OSCAR: A Cognitive Architecture for Intelligent Agents

John L. Pollock
Department of Philosophy
University of Arizona
Tucson, Arizona 85721
pollock@arizona.edu
http://www.u.arizona.edu/~pollock

The "grand problem" of AI has always been to build artificial agents with human-like intelligence. That is the stuff of science fiction, but it is also the ultimate aspiration of AI. In retrospect, we can understand what a difficult problem this is, so since its inception AI has focused more on small manageable problems, with the hope that progress there will have useful implications for the grand problem. Now there is a resurgence of interest in tackling the grand problem head-on. Perhaps AI has made enough progress on the little problems that we can fruitfully address the big problem. The objective is to build agents of human-level intelligence capable of operating in environments of real-world complexity. I will refer to these as GIAs — "generally intelligent agents". OSCAR is a cognitive architecture for GIAs, implemented in LISP.[1] OSCAR draws heavily on my work in philosophy concerning both epistemology (Pollock 1974, 1986, 1990, 1995, 1998, 2008b, 2008; Pollock and Cruz 1999; Pollock and Oved, 2005) and rational decision making (2005, 2006, 2006a).

1. Epistemic Cognition

1.1 Reasoning in the Face of Pervasive Ignorance

The OSCAR architecture takes as its starting point the observation that GIAs must be able to form reasonable beliefs and make rational decisions against a background of pervasive ignorance. Reflect on the fact that you are a GIA. Then think how little you really know about the world. What do you know about individual grains of sand, or individual kittens, or drops of rain, or apples hanging on all the apple trees scattered throughout the world? Suppose you want to adopt a kitten. Most AI planners make the closed world assumption, which would require us to know everything relevant about every kitten in the world. But such an assumption is simply preposterous. Our knowledge is worse than just gappy — it is *sparse*. We know a very little bit about just a few of the huge number of kittens residing in this world, but we are still able to decide to adopt a particular kitten. Our knowledge of general matters of fact is equally sparse. Modern science apprises us of some useful generalizations, but the most useful generalizations are high-level generalizations about how to repair cars, how to cook beef stroganoff, where the fish are apt to be biting in Piña Blanca Lake, etc., and surely, most such generalizations are known to no one. What human beings know about the world is *many* orders of magnitude smaller than what is true of the world. And the knowledge we lack is both of individual matters of fact and of general regularities holding in the world.

In light of our pervasive ignorance, we cannot get around in the world just forming beliefs that follow deductively from what we already know together with new sensor input.[2] If we cannot rely exclusively upon deduction and draw only conclusions that are logically guaranteed to be true given what we already know, then we must allow ourselves to form beliefs that are only made probable by our evidence. And, of course, this is the way humans work. For instance, in our normal environment, objects generally have the colors they appear to have, so we can rely upon this statistical fact in forming beliefs about the colors of objects that we see. We normally assume that

[1] OSCAR and *The OSCAR Manual* can be downloaded from from the OSCAR website at http://oscarhome.soc-sci.arizona.edu/ftp/OSCAR-web-page/oscar.html.

[2] This might once have been denied, but it is a lesson that 20th and 21st century epistemologists have taken to heart, and no one doubts it any longer. See Pollock and Cruz (1999) for an extended discussion of these matters from the perspective of epistemology.

things are as they appear, and we will usually get things right by making this assumption. Similarly, objects tend to retain many of their properties over time, so if we observe an object at one time, we tend to assume that, in most respects, it has not changed a short time later. For instance, suppose I am comparing the time on my wristwatch with the time on the clock tower. I look first at the clock tower, and make a judgment about what it reads on the basis of how it looks to me. Then I turn to my wristwatch and make a similar judgment. But when I am looking at my wristwatch, I am no longer looking at the clock tower. So to make the comparison, I must assume that its reading has not changed dramatically in that brief interval. I am not guaranteed to be right, but I usually will be. After making such comparisons on several days, I may go on to conclude inductively that my watch always reads three minutes faster than the clock tower. Inductive inferences do not deductively guarantee their conclusions either. At best, inferences like these make their conclusions probable given the premises.

GIAs come equipped (by evolution or design) with inference schemes that tend to be reliable in the circumstances in which the agent operates. That is, if the agent reasons in that way, its conclusions will tend to be true, but are not guaranteed to be true. These built-in inference schemes enable the agent to get started making judgments about its environment. The agent can go on to use these built-in inference schemes to survey its world and form inductive generalizations about the reliability of new inferences that are not simply built into its architecture. But it needs the built-in inference schemes to get started. It cannot learn anything about probabilities without them.

Once the agent has used its built-in cognitive machinery to learn about new probabilities, it can then use those probabilities to license inferences not directly licensed by its built-in inference schemes. For instance, if the agent discovers that the probability of an *A* being a *B* is high, then if it has reason to believe that an object *c* is an *A*, it can reasonably infer that *c* is a *B*, and the probability of this conclusion being true is high. This is an instance of the *statistical syllogism*. But notice that in order for the agent to reason this way with new probability information, the statistical syllogism must be one of its built-in inference schemes.

1.2 The Need for Defeasible Reasoning

An agent whose reasoning is based on inference schemes that are less than totally reliable will encounter two kinds of problems. First, reasoning in different ways (employing different subsets of the built-in inference schemes) can lead to conflicting conclusions. In the literature on defeasible reasoning, this is what is known as "rebutting defeat". For example, I may look at an object and note that it looks red to me. This gives me a reason for concluding that it is red. But my colleague Claudio, whom I regard as highly reliable, may tell me that it is not really red. Because I believe that Claudio is highly reliable (i.e., what he tells me is generally true), I can use his statement in conjunction with the statistical syllogism and acquire a reason for concluding that it is not red. So I have a reason for thinking that the object is red, and another reason for thinking it is not. An agent that reasons in ways like this needs some cognitive mechanism for deciding which conclusions to adopt when conflicts arise. The built-in inference schemes will have "strengths" associated with them. Some reasons are better reasons for their conclusions than other reasons are. In a well-designed agent, these strengths will track the reliability of the inference scheme. These strengths can be used in adjudicating disputes when we have reasons for conflicting conclusions. The conclusion for which we have the better reason wins.

Second, the reliability of an inference scheme can vary in different circumstances. For instance, in humans color vision is pretty reliable, but not when things are illuminated by colored lights. This is something that we discover inductively, and we can use that to "tune" our use of the built-in inference schemes in specific circumstances, lowering the degree of justification of a conclusion drawn from the inference scheme. We may also discover circumstances under which the inference scheme is totally unreliable, in which case its use conveys no justification to the conclusion. In those circumstances, we should make no inference, or if we have already made an inference, we should withdraw the conclusion. This is one kind of "undercutting defeat".[3] Undercutting defeaters attack an inference without attacking the conclusion itself. For instance, if I know that illumination by red light can make an object look red when it is not, and I see an object that looks red but I know that it is illuminated by red lights, I should refrain from concluding that it is red. But it might still be red. We can think of these considerations as giving us a reason for believing that (under the present circumstances) the object's looking red does not guarantee that it is red. It will be convenient to

[3] I called these "reliability defeaters", in (Pollock 1995, 1998) and (Pollock and Cruz 1999).

symbolize this as "(x looks red) $\otimes$ (x is red)".

As was just illustrated, we can discover new probabilistic information that provides us with an undercutting defeater for an inference that is based on a built-in inference scheme. Sometimes the system designer (or evolution) will already have noted this and built corresponding undercutting defeaters into the system so that the cognizer can take advantage of them without having to first make the empirical discovery. In the human cognitive architecture, we find a rich array of built-in inference schemes and attendant undercutting defeaters. One of the tasks of the philosophical epistemologist is to try to spell out the structure of these inference schemes and defeaters. I have made a number of concrete proposals about specific inference schemes (1974, 1986, 1990, 1995, 1998, 1999, 2008). Some of these proposals have been implemented and can be used by OSCAR. (The OSCAR architecture is modular, so one can freely add or delete inference schemes, as one see fit.) My experience has been that in analyzing specific kinds of defeasible reasoning, the hardest task is often to get the undercutting defeaters right. For example, in my (1998) I argued that the frame problem is easily solved if we correctly characterize the undercutting defeaters that are associated with the defeasible inference schemes that we employ in reasoning about causation, and I implemented the solution in OSCAR. To illustrate OSCAR's reasoning, I will discuss this example in more detail in section 4.

1.3 The Defeat-Status Computation

Deductive reasoning is "monotonic". Given acceptable premises, once an argument has been constructed for a conclusion, it becomes reasonable to accept the conclusion, and further reasoning is irrelevant. But defeasible reasoning is nonmonotonic. On the other hand, argument construction is still monotonic. That is, we build new arguments by recursively extending the previous set of arguments. But constructing an argument for a conclusion no longer guarantees that the conclusion is acceptable, because we can have other arguments that conflict with the given argument. We end up with a network of interacting arguments, some of which may support defeaters for some of the steps of others. Jointly, these constitute an *inference graph*. The logical problem then arises of deciding what conclusions a cognizer should accept on the basis of its entire inference graph. Theories of defeasible reasoning aim at solving this logical problem. This is the *defeat status* computation which computes which conclusions to accept and which to regard as defeated.

There are a number of different theories about how the defeat status computation should work. Before we can implement a defeat status computation, we need a characterization of the set of conclusions that should be regarded as undefeated given any particular inference graph. Such a characterization constitutes a "semantics for defeasible reasoning", and it provides the target for the defeat status computation.[4]

The ultimate test of a semantics for defeasible reasoning must be that it gives the right answer in cases in which we have clear intuitions. Let the *initial nodes* of an inference graph be those encoding "given" information — premises, suppositions, or perceptual input. An *argument* is a tree of inferences whose root nodes are initial nodes. Then in simple cases we can compute defeat statuses recursively using the following four rules:

(1) A conclusion is undefeated (relative to an inference graph) iff either it is an initial node or it is supported by at least one undefeated argument in the inference graph.

(2) An argument is undefeated iff every inference in the argument is undefeated.

(3) If an inference graph contains an undefeated argument supporting a defeater for an inference used in one of its arguments A, then A is defeated.

(4) If an inference graph contains no undefeated arguments supporting defeaters for inferences used in one of its arguments A, then A is undefeated.

For example, consider the inference graph diagrammed in figure 1, where the thin black arrows indicate inference links and the thick red arrows indicate defeat relations. By (1), A, C, and F are undefeated. Then by (4), ($F \otimes G$) is undefeated. Then by (3), the inference from F to G is defeated, and hence by (1) and (2), G is defeated. Then by (2) and (3), ($B \otimes H$) is defeated. Finally, by (4), H is undefeated.

[4] See Prakken and Vreeswijk (2001) for an excellent survey of recent semantical theories for defeasible reasoning.

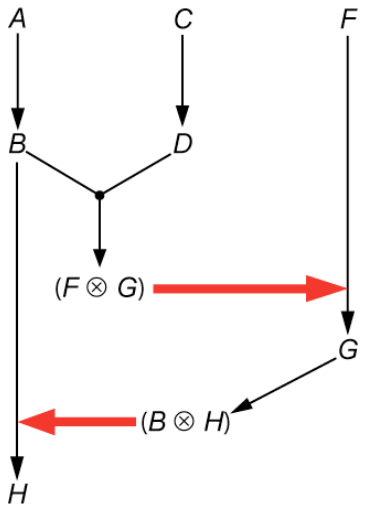

Figure 1. A simple inference graph

However, inference graphs can contain loops that make it impossible to apply these rules recursively. The simplest instance of this is a case of "collective defeat". Collective defeat arises when we have a set of two or more arguments and each argument in the set is defeated by some other argument in the set. Consider the inference graph of figure 2. Here we have no arguments lacking defeating arguments, so there is no way for the recursion to get started. What should we believe in such a case? Consider a simple example. Suppose we are in a closed and windowless room, and Jones, whom we regard as reliable, enters the room and tells us it is raining outside. Then we have a reason for believing it is raining. But Jones is followed by Smith, whom we also regard as reliable, and Smith tells us it is not raining. We have arguments for both "It is raining" and "It is not raining". What should we believe? It seems clear that in the absence of any other information, we should not form a belief about the weather. We should withhold belief, which is to treat both arguments as defeated. But this means that in cases of collective defeat, an argument is defeated even though its defeating arguments are also defeated. Accordingly, cases of collective defeat violate principle (4).

Figure 2. Collective defeat

Cases of collective defeat are encountered fairly often. Consider the *lottery paradox* (Kyburg, 1961). Suppose you hold a ticket in a fair lottery consisting of one million tickets. It occurs to you that the probability of any particular ticket being drawn is one in a million, and so in accordance with the statistical syllogism you can conclude defeasibly that your ticket will not be drawn. Should you throw the ticket away? Presumably not, because that probability is the same for every ticket in the lottery. Thus you get an equally good argument for each ticket that it will not be drawn. However, you are given that the lottery is fair, which means in part that some ticket will be drawn. So you have an inconsistent set of conclusions, viz., of each ticket n you have the conclusion $\sim D_n$ that it will not be drawn, but you also have the conclusion that some one of them will be drawn. This generates a case of collective defeat. Because the set of conclusions $\sim D_1$, $\sim D_2$,…, $\sim D_{1,000,000}$, $D_1 \vee \ldots \vee D_{1,000,000}$ is logically inconsistent, each subset of the form

$$\sim D_1, \sim D_2, \ldots, \sim D_{i-1}, \sim D_{i+1}, \ldots, \sim D_{1,000,000}, D_1 \vee \ldots \vee D_{1,000,000}$$

entails the negation of the remaining member, i.e., entails D_i. So from each such subset of

conclusions in the graph we can get an argument for the corresponding D_i, and that is a rebutting defeater for the argument to $\sim D_i$. More simply, pick one ticket. We have reason to think that it will lose. But we also have a reason to think it will win because we have reason to think that all the others will lose, and we know that one has to win. This yields the inference graph of figure 3. Here I have used dotted arrows for the inference supporting $D_{1,000,000}$, dashed arrows for the inference supporting D_2, and arrows combining dashes and dots for the inference supporting D_1. Thus for each conclusion $\sim D_i$ we can derive the rebutting defeater D_i from the other conclusions $\sim D_j$. For each ticket we have an argument for the conclusion that it will win, and also an argument for the conclusion that it will lose. Hence we have a case of collective defeat. Accordingly, given a theory of defeasible reasoning that can handle inference graphs with collective defeat, the lottery paradox is resolved by observing that we should not conclude of any ticket that it will not be drawn. (Of course, we can still conclude that it is highly unlikely that it will be drawn, but that yields no inconsistency.)

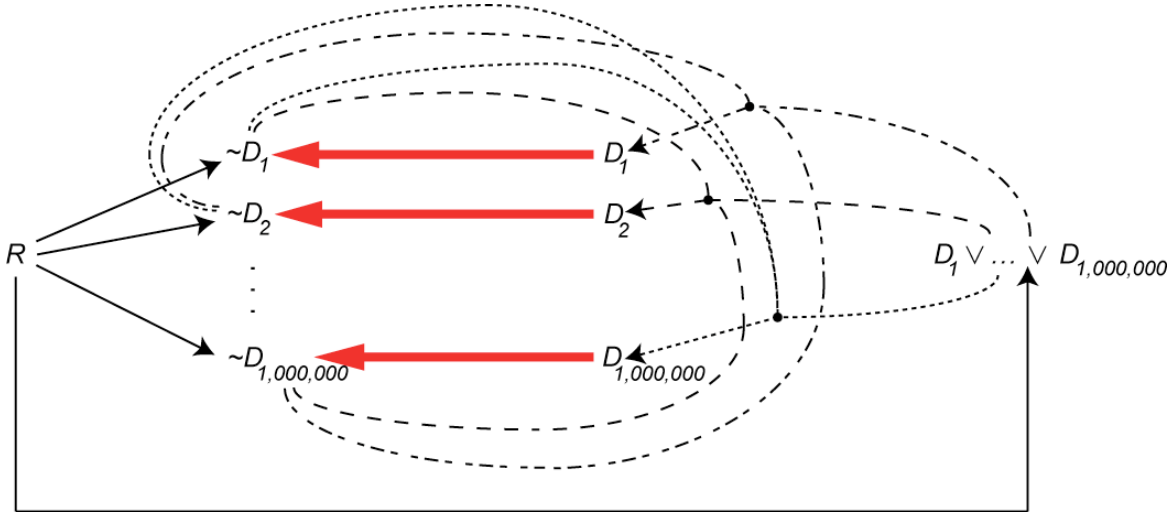

Figure 3. The lottery paradox as a case of collective defeat

Most theories of defeasible reasoning have some mechanism that enables them to get collective defeat right. But there is another kind of case they often have trouble with. This concerns "self-defeating arguments" that support defeaters for themselves. Figure 4 is a simple example of a self-defeating argument. In this example, it seems clear that Q should be defeated. If Q were undefeated, then R would be undefeated, because there is no defeater for the inference from Q to R, and then $(P \otimes Q)$ would be undefeated because it is inferred from R. But if $(P \otimes Q)$ is undefeated, Q must be defeated instead of undefeated. Thus Q has to be regarded as defeated. It seems to follow that R and hence $(P \otimes Q)$ are also defeated. If principle (4) above were correct (we have already seen that it is not), then it would follow that Q is undefeated rather than defeated. So this is another example in which principle (4) fails.

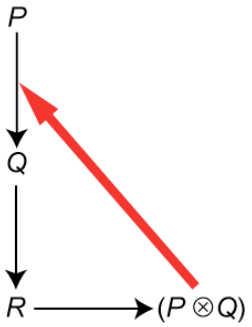

Figure 4. A self-defeating argument.

Although all standard theories of defeasible reasoning can handle simple cases of collective defeat, many of them have more trouble with self-defeat. For example Reiter's (1980) default logic has been quite popular in AI, but it is unable to distinguish between the status of P (which ought to be undefeated) and Q, R, and $(P \otimes Q)$ in figure 4, so it must either rule them all defeated or all undefeated. But either conclusion is unacceptable.[5]

Even if a theory can handle both collective defeat and simple cases of self-defeat, it may not be able to handle combinations of the two. Consider the example of figure 5. This results from extending the inference graph for the lottery paradox by noting two things. First, we typically have only a defeasible reason P for believing the description R of the lottery. For example, we may read it in the newspaper. Second, we can combine the arguments for the individual $\sim D_i$'s to obtain an argument for $\sim(D_1 \vee \ldots \vee \sim D_{1,000,000})$, and that yields an argument for $\sim R$ (because R entails $D_1 \vee \ldots \vee D_{1,000,000}$). Thus the argument from P is self-defeating. I call this "the lottery paradox paradox" (Pollock 1991).

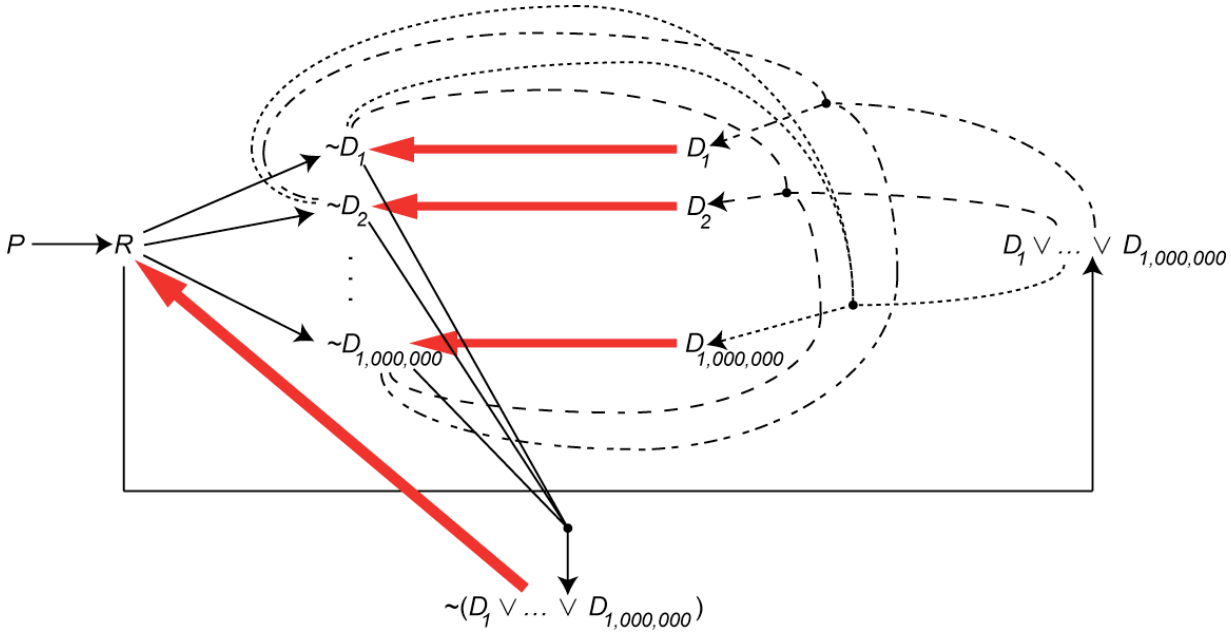

Figure 5. The lottery paradox paradox

If we distill the self-defeating subargument involving R out of figure 5, we get the inference graph of figure 6. This has the essentially the same structure as figure 4, so if we give it the same treatment we should end up concluding that we are not justified in believing R. That is, we should not believe the description of the lottery we get from the newspaper report. But that is clearly wrong — of course we should believe it. So apparently the other parts of the inference graph change its structure in ways that alter the way the defeat statuses are computed.

[5] Technically, default logic has the consequence that there are no "extensions" for this default theory.

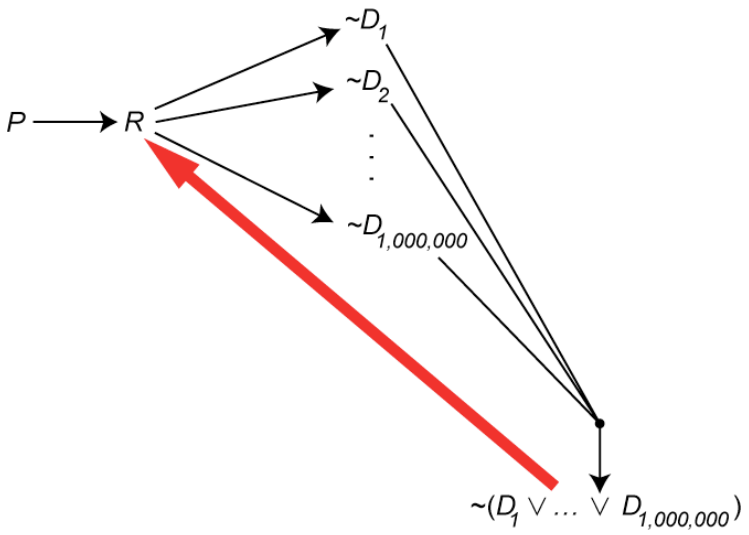

Figure 6. The self-defeating sub-argument embedded in the lottery paradox paradox

The lottery paradox paradox is a counterexample to the semantics for defeasible reasoning that I proposed in Pollock (1986). Other theories also have trouble with it. For example, simple versions of circumscription (McCarthy 1980) pronounce *R* defeated when it should not be.[6] However, in Pollock (1995) I proposed a semantics that yields the intuitively correct answers in all of these examples.

My (1995) semantics turns upon principles (1) - (4) above. An *inference/defeat loop* is a loop constructed by following a path consisting of inference links and defeat links. When an inference graph contains no inference/defeat loops, there is just one way of assigning defeat statuses that is consistent with (1) - (4), and we can compute it recursively. In that case, (1) - (4) seem to give the right answer. But in the presence of inference/defeat loops, there is no way to apply (1) - (4) recursively. This reflects the fact that there may be multiple ways of assigning defeat statuses that are consistent with (1) - (4). For example, in figure 2, there are two different ways of assigning defeat statuses to the conclusions making up the inference graph in such a way that principles (1) - (4) are satisfied. This is diagramed in figure 7, where "+" indicates an assignment of "undefeated" and "–" indicates an assignment of "defeated". The conclusions that we want to regard as undefeated simpliciter are those that are assigned "+" by all of the ways of assigning defeat statuses consistent with (1) - (4).

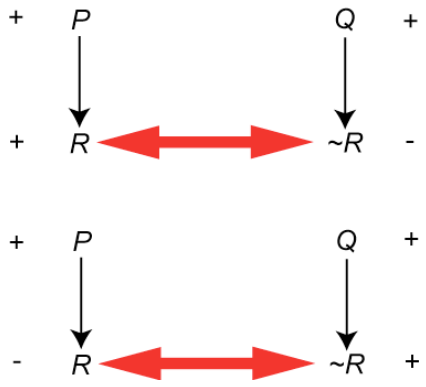

Figure 7. Two ways of assigning defeat statuses

The lottery paradox works similarly. For each *i*, there is a way of assigning defeat statuses

[6] There are many forms of circumscription, and by using what are essentially ad hoc prioritization rules it may be possible to get the right answer in figure 5. But because the moves required are ad hoc, I don't think this shows anything.

according to which $\sim D_i$ is assigned "–", but for all $j \neq i$, $\sim D_j$ is assigned "+". So again, the conclusions that are intuitively undefeated are those that are always assigned "+".

If we turn to the lottery paradox paradox, the same thing holds. There are the same 1,000,000 assignments of defeat statuses, but now for every one of them $\sim D_1$ & … & $\sim D_{1,000,000}$ is assigned "–", and hence "$\sim R$" is assigned "–" and "R" is assigned "+". Thus we get the desired result that we are justified in believing the description of the lottery, but we are not justified in believing that any particular ticket will not be drawn.

However, when we turn to the simpler case of self-defeat in figure 4, things become more complicated. There is no way to assign defeat statuses consistent with principles (1) – (4). By (1), P must be assigned "+", which is unproblematic. But there is no way to assign a defeat status to Q. Suppose we assign "+" to Q. Then we must assign "+" to R and to $(P \otimes Q)$. But then we would have to assign "–" to Q, contrary to our original assignment. If instead we assign "–" to Q, then we must assign "–" to R and to $(P \otimes Q)$. But then we would have to assign "+" to Q, again contrary to our original assignment. So in this example there can be at most a partial assignment of defeat statuses consistent with (1) – (4). On the other hand, it remains true that the intuitively undefeated conclusions are those that are assigned "+" in all partial status assignments that assign statuses to as many conclusions as possible. Let us define:

> A *partial status assignment* is an assignment of defeat statuses ("+" or "–") to a subset of the conclusions and arguments of an inference graph in a manner consistent with principles (1) - (4).
>
> A *status assignment* is a maximal partial status assignment, i.e., a partial status assignment that cannot be extended to further conclusions or arguments and remain consistent with principles (1) - (4).

My (1995) proposal was then:

The defeat status of an argument

> An argument is undefeated (relative to an inference graph) iff it is assigned "+" by every status assignment for that inference graph.

It would be natural to propose:

> A conclusion is undefeated (relative to an inference graph) iff it is assigned "+" by every status assignment for that inference graph.

Indeed, this works in all the preceding examples, but that is only because, in those examples, there are no conclusions supported by multiple arguments.[7] To see that this does not work in general, consider the case of collective defeat diagramed in figure 8. Once again, there are two status assignments. One assigns "+" to R, S, and $(S \vee T)$, and "–" to $\sim R$ and T. The other assigns "–" to R and S and "+" to $\sim R$, T, and $(S \vee T)$. On both assignments, $(S \vee T)$ is assigned "+". However, there is no argument supporting $(S \vee T)$ all of whose inferences are undefeated relative to both assignments, so there is no undefeated argument supporting $(S \vee T)$. If we regard $(S \vee T)$ as undefeated, then we are denying principle (1), according to which a non-initial node is only undefeated if it is supported by an undefeated argument. However, it seems that principle (1) ought to be true, so instead of the preceding I proposed:

The defeat status of a conclusion

> A conclusion is undefeated iff it is supported by an undefeated argument.

[7] This observation is due to Makinson and Schlechta (1991).

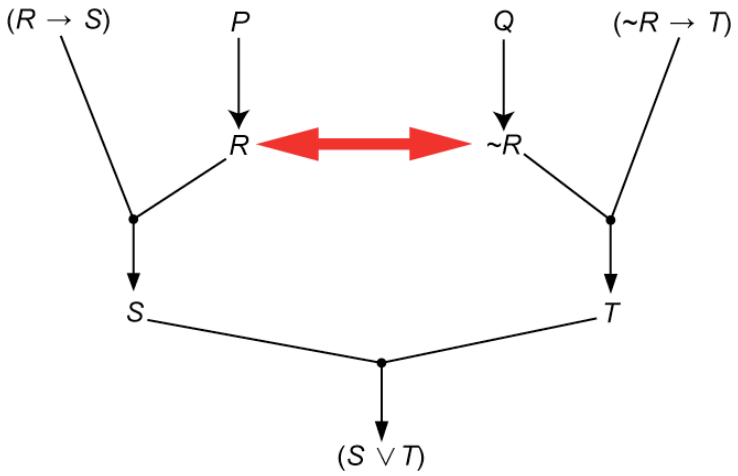

Figure 8. Collective defeat with multiple arguments

Originally (1980 - 1993), OSCAR was based on a semantics (Pollock 1986) that was later proven equivalent (Prakken and Vreeswijk 2001) to Dung's (1995) subsequently developed "admissible model semantics". More recently (since 1993), in response to difficulties involving self-defeating arguments (Pollock 1991), OSCAR has been based on a semantics (Pollock 1994, 1995) that was recently proven (Vo, et al, 2005) equivalent to the subsequently developed "preferred model semantics" of Bondarenko, et al (1997).

OSCAR incorporates an efficient algorithm for computing the defeat statuses dictated by this semantics. That algorithm is described in my (1995), and more fully in *The OSCAR Manual*. It is to be emphasized, however, that the user of OSCAR is not locked into this semantics. OSCAR is modular, and any other defeat status computation can be used in its place.

Most semantics for defeasible reasoning ignore a crucial problem. It was noted above that some inference schemes support their conclusions more strongly than others do. Ideally, the strength of an inference scheme should track its reliability. A semantics for defeasible reasoning should take account of these differences and use it in determining which conclusions to accept when there are conflicts. As far as I know, OSCAR is the only implemented system that tries to handle this, although I am not confident that OSCAR does this correctly.[8]

1.3 Implementing Defeasible Reasoning

A defeasible reasoner must construct arguments, use them to build an inference graph, and then decide what to believe by computing the defeat statuses of the conclusions drawn relative to its entire inference graph. The natural temptation is to try to build an implemented defeasible reasoner on the model of familiar deductive reasoners. For instance, first-order deductive reasoners generate the members of the recursively enumerable set of deductive consequences of the given premises. By Church's theorem, this set of consequences is not decidable, but because it is recursively enumerable, its members can be systematically generated by an algorithm for constructing arguments. (This is what the completeness theorem for first-order logic establishes.) However, the situation in defeasible reasoning is more complex. If we assume that it is not decidable whether there is an argument supporting a particular conclusion (for first-order logic, this is Church's theorem), then it cannot be decidable whether there are arguments supporting defeaters for a given argument. This means that in constructing defeasible arguments, we cannot wait to rule out the possibility of defeat before adding a new step to an argument. We must go ahead and construct arguments without worrying about defeat, and then as a second step, compute the defeat statuses in terms of the set of arguments that have been constructed. Accordingly, argument construction must be separated from the defeat status computation. Many implemented systems of defeasible reasoning do not make this separation, and as a result they are forced to focus exclusively on decidable underlying logics, like the propositional calculus. But the expressive power of languages having decidable logics is too weak for a GIA. We need at least full first-order logic.

We cannot wait until all possibly relevant arguments have been constructed before we compute

[8] See my 2002 for a more recent attempt. My research on this topic is ongoing.

defeat statuses, because the process of argument construction is non-terminating. We must instead compute defeat statuses *provisionally*, as we go along. Agents have to act, and they cannot wait until the end of a non-terminating search for defeaters. Decisions about how to act must be based on the conclusions that are undefeated relative to all the arguments the agent has so far constructed. But the agent should also be prepared to change its mind about defeat statuses if it finds new relevant arguments. In other words, the defeat status computation must itself be defeasible. Notice that this is precisely the way human reasoning works. We decide whether to accept conclusions on the basis of what arguments are currently at our disposal, but if we construct new arguments that are relevant to the conclusion, we may change our mind about whether to accept the conclusion. I put this point in my (1995) by saying that there are two kinds of defeasibility. The literature on nonmonotonic logic and most of the literature on defeasible reasoning has focused on what might be called *simple defeasibility*. This is defeasibility that arises from the fact that newly discovered information can lead to the withdrawal of previously justified conclusions. But as we have seen, there is a second source of defeasibility that arises simply from constructing new arguments without adding any new information to the system. We can put this by saying that the reasoning is *doubly defeasible*.[9]

There is a technical point behind all of this. Let us say that a conclusion is *justified* relative to an inference graph iff the defeat status computation renders the conclusion undefeated relative to that inference graph. The justified conclusions (relative to the agent's current inference graph) are those that the agent currently has at its disposal, and if the agent must take action, its actions must be based upon its currently justified conclusions. As we have seen, given a sophisticated logic, argument construction is non-terminating, so there will never be a time at which a GIA has added all possible arguments to its inference graph. Nevertheless, we can talk about the inference graph that consists of all possible arguments that can be constructed on the basis of a fixed set of inputs. Let us say that a conclusion is *warranted*, relative to those inputs, iff it is justified relative to that maximally comprehensive inference graph. The set of warranted conclusions is, in a sense, what the epistemic cognition of a GIA is aiming at, but because reasoning is non-terminating, the justified conclusions of a GIA can only approximate the set of warranted conclusions.

If a GIA did only deductive reasoning, then once a conclusion became justified, it would remain justified as the agent did more reasoning and added more arguments to its inference graph. This corresponds to the fact that (for a complete logic, like first-order logic) the set of deductive consequences of a set of inputs is recursively enumerable. A recursively enumerable set A is one that can be "approximated from below". More precisely, there is an algorithm for generating finite sets A_i such that for each i, j, if $i < j$ then $A_i \subseteq A_j$, and A is the union of all the A_i's. That is, $A = \bigcup_i A_i$. For deductive reasoning, if A is the set of warranted conclusions (i.e., deductive consequences of the input), we can take A_i to be the set of conclusions justified after i steps of reasoning.

However, for a reasoner that performs defeasible reasoning, adding arguments to the inference graph can result in previously justified conclusions becoming unjustified. The result, noted long ago by Reiter (1980) and Isreal (1980), is that for defeasible reasoning, the set of warranted conclusions is not recursively enumerable. It is instead Δ_2 in the hyperarithmetic hierarchy (Pollock 1992, 1995). If a set A is Δ_2 but not recursively enumerable, it can still be systematically approximated, but not from below. Instead, there is an algorithm for generating finite sets A_i such that (1) for each member x of A, there is an n such that for all $i > n$, $x \in A_i$, and (2) for each x not in A, there is an n such that for all $i > n$, $x \notin A_i$. In other words, as we progress through the sequence of A_i's, the status of each member or non-member of A eventually stabilizes so that it is in A_i iff it is in A. But for each i, A_i will both fail to contain some members of A and contain some non-members of A. It is the latter that makes the set non-recursive. Technically, $A = \bigcap_i \bigcup_{j>i} A_j$. You can think of this on the analogy of a reverberating soap bubble whose reverberations dampen out over time. The A_i's correspond to the volumes of space occupied by the soap bubble at different times, and A corresponds to the limit to which the A_i's go as the reverberations dampen out. Each A_i fails to contain some of the spatial region contained in A, and it also contains some points lying outside of A, but as the reverberations dampen out, this divergence gets smaller and smaller. For defeasible reasoning, if A is the set of warranted conclusions, we can again take A_i to be the set of conclusions justified after i steps of

[9] In my (1995) I used the less perspicuous terminology "synchronically defeasible" and "diachronically defeasible".

reasoning.

It is because the set of warranted conclusions for a GIA doing defeasible reasoning is Δ_2 that its reasoning must be doubly defeasible. It is impossible to build the set of warranted conclusions by systematically approximating it from below, as a deductive reasoner does. A defeasible reasoner must be prepared not only to add new conclusions, but also to retract old conclusions in the face of new reasoning. To the best of my knowledge, OSCAR is the only implemented defeasible reasoner that accommodates double defeasibility. Accordingly, OSCAR is the only implemented defeasible reasoner that can make use of languages with strong underlying deductive logics like first-order logic.

I have made four important observations about the reasoning of a GIA. First, it must have an expressive system of representations for encoding its beliefs, including at least a full first-order language. Second, this means that it must employ powerful systems of reasoning, including at least first-order logic. Third, this means that argument construction will be non-terminating. There can always be more relevant arguments to be constructed. Accordingly, the existence of arguments supporting particular conclusions is undecidable. But fourth, that forces the reasoning to be doubly defeasible.

These observations have profound implications for the logical structure of a GIA. In particular, they have the consequence that a GIA cannot be viewed as a problem solver, in the classical AI sense. The life of a GIA does not consist of being presented with (or constructing) a sequence of isolated problems, solving each in turn and being done with it, and then going on to the next. Because reasoning is non-terminating, a GIA can only solve problems defeasibly. When the agent has to act, it acts on the basis of the solutions it has found to date, but if it has more time, there is always the possibility that better solutions will emerge or that difficulties will be found for previously discovered solutions. When defeasible solutions are found, problems can be set aside, or at least lower priority can be assigned to further reasoning about the problems, but problems can never been forgotten. There is always the possibility that an agent will have to return to a particular problem. Furthermore, solutions that were found for old problems but have not yet been acted upon may interact with what solutions are available for new problems, and that may necessitate looking for different solutions for the old problems. For a GIA (as for humans), nothing is ever completely finished. An implementation of a GIA is an infinite loop, not a finitely terminating problem solver.

2. Practical Cognition

Thus far I have focused on those aspects of cognition that issue in beliefs. However, agents are most fundamentally entities that act on their environment, and the main purpose of cognition is to direct action. In crude agents, actions may be nothing but reflex responses to sensor input, but in sophisticated agents, actions are directed in part by appeal to the agents' beliefs about their environment. The cognitive processes that issue in action make up what is called "practical cognition". As we will see below, the distinction between epistemic and practical cognition is not a clean one, but it is useful nevertheless. Roughly, practical cognition consists of those aspects of cognition that deal with adopting and executing plans.

We can distinguish between plan construction and plan adoption. In sophisticated agents, multiple plans may be constructed, aiming both at the same goal and at different goals. Plans aimed at different goals may interact in various ways (competing for resources, sharing actions, etc.), and this affects which plans the agent should adopt. So plan construction is one thing, plan adoption another. I will talk about plan adoption below, but first let us think a bit about plan construction. In a GIA, should that be part of epistemic cognition, or should it be a separate kind of cognition?

2.1 High Performance Planners

In recent years, there have been impressive advances in planning theory, resulting in "high performance planners" that are capable of solving much harder problems than classical goal regression planners could solve. Unfortunately, most of this work is inapplicable to building GIAs. This is for two reasons. First, existing high performance planners invariably make the closed world assumption, and do so in an absolutely essential way that cannot be relaxed. But this flies in the face of our initial observation of pervasive ignorance. A GIA cannot be constructed on the basis of any planner that requires the closed world assumption.

Second, a GIA will also lack knowledge of what the outcomes of actions will definitely be. It will at best be able to predict some outcomes probabilistically, and other outcomes will be completely unexpected and unpredictable. Accordingly, planning must be based on probabilities. Furthermore, GIAs do not face isolated planning problems. As an agent learns more about its environment, new opportunities arise and new threats arise, resulting in new planning problems. It may not be possible for an agent to achieve all its goals. So even if it can construct a plan for achieving a goal, that does not mean that the plan should be adopted. It must be considered how the plan is related to other plans. If the plan is adopted, will the agent have to reject another plan because, for example, it lacks the resources to execute both? In choosing between conflicting plans, the agent must take account of their costs and benefits. So the requisite kind of planning for a GIA is probabilistic and decision-theoretic.

Existing decision-theoretic planners almost invariably make two kinds of assumptions that are wholly unrealistic when applied to GIAs. First, they make the closed world assumption regarding individual matters of fact in the start state. Second, they assume that the cognitive agent has at its disposal a complete probability distribution regarding the probabilistic connections between any matters of fact that are relevant to the planning problem. For example, they often use this to encode information about actions in Bayesian nets. We have already seen that the closed world assumption is unacceptable for GIAs, but the assumption of a complete probability distribution is even more preposterous. Suppose a problem is described by logical compounds of a set of simple propositions $P_1,\ldots,P_n$. Then to be able to compute the probabilities of all logical compounds of these simple propositions, what we must generally know is the probabilities of every conjunction of the form $((\sim)P_1\&\ldots\&(\sim)P_n)$. The tildes enclosed in parentheses can be either present or absent. These n-fold conjunctions are called *Boolean conjunctions*, and jointly they constitute a partition. Given fewer than all but one of them, the only constraint the probability calculus imposes on the probabilities of the remaining Boolean conjunctions is that the sum of all of them must be 1. Together, the probabilities of all the Boolean conjunctions determine a complete probability distribution — an assignment of unique probabilities to every logical compound of the simple propositions.

In decision-theoretic planning, it is generally assumed that we come to a problem equipped with a complete probability distribution. However, in real life this assumption is totally unrealistic. In general, given n simple propositions, there will be 2^n logically independent probabilities of Boolean conjunctions. For a rather small number of simple propositions, there is a completely intractable number of logically independent probabilities. For example, given just 300 simple propositions, a grossly inadequate number for describing many real-life problems, there will be 2^{300} logically independent probabilities of Boolean conjunctions. 2^{300} is approximately equal to 10^{90}. To illustrate what an immense number this is, recent estimates of the number of elementary particles in the universe put it between 10^{80} and 10^{85}. Thus to know the probabilities of all the Boolean conjunctions, we would have to know 5 – 10 orders of magnitude more logically independent probabilities than the number of elementary particles in the universe. And this is from a problem that can be described in terms of just 300 simple propositions. In the real world we need vastly more.

Lest one think this is an unrealistic problem, consider a simple example. In my (2006a) I described a challenge problem for AI planners. This problem generalizes Kushmerick, Hanks and Weld's (1995) "slippery gripper" problem. We are presented with a table on which there are 300 numbered blocks, and a panel of correspondingly numbered buttons. Pushing a button activates a robot arm which attempts to pick up the corresponding block and remove it from the table. We get 100 dollars for each block that is removed. Pushing a button costs two dollars. The hitch is that half of the blocks are greasy. If a block is not greasy, pushing the button will result in its being removed from the table with probability 1.0, but if it is greasy the probability is only 0.1. We are given exactly 300 opportunities to either push a button or do nothing. Between button pushes, we are given the opportunity to look at the table, which costs one dollar. Looking will reveal what blocks are still on the table, but will not reveal directly whether a block is greasy. What should we do? Humans find this problem terribly easy. Most people quickly produce the optimal plan: push each button once, and don't bother to look at the table. But when I last surveyed them, existing AI planners could not even encode this problem, much less solve it. The difficulty is that there are too many logically independent probabilities. For every subset K of the 300 blocks, let $p_{K,i}$ be the probability that, when K is the set of blocks on the table, block i is still on the table after the button corresponding to block i is pushed. There are 2^{300} choices of K, so there are more than 2^{300} probabilities $p_{K,i}$ such that $i\in K$. Furthermore, none of them can be derived from any of the others. Thus they must each be encoded separately in describing a complete probability distribution for the problem. It is impossible for a

real cognitive agent to encode such a probability distribution.

The encoding of a probability distribution can often be simplified considerably if we can assume that most propositions are statistically independent of one another. Then there are compact techniques for storing complete probability distributions using Bayesian nets (Pearl 1988). The use of Bayesian nets allows us to explicitly store just a subset of probabilities that cannot be derived from each other, and provides an efficient inference mechanism for recovering derivable probabilities from them. However, this is not the entire solution to the problem of sparse probability knowledge, because in the slippery blocks problem, none of the probabilities $p_{K,i}$ can be derived from others, so they would all have to be encoded separately in a Bayesian net, and that would make the Bayesian net impossibly large.

The upshot is that most existing work in decision-theoretic planners is not directly applicable to building GIAs.

2.2 Reasoning vs. Computing

There is a more profound difficulty for using existing planning technology in a GIA. Existing planners "compute plans". Their input is a planning problem which includes all of the information needed to find a plan, and then they run a program that computes a plan and terminates. This strategy is inapplicable to GIAs, for several reasons.

First, in the real world a GIA cannot be expected to come to a planning problem already knowing everything that is relevant to solving the problem. In the course of trying to construct a plan, an agent will typically encounter things it would like to know but does not know. For instance, if it is planning how to make a peanut butter and jelly sandwich, it may discover that it does not know where the peanut butter is. This illustrates that addressing a typical planning problem may give rise to epistemic queries which will initiate further epistemic cognition. Planning and epistemic cognition must be smoothly interleaved. This cannot be handled by doing all the epistemic cognition before beginning the planning, because we often do not know what information we will need until the plan is partly constructed. For example, in a goal regression planner we may need to find out whether the preconditions for some action are satisfied, but we will not know which actions are relevant until the planning is already underway and a partial plan constructed. Most emphatically, we cannot require a GIA to already know everything that might be relevant.

A partial response to the need for interleaving planning and epistemic cognition is to engage in hierarchical planning. For instance, if a human is faced with a planning problem like that of whether and how to attend a conference, he or she constructs a schematic plan involving high-level actions like "fly to Los Angeles" and "rent a car". Only later do they engage in further planning regarding how to fill in the details. This is for two separate reasons. First, planning is a computationally difficult and time-consuming process. An agent may have to make decisions (e.g., whether to accept an invitation to give a talk at the conference) that depend upon whether one can construct a good plan for attending the conference. Constructing a schematic plan may suffice for reasonably believing that one will be able to construct a full plan, because one may have general beliefs about the circumstances under which such schematic plans can be expanded. Thus one can quickly make the decision about whether to attend the conference, and then work out the details later. Second, and more important for present purposes, one often lacks the knowledge required to expand the schematic plan and has no way to get that knowledge until a later time. For instance, I might agree to pick up a friend at the airport without knowing the exact time at which his flight will arrive (it is "sometime Thursday afternoon"). This may be knowledge I cannot possibly acquire until shortly before the plane arrives (my friend may not even have purchased his ticket yet). So it is impossible to construct more than a schematic plan, but it is essential to do that much before agreeing to pick him up. Then when I learn more, I can expand the plan. We almost never expand schematic plans fully until we begin executing them. In the process of executing the initial steps, we typically acquire further knowledge that is required for expanding the later parts of the plan. For instance, consider planning a route for driving across a strange city. You will decide what highways to take, where to turn, etc., but you do not expand your plan to the point of deciding what lanes to drive in until you are actually executing the plan and can collect information about the local traffic flow. This illustrates that planning and knowledge acquisition must be interleaved. In building a GIA, we cannot assume that the agent knows everything it needs to know before it begins either planning or plan execution.

Many of these observations have been noted before by members of the planning community, but what has not generally been appreciated is that they profoundly change the logic of planning. If

planning and epistemic cognition are essentially interleaved, then because the epistemic cognition is defeasible, so must be the plan construction. If a plan is constructed by assuming beliefs that are later withdrawn, the plan should be withdrawn. In principle, this might be handled by replanning from scratch, but because planning is so computationally difficult, that would be very inefficient. It seems that it would be better to keep track of where in the planning process various beliefs are used, and then try to repair the plan if those beliefs are withdrawn. This is the way defeasible reasoning works. We have to keep track of our arguments so that we know what conclusions were used in inferring a given conclusion, and then we use that information in deciding which conclusions to accept in the face of defeaters. It looks like the structure of defeasible planning must be exactly similar, and will require all the same sophistications as full-fledged defeasible epistemic cognition. Rather than duplicating all of this in a separate planning module, and somehow integrating that module with epistemic cognition, it seems more efficient to do it all with a single module. In other words, do defeasible planning within the system of epistemic cognition by reasoning defeasibly about plans. It is at least plausible that this is the way humans construct plans.

Let us look more carefully at the idea that a GIA should construct plans by reasoning about them epistemically rather than by computing them using a separate module. How well this might work will depend upon the structure of the planning. It is an open question what form of planning should be employed by an GIA, but in the absence of the closed world assumption, the only *familiar* kind of planner that can be employed in a GIA is something like a classical goal regression planner. I don't want to claim that this is exactly the way planning should work in a GIA, but perhaps we can at least assume that the planner is a refinement planner that (1) constructs a basic plan, (2) looks for problems ("threats"), and (3) tries to fix the plan. It follows from our observations about epistemic cognition that the set of ⟨planning,solution⟩ pairs cannot be recursively enumerable, and planning cannot be performed by a terminating computational process (Pollock 1999). This is because the search for threats will not be terminating. Even if it could be performed by doing only first-order deductive reasoning, it would be at best recursively enumerable. More realistically, general defeasible reasoning will be used. Then the set of threats may not even be recursively enumerable. In either case, the set of ⟨planning,solution⟩ pairs will be Δ_2, just like the set of defeasible conclusions in general. The upshot is that, just like for other defeasible reasoning, planning cannot be done by "computing" the plans to be adopted. The computation could never terminate. If planning is Δ_2, a GIA must be prepared to adopt plans provisionally in the absence of knowledge of threats (that is, assume defeasibly that there are no threats), and then withdraw the plans and try to repair them when threats (defeaters for the planning) are found. So the logical structure of planning is indistinguishable from general defeasible reasoning.

The observation that planning is Δ_2 changes not only the logic of plan construction, but also the logic of the decision-theoretic reasoning involved in adopting plans. For a GIA to adopt a decision-theoretic plan, the requirement cannot be that it is optimal (i.e., at least as good as any other possible plan for achieving the same goals). This is because there will be no time at which it is not still possible that we will later find a better plan. Instead, a GIA must adopt "good" plans provisionally, being prepared to replace them by better plans if the latter are found. But it may never find optimal plans.[10]

My suggestion is that a GIA should reason epistemically about plans rather than computing them with some computational black box. To explore the feasibility of this approach, I implemented a classical goal regression planner within OSCAR that worked by reasoning about plans (1998a, 1998b). Various defeasible inference schemes were supplied, and then the planning proceeded by using them in reasoning. The planner was not particularly impressive (roughly equivalent to UCPOP (Penberthy et al 1992), on which it was modeled), but it worked, indicating that this approach is in principle feasible. I will give an example of the reasoning employed by this planner in section 3.3.

Classical goal-regression planning is now regarded as hopelessly inefficient. In part, that is an unfair charge, because it derives from comparing goal-regression planning to the current high-performance planners, and as we have seen, they make essential use of the closed world assumption and as such are not viable candidates for use in GIAs. Nevertheless, classical goal regression planners do not seem to be able to solve very difficult problems. There may be reason to hope, however, that decision-theoretic goal-regression planners can do better. This is because

[10] This is discussed at length in my 2006a, where a more general theory of the logical foundations for decision-theoretic planning is proposed.

knowledge of probabilities and utilities gives us important additional resources for use in directing search. Whether this will really work remains to be seen, but my current work is aimed at designing and implementing a defeasible goal-regression decision-theoretic planner that can be incorporated into OSCAR. The theory underlying this is presented in my (2006a).

The proposed implementation of the planner is based heavily on recent results (Pollock 2007, 2008) that I have obtained regarding reasoning defeasibly about probabilities. The main result is as follows. Given a list of variables $X_1,\ldots,X_n$ ranging over propositions, truth-functional compounds of these variables are formed by combining them with &, ∨, ~, etc. So, for example (X∨Y)&~Z is a truth-functional compound of *X*, *Y*, and *Z*. *Linear constraints* on the probabilities of truth-functional compounds either state the values of certain probabilities, e.g., stipulating that prob(X/Y) = *r*, or they relate probabilities using linear equations. For example, if we know that $X = Y \vee Z$, that generates the linear constraint

$$\text{prob}(X) = \text{prob}(Y) + \text{prob}(Z) - \text{prob}(X\&Z).$$

Then my main result is:

Expectable Probabilities Principle:

> Let $X_1,\ldots,X_n$ be a set of variables ranging over propositions, and consider a finite set *LC* of linear constraints on probabilities of truth-functional compounds of those variables. If *LC* is consistent with the probability calculus, then for any pair of truth-functional compounds *P*,*Q* of $X_1,\ldots,X_n$ there is a real number *r* between 0 and 1 such that, given the constraints *LC*, we can defeasibly expect that prob(*P*/*Q*) = *r*.

Furthermore, there is an algorithm for computing the value of *r*. This makes it possible for a GIA, engaging in decision-theoretic planning, to fill in the gaps in its probability knowledge defeasibly. This obviates the need for a GIA to have a complete probability distribution at its disposal. For example, suppose we have two seemingly unrelated diagnostic tests for a disease, and Bernard tests positive on both tests. We know that the probability of a person with his general symptoms having the disease is .6. We also know that the probability of such a person having the disease if he tests positive on the first test is .8, and the probability if he tests positive on the second test is .75. But what should we conclude about the probability of his having the disease if he tests positive on both tests? The probability calculus gives us no guidance here. It is consistent with the probability calculus for the probability of his having the disease given that he tests positive on both tests to be anything between 0 and 1. However, let us define:

$$\text{Y}(r,s\,|\,a) = \frac{rs(1-a)}{a(1-r-s)+rs}$$

I was able to prove:

The Y-Principle:

> ⌜prob(*A*/*B*&*U*) = *r* & prob(*A*/*C*&*U*) = *s* & prob(*A*/*U*) = *a*⌝ is a defeasible reason for
>
> ⌜prob(*A*/*B*&*C*&*U*) = Y(*r*,*s*|*a*)⌝.

Thus, for example, we have a defeasible reason for expecting that the probability of Bernard's having the disease is Y(.8,.75|.6) = .923. Similar reasoning allows us to fill in the gaps in our probability knowledge defeasibly, forming reasonable expectations about the values of unknown probabilities without having to have a complete probability distribution at our disposal.

2.3 Plan Construction and Practical Cognition

I have argued that in a GIA, plan construction should not be part of practical cognition, at least if we understand practical cognition to, by definition, be something separate from epistemic cognition. My claim is that plan construction should be performed by epistemic cognition. What remains for practical cognition is the task of posing planning problems, evaluating plans once they are constructed, deciding which to adopt, and directing plan execution. All of these processes involve a lot of interaction between practical cognition and epistemic cognition. Basically, practical

cognition passes queries to epistemic cognition and epistemic cognition sends answers back to practical cognition. In the next section, I will give a high level description of how this all works in the OSCAR architecture.

3. An Architecture for Interleaving Practical and Epistemic Cognition

3.1 Practical Cognition

Let us focus first on practical cognition. We can think of practical cognition as having three parts, as diagrammed in figure 9. First, the agent evaluates the world (as represented by its beliefs) and forms goals for changing various aspects of the world to make it more to its liking. Then the agent constructs plans for how to achieve its goals. Finally, it executes those plans, producing actions.

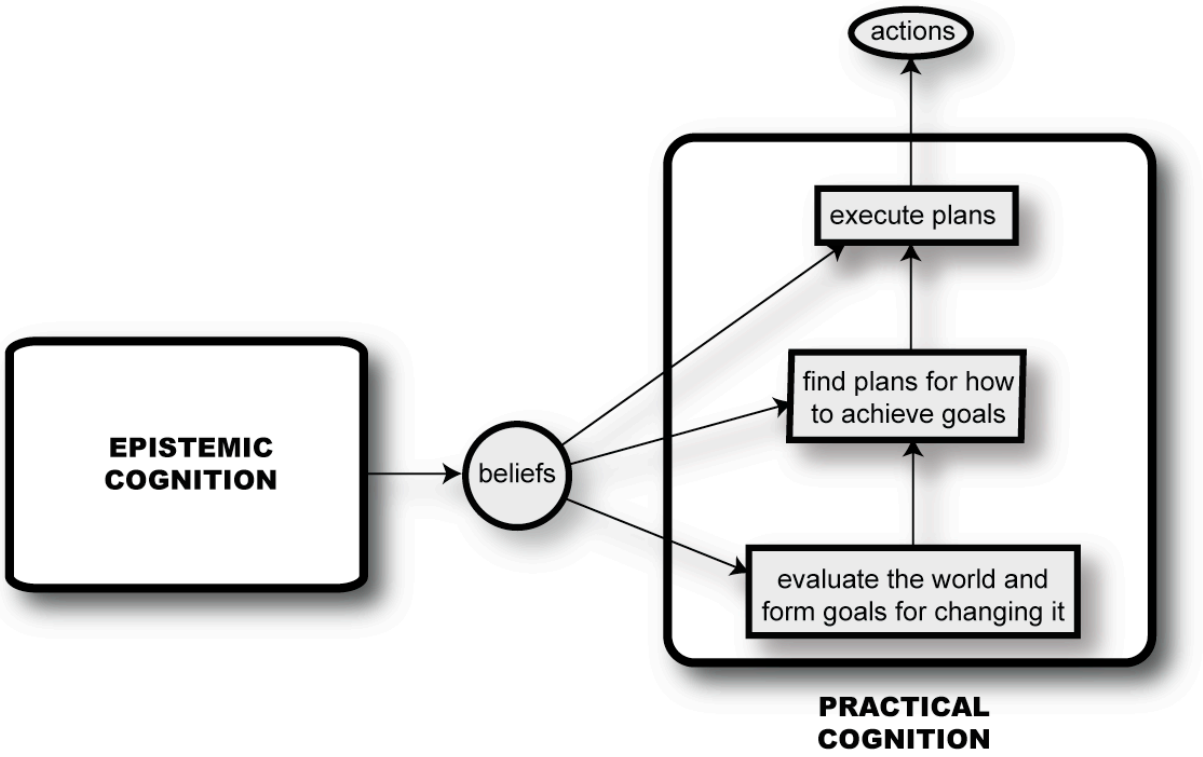

Figure 9. Practical cognition and epistemic cognition

The point of epistemic cognition is to provide the information required for practical cognition. This information is encoded in the form of beliefs, and the beliefs are used by all three modules of practical cognition, as diagrammed in figure 9. The agent evaluates the world, as represented by its beliefs, in order to form goals for changing various aspects of the world to make it more to its liking. So goals are adopted on the basis of the agent's beliefs about the way the world is. Plans are constructed on the basis of beliefs about what the consequences of actions are apt to be. And once a plan of action is adopted, its execution will typically turn upon the agent's beliefs about the world in at least two ways. First, the timing of actions may depend upon when various events occur, and the agent must have beliefs about that. Second, the agent will typically have to monitor the execution of the plan to make sure that various actions have the envisaged results.

3.2 Ultimate Epistemic Interests

We can imagine a cognitive architecture in which the agent just reasoned at random from its sensor input, and when it happened upon conclusions that were of use to practical cognition, they would be used appropriately. But such a system would be grossly inefficient, and it is clear that that

is not the way human cognition works. Any practical system of epistemic cognition must take account of what kind of information would be useful in the agent's practical endeavors, and focus its epistemic efforts accordingly. Practical cognition poses queries that are passed to epistemic cognition, and then epistemic cognition tries to answer them. For example, once the agent has adopted a particular goal, it tries to construct a plan for achieving it. In order to construct such a plan, a query should be passed to epistemic cognition concerning what plans are apt to achieve the goal. Similarly, when the agent adopts a plan, the timing of the execution will depend upon when various things happen in the world, so practical cognition should send a corresponding query to epistemic cognition. Epistemic cognition answers these queries by producing appropriate beliefs, which are then passed back to practical cognition. The queries posed by practical cognition comprise the set of *ultimate epistemic interests*. Thus we can expand the diagram of figure 9 to produce figure 10.

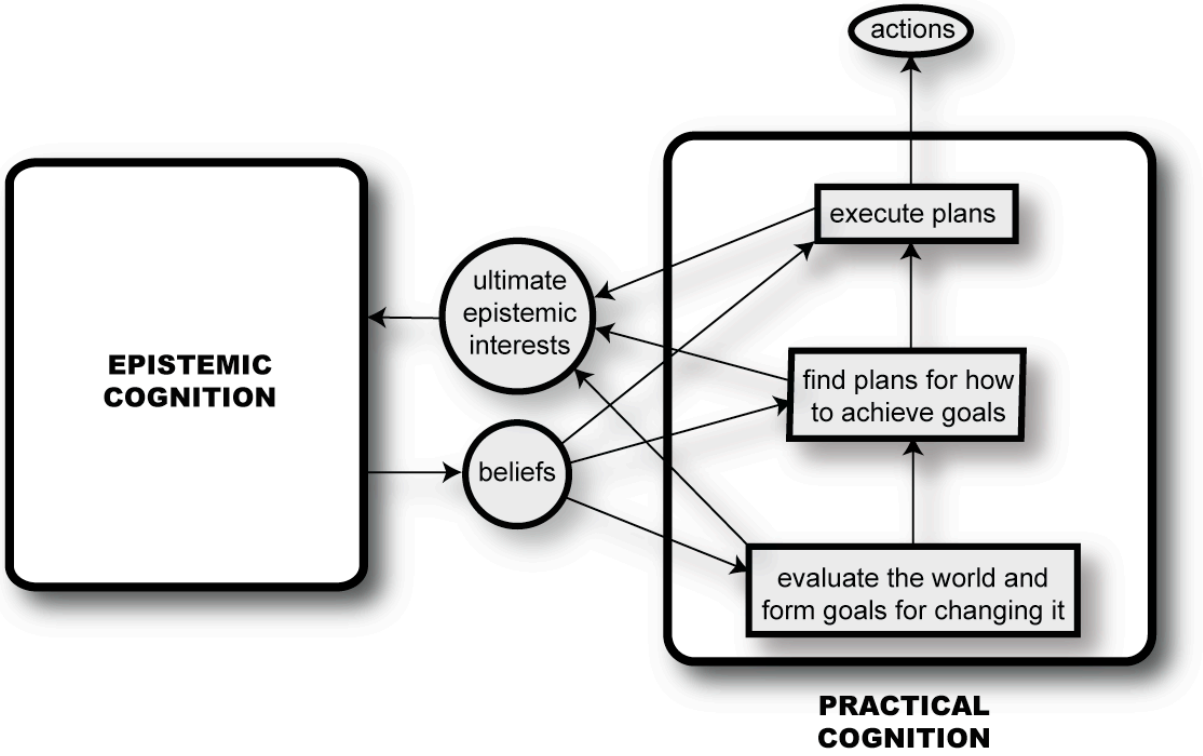

Figure 10. Query passing

3.3 Bidirectional Reasoning

Apparently the course of epistemic cognition must be driven by two different kinds of inputs. New information is input by perception, and queries are passed to it from practical cognition. The queries are *epistemic interests*. They represent things the agent wants to know. The OSCAR architecture accommodates these two kinds of inputs by implementing a system of bidirectional reasoning. Reasoning consists of building arguments, and one naturally thinks of that as a forward-directed enterprise. However, in OSCAR it is guided by backward reasoning. The inference schemes used by OSCAR are segregated into *forward-reasons* and *backward-reasons*. The forward-reasons are applied automatically whenever the agent has an argument for the premises of the reasons. For instance, *simplification* is a forward reason instructing the reasoner to infer P and Q from the conjunction $(P\&Q)$. So whenever the reason constructs an argument whose conclusion is $(P\&Q)$, the argument is immediately extended to draw the conclusions P and Q.

Logically, given arguments supporting P and Q separately, one can go on to construct an argument for $(P\&Q)$. This is the rule of *adjunction*. However, that is not something we would want a reasoner to do automatically. If adjunction were used as a forward reason, the result would be

combinatorially explosive. The reasoner would conjoin everything it knows, and then conjoin those conjunctions with each other, and so on *ad infinitum*. Obviously, human reasoners do not do that. We tend to use adjunction only when we already have an interest in the conjunction. OSCAR works similarly. Given an interest in (*P&Q*), OSCAR adopts interest in *P* and *Q*. Then if OSCAR constructs arguments for *P* and *Q*, it will "discharge" the interest in *P* and *Q* and infer (*P&Q*). Deriving an interest in *P* and in *Q* from an interest in (*P&Q*) is a matter of reasoning backward, and it proceeds by applying backward reasons to interests, starting with the ultimate epistemic interests. For example, here is a very simple instance of bidirectional reasoning in which the reasoner begins with the premises *A*, *B*, $(B \rightarrow C)$, and an interest in $(A \vee D)\&(C \vee E)$. The inference from $B,(B \rightarrow C)$ to *C* by *modus ponens* is a forward inference, and the inference from an interest in $(A \vee D)$ to an interest in *A* is a backward inference. OSCAR then reasons as follows:

```
# 1     A       given
# 2     B       given
# 3     (B → C)         given
                        # 1
                        interest: ((A v D) & (C v E))
                        This is of ultimate interest
                        # 2
                        interest: (A v D)
                        For interest 1 by adjunction
# 4     C
      Inferred by support-link #4 from { 3 , 2 } by modus-ponens
                        # 3
                        interest: A
                        For interest 4 by addition
                        This interest is discharged by node 1
# 5     (A v D)
      Inferred by support-link #6 from { 1 } by addition
      This node is inferred by discharging a link to interest #2
                        # 4
                        interest: (C v E)
                        For interest 1 by adjunction using node 5
                        # 5
                        interest: C
                        For interest 4 by addition
                        This interest is discharged by node 4
# 6     (C v E)
      Inferred by support-link #8 from { 4 } by addition
      This node is inferred by discharging a link to interest #5
# 7     ((A v D) & (C v E))
      Inferred by support-link #9 from { 5 , 6 } by adjunction
      This node is inferred by discharging a link to interest #1
```

By interleaving backward and forward reasoning, the reasoner is thus able to use its ultimate epistemic interests to help guide its epistemic cognition. The general architecture of OSCAR's bidirectional reasoning is diagrammed in figure 11. For abstract reasoning, the inputs can be premises. For a GIA, the inputs will be percepts, and OSCAR will be equipped with inference schemes for drawing conclusions from percepts. As new conclusions are drawn and new interests are adopted, they are put on the cognitive task queue. This is a prioritized queue of items waiting to be processed. When a conclusion is retrieved from the queue, it is sent to DISCHARGE-INTEREST to see whether it discharges any existing interests, and it is sent to REASON-FORWARD for use in further forward reasoning. When an interest is retrieved from the queue, it is send to DISCHARGE-INTEREST to see whether it is discharged by any conclusions already drawn, and it is sent to REASON-BACKWARD for use in further backward reasoning.

OSCAR's reasoning is determined by what inference schemes (backward and forward reasons) are supplied, and by the prioritization scheme used for the cognitive task queue. This system has been used to implement a natural deduction system for first-order logic, and it turns out that the bidirectional structure makes the deductive reasoner very efficient. OSCAR often outperforms the

more familiar resolution-refutation systems. A few years ago, Geoff Sutcliffe, one of the founders of the TPTP library ("Thousands of problems for theorem provers") issued a challenge to OSCAR. At CADE-13, a competition was held for clausal-form theorem provers. Otter was one of the most successful contestants. In addition, Otter is able to handle problems stated in natural form (as opposed to clausal form), and Otter is readily available for different platforms. Sutcliffe selected 212 problems from the TPTP library, and suggested that OSCAR and Otter run these problems on the same hardware. This "Shootout at the ATP corral" took place, with the result that OSCAR was on the average 40 times faster than Otter. In addition, OSCAR was able to find proofs for 16 problems on which Otter failed, and Otter was able to find proofs for only 3 problems on which OSCAR failed. Taking into account that Otter was written in C and OSCAR in LISP, the speed difference of the algorithms themselves could be as much as an order of magnitude greater.

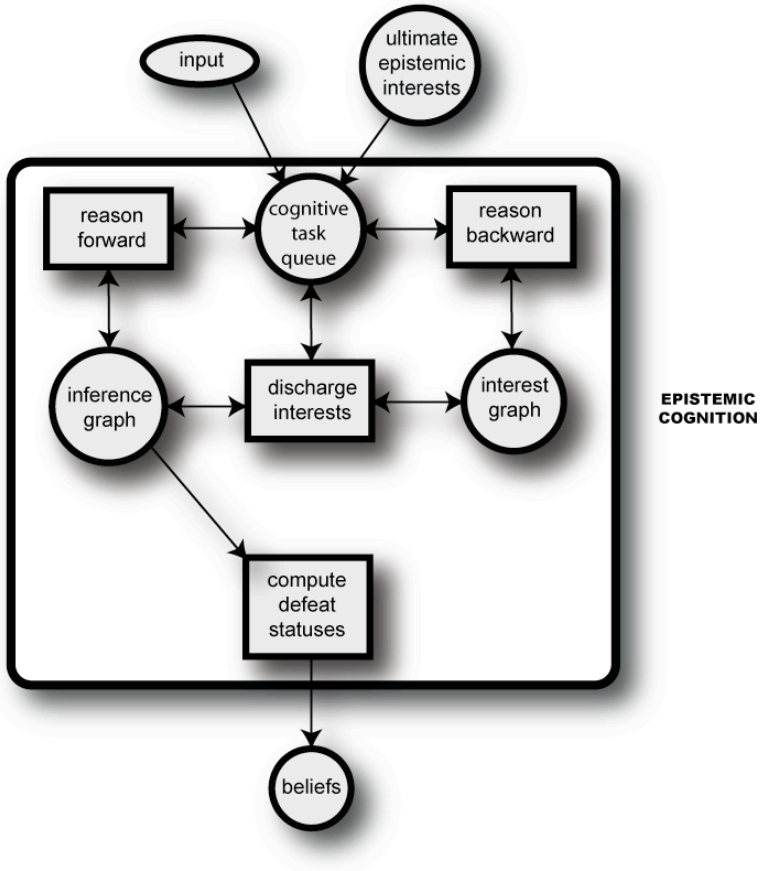

Figure 11. Bidirectional defeasible reasoning

In section 2.3 I talked briefly about implementing a classical planner in OSCAR by having OSCAR reason defeasibly about plans. It is OSCAR's bidirectional reasoning that makes this possible. For example, the fundamental operation of goal-regression is accomplished by the following reasoning-schema:

GOAL-REGRESSION

Given an interest in finding a plan for achieving *G*, adopt interest in finding planning-conditionals $(C \ \& \ A) \Rightarrow G$ having *G* as their consequent. Given such a conditional, adopt an interest in finding a plan for achieving *C*. If a plan *subplan* is proposed for achieving *C*, construct a plan by (1) adding a new step to the end of *subplan* where the new step prescribes the action *A*, (2) ordering the new step after all steps of *subplan*, and (3) adjusting the causal-links appropriately. Infer that the new plan will achieve *G*.

OSCAR contains a macro language for defining backward and forward reasons. GOAL-REGRESSION can be implemented in OSCAR as the following backward-reason:

```
(def-backward-reason GOAL-REGRESSION
  :conclusions "(plan-for plan goal goals nodes nodes-used links)"
  :condition (and (interest-variable plan) (not (mem goal goals)) (null nodes-used))
  :backwards-premises
     "((precondition & action) => goal)"
        (:condition (not (mem precondition goals)))
     "(define new-goals (cons goal goals))"
     "(plan-for subplan precondition new-goals nodes nodes-used links)"
     "(define plan (extend-plan action goal subplan links))"
        (:condition (not (null plan)))
  :variables precondition action goal plan subplan goals new-goals nodes nodes-used links)
```

The function (extend-plan *action goal subplan links*) constructs a new plan-node *node* having *action* as its action and *precondition* as its goal, and then constructs a new plan by adding this plan-node to *subplan*. The causal-links for the new node are constructed by taking the causal-links in the call-set of **finish** in *subplan*, and replacing their causal-link-targets by *node*. Then a new causal-link is added linking *node* to **finish** and having *goal* as its causal-link-goal.

To illustrate the reasoning with a very simple example, consider the following:

```
========================================================================
Goal-state:
   know-time

Given:
   at-library
   ((at-clock & read-clock) ⇒  know-time)
   ((at-library & go-to-clock) ⇒  at-clock)

========================================================================

 # 1
 at-library
 given
 This discharges interest 9
 # 2
 ((at-clock & read-clock) ⇒ know-time)
 given
 # 3
 ((at-library & go-to-clock) ⇒ at-clock)
 given
                # 4
                interest: (plan-for @y0 know-time)
                This is of ultimate interest
                # 6
                interest: (plan-for @y1 at-clock)
                For interest 4 by goal-regression
                This interest is discharged by node 5
                # 7
                interest: (plan-for @y2 at-library)
                For interest 6 by goal-regression
                This interest is discharged by node 4
                # 9
                interest: at-library
                For interest 7 by null-plan
                This interest is discharged by node 1
 # 4
 (plan-for <plan 1> at-library)
 Inferred by:
      support-link #4 from { 1 } by null-plan
 This node is inferred by discharging a link to interest #7

Plan #1 has been constructed:
```

```
  GOAL:  at-library
     established by:
      0 --> at-library

# 5
(plan-for <plan 2> at-clock)
Inferred by:
      support-link #5 from { 3 , 4 } by goal-regression
This node is inferred by discharging a link to interest #6

Plan #2 has been constructed:
  PLAN-STEPS:
   (2) go-to-clock
     causal-links:
      0 --at-library--> 2
  GOAL:  at-clock
     established by:
      2 --> at-clock

# 6
(plan-for <plan 3> know-time)
Inferred by:
      support-link #6 from { 2 , 5 } by goal-regression
This node is inferred by discharging a link to interest #4

Plan #3 has been constructed:
  PLAN-STEPS:
   (2) go-to-clock
     causal-links:
      0 --at-library--> 2
   (3) read-clock
     causal-links:
      2 --at-clock--> 3
      3 > 2
  GOAL:  know-time
     established by:
      3 --> know-time
========================================================================
```

More details and more interesting planning examples can be found in my (1998a).

3.4 Empirical Investigation

The architecture for epistemic cognition diagrammed in figure 11 would require the agent to answer queries posed by practical cognition just by thinking about them. It contains no other mechanism for answering questions. But many questions of practical interest cannot be answered just by thinking about what the cognizer already knows. To answer even so simple a question as "What time is it?", the agent may have to examine the world — at least look at a clock. More difficult questions may require looking things up in reference books, talking to other cognizers, searching one's closet, performing experiments in particle physics, etc. These are *empirical investigations*. What is characteristic of empirical investigations is that epistemic interests give rise to actions aimed at acquiring the information of interest. Actions are driven by practical cognition, so this involves a connection whereby epistemic cognition initiates further practical cognition. Practical cognition begins with the formation of goals, and then looks for plans that will achieve the goals. Accordingly, the mechanism whereby epistemic cognition can initiate practical cognition is by introducing "epistemic goals" — goals for the acquisition of information.

Empirical investigation introduces another interconnection between epistemic and practical cognition. Without thinking about it carefully, it is tempting to think of perception as spontaneous and undirected. However, there is a familiar distinction between looking and seeing. Seeing is passive, but looking involves putting yourself in an appropriate situation for seeing what you want to see. The same distinction applies to other forms of perception as well. We distinguish between sniffing and smelling, touching and feeling, listening and hearing, etc. In general, there is a distinction between "active" and "passive" perception, where active perception consists of putting oneself in an appropriate situation for passive perception. Some of the actions generated by practical cognition in the course of an empirical investigation will typically involve active perception. For instance, a GIA may look something up in a reference book by turning to the

appropriate page and then directing its eyes to the right place on the page.

Empirical investigation is accommodated by adding two links to the architecture of figures 9 and 10, generating figure 12. One link is from the generation of epistemic interests by epistemic cognition, to the formation of epistemic goals, and from there to practical cognition. The other link is from practical cognition to epistemic cognition via active perception. Adding these links has the consequence that we get loops from practical cognition to epistemic cognition, then back to practical cognition, and so on. Practical interests give rise to epistemic interests, which may in turn produce epistemic goals, which may initiate a search for a plan for how to find the desired information, which passes a new query back to epistemic cognition. That may give rise to new epistemic goals and further loops back through practical cognition.

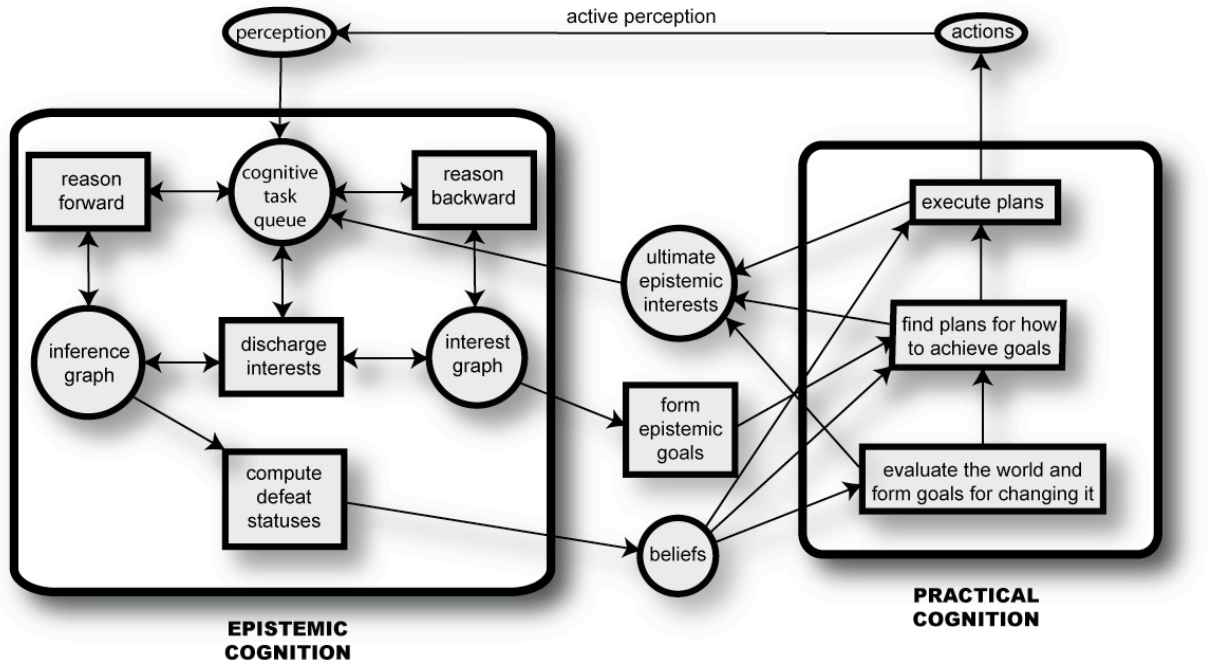

Figure 12. Empirical investigation

3.5 The Control Structure

In the OSCAR architecture, cognition is driven by (1) sensor input, (2) the production of new conclusions by reasoning, and (3) the production of new epistemic interests by either backward reasoning or queries passed from the various practical reasoning modules. There will typically be more cognitive tasks to be performed than can be performed at one time. So these four kinds of items are placed on a prioritized *cognitive task queue*, and popped off and processed in order of priority. OSCAR is then implemented as an infinite loop that repeatedly retrieves the top members of the cognitive task queue and passes it to the cognitive modules appropriate to its type, its type being any of (1) – (3). We can vary this control structure a bit by changing the ordering relation used for prioritizing the members of the cognitive task queue. It is not at all obvious what this ordering relation should be. We could employ simple orderings like *first-in-first-out* or *last-in-first-out*, but it is plausible that more sophisticated relations will improve performance. For first-order deductive reasoning, a great deal of experimentation has revealed that giving priority to simpler formulas over more complex formulas improves theorem-proving performance, although there is still considerable room for fine-tuning the complexity measure. For a full-fledged GIA, it seems plausible that the ordering should also take account of degrees of interest and degrees of justification. Perhaps more important interests (those with a higher associated degree) should be given priority. This is an issue that requires considerable experimentation, but that cannot be done very well until we have a suitable decision-theoretic planning system to be integrated into OSCAR.

3.6 Reflexive Cognition

Although epistemic cognition is initiated by practical cognition, it need not be *directed* by

practical cognition about how to answer questions. That would lead to an infinite regress, because practical cognition always requires beliefs about the world. If an agent could not acquire some such beliefs without first engaging in practical cognition, it could never get started. This indicates that there must be a *default control structure* governing epistemic cognition, and in particular, governing the way in which an agent tries to answer questions. It is this default control structure that was being discussed in the previous sections. However, in human beings it is also possible to override the default control structure. For example, consider taking an exam. A good strategy is often to do the easy problems first. This involves engaging in practical cognition about how to arrange our epistemic tasks. Our default control structure might, for example, take them in the order they are presented, but we can think about them and decide that it would be better to address them in a different order. Cognition about cognition is *reflexive cognition*.

When we redirect the course of our cognition by thinking about it and deciding what it would be best to do, we are engaging in practical cognition about how to cognize. One thing that humans can do is decide what to think about. In other words, they have the power (perhaps limited) to reorder their cognitive task queue. The ability to do this seems very important. It makes us more efficient problem solvers. For instance, when an experienced mathematician or scientist addresses a problem, he can draw on his experience for how best to proceed — what reasoning to try first. An equally bright college freshman may be unsuccessful at solving the problem primarily because he has no special knowledge about how to proceed and so must take things in the default order imposed on his cognitive task queue. This kind of reflexive cognition can be implemented in OSCAR by giving a GIA the power to reorder its cognitive task queue.

Consider a second kind of reflexive cognition exhibited by humans. We can have an argument for a conclusion, and see nothing wrong with it, but *without giving up the premises*, we may refrain from accepting the conclusion either because we have independent reason for thinking it is wrong, or because we are simply suspicious of it. This happens in mathematics all the time. In the course of trying to prove a theorem, I might "prove" something that just doesn't seem right. When that happens, I don't forge on. I set it aside and, if I care enough, try to figure out what went wrong. Something similar happens throughout science. Scientists get evidence for conclusions they find unbelievable, even though they cannot immediately see what is wrong with their data. They do not automatically accept the conclusions. They have the power to set them aside and examine them more carefully later. This can be implemented in OSCAR by giving a GIA some limited power to alter the computation of defeat statuses. If the GIA can refrain from marking a conclusion undefeated and hence adopting it as a belief, then it can refrain from accepting a conclusion even without knowing what is wrong with the arguments supporting it.

It is interesting to reflect upon just how much power an agent should have over its own cognition. In my (2008a) I surveyed some aspects of human reflexive cognition, and this may guide us in building reflexive cognition into OSCAR. There must be limits to how much power we give an agent over its own cognition. It is presumably undesirable for an agent to be able make itself believe something just because it wants to believe it. If it could do that it could just believe that its goals are satisfied, without doing anything to try to achieve them. Belief should be constrained by specifically epistemic cognition. On the other hand, it seems to be a good thing for an agent to be able to determine what it is going to think about next. But even here, there are dangers. Every scientist or scholar has experienced the following temptation. Suppose you have some cherished theory, and then it occurs to you that certain considerations might constitute a problem for your theory. It can be very tempting to just turn your attention elsewhere and not think about the potential problem. That is something you can do because you have the power to decide what to think about. But we would not want our GIAs to do this as a matter of course. It is not entirely obvious how to control this. Sometimes it is perfectly reasonable to think about other things. If you discover you are being stalked by a ravenous tiger, it would be foolish to focus your attention on potential problems for your scientific theory. How much power reflexive cognition should be given is a matter that will require considerable further thought and experimentation in the construction of GIAs.

We can accommodate the two kinds of reflexive cognition discussed above by adding one link from practical cognition to the cognitive task queue, and a second link to the module for computing defeat statuses. We may want to add more links later. This produces the final schematic form of the OSCAR architecture diagrammed in figure 13.

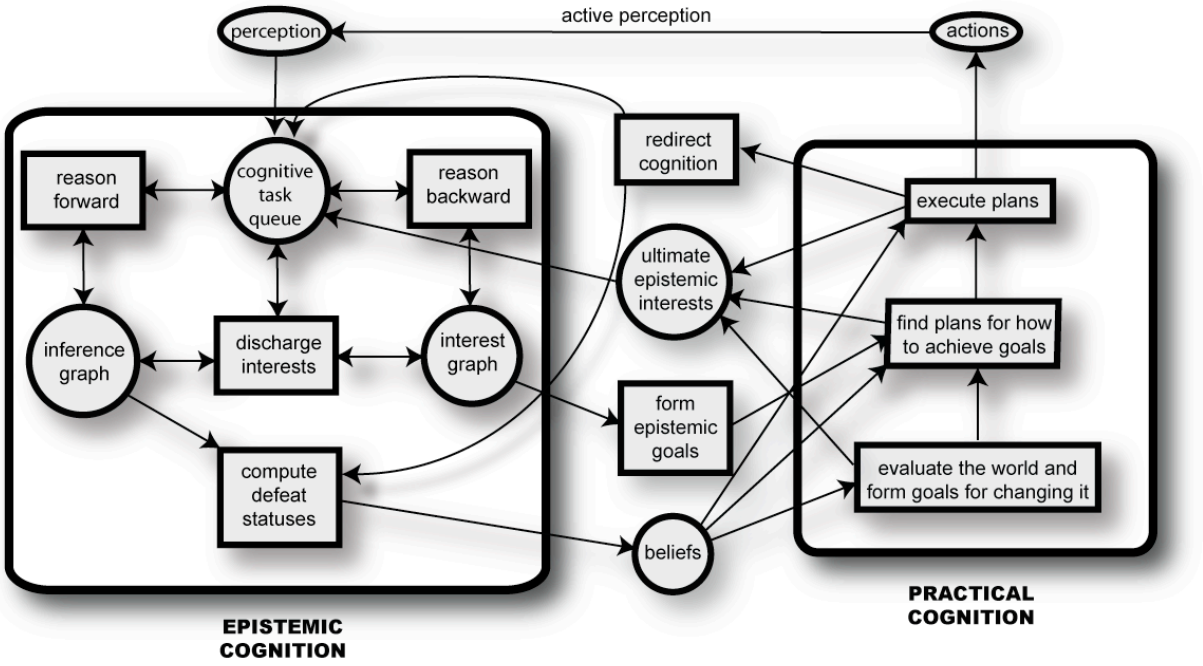

Figure 13. The OSCAR architecture

4. An Extended Example

Although the main function of an agent is to act, the OSCAR architecture shifts most of the burden of cognition onto the shoulders of epistemic cognition. The main roles of practical cognition are to formulate queries, which are passed to epistemic cognition, and then adopt and execute plans on the basis of the answers that epistemic cognition returns. So OSCAR gets its power from its facility at defeasible reasoning. How OSCAR reasons is determined by what inference schemes are supplied to OSCAR. A number of useful inference schemes have been implemented in OSCAR, and can be downloaded along with the OSCAR code. In addition, OSCAR has a built-in macro language that makes it simple for the user to define new inference schemes. I will illustrate this with an example drawn from my analysis of the frame problem (1998). This will not be a general discussion of the frame problem. Instead, I will just illustrate how OSCAR handles the Yale Shooting Problem.

Early discussions of the frame problem led AI researchers (Sandewall 1972; McDermott 1982; McCarthy 1986) to the realization that in reasoning about change, we normally assume defeasibly that things do not change unless they are forced to.[11] I illustrated the need for such a defeasible assumption in section 1.2, where I talked about comparing the time on my watch with that on a clock tower. To make the comparison, we have to assume that the time displayed on the clock tower does not change in the brief instant it takes me to glance away at my watch. To perform this reasoning, we need a defeasible inference scheme like the following:

TEMPORAL-PROJECTION

Believing *P-at-t* is a defeasible reason for the cognizer to believe *P-at-t+*Δt.

Here I am ignoring some important details discussed at length in my (1998). For example, we should take account of the fact that the strength of this inference should be a monotonic decreasing function of Δt. Using OSCAR's macro language, we can implement TEMPORAL-PROJECTION as follows:

[11] I proposed a similar principle in my (1974).

```
(def-backward-reason TEMPORAL-PROJECTION
   :conclusion  "(p at time)"
   :condition  (and (temporally-projectible p) (numberp time))
   :forwards-premises
      "(p at time0)"
   :backwards-premises
       "(time0 < time)"
   :variables p time0 time
   :defeasible? T
   :strength  (- (* 2 (expt *temporal-decay* (- time time0))) 1)
   :description
   "It is defeasibly reasonable to expect temporally projectible truths to remain unchanged.")
```

(I am leaving some important details unexplained. See my (1998) for further discussion of those details.)

For a while, some authors thought that the adoption of TEMPORAL-PROJECTION was all we needed to make our causal reasoning work properly, and hence it was supposed that this was a solution to the Frame Problem. Steve Hanks and Drew McDermott (1987) were the first to observe that even with defeasible principles of non-change, a reasoner will often be unable to determine what changes and what does not. They illustrated this with what has become known as "the Yale shooting problem". The general form of the problem is this. Suppose we have a causal law to the effect that if P is true at a time t and action A is performed at that time, then Q will be true shortly thereafter. Suppose we know that P is true now, and Q false. What should we conclude about the results of performing action A in the immediate future? Hanks and McDermott illustrated this by taking P to be "The gun is loaded, in working condition, and pointed at Jones", Q to be "Jones is dead", and A to be the action of pulling the trigger. We suppose (simplistically) that there is a causal law dictating that if the trigger is pulled on a loaded gun that is in working condition and pointed at someone, that person will shortly be dead. Under these circumstances, it seems clear that we should conclude that Jones will be dead shortly after the trigger is pulled.

The reasoning to the conclusion that Jones will die involves both temporal projection (in concluding that the gun will still be loaded when the trigger is pulled) and causal reasoning (in concluding that because the gun is loaded and trigger pulled, Jones will die). We can regard the causal reasoning as proceeding in accordance with the following defeasible inference scheme:

CAUSAL-IMPLICATION

If $(t+\varepsilon) \leq t^*$, then "(A when P is causally sufficient for Q after an interval ε) & A-at-t & P-at-t" is a defeasible reason for "Q-at-t^*.

This is implemented as follows:

```
(def-backward-reason CAUSAL-IMPLICATION
   :conclusion  "(Q at (op time*)"
   :forwards-premises
      "(A when P is causally sufficient for Q after an interval interval)"
        (:condition (every #'temporally-projectible (conjuncts Q)))
      "(A at time)"
        (:condition (time + interval) <= time*))
   :backwards-premises
      "(P at time)"
   :variables  A P Q interval time time* time** op
   :strength  (- (* 2 (expt *temporal-decay* (- time** time))) 1)
   :defeasible?  T)
```

The reasoning that leads to the conclusion that Jones will be dead can then be diagrammed as in Figure 14.

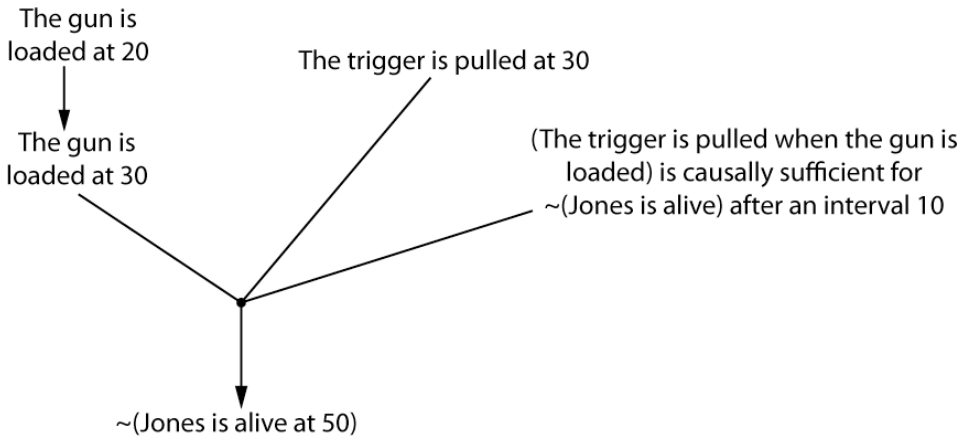

Figure 14. Causal reasoning

The difficulty is that because Jones is now alive, we have also a defeasible reason for believing that Jones will remain alive. This is diagrammed in figure 15. We know that one of these defeasible conclusions will be false, but we have no basis for choosing between them, so this becomes a case of collective defeat. That, however, is the intuitively wrong answer.

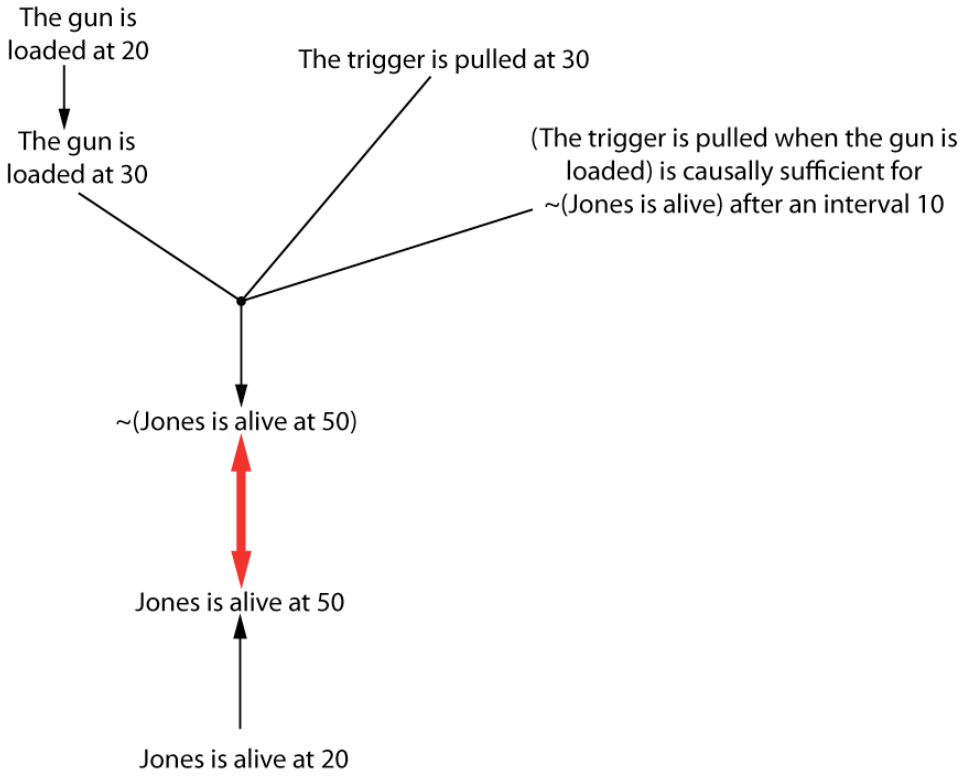

Figure 15. The Yale Shooting Problem

Notice that half of the collective defeat results from reasoning by temporal projection, and half results from reasoning causally from the conclusion of an *earlier* temporal projection. My proposal was that in causal reasoning, we view the world as unfolding in temporal order, so the earlier temporal projection takes precedence, and the causal reasoning from it takes precedence over the later temporal projection. This is made precise by adopting a causal undercutting defeater for temporal projection:

CAUSAL-UNDERCUTTER

Where $t_0 \leq t_1$ and $(t_1+\varepsilon) < t$, "A-at-t_1 & Q-at-t_1 & (A when Q is causally sufficient for ~P after an interval ε)" is a defeasible undercutting defeater for the inference from P-*at*-t_0 to P-at-t by TEMPORAL-PROJECTION.

This can be implemented as follows:

```
(def-backward-undercutter CAUSAL-UNDERCUTTER
   :defeatee temporal-projection
   :forwards-premises
      "(A when Q is causally sufficient for ~P after an interval interval)"
      "(A at time1)"
         (:condition (and (time0 <= time1) ((time1 + interval) < time)))
   :backwards-premises
      "(Q at time1)"
   :variables  A Q P time0 time time* time1 interval op
   :defeasible? T)
```

Equipped with these inference schemes, OSCAR solves the Yale Shooting Problem as follows:

```
================================================================================
This is the solved Yale Shooting Problem.  I know that the gun being fired while loaded will cause Jones to become
dead.  I know that the gun is initially loaded, and Jones is initially alive.  Later, the gun is fired.  Should I conclude that
Jones becomes dead?  (It is assumed that ~(Jones is alive) is temporally-projectible.)

Forwards-substantive-reasons:

Backwards-substantive-reasons:
        temporal-projection
        causal-undercutter
        causal-implication

Given:
        ((Jones is alive) at 20)
        (the gun is loaded at 20)
        (the gun is fired at 30)
        (the gun is fired when the gun is loaded is causally sufficient for ~(Jones is alive) after an interval 10)

Ultimate epistemic interests:
    (? ((Jones is alive) at 50))
================================================================================
THE FOLLOWING IS THE REASONING INVOLVED IN THE SOLUTION
Nodes marked DEFEATED have that status at the end of the reasoning.

 # 1
 ((Jones is alive) at 20)
 given
 # 2
 (the gun is loaded at 20)
 given
 # 3
 (the gun is fired at 30)
 given
 # 4
 (the gun is fired when the gun is loaded is causally sufficient for ~(Jones is alive) after an interval 10)
 given
```

1
interest: ((Jones is alive) at 50)
This is of ultimate interest
Of interest as a defeater for support-link 7 for node 7
2
interest: ~((Jones is alive) at 50)
This is of ultimate interest
Of interest as a defeater for support-link 5 for node 5

5
((Jones is alive) at 50) DEFEATED
Inferred by:
support-link #5 from { 1 } by indexical-temporal-projection defeaters: { 8 , 7 } DEFEATED
defeatees: { link 7 for node 7 }
This discharges interest 1
3
interest: (((Jones is alive) at 20) f (Jones is alive))
Of interest as a defeater for support-link 5 for node 5

==========
Justified belief in ((Jones is alive) at 50)
answers #<Query 1: (? (Jones is alive))>
==========
4
interest: (the gun is loaded at 30)
For interest 2 by indexical-causal-implication
For interest 3 by indexical-causal-undercutter
This interest is discharged by node 6

6
(the gun is loaded at 30)
Inferred by:
support-link #6 from { 2 } by temporal-projection
This discharges interest 4

7
~((Jones is alive) at 50)
Inferred by:
support-link #7 from { 4 , 3 , 6 } by indexical-causal-implication defeaters: { 5 }
defeatees: { link 5 for node 5 }
This node is inferred by discharging interest #2

8
(((Jones is alive) at 20) f (Jones is alive))
Inferred by:
support-link #8 from { 4 , 3 , 6 } by indexical-causal-undercutter
defeatees: { link 5 for node 5 }
This node is inferred by discharging interest #3

vvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
#<Node 5> has become defeated.
vvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
==========
Justified belief in ~(Jones is alive)
answers #<Query 1: (? (Jones is alive))>
==========
==========
Lowering the undefeated-degree-of-support of (Jones is alive)
retracts the previous answer to #<Query 1: (? (Jones is alive))>
==========

```
============================================================

================== ULTIMATE EPISTEMIC INTERESTS ==================
  Interest in (? (Jones is alive))
  is answered by node 7:  ~(Jones is alive)
------------------------------------------------------
```

This reasoning produces the inference-graph of figure 16, where the red arrows indicate defeat relations and "⊗" symbolizes undercutting defeat.

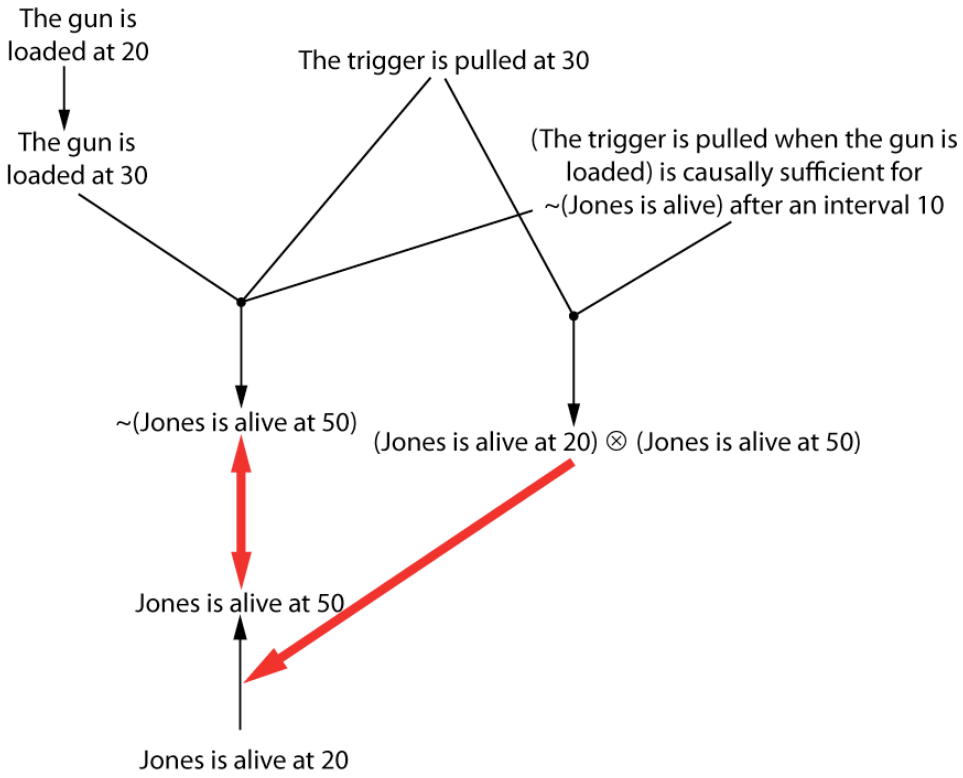

Figure 16. The Solved Yale Shooting Problem.

This, as far as I know, is the first *implemented* fully general solution to the Frame Problem.[12] It is noteworthy how simple it is to implement such principles in OSCAR, making OSCAR a potent tool for epistemological analysis, and an extremely useful platform for building a GIA.

5. Work to be Done

With the exception of reflexive cognition, the OSCAR architecture is fully implemented, and has been since the early 1990's. It is described in some detail in my (1995), and in more detail in *The OSCAR Manual*. But this is not to say that I have a fully implemented GIA. The architecture is the skeleton of an agent. To build an agent, we must supplement it with sensors, inference schemes, and extensors. I have written fairly extensively about some of the inference schemes required for

[12] Partially implemented solutions have been proposed by Loui (1987) and Kartha and Lifschitz (1995). Loui's solution was based upon THEORIST (Poole, et al 1987, Poole 1988). However, THEORIST produces a recursively enumerable set of consequences in any defeasible reasoning problem, and so is not capable of dealing with full first-order logic in which consistency is not decidable. The Kartha and Lifschitz implementation is based upon circumscription, and hence only works in the special case in which the circumscribed theory can be reduced to a first-order theory. In any cases of realistic complexity, the latter would seem to be impossible.

general epistemic cognition (Pollock 1974, 1986, 1990, 1995, 1998, 2008; Pollock and Cruz 1999, Pollock and Oved 2005), and my ongoing work on decision-theoretic planning (2006a) aims at supplying some more of the required inference schemes. Further inference schemes will be needed to direct reasoning about plan execution, and there are other gaps to be filled.

The OSCAR architecture is modular. The implemented system includes an efficient natural deduction theorem prover for first-order logic, and a system of defeasible reasoning that implements the defeat status algorithm of Pollock (1995). However, the first-order theorem prover can be changed by simply altering the inference schemes used by the system. This is just a matter of changing the contents of a file. As illustrated above, OSCAR includes a powerful macro language for defining new inference schemes. Different algorithms for computing defeat status can be adopted simply by changing the definition of the LISP function UPDATE-DEFEAT-STATUSES. Existing AI systems can be incorporated into the architecture without change by treating them as primitive inference schemes. In many cases it may be possible to re-implement them as sets of inference schemes, much like the way I built a planner modeled on UCPOP. Then, rather than operating as black boxes, they can be integrated into the more sophisticated control structure of the architecture.

Work on extensors must be relegated to robotics. It is beyond the scope of my expertise. I hope that I will eventually be able to collaborate with roboticists on this.

Work on the sensors is also beyond the scope of my expertise, but the development of sensors must be integrated into the general inference schemes of epistemic cognition. This is something that I have written about elsewhere (Pollock and Oved, 2005). Humans do not just take the output of video-camera-like eyes and plug that into defeasible inference schemes. In humans, reasoning begins with the visual image, but the visual image is the product of very complex computational preprocessing that is only partly understood at this time.[13] The human visual system takes two two-dimensional retinal bitmaps and converts them into a complex three-dimensional image already parsed into surfaces, objects, edges, corners, etc., located spatially with respect to one another. Furthermore, the image is the product of sensor output over an interval — not just instantaneous sensor output. This is illustrated by a number of visual phenomena. For example, when driving beside a fence with narrow slits in it, we can often see through it and have an image of what is on the other side of the fence, but when stationary we cannot. However, the instantaneous retinal bitmap is the same whether we are moving or stationary. The visual system integrates these retinal bitmaps over a small time interval to compute an image of what is on the other side of the fence. Somewhat similarly, we can literally see motion. This is important for many reasons. For example, motion parallax (the differential apparent movement of objects at different distances when we are moving with respect to them) plays an important role in computing the three dimensional aspects of the image. Because the image involves all this preprocessing, the epistemic cognition that begins with the image can be much simpler. What we are given is an image of objects and their properties — not just retinal bitmaps — and then we can reason from that sophisticated input, assuming defeasibly, for example, that objects tend to have the properties they are represented as having in the image. Building the reasoning system is probably much easier than building the visual system that is capable of doing this preprocessing.

6. Acknowledgment

This work was supported by NSF grant no. IIS-0412791.

References

Bondarenko, A., Dung., P.M., Kowalski, R.A., and Toni, F.
1997 "An abstract argumentation-theoretic approach to default reasoning." *Artificial Intelligence* **93**, 63-101.

Dung, P. M.
1995 "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and *n*-person games", *Artificial Intelligence* **77**, 321-357.

Hanks, Steve, and McDermott, Drew
1987 "Nonmonotonic logic and temporal projection", *Artificial Intelligence* **33**, 379-412.

[13] For a good survey of the current state of vision science, see Hoffman (1998).

Hoffman, Donald
1998 *Visual Intelligence*, Norton: New York.
Israel, David
1980 "What's wrong with non-monotonic logic?" *Proceedings of the First Annual National Conference on Artificial Intelligence*. 99–101.
Kartha, G. Neelakantan, and Lifschitz, Vladimir
1995 "A simple formalization of actions using circumscription". *Proceedings of IJCAI*,1970-1975.
Kushmerick, N., Hanks, S., and Weld, D.
1995 "An algorithm for probabilistic planning", *Artificial Intelligence* **76**, 239-286.
Kyburg, Henry, Jr.
1961 *Probability and the Logic of Rational Belief*. Middletown, Conn.: Wesleyan University Press.
Loui, Ron
1987 "Response to Hanks and McDermott: temporal evolution of beliefs and beliefs about temporal evolution", *Cognitive Science* **11**, 283-298.
Makinson, D. and Schlechta, K.
1991 "Floating conclusions and zombie paths: Two deep difficulties in the 'directly skeptical' approach to inheritance nets", *Artificial Intelligence* **48**, 199-209.
McCarthy, John
1980 "Circumscription — A form of non-monotonic reasoning". *Artificial Intelligence* **13**, 27-39, 171-172.
1986 "Applications of circumscription to formalizing common sense knowledge." *Artificial Intelligence* **26**, 89-116.
McDermott, Drew
1982 "A temporal logic for reasoning about processes and plans", *Cognitive Science* **6**, 101-155.
Pearl, Judea
1988 *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, CA: Morgan Kaufmann.
Penberthy, J. Scott, and Weld, Daniel
1992 "UCPOP: a sound, complete, partial order planner for ADL". *Proceedings 3rd International Conference on Principles of Knowledge Representation and Reasoning*, 103-114.
Pollock, John
1974 *Knowledge and Justification*, Princeton University Press.
1986 *Contemporary Theories of Knowledge*, Rowman and Littlefield.
1990 *Nomic Probability and the Foundations of Induction*, Oxford University Press.
1991 "Self-defeating arguments". *Minds and Machines* **1** (1991), 367-392.
1992 "How to reason defeasibly", *Artificial Intelligence*, **57**, 1-42.
1994 "Justification and defeat", *Artificial Intelligence* 67: 377-408.
1995 *Cognitive Carpentry*, MIT Press.
1998 "Perceiving and reasoning about a changing world", *Computational Intelligence*. **14**, 498-562.
1998a "Reasoning defeasibly about plans", OSCAR project technical report. This can be downloaded from http://www.u.arizona.edu/~pollock/.
1998b "Defeasible planning", in the proceedings of the AAAI Workshop, *Integrating Planning, Scheduling and Execution in Dynamic and Uncertain Environments*, Carnegie Mellon University, June 7, 1998, ed. Ralph Bermann and Alexander Kott. AAAI Technical Report WS-98-02.
1999 "The logical foundations of goal-regression planning in autonomous agents", *Artificial Intelligence* **106**, 267-335.
2002 "Defeasible reasoning with variable degrees of justification", *Artificial Intelligence* **133**, 233-282.
2005 "Plans and decisions", *Theory and Decision* **57**, 79-107.
2006 "Against optimality", *Computational Intelligence* **22** (2006), 1-25.
2006a *Thinking about Acting: Logical Foundations for Rational Decision Making*, New York: Oxford University Press.
2007 "Probable probabilities", OSCAR project technical report. This can be downloaded from http://www.u.arizona.edu/~pollock/.
2008 "Reasoning defeasibly about probabilities", in Michael O'Rourke and Joseph Cambell (eds.), *Knowledge and Skepticism*, Cambridge, MA: MIT Press.
2008a "Irrationality and cognition", in *Epistemology: New Philosophical Essays*, ed. Quentin Smith, New York: Oxford University Press.

2008b "Defeasible reasoning", in *Reasoning: Studies of Human Inference and its Foundations*, (ed) Jonathan Adler and Lance Rips, Cambridge: Cambridge University Press, 2006.
Pollock, John, and Joseph Cruz
1999 *Contemporary Theories of Knowledge*, 2nd edition, Lanham, Maryland: Rowman and Littlefield.
Pollock, John and Iris Oved
2005 "Vision, knowledge, and the mystery link", with Iris Oved. In *Phlosophical Perspectives* **19**, 309-351.
Poole, David
1988 "A logical framework for default reasoning", *Artificial Intelligence* **36**, 27-47.
Poole, David, Goebel, R. G., Aleliunas, R.
1987 "Theorist: a logical reasoning system for defaults and diagnosis", in N. Cercone and G. McCalla (eds.), *The Knowledge Frontier: Essays in the Representation of Knowledge*, Springer, New York.
Prakken, Henry and Gerard Vreeswijk
2001 "Logics for Defeasible Argumentation", in *Handbook of Philosophical Logic*, 2nd Edition, vol. 5, ed. D. Gabbay and F. Guenthner, Kluwer: Dordrecht.
Reiter, Raymond
1980 "A logic for default reasoning". *Artificial Intelligence* **13**, 81–132.
Sandewall, Erik
1972 "An approach to the frame problem and its implementation". In B. Metzer & D. Michie (eds.), *Machine Intelligence 7*. Edinburgh: Edinburgh University Press.
Vo, Quoc Bao, Norman Y. Foo, and Joe Thurbon
2005 "Semantics for a theory of defeasible reasoning", *Annals of Mathematics and Artificial Intelligence*, **44**(1-2), pp. 87-119.