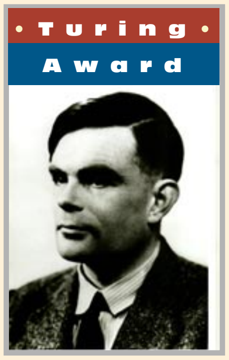


The Computer Museum

Raj Reddy

To Dream The Possible Dream

It is an honor and a pleasure for me to accept this award from ACM. It is especially gratifying to share this award with Ed Feigenbaum, who has been a close friend and helpful colleague for nearly 30 years.

As a second-generation artificial intelligence (AI) researcher, I was fortunate to have known and worked with many of the founding fathers of AI. By observing John McCarthy, my thesis advisor, during the golden age of AI Labs at Stanford in the 1960s, I have learned the importance of fostering and nurturing diversity in research far beyond one's own personal research agenda. Although his own primary interest was in common-sense reasoning and epistemology, under John's leadership, research in speech, vision, robotics, language, knowledge systems, game playing, and music, thrived at the AI labs. In addition, a great deal of path-breaking systems research flourished in areas such as Lisp, time-sharing, video displays, and a precursor to Windows called "pieces of glass."

From Marvin Minsky, who was visiting Stanford and helping to build the Mars Rover in '66, I learned the importance of pursuing bold visions of the future. And from Allen Newell and Herb Simon, my colleagues and mentors at Carnegie Mellon University (CMU) for over 20 years, I learned how one can turn bold visions into practical reality by careful design of experiments and following the scientific method.

I was also fortunate to have known and worked with Alan Perlis, a giant in the '50s and '60s computing scene and the first recipient of the Turing Award in 1966, presented at the ACM conference held in Los Angeles, which I attended while still a graduate student at Stanford.

While I did not know Alan Turing, I may be one of the select few here who used a computer designed by him. In the late 1950s, I had the pleasure of using a mercury delay-line computer (English Electric Deuce Mark II) based on Turing's original design of ACE. Given his early papers on "Intelligent Machines," Turing can be reasonably called one of the grandfathers of AI, along with early pioneers such as Vannevar Bush.

That brings me to the topic of this talk, "To dream the possible dream." AI is thought to be an impossible dream by many. But not to us in AI. It is not only a possible dream, but, from one point of view, AI has been a reality that has been demonstrating results for nearly 40 years. And the future promises to generate an impact greater by orders of magnitude than progress to date. In this talk I will attempt to demystify the process of what AI researchers do and explore the nature of AI and its relationship to algorithms and software systems research. I will discuss what AI has been able to accomplish to date and its impact on society. I will also conclude with a few comments on the long-term grand challenges.

Human and Other Forms of Intelligence

Can a computer exhibit real intelligence? Simon provides an incisive answer: "I know of only one operational meaning for 'intelligence.' A (mental) act or series of acts is intelligent if it accomplishes something that, if accomplished by a human being, would be called intelligent. I know my friend is intelligent because he plays pretty good chess (can keep a car on the road, can diagnose symptoms of a disease, can solve the problem of the missionaries and cannibals, etc.) I know that computer A is intelligent because it can play excellent chess (better than all but about 200 humans in the entire world). I know that Navlab is intelligent because it can stay on the road. The trouble with those people who think that computer intelligence is in the future is that they have never done serious research on human intelligence. Shall we write a book on 'What Humans Can't Do?' It will be at least as long as Dreyfus' book. Computer intelligence has been a

Can AI equal human intelligence? Some philosophers and physicists have made successful lifetime careers out of attempting to answer this question. The answer is, AI can be both more and less than human intelligence.

fact at least since 1956, when the Logic Theory machine found a proof that was better than the one found by Whitehead and Russell, or when the engineers at Westinghouse wrote a program that designed electric motors automatically. Let's stop using the future tense when talking about computer intelligence."

Can AI equal human intelligence? Some philosophers and physicists have made successful lifetime careers out of attempting to answer this question. The answer is, AI can be both more and less than human intelligence. It doesn't take large tomes to prove that they cannot be 100% equivalent. There will be properties of human intelligence that may not be exhibited in an AI system (sometimes because we have no particular reason for doing so or because we have not yet gotten around to it). Conversely, there will be capabilities of an AI system that will be beyond the reach of human intelligence. Ultimately, what will be accomplished by AI will depend more on what society needs and where AI may have a *comparative advantage* than on philosophical considerations.

Let me illustrate the point by two analogies that are not AI problems in themselves but whose solutions require some infusion of AI techniques. These problems, currently at the top of the research agenda within the information industry, are *digital libraries* and *electronic commerce.*

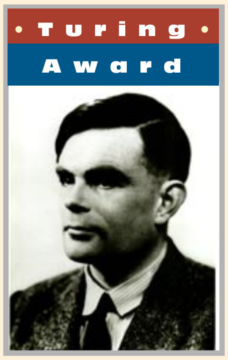
The basic unit of a digital library is an electronic book. An electronic book provides the same information as a real book. One can read and use the information just as we can in a real book. However, it is difficult to lie in bed and read an electronic book. With expected technological advances, it is conceivable a subnotebook computer will weigh less than 12 ounces and have a 6" x 8" high resolution color screen, making it look and feel like a book that you might read in bed. However, the analogy stops there. An electronic book cannot be used as part of your rare book collection, nor can it be used to light a fire on a cold night to keep you warm. You can probably throw it at someone, but it would be expensive. On the other hand, using an electronic book, you can process, index, and search for information; open the right page; highlight information; change font size if you don't have your reading glasses; and so on. The point is, an electronic book is not the same as a real book. It is both more and less.

A key component of electronic commerce is the electronic shopping mall. In this virtual mall, you can walk into a store, try on some *virtual* clothing, admire the way you look, place an order and have the real thing delivered to your home in 24 hours. Obviously, this does not give you the thrill of going into a real mall, rubbing shoulders with real people and trying on real clothing before you make your purchase. However, it also eliminates the problems of getting dressed, fighting the traffic and waiting in line. More importantly, you can purchase your dress in Paris, your shoes in Milan and your Rolex in Hong Kong without ever leaving your home. Again, the point is that an electronic shopping mall is not the same as a real shopping mall. It is both more and less.

Similarly, AI is both more and less than human intelligence. There will be certain human capabilities that might be impossible for an AI system to reach. The boundary of what can or cannot be done will continue to change with time. More important, however, it is clear that some AI systems will have super human capabilities that would extend the reach and functionality of individuals and communities. Those who possess these tools will make the rest of us look like primitive tribes. By the way, this has been true of every artifact created by the human species, such as the airplane. It just so happens that AI is about creating artifacts that enhance the mental capabilities of the human being.

AI and Algorithms

Isn't AI just a special class of algorithms? In a sense it is; albeit a very rich class of algorithms, which have not yet received the attention they deserve. Second, a major part of AI research is concerned with problem definition rather than just problem solution. Like complexity theorists, AI researchers also tend to be concerned with NP-complete problems. But unlike their interest in the complexity of a given problem, the focus of research in AI tends to revolve around finding algorithms that provide approximate, satisfying solutions with no guarantee of optimality.



The concept of satisfying solutions comes from Simon's pioneering research on decision making in organizations leading to his Nobel Prize. Prior to Simon's work on human decision making, it was assumed that, given all the relevant facts, the human being is capable of rational choice weighing all the facts. Simon's research showed that "computational constraints on human thinking" lead people to be satisfied with "good enough" solutions rather than attempting to find rational optimal solutions weighing all the facts. Simon calls this the principle of "bounded rationality." When people have to make decisions under conditions which overload human thinking capabilities, they don't give up, saying the problem is NP-complete. They use strategies and tactics of *optimal-least-computation search* and not those of *optimal-shortest-path search*.

Optimal-least-computation search is the study of approximate algorithms that can find the best possible solution given certain constraints on the computation, such as limited memory capacity, limited time, or limited bandwidth. This is an area worthy of serious research by future complexity theorists!

Besides finding solutions to exponential problems, AI algorithms often have to satisfy one or more of the following constraints: exhibit adaptive goal-oriented behavior, learn from experience, use vast amounts of knowledge, tolerate error and ambiguity in communication, interact with humans using language and speech, and respond in real time.

Algorithms that exhibit adaptive goal oriented behavior. Goals and subgoals arise naturally in problems where algorithm specification is in the form of "What" is to be done rather than "How" it is to be done. For example, consider the simple task of asking an agent, "Get me Ken." This requires converting this goal into subgoals, such as look up the phone directory, dial the number, talk to the answering agent, and so on. Each subgoal must then be converted from What to How and executed. Creation and execution of plans has been studied extensively within AI. Other systems such as report generators, 4GL systems, and data base query-by-example methods, use simple template-based solutions to solve the "What to How" problem. In general, to solve such problems, an algorithm must be capable of creating for itself an agenda of goals to be satisfied using known operations and methods, the so-called "GOMs approach." Means-ends analysis, a form of goal-oriented behavior, is used in most expert systems.

Algorithms that learn from experience. Learning from experience implies the algorithm has built-in mechanisms for modifying internal structure and function. For example, in the "Get me Ken" task, suppose Ken is ambiguous and you help the agent to call the right Ken; next time you ask the agent to "Get me Ken," it should use the same heuristic that you used to resolve the ambiguity. This implies the agent is capable of acquiring, representing, and using new knowledge and engaging in a clarification dialog, where necessary, in the learning process. Dynamic modification of internal structure and function is considered to be dangerous in some computer science circles because of the potential for accidental overwriting of other structures. Modifying probabilities and modifying contents of tables (or data structures) have been used successfully in learning tasks where the problem structure permits such a formulation of learning. The Soar architecture developed by Newell et al., which uses rule-based system architecture, is able to discover and add new rules (actually "productions") and is perhaps the most ambitious undertaking to date to create a program that improves with experience.

Algorithms that interact with humans using language and speech. Algorithms that can effectively use speech and language in human-computer interface will be essential as we move toward a society where nonexperts use computers in their day-to-day problems. In the previous example of the "Get me Ken" task, unless the agent can conduct the clarification dialog with the human master using language and speech or some other natural form of communication, such as "Form Filling," widespread use of agents will be a long time coming. Use of language and speech involves creating algorithms that can deal with not only ambiguity and nongrammatical-

It just so happens that AI is about creating artifacts that enhance the mental capabilities of the human being.

ity but also with parsing and interpreting of natural language with a large, dynamic vocabulary.

Algorithms that can effectively use vast amounts of knowledge. Large amounts of knowledge not only require large memory capacity but creates the more difficult problem of selecting the right knowledge to apply for a given task and context. There is an illustration that John McCarthy is fond of using. Suppose one asks the question, "Is Reagan sitting or standing right now?" A system with a large database of facts might proceed to systematically search the terabytes of data before finally coming to the conclusion that it does not know the answer. A person faced with the same problem would immediately say, "I don't know," and might even say, "and I don't care." The question of designing algorithms "that know what they do not know" is currently an unsolved problem. With the prospect of very large knowledge bases looming around the corner, "knowledge search" will become an important algorithm design problem.

Algorithms that tolerate error and ambiguity in communication. Error and ambiguity are a way of life in human-to-human communication. Warren Teitelman developed nearly 25 years ago an interface called "Do What I Mean (DWIM)." Given the requirements of efficiency and getting the software done in time, all such ideas were deemed to be frivolous. Today, with the prospect of Giga PCs around the corner, we need to revisit such error-forgiving concepts and algorithms. Rather than saying "illegal syntax," we need to develop algorithms that can detect ambiguity (i.e., multiple possible interpretations, including null) and resolve it where possible by simultaneous parallel evaluation or by engaging in clarification dialog.

Algorithms that have real-time constraints. Many software systems already cope with this problem, but only through careful, painstaking analysis of the code. Not much work has been done on how to create algorithms that can accept a "hurry-up" command! One approach to this problem appears in the Prodigy system, which generates an approximate plan immediately and gradually improves and replaces that plan with a better one as the system finds more time to think. Thus, it always has an answer but the quality of the answer, improves with time. It is interesting to note that many of the iterative algorithms in numerical analysis can be recast in this form of "anytime answers."

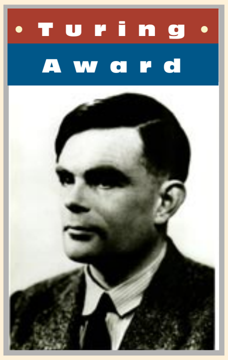
Algorithms with self-awareness. Algorithms that can explain their capabilities (e.g., a Help command capable of answering "how-to" and "what-if" questions), and monitor, diagnose, and repair themselves in the presence of viruses require internal mechanisms that can be loosely called "self-awareness." Online hypertext manuals and checksums are simpler examples of such mechanisms. Answering how-to and what-if questions is harder. Monitoring and diagnosis of invading viruses implies interpreting incoming action requests rather than just executing them.

Such algorithm design issues seem to arise naturally in AI, and numerous solutions have been created by AI researchers in specific contexts. Further development and generalization of such solutions will enrich all of computer science.

Software Systems and AI

Isn't AI just software? Isn't a TV set just electronics? In a general sense, AI is just software, although AI systems tend to get large and complex. Attempting to build AI systems often implies building the necessary software tools. For example, AI researchers were responsible for many of the early advances in programming languages, such as list structures, pointers, virtual memory, dynamic memory allocation, and garbage collection, among others.

Large AI systems, especially those that are deployed and in daily use, share many of the common problems that are observed in developing other complex software systems, i.e., the problems of "not on time," " over budget," and "brittle." The "mythical man-month" principle affects AI systems with a vengeance. Not only do these systems tend to be large and complex, but they often cannot fall back on a learning curve based on having designed similar systems before. Thus, it is difficult to formulate requirement specifications or testing pro-



cedures as is usual in conventional software engineering tasks.

There are a number of new concepts that are emerging within the context of AI that suggest new approaches and methodologies for all of computer science:

- *Plug-and-play architectures.* To produce an integrated concept demonstration of an AI system that routinely uses components that learn from experience, use knowledge, tolerate error, use language, and operate in real time, one cannot start from scratch each time. The application component of such systems can be less than 10% of the total effort. If the system is to be operational in a reasonable amount of time, one needs to rely on interfaces and software architectures that consist of components that plug and play together. This idea is similar to the old blackboard concept: cooperating agents that can work together but don't require each other explicitly.
- *The 80/20 rule.* The last 40 years of attempting to build systems that exhibit intelligent behavior has shown us how difficult the task really is. A recent paradigm shift is to move away from autonomous systems that attempt to entirely replace a human capability to systems that support a human master as intelligent agents. For example, rather than trying to create a machine translation system, create a *translation assistant* that provides the best alternative interpretations to a human translator. Thus, the goal would be to create a system that does 80% of a task and leaves the remaining 20% to the user. Once such a system is operational, the research agenda would target the remaining 20% and repeat the 80/20 rule on the next version. Thus, the system would incrementally approach human performance, while at all times providing a usable artifact that improves human productivity by factors of 3 to 5 after every iteration. However, the task is not as simple as it may appear. With this new paradigm, the old problem of "how can a system know what it does not know" raises its ugly head. For an intelligent agent to be able to say, "Please wait, I will call my supervisor," it must be self-aware! It must know what it can and cannot do. Not an easy task either, but one that needs to be solved anyway.
- *Fail-fast strategies.* In engineering design there are two accepted practices: "get it right the first time" and "fail fast." The former is used when one is designing a product that has been produced many times before. In building systems or robots that have never been built before, attempting to "get it right the first time" may not be the best strategy. Our recent experiment for NASA in building the Dante robot for exploring volcanic environments is one such example. Most NASA experiments are required to be fail-safe, because in many missions human lives are at risk. This requirement to be error free leads to 15-year planning cycles and billion-dollar budgets. The fail-fast strategy says that if you are trying to build a complex system that has never been built before, it is prudent to build a series of throw-away systems and improve the learning curve. Since such systems will fail in unforeseeable ways, rather than viewing failure as an unacceptable outcome one should view failure as a stepping-stone to success. Our first Dante failed after taking a few steps. The second one lasted a week. We are now working on a third generation model. Both time and cost expended on this experiment is at least an order of magnitude smaller than those for comparable conventional missions.
- *The scientific method.* At an NRC study group meeting, a physicist asked, "Where is the science in computer science?" I am happy to say that we are beginning to have examples of the "hypothesis, experiment, validation, and replication" paradigm within some AI systems. Much of the speech-recognition research has been following this path for the past 10 years. Using common databases, competing models are evaluated within operational systems. The successful ideas then seem to appear magically in other systems within a few months, leading to validation or refutation of specific mechanisms for modeling speech. ARPA deserves a lot of the credit for requiring the community to use such validation procedures. All of experimental computer science could benefit from such disciplined experiments. As Newell used to say, "Rather than

AI continues to be a possible dream worthy of dreaming. Advances in AI have been significant. AI will continue to be the generator of interesting problems and solutions.

argue, let us design an experiment to prove or disprove the idea!"

AI and Society

Why should society support AI research and more broadly, computer science research? The information technology industry, which was nonexistent 50 years ago, has grown to be over 10% of the GNP and is responsible for over 5% of the total employment in the country. While AI has not played a substantial role in the growth of this industry, except for the expert system technology, it appears possible that we are poised on the threshold of a number of important applications that could have a major impact on society. Two such applications include an accident-avoiding car and a reading coach:

The Navlab: The Carnegie Mellon Navlab project brings together computer vision, advanced sensors, high-speed processors, planning, and control to build robot vehicles that drive themselves on roads and cross country.

The project began in 1984 as part of ARPA's Autonomous Land Vehicle (ALV) program. In the early 1980s, most robots were small, slow indoor vehicles tethered to big computers. The Stanford Cart took 15 minutes to map obstacles, plan a path, and move each meter. The CMU Imp and Neptune improved on the Cart's top speed, but still moved in short bursts separated by long periods of looking and thinking. In contrast, ARPA's 10-year goals for the ALV were to achieve 80 kph on roads and to travel long distances across open terrain.

The Navlab, built in 1986, was our first self-contained testbed. It had room for on-board generators, on-board sensors, on-board computers, and, most importantly, on-board graduate students. Researchers riding on board were in a much better position to observe its behavior and to debug or modify the programs. They were also highly motivated to get things working correctly.

The latest is the Navlab II, an army ambulance HMMWV. It has many of the sensors used on earlier vehicles, plus cameras on pan/tilt mounts and three aligned cameras for trinocular stereo vision. The HMMWV has high ground clearance for driving on rough terrain, and a 110-kph top speed for highway driving. Computer-controlled motors turn the steering wheel and control the brake and throttle.

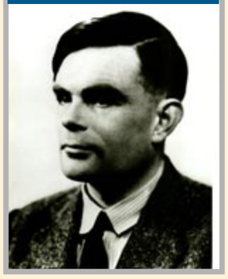
Perception and planning capabilities have evolved with the vehicles. ALVINN is the current main road-following vision system. ALVINN is a neural network, which learns to drive by watching a human driver. ALVINN has driven as far as 140 km, and at speeds over 110 kph.

Ranger finds paths through rugged terrain. It takes range images, projects them onto the terrain, and builds Cartesian elevation maps. Ranger has driven the vehicle for 16 km on our test course.

SMARTY and D* find and follow cross-country routes. D* plans a route using A* search. As the vehicle drives, SMARTY finds obstacles using GANESHA's map, steers the vehicle around them and passes the obstacles to D*. D* adds the new obstacles to its global map and replans the optimal path.

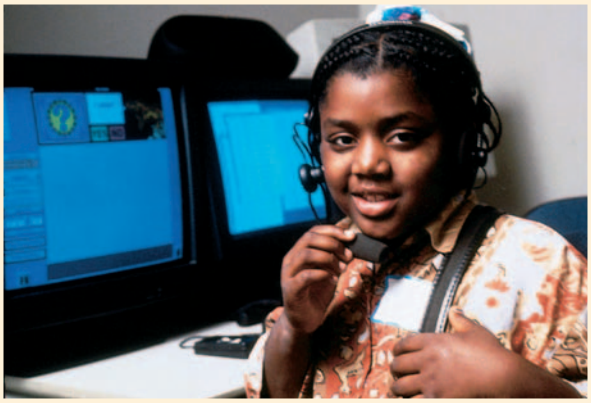
Other Navlab modules include RACCOON, which follows a lead vehicle by tail-light tracking; YARF, which tracks lines and detects intersections; and PANACEA, which controls a pan/tilt mount to see around curves.

The 10-year goals of the ALV program have been met. Navlab technology is being integrated with specialized planners and sensors to demonstrate practical missions, and a new project funded by the Department of Transportation is investigating Navlab technology for preventing highway accidents. As basic driving capabilities mature, the Navlab continues to provide new opportunities



both for applications and for continued research in perception, planning, and intelligent robot systems.

There are six million automotive accidents in the U.S. each year, costing over $50 billion to repair. These accidents result in 40,000 fatalities, 96% of which are caused by driver error. Using Navlab technology can eliminate a significant fraction of those accidents, given appropriate sensors and computation integrated into each car. AI and computer science researchers can justifiably be proud of such a contribution to society.



The LISTEN Project: At Carnegie Mellon, Project LISTEN is taking a novel approach to the problem of illiteracy. We have developed a prototype automated reading coach that listens to a child read aloud, and helps when needed. The system is based on the CMU Sphinx II speech recognition technology. The coach provides a combination of reading and listening, in which the child reads wherever possible and the coach helps wherever necessary—a bit like training wheels on a bicycle.

The coach is designed to emphasize comprehension and ignore minor mistakes, such as false starts or repeated words. When the reader gets stuck, the coach jumps in, enabling the reader to complete thesentence. When the reader misses an important word, the coach rereads the words that led up to it, just like the expert reading teachers after whom the coach is modeled. This context often helps the reader correct the word on the second try. When the reader runs into more difficulty, the coach rereads the sentence to help the reader comprehend it. The coach's ability to listen enables it to detect when and where the reader needs help. Further research is needed to turn this prototype into robust educational software. Experiments to date suggest that it has the potential to reduce children's reading mistakes by a factor of five and enable them to comprehend material at least six months more advanced than they can read on their own.

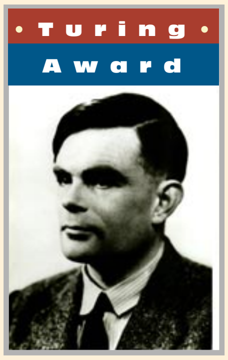
Illiteracy costs the U.S. over $225 *billion* dollars annually in corporate retraining, industrial accidents, and lost competitiveness. If we can reduce illiteracy by just 20%, Project LISTEN could save the nation over $45 *billion* a year.

Other AI research to have a major impact on society includes Feigenbaum's early research on knowledge systems; Kurzweil's work on reading machines for the blind; Papert, Simon, Anderson, and Shank's contributions to learning by doing; and the robotics work at MIT, Stanford, and CMU.

Several new areas where advances in AI are likely to have an impact on society are:

- •Creation of intelligent agents to monitor and manage the information glut by filtering, digesting, abstracting, and acting as an agent of a human master;
- •Making computers easier to use: use of multimodal interfaces, DWIM techniques, intelligent help, advice giving agents, and cognitive models;
- •Aids for the disabled: Development of aids for people with sensory, cognitive, or motor disabilities based on AI technologies appears promising, especially the area of creating aids for cognitive disabilities such as memory and reasoning deficiencies; and
- •Discovery techniques: Deriving knowledge and information for a large amount of data, whether the data is scientific or financial, has the potential for a major impact.

Whether it is to save lives, improve education, improve productivity, overcome disabilities, or foster discovery, AI has produced and will continue to produce research results with the potential to have



a profound impact on the way we live and work in the future.

Grand Challenges in AI

What's next for AI? There are several seemingly reasonable problems which are exciting and challenging, and yet are currently unsolvable. Solutions to these problems will require major new insights and fundamental advances in computer science and artificial intelligence. Such challenges include: a world champion chess machine, a translating telephone, and discovery of a major mathematical result by a computer, among others. Here, I will present two such grand challenges which, if successful, can be expected to have a major impact on society:

Self-Organizing Systems: There has been a long and continuing interest in systems that learn and discover from examples, from observations, and from books. Currently, there is a lot of interest in neural networks that can learn from signals and symbols through an evolutionary process. Two long-term grand challenges for systems that acquire capability through development are: read a chapter in a college freshman text (say, physics or accounting) and answer the questions at the end of the chapter; and learn to assemble an appliance (such as a food processor) from observing a person doing the same task. Both are extremely hard problems, requiring advances in vision, language, problem-solving techniques, and learning theory. Both are essential to the demonstration of a self-organizing system that acquires capability through (possibly unsupervised) development.

Self-Replicating Systems: There have been several theoretical studies in this area since the 1950's. The problem is of some practical interest in areas such as space manufacturing. Rather than uplifting a whole factory, is it possible to have a small set of machine tools that can produce, say, 80% of the parts needed for the factory (using the 80/20 rule discussed earlier), using locally available raw materials and assemble it in situ? The solution to this problem of manufacturing on Mars involves many different disciplines, including materials and energy technologies. Research problems in AI include knowledge capture for replication; design for manufacturability; and design of systems with self-monitoring; self diagnosis; and self-repair capabilities.

Conclusion

Let me conclude by saying that AI continues to be a possible dream worthy of dreaming. Advances in AI have been significant. AI will continue to be the generator of interesting problems and solutions. However, ultimately the goals of AI will be realized only with developments within the broader context of computer science, such as the availability of a Giga-PC—i.e., a billion-operations-per-second computer at PC prices—and advances in software tools, environments, and algorithms.

Acknowledgment

I was fortunate to have been supported by ARPA for nearly 30 years. I would like thank Lick Licklider, Ivan Sutherland, Larry Roberts, and Bob Kahn for their vision in fostering AI research even though it was not part of their own personal research agenda, and the current leadership at ARPA: Duane Adams, Ed Thompson, and Allen Sears, for continuing the tradition.

I am grateful to Herb Simon, Ed Feigenbaum, Jaime Carbonell, Tom Mitchell, Jack Mostow, Chuck Thorpe, and Jim Kocher for suggestions, comments, and help in the preparation of this lecture.

RAJ REDDY, Herbert A. Simon University Professor, is Dean of the School of Computer Science at CMU. His mailing address is Carnegie Mellon University, School of Computer Science, 5325 Wean Hall, Pittsburgh, PA 15123-3890. email:rr@cmu.edu.