A Personal View of Expert Systems:
Looking Back and Looking Ahead

by

Edward A. Feigenbaum
A Personal View of Expert Systems: 
Looking Back and Looking Ahead* 

Edward A. Feigenbaum 

I feel greatly honored by the decision of the World Congress on Expert Systems to establish in my name its award for achievement. That is truly an extraordinary event in the life of a scientist and scholar. It is the kind of event one normally associates with dead people of mythical proportions, like Marconi or Turing! I would like to give my heartfelt thanks to those who made the decision; and I want particularly to thank Professor Jay Liebowitz, whose energy and vision made possible the birth and health of the World Congress on Expert Systems.

In this paper, I will do some storytelling—of my career in AI, of the events preceding and accompanying the birth of expert systems, and of the recent birth of the second era of knowledge based systems. When a movie tells a story, sometimes the credits are given at the front and sometimes they are left to the end. I am moved to want to give credit at the very beginning of this paper to many extraordinary people who have been my intellectual companions and great friends along the way. In truth, I feel like a stand-in for the staff and students of the Stanford Heuristic Programming Project, whose work over 15 years was responsible for the birth and early development of expert systems.

Buchanan, Lederberg, and Djerassi

Of course Feigenbaum is but one long syllable of a polysyllabic contributor named BuchananandFeigenbaum. I owe an extraordinary intellectual and personal debt to Bruce Buchanan. For more than twenty years, we two thought like one and worked together in total friendship and harmony toward our common intellectual goals.

What can one say about the 13 year collaboration that Bruce and I had with Joshua Lederberg on DENDRAL, MOLGEN, and the development of a national community of AI in Medicine and Biology? And the 18 year collaboration that we had with Carl Djerassi on the DENDRAL Project? Lederberg, the Nobel Prize geneticist who suggested DENDRAL to me in the first instance, and then helped us carry it out. Djerassi, inventor of the birth control pill, also one of the world's most eminent mass spectrometrists, guiding us through the fields of world-class

* This was an acceptance speech for the Feigenbaum Medal presented at the World Congress on Expert Systems at Orlando, Florida, December 1991.

This work was supported by NASA Cooperative Agreement No. NAG2-581 (under ARPA Order 6822) and by NASA Cooperative Agreement NCC2-537.
expertise. It is a remarkable part of the story of the birth of expert systems that that birth was assisted by two of the greatest scientists in the world.

Some Stories Before and At The Birth of Expert Systems

Certain aspects of my background, my education, and my early professional development set the stage for the innovations I made that lead to expert systems. I would like to recount these because too often such innovations are treated as if they were like Athena who, in the Greek myth, was born fully clothed from the head of Zeus. In fact each innovation has a context in someone's personal development.

As a child, I was fascinated with science and with mathematics (there's nothing unusual with that), but I was also fascinated with calculators, and that was unusual in the 1940's. My father was an accountant. He owned a large heavy mechanical calculator known as a Monroe, with a motor that turned number wheels, a carriage that had to be shifted one position at a time by hand, and a very large numeric keypad. He taught me to use it expertly. Proudly I carted its heavy bulk on the school bus to show off my prowess in operating this amazing machine to my (not very interested) high school friends.

In 1952, I entered engineering school at Carnegie Institute of Technology (now Carnegie Mellon University). The only calculator I saw there until 1956 was my slide rule. (I keep it now in my fireproof safe because it's so valuable as an antique instrument, though I bought it new in 1952!)

Stumbling In On The Birth of AI

When the rigors of freshman Engineering were over, I felt the need to stretch my mind beyond the standard engineering subjects of the curriculum. I was fortunate to be at Carnegie. They had a splendid Arts school. And they had a new school called the Graduate School of Industrial Administration (GSIA). I found a course in game theory, some research activity in decision making, and in Fall, 1955 a course from Professor Herbert Simon on Mathematical Models in the Social Sciences. This course was to be a major turning point in my life.

Both McCorduck's history of AI (McCorduck, 1979) and Simon's autobiography (Simon, 1991) tell the story of Simon informing his small seminar, at their first meeting after New Year 1956, that "over the Christmas break, Allen Newell and I invented a thinking machine." To us in the seminar, what could that possibly mean? He gave us an operating manual for a machine that we did not have at Carnegie Tech, a computer called an IBM 701. The 701 was IBM's first entry into the large-scale commercial computing market. Newell and Simon were using a 701 at the RAND Corporation in Los Angeles, a continent away, in a collaboration with their colleague J.C. Shaw. I remember staying up all night to read the IBM 701 manual; and as the dawn came, I rose a born-again computer
scientist, though of course the term had not been invented yet, and would not be for almost another decade.

It is almost impossible now to have a feeling for early computing unless you lived through those years. Carnegie Tech, a major engineering school, had no electronic digital computer in January 1956. The big commercial machines of the day (such as the 701) cost millions of dollars. They were built of power-hungry and unreliable vacuum tubes; and had electrostatic (!) volatile memories of 4K words at 36 bits per word, i.e., less than 20K bytes.

The "thinking machine" that Newell and Simon had pioneered was the first heuristic program, the Logic Theory program. John McCarthy had wordsmithed a name for the new field, "artificial intelligence." What better work for a doctorate than this brand new and fascinating field, that was built on top of the "mechanical calculator" that so fascinated me at an early age? I decided to stay on at Carnegie Tech for Ph.D. work with Simon. But in the summer before graduate school (1956), I went to New York to work for IBM, where I was taught the early rudiments of the art of computer programming.

To give you a feeling for what programming was like in 1956, symbolic assemblers had just been invented; and as part of the summer program at IBM, we summer students heard from a then-unknown IBM scientist, John Backus, about wonderful new concepts called "compilers" and "FORTRAN" that would appear later in the year.

The Miracle Years at Carnegie

When I returned for the Fall term, a computer had arrived, an IBM650, whose memory was a mechanically rotating magnetic drum of 2K words of ten digits each. Fortunately I had learned to program the 650 at IBM in the summer. Since almost no one else knew how to program the machine, I and a handful of others shared the 650 (for a few months) as a kind of million-dollar shared personal computer.

There were several themes that were emerging in the new field of AI right after the "big bang":

a. the continuing invention of the concepts of the list processing languages.

b. the modeling of the information processes underlying human cognition; and the testing of these models against the results of experiments with human subjects.

c. the construction of intelligent programs whose competence would rival or exceed human competence in the same tasks.
Simon was now my mentor, and as is usual in mentor-student relationships, he suggested to me a problem to work on: modeling the information processing activities underlying phenomena of human rote memory, as studied by psychologists since the turn of the century.

That model of human cognition—Elementary Perceiver and Memorizer (EPAM)—became my thesis work. EPAM is today still alive—a model of considerable predictive power, still being worked on by Simon and others at Carnegie Mellon. It has given detailed predictions and sometimes startling explanations of human learning and memory phenomena in terms of elementary information processes. My programming of EPAM called for one major conceptual advance. Of course, since most of programming technology was not yet invented, it was easy to make a major conceptual advance. The memory structures of EPAM were involved in learning new material, hence they needed to grow dynamically. I invented the dynamically growing decision trees, which I called "discrimination nets." In AI they are sometimes called "EPAM nets," and in Computer Science "sorting networks."

The work on EPAM was very exciting and rewarding but in the deepest sense unsatisfying because it failed to satisfy my most basic motive, growing out of my early fascination with calculating machines—the motive of building highly competent intelligent machines, the motive that I now call the "performance motive" of AI. It entails a vision with deep engineering aspirations (less theory, more system building). Thus, even at this earliest period of AI's history and my work in the field, the seeds of my work on expert systems, that were to germinate and grow in the 1960's and 70's, were already planted.

Other Events of 1956-59

I can not leave the "miracle years" without at least mentioning some of the other events going on around me that helped to shape me.

Newell, Shaw and Simon were inventing and programming the General Problem Solver (GPS), the best known AI program of its era. I was present at the very first seminar at which Al Newell showed us how it would work. Newell, Shaw, and Simon were also programming their path-breaking chess playing program.

The first public list processing language and system was released: IPL-5 for the IBM704. I did most of the programming of that IPL (thereby beginning and ending my career as a system programmer and computer language implementer!).

In addition, dramatic new economic theory was being created all around me at the Graduate School of Industrial Administration. Indeed my first published paper involved not AI but economics—one of the first computer simulations
ever done of an economic model. It was a joint effort with a young economist, Richard Cyert, who would later become president of Carnegie Mellon University. The work done at that graduate school at that time has won three Nobel Prizes in Economics: Modigliani, Miller, and Simon. One can not help but be moved and molded forever by such an environment of innovation, achievement, excellence, and challenge.

**Supernova and Beyond**

Let me try to draw the threads of this part of the story together, with some reflections. It is rare to experience life at the center of an intellectual supernova. As I reexperience it by reflection and in retrospect, it is an amazing event and experience. Alas, but only in retrospect. Strange as it may seem, it is difficult to understand and to savor it while it is happening. While one is living it, one adapts to the extraordinary as if it were the usual. In my career I have had the incredible luck to have the supernova experience twice. The second time involved DENDRAL; the birth of expert systems; and the subsequent explosion of early expert systems in my laboratory. And the second time around, I was no better at recognizing what was happening than I was the first time!

I worked through the early 1956-65 period of AI in an exciting and creative way. But ultimately, teaching at Berkeley, away from the direct influence of Simon and Newell, I found my path unsatisfying. EPAM and other work on information processing models of human cognition was not lined up with my personal dream—the dream of the super-intelligent machine. I did not resonate with cognitive psychologists; I resonated with those who were beginning to be called computer scientists, especially those with an engineering approach. I believe it is very important for young people to sense these directions and mismatches early in their career and head in the direction of their dreams.

**Precursors to Expert Systems**

It is one thing to head towards your dream. It is entirely another thing to design a path to get yourself there. That was my task in 1962-64. It was long and tortuous, like Jacob wrestling with the angel.

The state of my thinking is captured by something I wrote in 1961 for the preface to Part I of the anthology *Computers and Thought* (Feigenbaum and Feldman, 1963):

"Artificial intelligence currently is strong on deductive inference, weak on inductive inference. Yet, in the melting pot of everyday intelligence, inductive inference is certainly the more significant ingredient. One way of looking at the problem is that we need programs which will in some sense induce internally stored "models" of external environments...Looked at in another way, this is the problem of hypothesis formation by machine..."
This was a key element in my thinking and it led eventually to DENDRAL.

Of the many indelible things I learned from Newell and Simon, of the greatest importance to my style of inquiry was the experimental method. It was essential that I find a concrete problem to ground my inquiry and focus my efforts. I was searching for an hypothesis formation task to serve my needs—as the chess task and the theorem proving task had served the needs of Newell and Simon.

Wrestling With The Angel

Thus began the wrestling match. I struggled with the feasibility of various tasks (given the intellectual and computer resources of the time); the likelihood that something important would be learned from studying the task; and the significance of the task (would it be taken seriously as a human challenge, as chess was, as propositional calculus was?) I don't now remember all the various tasks I examined, but I do remember carefully considering the task of inducing a model of the rules of baseball from the flow of events in a game—a task that I discarded but was later used by Professor Elliott Soloway.

I remember being very much aware of and guided by what I considered to be a fundamental task selection issue. Later, it would be termed an issue of knowledge acquisition. To me it laid itself out this way:

In order to study and model the processes of hypothesis induction, I had to work with someone who was a very good performer of that task, with much the same spirit that led Samuel to interview master checkers players, and Simon to consult books on great chess play. What were the heuristics of performance in hypothesis formation? Where were they likely to be most easily studied and observed? I came to believe the answer to be "scientists; in scientific thinking." I thought of scientists as "professional inducers." That was their job. They were trained to do it well, and they did a great deal of it. Furthermore, they were generally of a highly rationalistic mindset, so that it would be easy to explain to them what the AI methodology was all about, and easy for them to accommodate to it by exposing in detail the pattern of their thought processes. In other words, for scientists, more than anyone else, I believed, induction and hypothesis formation were at the surface of their mental processes and could be studied most easily. These were overly simplistic intuitions and speculations. They could have been wildly wrong. But they were right. The "work with scientists" intuition was one of the most fruitful I have ever had.

It has been said that good things happen to the prepared mind. A monthly gathering at Stanford of the few people in the San Francisco Bay Area (literally a handful) interested in machine intelligence was started in 1964. John McCarthy attended, and there began a renewal of our acquaintanceship that ultimately led to a job offer and a long career at Stanford. (Thank you, John!) Also attending
was Joshua Lederberg, chairman of Stanford's Genetics Department. In 1964, Lederberg's long-standing interest in computing had been revived. He had become deeply interested in symbolic computation and to some extent in AI-like goals of modeling (or at least augmenting) human intelligence. He was particularly interested in how the new ideas of AI might be put to significant use in the service of science.

The Birth of Expert Systems at Stanford

I joined the Stanford faculty in January of 1965, thereby missing out on all the exciting chaos of the Berkeley of the mid-Sixties. I immediately began discussions with Lederberg about my problem of selecting a task area in which to study the hypothesis formation behavior of scientists.

Lederberg suggested the task of inducing the topology of molecules from their mass spectra. He was then directing a project whose goal was to explore for life on Mars (or at least for life-precursor molecules) Indeed, it was Lederberg who invented the term exobiology. In early 1965, his lab was measuring the mass spectra of amino acids. He thought mass spectrometry to be:

- a quintessential inductive task of hypothesis formation
- a task that might have practical consequences in terms of analyzing samples in a Mars landing probe
- a task that could be based on an algorithm he had recently developed (called the DENDRAL algorithm; the algorithm would give us a starting point, a legal-move-generator).

Lederberg had suggested a task that met all my specifications.

My immediate inquiry was to see if existing AI concepts could be applied to this task, and how; and to recruit graduate students to work on the project. A short paper emerged in the Spring of 1965, the first DENDRAL paper. It was essentially an advertising piece for the work, whose cover sheet was emblazoned with the slogan "Opportunities for Graduate Research." Two masters students, William White and Georgia Sutherland, seized the opportunity and became long-term contributors to the DENDRAL Project.

In 1965, Bruce Buchanan joined the DENDRAL Project immediately after completing his Ph.D. thesis on the Logics of Scientific Discovery (Buchanan, 1965). How pertinent a thesis topic for the work we were to undertake together!
Machines We Used For The DENDRAL Research

The machines of 1965 had RAM core memories of 32K words (about 130K bytes) and were many times slower than today's desk-top machines. (How many readers remember core memories? Probably as many as remember vacuum tubes. Not many.)

We worked on an IBM 7090, later on a DEC PDP-6, and then many years later its much faster successor, the DEC PDP-10. We also worked on early models of the IBM 360 series.

Time sharing was not available at first. The Stanford Artificial Intelligence Laboratory was developing its pioneering time sharing system for the PDP-6; and the Stanford Computer Center was developing a remote job entry system for the IBM 360 following the collapse of IBM's TSS-360 time sharing project.

The Early Insights of the DENDRAL Project

Much that is important to the history of knowledge based systems was first observed, understood, and refined in the DENDRAL project.

The theme that became the banner of the DENDRAL effort, and later the other expert system projects that our laboratory did, that "in the knowledge lies the power," quickly surfaced. Knowledge of organic chemistry, topology, and mass spectrometry were essential to our success. We learned that the knowledge of mass spectrometry of a geneticist (Lederberg) was neither broad enough nor deep enough, and that a specialist would have to be recruited. In 1966, there was no off-the-shelf concept of knowledge acquisition ("OK, team, we're entering the knowledge acquisition phase of this project. Let's go recruit an expert!) Rather, we could just see from our experimental work--our emerging programs--that more knowledge was what we needed to improve the competence of the programs. With some careful thought, we devised a strategy for capturing and holding the attention of Lederberg's friend, Carl Djerassi, head of the Stanford Mass Spectrometry Laboratory.

We set up a demo of DENDRAL's abilities. First we showed its moderately competent behavior on some of the simpler amino acids. Next we showed its incompetent behavior on simple families called ketones and alcohols. We knew the behavior was incompetent because the program knew almost nothing about the mass spectrometry of these families; and we predicted that Djerassi would tell us immediately how stupidly the program was performing. But we had prepared an approach for that.

"Yes, Carl, but what is it that you know that DENDRAL doesn't know that allows you to solve the problem?"
This we regarded as "setting the hook". It worked.

We did not need to invent an entirely new problem solving architecture to handle the DENDRAL task. We were able to do a variation of the familiar generate-and-test, which we called plan-generate-test. In many ways, DENDRAL was similar to chess playing programs. Lederberg's DENDRAL algorithm was used as the legal move generator. It was capable of producing a large number of topologically legal structure candidates using the knowledge of chemical valence. These were pruned to a small number of chemically plausible structure candidates using chemical knowledge and mass spectral interpretation knowledge.

It seemed like a little miracle to me at the time, but all the pieces of the puzzle that had been troubling me for three years seemed to fit into place. And an astonishingly competent collection of people had been brought together by mutual interest to do the work.

The early results using the amino acid spectra from Lederberg's Mars project were promising. The program's performance on ketones and alcohols improved dramatically after Djerassi became involved, and he organized for us a carefully orchestrated sequence of organic chemical families to attack: ethers, thioethers and amines. We were following a knowledge acquisition plan structured along the lines of Djerassi's well-known "bible" of mass spectrometry. We were harvesting our results in the form of codified knowledge and program structure from a large number of computational experiments.

Rule-based Representation of Knowledge: How It Happened

A major insight about knowledge representation came from the emerging AI science circa 1965 as it impacted my continuing struggle with the questions of the aforementioned "internal models of external environments"--their representation, their manipulation, and particularly how a program could learn these models.

By 1965, my struggle had focused in on a specific desideratum: the need for relatively small chunks of knowledge, represented in a highly modular way. Could little pieces of LISP code be made to be such a representation? I didn't see a solution down that path.

Early in the life of Stanford's new Computer Science Department, it hosted Newell as a Distinguished Lecturer. Newell's lectures introduced me to the idea of productions. Here was a promising seed for thinking about knowledge in small chunks, highly modular.

Don Waterman was a Ph.D. student of mine at that time. After Newell's lectures I gave to Don the problem of thinking through how the idea of productions
could be used as a new and different representation for the knowledge that we were acquiring from the chemists in the DENDRAL project. DENDRAL was accumulating so much knowledge so fast (in the form of LISP code) that we were approaching a crisis of complexity.

Don worked on this for several months, and then wrote a report in which he said:

a) he hadn't figured out how to do it for DENDRAL but

b) he had thought of how to use these productions to represent the knowledge of a card game for which he was building a learning program. (This eventually became his thesis.) (Waterman, 1968)

For Don's use in the card game, the tool seemed to fit the need in precisely the way I had abstractly envisioned, and I was very encouraged. So was Bruce Buchanan, who had been carefully following these events while primarily occupied with DENDRAL program development and with knowledge acquisition from Djerassi's chemists.

In 1967, Buchanan responded to the crisis of complexity by a sustained piece of thinking and programming that lasted six months in which he did what Waterman was not able to do: he recast the entire knowledge base of DENDRAL into the form that we now call production rules.

There is an important lesson here for experimental science. Buchanan succeeded where Waterman failed because Buchanan was immersed in the details of the chemistry, the knowledge representation problem and the programming of the reasoning process. Waterman was only an onlooker. The immense importance of the experimental method in AI, and more broadly in CS, is that it provides the necessary mental data in sufficient detail to stimulate innovation and discovery. Perhaps it's easier to discover new ideas than to invent them!

The production rules of DENDRAL were different from Newell's productions and were used in different ways. The recasting of DENDRAL's knowledge into a separate knowledge base of production rules marks the invention of production rules as a representation of knowledge for knowledge based systems. They were for twenty years, and perhaps still are today, the most widely used representational form for the knowledge of expert systems.

The Major Scientific Result: Knowledge-is-power Hypothesis

DENDRAL was each week solving increasingly more complex problems. By 1968, hundreds or perhaps thousands of cases had been run through the program. It became time to organize the insights for the AI science from all of
this experimental system building work. The handy venue was the annual Machine Intelligence Workshop at the University of Edinburgh.

That jointly authored paper was in retrospect probably the most important paper of my career (Feigenbaum, et al, 1971).

The theme of the paper was generality (of method) vs power (to deliver effective problem solving). The "mainstream" in problem solving research was oriented toward general problem solving methods: on the nonmathematical side, the General Problem Solver; and on the side of mathematical logic, problem solvers based on theorem proving using the resolution method. The results of the DENDRAL experiments were being presented against that background and as a contrast to the mainstream.

The title of the paper was chosen to signal that to the reader. It was an echo of the title of a well known book of its time, the book by Ernst and Newell on generality and problem solving (Ernst and Newell, 1969)

Our major experimental generalization was presented and defended:

that in a program's knowledge lies its power; that to achieve high levels of competence in performance, AI programs must be knowledge-intensive.

This theme was elaborated in my 1977 IJCAI invited paper, bringing together the results of another 9 years of work. Analogous to Donald Knuth's use of the word "art" in "The Art of Computer Programming," this paper cast knowledge engineering as "The Art of Artificial Intelligence" (Feigenbaum, 1977).

Because so much additional evidence had accumulated from all the knowledge-based systems built in the 1980's, the theme was recast as the Knowledge Principle of AI in a 1987 IJCAI invited paper that I coauthored with Douglas Lenat (Lenat and Feigenbaum, 1987).

The Rise of the Term "Expert System"

The 1968 paper was the first place in the literature where the word "expert" was used in conjunction with these systems. The conjunction caught on and the felicitous phrase "expert systems" came into use at Stanford and then quickly in other places.

The 1968 paper opens as follows:

"In discussing the capability of a problem solving system, one should distinguish between generality and expertness. Generality is being questioned when we ask: how broad a universe of problems is the problem solver prepared to work on? Expertness is being questioned when we ask: how good are the answers and were
they arrived at with reasonable cost? Generality has great utility in some ways, but is not often associated with superior performance. The experts usually are specialists."

In the next paragraph, speaking of the DENDRAL program: "For some families of molecules, it is an expert, even when compared with the best human performance.

Later: "The Predictor program is an 'expert' on the general theory of mass spectrometry."

Finally, in the conclusion section:

"It is observed that the transfer of expertness between specialty areas is slight. A chess master is unlikely to be an expert algebraist or an expert mass spectrum analyst, etc. In this view, the expert is the specialist, with a specialist's methods and heuristics."

Exploring A Byway of Expert Systems History: Names

Being first into a territory provides rich opportunities for creative naming, though the origins of names are often obscure, clouded or controversial.

I have often been asked about one of those names: knowledge engineering. My friend Professor Donald Michie and I both have claimed invention of the name, which is not surprising since we had been chatting together about such things at the Machine Intelligence workshop in Edinburgh in 1968. John McCarthy had earlier referred to AI as "applied epistemology." Donald reflected to me that the line of work represented by DENDRAL should therefore be called "epistemological engineering." This was a good name because it correctly captured the spirit of the enterprise. But it was too complicated, arcane, and academic for people to remember and use. Because the knowledge-is-power theme was by this time emblazoned on my forehead, I simplified "epistemological engineering" to "knowledge engineering." Such is my recollection. Michie must have made the same leap independently.

There is one naming that no one has ever asked me about, so I will volunteer the story. The term is "knowledge acquisition" and the time was early 1973. By that time, the primacy of knowledge as the source of competence in AI programs was the solid foundation of all of my thinking about AI. As happened periodically, both the field of AI and the term "Artificial Intelligence," were under attack, as were other anthropomorphisms (such as "learning") suggesting human-like mental activities. I was asked to write a document for the Director of DARPA presenting our field's view of "AI: Where are we and where are we going?" In my outline, I consulted the slogan emblazoned on my forehead and broke the field down into three main parts:
a. the activities involved with the knowledge bases
b. the reasoning processes that used the knowledge to solve problems
c. the activities involved in getting the knowledge into the knowledge bases.

The first of these already had the name "knowledge representation." I struggled to grasp similar names for the other two.

For the processes that used the knowledge to solve problems, I coined the phrase "knowledge utilization."

For the third category, the anthropomorphisms were teaching and learning: teaching, if we humans crafted the knowledge and inserted it into the KB; learning, if a program was able to infer the knowledge needed by itself. Indeed, the metaphor of teaching had already been used by Minsky in 1964 in the Introduction to the book *Semantic Information Processing* (Minsky, 1964); and machine learning was a well-used phrase.

To me these were two sides of the same coin: getting knowledge into the knowledge base, either automatically or manually—in short, acquiring the knowledge, either by program or by knowledge engineers. I therefore created the term "knowledge acquisition" to go with knowledge representation and knowledge utilization.

Well, some things have a life of their own and some do not. Knowledge acquisition is still with us, although it has only part of the dual meaning I intended. Knowledge utilization was never used by anyone and it died a quick death.

Knowledge acquisition is now taken to refer to the work of the knowledge engineer. When machine learning research had a renaissance in the 1980's, the researchers felt the need to recapture a separate term for the work of programs that did the acquisition job, and their annual workshop publications revived the usage "machine learning" in preference to what I consider to be the more technically descriptive usage "automatic knowledge acquisition."

**Fast-Forward Through the First Era of Expert Systems**

I will end my talk with a look at the future—the "vision thing." But before I move to the future, I want to make some observations about the period in the history of expert systems that you know best: the period from about 1975 to 1991, the period that I call "the first era" of expert systems. This is the period that includes the major academic experiments in technology development and then
the first wave of industrial adopters, now numbering in the thousands of companies.

These observations sum up for me the first era of expert systems:

First, the application of expert system technology has brought enormous economic benefit, improving productivity or quality (or both) by a factor of ten or more (often much more). In technology, a change of one order of magnitude is thought to have revolutionary effects. Consider that the average driving speed is about one order of magnitude faster than walking speed; and the jet plane is about one order of magnitude faster than the car.

I first noticed the “order of magnitude effect” of expert systems while doing interviews for The Rise of the Expert Company (Feigenbaum, et al, 1988). In almost every expert system application examined, some dimension of economic gain was improved by a factor of ten or more. I have looked for this in hundreds of other systems, and almost always I find the factor of ten (or more).

When you think about expert systems, think of a remarkable technology that improves by an order-of-magnitude the human professional and semi-professional work that once was thought to be beyond the help of computerized information processing.

Second, what turned out to be of the greatest importance and interest during the first era was not the dance of the technologists as they manipulated the methods to fit the problems; but rather the dance of the managers, the champions, and the antagonists as they sought to advance or hinder the insertion of expert system technology into the business mainstream.

When my coauthors and I wrote The Rise of the Expert Company, I thought the book was about expert system technology. It was not. It really was about technology insertion. Those who watch the expert system field seem to agree: they have seen very few cases of failure of the technology, but many cases of failure of the technology insertion process.

Third, and most important as I move toward my look at the future, the thousands of expert systems that have been done provide massive overwhelming evidence for the soundness of the knowledge-is-power hypothesis. By 1987, I began to elevate its status by calling it the Knowledge Principle:

A system exhibits intelligent understanding and action at a high level of competence primarily because of the specific knowledge that it contains about its domain of endeavor.
The reasoning processes of an intelligent system, being general, and therefore weak, are not the important source of power that leads to high levels of competence in behavior. The Knowledge Principle simply says that, if a program is to perform well, it must know a great deal about the world in which it operates. In the absence of knowledge, reasoning can not help performance.

The Future of Knowledge-Based Systems

Imagine this image for a moment: a mesa-like tower of knowledge, very high, very narrow. It doesn't cover much acreage in the vast fields of knowledge. Our expert systems operate at a high level of competence at the top of the tower of knowledge, until they come to the knife-edge cliff. If they exceed the bounds of their tower of knowledge, they fall immediately to levels of zero competence.

The expert systems of the first era are brittle. There is no ability to create a parachute for a soft fall by using more general knowledge of perhaps a non-specialist sort to do some kind of problem solving when the detailed expert knowledge is lacking.

Expert systems of the first era are also isolated. The mesa-like towers of knowledge stand lonely and apart like the mesas of the American Southwest. I believe that in the entire first era of expert systems there has never been a case of two separately engineered knowledge bases interoperating at a later time to solve a bigger or a different problem.

Every expert system appears to be a custom-crafted cottage industry event! One might wish for a kind of Library of Congress of knowledge codified and represented for expert system use, to provide at least partially automated assistance in the construction of new expert systems. No such national knowledge base exists today. The dream of having one barely exists.

These two shortcomings of expert systems of the first era—brittleness and isolation—provide shape for the next decade of knowledge-based systems research. While the technology of the first era is being inserted, improved, and applied, a second era of knowledge based systems is being invented.

The thrusts of the second era research are concepts of large knowledge bases, knowledge sharing, and the interoperability of knowledge bases that are geographically distributed.

Large Knowledge Bases

The typical expert system knowledge base of today represents a few hundred to a few thousand facts and heuristics; seldom in the tens of thousands; rarely (perhaps only two) in the hundreds of thousands of knowledge elements. Consider the following example. To provide the general knowledge that would
form the epistemological bridges between the many expert systems in medicine, i.e. to provide basic medical knowledge about human biology, biochemistry, physiology, anatomy, disease processes, psychological processes, and so on, would require a knowledge base of many millions of individual but interlinked units of knowledge. To build such knowledge bases, we are facing a huge task of knowledge engineering and knowledge acquisition without much computer assistance.

Help is on the way. The knowledge based systems research community is gearing up for a very substantial effort at inventing the technology for the development of very large knowledge bases; struggling with the difficult problems of scaling up; and inventing the necessary tools and infrastructure to enable knowledge sharing.

Knowledge sharing, to these researchers, means more than just the knowledge bases of several expert systems interoperating to solve a problem. Knowledge sharing also means the computer-facilitated cooperation of many people in the building of a large body of codified knowledge. The vision is that hundreds or thousands of knowledge base builders would cooperate.

A vision this grand, or should I say grandiose, needs good stories to make it tangible and credible.

Lenat, Guha, and their CYC group have been building the largest general purpose knowledge base the AI world has ever seen, developing knowledge sharing methods and technology that are unique to their substantial group of scientists and "knowledge-enterers," as they are called (Guha and Lenat, 1989). CYC's knowledge base representing a subset of "common sense" contains millions of facts, beliefs, and other bits of knowledge. The intention is that when complete it will codify a major portion of consensus common sense knowledge; and will be at the foundation of a wide variety of new computer applications whose success requires that a program exhibit some reasonable common sense. Typical of the applications being planned are natural language understanding and data base maintenance.

In Palo Alto, California, at and near Stanford University, another group of researchers is approaching large knowledge bases from a different direction. First, the knowledge itself is technical, drawn from electrical, mechanical and software engineering and from computer science. Second, the work starts from the premise that the large knowledge bases we seek will be built by many people, in many places, on many subjects, each with his own ideas about knowledge representation. Yet these knowledge systems will collaborate to solve problems. The vision is a distributed national (or international) knowledge base.

Such a journey of a thousand miles, as the Chinese say, must begin with a single step, and that step is the recent work of PACT, the Palo Alto Collaborative
Testbed (Mark, et al, 1992). PACT's members include the Knowledge Systems Laboratory of Professor Fikes and myself; Professor Genesereth's Designworld Project that includes Hewlett Packard scientists; Professor Cutkosky's NEXT CUT Project and knowledge system; and the NVISAGE Project of the Lockheed Artificial Intelligence Center.

For an initial effort, the problem to be solved was the redesign of a robotic manipulator (i.e. robotic arm). The knowledge systems participating in the problem solving included:

- NEXT CUT, that knew the original mechanical engineering design of the arm; and knows dynamics and machining.
- Designworld, that knows digital electronics; and also does a parts catalog service, looking up commercially available engineering parts.
- Device Modeling Environment (DME), that knows about power supplies and motors; knows about the sensors that detect movement of the arm; and also knows how to do qualitative and quantitative simulation of devices.
- NVISAGE, that designs controller software.

A low level knowledge exchange language, KIF, has been worked out, as well as a knowledge sharing communications protocol KQML. These combine with a network to constitute what we call a "knowledge bus." Also agreed among the collaborators are sets of terms and relations for entities and concepts, called shared ontologies.

For the October 1991 demonstration, the cooperative task was one of redesign. NVISAGE, designing the controller software, sent a request to the other systems for knowledge of the dynamics of the robotic arm. NEXT CUT fielded the request and sent the information. A cycle of simulation involving all the systems was begun, in which each specialist system simulated the aspects of the arm within its areas of specialization.

DME found that the motors would burn out in the redesigned arm. A power subsystem designer (human) replaced the motors with a larger model. The change notice was received by NEXT CUT, which noted that the fixtures that hold the motors might need to be redesigned. NEXT CUT then sent a request for structural data on the new motor. The request was handled by Designworld's catalog service; it sent back geometric information to NEXT CUT, which then redesigned the fixtures and prepared a process plan for their manufacture.

Fifteen different computers, located at various sites on or near the Stanford campus, were actually used in this distributed engineering environment. Each
system ran on its own hardware and software base. Each system had its own knowledge base, which was a deliberate design decision to permit scalability. There was no central knowledge base which all systems accessed.

The PACT work is a fragile seedling, but I believe that it marks the beginning of the national knowledge base, at least for engineering and science; and when joined with CYC (which I hope will happen some day) for general common sense knowledge as well.

Books That Talk To Each Other

Several years ago, a small workshop met to formulate the national knowledge base project and to discuss the Library of the Future. Attending the workshop was Professor Marvin Minsky of MIT, one of the founders of Artificial Intelligence. He and I engaged ourselves in an envisioning exercise, putting our virtual selves into the library of 50 years hence, trying to imagine what it would be like. Minsky said something, half in jest but profound, looking back at today from the vantage point of fifty years out: "Can you imagine, they used to have libraries in which the books didn't talk to each other!"

As we work to invent the second era of knowledge based systems, we are on the road to the books that talk to each other, to the library of the future.
Bibliography


