A reprint from

American Scientist
the magazine of Sigma Xi, The Scientific Research Society

The Manifest Destiny of Artificial Intelligence

Brian Hayes

Artificial intelligence began with an ambitious research agenda: To endow machines with some of the traits we value most highly in ourselves—the faculty of reason, skill in solving problems, creativity, the capacity to learn from experience. Early results were promising. Computers were programmed to play checkers and chess, to prove theorems in geometry, to solve analogy puzzles from IQ tests, to recognize letters of the alphabet. Marvin Minsky, one of the pioneers, declared in 1961: "We are on the threshold of an era that will be strongly influenced, and quite possibly dominated, by intelligent problem-solving machines."

Fifty years later, problem-solving machines are a familiar presence in daily life. Computer programs suggest the best route through cross-town traffic, recommend movies you might like to see, recognize faces in photographs, transcribe your voicemail messages and translate documents from one language to another. As for checkers and chess, computers are not merely good players; they are unbeatable. Even on the television quiz show *Jeopardy*, the best human contestants were trounced by a computer.

In spite of these achievements, the status of artificial intelligence remains unsettled. We have many clever gadgets, but it's not at all clear they add up to a "thinking machine." Their methods and inner mechanisms seem nothing like human mental processes. Perhaps we should not be bragging about how smart our machines have become; rather, we should marvel at how much those machines accomplish without any genuine intelligence.

Brian Hayes is senior writer for American Scientist. *Additional material related to the* Computing Science *column appears at http://bit-player.org. Address: 11 Chandler St. #2, Somerville, MA 02144. E-mail: brian@bit-player.org*

Will AI create mindlike machines, or will it show how much a mindless machine can do?

It is not only critics from outside the field who express such qualms about the direction of AI. Fifteen years ago Minsky told an interviewer: "The bottom line is that we really haven't progressed too far toward a truly intelligent machine. We have collections of dumb specialists in small domains; the true majesty of general intelligence still awaits our attack." At a recent mathematics meeting I heard Minsky offer a similar assessment, lamenting the neglect of the field's deepest long-range goals. His comments prompted me to look back at the early literature of artificial intelligence, and then survey some of the recent accomplishments. Has AI strayed from the true path, or has it found a better way forward?

Neats Versus Scruffies

At the outset, research in artificial intelligence was the project of a very small community. An inaugural conference in 1956 had just 10 participants. They included Allen Newell and Herbert A. Simon of Carnegie Tech (now Carnegie Mellon University); Minsky, who had just begun his career at MIT; and John McCarthy, who left MIT to start a new laboratory at Stanford. A major share of the early work in AI was done by these four individuals and their students.

It was a small community, but big enough for schisms and factional strife. One early conflict pitted "the neats" against "the scruffies." The neats emphasized the role of deductive logic; the scruffies embraced other modes of problem-solving, such as analogy, metaphor and reasoning from example. McCarthy was a neat, Minsky a scruffy.

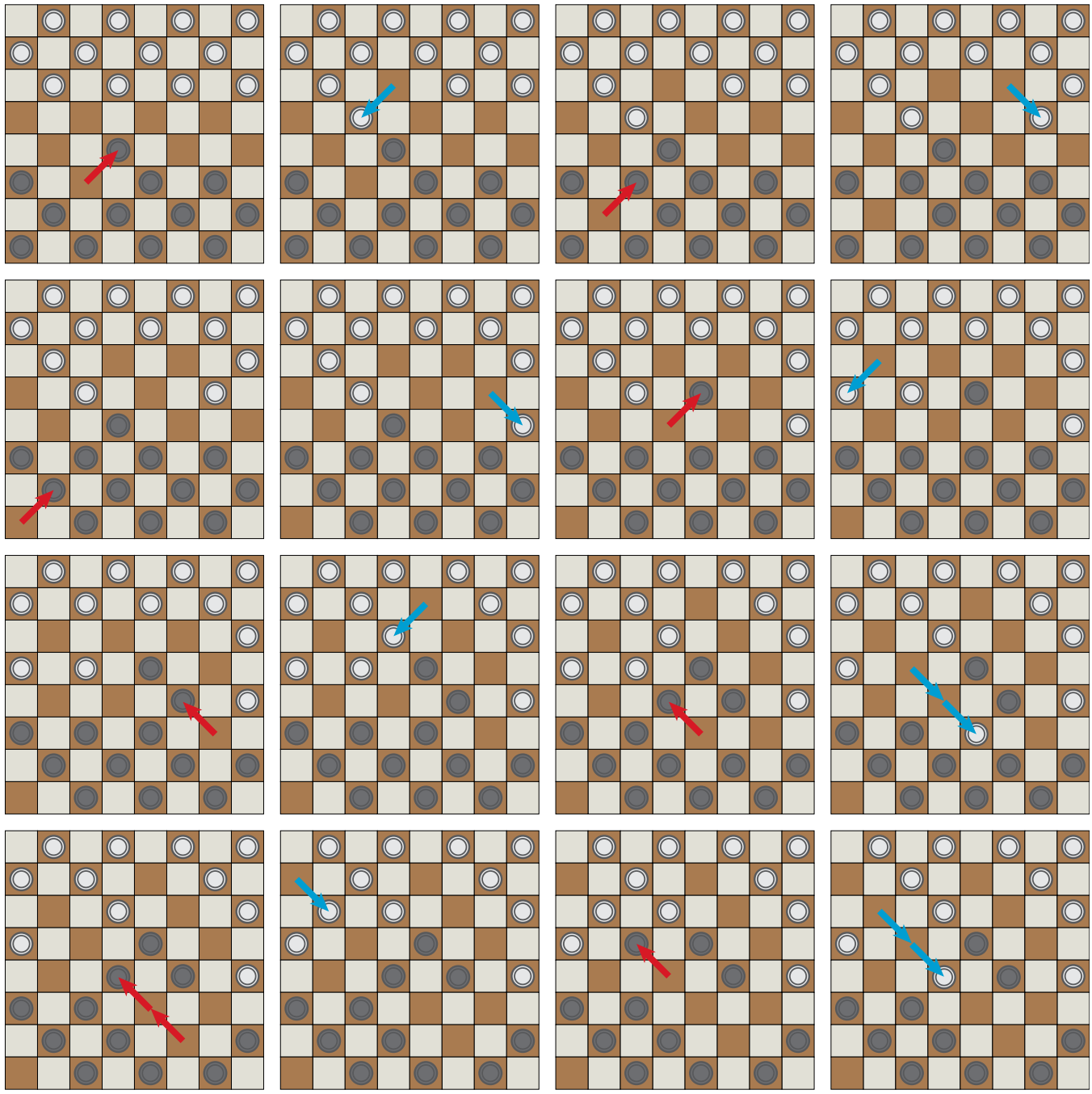
An even older and deeper rift divides the "symbolic" and the "connectionist" approaches to artificial intelligence. Are the basic atoms of thought ideas, propositions and other such abstractions? Or is thought something that emerges from patterns of activity in neural networks? In other words, is the proper object of study the mind or the brain?

If an artificial intelligence needs a brain, maybe it also needs a body, with sensors that connect it to the physical world; thus AI becomes a branch of robotics. Still another faction argues that if a machine is to think, it must have something to think about, and so the first priority is encoding knowledge in computer-digestible form.

A backdrop to all of these diverging views within the AI community is a long-running philosophical debate over whether artificial intelligence is possible at all. Some skeptics hold that human thought is inherently nonalgorithmic, and so a deterministic machine cannot reproduce everything that happens in the brain. (It's the kind of dispute that ends not with the resolution of the issue but with the exhaustion of the participants.)

Through the 1970s, most AI projects were small, proof-of-concept studies. The scale of the enterprise began to change in the 1980s with the popularity of "expert systems," which applied AI principles to narrow domains such as medical diagnosis or mineral prospecting. This brief flurry of entrepreneurial activity was followed by a longer period of retrenchment known as "the AI winter."

The present era must be the AI spring, for there has been an extraordinary revival of interest. Last year Pe-



A game of checkers played 50 years ago (July 12, 1962) pitted Robert Nealey, a former champion of Connecticut, against a computer program written by Arthur L. Samuel of IBM. Nealey played the white pieces. The first eight moves for each side are shown here; after the computer's 27th move, Nealey resigned. Samuel's program was designed to improve its strategy by learning from experience, much as a human player would. Later programs achieved stronger performance by relying instead on the computer's capacity to sift through billions of candidate moves.

ter Norvig and Sebastian Thrun were teaching an introductory AI course at Stanford and opened it to free enrollment over the Internet. They attracted 160,000 online students (though "only" 23,000 successfully completed the course). The revival comes with a new computational toolkit and a new attitude: Intelligent machines are no longer just a dream for the future but a practical technology we can exploit here and now. I'm going to illustrate these changes with three examples of AI then and now: machines that play games (in particular checkers), machines that translate languages and machines that answer questions.

Gaming the System

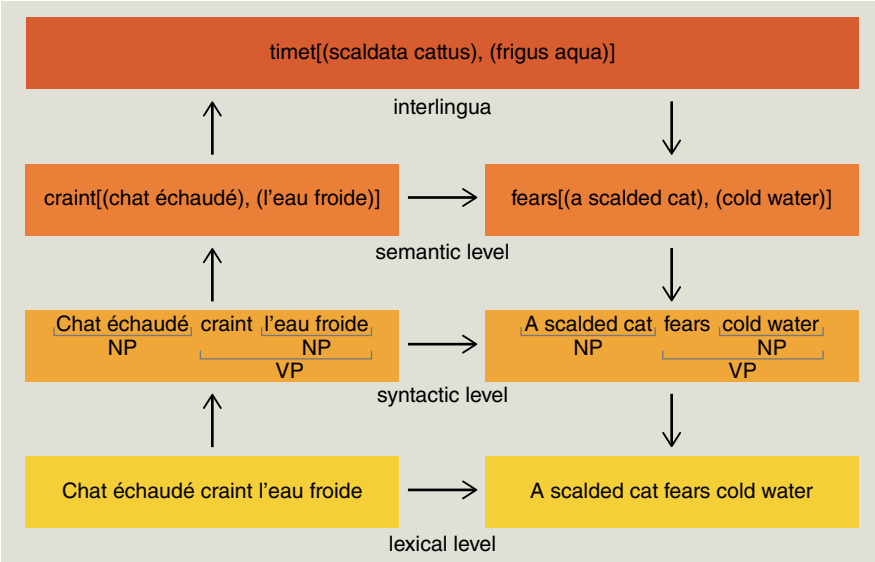
The game of checkers was the subject of one of the earliest success stories in AI. Arthur L. Samuel of IBM started work on a checkers-playing program in the early 1950s and returned to the project several times over the next 20 years. The program was noteworthy not only for playing reasonably well—quite early on, it began beating its creator—but also for learning the game in much the same way that people do. It played against various opponents (including itself!) and drew lessons from its own wins and losses.

Samuel explained the program's operation in terms of goals and subgoals. The overall goal was to reach a winning position, where the opponent has no legal move. The program identified subgoals that would mark progress toward the goal. Experienced players pointed out that the program's main weakness was the lack of any sustained strategy or "deep objective."

The subsequent history of computer checkers is dominated by the work of Jonathan Schaeffer and his colleagues at the University of Alberta. In 1989 they began work on a program called Chinook, which quickly became a player of world-champion caliber. It played twice against Marion Tinsley, who was the preeminent human checkers player of the era. (Tinsley won every tournament he entered from 1950 to 1994.) In a 1992 match, Tinsley defeated Chinook 4–2 with 33 draws. A rematch two years later ended prematurely when Tinsley withdrew because of illness. The six games played up to that point had all been draws, and Chinook became champion by forfeit. Tinsley died a few months later, so there was never a final showdown over the board.

Chinook's approach to the game was quite unlike that of Samuel's earlier program. There was no hierarchy of goals and subgoals, and no attempt to imitate the strategic thinking of human players. Chinook's strength lay entirely in capacious memory and rapid computation. At the time of the second Tinsley match, the program was searching sequences of moves to a minimum depth of 19 plies. (A ply is a move by one side or the other.) Chinook had a library of 60,000 opening positions and an endgame database with precomputed outcomes for every position with eight or fewer pieces on the board. There are 443,748,401,247 such positions.

More recently, Schaeffer and his colleagues have gone on from creating strong checkers players to solving the game altogether. After a series of computations that ended in 2007, they declared that checkers is "weakly solved." The weak solution identifies a provably optimal line of play from the starting position to the end—which turns out to be a draw. Neither player can improve his or her (or its) outcome by departing from this canonical sequence of moves. (A "strong" solution would give the correct line of play from *any* legally reachable board position.) By the time this proof was completed, the endgame database encompassed all positions with 10 or fewer pieces (almost 40 trillion of them).



Machine translation has improved with the adoption of statistical methods that extract word patterns from large collections of bilingual texts. Earlier programs, imitating the methods of human translators, performed a multilevel analysis, looking not just at the words themselves (the lexical level) but also at syntactic structures such as noun phrases (NP) and verb phrases (VP) and at semantics, or meanings. At the top of the hierarchy is the *interlingua*, which is meant to be a language-independent representation of meaning. The statistical approach takes a shortcut directly from the lexical level of the source language to that of the target language. The French-to-English translation shown here was generated by just such a program, that of the Google Translate service. (So was the Latin used as a stand-in for an interlingua.)

Schaeffer notes that his checkers-playing program doesn't need to know much about checkers:

> Perhaps the biggest contribution of applying AI technology to developing game-playing programs was the realization that a search-intensive ("brute-force") approach could produce high-quality performance using minimal application-dependent knowledge.

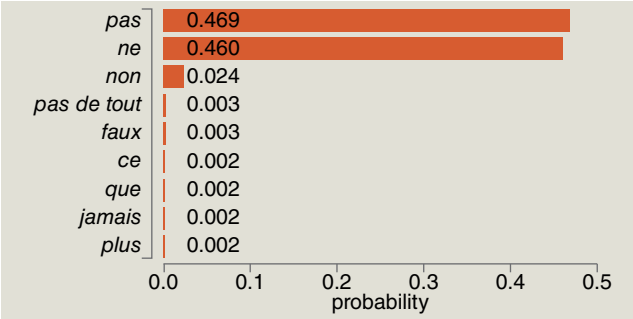
There is room here for a devil's advocate to offer a counterargument. Winning isn't everything, and playing the game without understanding it is not the most obvious route to wisdom. When Samuel began his work on checkers, his aim was not just to create an invincible opponent but to learn something about how people play games—indeed, to learn something about learning itself. Progress toward these broader goals could have influence beyond the world of board games.

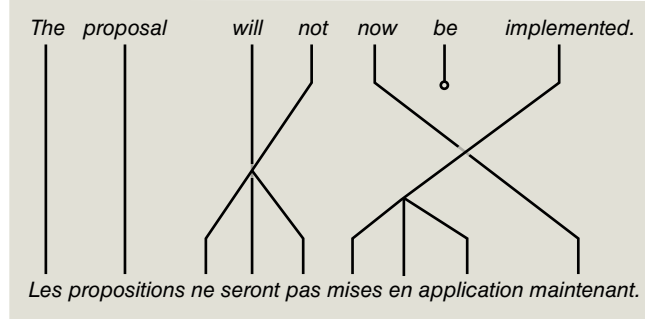
But this position is difficult to defend. It turns out that brute-force methods like those of Chinook have been highly productive in a variety of other areas. They are not just tricks for winning games; Schaeffer cites bioinformatics and optimization among other application areas. The anthropocentric scheme, taking human thought patterns as the model for computer programs, has been less fruitful so far.

The Vodka Is Strong

In *The Hitchhiker's Guide to the Galaxy*, Douglas Adams introduces the Babel Fish: "If you stick one in your ear, you can instantly understand anything said to you in any form of language." Here in our little corner of the galaxy, Babel Fish is the name of a web service (part of Yahoo) that also performs translation, though it's limited to languages from Planet Earth. Google and Microsoft offer similar services. Depending on your needs and expectations, the quality of the results can seem either amazing or risible.

Efforts to build a translation machine were already under way in the 1950s. The simplest of the early schemes was essentially an automated bilingual dictionary: The machine would read each word in the source text, look it up in the dictionary, and return the corresponding word or words in the target language. The failure of this approach is sometimes dramatized with the tale of





Candidate French translations of the English word *not* were identified by a program that has no understanding of either language; the list *(left panel)* was generated simply by searching for words that appear with elevated frequency in French translations of English sentences that include the word *not*. The most likely candidates are *ne* and *pas*, which together form the most common French negation. The final step in a statistical translation algorithm is an alignment *(right panel)* that maps words of the source sentence to those of the target. Both of the examples shown here are adapted from a 1990 article by Peter F. Brown and his colleagues, using a bilingual corpus of Canadian parliamentary proceedings.

the English→Russian→English translation that began with "The spirit is willing but the flesh is weak" and ended with "The vodka is strong but the meat is rotten." John Hutchins, in a history of machine translation, thoroughly debunks that story, but the fact remains that word-by-word dictionary lookup was eventually dismissed as useless.

Later programs worked with higher-level linguistic structures—phrases and sentences rather than individual words. In the early 1970s Yorick Wilks, who was then at Stanford, built an English-to-French translation program that explicitly tried to reproduce some of the mental processes of a human translator. The program would read a sentence, break it into component phrases, try to assign meanings to the words based on their local context, and then generate corresponding phrases in the target language.

Wilks's project never got beyond the prototype stage, but another translation system with roots in the same era is still in wide use and under active development. SYSTRAN, founded in 1968 by Peter Toma, now powers the Babel Fish service. It began as a dictionary-based system but now has a complex structure with many modules.

In recent years a quite different approach to machine translation has attracted more interest and enthusiasm. The idea is to ignore the entire hierarchy of syntactic and semantic structures—the nouns and verbs, the subjects and predicates, even the definitions of words—and simply tabulate correlations between words in a large collection of bilingual texts. The early work on this statistical approach to translation was done by Peter F. Brown and his colleagues at IBM, who had already applied analogous ideas to problems of speech recognition. The method is now the basis of translation services from both Google and Microsoft.

Deliberately ignoring everything we know about grammar and meaning would seem to be a step backward. However, all the information encoded in grammar rules and dictionary definitions is implicitly present in a large collection of texts; after all, that's where the grammarians and the lexicographers get it from in the first place. Where do the bilingual texts come from? Government bodies that publish official documents in two or more languages, such as Canada and the European Union, have been an important source.

Suppose you have a large corpus of parallel documents in French and English, broken down into pairs of matched sentences. With this resource in hand, you are asked to provide an English translation of the French proverb *Chat échaudé craint l'eau froide*. Here is one way to begin: Take each word of the proverb, find all the French sentences in the corpus that include this word, retrieve the corresponding English sentences, and look for words that appear in these sentences with unusually high frequency. In some cases the outcome of this process will be easy to interpret. If a French sentence includes the word *chat*, the English version is very likely to mention *cat*. Other cases could be equivocal. The French *craint* might be strongly correlated with several English words, such as *fears*, *dreads* and *afraid*. And occasionally it might happen that no word stands out clearly. By taking all the English words identified in this way, and perhaps applying a threshold rule of some kind, you can come up with a list of words that have a good chance of appearing in the translation.

Now the task is to put the selected words in order, and thus make an English sentence out of them. This too can be done by a probabilistic process, guided by the relative frequencies of short sequences of words (*n*-grams) in English text. The likeliest arrangement of the words is taken as the translation.

In practice, statistical translation programs are not quite as crude and simple-minded as the algorithm presented here. In particular, the ordering of the English words is done by an alignment process that starts with the French sequence and allows for insertions, deletions and transpositions. Still, the entire translation is done in total ignorance of meaning and grammatical structure. It seems a bit of a miracle when something sensible comes out. For *Chat échaudé craint l'eau froide,* Google Translate suggests: *A scalded cat fears cold water*. My high school French teacher would have given full credit for that answer.

This numerical or statistical approach to translation seems utterly alien to the human experience of language. As in the case of game-playing, I am tempted to protest that the computer has solved the problem but missed the point. Surely a human translator works at a higher level, seeking not just statistical correlations but an equivalence of meaning and perhaps also of mood and tone. In the case at hand, the artful translator might render a proverb with another proverb: *Once burned, twice shy*. That kind of deft linkage between languages seems beyond the reach of programs that merely shuffle symbols.

But are human readers really so different from the computer plodding through its database of sentences? How do people learn all those nuanced meanings of thousands of words and the elaborate rules for putting them together in well-formed sentences? We don't do it by consulting dictionaries and grammar books. Starting as young children, we absorb this knowledge from exposure to language itself; we listen and talk, then later we read and write. For the lucky polyglots among us, these activities go on in multiple languages at once, with fluid interchange between them. In other words, we infer syntax and semantics from a corpus of texts, just as the statistical translator does. Most likely the mental process underlying the acquisition of language involves no explicit calculation of probabilities of *n*-grams, but it also requires no dictionary definitions or memorized conjugations of verbs. In spirit at least, our experience of language seems closer to statistical inference than to rule-based deduction.

Q and A

My final example comes from an area of AI where algorithmic ingenuity and high-performance computing have yet to triumph fully. The task is to answer questions formulated in ordinary language.

In a sense, we already have an extraordinary question-answering technology: Web search engines such as Google and Bing put the world at our fingertips. For the most part, however, search engines don't actually answer questions; they provide pointers to documents that might or might not supply an answer. To put it another way, search engines are equipped to answer only one type of question: "Which documents on the Web mention *X*?," where *X* is the set of keywords you type into the search box. The questions people really want to ask are much more varied.

One early experiment in question answering was a program called Baseball, written at MIT circa 1960 by Bert F. Green Jr., and three colleagues. The program was able to understand and answer questions such as "Who did the Red Sox lose to on July 5, 1960?" This was an impressive feat at the time, but the domain of discourse was very small (a single season of professional baseball games) and the form of the queries was also highly constrained. You couldn't ask, for example, "Which team won the most games?"

For a glimpse of current research on question answering we can turn to the splendidly named KnowItAll project of Oren Etzioni and his colleagues at the University of Washington. Several programs written by the Etzioni group address questions of the "Who did what to whom?" variety, extracting answers from a large collection of texts (including a snapshot of the web supplied by Google). There's a demo at openie.cs.washington.edu. Instead of matching simple keywords, the KnowItAll programs employ a template of the form $X \sim Y$, where X and Y are generally noun phrases of some kind and "~" is a relation between them, as in "John loves Mary." If you leave one element of the template blank, the system attempts to fill in all appropriate values from the database of texts. For example, the query "___ defeated the Red Sox" elicits a list of 59 entries. (But "___ defeated the Red Sox on July 5, 1960" comes up empty.)

KnowItAll is still a research project, but a few other question-answering systems have been released into the wild. True Knowledge parses natural-language queries and tries to find answers in a hand-crafted semantic database. Ask.com combines question answering with conventional keyword Web searching. Apple offers the Siri service on the latest iPhone. Wolfram Alpha specializes in quantitative and mathematical subjects. I have tried all of these services except Siri; on the whole, unfortunately, the experience has been more frustrating than satisfying.

A bright spot on the question-answering horizon is Watson, the system created by David Ferrucci and a team from IBM and Carnegie Mellon to compete on *Jeopardy*. The winning performance was dazzling. On the other hand, even after reading Ferrucci's explanation of Watson's inner architecture, I don't really understand how it works. In particular I don't know how much of its success came from semantic analysis and how much from shallower keyword matching or statistical techniques. When Watson responded correctly to the clue "Even a broken one of these on your wall is right twice a day," was it reasoning about the properties of 12-hour clocks in a 24-hour world? Or did it stumble upon the phrase "right twice a day" in a list of riddles that amuse eight-year-olds?

Writing in *Nature* last year, Etzioni remarked, "The main obstacle to the paradigm shift from information retrieval to question answering seems to be a curious lack of ambition and imagination." I disagree. I think the main obstacle is that keyword search, though roundabout and imprecise, has proved to be a remarkably effective way to discover stuff. In the case of my baseball question, Google led me straight to the answer: On July 5, 1960, the Red Sox lost to the Orioles, 9 to 4. Once again, shallow methods that look only at the superficial structure of a problem seem to be outperforming deeper analysis.

Applied Computer Science

Edward A. Feigenbaum, a veteran of AI's first-generation, has declared that "computational intelligence is the manifest destiny of computer science." The slogan "manifest destiny" once expressed the sea-to-shining-sea territorial ambitions of the young United States. Feigenbaum, by analogy, is telling us there's no stopping AI until it reaches the level of human intelligence. (And then why stop there?)

Feigenbaum's declaration reiterates Minsky's prophecy from 50 years ago. They may both be proved right, one of these days. In the meantime, I see a different kind of territorial aggrandizement going on. AI is expanding into turf that once belonged to other specialities. It looks like the destiny of artificial intelligence may be to assimilate all the rest of applied computer science.

I came to appreciate how much the field has broadened as I was reading *Artificial Intelligence: A Modern Approach*, a recent textbook by Stuart Russell and Norvig. (The third edition came out in 2010.) It's a splendid book, and I recommend it not just for students of AI but for anyone seriously interested in computer science. And that's the point: Many of the ideas and methods introduced here would be quite at home in a text on algorithm design or optimization theory. Some of the more traditional AI themes are scarcely mentioned.

Russell and Norvig give a brief history of AI, where recent developments are introduced under the rubric "AI adopts the scientific method." This characterization seems a bit heavy-handed. Are we to conclude that previous generations were *un-*

scientific, toiling in benighted pursuit of cognitive phlogiston? The book's subtitle, "A Modern Approach," reinforces this impression. I'm sure the intent is not to be dismissive or scornful; it's just that the questions that animated AI research in its first decades no longer seem so urgent or central. But that's *not* because those questions have all been answered.

Meanwhile, the brash new style of AI plunges ahead. The roaring success of all those "shallow" methods—such as treating natural language as a sequence of *n*-grams—is something I find both exciting and perplexing. Exciting because these are algorithms we can implement and programs we can run; AI becomes a technology rather than a daydream. Perplexing because the shallow methods weren't supposed to work, and now we have to explain their unreasonable effectiveness. Perhaps this is how we'll get back to those deeper questions that Minsky warns we are neglecting.

Bibliography

Brown, P. F., et al. 1990. A statistical approach to machine translation. *Computational Linguistics* 169(2):79–85.

Etzioni, O. 2011. Search needs a shake-up. *Nature* 476:25–26.

Feigenbaum, E. A. 2003. Some challenges and grand challenges for computational intelligence. *Journal of the ACM* 50:32–40.

Feigenbaum, E. A., and J. Feldman. 1963. *Computers and Thought*. New York: McGraw-Hill.

Ferrucci, D., et al. 2010. Building Watson: An overview of the DeepQA project. *AI Magazine* 31(3):59–79.

Green, B. F., et al. 1961. Baseball: An automatic question answerer. In *Proceedings of the Western Joint Computer Conference*, February 1961, Vol. 19., pp. 219–224.

Hutchins, J. 1986. *Machine Translation: Past, Present, Future*. Chichester, U.K.: Ellis Horwood.

McCorduck, P. 2004. *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*. Twenty-Fifth Anniversary Edition. Natick, MA: A. K. Peters.

Minsky, M. 1961. Steps toward artificial intelligence. *Proceedings of the IRE* 49:8–30.

Nilsson, N. J. 2010. *The Quest for Artificial Intelligence: A History of Ideas and Achievements*. Cambridge: Cambridge University Press.

Russell, S. J., and P. Norvig. 2010. *Artificial Intelligence: A Modern Approach*. Third edition. Upper Saddle River, NJ: Prentice Hall.

Samuel, A. L. 1959. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development* 3:210–229.

Schaeffer, J., et al. 2007. Checkers is solved. *Science* 317:1518–1522.

Wilks, Y. 1973. The artificial intelligence approach to machine translation. In *Computer Models of Thought and Language*, R. C. Shank and K. M. Colby, eds. San Francisco: W. H. Freeman, pp. 114–151.