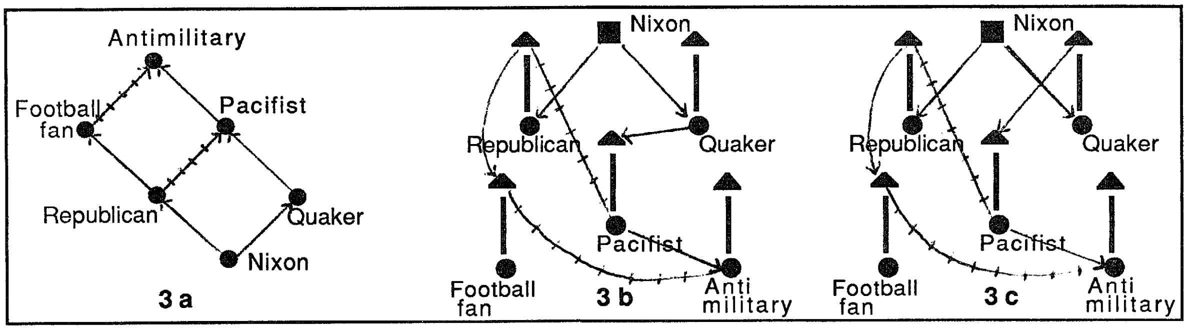

Figure 3

By using the formulas as previously we reason as follows:

Ds(Nixon) ⊒ Cr(Qkr) ⊒ Df(Pac.) ⊒ Cr(Pac.)
⊒ Cr(Antimil.)
DS(Nixon) ⊒ Cr(Repub.) & Df(Repub.) ⊒ Df(Foot. fan)
[∈ NOT(Cr(Antimil.))]
...Df(Repub.) [∈ NOT(Cr(Pac.))]

We obtain a single extension in which Nixon is a Quaker, Republican, Pacifist, Antimilitary and Football fan. This is disturbing to our intuitions as Nixon is a particular individual who we know to have been non–pacifist and non–antimilitaristic. This serves to illustrate the point previously noted, that individuals must be allowed to fall even below the type core, where explicitly stated to do so. However this does not justify the building of a hypothetical class of militaristic Quakers. We are currently working on an appropriate representation for individuals who fall below the core, that will enable us to continue the desired reasoning about such an individual.

6.2.2 Typical Quakers are Pacifists (Figure 3c)

In figure 3c we replace the original symmetry of the Nixon diamond by showing only typical Quakers to be Pacifists. By doing the reasoning as previously (not shown due to space limitations) we obtain two extensions, one the same as with the previous representation, and one in which Nixon is Quaker, Republican, Football fan, NOT Pacifist and NOT Antimilitary.

If we had shown the link from Pacifist to Antimilitary as weaker than we did (i.e. as *Default*(Pacifist) ⊒ *Core*(Antimilitary)) we would also have generated an extension in which Nixon was Quaker, Republican, Football fan, Pacifist and NOT Antimilitary.

The results here are consistent with Touretsky's description of a credulous reasoner, in that all possible extensions are found. In this case there is no simple preference between extensions based on assumption of typicality at the most specific point possible. However all extensions are obtained if one wishes to then choose among extensions on the basis of various heuristics.

6.3 Example 4

Figure 4 shows an example from Fahlman et al (FAH81). The figure shows both the representation as it is in NETL, and also the more informative representation of Etherington, who uses a wider variety of link types. The meanings of Etherington's link types are given in figure 5.

The lattice representation comes directly from Etherington's network and does not contain any extra information that is intuitively assumed in choosing between core and default. The information is already implicit in the graph.

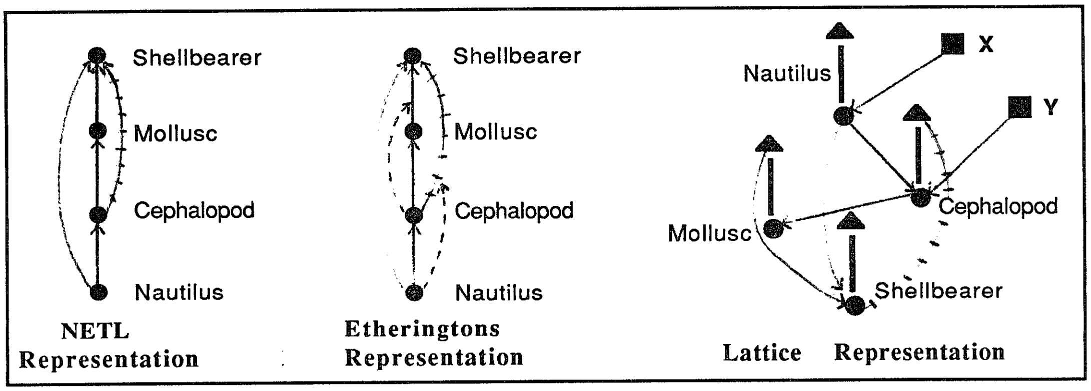

Figure 4

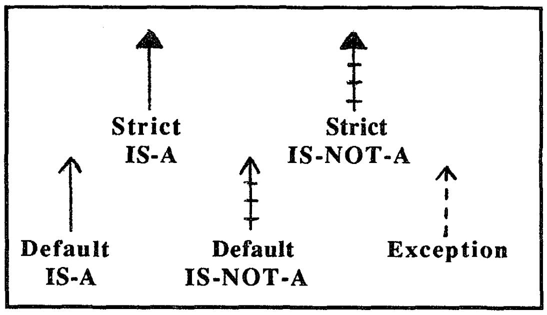

Figure 5: Etherington's link types

Reasoning about the individual Nautilus, X, we get:
X $\sqsupseteq$ C(Nautilus) $\sqsupseteq$ C(Cephalopod) $\sqsupseteq$ C(Mollusc)
...C(Ceph.) $\mathcal{E}$ D(Ceph.) [$\in$ NOT(C(Shell.))]
...C(Mollusc) $\mathcal{E}$ D(Mollusc) $\sqsupseteq$ C(Shellbearer)
...C(Nautilus) $\sqsupseteq$ C(Shellbearer)
We reach the same conclusion as both NETL and Etherington's system, that X is without competing extensions or uncertain assumptions, a Cephalopod, a Mollusc and a Shellbearer.

Reasoning about the individual Cephalopod, Y, we get:
Y $\sqsupseteq$ C(Cephalopod) $\sqsupseteq$ C(Mollusc) $\mathcal{E}$ D(Mollusc) $\sqsupseteq$ C(Shellbearer)
...C(Ceph.) $\mathcal{E}$ D(Ceph.) $\in$ NOT(C(Shellbearer))
This corresponds to two extensions, one in which Y is a shellbearer, and one in which it is not. The extension in which it is not a shellbearer is preferred on the basis of its resulting from an extension operation at a more specific point.

This conclusion is neither the same as NETL nor Etherington's conclusions (which also differ from each other here). Etherington's method provides only one extension, the same as our preferred extension. NETL gives both extensions but does not discriminate at all between them. We feel that our solution, providing both extensions, but with a clear and intuitively reasonable preference, is the most desirable.

7. Discussion

The representation and reasoning methods proposed here appear to offer clear advantages when compared to systems such as TMOIS and NETL which use network representations with two sorts of links (positive and negative) plus possibly exceptions.

The largest gain results from the division of type descriptors into two parts, giving a similar effect to what Touretskty classifies as heterogeneous, bipolar systems. Heterogeneous refers to the ability to have some information which is certain (monotonic), while other is uncertain (nonmonotonic).

The inference mechanism for reasoning about objects within a lattice based type schema appears to be cleaner and more consistent than the inference mechanisms developed for reasoning about network based representations. Many of the examples which have proved problematic for network based systems, and which have required new inference mechanisms, are simple and clear within the lattice model. The lattice formalism also gives greater clarity to the semantics of negative links than has been evident in network representations.

A comparison of the lattice based model with Etherington's system is less definitive with respect to which is better. By increasing the number of link types he also specifies a heterogeneous system. This has similar advantages to our system in that he represents more information, which is necessary for achieving the desired reasoning.

An essential difference between Etherington's model and ours is that he represents explicit information about exceptions, whereas we simply represent lack of typicality. Both approaches have their advantages. The advantage with our model is that if we know there are exceptions we generate both extensions, with a preference for the most likely. Etherington's method generates only the typical extension, unless one knows that the particular exception causing case is present.

We plan to implement the described system and experiment with larger, more complex examples, to determine whether the described model does provide the reasoning and representation facilities to achieve intuitively desirable results.

Acknowledgements

We thank Erik Sandewall and Ralph Rönnquist for helpful discussions of the ideas presented here, and Jalal Maleki for careful proof reading and useful comments.

References

[Etherington, 87] Etherington, D. W. Formalizing Nonmonotonic Reasoning Systems. *Artificial Intelligence,* vol. 31, 1987, pp. 41–85.

[Fahlman, 79] Fahlman, S.E. *NETL: A System for Representing and Using Real-World Knowledge.* The MIT Press, Cambridge, MA, 1979.

[Fahlman *et al.*, 1981] Fahlman, S.E., Touretsky, D.S. and van Roggen, W. Cancellation in a Parallel Semantic Network. *Proceedings of IJCAI-81,* 1981, pp. 257–263.

[Sandewall, 1986] Sandewell, E. Non–monotonic Inference Rules for Multiple Inheritance with Exceptions. *Proceedings of the IEEE,* vol 74, 1986, pp. 1345–1353.

[Touretsky, 1986] Touretzky, D.S. *The Mathematics of Inheritance Systems.* Morgan Kaufmann Publishers, Los Altos, CA, 1986.

[Touretsky *et al.,* 1987] Touretzky, D.S., Horty, J.F., Thomason, R.H. A Clash of Intuitions: The Current State of Nonmonotonic Multiple Inheritance Systems. *Proceedings of IJCAI 87,* Milan, August 23–28, 1987, vol 1, pp. 476–482.