OSCAR: An Architecture for Generally Intelligent Agents

John L. POLLOCK
Department of Philosophy
University of Arizona

Abstract OSCAR is a fully implemented architecture for a cognitive agent, based largely on the author's work in philosophy concerning epistemology and practical cognition. The seminal idea is that a generally intelligent agent must be able to function in an environment in which it is ignorant of most matters of fact. The architecture incorporates a general-purpose defeasible reasoner, built on top of an efficient natural deduction reasoner for first-order logic. It is based upon a detailed theory about how the various aspects of epistemic and practical cognition should interact, and many of the details are driven by theoretical results concerning defeasible reasoning. The architecture is easily extensible by changing the set of inference schemes supplied to the reasoner. Existing inference schemes handle many kinds of epistemic cognition, including reasoning from perceptual input, causal reasoning and the frame problem, and reasoning defeasibly about probabilities. Work is underway to implement a system of defeasible decision-theoretic planning.

Keywords: OSCAR, defeasible, epistemic cognition, planning, practical reasoning

1. Epistemic Cognition and Defeasible Reasoning

OSCAR is a fully implemented architecture for cognitive agents, based on the author's extensive work in philosophy concerning epistemology [1,2,3,4] and the theory of practical reasoning [5]. OSCAR is written in LISP, and can be downloaded from the OSCAR website at http://oscarhome.soc-sci.arizona.edu/ftp/OSCAR-web-page/oscar.html.

The basic observation that motivates the OSCAR architecture is that agents of human-level intelligence operating in an environment of real-world complexity (henceforth, GIAs — "generally intelligent agents") must be able to form beliefs and make decisions against a background of pervasive ignorance. It takes little reflection to realize that what human beings know about the world is *many* orders of magnitude smaller than what is true of the world. Our ignorance is of two sorts. First, we lack knowledge of almost all individual matters of fact. What do you know about individual grains of sand, or individual kittens, or apples hanging on all the apple trees scattered throughout the world? Suppose you want to adopt a kitten. Most 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 little about just a few of the very large number of kittens in the 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.

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. We must allow ourselves to form beliefs that are made probable by our evidence, but that are not logically guaranteed to be true. For instance, in our normal environment, objects generally have the colors they appear to have, so we can rely upon this statistical generalization in forming beliefs about the colors of objects that we see. Similarly, objects tend to retain certain kinds of 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. None of these inferences can deductively guarantee their conclusions. At best, they make the conclusions probable given the premises.

GIAs come equipped (by evolution or design) with inference schemes that are 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. Once the cognizer has some basic reliable inference schemes, it can use those 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 does discover 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* [9]. 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.

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" [6,7]. 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 assert that it is not really red. Because I believe that he is highly reliable, the statistical syllogism gives me a reason for concluding that it is not red. An agent that reasons in ways like this needs some cognitive mechanism for deciding which conclusions to adopt when conflicts arise.

Second, as noted above, the agent can learn that it is in circumstances in which one of its built-in inference schemes is not reliable, or less reliable than it is assumed by default to be. If the agent discovers empirically that an inference scheme is unreliable in some specific circumstances, then when it learns it is in those circumstances, it should not make the inference, or if it has already made the inference, it should withdraw the inference. This is one kind of "undercutting defeat" [6,7]. Undercutting defeaters attack an inference without attacking the conclusion itself. For intance, 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 it is illuminated by red lights, I should refrain from concluding that it is red, but it might still be red.

As I have just illustrated, we can discover new probabilistic information that provides us with an undercutting defeater for an inference. 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 has been 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 [1,8,9,6,2,3,4]. Often, the hardest task is to get the undercutting defeaters right. For example, in [2] 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.

Deductive reasoning is "monotonic". 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. Argument construction is still monotonic, 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 what the defeat status computation should be computing. We need a characterization of the set of conclusions that should be regarded as defeated and the set 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. Originally (1980 - 1993), OSCAR was based on a semantics [1] that was later proven equivalent [10] to Dung's [11] subsequently developed "admissible model semantics". More recently (since 1993), in response to difficulties involving self-defeating arguments [12], OSCAR has been based on a semantics [6,13] that was recently proved [14] equivalent to the subsequently developed "preferred model semantics" of Bondarenko, et al [15].

The natural temptation is to try to build an implemented defeasible reasoner on the model of familiar deductive reasoners. First-order deductive reasoners generate the members of a recursively enumerable set of deductive consequences of the given premises. By Church's theorem, the set of consequences is not decidable, but because it is r.e., 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. So 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 that is too weak for a GIA.)

A GIA cannot wait until all possibly relevant arguments have been constructed before we compute defeat statuses, because the process of argument construction is non-terminating. It must instead compute defeat statuses *provisionally*, on the basis of the arguments constructed so far, but 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. 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.

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.*

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 first-order representations. Second, this means that it must employ powerful systems of reasoning, including at least full first-order logic. Third, this means that argument construction will be non-terminating, and the existence of arguments supporting particular conclusions will be undecidable. But fourth, that forces the reasoning to be doubly defeasible. To the best of my knowledge, OSCAR is the only implemented defeasible reasoner that accommodates double defeasibility [16,6]. Accordingly, OSCAR is the only implemented defeasible reasoner that can make use of strong reasoning techniques like first-order logic.

These observations have the consequence that a GIA cannot be viewed as a problem solver. The life of a GIA does not consist of being presented with (or constructing) a sequence of 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. Problems can be set aside when defeasible solutions are found, but they can never be 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 epistemic cognition. 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?

In recent years, there have been impressive advances in planning theory, resulting in "high performance planners" that are capable of solving problems much harder than those classical goal regression planners could solve. But most of this work is inapplicable to building GIAs. This is for two reasons. First, high performance planners almost 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. So GIAs cannot be constructed on the basis of such planners.

Second, a GIA will also lack knowledge of what the outcomes of actions will definitely be. It will at best be able to predict outcomes probabilistically. 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 equally preposterous. Not only do we not know the values of most probabilities — a GIA could not encode their values even if they were known. Suppose a problem is described by logical compounds of a set of *n* simple propositions. Then to specify a complete probability distribution we must provide the values for 2^n logically independent probabilities. 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. 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 all the probabilities required for a complete probability distribution, a GIA would have to encode 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.

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 we will have to know 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.

Second, humans engage in hierarchical planning. That is, faced with a planning problem like that of how to attend a conference, they construct 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. Note further that, 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. What this illustrates is that planning and knowledge acquisition must be interleaved. In building a GIA, we cannot assume that it knows everything it needs to know before it begins planning.

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 difficulty, 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 [17]. This is because the search for threats will not be terminating. The result is that, just like other defeasible reasoning, planning cannot be done by "computing" the plans to be adopted. The computation could never terminate. 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 defeaters are found. So the logical structure of planning is indistinguishable from general defeasible reasoning.

The observation that planning is defeasible changes not only the logical structure of plan construction, but also the logical structure 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 [5]. But it may never find optimal plans.

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 [18,19]. 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 [20]), but it worked, indicating that this approach is in principle feasible.

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 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 [5]. The proposed implementation is based heavily on recent results that I have obtained regarding reasoning defeasibly about probabilities [8] (thus obviating the need for a GIA to have a complete probability distribution at its disposal).

3. An Architecture for Interleaving Practical and Epistemic Cognition

I have argued that in a GIA, 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 this section, I will give a high level description of how this all works in the OSCAR architecture. It is diagrammed in figure 1.

The point of epistemic cognition is to provide the information required for practical cognition. 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 which are passed to epistemic cognition, and then epistemic cognition tries to answer them. Different queries are passed to epistemic cognition depending upon what practical cognition has already occurred. 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 passed back to practical cognition. The queries posed by practical cognition comprise the set of *ultimate epistemic interests*.

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 agent can be equipped with inference-schemes that would allow it to answer a particular query if it knew something else. For example, given an interest in knowing (*P* & *Q*), the agent could satisfy that interest if it knew *P* and *Q* separately. So the agent *reasons backward* and adopts interest in *P* and *Q*. These are

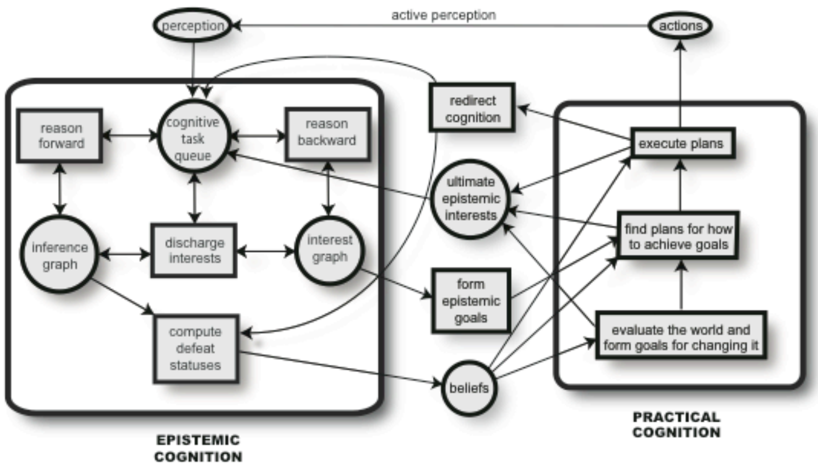

Figure 1. The OSCAR architecture.

derived epistemic interests. In this way the agent can reason backward from its ultimate epistemic interests and forwards from perceptual input until it finds itself in a situation in which forward reasoning has produced a belief that answers a query at which it has arrived by backward reasoning. This *discharges* the interest, enabling the agent to use the answer to that query in answering the query from which it was derived, and so on. By interleaving backward and forward reasoning, the agent is thus able to use its ultimate epistemic interests to help guide its epistemic cognition. OSCAR implements such bidirectional reasoning, and uses that to implement a natural deduction system for first-order logic [6]. It turns out that the bidirectional structure makes the deductive reasoner very efficient, and it 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.

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. There is a general 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, the agent 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. One link is from epistemic cognition, via the formation of epistemic goals, to practical cognition, and the other links 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.

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. 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 first, we are engaging in practical cognition about how to cognize. Clearly it is a good idea for a sophisticated cognitive agent to be able to modify the course of its own cognitive endeavors by engaging in practical cognition about how best to pursue them. An agent can learn that certain natural (default) strategies are unlikely to be effective in specific circumstances, and new strategies may be discovered that are more effective. The latter are often obtained by analogy from previous problem solving. It is desirable for a rational agent to use practical cognition in this manner to direct the course of either epistemic or practical cognition.

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 three 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).

4. Work to be Done

The OSCAR architecture is fully implemented, and has been since the early 1990's. It is described in some detail in my [6], and in more detail in *The OSCAR Manual* [21]. 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 general epistemic cognition, and my ongoing work on decision-theoretic planning aims at filling in 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 a natural deduction theorem prover for first-order logic, and a system of defeasible reasoning that implements the defeat status algorithm of [5]. However, the first-order theorem prover can be changed by simply altering the inference schemes supplied to the system, and different algorithms for computing defeat status can be used. OSCAR is equipped with inference schemes for reasoning defeasibly about perceptual input, temporal projection, causation, and the frame problem, and recent work has produced a system of inference schemes for reasoning defeasibly about probabilities. Existing AI systems can be incorporated into the architecture without change by treating them as Q&I modules. In many cases it may be possible to re-implement them as sets of inference schemes. Then, rather than operating as black boxes, they can be integrated into the more sophisticated control schemes of the architecture.

Work on extensors must be relegated to robotics. That 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 [3,4]. 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. 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. We can see through a slotted fence while driving beside it (but not while we are stationary), we can literally see motion, and motion parallax plays an important role in computing the three dimensional aspects of the image. Because the image involves all this preprocessing, the epistemic cognition that appeals to it can be 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.

5. Acknowledgment

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

References

[1] Pollock, John, *Contemporary Theories of Knowledge*, Rowman and Littlefield, 1986.

[2] Pollock, John, "Perceiving and reasoning about a changing world", *Computational Intelligence*. **14** (1998), 498-562.

[3] Pollock, John, and Joseph Cruz, *Contemporary Theories of Knowledge*, 2nd edition, Lanham, Maryland: Rowman and Littlefield, 1999.

[4] Pollock, John and Iris Oved, "Vision, knowledge, and the mystery link", with Iris Oved. In *Phlosophical Perspectives* **19** (2005), 309-351.

[5] Pollock, John, *Thinking about Acting: Logical Foundations for Rational Decision Making*, New York: Oxford University Press, 2006.

[6] Pollock, John, *Cognitive Carpentry*, MIT Press, 1995.

[7] Pollock, John, , "Defeasible reasoning", in *Reasoning: Studies of Human Inference and its Foundations*, (ed) Jonathan Adler and Lance Rips, Cambridge: Cambridge University Press, 2008.

[8] Pollock, John, "Reasoning defeasibly about probabilities", in Michael O'Rourke and Joseph Cambell (eds.), *Knowledge and Skepticism*, Cambridge, MA: MIT Press, 2008.

[9] Pollock, John, *Nomic Probability and the Foundations of Induction*, Oxford University Press, 1990.

[10] Prakken, Henry and Gerard Vreeswijk, "Logics for Defeasible Argumentation", to appear in *Handbook of Philosophical Logic*, 2nd Edition, vol. 5, ed. D. Gabbay and F. Guenthner, Kluwer: Dordrecht, 2001.

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

[12] Pollock, John, "Self-defeating arguments". *Minds and Machines* **1** (1991), 367-392.

[13] Pollock, John, "Justification and defeat", *Artificial Intelligence* **67** (1994), 377-408.

[14] Vo, Quoc Bao, Norman Y. Foo, and Joe Thurbon, "Semantics for a theory of defeasible reasoning", *Annals of Mathematics and Artificial Intelligence*, **44**(2005), pp. 87-119.

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

[16] Pollock, John, "How to reason defeasibly", *Artificial Intelligence*, **57** (1992), 1-42.

[17] Pollock, John, "The logical foundations of goal-regression planning in autonomous agents", *Artificial Intelligence* **106** (1999), 267-335.

[18] Pollock, John, "Reasoning defeasibly about plans", OSCAR project technical report. This can be downloaded from http://www.u.arizona.edu/~pollock/.

[19] Pollock, John, "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.

[20] Penberthy, J. Scott, and Weld, Daniel, "UCPOP: a sound, complete, partial order planner for ADL". *Proceedings 3rd International Conference on Principles of Knowledge Representation and Reasoning*, 1992, 103-114.

[21] Pollock, John, *The OSCAR Manual*, available online from http://www.u.arizona.edu/~pollock.