Rationality and Intelligence

Stuart Russell
Computer Science Division
University of California
Berkeley, CA 94720, USA

Abstract

The long-term goal of our field is the creation and understanding of intelligence. Productive research in AI, both practical and theoretical, benefits from a notion of intelligence that is precise enough to allow the cumulative development of robust systems and general results. The concept of *rational agency* has long been considered a leading candidate to fulfill this role. This paper outlines a gradual evolution in the formal conception of rationality that brings it closer to our informal conception of intelligence and simultaneously reduces the gap between theory and practice. Some directions for future research are indicated.

1 Artificial Intelligence

AI is a field whose ultimate goal has often been somewhat ill-defined and subject to dispute. Some researchers aim to emulate human cognition, others aim at the creation of intelligence without concern for human characteristics, and still others aim to create useful artifacts without concern for abstract notions of intelligence.

This variety is not *necessarily* a bad thing, since each approach uncovers new ideas and provides fertilization to the others. But one can argue that, since philosophers abhor a definitional vacuum, many of the damaging and ill-informed debates about the feasibility of AI have been about definitions of AI to which we as AI researchers do not subscribe.

My own motivation for studying AI is to create and understand intelligence as a general property of systems, rather than as a specific attribute of humans. I believe this to be an appropriate goal for the field as a whole, and it certainly includes the creation of useful artifacts—both as a spin-off and as a focus and driving force for technological development. The difficulty with this "creation of intelligence" view, however, is that it presupposes that we have some productive notion of what intelligence is. Cognitive scientists can say "Look, my model correctly predicted this experimental observation of human cognition," and artifact developers can say "Look, my system is saving lives/megabucks," but few of us are happy with papers saying "Look, my system is intelligent." This difficulty is compounded further by the need for theoretical scaffolding to allow us to design complex systems with confidence and to build on the results of others. "Intelligent" must be given a definition that can be related directly to the system's input, structure, and output. Such a definition must also be *general*. Otherwise, AI subsides into a smorgasbord of fields—intelligence as chess playing, intelligence as vehicle control, intelligence as medical diagnosis.

In this paper, I shall outline the development of such definitions over the history of AI and related disciplines. I shall examine each definition as a predicate P that can be applied, supposedly, to characterize systems that are intelligent. For each P, I shall discuss whether the statement "Look, my system is P" is interesting and at least sometimes true, and the sort of research and technological development to which the study of P-systems leads.

I shall begin with the idea that intelligence is strongly related to the capacity for successful behaviour—the so-called "agent-based" view of AI. The candidates for formal definitions of intelligence are as follows:

- P_1: *Perfect rationality*, or the capacity to generate maximally successful behaviour given the available information.

- P_2: *Calculative rationality*, or the in-principle capacity to compute the perfectly rational decision given the initially available information.

- P_3: *Metalevel rationality*, or the capacity to select the optimal combination of computation-sequence-plus-action, under the constraint that the action must be selected by the computation.

- P_4: *Bounded optimality*, or the capacity to generate maximally successful behaviour given the available information and computational resources.

All four definitions will be fleshed out in detail, and I will describe some results that have been obtained so far along these lines. Then I will describe ongoing and future work under the headings of calculative rationality and bounded optimality.

I shall be arguing that, of these candidates, bounded optimality comes closest to meeting the needs of AI research. There is always a danger, in this sort of claim, that its acceptance can lead to "premature mathematization," a condition characterized by increasingly technical results that have increasingly little to do with the original problem—in the case of AI, the problem of creating intelligence. Is research on bounded optimality a suitable stand-in for research on intelligence? I hope to show that P_4, bounded optimality, is more suitable than P_1 through P_3 because it is a real problem with real and desirable solutions, and also because it satisfies some essential intuitions about the nature of intelligence. Some important questions about intelligence can only be formulated and answered within the framework of bounded optimality or some relative thereof. Only time will tell, however, whether bounded optimality research, perhaps with additional refinements, can generate enough theoretical scaffolding to support significant practical progress in AI.

2 Agents

Until fairly recently, it was common to define AI as the computational study of "mental faculties" or "intelligent systems," catalogue various kinds, and leave it at that. This doesn't provide much guidance. Instead, one can define AI as the problem of designing systems that *do the right thing*. Now we just need a definition for "right."

This approach involves considering the intelligent entity as an *agent*, that is to say a system that senses its environment and acts upon it. Formally speaking, an agent is defined by the mapping from percept sequences to actions that the agent instantiates. Let $\mathbf{O}$ be the set of percepts that the agent can observe at any instant, and $\mathbf{A}$ be the set of possible actions the agent can carry out in the external world (including the action of doing nothing). Thus the *agent function* $f : \mathbf{O}^* \rightarrow \mathbf{A}$ defines how an agent behaves under all circumstances. What counts in the first instance is what the agent does, not necessarily what it thinks, or even whether it thinks at all. This initial refusal to consider further constraints on the internal workings of the agent (such as that it should reason logically, for example) helps in three ways: first, it allows us to view such "cognitive faculties" as planning and reasoning as occurring *in the service of* finding the right thing to do; second, it encompasses rather than excludes the position that systems can do the right thing without such cognitive faculties [Agre and Chapman, 1987; Brooks, 1989]; third, it allows more freedom to consider various specifications, boundaries, and interconnections of subsystems.

The agent-based view of AI has moved quickly from workshops on "situatedness" and "embeddedness" to mainstream textbooks [Russell and Norvig, 1995; Dean *et al.*, 1995] and buzzwords in Newsweek. *Rational* agents, loosely speaking, are agents whose actions make sense from the point of view of the information possessed by the agent and its goals (or the task for which it was designed). Rationality is a property of actions and does not specify—although it does constrain—the process by which the actions are selected. This was a point emphasized by Simon [1958], who coined the terms *substantive rationality* and *procedural rationality* to describe the difference between the question of *what* decision to make and the question of *how* to make it. That Rod Brooks's 1991 Computers and Thought lecture was titled "Intelligence without Reason" (see also [Brooks, 1991]) emphasizes the fact that reasoning is (perhaps) a *derived* property of agents that might, or might not, be a good implementation scheme to achieve rational behaviour. Justifying the cognitive structures that many AI researchers take for granted is not an easy problem.

One other consequence of the agent-based view of intelligence is that it opens AI up to competition from other fields that have traditionally looked on the embedded agent as a natural topic of study. Control theory is foremost among these, but evolutionary programming and indeed evolutionary biology itself also have ideas to contribute.[1] The prevalence of the agent view has also helped the field move towards solving real problems, avoiding what Brooks calls the "hallucination" problem that arises when the fragility of a subsystem is masked by having an intelligent human providing input to it and interpreting its outputs.

3 Perfect Rationality

Perfect rationality constrains an agent's actions to provide the maximum expectation of success given the information available. We can expand this notion as follows (see Figure 1). The fundamental inputs to the definition are the environment class $\mathbf{E}$ in which the agent is to operate and the performance measure U which evaluates the sequence of states through which the agent drives the actual environment. Let $V(f, \mathbf{E}, U)$ denote the expected value according to U obtained by an agent function f in environment class $\mathbf{E}$, where (for now) we will assume a probability distribution over elements of $\mathbf{E}$. Then a perfectly rational agent is defined by an agent function f_{opt} such that

$$f_{opt} = argmax_f V(f, \mathbf{E}, U)$$

[1] I view this as a very positive development. AI is a field defined by its problems, not its methods. Its principal insights—among them the learning, use, and compilation of explicit knowledge in the service of decision making—can certainly withstand the influx of new methods from other fields. This is especially true when other fields are simultaneously embracing the insights derived within AI.

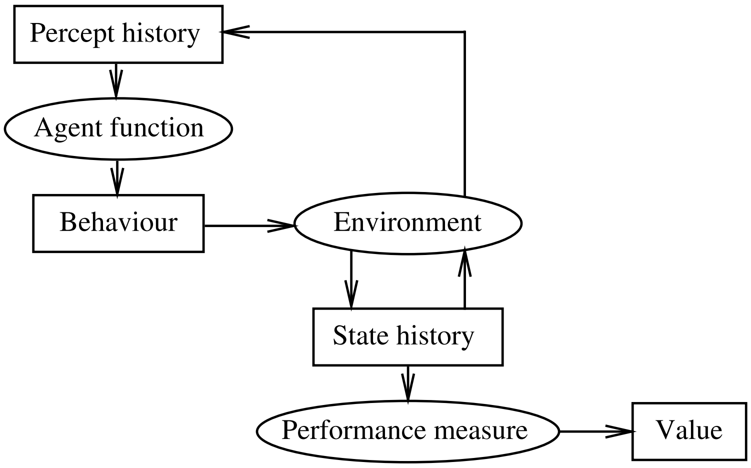

Figure 1: The agent receives percepts from the environment and generates a behaviour which in turn causes the environment to generate a state history. The performance measure evaluates the state history to arrive at the value of the agent.

This is just a fancy way of saying that the best agent does the best it can. The point is that perfectly rational behaviour is a well-defined function of $\mathbf{E}$ and U, which I will call the *task environment*. The problem of computing this function is addressed below.

The theoretical role of perfect rationality within AI is well-described by Newell's paper on the Knowledge Level [Newell, 1982]. Knowledge-level analysis of AI systems relies on an assumption of perfect rationality. It can be used to establish an upper bound on the performance of any possible system, by establishing what a perfectly rational agent would do given the same knowledge.

Although the knowledge that a perfectly rational agent has determines the actions that it will take given its goals, the question of where the knowledge comes from is not well understood. That is, we need to understand rational learning as well as rational action. In the logical view of rationality, learning has received almost no attention—indeed, Newell's analysis precludes learning at the knowledge level. In the decision-theoretic view, Bayesian updating provides a model for rational learning, but this pushes the question back to the prior [Carnap, 1950]. The question of rational priors, particularly for expressive representation languages, remains unsettled.

Another aspect of perfect rationality that is lacking is the development of a suitable body of techniques for the specification of utility functions. In economics, many results have been derived on the decomposition of overall utility into attributes that can be combined in various ways [Keeney and Raiffa, 1976], yet such methods have made few inroads into AI (but see [Wellman, 1985; Bacchus and Grove, 1995]). We also have little idea how to specify utility over time, and although the question has been raised often, we do not have a satisfactory understanding of the relationship between goals and utility.

The good thing about perfectly rational agents is that if you have one handy, you prefer it to any other agent. Furthermore, if you are an economist you can prove nice results about economies populated by them; and if you want to design distributed intelligent systems, assuming perfect rationality on the part of each agent makes the design of the interaction mechanisms much easier. The bad thing is that the theory of perfect rationality does not provide for the analysis of the internal design of the agent: one perfectly rational agent is as good as another. The *really* bad thing, as pointed out by Simon, is that perfectly rational agents do not exist. Physical mechanisms take time to process information and select actions, hence the behaviour of real agents cannot immediately reflect changes in the environment and will generally be suboptimal.

4 Calculative Rationality

Before discussing calculative rationality, it is necessary to introduce a distinction between the agent function and the *agent program*. In AI, an agent is implemented as a program, which I shall call l, running on a machine, which I shall call M. An agent program receives as input the current percept, but also has internal state that reflects, in some form, the previous percepts. It outputs actions when they have been selected. From the outside, the behaviour of the agent consists of the selected actions *interspersed with inaction* (or whatever default actions the machine generates).

Calculative rationality is displayed by programs that, *if executed infinitely fast*, would result in perfectly rational behaviour. Unlike perfect rationality, calculative rationality is a requirement that can be fulfilled by many real programs. Also unlike perfect rationality, calculative rationality is not necessarily a desirable property. For example, a calculatively rational chess program will choose the "right" move, but may take 10^{50} times too long to do so.

The pursuit of calculative rationality has nonetheless been the main activity of theoretically well-founded research in AI. In the early stages of the field, it was important to concentrate on "epistemological adequacy" before "heuristic adequacy"—that is, capability in principle rather than in practice.[2] Calculative rationality has been the mainstay of both the logical and the decision-theoretic traditions. In the logical tradition, the performance measure accepts behaviours that achieve the specified goal in all cases and rejects any others. Thus Newell [1982] defines rational actions as those that are guaranteed to achieve one of the agent's goals. Logical planning systems, such as theorem-provers using situation calculus, satisfy the conditions of calculative rationality

[2]Perhaps not coincidentally, this decision was taken before the question of computational intractability was properly understood in computer science.

under this definition. In the decision-theoretic tradition, the design of calculatively rational agents has largely gone on outside AI—for example, in stochastic optimal control theory [Kumar and Varaiya, 1986]. Representations have usually been very impoverished (state-based rather than sentential) and solvable problems have been either very small or very specialized. Within AI, the development of probabilistic networks or belief networks has opened up many new possibilities for agent design, providing in many cases an exponential reduction in representational complexity. Systems based on influence diagrams (probabilistic networks with action and value nodes added) satisfy the decision-theoretic version of calculative rationality.

In practice, neither the logical nor the decision-theoretic traditions can avoid the intractability of the decision problems posed by the requirement of calculative rationality. One response is to rule out sources of exponential complexity in the representations and reasoning tasks addressed, so that calculative and perfect rationality coincide—at least, if we ignore the little matter of polynomial-time computation. This position was expounded in two fascinating Computers and Thought lectures given by Hector Levesque in 1985 [Levesque, 1986; Levesque and Brachman, 1987] and by Henry Kautz in 1989. The accompanying research results on tractable sublanguages are perhaps best seen as indications of where complexity may be an issue rather than as a solution to the problem of complexity. The idea of restricting expressiveness was strongly opposed by Doyle and Patil [1991], who pointed out that it also restricts the applicability of the representation and inference services designed under such constraints.[3]

In the area of distributed AI, the system designer has control over that part of each agent's environment that involves negotiations with other agents. Thus, one possible way to control complexity is to constrain the negotiation problem so that optimal decisions can be made easily. For example, the Clarke Tax mechanism can be used to ensure that the best policy for each agent is simply to state its preferences truthfully [Ephrati and Rosenschein, 1991]. Of course, this approach does not necessarily result in optimal behaviour by the *ensemble* of agents; nor does it solve the problem of complexity in interacting with the rest of the environment.

The most common response to complexity has been to use various speedup techniques and approximations in the hope of getting reasonable behaviour. AI has developed a very powerful armoury of methods for reducing complexity, including the decomposition of state representations into sentential form; sparse representations of environment models (as in STRIPS operators); solution decomposition methods such as partial-order planning and abstraction; approximate, pa-

[3]Doyle and Patil propose instead the idea of "rational management of inference." Representation systems "should be designed to offer a broad mix of services varying in cost and quality" and should take into account "the costs and benefits [of computations] as perceived by the system's user." That is, they suggest a solution based on rational metareasoning, as discussed in Section 5.

rameterized representations of value functions for reinforcement learning; compilation (chunking, macro-operators, EBL etc.); and the application of metalevel control. Although some of these methods can retain guarantees of optimality and are effective for moderately large problems that are well structured, it is inevitable that intelligent agents will be unable to act rationally in all circumstances. This observation has been a commonplace since the very beginning of AI. Yet systems that select suboptimal actions fall outside calculative rationality *per se*, and we need a better theory to understand them.

5 Metalevel Rationality

Metalevel rationality, also called Type II rationality by I. J. Good [1971], is based on the idea of finding an optimal tradeoff between computational costs and decision quality. Although Good never made his concept of Type II rationality very precise—he defines it as "the maximization of expected utility *taking into account deliberation costs*—it is clear that the aim was to take advantage of some sort of *metalevel architecture* to implement this tradeoff. Metalevel architecture is a design philosophy for intelligent agents that divides the agent into two (or more) notional parts. The *object level* carries out computations concerned with the application domain—for example, projecting the results of physical actions, computing the utility of certain states, and so on. The *metalevel* is a second decision-making process whose application domain consists of the object-level computations themselves and the computational objects and states that they affect. Metareasoning has a long history in AI, going back at least to the early 1970s (see [Russell and Wefald, 1991a] for historical details). One can also view selective search methods and pruning strategies as embodying metalevel expertise concerning the desirability of pursuing particular object-level search operations.

The theory of *rational metareasoning* formalizes the intuition that the metalevel can "do the right thinking." The basic idea is that object-level computations are actions with costs (the passage of time) and benefits (improvements in decision quality). A rational metalevel selects computations according to their expected utility. Rational metareasoning has as a precursor the theory of *information value* [Howard, 1966]—the notion that one can calculate the decision-theoretic value of acquiring an additional piece of information by simulating the decision process that would be followed given each possible outcome of the information request, thereby estimating the expected improvement in decision quality averaged over those outcomes. The application to computational processes, by analogy to information-gathering, seems to have originated with Matheson [1968]. In AI, Horvitz [1987; 1989], Breese and Fehling [1990], and Russell and Wefald [1989; 1991a; 1991b] all showed how the idea of value of computation could solve the basic problems of real-time decision making.

The work done with Eric Wefald looked in particular at search algorithms, in which the object-level computations extend projections of the results of various courses of actions further into the future. For example, in chess programs, each object-level computation expands a leaf node of the game tree. The metalevel problem is then to select nodes for expansion and to terminate search at the appropriate point. The principal problem with metareasoning in such systems is that the local effects of the computations do not *directly* translate into improved decisions, because there is also a complex process of propagating the local effects at the leaf back to the root and the move choice. It turns out that a general formula for the value of computation can be found in terms of the "local effects" and the "propagation function," such that the formula can be instantiated for any particular object-level system (such as minimax propagation), compiled, and executed efficiently at runtime. This method was implemented for two-player games, two-player games with chance nodes, and single-agent search. In each case, the same general metareasoning scheme resulted in efficiency improvements of roughly an order of magnitude over traditional, highly-engineered algorithms.

Another general class of metareasoning problems arises with *anytime* [Dean and Boddy, 1988] or *flexible* [Horvitz, 1987] algorithms, which are algorithms designed to return results whose quality varies with the amount of time allocated to computation. The simplest type of metareasoning trades off the expected increase in decision quality for a single algorithm, as measured by a *performance profile*, against the cost of time [Simon, 1955]. A greedy termination condition is optimal if the second derivative of the performance profile is negative. More complex problems arise if one wishes to build complex real-time systems from anytime components. First, one has to ensure the *interruptibility* of the composed system—that is, to ensure that the system as a whole can respond robustly to immediate demands for output. The solution is to interleave the execution of all the components, allocating time to each component so that the total time for each complete iterative improvement cycle of the system doubles at each iteration. In this way, we can construct a complex system that can handle arbitrary and unexpected real-time demands exactly as if it knew the exact time available in advance, with just a small (≤ 4) constant factor penalty in speed [Russell and Zilberstein, 1991]. Second, one has to allocate the available computation optimally among the components to maximize the total output quality. Although this is NP-hard for the general case, it can be solved in time linear in program size when the call graph of the components is tree-structured [Zilberstein and Russell, 1996]. Although these results are derived in the relatively clean context of anytime algorithms with well-defined performance profiles, there is reason to expect that the general problem of robust real-time decision-making in complex systems can be handled in practice.

Over the last few years, an interesting debate has emerged concerning the nature of metaknowledge and metareasoning. TEIRESIAS [Davis, 1980] established the idea that explicit, domain-specific metaknowledge was an important aspect of expert system creation. Thus, metaknowledge is a sort of "extra" domain knowledge, over and above the object-level domain knowledge, that one has to add to an AI system to get it to work well. On the other hand, in the work on rational metareasoning described above, it is clear that *the metatheory describing the effects of computations is domain-independent* [Russell and Wefald, 1991a; Ginsberg and Geddis, 1991]. In principle, no additional domain knowledge is needed to assess the benefits of a computation. In practice, metareasoning from first principles can be very expensive. To avoid this, the results of metalevel analysis for particular domains can be compiled into domain-specific metaknowledge, or such knowledge can be learned directly from experience [Russell and Wefald, 1991a, Chapter 6], [Minton, 1996]. This view of emerging "computational expertise" leads to a fundamental insight into intelligence—namely, that there is an interesting sense in which *algorithms are not a necessary part of AI systems*. Instead, one can imagine a general process of rationally guided computation interacting with properties of the environment to produce more and more efficient decision making. To my mind, this way of thinking finesses one major puzzle of AI: if what is required for AI is incredibly devious and superbly efficient algorithms far surpassing the current best efforts of computer scientists, how did evolution (and how will machine learning) ever get there?

Significant open problems remain in the area of rational metareasoning. One obvious difficulty is that almost all systems to date have adopted a *myopic* strategy—a greedy, depth-one search at the metalevel. Obviously, the problem of optimal selection of computation *sequences* is at least as intractable as the underlying object-level problem. Nonetheless, sequences must be considered because in some cases the value of a computation may not be apparent as an improvement in decision quality until further computations have been done. This suggests that techniques from reinforcement learning could be effective, especially as the "reward function" for computation—that is, the improvement in decision quality—is easily available to the metalevel *post hoc*. Other possible areas for research include the creation of effective metalevel controllers for more complex systems such as abstraction hierarchy planners, hybrid architectures, and so on.

Although rational metareasoning seems to be a useful tool in coping with complexity, the concept of metalevel rationality as a formal framework for resource-bounded agents does not seem to hold water. The reason is that, since metareasoning is expensive, it cannot be carried out optimally. The history of object-level rationality has repeated itself at the metalevel: perfect rationality at the metalevel is unattainable and calculative rationality at the metalevel is useless. Therefore, a time/optimality tradeoff has to be made for metalevel computations, as for example with the myopic approximation mentioned above. Within the framework of metalevel rationality, however, there is no way to identify the appropriate tradeoff of time for metalevel decision quality. Any attempt to do so via a metametalevel simply results in a conceptual regress. Furthermore, it is entirely possible that in some environments, the most effective agent design will do no metareasoning at all, but will simply respond to circumstances. These considerations suggest that the right approach is to step outside the agent, as it were; to refrain from micromanaging the individual decisions made by the agent. This is the approach taken in bounded optimality.

6 Bounded Optimality

The difficulties with perfect rationality and metalevel rationality arise from the imposition of constraints on things (actions, computations) that the agent designer does not directly control. Specifying that *actions* or *computations* be rational is of no use if no real agents can fulfill the specification. The designer controls the *program*. In [Russell and Subramanian, 1995], the notion of *feasibility* for a given machine is introduced to describe the set of all agent functions that can be implemented by some agent program running on that machine. This is somewhat analogous to the idea of computability, but is much stricter because it relates the operation of a program on a formal machine model with finite speed to the actual temporal behaviour generated by the agent.

Given this view, one is led immediately to the idea that optimal feasible behaviour is an interesting notion, and to the idea of finding the program that generates it. Suppose we define $Agent(l, M)$ to be the agent function implemented by the program l running on machine M. Then the bounded optimal program l_{opt} is defined by

$$l_{opt} = argmax_{l \in \mathcal{L}_M} V(Agent(l, M), \mathbf{E}, U)$$

where $\mathcal{L}_M$ is the finite set of all programs that can be run on M. This is P_4, bounded optimality.

In AI, the idea of bounded optimality floated around among several discussion groups interested in the general topic of resource-bounded rationality in the late 1980s, particularly those at Rockwell (organized by Michael Fehling) and Stanford (organized by Michael Bratman). The term "bounded optimality" seems to have been originated by Eric Horvitz [1989], who defined it informally as "the optimization of computational utility given a set of assumptions about expected problems and constraints on resources."

Similar ideas have also surfaced recently in game theory, where there has been a shift from consideration of optimal decisions in games to a consideration of optimal decision-making programs. This leads to different results because it limits the ability of each agent to do unlimited simulation of the other, who is also doing unlimited simulation of the first, and so on. Even the requirement of computability makes a significant difference [Megiddo and Wigderson, 1986]. Bounds on the complexity of players have also become a topic of intense interest. Papadimitriou and Yannakakis [1994] have shown that a collaborative equilibrium exists for the iterated Prisoner's Dilemma game if each agent is a finite automaton with a number of states that is less than exponential in the number of rounds. This is essentially a bounded optimality result, where the bound is on space rather than speed of computation.

Philosophy has also seen a gradual evolution in the definition of rationality. There has been a shift from consideration of *act utilitarianism*—the rationality of individual acts—to *rule utilitarianism*, or the rationality of general policies for acting. The requirement that policies be feasible for limited agents was discussed extensively by Cherniak [1986] and Harman [1983]. A philosophical proposal generally consistent with the notion of bounded optimality can be found in the "Moral First Aid Manual" [Dennett, 1986]. Dennett explicitly discusses the idea of reaching an optimum within the space of feasible decision procedures, using as an example the Ph.D. admissions procedure of a philosophy department. He points out that the bounded optimal admissions procedure may be somewhat messy and may have no obvious hallmark of "optimality"—in fact, the admissions committee may continue to tinker with it since bounded optimal systems may have no way to recognize their own bounded optimality.

In work with Devika Subramanian, the general idea of bounded optimality has been placed in a formal setting so that one can begin to derive rigorous results on bounded optimal programs. This involves setting up completely specified relationships among agents, programs, machines, environments, and time. We found this to be a very valuable exercise in itself. For example, the "folk AI" notions of "real-time environments" and "deadlines" ended up with definitions rather different than those we had initially imagined. From this foundation, a very simple machine architecture was investigated in which the program consists of decision procedures of fixed execution time and decision quality. In a "stochastic deadline" environment, it turns out that the utility attained by running several procedures in sequence until interrupted is often higher than that attainable by any single decision procedure. That is, it is often better first to prepare a "quick and dirty" answer before embarking on more involved calculations in case the latter do not finish in time.

The interesting aspect of these results, beyond their value as a demonstration of nontrivial proofs of bounded optimality, is that they exhibit in a simple way what I believe to be a major feature of bounded optimal agents: the fact that the pressure towards optimality within a finite machine results in more complex program structures. Intuitively, efficient decision-making in a complex environment requires a software architecture that offers a wide variety of possible computational options, so that in most situations the agent has at least some computations available that provide a significant increase in decision quality.

One possible objection to the basic model of bounded optimality outlined above is that solutions are not *robust* with respect to small variations in the environment or the machine. This in turn would lead to difficulties in analysing complex system designs. Theoretical computer science faced the same problem in describing the running time of algorithms, because counting steps and describing instruction sets exactly gives the same kind of fragile results on optimal algorithms. The $O()$ notation was developed to deal with this and provides a much more robust way to describe complexity that is independent of machine speeds and implementation details. This robustness is also essential in allowing complexity results to develop cumulatively. In [Russell and Subramanian, 1995], the corresponding notion is asymptotic bounded optimality (ABO). As with classical complexity, we can define both average-case and worst-case ABO, where "case" here means the environment. For example, worst-case ABO is defined as follows:

Worst-case asymptotic bounded optimality

an agent program l is timewise (or spacewise) worst-case ABO in $\boldsymbol{E}$ on M iff

$$\exists k, n_0 \, \forall l', n \;\; n > n_0 \;\; \Rightarrow \;\; V^*(Agent(l, kM), \boldsymbol{E}, U, n) \geq V^*(Agent(l', M), \boldsymbol{E}, U, n)$$

where kM denotes a version of M speeded up by a factor k (or with k times more memory) and $V^*(f, \boldsymbol{E}, U, n)$ is the minimum value of $V(f, E, U)$ for all E in $\boldsymbol{E}$ of complexity n.

In English, this means that the program is basically along the right lines if it just needs a faster (larger) machine to have worst-case behaviour as good as that of any other program in all environments.

Another possible objection to the idea of bounded optimality is that it simply shifts the intractable computational burden of metalevel rationality from the agent's metalevel to the designer's object level. Surely, one might argue, the designer now has to solve offline all the metalevel optimization problems that were intractable when online. This argument is not without merit—indeed, it would be surprising if the agent design problem turns out to be easy. There is however, a significant difference between the two problems, in that the agent designer is presumably creating an agent for an entire class of environments, whereas the putative metalevel agent is working in a specific environment. That this can make the problem *easier* for the designer can be seen by considering the example of sorting algorithms. It may be very difficult indeed to sort a list of a trillion elements, but it is relatively easy to design an asymptotically optimal algorithm for sorting. In fact, the difficulties of the two tasks are unrelated. The unrelatedness would still hold for BO as well as ABO design, but the ABO definitions make it a good deal clearer.

It can be shown easily that worst-case ABO is a generalization of asymptotically optimal algorithms, simply by constructing a "classical environment" in which classical algorithms operate and in which the utility of the algorithm's behaviour is a decreasing positive function of runtime if the output is correct and zero otherwise. Agents in more general environments may need to trade off output quality for time, generate multiple outputs over time, and so on. As an illustration of how ABO is a useful abstraction, one can show that under certain restrictions one can construct *universal* ABO programs that are ABO for any time variation in the utility function, using the doubling construction from [Russell and Zilberstein, 1991]. Further directions for bounded optimality research are discussed below.

7 What Is To Be Done?

This section describes some of the research activities that will, I hope, help to turn bounded optimality into a creative tool for AI system design. First, however, I shall describe work on calculatively rational systems that needs to be done in order to enrich the space of agent programs.

7.1 Components for Calculative Rationality

As mentioned above, the correct design for a rational agent depends on the task environment—the "physical" environment and the performance measure on environment histories. It is possible to define some basic properties of task environments that, together with the complexity of the problem, lead to identifiable requirements on the corresponding rational agent designs [Russell and Norvig, 1995, Ch. 2]. The principal properties are whether the environment is *fully observable* or *partially observable*, whether it is *deterministic* or *stochastic*, whether it is *static* (i.e., does not change except when the agent acts) or *dynamic*, and whether it is *discrete* or *continuous*. Although crude, these distinctions serve to lay out an agenda for basic research in AI. By analysing and solving each subcase and producing calculatively rational mechanisms with the required properties, theoreticians can produce the AI equivalent of bricks, beams, and mortar with which AI architects can build the equivalent of cathedrals. Unfortunately, many of the basic components are currently missing. Others are so fragile and non-scalable as to be barely able to support their own weight. This presents many opportunities for research of far-reaching impact.

The logicist tradition of goal-based agent design, based on the creation and execution of guaranteed plans, is firmly anchored in fully observable, deterministic, static, and discrete task environments. (Furthermore, tasks are usually specified as logically defined goals rather than general utility functions.) This means that agents need keep no internal state and can even execute plans without the use of perception.

The theory of optimal action in stochastic, partially observable environments goes under the heading of *POMDPs* (Partially Observable Markov Decision Problems), a class of problems first addressed in the work of Sondik [1971] but almost completely unknown in AI until recently [Cassandra *et al.*, 1994]. Similarly, very little work of a fundamental nature has been done in AI on dynamic environments, which require real-time decision making, or on continuous environments, which have been largely the province of geometry-based robotics. Since most real-world applications are partially observable, nondeterministic, dynamic, and continuous, the lack of emphasis is somewhat surprising.

There are, however, several new bricks under construction. For example, dynamic probabilistic networks (DPNs) [Dean and Kanazawa, 1989] provide a mechanism to maintain beliefs about the current state of a dynamic, partially observable, nondeterministic environment, and to project forward the effects of actions. Also, the rapid improvement in the speed and accuracy of computer vision systems has made interfacing with continuous physical environments more practical. In particular, the application of Kalman filtering [Kalman, 1960], a widely used technique in control theory, allows robust and efficient tracking of moving objects; DPNs extend Kalman filtering to allow more general representations of world state. Reinforcement learning, together with inductive learning methods for continuous function representations such as neural networks, allow learning from delayed rewards in continuous, nondeterministic environments. Recently, Parr and Russell [1995], among others, have had some success in applying reinforcement learning to partially observable environments. Finally, learning methods for static and dynamic probabilistic networks with hidden variables (i.e., for partially observable environments) may make it possible to acquire the necessary environment models [Lauritzen, 1995; Russell *et al.*, 1995].

The Bayesian Automated Taxi (a.k.a. BATmobile) project [Forbes *et al.*, 1995] is an attempt to combine all these new bricks to solve an interesting application problem, namely driving a car on a freeway. Technically, this can be viewed as a POMDP because the environment contains relevant variables (such as whether or not the Volvo on your left is intending to change lanes to the right) that are not observable, and because the behaviour of other vehicles and the effects of one's own actions are not exactly predictable. In a POMDP, the optimal decision depends on the joint probability distribution over the entire set of state variables. It turns out that a combination of real-time vision algorithms, Kalman filtering, and dynamic probabilistic networks can maintain the required distribution when observing a stream of traffic on a freeway. The BATmobile currently uses a hand-coded decision tree to make decisions on this basis, and is a fairly safe driver (although probably far from optimal) on our simulator. We are currently experimenting with lookahead methods to make approximately rational decisions, as well as supervised learning and reinforcement learning methods.

As well as extending the scope of AI applications, new bricks for planning under uncertainty significantly increase the opportunity for metareasoning to make a difference. With logical planners, a plan either does or does not work; it has proved very difficult to find heuristics to measure the "goodness" of a logical plan that does not guarantee success, or to estimate the likelihood that an abstract logical plan will have a successful concrete instance. This means that it is very hard to identify plan elaboration steps that are likely to have high value. In contrast, planners designed to handle uncertainty and utility have built-in information about the likelihood of success and there is a continuum from hopeless to perfect plans. Getting metareasoning to work for such systems is a high priority. It is also important to apply those methods such as partial-order planning and abstraction that have been so effective in extending the reach of classical planners.

7.2 Directions for Bounded Optimality

Ongoing research on bounded optimality aims to extend the initial results of [Russell and Subramanian, 1995] to more interesting agent designs. In this section, I will sketch some design dimensions and the issues involved in establishing bounded optimality results.

The general scheme to be followed involves defining a virtual machine M that runs programs from a class $\mathcal{L}_M$. Typically, programs will have a "fixed part" that is shared across some subclass and a "variable part" that is specific to the individual program. Then comparisons are made between the best programs in different subclasses for the same machine. For example, suppose M is a machine capable of running any feedforward neural network. $\mathcal{L}_M$ consists of all such networks, and we might be interested in comparing the subclasses defined by different network topologies, while within each subclass individual programs differ in the weights on the links of the network. Thus, the boundary between machine and program depends to some extent on the range of comparisons that the designer wishes to consider.

At the most general level of analysis, the methodology is now quite straightforward: choose a machine, choose a program that runs on the machine, then dump the resulting agent into a class of environments **E**. The program with the best performance is bounded optimal for M in **E**. For example, M is an IBM PC with a C compiler; $\mathcal{L}_M$ consists of C programs up to a certain size; the environment consists of a population of human chess opponents; the performance measure is the chess rating achieved; the bounded optimal program is the one with the highest rating.

This rather blunt and unenlightening approach has no doubt occurred to many engaged in the construction of chess programs. As stated, the problem is ridiculously hard to solve and the solution, once found, would be very domain-specific. The problem is to define a research agenda for bounded optimality that provides a little more guidance and generality. This can be done by exploiting structure in the definition of the problem, in particular the orthogonality of time and content, and by using more sophisticated agent designs, particularly those that incorporate mechanisms for adaptation and optimization. In this way, we can prove bounded optimality results for more general classes of task environments.

7.2.1 Mechanisms for optimization

Modular design using a hierarchy of components is commonly seen as the only way to build reliable complex systems. The components fulfill certain behavioural specifications and interact in well-defined ways. To produce a composite bounded-optimal design, the optimization problem involves allocating execution time to components [Zilberstein and Russell, 1996] or arranging the order of execution of the components [Russell and Subramanian, 1995] to maximize overall performance. As illustrated earlier in the discussion of universal ABO algorithms, the techniques for optimizing temporal behaviour are largely orthogonal to the *content* of the system components, which can therefore be optimized separately. Consider, for example, a composite system that uses an anytime inference algorithm over a belief network as one of its components. If a learning algorithm improves the accuracy of the belief network, the performance profile of the inference component will improve, which will result in a reallocation of execution time that is guaranteed to improve overall system performance. Thus, techniques such as the doubling construction and the time allocation algorithm in [Zilberstein and Russell, 1996] can be seen as domain-independent tools for agent design. They enable bounded optimality results that do not depend on the specific temporal aspects of the environment class. As a simple example, we might prove that a certain chess program design is ABO for all time controls ranging from blitz to full tournament play.

The results obtained so far for optimal time allocation have assumed a static, offline optimization process with predictable component performance profiles and fixed connections among components. One can imagine far more subtle designs in which individual components must deal with unexpectedly slow or fast progress in processing and changing needs for information from other components. This might involve exchanging computational resources among components, establishing new interfaces, and so on. This is more reminiscent of a computational market, as envisaged by Wellman [1994], than of the classical subroutine hierarchies, and would offer a useful additional level of abstraction in system design.

7.2.2 Mechanisms for adaptation

In addition to combinatorial optimization of the structure and temporal behaviour of an agent, we can also use learning methods to improve the design:

- The *content* of an agent's knowledge base can of course be improved by inductive learning. In [Russell and Subramanian, 1995], it is shown that approximately bounded optimal designs can be guaranteed with high probability if each component is learned in such a way that its output quality is close to optimal among all components of a given execution time. Results from computational learning theory, particularly in the *agnostic learning* model [Kearns *et al.*, 1992], can provide learning methods with the required properties. The key additional step is to analyze the way in which slight imperfection in each component carries through to slight imperfection in the whole agent.

- *Reinforcement learning* can be used to learn value information such as utility functions. Recent results [Tsitsiklis and Van Roy, 1996] provide convergence guarantees for reinforcement learning with a fairly broad class of function approximators. One can use such learning methods for metalevel information, e.g., the value of computation. In [Russell and Wefald, 1991a, Chapter 6], this is shown to be an effective technique. Formal results on convergence to optimal control of search would be of great interest. Further work is needed, however, since current theorems assume a stationary distribution that generates the agent's experiences whereas an agent that is improving its search control will presumably be exploring different populations of experiences over time.

- *Compilation* methods such as explanation-based learning can be used to transform an agent's representations to allow faster decision making. Several agent architectures including SOAR [Laird *et al.*, 1986] use compilation to speed up all forms of problem solving. Some nontrivial results on convergence have been obtained by Tadepalli [1991], based on the observation that after a given amount of experience, novel problems for which no solution has been stored should be encountered only infrequently.

Presumably, an agent architecture can incorporate all these learning mechanisms. One of the issues to be faced by bounded optimality research is how to prove convergence results when several adaptation and optimization mechanisms are operating simultaneously. A "quasistatic" approach, in which one mechanism reaches convergence before the other method is allowed to take its next step, seems theoretically adequate but not very practical.

7.2.3 Offline and online mechanisms

One can distinguish between *offline* and *online* mechanisms for constructing bounded-optimal agents. An offline construction mechanism is not itself part of the agent and is not the subject of bounded optimality constraints. Let C be an offline mechanism designed for a class of environments $\mathbf{E}$. Then a typical theorem will say that C operates in a specific environment $E \in \mathbf{E}$ and returns an agent design that is ABO (say) for E—that is, an environment-specific agent.

In the online case, the mechanism C is considered part of the agent. Then a typical theorem will say that the agent is ABO for all $E \in \mathbf{E}$. If the performance measure used is indifferent to the transient cost of the adaptation or optimization mechanism, the two types of theorems are essentially the same. On the other hand, if the cost cannot be ignored—for example, if an agent that learns quickly is to be preferred to an agent that reaches the same level of performance but learns more slowly—then the analysis becomes more difficult. It may become necessary to define asymptotic equivalence for "experience efficiency" in order to obtain robust results, as is done in computational learning theory.

It is worth noting that one can easily prove the value of "lifelong learning" in the ABO framework. An agent that devotes a constant fraction of its computational resources to learning-while-doing cannot do worse, in the ABO sense, than an agent that ceases learning after some point. If some improvement is still possible, the lifelong learning agent will always be preferred.

7.2.4 Fixed and variable computation costs

Another dimension of design space emerges when one considers the computational cost of the "variable part" of the agent design. The design problem is simplified considerably when the cost is fixed. Consider again the task of metalevel reinforcement learning, and to make things concrete let the metalevel decision be made by a Q function mapping from computational state and action to value. Suppose further that the Q function is to be represented by a neural net. If the topology of the neural net is fixed, then all Q functions in the space have the same execution time. Consequently, the optimality criterion used by the standard Q-learning process coincides with bounded optimality, and the equilibrium reached will be a bounded-optimal configuration.[4] On the other hand, if the topology of the network is subject to alteration as the design space is explored, then the execution time of the different Q-functions varies. In this case, the standard Q-learning process will not necessarily converge to a bounded-optimal configuration. A different adaptation mechanism must

[4] A similar observation was made by Horvitz and Breese [1990] for cases where the object level is so restricted that the metalevel decision problem can be solved in constant time.

be found that takes into account the passage of time and its effect on utility.

Whatever the solution to this problem turns out to be, the important point is that the notion of bounded optimality helps to distinguish adaptation mechanisms that will result in good performance from those that will not. Adaptation mechanisms derived from calculative rationality will fail in the more realistic setting where an agent cannot afford to aim for perfection.

7.2.5 Fully variable architectures

The discussion so far has been limited to fairly sedate forms of agent architecture in which the scope for adaptation is circumscribed to particular functional aspects such as metalevel Q functions. However, an agent must in general deal with an environment that is far more complex than itself and that exhibits variation over time at all levels of granularity. Limits on the size of the agent's memory may imply that almost complete revision of the agent's mental structure is needed to achieve high performance. For example, one can imagine that a simple rule-based agent living through cycles of winter and summer may have to discard all of its summer rules as winter approaches, and then relearn them from scratch the following year. Such situations may engender a rethinking of some of our notions of agent architecture and optimality, and suggest a view of agent programs as dynamical systems with various amounts of compiled and uncompiled knowledge and internal processes of inductive learning, forgetting, and compilation.

7.2.6 Towards a grammar of AI systems

The approach that seems to be emerging for bounded optimality research is to divide up the space of agent designs into "architectural classes" such that in each class the structural variation is sufficiently limited. Then ABO results can be obtained either by analytical optimization within the class or by showing that an empirical adaptation process results in an approximately ABO design. Once this is done, it should be possible to compare architecture classes directly, perhaps to establish asymptotic dominance of one class over another. For example, it might be the case that the inclusion of an appropriate "macro-operator formation" or "greedy metareasoning" capability in a given architecture will result in an improvement in behaviour in the limit of very complex environments—that is, one cannot compensate for the exclusion of the capability by increasing the machine speed by a constant factor. A central tool in such work will be the use of "no-cost" results where, for example, the allocation of a constant fraction of computational resources to learning or metareasoning can do no harm to an agent's ABO prospects.

Getting all these architectural devices to work together smoothly is an important unsolved problem in AI and must be addressed before we can make progress on understanding bounded optimality within these more complex architectural classes. If the notion of "architectural device" can be made sufficiently concrete, then AI may eventually develop a *grammar* for agent designs, describing the devices and their interrelations. As the grammar develops, so should the accompanying ABO dominance results.

8 Summary

I have outlined some directions for formally grounded AI research based on bounded optimality as the desired property of AI systems. This perspective on AI seems to be a logical consequence of the inevitable philosophical "move" from optimization over actions or computations to optimization over programs. I have suggested that such an approach should allow synergy between theoretical and practical AI research of a kind not afforded by other formal frameworks. In the same vein, I believe it is a satisfactory formal counterpart of the informal goal of creating intelligence. In particular, it is entirely consistent with our intuitions about the need for complex structure in real intelligent agents, the importance of the resource limitations faced by relatively tiny minds in large worlds, and the operation of evolution as a design optimization process. One can also argue that bounded optimality research is likely to satisfy better the needs of those who wish to emulate human intelligence, because it takes into account the limitations on computational resources that are presumably responsible for most of the regrettable deviation from perfect rationality exhibited by humans.

Bounded optimality and its asymptotic cousin are, of course, nothing but formally defined properties that one may want systems to satisfy. It is too early to tell whether ABO will do the same kind of work for AI that asymptotic complexity has done for theoretical computer science. Creativity in design is still the prerogative of AI researchers. It may, however be possible to systematize the design process somewhat and to automate the process of adapting a system to its computational resources and the demands of the environment. The concept of bounded optimality provides a way to make sure the adaptation process is "correct."

My hope is that with these kinds of investigations, it will eventually be possible to develop the conceptual and mathematical tools to answer some basic questions about intelligence. For example, *why* do complex intelligent systems (appear to) have declarative knowledge structures over which they reason explicitly? This has been a fundamental assumption that distinguishes AI from other disciplines for agent design, yet the answer is still unknown. Indeed, Rod Brooks, Hubert Dreyfus, and others flatly deny the assumption. What is clear is that it will need *something like* a theory of bounded optimal agent design to answer this question.

Most of the agent design features that I have discussed here, including the use of declarative knowledge, have been conceived within the standard methodology of "first build calculatively rational agents and then speed them up." Yet one can legitimately doubt that this methodology will enable the AI community to discover all the design features needed for general intelligence. The reason is that no conceivable computer will ever be remotely close to approximating perfect rationality for even moderately complex environments. Perfect rationality is, if you like, a "Newtonian" definition for intelligent agents whereas the real world is a particle accelerator. It may well be the case that agents based on improvements to calculatively rational designs are *not even close* to achieving the level of performance that is potentially achievable given the underlying computational resources. For this reason, I believe it is imperative not to dismiss ideas for agent designs that do not seem at first glance to fit into the "classical" calculatively rational framework. Instead, one must attempt to understand the potential of the bounded optimal configurations within the corresponding architectural class, and to see if one can design the appropriate adaptation mechanisms that might help in realizing these configurations.

As mentioned in the previous section, there is also plenty of work to do in the area of making more general and more robust "bricks" from which to construct AI systems for more realistic environments, and such work will provide added scope for the achievement of bounded optimality. In a sense, under this conception AI research is the same now as it always should have been.

Acknowledgements

An earlier version of this paper appeared in the Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, published by IJCAII. That paper drew on previous work with Eric Wefald [Russell and Wefald, 1991a] and Devika Subramanian [Russell and Subramanian, 1995]. The latter work contains a more rigorous analysis of many of the concepts presented here. Thanks also to Michael Wellman, Michael Fehling, Michael Genesereth, Russ Greiner, Eric Horvitz, Henry Kautz, Daphne Koller, and Bart Selman for many stimulating discussions and useful suggestions on the topic of bounded rationality. The research was supported by NSF grants IRI-8903146, IRI-9211512 and IRI-9058427, by a UK SERC Visiting Fellowship.

References

[Agre and Chapman, 1987] Philip E. Agre and David Chapman. Pengi: an implementation of a theory of activity. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence (IJCAI-87)*, pages 268–272, Milan, Italy, August 1987. Morgan Kaufmann.

[Bacchus and Grove, 1995] Fahiem Bacchus and Adam Grove. Graphical models for preference and utility. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (UAI-95)*, pages 3–10, Montreal, 1995. Morgan Kaufmann.

[Breese and Fehling, 1990] J. S. Breese and M. R. Fehling. Control of problem-solving: Principles and architecture. In R. D. Shachter, T. Levitt, L. Kanal, and J. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*. Elsevier/North-Holland, Amsterdam, London, New York, 1990.

[Brooks, 1989] R. A. Brooks. Engineering approach to building complete, intelligent beings. *Proceedings of the SPIE—The International Society for Optical Engineering*, 1002:618–625, 1989.

[Brooks, 1991] R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47(1–3):139–159, 1991.

[Carnap, 1950] Rudolf Carnap. *Logical Foundations of Probability*. University of Chicago Press, Chicago, Illinois, 1950.

[Cassandra *et al.*, 1994] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, pages 1023–1028, Seattle, Washington, August 1994. AAAI Press.

[Cherniak, 1986] C. Cherniak. *Minimal Rationality*. MIT Press, Cambridge, Massachusetts, 1986.

[Davis, 1980] Randall Davis. Meta-rules: reasoning about control. *Artificial Intelligence*, 15(3):179–222, December 1980.

[Dean and Boddy, 1988] Thomas Dean and Mark Boddy. An analysis of time-dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 49–54, St. Paul, Minnesota, 21–26 August 1988. Morgan Kaufmann.

[Dean and Kanazawa, 1989] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

[Dean *et al.*, 1995] Thomas L. Dean, John Aloimonos, and James F. Allen. *Artificial Intelligence: Theory and Practice*. Benjamin/Cummings, Redwood City, California, 1995.

[Dennett, 1986] Daniel C. Dennett. The moral first aid manual. Tanner lectures on human values, University of Michigan, 1986.

[Doyle and Patil, 1991] J. Doyle and R. S. Patil. Two theses of knowledge representation: language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence*, 48(3):261–297, April 1991.

[Ephrati and Rosenschein, 1991] E. Ephrati and J. S. Rosenschein. The clarke tax as a consensus mechanism among automated agents. In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, volume 1, pages 173–178, Anaheim, California, July 1991. AAAI Press.

[Forbes *et al.*, 1995] Jeff Forbes, Tim Huang, Keiji Kanazawa, and Stuart Russell. The BATmobile: Towards a Bayesian automated taxi. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, August 1995. Morgan Kaufmann.

[Ginsberg and Geddis, 1991] M. L. Ginsberg and D. F. Geddis. Is there any need for domain-dependent control information? In *Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, volume 1, pages 452–457, Anaheim, California, July 1991. AAAI Press.

[Good, 1971] I. J. Good. Twenty-seven principles of rationality. In V. P. Godambe and D. A. Sprott, editors, *Foundations of Statistical Inference*, pages 108–141. Holt, Rinehart, Winston, Toronto, 1971.

[Harman, 1983] Gilbert H. Harman. *Change in View: Principles of Reasoning*. MIT Press, Cambridge, Massachusetts, 1983.

[Horvitz and Breese, 1990] E.J. Horvitz and J.S. Breese. Ideal partition of resources for metareasoning. Technical Report KSL-90-26, Knowledge Systems Laboratory, Stanford University, Stanford, California, February 1990.

[Horvitz, 1987] E. J. Horvitz. Problem-solving design: Reasoning about computational value, trade-offs, and resources. In *Proceedings of the Second Annual NASA Research Forum*, pages 26–43, Moffett Field, California, 1987. NASA Ames Research Center.

[Horvitz, 1989] E. J. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In Laveen N. Kanal, Tod S. Levitt, and John F. Lemmer, editors, *Uncertainty in Artificial Intelligence 3*, pages 301–324. Elsevier/North-Holland, Amsterdam, London, New York, 1989.

[Howard, 1966] Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, SSC-2:22–26, 1966.

[Kalman, 1960] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, pages 35–46, March 1960.

[Kearns *et al.*, 1992] M. Kearns, R. Schapire, , and L. Sellie. Toward efficient agnostic learning. In *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory (COLT-92)*, Pittsburgh, Pennsylvania, July 1992. ACM Press.

[Keeney and Raiffa, 1976] Ralph L. Keeney and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Wiley, New York, 1976.

[Kumar and Varaiya, 1986] P.R. Kumar and Pravin Varaiya. *Stochastic systems: Estimation, identification, and adaptive control*. Prentice-Hall, Englewood Cliffs, New Jersey, 1986.

[Laird *et al.*, 1986] J. E. Laird, P. S. Rosenbloom, and A. Newell. Chunking in Soar: the anatomy of a general learning mechanism. *Machine Learning*, 1:11–46, 1986.

[Lauritzen, 1995] S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201, 1995.

[Levesque and Brachman, 1987] H. J. Levesque and R. J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3(2):78–93, May 1987.

[Levesque, 1986] H. J. Levesque. Making believers out of computers. *Artificial Intelligence*, 30(1):81–108, October 1986.

[Matheson, 1968] J. E. Matheson. The economic value of analysis and computation. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(3):325–332, 1968.

[Megiddo and Wigderson, 1986] N. Megiddo and A. Wigderson. On play by means of computing machines. In Joseph Y. Halpern, editor, *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the 1986 Conference (TARK-86)*, pages 259–274, Monterey, California, March 19–22 1986. IBM and AAAI, Morgan Kaufmann.

[Minton, 1996] Steven Minton. Is there any need for domain-dependent control information? a reply. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, volume 1, pages 855–862, Portland, Oregon, August 1996. AAAI Press.

[Newell, 1982] Allen Newell. The knowledge level. *Artificial Intelligence*, 18(1):82–127, 1982.

[Papadimitriou and Yannakakis, 1994] C. H. Papadimitriou and M. Yannakakis. On complexity as bounded rationality. In *Symposium on Theory of Computation (STOC-94)*, 1994.

[Parr and Russell, 1995] Ronald Parr and Stuart Russell. Approximating optimal policies for partially observable stochastic domains. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, August 1995. Morgan Kaufmann.

[Russell and Norvig, 1995] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, New Jersey, 1995.

[Russell and Subramanian, 1995] Stuart J. Russell and Devika Subramanian. Provably bounded-optimal agents. *Journal of Artificial Intelligence Research*, 3, May 1995.

[Russell and Wefald, 1989] Stuart J. Russell and Eric H. Wefald. On optimal game-tree search using rational meta-reasoning. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, pages 334–340, Detroit, Michigan, August 1989. Morgan Kaufmann.

[Russell and Wefald, 1991a] Stuart J. Russell and Eric H. Wefald. *Do the Right Thing: Studies in Limited Rationality*. MIT Press, Cambridge, Massachusetts, 1991.

[Russell and Wefald, 1991b] Stuart J. Russell and Eric H. Wefald. Principles of metareasoning. *Artificial Intelligence*, 49(1–3):361–395, May 1991.

[Russell and Zilberstein, 1991] S. Russell and S. Zilberstein. Composing real-time systems. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, Sydney, August 1991. Morgan Kaufmann.

[Russell *et al.*, 1995] Stuart Russell, John Binder, Daphne Koller, and Keiji Kanazawa. Local learning in probabilistic networks with hidden variables. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1146–52, Montreal, Canada, August 1995. Morgan Kaufmann.

[Simon, 1955] Herbert A. Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, 69:99–118, 1955.

[Simon, 1958] Herbert A. Simon. Rational choice and the structure of the environment. In *Models of Bounded Rationality*, volume 2. MIT Press, Cambridge, Massachusetts, 1958.

[Sondik, 1971] E. J. Sondik. *The Optimal Control of Partially Observable Markov Decision Processes*. PhD thesis, Stanford University, Stanford, California, 1971.

[Tadepalli, 1991] Prasad Tadepalli. A formalization of explanation-based macro-operator learning. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 616–622, Sydney, August 1991. Morgan Kaufmann.

[Tsitsiklis and Van Roy, 1996] John N. Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. Technical Report LIDS-P-2322, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, 1996.

[Wellman, 1985] Michael P. Wellman. Reasoning about preference models. Technical Report MIT/LCS/TR-340, Laboratory for Computer Science, MIT, Cambridge, Massachusetts, 1985.

[Wellman, 1994] Michael P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1(1):1–23, 1994.

[Zilberstein and Russell, 1996] Shlomo Zilberstein and Stuart Russell. Optimal composition of real-time systems. *Artificial Intelligence*, 83, 1996.