In *Cambridge Handbook of Intelligence* (3rd Edition),
R.J. Sternberg & S.B. Kaufman (Editors), 2011.

# Artificial Intelligence

*Ashok K. Goel*
School of Interactive Computing
Georgia Institute of Technology
goel@cc.gatech.edu

*Jim Davies*
Institute of Cognitive Science
Carleton University
jim@jimdavies.org

## Introduction

Artificial intelligence (AI) is the field of research that strives to understand, design and build cognitive systems. From computer programs that can beat top international grand masters at chess to robots that can help detect improvised explosive devices in war, AI has had many successes. As a science, it differs from cognitive psychology in two ways. Firstly, its main methodology is the exploration of cognitive theory by building intelligent artifacts. Though the design of any intelligent artifact would be classified as an AI, AI as a discipline is united in the core belief that intelligence is a kind of computation. Thus, in practice, AI artifacts are almost always computers or computer programs. This also explains why AI laboratories typically are found in computer science departments.

Secondly, psychology is mostly interested in the understanding of intelligence found naturally in humans and other animals, whereas, in addition, AI concerns itself with the understanding of intelligence in agents it designs. From the AI perspective, the concept of intelligence is not one that should be limited to

the abilities of humans or even animals in general, but should cover potentially any kind of intelligent system, be it human, computer, animal, or alien. Albus (1991, p. 474) puts it eloquently: "A useful definition of intelligence ... should include both biological and machine embodiments, and these should span an intellectual range from that of an insect to that of an Einstein, from that of a thermostat to that of the most sophisticated computer system that could ever be built."

To demonstrate this latter difference, it is helpful to distinguish AI research into two kinds. *Engineering AI* concerns itself with how to design the smartest intelligent artifacts possible, regardless of whether the processes implemented reflect those found in natural intelligences. The vast majority of AI research falls into this category. *Cognitive AI*, in contrast, endeavors to design artifacts that think the way people (or sometimes other animals) do. A sub-category of cognitive AI is *cognitive modeling*, which tries to quantitatively model empirical human participant data. Many cognitive modeling groups are working in psychology departments. AI cognitive models differ from other models in psychology in that AI always implements *information-processing* theories. That is, the theory describes intelligence in terms of content, representation, access, use and acqusition of information, as opposed to, for example, a statistical model of the influences on IQ (e.g., age) in a population.

This article focuses on cognitive AI for several reasons: The original dream of AI was to develop human-level intelligence, this handbook is intended for an audience of cognitive scientists, and the authors themselves work in this paradigm.

Be it a leftover from Cartesian dualism, or a desperate hold onto the uniqueness of humanity, many people have an almost mystical view of intelligence. One result is that when an AI program manages to accomplish some cognitive task, a common reaction is to claim that it's not an example of intelligence. Indeed, at one point arithmetic calculation was thought to be one of best displays at intelligence, but now almost no one wants to say their calculator is intelligent. Because of this moving of the goal posts, AI has been jokingly

referred to as standing for "Almost Implemented." For the most part, this is only a semantic issue. In fact, AI discoveries have revolutionized our world; although not always labeled as AI, the findings of the field have been used so widely in the software the runs our businesses and financial transactions that our economy as we know it would grind to a halt without the work of AI-inspired programs (Kurzweil, 2005). Among many, many other applications, AIs help land our airplanes, understand our voices on the phone, and detect credit card fraud.

**1. What AI Brings to Cognitive Sciences**

Critics of AI from psychology sometimes view AI programs as being psychologically implausible. Indeed, cognitive claims of AI theories typically are under-constrained by empirical human data, and thus, for the most part, criticisms of AI from psychology are not inaccurate. Most AI is engineering AI, and even cognitive AI must go out on limbs simply because there just is not enough data to constrain all the choices AI scientists need to make. However, AI contributes to the understanding of intelligence in several ways.

First, although they can be underconstrained, *AI programs demonstrate what kinds of data need to be collected.* Because AI works at a very precise level of detail, it brings to light theoretical ambiguities that psychology may not immediately or explicitly realize it needs to make. For example, it is one thing to say that a person can only comprehend one speaking voice heard at a time. It is quite another to create a computer implementation of this attentional effect --- to do so requires making decisions about the interaction and influences of volume, which one voice you are listening to first, what factors affect attentional switching, among many other issues. The level of detail that makes AI programs underconstrained is the very quality that brings to light previously un-conceived factors.

Humans obviously have only limited information and information-processing resources, and, thus, their rationality is intrinsically bounded (Simon 1969). However, it is also true that many cognitive problems that people routinely solve are computationally intractable. For example, deciding how to design a

poster for a concert offers more possibilities than can possibly be considered in reasonable time. *AI approaches to solving intractable problems shed light on what ways will not work.* If AI shows that a means for solving a problem will take too long to be practical, then AI has shown that people cannot be doing it that way, at least not routinely.

On the other hand, *AI can show that certain methods are possible.* Though showing that something is possible is far from proving that it *is,* many current theories in psychology do not have such proof. AI serves a valuable function as creating proofs-of-concept.

Another thing AI is particularly good at is *exploring the benefits and limitations of various ways to represent and organize knowledge in memory.* Many of these benefits are only clear when dealing with a strict information-processing level of detail. Are beliefs represented as words, pictures, or something else? Given all of the cognitive tasks memories are supposed to contribute to, AI is in a good position to shed light on such issues. As we will describe in more detail below, this subfield of AI is known as "knowledge representation."

Finally, once there is an AI program that resembles some part of human thinking to a researcher's satisfaction, *it is possible to run experiments on the program that are either unethical or too expensive (in terms of time or money) to run on living beings.* In simulation you can run thousands of experiments in a day, with exquisite control over all variables.

## 2. Navigational Planning: An Illustrative Example

We want to illustrate a simple example of AI in some detail so that this chapter is more than just so many big words. Let us suppose that Sunny, a cheerful AI agent, is about to start a new job in a new city. Sunny starts its car at its apartment and needs to navigate to an office building in downtown. How might Sunny think and what might Sunny do, given that this is its first day in the city and it has never been to the office building? Our goals in this section are to

explain some dimensions in designing AI agents as well as describe some issues in putting multiple capabilities into an AI agent.[1]

### 2.1 Action, Perception and Cognition

To reach its office from its apartment, Sunny might use one (or more) of several possible strategies. For example, it might drive its car a short distance in some direction, and then see if it has reached the office building. If it has, then it has accomplished its goal. If it has not, then it might again drive a short distance in some direction, and then again see if has reached the building. Sunny could repeat this process until it reaches its goal. Blindly moving about like this would likely take a very long time, but in terms of internal processing, this method is very efficient. This *perceive-act* internal computational processing, called *situated action* (or *reactive control;* Arkin 1999), works by perceiving the immediate environment, acting based on those perceptions, and then repeating. The computational processing in reactive control is very efficient and requires no memory. However, depending on the environment and the goal, it may produce needlessly complicated external behavior since Sunny could be driving short distances in arbitrary directions for a very long time before it reaches its goal. In fact, this strategy does not guarantee that the goal will ever be reached.

Alternatively, when Sunny started at its apartment, it might simply ask Honey, a sweet AI agent who happens to be passing by, about how to reach the office building. Honey, a long-time resident of the city, might give Sunny detailed directions, which Sunny could simply follow. In contrast to the previous strategy, this strategy produces very efficient output behavior: Assuming that Honey's directions are good, Sunny should reach its goal quite efficiently. However, this strategy of *asking* requires a society of intelligent agents (human or AI), each

---

[1] Much of our discussion of this problem is based on the work of the first author and his students in the early nineties when they developed a computer program called Router for addressing this class of problems (Goel, Ali, Donnellan, Gomez, & Callantine, 1994) and instantiated Router on a mobile reactive robot called Stimpy (Ali & Goel 1996). They also developed a knowledge-based shell called Autognostic for learning by reflection on Router program embodied in Stimpy (Stroulia & Goel 1999) as well as reflection on Stimpy's reactive controller (Goel, Stroulia, Chen, & Rowland, 1997).

with different knowledge. It also requires a culture in which Sunny may in fact approach Honey for directions, Honey may in fact stop to help Sunny, and the two can communicate in a shared language, Sunny may trust Honey, a total stranger, enough to follow its directions in a new city, etc. AI research on robot societies and human-robot interaction is in its early stages, and so here we will briefly mention only a small set of selected issues.

How may Sunny and Honey talk with each other? How may Sunny talk with a human? Understanding and generating natural language is the goal of the AI subdiscipline of *natural language processing* (NLP). Researchers in the area of natural language understanding take written text or spoken language and create accurate knowledge representations reflecting the meaning of the input. Natural language generation works roughly in the reverse --- taking in knowledge and generating appropriate words and speech to communicate that meaning; This has received much less attention in AI. Two robots might be able to share knowledge very efficiently if that knowledge is represented in the same way. However, there is little agreement in AI over how knowledge should be represented in general. Different knowledge representation strategies appear to be better for different tasks.

When Honey gives advice, how is Sunny to know whether that advice is plausible? Except for limited environments, this problem seems to require general *commonsense reasoning*, a field closely related to knowledge representation. It is a widely held belief that most computer programs lack of common knowledge and an inability to reason with it effectively is a major problem for much of AI. The subfield of commonsense reasoning endeavors to overcome this challenge. The most famous is the CYC project (Lenat & Guha, 1990), a major project to manually encode all human commonsense knowledge. More recent strategies include Web-based knowledge collection methods, such as OpenMind Commonsense (Singh, Lin, Meuller, Lim, Perkins, & Zhu, 2002) and Peekaboom (von Ahn, Liu, & Blum, 2006).

Here is another strategy by which Sunny may reach its office buidling: Let us suppose that when Sunny was originally built in an AI laboratory, it was

bootstrapped with some knowledge. Some of this knowledge may have been *heuristic* in its content and encoded in the form of a *production rule*. A heuristic is like a "rule of thumb," and a production is an "If x then do y" kind of rule. So, for example, Sunny might be bootstrapped with the knowledge that "if the goal is to reach downtown in a city, then move in the direction of the tallest buildings." This knowledge directly uses the goal (reaching downtown) to suggest a high-level action (move in the direction of the tallest buildings), and is heuristic in its nature since it may not correctly apply in all cities. If Sunny had this knowledge, then it might begin by perceiving the environment around it, locating the tallest buildings in the horizon, deciding to head in their direction, and moving toward them. When it reaches the next intersection, Sunny might again locate the tallest buildings relative to its current location, change its direction if needed, and so on. This strategy of *perceive-think-act* not only requires some knowledge but also means more complex internal processing than the simpler perceive-act strategy of situated action. On the other hand, depending on the environment, perceive-think-act may result in a far simpler external behavior because now the behavior is more explicitly directed by the goal.

This kind of strategy can be implemented as a *production system* (Newell & Simon 1972), which represents "what to do," or procedural knowledge, with if-then rules. In Sunny's case, the rules dictate physical action in the environment. Production systems are often used for making changes in memory as well. Rules can add, change, and remove goals and elements in memory. Surprisingly complex behavior can result with this method. This particular approach has been very successful in cognitive modeling. Well known cognitive architectures such as SOAR (Laird, Newell & Rosenbloom) and ACT-R (Anderson & Lebiere, 1998) are production systems at heart. Production systems have representations of declarative and procedural knowledge. Declarative knowledge is relatively static and is used by the productions (the procedural knowledge), and is often represented as *frames* (Minsky 1975). Frames are similar to classes in object-oriented programming: They define a class of entities and what attributes they have. Instances of these frames take particular values for these attributes. For

example, the frame for PERSON might contain the attributes NAME and AGE, and an instance of person might have a NAME of "Julie" and an AGE of "45." Like frames, *semantic networks* (Sowa 1987) are a widely used representation scheme in AI. One can imagine a semantic network as a map of concepts, with nodes representing concepts (such as MAN and DOG) and labeled links between them (labeled, for example, with OWNS). Frames and semantic networks are thought to be informationally equivalent, which means that there is no loss of information when translating from one to another. Semantic networks are one kind of belief representation, called in the AI literature *knowledge representation*.

Another long standing and still very strong area of AI is representation and processing based on *logic*. Logic is used for inference, but has also been adapted for use in many other specific tasks, such as theorem proving McCarthy 1988).

Let us consider one other strategy for Sunny's task before we move on to the next topic: Sunny might consult a map of the new city. The important characteristics of a city map in this context are that it is an external representation of the world (i.e., it is not stored internally in Sunny) and that it is a visuospatial model of the world (i.e., there is a one-to-one structural correspondence between selected spatial objects and relations in the world and the objects and relations on the map; see Glasgow, Narayanan & Chandrasekaran, 1995). Sunny may use this map to plan a navigation route to the office building and then execute the plan. This too is a perceive-think-act strategy. However, as compared to the heuristic method, the "thinking" in this strategy uses very different content and representation of knowledge. The internal processing in this strategy in general may be more costly than the processing in heuristic search; however, depending on the environment, this strategy might lead to a solution that has a better chance of success, for example, the solution generated by this model-based method is less likely to get stuck in some *cul de sac* than the solution generated by the heuristic method.

Once Sunny has studied the map, it has some version of it stored in its memory. When Sunny needs to navigate to a location on it, it can refer to the map. Finding a route on a map is not trivial, however. At each intersection, a choice must be made. One of the first insights of the field was that a great many cognitive problems can be solved by systematically evaluating available options. This method, that of searching through a space of choices, is applicable in many domains and is still widely used. Researchers focusing on *search* compare the various search methods that have been invented and describe the classes of problems to which each is most applicable. Because most interesting search spaces are enormous (e.g., there are more possible chess game configurations than there are atoms in the universe), researchers invent heuristics to guide the AI to explore the more promising areas of the search space. One problem for which search has been particularly useful is in *planning*, which is the generation of a ordered sequence of actions prior to actually executing those actions.

Of course we can easily think of several other strategies for addressing Sunny's task, especially in today's world of the internet and the global positioning system. But more importantly for our present discussion we also can see some of the dimensions of designing an AI agent. Firstly, an AI agent lives in some environment, and what and how an agent can think depends in large part on the environment in which the agent lives. Some environments may contain other agents, who may be cooperative, competitive or combative. Some environments are dynamic. Some environments are only partially observable. Some environments are non-deterministic, and so on. One of the many contributions of AI is a more precise characterization and analysis of different kinds of environments, though much of the AI analysis so far has focused mostly on physical, not social, environments. Secondly, an agent may have access to different kinds of knowledge contents and representations. The knowledge may be engineered or acquired. The representations can be internal or external. The knowledge contents can range from nil to heuristic rules to detailed, high-fidelity models of the environment. Another major AI contribution is more precise and detailed account of knowledge contents and representations. Thirdly, different

strategies lead to very different trade-offs among knowledge requirements, the computational efficiency of internal processing, and the quality of generated solutions and behaviors. Yet another contribution of AI is more precise enumeration and analysis of these trade-offs.

**2.2 Reasoning, Learning and Memory**

So far we have talked only about what our hypothetical AI agent, Sunny, might think and do when trying to reach its office for the first time. However, because Sunny is an AI agent, it shall also learn from its interactions with the environment. What and how might Sunny learn from its experiences? Sunny acquires a new experience each time it interacts with the environment, including navigating from its apartment to its office, talking with Honey, etc., irrespective of what internal strategy it uses. Further, to the degree to which Sunny's internal processing is accessible to it, it may also acquire an internal experience each time it does internal processing. In addition, when Sunny executes a plan or an action on the environment, the environment might provide it with feedback. This feedback may come immediately after the execution of an action (e.g., taking a turn at an intersection and getting caught in a *cul de sac*), or after a series of actions (e.g., taking a sequence of turns and reaching the goal). The feedback might simply be that the outcome - success or failure --- of a plan, or it may contain more information, for example, a specific action in the plan failed because it led to a cul de sac. Thus, an experience can contain not only an interaction with the environment, but also some feedback on the interaction, and perhaps also a trace of the internal processing in that interaction.

Sunny might potentially learn many different things from its experiences in the environment. For example, Sunny might simply encapsulate experiences as *cases* and store them in memory for reuse in future. On the first day, for example, Sunny might use a map to plan a navigation route and then execute the plan in the environment, as indicated in the previous subsection. The next day, when Sunny again faces the task of navigating to its office from its apartment, it might find a solution simply by retrieving the navigation plan in the case acquired

from the previous day, rather than relying on general-purpose knowledge and rules. This is called *case-based reasoning* (Kolodner, 1993). This approach views reasoning largely as a memory task, that is, as a task of retrieving and modifying almost correct solutions from memory to address the current problem.

As Sunny learns from its experiences, its internal processing as well as its external behaviors may change. Initially, for example, Sunny might use a map of the environment for navigating through the new city. However, as it navigates through the world and stores its experiences as cases in its memory, it may increasingly generate new navigation plans by case-based reasoning. However, as the number of cases in memory increase, the cost of retrieving the case appropriate for a new problem also increases. Thus, again, each reasoning strategy offers computational trade-offs among knowledge requirements, processing efficiency and solution quality.

More generally, AI typically thinks of each strategy for action selection discussed in the previous subsection as setting up an associated learning goal, which in turn requires a corresponding strategy for learning from experiences. Let us suppose, for example, that Sunny uses the strategy of situated action for action selection. It may, for example, use a table (called a *policy*) that specifies mappings from percepts of the world into actions on it. Then, from the feedback, or the reward, on a series of actions, Sunny can learn updates to the policy so that over time its action selection is closer to optimal. This is called *reinforcement learning* (Sutton & Barto 1998). Note that if the series of actions results in success, then the reward will be positive; otherwise it is negative. Reinforcement learning is an especially useful learning strategy when the reward is delayed, that is, it comes after a series of actions rather than immediately after an action. Alternatively, suppose that Sunny uses the strategy of using production rules such as "If x then do y" to select actions. In this case, Sunny can use the learning strategy of *chunking* (Laird, Newell & Rosenbloom 1987) to learn new rules from its experiences over time. Thus, just as AI has developed many reasoning strategies for action selection, it has developed many learning strategies for acquiring the knowledge needed by the reasoning strategies. Further, just like

the reasoning strategies, the learning strategies too offer trade-offs among knowledge requirements, computational efficiency and solution quality.

Most of the methods described thus far fall roughly into a category that can be described as "symbolic" approaches, characterized by the manipulation of qualitative, recognizable, discrete symbols. Another broad approach is quantitative or sub-symbolic. Though the border between these two approaches is fuzzy, we can think of a symbolic representation having a symbol for the letter "R" and a subsymbolic system representing the letter with the dots that make it up on a screen. Since the dots, or pixels, are not meaningful in themselves, they are thought to be at a level of description below the symbol. The rest of the methods described in this subsection tend to use subsymbolic representations.

So far we have assumed that Sunny has perfect knowledge of the environment, even if that knowledge is limited. However, many real-world domains involve uncertainty, and AI methods based on *probability* have been very successful at working in these environments. Probability theory has been used in algorithms that use *Hidden Markov Models* to predict events based on what has happened in the past. Hidden Markov Models are mathematical representations that predict the values of some variables given a history of how the values of these and other variables have changed over time (Raibiner & Juang 1986). Probabilities are also used to determine beliefs, such as how likely it is that a street Sunny wants to use has been closed, given that the rain in that part of the city was 80% likely to have been freezing. Bayes' Rule is useful for determining such conditional probabilities of some events (e.g., a road being closed) given the probability of others (e.g., freezing rain). *Bayesian belief networks* are mathematical representations that predict the probability of certain beliefs being true, given the conditional probabilities of other beliefs being true (Pearl 2000). These networks are useful for updating probabilities of beliefs as information about events in the world arrives.

Statistics is the foundation of much of *machine learning*, a subdiscipline of AI that aims to create programs that use data and limited previous beliefs to create new beliefs. There are a great many kinds of learning algorithms,

including artificial *neural networks*, which are the basis of connectionism in cognitive science (Rumelhart, McClleland & the PDP Research Group, 1986, McCelland, Rumelhart, and the PDP Research Group, 1986). Whereas most of the systems we've discussed process recognizable symbols, neural networks represent information at a sub-symbolic level (such as in pixels or bits of sound) as activations of nodes in a network. The processing of a neural network depends on how the nodes change each others' activations. The output of a neural network is an interpretation of the activations of certain nodes (for example, indicating whether or not a room is dark). *Genetic algorithms* are another means of computation that is (often) based on processing subsymbolic representations. Inspired by the theory of biological evolution, genetic algorithms create solutions to problems by applying some fitness function to a population of potential solutions (Mitchell 1998). Solutions with a high fitness are used to generate members of the next generation (often with some mutation or crossover of features), after which the process repeats.

**2.3 Deliberation and Situated Action**

Although above we briefly discussed situated action (reactive control) and situated learning (reinforcement learning), much of our discussion about Sunny, our friendly robot, pertained to deliberation. While AI theories of deliberative action selection typically are explicitly goal directed, goals in situated action often are only implicit in the design an AI agent. Deliberation and situated action in AI agents occur at different time scales, with deliberation typically unfolding at longer time scales than situated action. In general, designs of AI agents include both deliberative and situated components. For example, the design of Sunny, our friendly robot, may contain a deliberative planner that generates plans to navigate from one location in a city to another. Note that since there are may people and other robots working or walking on the roads, Sunny's environment is dynamic in that the state of the world may change during the time Sunny takes to generate a plan. How may Sunny navigate from its apartment to its office building in this dynamic environment?

Sunny of course may use the deliberative planner to plan a path between offices. However, while the planner can produce navigation plans, it may not represent the movements of all the people and other robots on the roads. So deliberation by itself is not good enough for the dynamic urban environment. Alternatively, Sunny may use situated action (i.e., *perceive-act)* that we described in the previous section. While this may help Sunny avoid collisions with moving people - as it soon as it the nearby presence of a person, it may move away - its progress towards the goal of reaching a specific office is likely to be slow, perhaps painfully slow.

Yet another alternative is endow Sunny with the capability of both deliberative planning and situated action. In fact, this is exactly what many practical robots do. As a result, Sunny becomes capable of both long-range planning and short-range reaction. It may use its deliberative planner to come up with a plan for reaching the office building. Then, as it is executing the navigation plan, it constantly monitors the world around it and acts to avoid collisions with moving people. Next, as soon as it has moved away from a collision, it reverts to execution of its navigation plan. In this way, Sunny combines both deliberation and situated action. While this integration of deliberation and situated action has obvious benefits, it also has additional knowledge requirements as well as additional computational costs of shifting between strategies.

So far we have talked of perceiving the environment as though it were a minor task. For human beings, perception often appears to be effortless, but automating perception in AI agents has proven to be one of the many difficult problems in AI. The field of *computer vision* creates programs that take as input photos and video, and generates beliefs about objects, textures, movements, as well as higher-level features such as emotions, movement styles, and gender. *Speech recognition* is another major field in perception. The ability of computers to understand your credit card number when you speak it into the phone is the result of over 50 years of AI work. Many of the the algorithms used to understand speech and sound are shared with those of machine learning.

Likewise, achieving physical motion in the real world is difficult. *Robotics* is the field of AI that controls machines that interact directly with the physical world (as opposed to a program that, say, bought stocks electronically.) Robotics uses computational perception, machine learning, and sometimes natural language processing. Some of the major problems specific to robotics are navigation and the handling of objects. Robots can work in collaboration with each other, which overlaps with the fields of *intelligent agents* (or *agent-based AI*), which builds intelligent programs that operate through the interaction of many individual agents, and *swarm intelligence*, in which the individual agents do not have much intelligence individually.

**2.4 Deliberation and Reflection**

Above, we briefly discussed the need for both longer-range planning and shorter-range situated action in autonomous AI agents because the environment in which they reside is dynamic. However, changes in the environment themselves may unfold over different time scales. In the short term, for example, people and robots may be moving around on the roads of a Sunny's city. In the long term, roads themselves may change, new apartments and office buildings may be constructed, etc. When the latter happens, the navigation plan that Sunny's deliberative planner produces will start failing upon execution. How might Sunny adapt its knowledge of the environment as the environment changes? Alternatively, if Sunny had been designed incorrectly to begin with, how might it adapt its reasoning process?

Recent AI research on meta-reasoning is starting to design AI agents capable of self-adaptation. Such an AI agent may contain a specification of its own design. For example, the meta-reasoner in Sunny may have a specification of Sunny's design, including its functions (e.g., its goals) as well as its mechanisms for achieving the functions (e.g., the method of map-based navigation planning). When Sunny generates a plan that fails upon execution, the Sunny's meta-reasoner uses the specification of its design to diagnose and repair its reasoning process. If the feedback from the world on the failed plan pertains

to an element of knowledge (e.g., at intersection A, I expected a road going directly towards downtown but when I reached there, I found no such road), then Sunny enters this new knowledge in its map of the city. Thus, while the deliberative planner in Sunny reasons about actions in the external world, Sunny's reflective meta-reasoner reasons about its external world as well as its internal knowledge and reasoning.

### 2.5 Putting It All Together

In this section, we took navigational planning as an example to illustrate how AI is putting together multiple capabilities ranging from perception, cognition and action on one hand, to reasoning, learning and memory on the other, to reflection, deliberation and situated action on yet another. Of course, the design choices we have outlined above are exactly that: choices. For example, instead of using deliberation to mediate between reflection and situated action as described above, an AI agent may reflect directly on situated action. Different AI researchers pursue different set of choices. In a way, the enterprise of AI is to explore such design choices and examine the computational trade-offs that each choice offers.

What has emerged out of this line of work is an understanding that the design of an AI agent depends on the environment it lives in, that no one design is necessarily the best for all environments. Further, the design of an AI agent in any non-trivial environment requires multiple capabilities, and multiple methods for achieving any capability such as reasoning and learning.

## 3. A Very Brief History of Artificial Intelligence

In the middle of the twentieth century, the scientific world experienced a shift in focus from descriptions of matter and energy to descriptions of information. One manifestation of information theory applied to real-world problems was the field of *cybernetics* (Weiner, 1948, 1961), the study of communication and control in self-regulating analog systems. Cybernetics' focus on analog signal contributed to its losing ground against symbolic-based

approaches, such as AI. Not only did AI approaches come to dominate the research into the same problems, but the symbol-processing approach came to dominate cognitive psychology as well.

Search was the first major paradigm of AI. The first artificial intelligence program ever written is the Logic Theorist (Newell, Shaw & Simon 1958). Many of the problems early AI researchers focused on were, in retrospect, simple. The early exuberance of AI was tempered with the first "AI Winter" that dominated the late sixties and the 1970s, characterized by a decrease of optimism and funding, and caused by the unfulfilled expectations. Early interest in associative processing was diminished by an influential book (Minsky & Papert, 1969) around the same time.

The AI Winter of the 1970s however also witnessed the emergence of new theories and paradigms. For example, ANALOGY (Evans 1968) solved simple geometric analogy problems that appear on some intelligence tests. SHRDLU (Winograd, 1972) performed natural language processing to understand commands to a robot to pick up and manipulate blocks. Marr (1982) developed a three-stage computational theory of vision. Schank (1975) first developed a theory of conceptual structures for natural language understanding (Schank 1975) and then a theory of memory, reasoning and learning (Schank 1982).

Working in a different paradigm, Feigenbaum, Buchanan and their colleagues first developed an expert system called Dendral that could generate hypotheses about molecular structures from spectroscopic data (Lindsay et al. 1980), and then an expert system called Mycin that could generate hypotheses about E. Coli bacterial diseases from heterogeneous patient data (Buchanan & Shortliffe 1984). AI's revival in the 1980s was due in part to the success of these *expert systems* that wre designed to replicate the expertise of individuals with a great deal of domain knowledge. Knowledge engineers would interview and observe experts, and then attempt to encode their knowledge into some form that an AI program could use. This was done with a variety of methods, including *decision trees* (which can be thought of as using the answers to a series of questions to classify some input, as in the game twenty questions). Since expert systems

were of use to business, there was a renewed interest in AI and its applications. Funding for AI research increased.

One of the ideological debates of the 1980s was between the "neaties" and the "scruffies," where the former used a formal, often logic-based approach, and the latter focused on modeling human intelligence and getting AIs to use semantic information processing. Geographically, the neaties were based at Stanford University and the west coast, and in Japan, and the scruffies at MIT and the east coast. Neaties thought that knowledge representation and processing should be mathematically rigorous and elegant, and evaluations should involve proofs. Scruffies believed that intelligence is so complex that it is unwise to put such constraints on at this early stage of development of AI theory and methodology. Today, most of the engineering AI research would be classified as neat. A good deal of, but not all, contemporary cognitive AI is scruffy.

In the 1980s, interest in artificial neural networks was revived by cognitive modeling by *connectionists* (Rumelhart, McClelland & the PDP Research Group, 1986; McClelland, Rumelhart, & the PDP Research Group 1986). Connectionism continues to have influence in modern cognitive science; in engineering AI, artificial neural networks are regarded as just one of many statistical learning mechanisms (such as Markov models and other methods mentioned in the previous section.) Interestingly, some of the approaches and ideas of the cyberneticists have had a revival in these subsymbolic approaches to AI.

Over time the limits of expert systems became clear. As they grew in size, they became difficult to maintain and could not learn. As a knowledge base grows, inconsistencies between different chunks of knowledge tend to arise. In part again because of unfulfilled expectatons, in the 1990s, AI is entered a second "winter," with diminished optimism, interest, and funding. Hwoever, during the second winter, again, new frameworks appeared, including *embodied cognition*, *situated cognition* and *distributed* cognition. These frameworks emphasize how the body and environment both constrain and afford cognition, how cognition always is in the context of the physical and social worlds where

these worlds themselves afford information to the cognitive agent. Similarly, *Agent-based AI* on one hand seeks to unify cogniton with perception and action, and on the other, studies AI agents as a member of a team of other agents (artificial or human).

At present, AI appears to have entered a new phase of revival. This is in part due to the new frameworks that appeared in the 1990s, especially that of agent-based AI. By now, AI is ubiquitous in industrialized societies, though it often does not go by that name. Many researchers avoid the term, feeling that it has been tarnished by the boom-and-bust cycle of interest and funding it has enjoyed in its 50 year history. However, techniques from AI are used in many practical applications, including understanding your voice when you talk to your credit card company, making recommendations for books when you shop online, the landing systems of our airplanes, the path-finding of characters in computer games, generating web search engine results, face detection in cameras and online photo archives, and automatic translation.

**4. Measuring the Intelligence of AIs**

When measuring the intelligence of human beings, the test need not have questions representing every kind of intelligent thing a person could do. Rather, the test result is intended to measure the general intelligence of the taker (Wechsler 1939; Raven 1962). Where one form of intelligence (e.g., mathematical) does not predict another (e.g., verbal), two tests are required.

In artificial intelligence, the problem is much bigger. Since AIs are designed by people, they have enormous variety, depending on the goals of the researcher creating them. As such, an AI that scores well on the SAT verbal section, like Latent Semantic Analysis (Landauer, 1998), will be likely to not only score poorly when tested on other cognitive tasks, but will likely not be able to take them at all. In short, performance on any given human IQ test will predict general intelligence in an AI even more poorly than it does in human beings. Depending on how the AI is built, it can have unique combinations of sensors, actuators, and ways to think. Not only are they often completely different from

each other, but they are also often very alien to our own experiences as human beings.

A further problem is that AIs tend to be computer programs that run on computers, which vary in speed. The same program running on a faster computer will be much more effective, and in any timed test this will make an enormous difference. It is a philosophical question whether or not the computer's speed should affect how we regard the intelligence of the AI. The chess playing programs of the early days of AI did not fail because of their bad algorithms; the computers they ran on were too slow to make those algorithms effective. Current chess champion AIs, such as Hydra (Donninger & Lorentz, 2004) are run on normal commercial PCs, rather than the special -purpose hardware required with the Deep Blue project that defeated Kasparov (Hsu, Campbell, & Hoane, 1995).

In the past, certain tasks, such as using memory and speed of calculation, were thought to be excellent examples of intelligence, and even modern tests often measure these things. These tasks are very easy for computer programs, but, for whatever reason, we are reluctant to attribute high intelligence to computer programs for being able to do them. Even chess can largely be played well using "brute-force" search methods (Hsu et al., 1995). Algorithms that don't work well today might work just fine on faster computers of the future. Note also, however, that if we were to find a human who could evaluate moves like a computer could, we would regard her as very intelligent indeed, at least in her own way.

AI researchers usually evaluate their programs with idiosyncratic methodology appropriate to the task. Though these are not thought of as intelligence tests, they could be thought of as specialized intelligence tests, just as there are sometimes special tests for certain sub-populations of human beings, such as children (Legg & Hutter, 2007). In contrast, PERI (Bringsjord, Selmer, Schimanski, & Bettina, 2003) is an AI project with the explicit goal of passing intelligence tests. As of 2003, it performed well on block design problems in the WAIS intelligence test (Wechsler 1939). Even if we could think of a single test for all AIs, the variance in their scores would be enormous in comparison to

people, for whom the IQ of an individual can usefully be scored relative to a large group (Legg & Hutter, 2007).

The most famous proposed test for AI is the the "imitation game," or, as it is more popularly called, the *Turing Test* (Turing, 1950). In this test, computers and human beings are put in typed chat sessions with human judges. If computers can reliably fool the judges into thinking they are human, then they pass the test. Turing formulated this test in response to the question "Can machines think?" Rather than answering that question, he reformulated it into a more concrete question of whether or not a machine could fool a human interrogator. Though Turing played it a bit safe, most interpretations of him do not, interpreting the purpose of the test to be to distinguish programs that have human-level intelligence from those that do not (e.g., Harnad, 1992). In this interpretation, the test is not a measurement of intelligence in the sense of giving a score that accurately reflects cognitive abilities, but as a pass-or-fail litmus test of general intelligence.

It has proven to be a very difficult test to pass, although some surprisingly simple programs, such as ELIZA (Weizenbaum, 1966) and PARRY (Raphael 1976), sometimes fool some people for short times. Because of this difficulty, competitions usually restrict judges to specific topics, as the general-topic version is impossible for state-of-the-art AI to pass. Some programs can pass the restricted test (according to Turing's suggested numbers), but they appear to do so at least in part because of aspects that are not relevant to intelligence, such as demonstrating typing errors (Johnson, 1992). Recently there even have been Turing test competetions and prizes (e.g., http://www.loebner.net/Prizef/loebner-prize.html).

## 4. Conclusion

In this chapter we have reviewed the history of AI and its major subfields, illustrated AI as a science and as a technology, and discussed the problems of the measurement of intelligence in AIs. The field has made so much progress

that now every year the Association for Advancement of Artificial Intelligence (http://www.aaai.org/home.html) organizes a conference for deployed AI applications (called Innovative Applications of Artificial Intelligence, http://www.aaai.org/Conferences/IAAI/iaai10.php).

Of course, we have not tried to cover each and every topic in AI. For example, over the last decade, there has been a lot of AI research on designing the *semantic web* (Berners-Lee, Hendler & Lassial 2001), a new version of the world wide web that would be capable of understanding information (e.g. webpages) stored on it. An another example, just over the last few years, *interactive games* have emerged as an important arena for AI research, especially agent-based AI. Nor, in this article, have we attended to AI ethics, which is becoming an increasingly important issue.

An important, and somewhat surprising, lesson from the history of AI is that cognitive tasks that seem difficult for humans to solve (e.g., mathematical, logical, and chess problems) are relatively easy to make programs solve, and those cognitive tasks that are apparently easy for humans to address (e.g., walking, talking, and perceiving) are extraordinarily difficult to make computers solve. This apparent paradox has resulted in repeated predictions about bold AI successes, only to see them go unfulfilled.

We suggest there may be two reasons for this paradox. First, our difficult problems require deliberate thought and strategies that are explicitly learned. As a result, we can often gain insight into how they are solved through introspection. Indeed, many of these strategies are actually written down, to be learned through reading. In contrast, nobody needs to tell human beings how to see, walk, or speak. As a result, our intuitions about how these processes work are, to put it mildly, unhelpful.

The second, perhaps more important, reason is that deliberate processing is likely a serial process running as a virtual machine on a network of neurons, whereas the automatic processes, the easy tasks, are running directly on the neural network. These easy tasks (called System 1 in Stanovich & West, 2003) are evolutionarily older, and the parts of our brains that accomplish them

(generally near the back of the head, Anderson, 2007) evolved to do just those things. In contrast, the more deliberate processing is evolutionarily younger, and makes use of the kind of hardware designed for System 1 tasks. System 2 struggles to do rational, serial processing on an essentially parallel pattern-matching machine (Stanovich, 2004). In another chapter in this volume, Kauffman provides a review of such dual-process theories.

Computers, and the languages we program them with, are naturally serial processors. When we implement artificial neural networks, we are doing it backward from nature: Whereas System 2 is a serial virtual machine running on parallel hardware, our artificial neural networks are parallel virtual machines running on serial hardware. Given this and the fact that we have no conscious access to System 1 processes, it is no wonder the AI community has had to work very hard to make progress in these areas. As a result, we have chess programs that can beat world grandmasters, but no robots that can walk down a street even as well as a five-year-old child. We expect that neuroscience findings may illuminate the nature of these processes, and the AI community will be able to build on them.

Given the track record of predictions about the future of AI, we will refrain from making our own (see Kurzweil 2005 for one possible future). What we can and will claim is that AI already has had a profound impact not only in computer science and information technology, but also more generally on our culture and our philosophy. If the last fifty year history of AI is any guide, then the next fifty years will not only be full of exciting discoveries and bold inventions, but they will also raise new questions about who we are as humans and what we want to be.

**Acknowledgements**

We thank the editors of this volume and members of the Design & Intelligence Laboratory at Georgia Tech for their comments on earlier drafts of this article. During the writing of this article, Goel's has been partially supported by NSF grants (#0632519, Learning About Complex Systems in Middle School by Constructing Structure-Behavior-Function Models; #0613744, Teleological Reasoning in Adaptive Software Design; and

#0855916, Computational Tools for Enhancing Creativity in Biologically Inspired Engineering Design).

**References**

Ali, K. & Goel, A. (1996) Combining navigational planning and reactive control. *Proceedings of the AAAI-96 Workshop on Reasoning About Actions, Planning and Control: Bridging the Gap* (pp. 1-7). Portland.

Albus, J.S. (1991). Outline for a theory of intelligence. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3).

Anderson, J. R. & Lebiere, C. (1998). *The Atomic Components of Thought*. Mahwah, NJ: Lawrence Erlbaum Associates.

Anderson, M.L. (2007). Massive redeployment, exaptation, and the functional integration of cognitive operations. *Synthese*, 159(3), 329-345.

Arkin, R. (1999) *Behavior-Based Robotics*. Cambridge, MA: MIT Press.

Berners-Lee, T., Hendler, J., & Lassila, O (2001) Semactic Web. *Scientific American.*

Bringsjord, S. (1998) Chess is too easy. *Technology Review*, 101(2), 23-28.

Bringsjord, S., & Schimanski, B. (2003). What is artificial intelligence? Psychometric AI as an answer. *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI–03)* (pp. 887-893) San Francisco, CA: Morgan Kaufmann.

Buchanan, B., & Shortliffe, E. (1984) Rule Based Expert Systems: The Mycin Experiments of the Stanford Heuristic Programming Project. Boston, MA: Addison-Wesley.

Donninger, C., & Lorenz, U. (2004). The chess monster hydra. *Proceedings of the 14th International Conference on Field-Programmable Logic and Applications (FPL)* (pp. 927-932). Antwerp, Belgium: LNCS 3203.

Evans, T. G. (1968). A Program for the solution of a class of geometric-analogy intelligence-test questions. In M. Minsky (Ed.), *Semantic Information Processing* (pp. 271-353). Cambridge, MA: MIT Press.

Glasgow, J., Narayanan, N.H., & Chandrasekaran, B. (Eds.). (1995). *Diagrammatic Reasoning: Cognitive and Computational Perspectives*. Cambridge, MA: MIT Press.

Goel, A., Ali, K., Donnellan, M., Gomez, A. & Callantine, T. (1994). Multistrategy adaptive navigational path planning. *IEEE Expert*, 9(6), 57-65.

Goel, A., Stroulia, E., Chen, Z & Rowland, P. (1997). Model-based reconfiguration of schema-based reactive control architectures. *Proceedings of the AAAI Fall Symposium on Model-Based Autonomy*.

Harnad, S. (1992) The Turing Test is not a trick: Turing indistinguishability is a scientific criterion. *SIGART Bulletin* 3(4) 9–10.

Hsu, F.H., Campbell, M.S., & Hoane, A.J. (1995) Deep Blue System Overview. *In Proceedings of the 1995 International Conference on Supercomputing* (pp. 240–244).

Johnson, W.L. (1992). Needed: A new test of intelligence. *SIGARTN: SIGART Newsletter* 3(4), 7-9.

Kauffman, S.B. (2010) *This volume.*

Kolodner, J. (1993) *Case-Based Reasoning*. San Francisco, CA: Morgan Kaufmann.

Kurzweil, R. (2005) *The Singularity is Near*: *When Humans Transcend Biology*. New York, NY: Viking Adult.

Laird, J., Newell, A., & Rosenbloom, P. (1987) Soar: An Architecture for General Intelligence. *Artificial Intelligence*, 33: 1-64.

Landauer, T. K. (1998). Learning and representing verbal meaning: The latent semantic analysis theory. *Current Directions in Psychological Science*. 7(5), 161-164.

Legg, S., & Hutter, M. (2007). Universal intelligence: A definition of machine intelligence. *Minds & Machines*. 17(4), 391–444.

Lindsay, R., Buchanan, B., Feigenbaum, E., & Lederberg, J. (1980) Applications of Artificial Intelligence for Chemical Inference: The Dendral Project. New York: McGraw-Hill.

Lenat, D., & Guha, R. (1990) *Building Large Knowledge Based Systems: Representation and Inference in the Cyc Project*. Boston, MA: Addison-Wesley Longman Publishing.

Marr, D. (1982) *Vision.* New York: Henry Holt.

McCarthy, J. (1988) Mathematical Logic in AI. *Daedalus* 117(1): 297-311.

McClelland, J.L., Rumelhart, D.E., & PDP Research Group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 2: Psychological and Biological Models*. Cambridge, MA: MIT Press.

Minsky, M. L., & Papert S.A. (1969). *Perceptrons*. Cambridge, MA: MIT Press.

Minsky, M. L. (1975) *A Framework for Representing Knowledge*, in: Patrick Henry Winston (ed.), The Psychology of Computer Vision. McGraw-Hill, New York (U.S.A.), 1975.

Mitchell, M. (1998) *An Introduction to Genetic Algorithms.* Cambridge: MIT Press.

Newell, A., Shaw, J.C.; Simon, H.A. (1958) Elements of a Theory of Problem Solving. *Psychological Review* 63(3): 151-166.

Newell, A., & Simon, H.A. (1972) *Human Probem Solving*. Prentice-Hall.

Pearl, J. (2000) *Causality: Models, Reasoning and Inference.* New York: Cambridge University Press.

Rabiner, L., & Juang, B.H. (1986) An Introduction to Hidden Markov Models. IEEE AASP Magazine, pp. 4-16.

Raphael, B. (1976) *The Thinking Computer.* New York: W.H. Freeman.

Raven, J.C. (1962). *Advanced Progressive Matrices Set II*. London, United Kingdom: H.K. Lewis.

Rumelhart, D.E., McClelland, J.L., & PDP Research Group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*. Cambridge, MA: MIT Press.

Schank, R.C. (1975). *Conceptual Information Processing.* New York: Elsevier.

Schank, R.C. (1982). *Dynamic Memory.* Cambridge University Press 2nd Edition.

Simon, H.A. (1969) *Sciences of the Artificial.* Cambridge, MA: MIT Press.

Singh, P., Lin T., Mueller, E.T., Lim, G., Perkins, T., & Zhu, W.L. (2002). Open mind common sense: Knowledge acquisition from the general public. *Proceedings of the First International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems*. (pp. 1223-1237).

Sowa, J. (1987) Semantic Networks. In *Encylopedia of AI,* S. Shapiro (editor), New York: Wiley, pp: 1011-1024.

Stanovich, K. E. (2004). *The Robot's Rebellion*. Chicago, London: The University of Chicago Press.

Stanovich, K.E. & West, R.F (2003). The rationality debate as a progressive research program. *Behavioral and Brain Sciences,* 26 (4), 531-533.

Stroulia, E & Goel, A.K. (1999). Evaluating problem-solving methods in evolutionary design: The autognsotic experiments. *International Journal of Human-Computer Studies*, 51, 825-847.

Sutton, R. S. & Barto, A. (1998) *Reinforcement Learning: An Introduction.* Cambridge: MIT Press.

Turing, A.M. (1950) Computing machinery and intelligence. *Mind*, 59, 433-460.

Von Anh, L., Liu, R., & Blum, M. (2006). Peekaboom: a game for locating objects in images. *Proceedings of the SIGCHI conference on Human Factors in computing systems* (pp. 55–64).

Wechsler, David (1939). *The measurement of adult intelligence*. Baltimore: Williams & Wilkins.

Weiner, N. (1948, 1961). *Cybernetics*. First and Second editions. MIT Press: Cambridge.

Weizenbaum, J. (1966). ELIZA - A Computer Program For the Study of Natural Language Communication Between Man And Machine. *Communications of the ACM* **9** (1): 36–45

Winograd, T. (1972). *Understanding Natural Language*. Academic Press.