AI Magazine Volume 28 Number 1 (2007) (© AAAI)

A Review of Recent Research in Metareasoning and Metalearning

Michael L. Anderson and Tim Oates

■ Recent years have seen a resurgence of interest in the use of metacognition in intelligent systems. This article is part of a small section meant to give interested researchers an overview and sampling of the kinds of work currently being pursued in this broad area. The current article offers a review of recent research in two main topic areas: the monitoring and control of reasoning (metareasoning) and the monitoring and control of learning (metalearning).

What Is Metacognition in Computation?

Rosie (the robot maid from the TV show *The Jetsons*) spends her days cooking, cleaning, ironing, and attending to the usual household tasks of late 21st century life. Because of a bug in one of her memory chips, however, she almost always forgets to buy dog food when she goes out. She has an adequate recovery plan for this: she simply feeds Astro some of the Jetson's dinner. But 21st century human food is expensive, so this strategy is wasteful. Realizing this, and recognizing that she has forgotten several times, Rosie adopts a special strategy to help her remember: she sticks the spare dog collar in her apron, where she will see it next time she is at the store. Rosie's special strategy is an instance of *metacognition:* Rosie monitored her performance in this cognitive task (remembering the grocery list), recognized a deficiency, and applied knowledge of her own operation (knowing she would see the collar in her apron) to take specific steps to address the deficiency.

Later that same day, Rosie consults her list of things to do. One of them, making the arrangements for the Jetsons' vacation, is going to involve a great deal of computationally intensive planning on her part, and Rosie's processor is old and slow. Knowing that doing this planning will take resources away from other tasks, and interfere with the other things she has to do that day, she schedules the computationally intensive task for the late evening, when her overall workload is less. This, too, is an instance of metacognition: knowing about her own capacities and scheduling tasks to make the best use of her limited resources. Essentially, metacognition is any such strategy that involves the monitoring, modeling, and control of cognition.

To put the matter more generally and formally, imagine two components, X and Y (where X and Y could be the same), related in such a way that state information flows from Y to X, and control information flows from X to Y. Component X is in a monitoring and control relationship with component Y, and when Y is a cognitive component, we call this relationship *metacognitive monitoring and control*. Put formally, then, the research question for the subject of metacognition in computation is: what are the sets $\{X, Y, S, E\}$—where Y is a

cognitive component of a computational system *S,* and *E* is its environment—such that having some *X* in such a relationship with *Y* provides benefits to the system (and what are these benefits)?

Recent years have seen a resurgence of interest in the topic of metacognition in computation. Metacognitive architectures and approaches have found application in diverse areas, from computer security (Caleiro, Vigan, and Basin, 2005; Welch and Stroud 2002; Kennedy 2003; Garfinkel and Rosenblum 2003) to cognitive modeling—including the modeling of decision making (Cohen and Thompson 2005, Oehlmann, 2003), commonsense psychology (Hobbs and Gordon 2005, Swanson and Gordon, 2005), the relations between emotion and judgement (Hudlicka 2005), and rhetorical force (Lundström, Hamfelt, and Nilsson 2005)—to computer gaming applications such as adversary generation (Zachary and Le Mentec 2000), to human-computer interaction (Kim 2005) and automated tutoring (Muldner and Conati 2005). Perhaps the most widely publicized metacognitive initiative has been IBM's autonomic computing project (Ganek and Corbi 2003), intended to use self-monitoring to improve the ability to build and manage both small- and large-scale computer systems. Other work on system manageability includes the use of model-based programming approaches to build long-lived, self-repairing autonomous systems (Williams et al. 2004); the development of techniques for leveraging explicit representations of system constraints to allow systems to self-manage incremental adaptation to changing task requirements (Murdock 2001); and the use of reflection to enhance the configurability of middleware objects (Costa 2001, Sullivan 2001). Finally, in the world of sponsored research, DARPA's recent Cognitive Information Processing Technology initiative foregrounds reflection (along with reaction and deliberation) as one of the three pillars required for flexible, robust AI systems.

This article, combined with two following articles, is meant to give interested researchers a sampling of the kinds of work currently being pursued in this area. The article by Stuart Shapiro and colleagues provides an overview of the SNePS knowledge representation and reasoning formalism, highlighting its metacognitive aspects, as well as its applications to such projects as the autonomous robot Cassie, while the article by Michael Cox discusses the requirements for producing a perpetual, self-aware cognitive agent that can continuously operate with independence. Among the central requirements for such an agent, Cox argues, is the capacity for introspection and self-improvement.

The current article rounds out the special section by offering a review of recent research in two main topic areas: the monitoring and control of reasoning (metareasoning) and the monitoring and control of learning (metalearning) (figure 1). The section on metareasoning focuses primarily on work published after 2000, partly as a (somewhat arbitrary) method for narrowing the task and partly because earlier work in this area has been ably summarized by Michael Cox (2005) and Stefania Costantini (2002).

Work on Metareasoning

Work on metareasoning falls (albeit not neatly) into two main areas: (1) scheduling and control of deliberation, and (2) generating and using higher-order knowledge about (or abstractions of) reasoning processes. The first area is of course closely related to the larger area of the control of computation, and it is to this that we turn first.

Control of Computation

Perhaps the most basic (but not for that reason the most easily solved!) question in the control of computation is the question of when to stop a given computational process. This problem can be especially difficult when the expected run time of the process is not known, as in the case of incomplete decision algorithms. Sandholm (2003) tackles this problem by developing a general decision-theoretic method for optimally terminating such algorithms. A key feature of the solution is a model of the algorithm's run-time distribution, the prior probability of a given answer (yes or no), and the value that different probability estimates for each answer would provide to the user at different times (based on the assumption that the user will act based on the probability estimates accorded each answer).

A slightly different aspect of this stopping problem is addessed by Jingfang Zheng and Michael Horsch (2005). As the authors correctly note, constraint optimization problems have two aspects that need to be balanced: *finding* the best solution, and *knowing* (that is, proving) one has found the best solution. To achieve the latter can take some time, often even longer than finding the solution itself. In fact, if there are resource bounds, a proof of optimality may not be possible under the constraints. Thus, the system (or the designer) faces a trade-off between (known) solution quality and time cost. Zheng

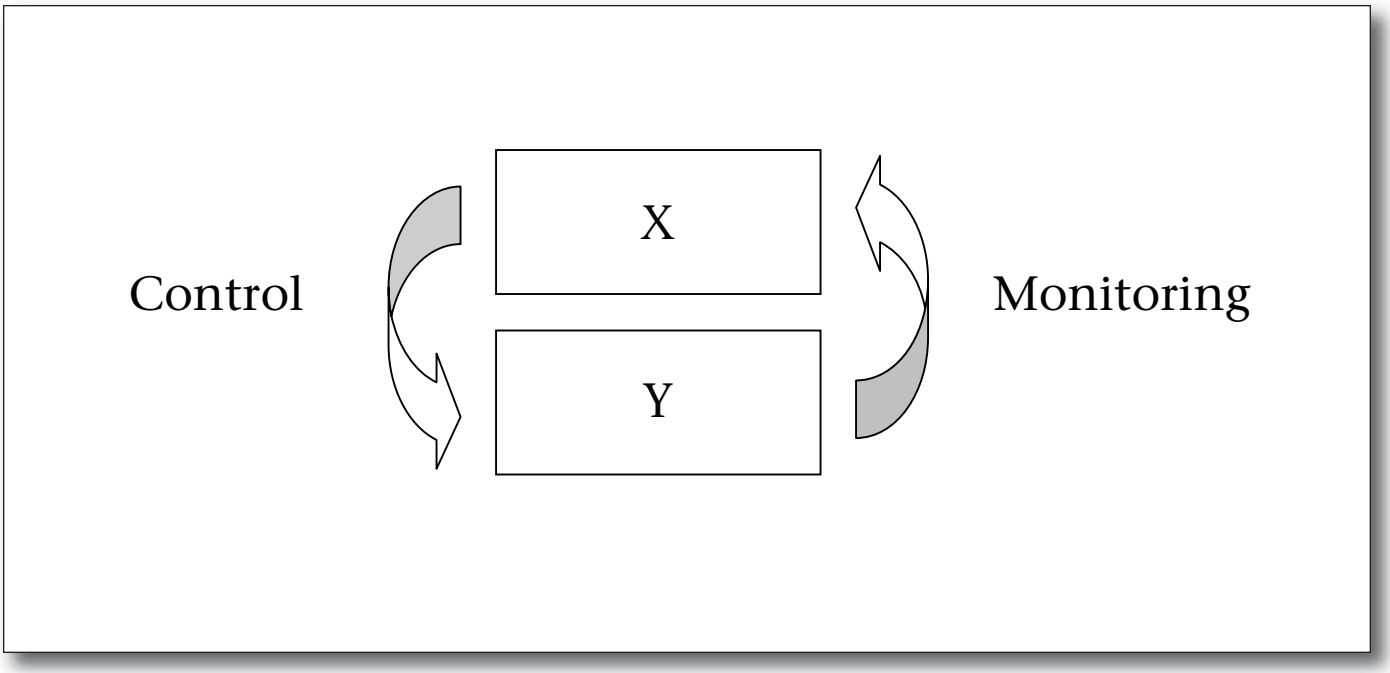

Figure 1. Schematic Representation of the Metacognitive Monitoring and Control Relation.

and Horsch develop and evaluate a decision-theoretic metareasoner that explicitly represents the costs and benefits of computation and uses this information to control a constraint-optimization problem solver. They demonstrate that by choosing actions with the estimated maximal expected utility, the metareasoner can find a stopping point with a good trade-off between the solution quality and time cost.

Perhaps the best-known work dealing with the control of deliberation under resource constraints is related to the theory of bounded optimality developed by Stuart Russell, Eric Horvitz, and others (see Russell [2002] for a recent overview). Bounded optimality is an approach to the design of agents with "rational" (that is, maximally beneficial) deliberation policies, given assumptions about resource constraints and the sorts of problems the agents will face. More precisely, bounded optimality is a formally defined characteristic of the agent that the designer attempts to achieve. The approach explicitly shifts the metalevel control task faced by the agent—deciding when to deliberate and when to stop—to the system designer, who implements a set of deliberation policies that are provably optimal over some range of assumed constraints. This can be a good strategy when one has reason to believe the designer to have better knowledge of the task characteristics and system constraints than does the agent itself, but in cases where the environment or agent can change in unpredicted or even unpredictable ways, there is some reason to return this responsibility to the agent operating in real time.

One project along these latter lines is presented by Martijn Schut and Michael Wooldridge (2001), who integrate a belief desire intention (BDI) architecture with the deliberation scheduling approaches of Stuart Russell and Eric Wefald (1991) such that the agent can dynamically choose a policy for intention reconsideration based on local factors. Schut and Wooldridge demonstrate improved performance in dynamic and open environments over agents with deliberation policies fixed at design time.

Similarly, Robert Goldman, David Musliner, and Kurt Krebsbach (2003) address the problem of deliberation scheduling in real-time intelligent control situations with hard deadlines, where time spent scheduling deliberation must be carefully controlled because of strict time constraints. A key requirement of the scenarios they discuss is that the deliberation policies must be flexible and under the control of the agent, because unpredictable variations in the time needed for each phase of a complex operation (for example, an unmanned aircraft flying a surveillance mission) can change the time available for other aspects of the operation, including the time available for various computations and adaptations. Their paper

addresses one aspect of this problem by developing simplified Markov decision process models that minimize the time cost of deliberation scheduling. In related work, Vincent Cicirello (2003) shows that generating explicit models of solution quality combined with online self-analysis of performance can allow a system to generate effective search control mechanisms.

Any-Time Algorithms and Continual Computation

Much of the recent work in the monitoring and control of deliberation has focused on the related concepts of any-time algorithms and continual computation, each of which grew out of different aspects of earlier work on bounded rationality, or optimal computation under constraints.

An any-time algorithm is one designed to generate solutions of continually increasing quality, such that whenever it is terminated it will produce the best currently available solution, given its time constraints. The question of interest here is what sorts of monitoring and control of any-time algorithms can enhance their performance.

Continual computation, in contrast, addresses the issue not of finding the best (boundedly optimal) use of time in solving a given problem, but the best use of idle computational resources between bouts of problem solving. This approach broadens the definition of a "problem" to include not just individual instances, but the class of challenges that a given computational system is expected to face over its lifetime. The metareasoning challenge here is to anticipate future needs and use idle time to prepare proactively for these expected problems.

Turning first to the monitoring and control of any-time algorithms, we observe that, given the basic character of these algorithms, the central (and in some ways, the only) control problem is determining when to terminate the algorithm: when is the best trade-off between resource use and solution quality achieved? If the rate of increase in solution quality is reliable and known, then this problem can be easily solved; these conditions are, however, rarely met.

Eric Hansen and Shlomo Zilberstein (2001) offer a comprehensive framework for approaching the issue of monitoring and control of any-time algorithms. They formalize the metalevel control problem as a sequential decision problem that can be solved by dynamic programming. Their approach takes into account factors such as the quality of the available solution, the prospect for further improvement in solution quality, the current time, the cost of delay in action, the current state of the environment, and the prospect for further change in the environment.

One interesting aspect of this general problem is addressed in detail by Lev Finkelstein and Shaul Markovitch (2001). If we assume that a given any-time algorithm can terminate as soon as it generates a solution meeting some threshold criterion then, clearly, the sooner it terminates after reaching that threshold, the more efficient will be its use of resources. However, because monitoring the progress of an algorithm itself consumes resources, continual, or even very frequent, monitoring could greatly reduce the overall efficiency of the process. Finkelstein and Markovitch tackle this issue, and present an offline solution to the problem of generating optimal monitoring schedules for any-time processes.

Turning our attention to continual computation, Eric Horvitz (2001) argues for a shift in focus from efficiently dealing with individual problems as they arise, to thinking about the optimal use of resources over the entire computational lifetime of a device. Horvitz suggests that systems should be designed to generate and evaluate information about the likelihood of future problem instances, ranging from detailed probability distributions to more qualitative orderings by likelihood, and implement computational resource allocation policies that can guide the ideal expenditure of idle time. He discusses various models of system use, derives optimal idle-time allocation policies for these models, and illustrates their utility with practical examples such as precaching media content based on predicted user behaviors while web-surfing.

In a variation on the issue of deliberation control, Claudia Goldman and Shlomo Zilberstein (2003) present a method for generating an optimal policy for information sharing between distributed decision makers, where these decision makers can be autonomous agents or individual computational processes in a single system. This extends the problem of deliberation scheduling to a multiagent context, where not just the cost of deliberation, but also the cost and risks of information transfer must be considered. Reinhard Stolle, Apollo Hogan, and Elizabeth Bradley (2005) also discuss an approach to this more general problem. Their solution involves a declarative representation of the state and operations of the distributed system that can be used to control the system in real time.

Using Metarepresentations in Reasoning

There is also a wealth of work exploring the

second area of metareasoning research, of trying to gather and represent information about the object-level reasoner or reasoning process, and using this information to improve performance. Most of this work falls into the category of logic-based AI and is aimed at improving the performance (speed, efficiency, expressivity, contradiction tolerance, and so on) of theorem provers.

For instance, Konstantine Arkoudas and Selmer Bringsjord's (2004) work in multiagent epistemic logics demonstrates that explicit metareasoning—reasoning that takes the epistemic logic as its object—can facilitate theorem proving in the object-level language, in part because it is possible to take advantage of higher-order structures and quantify over elements of the object-level language, thereby significantly enhancing efficiency.

A different benefit of metareasoning is emphasized by recent work on active logic, which suggests that a reasoning system equipped with the ability to represent aspects of its own state (such as the fact that the knowledge base contains a contradiction), as well as the ability to finely control inference based on such information (preventing further inference with the contradictands), is well suited to reliable operation in dynamic, real-world situations where both expressivity and contradiction tolerance are at a premium (Purang 2001).

Michael Anderson and Donald Perlis (2005a) extend this work on the importance of self-monitoring and self-control in logic-based systems to the problem of intelligent system design in general. They introduce the "metacognitive loop" (MCL), a capacity for self-monitoring and self-guided learning based on the conviction that artificial agents should be able to notice when something is amiss, assess the anomaly, and guide a solution into place. They argue that metacognitive skills are the key to robust and error-tolerant systems, outline a general architecture and approach to building such systems, and discuss some examples of implemented systems for which adding an MCL component improves performance. They show, for instance, that even simple metacognitive monitoring and control components can improve the learning speed of reinforcement learners in changing environments (Anderson et al. 2006), and that metareasoning and metalinguistic ability can improve the ability of natural-language human-computer interfaces to accurately interpret users' intentions, and to recover from miscommunication failures (Josyula 2005).

In addition to using representations of object-level reasoning to control and change those object-level processes, one can also use this technique to enhance their expressivity. Jonas Barklund, Pierangelo Dell'Acqua, Stefania Costantini, and Gaetano Lanzarone (2000) discuss the use of logic schema to capture basic properties of the domain represented in an object-level language. This results in a metalevel abstraction of the object-level domain knowledge, which allows for greater expressivity and increased inferential power in the system as a whole.

The natural complement to the effort to build metacognitive systems is the necessity to test those systems. Scott Wallace, for instance, is building reusable frameworks for validating the behavior of self-monitoring agents (Wallace 2005). In this area the ongoing National Institute of Standards and Technology (NIST) workshop series Performance Metrics for Intelligent Systems (Messina and Meystel 2004) deserves special mention, although the evaluation techniques developed and discussed are of course not restricted to only the evaluation of metacognitive systems.

As should be clear by now, most of the work in computational metareasoning is directed to practical ends. Nevertheless, there are still some theoretical issues that draw researchers' attention. For example, Vincent Conitzer and Tuomas Sandholm (2003) point out that while there is a wealth of results for the complexity of basic reasoning strategies, there has been little similar work for metareasoning. Thus, they present an analysis of some basic metareasoning strategies, and the results should be somewhat sobering for researchers enamored with metareasoning as a computational strategy. They find, for instance, that the problem of allocating deliberation time across any-time algorithms running on different problem instances is nondeterministic polynomial (NP) complete. The problem of dynamically allocating deliberation or information-gathering resources across multiple possible actions is shown to be NP-hard, even when evaluating each individual action is extremely simple. And finally they show that the problem of dynamically choosing a limited number of deliberation or information-gathering actions to disambiguate the state of the world is NP-hard under a natural restriction, and PSPACE-hard in general.

Tackling a different set of theoretical issues, Thomas Bolander (2003) lays out some of the problems inherent in logical reasoning with self-reference. In particular, he argues for the importance of (and demonstrates some methods for) avoiding certain paradoxes of self-ref-

erence, such as in the well-known example: "This sentence is false." In fact, the mysteries of self-reference and self-awareness continue to draw a great deal of attention from researchers in computer science. Len Schubert (2005) outlines some basic enhancements to typical knowledge representation and reasoning schemas that will be required if they are to be able to support self-knowledge, and Melanie Mitchell (2005) asks how self-representation and self-control should even be conceived in the case of distributed and decentralized systems, like the human brain "consisting of billions of cells with no central control." She draws some inspiration from decentralized biological systems like the immune system to help answer this question. Concerns about self-representation lead naturally to questions about self-consciousness, and while this topic is too large and diverse to be discussed in this context, we mention Holland (2003) and Anderson and Perlis (2005b) as computationally grounded starting places for researchers interested in this topic.

Work on Metalearning

Despite the fact that the machine-learning community, along with others such as statistics and data mining, has developed a wide array of "computer programs that automatically improve with experience" (Mitchell 1997), much of the burden of applying these programs falls on humans. Given a problem for which machine learning is an appropriate solution, one must select a learning algorithm, set values of parameters required by the algorithm, choose a set of features thought to be relevant in the domain, gather data, run the algorithm, evaluate the results, and, often, revisit some of the decisions made earlier and iterate. There are many definitions of metalearning, but, informally, most definitions involve automating one or more of these decisions.

The decisions made by humans when solving problems with machine learning impose a *bias,* a preference for some hypotheses over others. David Hume's conclusion that there is no rational basis for induction (Hume 1740), David Wolpert's No Free Lunch (NFL) theorems (Wolpert and Macready 1995; Wolpert 2001), and Cullen Schaffer's Law of Conservation for Generalization Performance (LCGP) (Schaffer 1994) all point to the fact that successful generalization requires an appropriate bias (Mitchell 1991). The terms *metalearning, learning to learn,* and *lifelong learning* are often used interchangeably in the machine-learning literature, and all typically refer to automatically or dynamically learning an appropriate bias. This can take many forms, from learning to predict which algorithm(s) will perform best in a new problem domain based on features of the domain itself, to developing self-modifying learning algorithms, and many others that we will discuss later.

One theme common to much of the work on metalearning is learning from multiple, related tasks. Very recently, the Defense Advanced Research Projects Agency (DARPA) Transfer Learning program began funding work in precisely this area, and there was a workshop on structural knowledge transfer at the 2006 International Conference on Machine Learning (ICML-06). Computationally, the assumption that the tasks faced by the learner are related to one another represents a powerful bias at the metalevel, ensuring that metalearning algorithms do not run afoul of the NFL theorems or the LCGP. Indeed, it has been shown both empirically (Thrun 1996) and theoretically (Baxter 2000) that this bias can improve generalization performance, especially when training data are scarce. Pragmatically, much human learning (some would claim all human learning [Hintzman 1994]) involves transfer—adapting knowledge learned in one domain to facilitate learning in another domain—and we would like to deploy machine learners in domains similar to those faced by humans. For example, when learning to play lacrosse, everything from motor skills to tactics to strategies learned while playing soccer can serve as a starting point, with subsequent modification through learning given experience in the lacrosse domain.

In what follows, we will use the term *metalearning* somewhat broadly, in the informal sense described above of automating one or more of the decisions required to apply machine learning to a task or tasks. This high-level view of metalearning can be instantiated in a variety of ways, and we now turn our attention to describing a representative (though necessarily incomplete) sample of them.

Learning to Choose a Learning Algorithm

Some of the earliest work on metalearning in the machine learning community focused on automated algorithm selection (Aha 1992, Brodley 1994), particularly in the area of supervised classification. Given a dataset, one can compute metalevel features of the dataset such as the number of instances, the number of class labels, the number and type of attributes per instance, and so on. When these features are

computed for a large number of datasets, and the performance of a variety of learning algorithms is measured on these datasets, learning can occur at the metalevel where the goal is to learn a mapping from dataset features to algorithm performance.

Recent work in this area has considered using features other than those derived directly from the data as input to the metalevel learner. For example, Hilan Bensusan and colleagues (Bensusan, Giraud-Carrier, and Kennedy 2000) noted that a decision tree is learned for each domain, and features of the learned tree—for example, number of nodes, depth, shape—serve as features that are used to predict the performance of other classification algorithms on the same dataset. Bernhard Pfahringer (Pfahringer, Bensusan, and Giraud-Carrier 2000) noted that learning algorithms are divided into two sets, and the performance of the algorithms in the first set serves as a feature vector that is used when learning to predict the performance of the algorithms in the second set.

Ensemble Methods: Metalearning or Not?

Ensemble methods combine the outputs of multiple classifiers (hypotheses) in various ways. In stacked generalization (Wolpert 1992), a set of base classifiers is trained either on different subsets of the data or using different types of classifiers on the entire dataset, and a metalevel classifier is trained to predict a class label given the predictions of the base classifiers for each instance (but not the feature values used to train the base classifiers). In bagging (Breiman 1996), multiple classifiers are trained by sampling from the full dataset; and in boosting (Freund and Schapire 1996), multiple classifiers are trained by iteratively reweighting instances to emphasize those that are misclassified, retraining on the reweighted data, and adding the resulting classifier to the ensemble. All three of these methods have been shown to improve generalization accuracy. Are these methods metalearning methods? It depends on your working definition of metalearning. In their survey of metalearning, Ricardo Vilalta and Youssef Drissi (2002) claim that stacked generalization is metalearning because the transformed dataset conveys metalevel information about the performance of the base classifiers, but that bagging and boosting are not metalearning methods. From the standpoint of metalearning as dynamic bias change, bagging and stacking tend to reduce errors by reducing variance while having little impact on bias (see Friedman [1997] for a good introduction to the bias/variance decomposition), whereas the opposite is true of boosting, contradicting the conclusions of Vilalta and Drissi.

Parameter Sharing

When classification tasks are related, it is reasonable to assume that the solutions to the tasks (chosen hypotheses) will resemble one another. One way this has been cashed out is through parameter sharing among learned models. Given a set of N classification tasks with common input spaces, rather than training N neural networks, each with a single output, Rich Caruana (1997) constructed one neural network with N outputs. Training such a network to predict all N class labels simultaneously for each input produces representations in the hidden layer that are useful across tasks, rather than within a single task.

The idea of parameter sharing is cashed out differently in Bayesian approaches to metalearning. Some of the very early work on metalearning was done in the statistics community under the rubric of hierarchical Bayes (Berger 1985; Good 1980). More recently, Greg Allenby and Peter Rossi (1999) considered learning linear functions where the weight vectors defining the functions to be learned are drawn from a common multidimensional Gaussian distribution. The smaller (larger) the variance of this distribution, the more (less) closely related the tasks. An iterative Gibbs sampling algorithm is used to simultaneously estimate the parameters of the Gaussian and the weight vectors of the individual functions. Bart Bakker and Tom Heskes (2003) take a similar approach, but assume a mixture of Gaussians rather than a single Gaussian.

In a similar vein, rather than assuming the weight vectors are drawn from a common distribution, in the context of support vector machines, Theodoros Evgeniou and Massimiliano Pontil (2004) assume each weight vector is the sum of W_0, which is shared by all tasks, and V_i, which is specific to the ith task. That is, $W_i = W_0 + V_i$. The standard optimization problem is modified to include a weighted penalty on the sum of the magnitudes of the V_i. As that weight increases, the V_i are forced to be smaller, and the tasks become increasingly similar because W_0 dominates. This work was later augmented to include kernels that produce vector valued outputs (Evgeniou and Pontil 2004, Micchelli and Pontil 2005), much like Caruana's neural networks with vector valued outputs.

Theoretical investigations

A large body of theoretical results exists for single-task learning, and some recent work has begun to shed light on theoretical issues in multitask learning. Jonathan Baxter (2000) considers learners who are given a family of hypothesis spaces and a set of related tasks, and must find a hypothesis space that is suitable for the entire set of tasks. An extension of the well-known VC dimension is defined and then used to derive upper bounds on the sample complexity (number of training instances) required for good generalization. Shai Ben-David and Reba Schuller (2003) assume that the tasks are related through a data generation mechanism that they define and obtain tighter bounds that hold on a per task basis, rather than averaged over all tasks.

Learning New Learning Algorithms

Marvin Minsky asks "should we suppose that outstanding minds are any different from ordinary minds at all, except in matters of degree?" (Minsky 1982). His answer is, partially, no; to have an outstanding mind rather than an ordinary mind "one must learn to be better at learning!" Perhaps the most tantalizing prospect of metalearning is precisely that, algorithms that learn to improve their ability to learn. The work of Juergen Schmidhuber and his colleagues, especially Marcus Hutter (2004), addresses this prospect.

The Gödel machine (Schmidhuber 2005) is "the first class of mathematically rigorous, general, fully self-referential, self-improving, optimal reinforcement learning systems." The idea here is to start the machine with a suboptimal program *P* for interacting with the environment, such as Q-learning, and a theorem prover that systematically generates pairs of the form *(switchprog, proof)* until it finds a *proof* that executing *switchprog* on *P* will result in higher utility than leaving *P* alone. This approach is shown to be "globally optimal" (that is, there are no local minima) because proving that the new program obtained by running *switchprog* has higher utility than continuing with the current program includes cases where the current program would find an even better *switchprog* later. Sadly, as one might expect, the computational complexity of this approach is prohibitively expensive. Despite the fact that the Gödel machine is practically infeasible, the importance of this works derives from the theoretical characterization of the problem and showing that at least one solution exists.

Conclusion

Natural intelligent systems tend to be robust; artificial intelligent systems tend to be brittle. Humans may not behave optimally very often, but we can muddle through in just about any circumstances. In contrast, many artificial systems exist that behave optimally in narrow domains but simply cannot function at all outside of those domains. Worse still, these systems often do not even recognize that the domain has changed and blithely continue computing solutions to the wrong problems. One pillar upon which robustness rests is metacognition (Flavell 1979, 1987), including both metareasoning and metalearning. If artificial intelligent systems are to move beyond the ingenuity of their designers and have extended interactions with a dynamic world, research in metacognition is vitally important. As this review and the companion articles suggest, there is much to be excited about in this area.

References

Aha, D. 1992. Generalizing from Case Studies: A Case Study. In *Proceedings of the 9th International Workshop on Machine Learning,* 1–10. San Francisco: Morgan Kaufmann Publishers.

Allenby, G. M., and Rossi, P. E. 1999. Marketing Models of Consumer Heterogeneity. *Journal of Econometrics* 89(1–2): 57–78.

Anderson, M.; Oates, T.; Chong, Y.; and Perlis, D. 2006. Enhancing Reinforcement Learning with Metacognitive Monitoring and Control for Improved Perturbation Tolerance. *Journal of Experimental and Theoretical Artificial Intelligence*. 18(3): 387–411.

Anderson, M., and Perlis, D. 2005a. Logic, Self-Awareness, and Self-Improvement: The Metacognitive Loop and the Problem of Brittleness. *Journal of Logic and Computation* 15(1): 21–40.

Anderson, M., and Perlis, D. 2005b. The Roots of Self-Awareness. *Phenomenology and the Cognitive Sciences* 4(3): 297–333.

Arkoudas, K., and Bringsjord, S. 2004. Metareasoning for Multi-Agent Epistemic Logics. In *Proceedings of the Fifth International Workshop on Computational Logic in Multi-Agent Systems,* Lecture Notes in Artificial Intelligence 3487, ed. João Leite and Paolo Torroni, 111–25. Berlin: Springer.

Bakker, B., and Heskes, T. 2003. Task Clustering and Gating for Bayesian Multitask Learning. *Journal of Machine Learning Research* 4(2003): 83–99.

Barklund, J.; Dell'acqua, P.; Costantini, S.; and Lanzarone, G. 2000. Reflection Principles in Computational Logic. *Journal of Logic and Computation* 10(6): 743–86.

Baxter, J. 2000. A Model of Inductive Bias Learning. *Journal of Artificial Intelligence Research,* 12: 149–198.

Ben-David, S., and Schuller, R. 2003. Exploiting Task Relatedness for Multiple Task Learning. In *Learning*

Theory and Kernel Machines, Lecture Notes in Artificial Intelligence 2777, ed. B. Schölkopf and M. Warmuth, 567–80. Berlin: Springer.

Bensusan, H.; Giraud-Carrier, C.; and Kennedy, C. 2000. A High Order Approach to Metalearning. Paper presented at the Eleventh European Conference on Machine Learning, Workshop on Metalearning, Barcelona, Spain, May–June.

Berger, J. O. 1985. *Statistical Decision Theory and Bayesian Analysis*. Berlin: Springer.

Bolander, T. 2003. Logical Theories for Agent Introspection. Ph.D. dissertation, Computer Science, Technical University of Denmark, Lyngby, Denmark.

Brieman, L. 1996. Bagging Predictors. *Machine Learning* 24(2): 123–40.

Brodley, C. 1994. Recursive Automatic Bias Selection for Classifier Construction. *Machine Learning* 20(1–2): 63 – 94

Caleiro, C.; Vigan, L.; and Basin, D. 2005. Metareasoning about Security Protocols Using Distributed Temporal Logic. In Proceedings of the Workshop on Automated Reasoning for Security Protocol Analysis, *Electronic Notes In Theoretical Computer Science* 125(1): 67–89.

Caruana, R. 1997. Multitask Learning. *Machine Learning* 28(1): 41–75.

Cicirello, V. 2003. Boosting Stochastic Problem Solvers through Online Self-Analysis of Performance. Ph.D. dissertation, Carnegie Mellon University, Technical Report CMU-RI-TR-03-27, Pittsburgh, PA.

Cohen, M., and Thompson, B. 2005. Metacognitive Processes for Uncertainty Handling: Connectionist Implementation of a Cognitive Model. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Conitzer, V., and Sandholm, T. 2003. Definition and Complexity of Some Basic Metareasoning Problems. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. San Francisco: Morgan Kaufmann Publishers.

Costa, F. 2001. Combining Meta-Information Management and Reflection in an Architecture for Configurable and Reconfigurable Middleware. Ph.D. dissertation, Lancaster University, Computing Department, Lancaster, UK.

Costantini, S. 2002. Meta-Reasoning: A Survey. In *Computational Logic: From Logic Programming into the Future: Special Volume In Honour of Bob Kowalski,* ed. A. Kakas and F. Sadri. Berlin: Springer-Verlag.

Cox, M. 2005. Metacognition in Computation: A Selected Research Review. *Artificial Intelligence* 169(2): 104–41.

Evgeniou, T., and Pontil, M. 2004. Regularized Multitask Learning. In *Proceedings of the Tenth Conference on Knowledge Discovery and Data Mining*. New York: Association for Computing Machinery.

Finkelstein, L., and Markovitch, S. 2001. Optimal Schedules for Monitoring Anytime Algorithms. *Artificial Intelligence* 126(1–2): 63–108.

Flavell, J. H. 1979. Metacognition and Cognitive Monitoring: A New Area of Cognitive-Developmental Inquiry. *American Psychologist* 34(10): 906–11.

Flavell, J. H. 1987. Speculations about the Nature and Development of Metacognition. In *Metacognition, Motivation and Understanding,* ed. F. E. Weinert and R. H. Kluwe, 21–9. Hillside, New Jersey: Lawrence Erlbaum Associates.

Freund, Y. and Schapire, R. 1996. Experiments with a New Boosting Algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning,* 148–56. San Francisco: Morgan Kaufmann Publishers.

Friedman, J. 1997. On Bias, Variance, 0/1-Loss, and the Curse of Dimensionality. *Data Mining and Knowledge Discovery* 1(1): 55–77.

Ganek, A. G., and Corbi, T. A. 2003. The Dawning of the Autonomic Computing Era. *IBM Systems Journal* 42(1): 5–18.

Garfinkel, T., and Rosenblum, M. 2003. A Virtual Machine Introspection-Based Architecture for Intrusion Detection. In *Proceedings of the Network and Distributed Systems Security Symposium*. Reston, VA: Internet Society.

Goldman, C. V., and Zilberstein, S. 2003. Optimizing Information Exchange in Cooperative Multi-Agent Systems. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*. New York: Association for Computing Machinery.

Goldman, R.; Musliner, D.; and Krebsbach, K. 2003. *Managing Online Self-Adaptation In Real-Time Environments,* Lecture Notes In Computer Science, 2614, 6–23. Berlin: Springer.

Good, I. J. 1980. Some History of the Hierarchical Bayesian Methodology. In *Bayesian Statistics II,* ed. J. M. Bernardo, M. H. D. Groot, D. V. Lindley, and A. F. M. Smith. Oxford: Oxford University Press.

Hansen E., and Zilberstein, S. 2001. Monitoring and Control of Anytime Algorithms: A Dynamic Programming Approach. *Artificial Intelligence* 126(1-2): 139–157.

Hintzman, D. L. 1994. Twenty-Five Years of Learning and Memory: Was the Cognitive Revolution a Mistake? In *Attention and Performance,* Volume XV, ed. C. Umlita and M. Moscovitch, chapter 16: 360–91. Cambridge, MA: The MIT Press.

Hobbs, J., and Gordon, A. 2005. Toward a Large-Scale Formal Theory of Commonsense Psychology for Metacognition. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Holland, O., ed. 2003. *Machine Consciousness*. London: Academic Press.

Horvitz, E. 2001. Principles and Applications of Continual Computation. *Artificial Intelligence* 126(1-2): 159–96.

Hudlicka, E. 2005. Modeling Interactions between Metacognition and Emotion in a Cognitive Architecture. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Hume, D. 1740. *A Treatise of Human Nature*. Ed. D. F. Norton and M. J. Norton. Oxford: Oxford University Press, 2000.

Hutter, M. 2004. *Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability.* Berlin: Springer.

Josyula, D. 2005. A Unified Theory of Acting and Agency for a Universal Interfacing Agent. Ph.D. dissertation, Dept. of Computer Science, University of Maryland, College Park, MD.

Kennedy, C. 2003. Distributed Reflective Architectures for Anomaly Detection and Autonomous Recovery. Ph.D. dissertation, School of Computer Science, University of Birmingham, Birmingham, UK.

Kim. J. 2005) Memory Based Meta-Level Reasoning for Interactive Knowledge Capture. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Lundström, J.; Hamfelt A.; and Nilsson, J. 2005. Argumentation as a Metacognitive Skill of Passing Acceptance. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Messina, E., and Meystel, A., eds. 2004. *Proceedings of the 2004 Performance Metrics For Intelligent Systems Workshop*. Gaithersburg, MD: National Institute of Standards and Technology.

Micchelli, C. A.; and Pontil, M. 2005. Kernels for Multitask Learning. In *Proceedings of the 18th Conference on Neural Information Processing Systems*. Cambridge, MA: The MIT Press.

Minsky, M. 1982. Why People Think Computers Can't. *AI Magazine* 3(4): 3–15.

Mitchell, M. 2005. Self-Awareness and Control in Decentralized Systems. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Mitchell, T. 1991. The Need for Bias in Learning Generalizations. In *Readings In Machine Learning,* ed. T. Dietterich and J. Shavlik. San Francisco: Morgan Kaufmann Publishers.

Mitchell, T. 1997. *Machine Learning*. New York: McGraw-Hill.

Muldner, K., and Conati, C. 2005. Providing Adaptive Support for Meta-Cognitive Skills to Improve Learning. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Murdock, J. W. 2001. Self-Improvement through Self-Understanding: Model-Based Reflection for Agent Adaptation. Ph.D. dissertation, Dept. of Computer Science, Georgia Institute of Technology, Atlanta, GA.

Oehlmann, R. 2003. Metacognitive and Computational Aspects of Chance Discovery. *New Generation Computing* 21(1): 3–12.

Pfahringer, B.; Bensusan, H.; and Giraud-Carrier, C. 2000. Metalearning by Landmarking Various Learning Algorithms. In *Proceedings of the Seventeenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann Publishers.

Purang, K. 2001. Systems that Detect and Repair Their Own Mistakes. Ph.D. dissertation, Dept. of Computer Science, University of Maryland, College Park, MD.

Russell, S. 2002. Rationality and Intelligence. In *Common Sense, Reasoning, and Rationality,* ed. R. Elio, 337–59. Oxford: Oxford University Press.

Russell, S., and Wefald, E. 1991. Principles of Metareasoning. *Artificial Intelligence* 49(1–3): 361–95.

Sandholm, T. 2003. *Principles and Practice of Constraint Programming*. Lecture Notes In Computer Science, 2833: 950–55. Berlin: Springer.

Schaffer, C. 1994. A Conservation Law for Generalization Performance. In *Proceedings of the Eleventh International Conference on Machine Learning,* 259–65. San Francisco: Morgan Kaufmann Publishers.

Schmidhuber, J. 2005. Completely Self-Referential Optimal Reinforcement Learners. Paper presented at the International Conference on Artificial Neural Networks (IJCNN), 31 July–4 August, Montreal, Quebec, Canada.

Schubert, L. 2005. Some Kr&R Requirements for Self-Awareness. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Schut, M., and Wooldridge, M. 2001. Principles of Intention Reconsideration. In *Proceedings of the Fifth International Conference on Autonomous Agents,* 340–47. New York: Association for Computing Machinery.

Stolle, R.; Hogan, A.; and Bradley, E. 2005. Agenda Control for Heterogeneous Reasoners. *Journal of Logic and Algebraic Programming* 62(1): 41–69.

Sullivan, G. 2001. Aspect-Oriented Programming Using Reflection and Metaobject Protocols. *Communications of the ACM* 44(10): 95–7.

Swanson, R., and Gordon, A. 2005. Automated Commonsense Reasoning About Human Memory. In *Metacognition in Computation: Papers from the 2005 AAAI Spring Symposium*, Technical Report SS-05-04, ed. M. Anderson and T. Oates. Menlo Park, CA: AAAI Press.

Thrun, S. 1996. Is Learning the Nth Thing Any Easier Than Learning the First? *Advances In Neural Information Processing Systems 8*, 640–6. Cambridge, MA: The MIT Press.

Vilalta, R., and Drissi, Y. 2002. A Perspective View and Survey of Metalearning. *Artificial Intelligence Review* 18(2): 77–95.

Wallace, S. 2005. S-Assess: A Library for Self-Assessment. In *Proceedings of the Fourth International Conference on Autonomous Agents and Multiagent Systems,* 256–63. New York: Association for Computing Machinery.

Welch, I., and Stroud, R. J. 2002. Using Reflection as a Mechanism for Enforcing Security Policies on Compiled Code. *Journal of Computer Security* 10(4): 399–432.

Williams, B.; Ingham, M.; Chung, S.; Elliott, P.; and Hofbaur, M. 2004. Model-Based Programming of Fault-Aware Systems. *AI Magazine* 24(4): 61–75.

Wolpert, D. 1992. Stacked Generalization. *Neural Networks* 5(2): 241–59.

Wolpert, D. 2001. The Supervised Learning No Free Lunch Theorems. Paper presented at the Sixth On-Line World Conference on Soft Computing in Industrial Applications, Berlin, 10–24 September.

Wolpert, D., and Macready, W. 1995. *No Free Lunch Theorems for Search*. Technical Report SFI-TR-95-02-010. Santa Fe Institute, Santa Fe, NM.

Zachary, W., and Le Mentec, J.-C. 2000. Incorporating Metacognitive Capabilities in Synthetic Cognition. Paper presented at the Ninth Conference on Computer Generated Forces, Orlando, FL, May.

Zheng, J., and Horsch, M. 2005. *A Decision Theoretic Meta-Reasoner for Constraint Optimization,* Lecture Notes In Computer Science, 3501, 53–65. Berlin: Springer.

Michael L. Anderson (michael.anderson@fandm.edu) is an assistant professor in the Department of Psychology at Franklin and Marshall College and visiting assistant professor at the Institute for Advanced Computer Studies at the University of Maryland, College Park, where he is also a member of the graduate faculty in the program in neuroscience and cognitive science. He earned a B.S. in premedical studies from the University of Notre Dame and a Ph.D. in philosophy from Yale University, and he did his postdoctoral training in computer science at the University of Maryland. Anderson is author or coauthor of more than 40 scholarly and scientific publications in artificial intelligence, cognitive science, and philosophy of mind. His primary areas of research include the role of behavior, and of the brain's motor-control areas, in supporting higher-order cognitive functions; the foundations of intentionality (the connection between objects of thought and things in the world); and the role of self-monitoring and self-control in maintaining robust real-world agency.

Tim Oates is an assistant professor in the Department of Computer Science and Electrical Engineering at the University of Maryland Baltimore County. He received his Ph.D. from the University of Massachusetts Amherst in 2001. In 2004 he received an NSF CAREER award. His research interests are in artificial intelligence and machine learning, with a focus on the sensorimotor foundations of knowledge.