The Representation of Defaults in Cyc

Ramanathan V. Guha
MCC
3500 West Balcones Center Drive
Austin, Texas, 78759-6509
guha@mcc.com

Abstract

This paper provides an account of the representation of defaults in Cyc and their semantics in terms of first order logic with reification. Default reasoning is a complex thing, and we have found it beneficial to separate various complex issues whose "current best solution" is likely to change now and then – such as deciding between extensions, preferring one default to another, etc. – and deal with them explicitly in the knowledge base, thus allowing us to adopt a simple (and hopefully fixed) logical mechanism to handle the basic non-monotonicity itself. We also briefly describe how this default reasoning scheme is implemented in Cyc.

Background

The Cyc project (3) is an effort aimed at building a large common sense knowledge base. CycL is the language in which the Cyc KB is encoded. Since much of common sense knowledge is default in nature, it is important for CycL to provide facilities for expressing defaults. This paper describes the scheme used to do default reasoning in Cyc.

We are trying to build a Knowledge Base (KB) that can be used by a number of programs and it is important for us to be able to provide an account of the contents of the KB in a language with clean and simple semantics. We are also interested in providing certain inferential services with the KB and would like these to be efficient. Various special purpose constructs for dealing with common cases, special inference procedures (and associated special representations, etc.) are used to improve the efficiency of the inference mechanism. However, special constructs and domain specific inference procedures make the task of giving an account of the contents of the KB in a simple language very difficult.

Since these two requirements, simplicity and efficiency, are hard to obtain in a single language, we have divided CycL into two levels, one for obtaining each of these goals. The Epistemological Level (EL) is meant for communicating the contents of the KB to humans and other programs and attempts to use a simple language, while the Heuristic Level (HL) has a variety of special representations and inference procedures to help speed up inference. This distinction follows (7). A translator, the TA (1), is capable of translating expressions from the EL to the HL and vice versa.

The EL uses first order predicate calculus with reification, and defaults are stated using these. This paper provides an account of the defaults in CycL largely at the EL. Guha and Lenat (9), (3) and a forthcoming paper go into the full details of the HL, so this paper limits its description to some of the more important issues related to implementing the scheme presented here.

Research on default reasoning is not our primary goal and the only reason we are building CycL is so we can encode Cyc in it. The representation of defaults is an active area of research and there is no commonly accepted standard. This makes it very hard for a person to build a representation language with the intention of using it to build a KB over a number of years. Though we expect the KB to keep changing and growing, it would be most inconvenient for us to have to change the logic underlying our language as we encounter difficulties in default reasoning.

Default reasoning is complex and it would be beneficial to separate logic level issues such as nonmonotonicity from other issues such as deciding between extensions, preferring one default to another, etc. We therefore use only the simplest logical mechanisms to obtain the basic nonmonotonicity, and we deal with the other issues in the KB rather than in the logic.

The next section describes the intuition behind our scheme and the following section provides a more formal discussion. We then discuss certain crucial issues in default reasoning such as preferring one of a set of possible (mutually contradictory) default conclusions over another. Then comes the discussion of how this is currently implemented in CycL, and the last section is a sketch of some promising directions for our future research.

The Intuition

The basic idea is as follows. When determing whether some proposition P is true, one constructs *arguments* for and against that proposition and decides one way or the other after comparing these arguments. The addition of information can change the availability of arguments for or against P, and this is what is responsible for the nonmonotonicity. The comparison of arguments is a complex issue and an arbitrary amount of knowledge could potentially be used in this comparison process. This suggests that this comparison be done using a knowledge based approach through axioms in the KB explicitly for this purpose. Such axioms (which enable us to determine which argument to prefer) are called *preference axioms* and are as much a part of the KB as are axioms about phenomena such as eating and sleeping. The aim is to have all the mechanisms available to do common sense reasoning and expert reasoning should be available to deal with default reasoning as well.

In order to be able to state the preference axioms as regular axioms in the KB, we need to be able to treat arguments as first class objects. Since arguments are sequences (or possibly richer structures) of sentences, we use reification (5), allowing arbitrary sentences to be reified. These reifications can be treated as first class *rich* (5) objects.

Axioms that refer to arguments for sentences will need to use reified forms of these sentences. Since we need to relate arguments for a sentence to the truth-value of the sentence, a single axiom might refer to both a sentence and its reification; i.e., we are going to need mixed level statements.

Structure Of Default Statements

Intuitively, the main difference between "normal" axioms (i.e. ones that are not defaults) and defaults is that the defaults are weaker and incorporate some scheme by means of which they may be "beaten". The concept of abnormality predicates (6) is an ideal candidate for expressing this weakening and is the one used to encode the defaults in Cyc. So if we consider the canonical example of birds flying as a default and penguins being an exception, the syntactic structure of the defaults as they would be stated in Cyc is as suggested by McCarthy (6) and is as follows:[1]

$\neg\mathrm{ab}(x, \mathrm{Aspect1}) \land \mathrm{bird}(x) \supset \mathrm{flies}(x)$

$\mathrm{penguin}(x) \supset \mathrm{ab}(x, \mathrm{Aspect1})$

However, unlike circumscription, we don't minimize the extent of ab to conclude ¬ab(Tweety,Aspect1). Instead we use the concept of arguments to conclude, from the above default, that Tweety flies.

The 'ab-literals' (the literals such as ¬ab(Tweety,Aspect1)) have a special status in that we assume they can be distinguished from other formulae. This special status is quite easily represented by defining a unary predicate *abLiteral* such that (abLiteral x) is true if and only if x is the reification of a negated literal involving an abnormality predicate.

The next section describes what exactly an argument is and how these are used to derive conclusions.

Arguments and their Use

If there is a proof for a sentence, then that sentence is logically entailed by our KB (provided we are using sound inference rules) and therefore proofs are the primary mechanism for obtaining conclusions from a KB. The role of arguments in our default reasoning scheme is analogous to the role of proofs in monotonic theories. Since an argument is very similar to a proof both in structure and use, we give a description of an argument by comparing it to a proof.

A proof is a finite sequence of sentences such that the last sentence is the sentence being proved and each sentence is either a given axiom, an instantiation of a given axiom schema, or follows by the application of an inference rule to some set of sentences earlier in the proof. Therefore each sentence in the proof is also a theorem that follows from the KB.

An argument is similar, but we weaken it a bit. For a given sentence P, an argument for P is a finite sequence of sentences (ending with P) such that each sentence is either a given axiom, an instantiation of an axiom schema, follows by the application of an inference rule to previous sentences in the argument, *or is a negative abnormality literal* (i.e., the reification of the sentence satisfies the predicate *abLiteral*.)(8) gives the complete set of axioms that defines arguments (and the predicate argumentFor mentioned below.)

Intuitively, an argument is a weakening of a proof in the sense that we are 'asking' to be allowed to make a certain set of assumptions and these assumptions are nothing but the abnormality literals (ab-literals). The concept of an ab-literal is closely related to that of an assumption and so the predicate ab-literal can easily be generalized to a predicate such as *assumable*, using which any sentence can be specified to be assumable.

Another difference between arguments and proofs is that while proofs are (usually) only objects in the meta-theory of the logic, arguments are objects in the domain of discourse (i.e. are "things" in the KB.)

How does the existence of an argument for P relate to whether P is a theorem? Intuitively, if we have an 'acceptable' argument for P, and if there isn't any 'better' argument for ¬P, then we would like to accept P as a theorem. This notion is captured by the following axiom, the *Argumentation Axiom* (which is also an axiom in the KB along with axioms that characterize what an argument is).

[1] The symbols x, a_1, a_2 etc. are all universally quantified unless other mentioned

$(\forall\ (a_1,'p)\ (\text{argumentFor}(a_1,'p)$[2] $\wedge \neg\text{invalidArg}(a_1) \wedge$
$(\forall\ (a_2)\ (\text{argumentFor}(a_2,'\neg p)$
$\supset (\text{invalidArg}(a_2) \vee \text{preferred}(a_1,a_2)))))$
$\supset \text{True}('p))$

A few comments about this axiom are in order. (Assume we want to know if P is a theorem.)

a. If we had to *prove* that there does not exist a preferred argument for ¬P, we might never conclude anything. We therefore make a closed world assumption for the predicates *argumentFor* and *invalidArg*. This can be done by using any of the existing formalisms for obtaining nonmonotonicity. This closed world assumption for these two predicates is the only mechanism that is used to obtain nonmonotonicity. An alternative to making the closed world assumption would be to add an axiom schema that minimized the extents of argumentFor and invalidArgument (in that order.)

b. The presence of a truth predicate in the language leads to the possibility of paradoxs. To avoid this, our truth predicate is a weak one. I.e. given a reified sentence P, (True('P) ∨ True('¬P)) is not a theorem. The truth predicate True differs from truth-values of formulae on those formulae capable of leading to paradoxes.

c. Given an argument A for P, another argument A1 can be obtained by simply adding more true sentences to A. Though it is tempting to restrict our attention to minimal arguments, we resist doing this for the following reason. There are times when we would like to allow extra information in an argument that makes it easier to compare it to other arguments. This turns out to be useful for solving problems such as the Yale Shooting Problem (10). Later examples of preference criteria can be seen to make use of this. However, as a default, if an argument A1 for P subsumes another argument A for P, we prefer A to A1. This is a case of using default reasoning to determine the preference ordering between arguments itself.

d. While it is acceptable to make assumptions about the truth-value of ab-literals, we don't want to assume that a particular ab-literal is true if we can prove that it is not. I.e., in such cases, we would like to consider the argument as being invalid. This is captured by the following axiom schema:

sentenceInArg(a,'q) ∧ ¬q ⊃ invalidArg(a)

e. As mentioned earlier, we determine whether or not one argument is preferred to another using axioms in the KB. Some examples of these axioms (the preference axioms) are given in the next section.

f. One can imagine a rich theory of argument types and dialectic patterns being used to do default reasoning. Sample categories of argument types include *reductio ad absurdum* arguments, inductive arguments, arguments for the truth of a sentence that provide a possible sequence of events that could have caused it, etc. Associated with each of these could be ways of countering the argument, reinforcing it, etc. We later present one of these *cliched* argument patterns called a narration. The scheme presented here seems to provide an adequate framework for capturing all this.

g. One of the aspects of a proof (which carries over to arguments) is that it is finite in length. Since we can't axiomatize the notion of finite, we need to place some restriction on the length of our arguments. Also, given we can expend only a finite amount of resources on searching for arguments and in ensuring that they are not invalid, we need to incorporate some means for specifying the resources that may be spent in doing this. We do this by saying that any argument that requires more than a certain amount of resources to compute is invalid. Since attempts to prove arguments invalid are themselves likely to involve arguments, this also has the effect of limiting the resources spent on trying to prove arguments invalid. This notion is captured by the following axiom.

$(>\ (\text{resourcesRequired}\ a_1)\ (\text{resources-available})) \supset \text{invalidArg}(a_1)$

In this axiom we use the indexical function *resources-available* to compute the resources available for a given problem. There are two ways in which this axiom can be used. If we have used up the resources available for generating arguments, then this axiom tells us that we are justified in giving up, since even if we *could* carry on (overusing the available resources), any argument we generated would be invalid and hence useless. Alternately, if we had some means of estimating the resources that would be required to generate an argument, and this turns out to be greater than the resources available, then this axiom gives us a justification for ignoring such arguments. So if there doesn't seem to be any obvious argument, but some reasoning suggests the possibility of some highly contrived arguments, this axiom provides the justification for ignoring such arguments.

[2]We use 'P to refer to the reification of the formula P

Preference Criteria

Clearly, one of the central issues in this whole scheme is the task of coming up with and axiomatizing criteria for comparing arguments and deciding which to prefer. In this section we describe a few sample criteria and, as an example, show how one of these can be axiomatized.

Some sample preference criteria include the following:

- **Using Inferential Distance:** Inheriting properties from classes is a common use of default reasoning.

Sometimes, the properties inherited from two different classes could be contradictory. In such cases, if one class is a subset of the other, we can prefer the value inherited from the smaller set (2). (For example, consider the average IQ of Mammals in general, and Humans in particular.) This criterion is currently used in Cyc.

- **Causal Arguments:** A strong case has been made elsewhere (4) that arguments with a 'causal flavor' are preferred to arguments that use *reductio ad absurdum* and other non-causal and non-constructive methods. This intuition can be made more precise as follows. Certain sentences which include the material implication operator are labeled as being 'causal'.[3] Once we have this labeling, we can prefer an argument that uses one of these causal sentences to one that does not. This criterion is used with certain specialized kinds of arguments and an example of this is presented later. One has to be careful about introducing redundant causal sentences to support arguments and this is taken care off by the default that we prefer subsumed arguments. There are many useful specializations of this criterion and some of these are currently used by Cyc.

- **Bias From Desires:** Consider reasoning about the beliefs of an agent. It is well known that given evidence for and against some fact, there is a bias towards believing in the position that is favourable to ones own goals and desires. Though this might not be the most rational thing to do in general, it is something to be taken into account when reasoning about the beliefs of other agents. This can be formalized quite easily in this scheme. Given arguments for beliefs(A,P) and for beliefs(A,¬P) (which implies ¬beliefs(A,P)), if P also happens to be one of the desires of A (i.e. desires(A,P) is true), then the argument for beliefs(A,P) is preferred over that for beliefs(A,¬P) (and ¬beliefs(A,P).) More elaborate versions of this can be obtained by incorporating notions of the objectivity of the agent involved in this preference criterion. We are planning on including this preference criterion into Cyc.

- **Avoiding Ignorance:** Tversky (12) describes experiments where human subjects were found to be more willing to retract some belief if they were given an alternative, as opposed to simply retracting the belief without substituting another belief in its place. For example, in reading a murder mystery, the reader often hypothesizes that some particular suspect (Fred) was the murderer, even if there is some contradictory evidence, rather than remaining 'uncommitted' about who the murderer was. However, faced with the same counter evidence but supplied with suggestions of a particular alternative suspect (Jane), the same reader might decide to switch his 'running hypothesis' of the guilty party from Fred to Jane. We can formalize this somewhat irrational tendency of humans to cling onto tenous beliefs in the absence of alternate beliefs in certain cases as follows. Given a sentence of the form s(u,v1), we prefer an argument for this over an argument for its negation if there exists no other v2 such that s(u,v2). So if we have an argument for s(u,v1) and one for ¬s(u,v1) and these two arguments are incomparable (without using this heuristic) then, if there exists no v2 such as s(u,v2), we prefer the argument for s(u,v1) over that for ¬s(u,v1). If there does exist such a v2, we might prefer the argument for ¬s(u,v1) or we might leave the situation unresolved (i.e. neither s(u,v1) nor ¬s(u,v1) is a theorem.) This preference criterion is included in the current version of Cyc.

We now describe how the first criterion can be formalized. This exercise is meant largely to provide a flavor for these axioms. After that we provide an informal description of the second criterion and show this can be used to solve the Yale Shooting Problem.

Example 1: Animals are, as a default, quite stupid, but humans (who are animals) are quite smart. Also, nothing can be both stupid and smart. Given a human (Fred), we have one argument that he is smart and another argument that he is stupid. We would like to conclude that he is smart. The axiomatization of this example follows.

(R1)isa(x,Human)∧¬ab(x,Human)⊃iq(x,High)
(R2)isa(x,Animal)∧¬ab(x,Animal)⊃iq(x,Low)
(R3)¬(iq(x,High)∧iq(x, Low))
(R4) subClass(Animal,Human)
(H) isa(Fred,Human)
(A) isa(Fred,Animal) ;;; from H and R4

Given that Fred is an Animal, either iq(Fred, High) (denoted as P) holds or iq(Fred, Low) (denoted as Q) holds, but not both. We want to write a preference axiom that will enable us to conclude P (because Human is a subset of, hence more specific than, Animal). This will happen if we can somehow defeat the arguments for ¬P and Q. The preference axiom that gives us this result is as follows:

$$
\begin{array}{l}
(\forall(a_1,'p,a_2)(\text{argumentFor}(a_1,'p)\wedge\text{argumentFor}(a_2,'\neg p)\wedge\\
\quad(\forall\ (ablit_2)\ \text{abLitOfArg}(a_2,ablit_2)\ \wedge\\
\qquad\qquad\text{classOfAbLit}(ablit_2,\text{c2})\ \wedge\\
\quad\ (\exists\ (ablit_1)\ \text{abLitOfArg}(a_1,ablit_1)\ \wedge\\
\qquad\qquad\text{classOfAbLit}(ablit_1,\text{c1})\ \wedge\\
\qquad\qquad\text{subclass(c2,c1))))}\\
\ \supset \text{preferred}(a_1,\ a_2))
\end{array}
$$

When determining whether P is true, we have an argument for it based on Fred being a Human. The

[3]This can be easily done using reification. Of course if the sentence (p⊃q) is labeled causal, this does not in itself imply that (¬q⊃¬p) is causal. The sentence (¬q⊃¬p) will not be labeled causal just because (p⊃q) has been, since labeling is on the reified forms of the sentences. Remember we reify sentences, not propositions; the reification of (p⊃q) is not equal to the reification of (¬q⊃¬p).

counter argument to this is that since he an Animal, Q must be true and since P and Q can't both be true, P must be false. This counter argument is however defeated since the counter argument assumes that ab(Fred, Animal) is false while the argument for P assumes that ab(Fred, Human) is false. But since Human is a subclass of Animal, the preference axiom applies and P∧¬Q follows. The actual arguments for P and ¬P are :

For P : [H, ¬ab(Fred, Human), R1, P]

For ¬P : [H, R4, A, ¬ab(Fred, Animal), R2, Q, R3, ¬P]

It is easy to see that using the preference axiom, we can conclude that the argument for P is preferred over the argument for ¬P, from which (and R3) ¬Q follows.

Actually this is just one of the axioms for capturing the preference criterion based on inferential distance. Also, these heuristics can be made more intuitive by using a class of arguments corresponding to using inheritance to conclude default properties etc.

Example 2: Let us take a look at an example of the use of a specialized kind of argument. A standard kind of argument for the truth of a temporal fact is to provide a plausible sequence of events (along with the changes they caused) to explain how some fact came to be true. We call such an argument a *narration* argument. We make this notion a little more precise and show how this can be used to solve the Yale Shooting Problem.

A narration argument has the following structure. The argument is divided into a number of subsequences, with each corresponding to one step/increment in time. Each subsequence consists of three parts :

- The sentences describing the world before that step,
- The sentences describing actions that took place during the event, the sentences describing the effects of these actions (most of which are likely to be causal sentences) and possibly certain assumptions such as those made by the frame axiom,
- The state of the world after that step.

Sentences describing the intra-state contraints, the frame-axiom, etc. could be included in a header to the argument. The first block of one sub-sequence can be the last block of a preceding sub-sequence. The same set of descriptors is supposed to be used in describing all the situations. This set of descriptors is not a complete description of the world, but only includes those relevant to P. The exact form of these descriptors depends on the formalism being used for time. So if one were using situation calculus, these descriptors would be of the form holds(f,s) and the same set of fluents should be used in the first and third block of every subsequence. The last sentence of the third block of the last subsequence is the sentence we are trying to prove.[4]

[4]It should be noted that all we have done here is to add some sentences to the argument and give it more structure. All that was said about arguments holds good for this kind of argument.

Given two such narrations, one can exploit the notion of causation to compare them. As we mentioned earlier, we can label certain sentences as being causal in nature and we label changes (each change in the truthvalue of a state descriptor is a change) deduced by using causal sentences as being causal changes. Given two narratives, we prefer the one with fewer non-causal changes. We could go one step further and prefer narrative proofs with no non-causal changes to a non-narrative proof. Let us now see how this approach can be used to solve the Yale Shooting Problem.

Stated informally, the Yale Shooting Problem is as follows. At time s_0 we have a loaded gun and Fred is alive. We wait for a step and at time s_1 we shoot Fred. We want to know whether Fred is dead in s_2. We are given a background theory which says that if a person is shot with a loaded gun, the person dies. The frame problem comes into play since we need to deduce that if the gun became loaded in s_0, it "must" still be loaded at time s_1. One way to solve this quite generally is to use the frame axiom which is as follows. Unless a fact is abnormal with respect to some action in a situation (i.e. some action capable of changing this takes place in the situation), it remains true in the next situation. As a result, the gun can remain loaded at time s_1, therefore Fred dies. However this is not the only possibility. The other is to start with the fact that Fred is alive at time s_1, decide that this remains true, which means that the gun was somehow unloaded at time s_0. Note that the second possibility involves the same *number* of abnormalities as the first one (in the first one the fact that Fred lives changes, and in the second one the gun being loaded changes). How does one eliminate the second possibility?[5]

Intuitively, the second possibility seems flawed since the gun somehow miraculously became unloaded in s_0. Let's see how this intuition can be captured in our framework. We label the axiom that says "a person dies when shot with a loaded gun" as being causal. Then, using the heuristic about preferring causal narrative arguments, we get the right answer as follows. While the first possibility has a straightforward narrative argument (using only causal rules to explain changes), there does not seem to be any simple narrative for the second possibility. Though one can add enough statements to the header to obtain a narrative argument for the second possibility, the change in the gun being loaded from s_0 to s_1 (when a wait is performed) does not have any causal sentence associated with it. Because of this, we prefer the first argument to the second from which it follows that Fred dies.

However this solution suffers from the following defect[6]. Consider the following extension to the prob-

[5]We would like to keep as much as the original axiomatization as possible. Completely redoing it would be cheating!

[6]This was pointed out by John McCarthy and turned out to be a good exercise in correcting undesired behaviour by

lem. We are told that there are two waits performed (i.e. we wait for two steps in time) and then shoot. We are also told that Fred is alive after the shooting. The only way in which this could have happened is if the gun got unloaded. So we would like to conclude that the gun got unloaded either during the first wait or during the second wait. However, the above scheme will insist that the unload took place during the second wait and not during the first one (i.e. that loaded was true in s_1 and unloaded was true is s_2). Insisting that the gun was unloaded during the second wait and not during the first is unintuitive. The problem is that our narrations only swept forward in time and any scheme that is biased towards one direction, forward or backward, is likely to exhibit such a behaviour. This can be corrected as follows.

We introduce a notion of a backward narration where each subsequence of the narration temporally follows (as opposed to preceeding it as in the earlier case) the next subsequence. Of course, since this kind of narrative is not likely to have sentences describing the effects of actions, it is unlikely that there will be causal sentences in it. So, when being compared to forward narrations, causality should not be a criterion used in the comparison. The other constraints we specified for forward narrations hold. A narration can be a forward or backward narration. We also add a 'reverse inertia' axiom that says that if a fact p is true in some situation s_i, unless it is abnormal in some action performed in s_{i-1}, it must have been true in s_{i-1}. With these two constructs, we get one argument (from a forward narration) for the gun being loaded in s_1 and one for it being unloaded (from a backward narration) in s_1 and since we dont prefer one over the other, we are prevented from drawing the unintuitive conclusion that the gun was loaded in s_1.

Though this may seem to be a complicated scheme, we have to remember that all the intuition was captured directly in the axioms in our KB without changing *any* of our original representation (an axiomatization of this works on the original formulation of the problem given by Hanks and McDermott (10)), and without changing the logic. And once we took the pains to identify the class of narrative arguments, this can be used for any number of other examples.

Implementation

We strongly believe that the best way to actually test the feasibility of any proposed scheme for reasoning is to implement it and try to obtain the desired results from it. In fact the scheme described above evolved through a cycle of implementing something, obtaining an abstraction of it, improving it at the abstract level, implementing the new abstraction, and so on.[7] The task of using the defaults to obtain conclusions is the task of the Heuristic Level (HL) which is not really the topic of this paper. However, this section provides a short description of the approach used. The HL does not provide a complete inference mechanism (it can't, since with reification and mixed level statements the language becomes undecidable), but it does cover part of what can be done using the above formalism in an efficient fashion. We are currently in the process of redoing part of the HL to make it cover more of the above formalism.

The concept of generating arguments and comparing them lends itself very conveniently to an implementation. The HL is largely organized around default reasoning and is divided into the following modules.

a. Argument generator: Given a sentence P, this module generates arguments for it.

b. Argument Validator and Comparator: Given two arguments, this module checks for the validity of the arguments and compares them.

c. Contradiction Detector: This module tries to detect when there is a contradiction, detects the wrong assumptions underlying this, and tries to fix them.

d. Conclusion Retractor: When a fact ceases to be true, this module retracts conclusions that were made based on it.

Given a query P, module [a] is called twice, to generate arguments for P and for ¬P. These arguments are then handed to module [b]. It checks their validity, compares them, and decides on one of them (or none of them if there is an unresolved tie) and adds the sentence with the winning argument to the to the KB. Though this module might itself call on the problem solver (since the preference axioms are just like other axioms) we have proceduralized many of the preference axioms in the EL at the HL for the sake of efficiency.

At the HL, the representation of defaults is quite different than that at the EL. The abnormality literals are stripped from the axioms and classes of axioms with the same abnormality literal are formed and labeled with the literal. The argument validator keeps track of the truth-values for these labels (actually instantiations of these) and uses them in checking for the validity of arguments. Of course, the argument comparator makes heavy use of these labels (and the other sentences used in the arguments for P and ¬P) to compare arguments.

The argument generator is not concerned with the labels and deals only with the versions of axioms that are stripped of them. Certain precautions have to be taken against an apparently contradictory KB since the ab-literals have been stripped away. Overall, though, this significantly reduces the complexity of generating arguments.

Further details on the HL can be found in (3), (9).

just adding axioms about arguments

[7] We are currently on our fifth cycle, having tried a number of things including numeric certainty factors and other probablity-like schemes over the previous five years.

Discussion and Future Work

In this paper we presented some of the salient aspects of the architecture of CycL with an emphasis on default reasoning. We gave a description of a scheme for doing default reasoning based on the notion of arguments. The basic idea was to tackle some of the hardest issues in default reasoning in the KB as opposed to dealing with them in the logic.

A clear separation was made between the mechanism used to incorporate the nonmonotonicity in the logic and the other issues in default reasoning. The non-monotonicity is incorporated using just the closed world assumption and can be easily formalized using any of the available formalisms. This lack of dependence on the non-monotonic formalism is desirable for anyone keen on actually encoding information using the logic since it makes it less likely that their work is going to be undermined by subsequent changes in the logic.

The assorted (known and unknown) problems related to default reasoning are dealt with not in the logic, but by using axioms in the KB (they are as much a part of the KB as are the domain axioms). This not only gives us greater control over the conclusions drawn but also enables us to control what is concluded by changing the KB, something that is vastly easier than changing the logic. Since it is unlikely that the basic axioms (such as the Argumentation Axiom) are going to be removed from the KB, it is worth building faster inference schemes for using them. For example, one could provide procedural attachments for predicates such as argumentFor, etc., and exploit the fact that the basic structure of this formulation of default reasoning tries to mimic a reasoning process.

There are two main topics for our future work. The first is to obtain an efficient implementation capable of the full scheme presented here. The second and more important task is to develop a rich theory of arguments, their basic types and properties, and interesting preference criteria. We expect that as the Cyc group tries to axiomatize new domains, we will need new preference criteria. This should provide both a test for the existing framework and also give us better insights into the nature of default reasoning.

Acknowledgments

I would like to thank Doug Lenat and Mark Derthick for comments and suggestions, and them and David V. Wallace for helping me in the implemention. I thank Mary Shepherd, Karen Pittman, and the other members of the Cyc group for testing all this and putting up with the all the bugs in the implementation. I thank John McCarthy, Doug Lenat and Mark Derthick for comments on an earlier draft and Mary Shepherd for making this paper more readable.

References

[1] M. Derthick. An epistemological level interface for cyc. Technical Report ACT-CYC-084-90, MCC, February 1990.

[2] D.S.Touretsky. Implicit ordering of defaults in inheritance systems. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[3] D. B. Lenat and R. Guha. *Building Large Knowledge Bases.* Addison Wesley, Reading Mass., 1990.

[4] V. Lifschitz. Formal theories of action. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[5] J. McCarthy. First order theories of individual concepts and propositions. In H.Levesque and R.Brachman, editors, *Readings In Knowledge Representation.* Morgan Kaufmann, Los Altos, CA, 1986.

[6] J. McCarthy. Applications of circumscription to formalizing common sense knowledge. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[7] J. McCarthy and P.J.Hayes. Some philosophical problems from the standpoint of ai. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[8] R.V.Guha. Default reasoning using arguments. In preparation.

[9] R.V.Guha and D.B.Lenat. Cycl: The cyc representation language, part 3. Technical Report ACT-CYC-454-89, MCC, December 1989.

[10] S.Hanks and D.McDermott. Default reasoning, nonmonotonic logics and the frame problem. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[11] Y. Shoham. Chronological ignorance. In M. Ginsberg, editor, *Readings In Nonmonotonic Reasoning.* Morgan Kaufmann, Los Altos, CA, 1987.

[12] A. Tversky and D.Kahneman. Judgement under uncertainity: Heuristics and biases. *Science*, 185:1124–1131, 1974.