# Oral History of Edward Feigenbaum

Interviewed by:
Nils Nilsson

Recorded: June 20 and June 27, 2007
Mountain View, California

CHM Reference number: X3896.2007

**Nils Nilsson:** Today, on June 20th, 2007, we're having a conversation with Ed Feigenbaum, who is the Kumagai Professor Emeritus at Stanford University. Ed is going to be talking about his life in artificial intelligence. Before he starts, let me say that Ed has had a great career in artificial intelligence. He's known as the father of expert systems. He also is a Turing Award winner, and a member of the National Academy of Engineering. Let's start, Ed, by the subject of how you originally got interested in science. What are some of the early influences in your life, your education, high school, and so on?

**Edward Feigenbaum:** Going back to the time when I was a youngster, and that means maybe starting at eight or nine or ten years old, I'm not exactly sure which, there were a number of small influences in my life. I don't want to go into great detail about it. My stepfather, Fred Rothman [ph?], would take me once a month to the Hayden Planetarium in New York City, which is part of the American Museum of Natural History. We lived across the river from New York City in the New Jersey side of the Palisades. Once a month, at that time because there were different -- the costs of running a museum were not as expensive -- the Hayden Planetarium would change its show once a month. So once a month my stepfather and I would go over there to see the new show at the Haden Planetarium. Since we were there, we would go through one room of the Natural History Museum every time we went there. I got really interested in science, but mostly through the astronomy end, which I guess happens to a lot of young kids. It also happened that my stepfather was an accountant by trade, and he would bring home a lot of work. He would do it at home on one of these mechanical calculators, which for people who visit the Computer History Museum, you could see examples of that. Some of them are electric driven. Actually, the one I started playing with when he had it was not even electric driven. It was mechanical driven -- cranking of the wheels. I learned to really know how to use that machine, and I was very proud of it. I would show it to my friends. Therefore I kind of had a bias for being interested in calculating machines, which later turned out to be an interest in computing, eventually. But I was a science kid. I just would read *Scientific American* whenever I could get -- I mean, every month I could get a hold of it. Go the library and get a hold of it, and I would read everything I could. I remember, as so many other people of my generation have said-- and the previous generation -- one book that really, really sucked me into science was "Microbe Hunters." We need a lot more books like "Microbe Hunters" that will bring a lot more young people into science now.

**Nilsson:** So these interests in science continued through high school?

**Feigenbaum:** Right through high school. They were my best subjects. Well, I was… My best subjects were everything. That is. I got A's in everything, but the ones I really enjoyed the most were math and physics and chemistry. Not so much biology. I don't know whether that's a function of myself, or whether the teacher was not so good. But I would get a big thrill when the math teacher would give us a problem. I'd go home and I remember I'd work on it late into the night. My parents would want me to go to sleep, and I just hadn't cracked that theorem yet. When I did, I was so happy.

**Nilsson:** So toward the end of high school, when you began thinking about college, how did you decide where you might want to go, what sorts of things you might want to study?

**Feigenbaum:** Nils, that's a great question. Here I am in New Jersey, on the Palisades overlooking New York City. A high school which was an academic-oriented high school. I was a straight "A" student. They have an academic advisor-- a woman who sort of gives you the clue as to what you should do. The initials MIT never came up. Not once in any conversation. You would think that a kid like me would normally gravitate toward MIT and apply for a scholarship or something like that. Wasn't even mentioned. I happened to see an advertisement for what were called at the time the "Westinghouse Scholarships" being offered by an engineering school in Pittsburgh called Carnegie Institute of Technology. Since I really needed money to go to school, I applied for one of those scholarships. It turns out that I got a Carnegie Tech scholarship, not a Westinghouse Scholarship, to go to Carnegie Tech -- four year scholarship -- so that's what I did. But nobody mentioned Harvard, no one mentioned MIT, no one mentioned Columbia, no one mentioned anything. The only other alternative that I thought of seriously was Stevens Tech in Hoboken, because then I could live at home and go to school.

**Nilsson:** So you went to Carnegie Institute of Technology in Pittsburgh. What year was that, and what did you decide that you wanted to study?

**Feigenbaum:** The year was 1952. September of 1952. The choice of subject matter was electrical engineering. Now, the question is: why was it electrical engineering? Why did I decide to do that, as opposed to going into physics? Well, first of all, around my family, no one ever heard of a thing called a physicist. It just wasn't-- the word never came up. Secondly, my parents knew that engineers could go out and get jobs and make money. What did I like best? Well, I really didn't like puttering around with mechanical things, so I wasn't a mechanical engineer. I wasn't really too interested in chemistry, so I wasn't a chemical engineer. Well, electrical things -- that was a lot of fun. That I could do sort of clean hands on the table. Turns out, that isn't quite accurate when you get to college, but I decided to be an electrical engineer.

**Nilsson:** As I remember from my days as an electrical engineer, you take a lot of courses in circuit theory. You have labs in which you test things with oscilloscopes. All of those things happened at Carnegie Tech?

**Feigenbaum:** Yes.

**Nilsson:** And you enjoyed those sorts of courses?

**Feigenbaum:** Yes, I really did. But there's a calibration we need. And that is that at Carnegie Tech, yes we did circuits, yes we did what you might call abstract electrical engineering, mathematical electrical engineering. But when you got into labs, it was almost entirely power electrical engineering. These were still the days when generating power was a big deal, transmitting power was a big deal, and that's where I said that the hands dirty thing breaks down. Because you had to go into labs and you had to stick giant prongs into big boards. And those prongs were handling tremendous amounts of amperage and voltage, and it was really scary. I didn't really want to do that. I sort of got my friends to go stick things in these big boards.

**Nilsson:** Lots of rotating machinery.

**Feigenbaum:** Yes, lots of rotating machinery.

**Nilsson:** Power transformers.

**Feigenbaum:** They didn't have electronics until one course given in two semesters of the senior year with a relatively old textbook, and it was vacuum tube electronics. Transistors hadn't really come in much -- they had been invented but they hadn't come much into the teaching of the discipline. So it was pretty much power engineering.

**Nilsson:** There were probably some mathematics courses, differential equations...

**Feigenbaum:** Yes.

**Nilsson:** …courses like that, too, as part of the EE curriculum.

**Feigenbaum:** The standard ones, yes.

**Nilsson:** Now I know that, eventually, you ran into Herb Simon while you were an undergraduate. How did that all happen?

**Feigenbaum:** I cruised through my freshman year. Interesting stuff. Got all "A's", but there was something missing. You're supposed to be going to a university for something other than the craftsmanship of calculus or of engineering. I felt something lacking. I asked my advisor if I could do something else. Can I just pick a course and do something else? Like, Carnegie Tech happened to have one of the world's best drama schools in the art school. They had a whole college of arts. Or they had a design school. They had a very good English department. Anyway, there were other things to take. I looked through the catalog and I found a really interesting listing called "Ideas and Social Change." Wow. Except it was being taught at a place I never heard of, which was on the other side of campus, called the Graduate School of Industrial Administration. And I was going to be a sophomore. It was being taught by a new instructor. Nils, you know him now from Stanford. He's now at Stanford, James March. I decided to go over there and take that, and my advisor said, "Sure, go ahead and see." March let me into the course even though I was an undergraduate. First thing he did was to expose us to Von Neuman and Morgenstern's Theory of Games. Wow! This is mind-blowing! I couldn't believe this. Then we had a series of things like that, in ideas in social change, including modeling of behavior. I got really interested in that. March was the one who, first of all, employed me in the summer of my sophomore year to do experiments in social psychology. In fact, my first published paper was a paper with March in social psychology, on the decision-making of small groups. Secondly, he introduced me to a more senior and famous professor who was working with him on a book which, of course as you know, was March and Simon's "Organizations". Herbert Simon. That's how I got to know Simon. Simon took an interest in me, helped me get a summer student fellowship in the summer of my junior year. That led to my taking a course from Simon in the first semester of my senior year which was called Mathematical Models in the Social Sciences. Herb Simon had done many things in his career. And after that, of course, he did spectacular things in his career. But he had accumulated a bunch of modeling exercises in social science which he published around 1956 called "Models of Man". It included differential equations, models of social behavior, economic models, decision-making models, including two very famous papers that influenced the early days of AI. One of them was a behavioral theory of rational choice, which introduced the ideas of search and heuristics. Then the second one was “Rational Choice and the Structure of the Environment” which introduced-- it basically was the prelude to Simon’s famous analogy, a metaphor, of the ant on the beach. The environment shapes a great deal of behavior. Those two papers were in that book. And stochastic models were the last third of the course and part of Simon’s book. So Simon was teaching these modeling methods and modeling motivations -- techniques and so on -- to this small group of students. I’ve been trying to remember who exactly was in that group. I have a feeling that my later colleague, Julian Feldman, was in that group, and also perhaps Fred Tong [ph?] was in that group. But it was about six graduate students. Not me, I’m sorry; five graduate students and myself making six -- I was a senior -- making six students in the class. That was the particular class that I tell the famous story about, where Herb Simon comes in and, right after New Year’s-- it’s a semester kind of course, so it continues on after Christmas-New Year’s for awhile. And he comes in and says, “Over Christmas, Alan Newell, and I invented a thinking machine.”

**Nilsson:** This was in January ’56?

**Feigenbaum:** It’s January ’56. It turns out Herb dates the particular date as the peak of his career was December 15th, 1955. In his autobiography, he says that’s where he hit the top of the hill. He and Newell had formulated the Logic Theorist on December 15th, 1955. Then they did a quick bunch of hand simulations, and a little bit of non-machine programming -- paper-level programming -- and by the time we got back after the New Year’s holiday, he mentioned that he and Newell had invented a thinking machine. And that just… What could he possibly mean by that? First of all, what did he mean by a machine? And then, what could he mean by that machine doing thinking?

**Nilsson:** So was it about that time that you had your first experiences with computers? How did your contact and introduction to computers happen?

**Feigenbaum:** It’s a pair of events. As an undergraduate, I had been what you might call a science journalist for an undergraduate publication called *The Carnegie Technical.* By the time I was a senior I was editor of *The Carnegie Technical*. Or maybe when I was a junior I was editor of *The Carnegie Technical.* In any case, in the course of putting together some issues of this undergraduate publication, one of the issues concerned a new thing called a computer-- what that might be like. But that didn’t have much impression on me. Where the impression came in was when we asked Herb in that class, “What do you mean by a machine?” and he handed us an IBM 701 manual, an early IBM vacuum tube computer. It was their first scientific computer product. They of course had a lot of commercial computer products in the punch card area-- punch card calculators. But at one point, they introduced a binary machine for scientific calculation and a character-oriented machine for commercial calculation. The 701, 702. They were both vacuum tube computers. Herb gave us a manual, and that was a born-again experience. That was, taking that manual home, reading it all night long. By the dawn, I was a born-again -- or I shouldn’t say born-again -- I was just hooked on computers and decided what I had to do was, first of all, got to learn more about these things. Carnegie Tech did not have any computers at that time. So I got a job at IBM for the summer of 1956 in New York. Had to earn some money for graduate school. But where was I going to do graduate school? Well, I was going to do graduate school with Herb Simon. Oh, and his colleague, Alan Newell, who-- I’ll tell you a little bit about that. But, I was going to stay with Simon and get my PhD at Carnegie in the Graduate School of Industrial Administration.

**Nilsson:** Was getting a PhD on the horizon for quite a long time while you were an undergraduate, or was this is a sudden decision as a senior?

**Feigenbaum:** Completely sudden decision. A tremendous shock to my parents. They had never heard of this idea, and anyway, engineers could get a job and earn money, and why were you going after a PhD when all that was was just a fellowship that keeps you in school? That was a kind of bizarre idea for my family. It just sprung up at the time when I got involved with Herb in this AI work. Or maybe a little bit before when I got involved with March and Simon on various social science things, and the idea that if you wanted to continue to do that, you'd better get a PhD. But it certainly wasn't a long-term plan. During the winter and spring of my senior year, I was actually doing interviews. I actually came out here to Palo Alto to do an interview at a place called, I believe it was Raytheon at the time had a Palo Alto location. But that was just to be an engineer.

**Nilsson:** So then in the summer between your undergraduate and undergraduate work, you were at IBM having a summer job?

**Feigenbaum:** Yes.

**Nilsson:** How did you do that?

**Feigenbaum:** I went to New York. I could live at home then. IBM sent me to the Watson Laboratory at Columbia where they had a 650. And at Watson Lab, I learned how to program the 650. Oh, because by that time it became known that Carnegie Tech was going to get a 650 in the summer, so I better learn the 650 so I can go back and be a user of it. Secondly, they taught me, believe it or not, wired board programming. I'll bet almost no one hearing this question and answer session knows what a wired board program thing is. But it had 16 registers and a lot of wires that transferred signals from register to another. You had 16 instructions and you could manipulate [punched] cards. They taught me that. Then they took me back to downtown, near 590 Madison Avenue, around the corner in a brownstone, where the graduate students were for the summer, and taught me the IBM 704, which was a successor machine to the 701, which was good because its architecture was virtually identical to the 701. I think maybe there was an introduction of index registers to the 704 that the 701 didn't have, but other than that, it was easy for me to program that. Then I did work for them in the summer.

**Nilsson:** How did you get interested in programming other than these wired boards and so on? What was your first programming language, and when did you start getting more involved with the sorts of programming languages used in AI?

**Feigenbaum:** There were two things essentially going on in parallel that constituted my first important programming experiences other than the summer job at IBM. When I got back to Carnegie Tech in September 1956 and began my graduate work, there was a 650 there, and there was a lot more than that. There was Alan Perlis, a wonderful computer genius. Later, I think the first Turing Award winner was Alan Perlis. And two other people who were assisting Al Perlis in setting up this IBM 650 for Carnegie Tech, Joe Smith and Hal Van Zorn. Later they were joined by Art Evans, who a lot of us know as one of the big contributor to early computer science at Carnegie Tech. Perlis was finishing up an amazing thing called a compiler, which was IT-- Internal Translator. It actually preceded Fortran out into the user space by about nine months. It was just about done. It occupied 1998 words on a 2000-word IBM 650 drum. So there were two spaces left, and lots of bugs to fix that needed fixing, and they couldn't because they only had two spaces left on the drum. It was an algebraic language, and you could write complex algebraic things in it. I had known about this idea because, at the brownstone in the summer at IBM, a guy had come down from the fourth floor of this brownstone to talk to the graduate students and tell them about a new thing that had just hit the scene. They were doing it upstairs. You didn't have to write these "CLA" for "clear and add", and you didn't have to write "OO5" for add. You could actually write algebraic things, and it was a formula. And there's a program that would translate that formula into machine language: FOR-TRAN. And that was John Backus who had come downstairs to talk to us. So I actually knew about this idea. But Perlis actually had one up and running. So I was able to write a rather complicated, for that time -- now it would seem like a freshman/undergraduate exercise, but at that time, it seemed to be complicated -- a simulation of two companies engaged in a duopolistic decision-making dual about pricing of tin cans in the can industry. Why did I get involved with that? Because if I were going to get a PhD in the Graduate School of Industrial Administration, I sure had to pass some exams. One of the exams I had to pass, I knew, was going to be an exam in economics, which includes micro and macroeconomics. I figured I'd better take a course in this. I was taking a course from Dick Cyert. Cyert was teaching, I think at that time, microeconomics.

**Nilsson:** Cyert later became president.

**Feigenbaum:** Cyert later became president of Carnegie Mellon. Simon's ideas were blooming at Carnegie, including the earlier paper on behavioral theory of rational choice. Then it was blooming into a model of human decision-making, individual human decision-making and problem solving. I needed to do a paper, and I talked to Cyert about what kind of term paper I should do for this course. We decided to try to do one of these Simon-like satisficing models, search and satisficing models for the decision-making behavior of companies, not just individuals. Since that's complicated for lots of companies, let's just do two companies. That's a duopoly. There were some famous duopolies that economists at the time knew about, where there was data involved in how they behaved. In particular for our thing, pricing behavior. So I did one of those. Of course, I stumbled through it, not knowing a lot of economics. Cyert was a very fine advisor, but he quickly got Jim March involved with it. So the three of us were doing this modeling piece of work. In that case, it led to the first paper that I actually delivered at a scientific conference. It was in the fall of 1958. Cyert and March asked me if I wanted to deliver the paper on this at the American Economic Association annual meetings in December. So I actually went there and delivered a real professional paper on this piece of work. The piece of work was published in 1959 in the journal *Behavioral Science*. It had gotten rejected from *The American Economic Association Journal* in '58 because it was about a subject that was not considered to be economics, namely simulation. Simulation was not a way to do economics. By '59, they had already decided to publish a special issue on simulation. So that was the first big programming job I did in non-AI formats. Then go back to the AI question, remember IPL-I was a paper language, the language in which the logic theory program of December 15th, 1955 and thereafter…

**Nilsson:** That was the computer that thought. That was the thinking machine.

**Feigenbaum:** That was the thinking machine that Simon was referring to. The Logic Theorist, essentially the first heuristic program. Newell and his colleague, Cliff Shaw, at the RAND Corporation, quickly began to implement IPL-I in a real computer. That was the Johnniac computer. Johnniac computer is actually on display at the Computer History Museum. They've redone it beautifully.

**Nilsson:** So what does IPL stand for?

**Feigenbaum:** Information Processing Language. It was the first language that was dedicated to symbolic manipulation, as opposed to numerical calculation. It didn't use algebra as it's metaphor, with plus, minus, times, and divide. It used symbols and lists of symbols as its metaphor. In later papers, Newell and Simon actually tried to make more than metaphoric use of this in trying to describe how the human memory would be organized. I'm not sure that was a convincing metaphor. But these languages were oriented toward comparing symbols, organizing them into serial lists, organizing them into… Since every list had a symbolic name itself, therefore it was natural to have lists of lists which, of course, led immediately to the recursion idea. Programs themselves were symbols, so you could just execute these lists of symbols by throwing the name of the program into the interpreter. It was an interpretive language. You could have things called description lists, which were pairs of attributes and values. The attribute was a symbolic entity, and the value itself didn't need to be one symbolic entity, it could be a list of symbolic entities. So you could have an attribute of color and then you could have a value which is a list of the different of colors a thing could be: blue or red or purple or something like that. Eventually, the attribute-value list idea became very important in not only list processing, but that particular version of it became very important in objected-oriented programming.

**Nilsson:** Now these ideas, both the Logical Theorist and IPL, both of them made quite a hit at the Dartmouth Conference in the summer of '56. Were you involved at all in Newell and Simon's going up there, and what are your recollections about their ideas from that conference?

**Feigenbaum:** I was not involved. I didn't know about it. I only heard about it after they got back from actually a pair of conferences of which the Dartmouth conference was one. They talked a lot about it. I found out then that they had basically taken the Logic Theorist to the conference, with the IPL idea. They spoke of some of the reactions of some of the other people, particularly Herb Gelernter, who got very excited about this. Herb Gelernter of IBM got very excited about all these ideas and quickly proceeded to implement both a language, which was a list processing language in Fortran, and an application of it to Euclidean geometry. So they talked about that. They also talked about what they thought was a spectacular paper. They gave another paper at a conference of the IEEE -- the IEEE Information Theory Proceedings, or something conference in September of '56 -- and they heard a spectacular paper by Chomsky. That was Chomsky's original paper on trying to frame computation linguistics. So the answer is: I didn't know anything about it. I had no information prior. Afterwards, just a small glow but nothing more than a few days worth of discussion about it in around Carnegie and then it disappeared into the void.

**Nilsson:** So here you are, a graduate student in the Graduate School of Industrial Administration at Carnegie Tech. And of course, one of the things that one does as a graduate is settle on research and a dissertation topic. Can you say a little bit about how you decided on what to do about all of that?

**Feigenbaum:** So, I got back in September of 1956 to start the new year, and I walk into Herb Simon's office saying, "I'm here. I have this fellowship What do I do?" He opens up the current issue of

*Scientific American* or maybe the issue before last or something like that. And he shows me an article about human learning. It was specifically about human serial learning, and it was about a thing called the serial position effect. It had to do with learning of nonsense human syllables, that is, trigrams which had a vowel in the middle and two consonants on the ends, and otherwise made no sense at all, except that the psychologists had carefully, over the decades, graded them in terms of meaningfulness.

**Nilsson:** Were they pronounceable?

**Feigenbaum:** Yes, they were pronounceable. Like the ones that I use a lot in my book, in my articles, is DAX, D-A-X, and JIR, J-I-R. But otherwise, had no meaning. There were, it turns out, quite a set of phenomena associated with this part of human psychology called human verbal learning, verbal meaning word-like things, especially human nonsense syllable learning. The experiments went back to a guy named Ebbinghaus back in the late 19th century, and continued through the particular moment we're talking about, 1956, with massive amounts of results. An article in the "Handbook of Experimental Psychology", I think by Carl Hovland, famous psychologist from Yale. And Simon said, "Okay, let's make sense of this. Here's the data. This is elementary learning. We can't ask for anything simpler than this. This doesn't have meaning associated with it. It must have something to do with memory structures. Let's see if we can construct an information processing model, basically as simple as we can, to explain this data". By "explain" he meant numerically explain -- predict these numbers that the psychologists felt were stabilized numbers. That is, they would talk about them in terms of something like the serial position curve, was a real curve. It had real numbers at each end and it varied with the meaningfulness of the syllables.

**Nilsson:** By explaining it, did he have in mind that the model that explained it would be the model of the way the brain actually does this?

**Feigenbaum:** No, not the brain. That's a very important thing, Nils, to mention. Never, ever was the brain brought up. This was totally a model of mind, a model of human information processing with symbols at the lowest levels. Now this shows up, in very great elaboration, in the 1972 book by Newell and Simon called "Human Problem Solving", where they go into great detail about what they call elementary information processes, EIPs. But it also comes up in their Turing Award speech in which they focus on the level of symbolic manipulation as a level of explanation for what's going on in the mind. There never was any discussion about brain theory, even though there were books at the time like, I think, it was Ed Berkeley's book called "Giant Brains." The metaphor was running around, but it never cluttered the scientific air at Carnegie. I've not only lived with that metaphor, I've adopted that -- I'm sorry, not that metaphor, that level of explanation of human problem solving activity -- over my whole career. There's now a new book out. And that's become common among most AI -- no I shouldn't say most, what you would call the mainstream of AI thought for three decades or so, has been at that level of explanation. There has been a bit of work starting in '78 or so, where people were looking at what does the brain really do. They looked at what's a model of neuron, and how do neurons connect. Then came the neural networks approach. But the symbolic processing level of explanation of human intelligence, namely the mind explanation of human intelligence, has dominated.

There's a very recent book -- for those of you who are watching this fifty years from now, I'm talking about 2007 -- by Doug Hofstadter called "I Am a Strange Loop", in which his main point is to emphasize the symbolic level of explanation. He uses as a metaphor a view that many of us have been using over decades, which is that it's a lot different to explain pressure in physical terms, let's say at the macro level where we do PV=nRT, than…. It's a different level of explanation than the statistical mechanics where you look at individual molecules hitting up against the wall of a container. And that, for now, the level of explanation of symbolic manipulation is the right level of explanation. Not that it can't be informed by brain research, but that level of communication between neurophysiological work and psychological work hasn't been, up till now, very productive.

**Nilsson:** So your work then, after talking with Simon about attempting to build this kind of a model -- what happened next?

**Feigenbaum:** A model called EPAM1. EPAM incidentally is a name of a program or a sequence or programs, stands for Elementary Perceiver and Memorizer. Hence indicating that we were looking both at the question of how is a symbol perceived, and then how does it, on the basis of that perception, retrieve memories, stored memories. EPAM1 was a simplistic-- no, sorry. Back off that. It was a simple model of why the serial position effect, which is the numerical curve that indicates people paying attention to symbols at the ends of lists first, rather than the symbols in the middle of the list. And it has a very replicatable shape. What's the mathematical model that predicts that? And also, what's the computer model that is equivalent to that mathematical model that predicts that? That was EPAM1. But that didn't say anything about the details of the perception and memorization of the individual nonsense syllable. So EPAM2…

**Nilsson:** It was more a description and not a simulation?

**Feigenbaum:** No, it was a simulation actually. In fact, the appendix to the paper we published in *Psychological Review*, which is a main theoretical journal of psychology, the appendix to that paper actually gave the mathematical model that predicted the same as the simulation model. The simulation model,I don't remember what language it was written in. I didn't finish my story of the IPLs, I can see. But it may have been written in IPL-II at the time. I'm not sure. I have to go back to the paper and look. But anyway, I have to go back to the IPL story after this. I then took up real lists, not just positions of things in lists, but actually syllables, and at that point--

**Nilsson:** Perhaps we ought to continue that on the next tape.

END OF TAPE 1

**Nilsson:** Let's continue our conversation by going back to talking about the IPL languages. I think you had mentioned something about IPL-I, that was a kind of a paper language. There were actual languages, too, so how did that all go?

**Feigenbaum:** To actually implement the logic theory program, Newell and Simon and their colleague Shaw at the Rand Corporation did an interpretive language for the Johnniac computer. The Johnniac computer had a relatively small amount of memory, I think maybe 4K at the time. They did a language which was very much symbolic-manipulation oriented, with lists and list structures and attribute-value lists, but it had very much of a serial programming flavor to it of the typical variety of the path on which computing had come. And that worked fine. They actually had a running program that proved theorems interestingly in propositional logic. But Newell and Shaw were very dynamic about changing the language and morphing the language. In other words, they viewed language construction as an area of scientific and engineering discovery. It just wasn't a tool, it was something to look into. The next thing to look into was IPL-III, which changed the mode rather amazingly. That is, it took an extremum, and that was essentially trying to do everything by nested subroutines and recursion. Now, recursion was possible in the old IPL-II structure, but it was not convenient, and you didn't think of it a lot. In IPL-III, you didn't think of… there was no other way to do it.

**Nilsson:** Was that the beginning of pushdown stacks? Was that?

**Feigenbaum:** I don't remember if it was IPL-III or IPL-II that had the first pushdown stacks.

**Nilsson:** Which is basic to recursion.

**Feigenbaum:** Yeah, but I think it was in IPL-II, because recursion was possible in IPL-II. But anyway, IPL-III made a big deal of it. That was going to be the central element of this. Well, as with all -- not all, many -- experiments, you can push a thing to an extreme, and it gets to be awkward, and it's beautiful, but not convenient and practical. IPL-III really was beautiful, but who the heck can make sense of these kind of programs? You see this whole thing replayed during LISP. LISP, when McCarthy got around to writing an elegant notation for all of this in LISP, McCarthy's view was you've got to write this out in recursive form. This is exactly why I did it, John would say. Then you've got this rebellion about, "We can't do this. We can't think that way. We've got to have a PROG function, which enables us to think serially like a programmer. So you find in the LISP world the same tensions going on that went on the IPL-III days. Well, it came time, during all this IPL-III work, to think about a public version of IPLs, because it was all on Johnniac. There was only one Johnniac in the world, and there was a copy of it-- I mean, Johnniac was a copy of the Institute for Advanced Study machine in Princeton, but that was it. So IPL-IV got born. IPL-IV was really backing off from IPL-III, in the direction of giving the programmer more flexibility. But it was still a programming language. It wasn't like an elegant high-level notation. So, I got the job... Newell asked me if I wanted to... I'd worked in the summer at the Rand Corporation in California, where Newell was still an employee of the Rand Corporation at that time. He hadn't yet taken an offer from Carnegie Tech.

**Nilsson:** So this is while you were a PhD student, you worked at Rand?

**Feigenbaum:** Yeah.

**Nilsson:** In the summers?

**Feigenbaum:** In the summers. And Newell asked me if I wanted to be the programmer for IPL-IV. And sure, that was fun. I was a good programmer.

**Nilsson:** You were in the middle of working on EPAM at the time?

**Feigenbaum:** Yeah, I was in the middle of working on EPAM. To be part of a project, it was a big deal, to do this IPL stuff and working with Newell and so on. We actually did the manual first, just so that we wouldn't have a moving target while we were doing a language. We had a crystal clear idea of what we wanted to do and then...

**Nilsson:** Those were the specs, the manual.

**Feigenbaum:** The manual essentially became the specs. But it was actually a manual. It was actually published. It was the first time I ever got my name on the cover of a book. It was Newell, I don't know, Newell, Shaw, Feigenbaum and Meely, or something like that. George Meely was a programmer at Bell Labs who did the I/O for the system. I believe this was a 709. I believe by the time we got it done, it was a 709 or 7090 implementation. But I never was very good at I/O, so George Meely did the I/O and I did the rest of the programming. And IPL-IV became a public language. Then, of course, LISP was born right around that time, roughly speaking, 1959.

**Nilsson:** So you were writing EPAM in some combination of IPL-II, IPL-III, IPL-IV?

**Feigenbaum:** Yeah. But, no, not IPL-III. I think I skipped IPL-III because I think it was really hard to program in that language. So, back to EPAM. EPAM-I was a relatively straightforward model without going into the details of how individual memory items are stored. EPAM-II dealt with real lists, serial lists of nonsense syllables and the other favorite paradigm of psychologists, nonsense syllables in associate pairs, where one is called a stimulus and the other is called a response. The order of the associate pairs are randomized for the subject's learning, just so the subject… so you don't confound serial learning with the learning of the pairs. You might think of serial learning as a degenerate case of stimulus-response, and the response becomes a stimulus for the next one, and so on. That's a subcase of paired associate learning. What makes one nonsense syllable different from another? Well, they have different letters. And what makes one letter different from another is they have different shapes. If we had a coding of the shapes in a method similar to what the computer vision people would have done two or three years from then, in the late '50s, early '60s, namely loops and lines and things like that, if we code up the shapes, then we can make a test that tests that one thing is different from another. If they are different… Let's say these two nonsense syllables have different first letters. There's, for example, no need to discriminate further… If no other nonsense syllables in the list have overlapping first letters, there's no need to make anything more than one test that discriminates some feature of one from some feature of another.

**Nilsson:** That's how an engineer would do it.

**Feigenbaum:** Yeah. It turns out that first of all… Because we had IPLs, that was the first time that it was absolutely natural to conceptualize that as a growing list structure. You have nothing at the beginning. The memory is not structured at the beginning, and all of a sudden it has a branch.

**Nilsson:** First letter.

**Feigenbaum:** That node has a symbol. And that symbol can become a branch, and that symbol can become a branch. It happens that we initially used binary trees. I initially grew binary trees, but binary was not essential. These lists can be more than two long. So, ternary trees are okay, and N-ary trees are okay. Then at the very bottom of these trees, you actually have the place where you can store the real information about this nonsense syllable, namely, the information you need to give it out through the mouth through the experimenter. You have to have it all. You can't just know that the first letter has a loop. You have to actually memorize the material.

**Nilsson:** Or to the printer if you're <inaudible>?

**Feigenbaum:** Or to the printer, yeah. And memorizing all the material, Simon and I calculated over a very large number of experiments done in psychology, it's roughly speaking 30 seconds per nonsense syllable that the subject has to spend on, it in repeated trials through the list. If you don't give them 30 seconds, you're going to have to go through another trial and another trial and another trial and another trial.

**Nilsson:** Until it adds up to that.

**Feigenbaum:** Until it adds up to about 30 seconds per nonsense syllable. There's a big difference between recognition and recall, as psychologists knew by that time, knew very well. Of course, in EPAM the difference is that for recognition you only have to have as many branches as it takes to discriminate the small number of syllables you have. But for recall, you have to have it all, in order to output it. That was, as far as I can tell, that's the first instance in computer science of adaptive growing trees. Now, what I didn't do was to explore <u>that</u> as a subject of scientific discovery. Trees were not interesting to me except as an organization of human memory to model some psychological experiments. To put it another way, Simon and I were sitting there, and doing EPAM, we were behaving like psychologists. We weren't behaving like computer scientists. Now, on the other side, there was Newell working on, what you might say, both. He was working on human problem solving, but he was also working… He really sort of had an AI idea in mind, leading to more general things for AI. He also had a list processing idea in mind. But for me it was psychology and I was doing my thesis, so I didn't explore trees. Of course, trees became a very important subject in computer science.

**Nilsson:** Especially these decision trees, as they later were called, in machine learning, right?

**Feigenbaum:** Yes. It would have been good to have some of the early publications in trees. Ed Fredkin at MI-- at BBN or something, not MIT, but I think BBN, also <phone ringing> also discovered.... So Fredkin at, I think he was at BBN at the time, packaged up this. He also reinvented this idea and put it in another context, and he published a paper on it called "Trie memory." But there were several other computer science oriented approaches to growing trees that got published in the next several years. But none of those were by me, except insofar as it influenced psychological modeling and early AI. AI people picked up on it because I was publishing in the AI literature. So that was good.

**Nilsson:** So here we are with EPAM, and you're getting toward the end of your graduate career, and EPAM became the main work for your dissertation; is that right?

**Feigenbaum:** Well, that was my dissertation.

**Nilsson:** That was your dissertation.

**Feigenbaum:** Yeah. It started out the moment I arrived at the graduate school in September of '56 and it just flowed right through until the point it was time to write it up.

**Nilsson:** And you wrote it up.

**Feigenbaum:** I wrote it up and took my oral exams. It was a thesis type exam, on the thesis, in September of 1959, in the morning of the day that I was supposed to get to New York by the afternoon and get on the S.S. United States for my Fulbright scholarship.

**Nilsson:** I see. So after your graduate work at Carnegie Tech, and well, during that time, you applied for a Fulbright and ended up getting a Fulbright or a Fulbright was awarded in some way?

**Feigenbaum:** Yeah. In 1958, I got married, and my wife and I decided that it would be interesting to spend a year in Europe. So during that time I applied for a Fulbright scholarship, which was the only quick way I knew of getting to Europe was to get a fellowship of some sort to do postdoctoral work. I didn't know much about where I should go in Europe to do this. But in 1958, the National Physical Laboratory had held what amounts to the first big AI conference, the Mechanization of Thought Processes conference in the London area. I had the name of the guy who organized that conference.

**Nilsson:** The National Physical Laboratory was a unit of the British scientific establishment?

**Feigenbaum:** It played the same role in Britain as our National Bureau of Standards did in Washington, now called NIST, National Institute of Science-- what is it? National...

**Nilsson:** Institute of Science and Technology.

**Feigenbaum:** Is that what it is? Anyway, it was like the National Bureau of Standards at the time. So it was a government institution. Turing had gone there after the war to follow up on some of his insights into how to build computers and what to do with them. Turing, of course, had died in the early '50s, so by the time I got there in '59, there were still people working on that line of work, and there was still a machine there called the Pilot Ace.

**Nilsson:** By the way, before going on and talking a little bit about what you did on your Fulbright at the National Physical Laboratory, the subject, the name Turing came up. Did you read Turing's 1950 paper, and did it have any influence on what you decided to do?

**Feigenbaum:** Yes.

**Nilsson:** This is the paper which Turing wrote on computing machinery and intelligence.

**Feigenbaum:** Sure. I sure did. So did we all. Its influence was nothing like the a-ha experience of having the actual thing that Simon and Newell were doing. But it was kind of inspirational that someone was talking about the big picture. And it was very interesting that he was talking about the big picture in the sense of experimental-- an empirical approach to intelligent behavior. Which is, “intelligence is as intelligence does”, not some big theoretical view of intelligence. So, that was shaping.

**Nilsson:** Back to the Fulbright year. What happened during that year? What sorts of work were you doing?

**Feigenbaum:** In the Fulbright year, I had no reason to expect, honestly speaking, from if I had done a good empirical study, that anything exciting would be going on at the National Physical Laboratory. And indeed, nothing really was. But it turned out that there were other places in Britain where there were very good people pursuing very good ideas, and one of them was… In Cambridge I met two such people. They also happened to be husband and wife. Roger Needham and Karen Spark Jones. Karen was working with Margaret Masterman on computational linguistics. That was a very hot group at the time. There weren't too many of those in the world, and Margaret was a really good leader of that group. Karen was very young, but she was also very good. Roger was a young person working in what was then called, I believe, the computation center. I'm not sure exactly what it was called, but Roger eventually became "the professor" in that area. But he was interested in the chain of intellectual events and the detail that led to what the Americans were doing in compilers, particularly in the list processing area. He wanted to know more about that. Although we hadn't done a compiler; we had done an interpreter. He wanted to know exactly how that would work. He was really coming into it from a computer science point of view, rather than an AI point of view, which is what Karen was doing. So I met the two of them, and we traded lots of ideas. They came down to NPL sometimes; I would go up to Cambridge sometimes. It was actually a lot more fun to go to Cambridge, because there was a big group of people working up there. That became more or less the best thing that happened on my Fulbright.

**Nilsson:** Now, wasn't Seymour Papert at NPL around those days?

**Feigenbaum:** That's right. There's a sort of a long and funny story involved with me meeting Papert, but I won't waste the time here to tell that funny story. Papert showed up exactly at the same time that I was showing up, from South Africa via Europe to the National Physical Lab. Papert was a leading mathematician in South Africa and I think I once… Papert's wife, who he was not living with at the time, but whom I met several times, it was said that she was South Africa's second best mathematician, and Papert was South Africa's fifth best mathematician. So he was really a mathematician. But he had been interested in complex models of behavior. He went to Europe to work on that with, I believe, with Piaget and he spent some time. When I knew him, he had just come out of experimental work with a psychologist named Kerler who worked on perception in the situation where the perceiver has been wearing glasses, eyeglasses, to change the world, like upside down or right/left confusion. Indeed it led to some hairy episodes with driving with Papert in London when he would drive into a roundabout, and he would have to stop and think about whether he was going to go left or right, and what indeed was left.

**Nilsson:** Then while you were in England, you were probably thinking a little bit about, okay, what do I do next, and what kind of career do I have?

**Feigenbaum:** Yeah, I already had a job before I left Carnegie. In typical fashion, as we know today even, if you're the supervisor of a graduate student, one of your responsibilities is to help place the graduate student in a proper job, hopefully academic job somewhere. Since I was working with Herb Simon on basically psychology, I was a card-carrying psychologist, member of the American Psychological Association. It seemed reasonable to go to a place that had a really good psychology department and was doing sort of cutting edge working in modeling, psychological modeling. And that was Stanford University. Dick Atkinson was here, Pat Suppes was here. Gordon Bower was at Stanford. So Herb approached Pat Suppes to try to get an in for me to get a job, and basically the answer was no.

**Nilsson:** They weren't hiring that year?

**Feigenbaum:** Or maybe they didn't want to hire in computer simulation, or maybe Pat didn't like the work, or whatever. I've mentioned this numerous times to Pat and he chuckles all the time, doesn't remember any of this. But where it was easy to get a job was in a school of business administration, because that's where I had a union card, and a union card signed by one of the best in the area, namely Herb Simon. So that turned out to be… I had wanted to come to the West Coast to particularly the San Francisco area. Berkeley was excited about getting me, and about getting Julian Feldman. Both of us left Carnegie at roughly the same time. Julian got to Berkeley a year before I did because I went on a Fulbright. I got there in September of 1960, then the two of us started to teach essentially two things. One was organization theory a la March and Simon, which we both knew from our graduate work, and this new discipline called artificial intelligence or in some variations of it computer simulation of cognitive processes. We were teaching that to Berkeley students in '61 and '62. That's when we were having a lot of trouble getting… I mean, what would we do? There were no books on the subject. But there were some excellent papers, and we would send the kids over to the library, and we'd have photocopies of the papers. I don't even know if Xerox machines existed at the time. But we had photocopies of the papers.

**Nilsson:** Thermofax.

**Feigenbaum:** Thermofax or something. We just decided that we needed to do a collection. It wasn't just our students who were interested in this. There was a whole nation, or maybe an international need, for such a thing. So we took the papers that we had collected up, plus a few more that we asked people to write, and put together a collection called Computers and Thought, published in 1963. Interesting story about that is, tried to get it published. Herb Simon happened to be the business publications editor for Prentice-Hall. So of course we went to Herb and said, "Herb, here's this book." He advised Prentice-Hall not to publish it because it wouldn't sell enough copies. He later told me that was one of the big mistakes of his career. It did sell a lot of copies. And it really helped to… As I say, there weren't very many books in AI at the time, so it really helped to launch the field.

**Nilsson:** It had papers both in cognitive science and in what later became more AI.

**Feigenbaum:** That's what we decided to do. I don't think we sort of made that up. I think that was a kind of a thing that was going on in the culture at the time, namely, there were a group of people who were behaving like psychologists and thinking of their work as computer models of cognitive processes using simulation as a technique. And there were other people who were interested in the problem of making smart machines, whether or not the processes were like what people were doing. The kind of sentence that went around and lots of us used it, and I notice it's still being used routinely is, we don't make airplanes by modeling the feathered wings of birds that flip up and down. For engineering purposes we have different kinds of wings.

**Nilsson:** But right about that time, maybe in the late '50s, the phrase cognitive science first appeared, didn't it? People were actually thinking of "the discipline" called cognitive science.

**Feigenbaum:** Boy. Let me give you my impression of that. I could be very wrong, and I'm not a history buff in that sense, like you are. It's my impression that the phrase being used at that time was cognitive psychology, not cognitive science. And that the people that were practicing the simulation part were using the term “information processing models”. In fact, I think Newell and Simon's working paper series had that kind of a phrasing to it. They weren't called “computer simulation of cognition working papers” or “AI working papers”. They were called something like “information processing working papers”. It's my impression that the term cognitive science arose when, in the late '70s, the AI community was trying to get organized into a society, as opposed to the confusion -- that everyone-doing-their-own-thing confusion.

**Nilsson:** I think you're right about that.

**Feigenbaum:** I was strongly arguing that the term “artificial intelligence” really wasn't the right term to use, because it just hit the wrong nerve in the public at large. It gave some hint of hubris and anyway, who wants something that's artificial? I had been proposing that we AI people change our name to cognitive science, which makes a lot of sense. That's what we do. Well, that really got people upset in that other community, that cognitive psychology community, examples being Don Norman as being one of the people who got upset. But there was a whole bunch -- I think Roger Schenck may have been another one -- but they quickly assembled themselves into a thing called the Cognitive Science Society, and having sort of sewn up that name, that was something we couldn't use for what society we were trying to form.

**Nilsson:** They had a founding meeting, I think, didn't they, in San Diego or maybe UCLA in 1979 or '80, something like that?

**Feigenbaum:** Yeah. It was '79, in August of '79, was when the group that had been working on the AI society, which ended up being the American Association for Artificial Intelligence, AAAI, met in August of '79 at the IJCAI, Tokyo, meeting to plan our next step -- what we were actually going to do. Then Newell was asked to be the first president of the new society. I was asked to be the second president. Raj Reddy was really organizing the whole thing. By that time the cognitive science people had actually incorporated as the Cognitive Science Society, and indeed, they did have their first meeting about the same time the AAAI was having its first meeting.

**Nilsson:** Now, back to Computers and Thought for a moment. You and Julian did a very generous thing, I think, with the royalties there. Do you want to say something about that?

**Feigenbaum:** Oh, that worked out wonderfully. We were in a situation where we had many, many papers, maybe 20, from different people, and we didn't-- how would we distribute the royalties? We didn't know that the book was going to sell many copies anyway. Herb Simon certainly thought it wasn't going to. What, we were going to distribute the royalties in dribs and drabs to individuals? So we just wrote to everyone and said, "How about if we just set up a prize with these royalties? We'll call it "The Computers And Thought Award." It was a prize for young people, not like we see these days where the older, more mature people who have had a long career get big prizes toward the end of their career. This would be for, like, the best thesis of the year in AI, or if not the thesis, then the best contribution of a very young person in their maybe first years of their academic career. So that was the Computers and Thought Award. I don't know exactly when IJCAI came into existence, but we put it in the hands of IJCAI -- International Joint Conference on Artificial Intelligence, which was just popping up around that time -- to make the award and give out the word at their biannual conference.

**Nilsson:** Back to research at Berkeley. We've talked a little bit about your teaching a course, and putting together this collection of papers in Computers and Thought. How was your research life going during those days at Berkeley?

**Feigenbaum:** It was, I would say, very frustrating for actually several reasons. But the ones that jump to mind are, first of all, it does matter what department you're in in a university. It's not a casual matter. If you are a professor in the school of business administration, you're going to get the PhD students who have come by that graduate school. If you're not in the place where what we would now think of as the right kind of person for computer science, or even psychology, if you're not at that place, they're not going to flow by you. I was not in the psychology department. I was a member of the thing called "The Center for Human Learning" which had its own frustrations, which I will tell you about in a minute, but I didn't get that flow of graduate students. I didn't get the flow of graduate students that were coming by Harry Huskey in EE, and I didn't get the ones that were coming by the mathematics department oriented computer science students. Berkeley was a big mess at the time in terms of where computer science was located. They eventually ended up with two departments, which had to be then later welded into one department. But I didn't get any of those graduate students. That was a problem. I had not a very good set of graduate students. Secondly, I was really not very happy doing models in psychology. It was great to do a thesis in that area, because I had both the interest and the support of Herb Simon as I was doing it, and I learned a great deal. But at the very heart of things, I was really interested in artificial intelligence, not in exactly how to formulate models that would replicate the results of particular kinds of psychological experiments. In other words, I wasn't too interested in my union card in psychology. I really liked AI.

**Nilsson:** You didn't want to be constrained by having to have feathers on the wings.

**Feigenbaum:** Yes. And in the culture that I came from, the feathers on the wings were taken very seriously. If you look at Newell and Simon's book, *Human Problem Solving*, you will see hundreds of pages of direct matching of what GPS was able to do with exactly what human protocols were doing in the same experiment. Newell and Simon took that stuff really seriously. Similarly with the EPAM work, we took that really seriously. I was not interested in doing the psychological research myself to validate certain aspects of EPAM. The Center for Human Learning at Berkeley was not interested in doing it. They were very experimental people, but they were not interested in anybody's particular theory. They were interested in 82 variations of the same experiment by turning the knobs in all these different directions. That was very discouraging.

**Nilsson:** You did finish EPAM. Didn't you do another version of EPAM?

**Feigenbaum:** Yeah, I did. I did another version of EPAM long distance with Herb Simon, EPAM III. We had a bunch of things to pursue at the end of my thesis. Every thesis is that way. It leaves open questions. You want to finish them off. We did that. We published a paper in, I think, 1965 about EPAM III. I think I wrote a couple of surveyish kind of articles which summarized all of that research and then essentially I got out of that field, although Herb stayed with it. Herb was publishing EPAM papers right up until the time he died, back in the early years of this century. I don't remember exactly -- 2002 or something like that. He was publishing papers on EPAM right up until that time, and he went through various versions of EPAM with a student of his, Howard Richmond. I think they may have gone up through EPAM VII or VIII or IX in various variations of the model. I did occasionally publish a paper with Herb along the way, maybe one per decade, on some aspect of EPAM that Herb was interested in and invited me to be a coauthor on, but essentially I gave that up.

**Nilsson:** Did you have research support when you were at Berkeley?

**Feigenbaum:** Yeah, I did.

**Nilsson:** You had some research contracts?

**Feigenbaum:** Initially, I got a foundation grant. I'm blocking on the name of what foundation it was. It was a relatively small grant, but it was enough to keep Julian and me going, and it came with no government strings attached. Herb Simon had recommended, to a foundation that was beginning to make grants in this area, that we were good people to give money to. Then we had a wonderful event, which is that JCR Licklider came by. He had taken over… He had started a new office for ARPA, not DARPA at the time, ARPA, Advanced Research Projects Agency, in information processing techniques. He had picked up the word that there were some good people at Berkeley in AI, but also some good computing people at Berkeley, Harry Huskey in particular, a pioneer in computer architecture. So he came out to discuss ARPA contracts. We weren't, of course, the only ones. He was doing a big one at MIT with project MAC, and a big one at Carnegie Tech or maybe Carnegie Mellon by that time, called a Center of Excellence grant for the work that Newell and Simon and Perlis were doing. At Stanford, he was interested in funding John McCarthy. I believe he was funding John for AI work, not timesharing work, but I'm not positive of that. At Berkeley, he wanted Julian and myself to do AI work, and he wanted Harry Huskey and an EE team to build a medium-sized timesharing system, which we eventually did. I was co-PI of a contract that had two branches. It had one branch doing EPAM and other related AI things that I might be interested in, not too definite, because you didn't have to be very definite with ARPA at that time. And another branch that was supposed to do something halfway between the PDP-1 type timesharing system that McCarthy had pioneered, and the project MAC timesharing system, which was huge, that they were doing at MIT. That was actually a very, very interesting experience, not on the AI side but on the computer side, because Harry Huskey took a sabbatical in India, and the guy who substituted for him as PI was a superb computer engineer: Dave Evans, the designer of some of the Bendix early computers and later cofounder of Evans and Sutherland. I really learned a heck of a lot about how computers are designed from Dave as we were selecting a machine to modify that later on would become the SDS 940 timesharing system. But anyway, that was the grant support that I had.

**Nilsson:** That's great. Maybe we'll go on to the next tape.

END OF TAPE 2

**Nilsson:** Ed, we had talked about you and Harry Huskey having the first ARPA grant at Berkeley, and under that grant you had done some work finishing EPAM 3. What sort of intellectual ideas were you interested in at that time?

**Feigenbaum:** Actually, for me, it was a period of kind of an intellectual turmoil, or a searching, or a lack of direction, because I hadn’t quite settled yet on where I wanted to go. I had pretty much decided that I wasn’t going to go down the psychology route, and as I mentioned, I was more interested in building intelligent machines than in building models that had a great verisimilitude with the human data that the psychologists had. But when I look back on it, I have a feeling that it was not that bland. I had a feeling that there was a much deeper component to it, and I only get this feeling looking back in time, that for reasons that are not totally clear to me, I really, really wanted smart machines. Or I should put the really in another place. I really wanted really smart machines.

**Nilsson:** Super smart machines?

**Feigenbaum:** Yeah. I remember very clearly a paper by I.J. Good on the ultra-intelligent computer. In fact I.J. Good published two versions of it, one in ’56 and one maybe in the early ‘60s. But, yes, that’s what I wanted, the ultra intelligent computer. I wasn’t going to get there by either walking down the road that I was doing with EPAMs, which is models of verbal learning, or doing what I then considered wonderfully inventive puzzle solving tasks, like Newell’s. He was doing trigonometric identities, cryptarithmetic, those kind of things. “Donald plus Gerald equals Robert.” You know, come on, that really is not it. That’s not where we’re going. For him that was where we were going, because he was interested in detailed models of human thinking. But I wanted really big stuff. Big fish. You had a name for it recently in one of your papers, “going after the dream” or something like that. What was it called?

**Nilsson:** Eye on the prize.

**Feigenbaum:** Yeah, "eye on the prize". So, yeah, the big idea. For Newell and Simon the big idea was to do the best psychological models that had ever been done. And in fact, they both did win the equivalent of the Turing Award in the American Psychological Association, as well as winning the Turing Award. So, they succeeded, at least in recognition. But I wanted something bigger. When I thought of something bigger, I couldn't think of anything bigger, and I really still can't, than induction. That is, induction is what we are doing almost every moment of almost all the time.

**Nilsson:** Both informally and as scientists.

**Feigenbaum:** Yeah, that's right, and that time I was thinking informally. That is, we're taking in a lot of data and we're building a model of the world, and we're constantly testing that model of the world and revising it. So I wrote a couple paragraphs on this that were the end of the introduction to Computers and Thought. Not a section introduction, but the main introduction, was about induction and why I thought that was the way we were going, that's what was going to carry forward into the future. Maybe that's a good strategic plan, but it wasn't a tactical plan. I didn't have any plan for how to do it. I thought hard about it, and during those years you were talking about, I was kind of in the middle of thinking about it. I had thought and picked up several, what Newell and Simon would call task environments, and had rejected them for one reason or another as not being the right ones.

**Nilsson:** What's a task environment again?

**Feigenbaum:** It's a sandbox in which to specifically work out your ideas. It's a domain in which you can experiment. You have actual problems. You have actual knowledge. You have actual procedures that can work in that domain. For example, for Newell and Simon it started out being propositional calculus. The next one was chess. For GPS they chose lots of puzzles. Galernter had the geometry theorem prover. Jim Slagle had calculus. Everybody had some task domain in which they were working. I think it's very important to emphasize, to this generation and every other generation of AI researchers, how important experimental AI is. AI is not much of theoretical discipline. It needs to work in specific task environments. I used to tell McCarthy and others -- I think I may have even written it down -- that I'm much better at discovering than inventing. If you're in an experimental environment, you put yourself in the situation where you can discover things about AI. You don't have to create them de novo out of your head. I have a feeling that my colleague John McCarthy is more of an inventor than he is a discoverer. He doesn't play around with his ideas a lot in experimental task domains. But I'm from the experimental AI school. So I was looking for a task environment, or a complex sandbox, to discover induction methods, induction ideas, induction problem solving. It had occurred to me that scientists were a good place to look. The reason is that if you look back at, let's say, the early chess machines, Simon spent a heck of a lot of time examining books on chess -- openings, end games, tactics for the middle game. Alex Bernstein, who worked on one of the early chess programs, was himself a chess master. It seemed like that we in AI were respecting expertise in those areas. So, who are the experts in induction? Well, scientists get paid to do that, or at least according to the stereotypic model, they get paid to do that. They have a lot of data, laboratory data. They form hypothesis, they make predictions, they revise their hypothesis.

**Nilsson:** So, the forming hypothesis part was what you were interested in?

**Feigenbaum:** That's right, and actually, theory formation -- I don't want to go into the difference right now between what I was regarding as hypothesis formation and theory formation -- but theory formation was the next step along the inductive process. I happened to be saying this, expressing it during conversations at Saturday morning sessions being run by a Stanford Professor in psychology, called [Karl] Pribram at the Center for Advanced Study in the Behavioral Sciences. It was on, basically, minds and machines, what we would now call minds and machines. I don't know what Karl called it at the time.

**Nilsson:** How did you happen to find out about that and go down there?

**Feigenbaum:** I have no idea. I was at Berkeley. It was happening at Stanford. Maybe I was just in touch with Karl. In fact, I think it was, because Karl was running-- that's right, Karl was running a series of annual workshops sponsored by the New York Academy of Sciences, being held at Princeton, called "Remembering, Learning and Forgetting." And Karl Pribram was in intrigued with EPAM. So he had invited me to a series of these things. I was in touch with Karl Pribram, so it must have been directly through Karl that I found out about this. Drove down the old Bayshore Freeway, the two lanes on each side thing. One of the other people who showed up there because he just was interested in the topic, but I have no idea how he knew about it, was Joshua Lederberg. Maybe McCarthy told him about it. Josh showed up because he was being reintroduced to computers. In the early days of his scientific career, he had been involved with computers from a card-punch calculator point of view. Now he was learning the Burroughs machine, learning ALGOL from Forsythe. He wanted to test his knowledge out on some complex problems, so he was writing an algorithm that elucidated chemical structures, and he was working on a mass spectrometer project, he was working on an entire instrumentation platform that would fly to Mars and look for life. I was mentioning this to Josh and he got interested in the subject. By that time I think I had decided, or at least I was in the process of deciding, to go to Stanford. I may have already decided. Josh said to get in touch with him and we'll work on it, so I did. First thing I did when I got to Stanford, other than find my way to the bathroom, was to go talk to Josh about this. Josh had a specific suggestion. When I told him exactly what I wanted, about the induction process, it wasn't any old scientific reasoning I was interested in, he said, "Yeah, I have just the one for you, and we're working on it in our lab." His lab was doing the mass spectrometry of amino acids, because their planet probe was going to look for amino acids as part of the method for discovering whether life-like forms existed on Mars. The question was, how do you go from looking at a spectrum of an amino acid to the actual structure of the amino acid, the chemical structure of the amino acid? That's how we started in with the DENDRAL Project. I was good at heuristic search. He had an algorithm, which was good at space generation. Namely something like, you know, I suppose there's a guy who's good at mastering chess because he knows thousands of good things to do on a chessboard, but you still have to have the move-generating algorithm.

**Nilsson:** So he had the legal move generator?

**Feigenbaum:** He had the legal move generator.

**Nilsson:** And you thought you could come up with the heuristics.

**Feigenbaum:** Yeah, so the legal move generator was called DENDRAL, and the AI program was Heuristic DENDRAL.

**Nilsson:** When you left Berkeley to go to Stanford, how did that go? Did Berkeley try to keep you, Stanford try to hire you? How did that work out?

**Feigenbaum:** Good question. Okay, so I've already told you that being a professor, untenured or tenured, at a business school wasn't really quite the happiest environment for a person who had the kind of goals I had, and interests. But in any case, some tenure is better than none, and Berkeley's a really nice place, and we had a beautiful house in the hills. Berkeley started, believe it or not, quite early in my career, they started in on the tenure process. I say quite early because I got there in 1960 but they started in on this tenure process from in the '63 through '64 year, because they had given me an acceleration at the front end of my assistant professorship in order to get the salary level up right. In the University of California system, the two are locked together, or at least at that time they were. So I was being evaluated for tenure a little bit early, and the business school really couldn't quite make up its mind as to whether it wanted this weird guy who basically talked about artificial intelligence. What's that? It's not accounting. It's not marketing. It's not finance. That kind of opened up my thoughts. It's not easy to open up your thoughts to searching around when you bought this beautiful house in the hills, and you're settled in and all that. But I began to search around, and I must have made it known to McCarthy that I was doing that, because John took the idea to Forsythe. In fact, I think, with a very positive recommendation said, "Let's hire him," and Forsythe said, "Great, let's hire him."

**Nilsson:** Those were the days when you could do it that easily.

**Feigenbaum:** Yeah, and a similar thing went on with Klaus Wirth. Harry Huskey was Klaus's main advisor, but I was an advisor on Klaus's thesis as well, so I knew Klaus Wirth. George hired Klaus to come down as an assistant professor in '64. I was off on a sabbatical or something in the last semester. The September through December period in '64, I traveled to Russia as a guest of the Academy of Sciences and then got back to Berkeley during all the turmoil with the Free Speech Movement, and Mario Savio, and all that. Anyway, left on December 31st. Came to Stanford on January 1st.

**Nilsson:** Now, you left someone at Berkeley--

**Feigenbaum:** So, I had this money and we were going to build an AI group. I was hiring the best AIers I could get. One of the best was one of Minsky's students, Bert Raphael. Terrific guy. By the time Bert actually showed up in Berkeley, in fact he showed up not only as a research staffer in AI at Berkeley, but also rented my house while I was off in Russia, he and Anne. But I was leaving, so that was not a great thing. I mean, Bert made a career move and then it turned out, there wasn't any AI at Berkeley except for me and Julian, and Julian was going to Irvine, and I was going to Stanford.

**Nilsson:** So there was Bert, high and dry.

**Feigenbaum:** Bert, "come down to SRI".

**Nilsson:** Yeah, good for us.

**Feigenbaum:** Good for SRI.

**Nilsson:** By the way, when he wound up teaching, he was teaching us how to do list programming and insisted that we recursion. No PROGs were allowed.

**Feigenbaum:** [laughs] Where did you get that?

**Nilsson:** When he was at SRI teaching us about LISP, it had to be recursion. One of the guys who was trying to learn LISP at the time thought, “Aha, this is like programming under water. You need scuba gear.” Anyway, now you’re at Stanford.

**Feigenbaum:** I’m at Stanford. Josh and I kind of framed up this project, what we were going to do, and I did a working paper with Dick Watson, who was another student of mine at Berkeley. I had been on his thesis committee. He was another system engineering guy, but he wanted to work in AI, he decided. Came down to Stanford. Left soon thereafter.

**Nilsson:** Went to SRI.

**Feigenbaum:** Went to SRI, yeah, and then Livermore, I think. Dick and I wrote a paper called Opportunities for Graduate Research, and it laid out… this was the first paper ever published in the DENDRAL project, but it was a kind of overall glossy view of the DENDRAL project. As a result… We didn’t get any doctoral students who were interested in doing that. That’s a good story in itself, as to why doctoral students in AI or computer science shy away from things like that. But we did get two Master’s students, one of whom you know very well because she worked at SRI later, Georgia Sutherland, an. absolutely fabulous knowledge engineer. And Bill White, also terrific. So we got these two wonderful Master’s students to work on the project, and in a way that was way better than getting PhD students, because Georgia and Bill were totally dedicated to the idea of getting these programs running, as opposed to PhD students who kind of wander all over the map.

**Nilsson:** Wanted to do their own thing?

**Feigenbaum:** Yeah.

**Nilsson:** What computer were you using?

**Feigenbaum:** We started out using the PDP, whatever PDP John [McCarthy] had in the AI lab. PDP-6 at the time, maybe. Maybe it morphed into a PDP-10 at some point. We started using that, and using it under timesharing. We ran into some problems there. I’ll tell you what they were, but we moved back onto campus using the IBM 360.

**Nilsson:** You were out at the AI lab?

**Feigenbaum:** Well, Bruce was, Bruce Buchannan. That was like our office. On campus, we had our sort of "thinking" office with Lederberg and myself, but the actual experimental work was going on on terminals in Stanford AI lab.

**Nilsson:** Out on Arastradero Road?

**Feigenbaum:** Arastradero Road, yeah. The problems there were partly that the machine -- it was just a wonderful timesharing system, one of the best in the world at the time -- and therefore it was tremendously overloaded. Everyone wanted to use it, number one. Number two, one of the people who wanted to use it the most was John Chowning, inventing these FM synthesis ideas for computer music.

**Nilsson:** Which made a lot of money for Stanford.

**Feigenbaum:** It made a huge amount of money for Stanford, but you can't believe how much computer resources that soaked up. John was on it all the time, heavily loading the machine. The other thing was Jerry Feldman was doing the SAIL language, and that was heavily loading up the machine. Plus, there was robotic experiments and other things going on.

**Nilsson:** And then they take the machine down every now and then?

**Feigenbaum:** That's normal, yeah. Then the other problem was that we were working with chemists doing what later was called knowledge engineering, or knowledge mining, out of the heads of these professionals, these experts. They would have a long drive. Like, Alan Duffield we were working with, he'd have to drive from the chemistry department down here on campus all the way out to Arastradero Road. That's about a 20 minute drive, 20 minutes back. Things would have to be very scheduled in order to do that. So we decided to move down to campus, and that means that we needed extra money in our budget to buy computer time at the 360 computation center. But DARPA was very nice about that, and they said, "Okay, sure, if that's the way you have to do it." So, we had a semi-interactive experience. It's sort of a remote job submission by terminal.

**Nilsson:** Was that Licklider at that time? Probably not, huh?

**Feigenbaum:** No, Ivan Sutherland had taken over, and then Bob Taylor. Lick was already gone. But things were still wonderful at DARPA. I mean, Ivan and Bob -- it couldn't have been better. Nobody asked us, in any respect at all, not even detail, whether the stuff we were doing in mass spectrometry had anything to do with the military. They didn't even ask in any detail what it had to do with AI. If you thought it was a good idea, we'll do it.

**Nilsson:** We're sponsoring good ideas, that's what we do here.

**Feigenbaum:** Yeah, that's what we do here. And at Carnegie MellonI it was called "Center of Excellence". They didn't even have to tell them what they were doing with the money. They were just being excellent. So we moved back onto campus, to answer your question. Then as things matured and we were really, by a little bit later on… Oh, Lederberg and I were participating -- he was PI and I was director of the project -- building a computer center for the medical school, which was called ACME, A Computer for Medical Research. ACME was an excellent timesharing system for its time. It was designed by Gio. Wiederhold. ACME had its day in the sun. By the time ACME was ready to essentially fold up as a project, either the medical school would support it or it wouldn't, but NIH had paid for its development and they were going to move on. We moved on to say, okay, what we need is now a computer center for biomedical AI, which included the DENDRAL project. It also included the work that we were starting to do with the doctors on MYCIN and other medically related investigations we were doing. We also wanted to enliven a national community of interest who didn't have timesharing, but who did have access, through Licklider's good offices, to the ARPANET through TIPs [Terminal Interface Processors] at their location. We applied to NIH for money to get a computing machine to do all this. Well, what was a computing machine? It was basically the same as McCarthy had, except eventually we just wanted to use the Tenex operating system. I think we used TOPS-10 or TOPS-20 at the beginning. McCarthy was using some other operating system, but basically the same machine.

**Nilsson:** Now, there was a time when you were using the Q-32?

**Feigenbaum:** Oh, yeah. That was Licklider's doing.

**Nilsson:** That was while you were doing the DENDRAL work?

**Feigenbaum:** That was right at the beginning, as soon as I got to Stanford. Licklider was still in office. His big projects were Project MAC on the east coast, and converting an air defense computer called the AN/FSQ-32 at System Development Corporation…

**Nilsson:** That had come from Lincoln Labs initially? Maybe not.

**Feigenbaum:** Hmm. It was one of the prototype machines for an upgrade of SAGE System, I believe, and SDC was programming the software for the SAGE system. So this computer was now surplus or something. He wanted it converted into a timesharing system and got SDC -- well, paid SDC -- to do it. But there had to be a user community. So kind of he strong-armed a bunch of us to be users of that Q-32. Well, it turned out in our case, it was okay. I strong-armed a psychology graduate student that was working with Gordon Bauer and myself, John Anderson, to use that as the means of writing his thesis, HAM, Human Associative Memory, which is another kind of memory model similar to EPAM, but more extensive. He did in on the Q-32 in a little converted closet up in Serra House, that old building that we--

**Nilsson:** So your work on the Q-32, then, was before you used the PDP-6 at the AI lab, or overlapped?

**Feigenbaum:** Well, they overlapped, but DENDRAL was not runing on the Q-32. If I remember right, it was only Anderson's work. I think. I could be wrong. Georgia Sutherland would know.

**Nilsson:** Before we get on to the business about the big computer for the medical center and that community, with the DENDRAL work, sometime during this time, Bruce Buchanan came. How did that come about?

**Feigenbaum:** Bruce was a student in the philosophy department at Michigan State, particularly doing philosophy of science work as a PhD candidate. He was working on, to use words that came out of the title of his thesis, the logics of scientific discovery. Those words were the kind of words -- I didn't think of them -- but they were the kind of words that could have described exactly what was on my mind in '64 when I was thinking about this, talking to Lederberg at the Pribram meetings and so on. Geez, isn't that a wonderful way to express it? Bruce had read writings of Newell, Shaw, and Simon, and had realized that these had emerged as Rand Corporation publications, and Bruce needed a job in the summer. So he wrote to Paul Armer, the head of the department [at RAND]. It wasn't called computer science, it was called numerical analysis department. He wrote to Paul, or wrote to RAND, and said, "Do you have a summer job for me?" Paul sent it on to me and said, "Do you want this guy for the summer?" When I looked at what he was doing, I said, "Sure, that looks like it's great. Couldn't be better." So, hired Bruce in the summer of '64. We exchanged a lot of ideas about scientific discovery, and how you would model it.

Then Bruce needed a job when he finished his thesis in the spring of '65. Bruce wanted a letter of recommendation to get an assistant professorship in philosophy somewhere. I said, "Bruce, you don't want to do that. We have this wonderful project that's just exactly described by your thesis, logic for scientific discovery." I told him about it, and we arranged for that. We were actually in a bridge period, bridging off of some funding that we got from NASA, which happened to be located in Berkeley. So Bruce showed up at Berkeley for a while to kind of satisfy that bridge obligation. Then in 1966, one year later, he made the transition to Stanford and started work on the campus.

**Nilsson:** Got right involved with the DENDRAL project.

**Feigenbaum:** Well, he was working on the DENDRAL project all along. It's just that he--

**Nilsson:** Now, he was doing it at Stanford. Why don't you tell me a little bit about DENDRAL, what it was really trying to do, some of the techniques you used, and having it do what it did.

**Feigenbaum:** The first thing I got to say, I will tell you in AI terms, try to keep it short, use AI terms as a shorthand, which may be not too good for people way out in the future there who don't know these AI terms. But what can I do except point you to references, lots of things, like a book on the DENDRAL project published by McGraw-Hill. Robert Lindsay, Buchanan, myself, Lederberg, I think are the main authors. Published in the '70s. I want to start out by saying that we did not have a grandiose vision. This was bottom up. This was really bottom up. We started with the DENDRAL algorithm, which was the legal move generator, and we started generating some chemical molecules that contained carbon atoms, hydrogen atoms.

**Nilsson:** These chemical molecules that you generated, these were in the form of what, sort of graphs?

**Feigenbaum:** Chemical graphs.

**Nilsson:** Chemical graphs with atoms in the nodes--

**Feigenbaum:** Topology but not geometry. We just didn't deal with 3D geometry, and never did. It was connectivity.

**Nilsson:** Like the graphs, you might see in a chemistry course?

**Feigenbaum:** Yeah. Carbon has a valence of four, so it's a four connected thing, and oxygen is a two connected thing, and hydrogen is a one connected thing. We started out with the DENDRAL algorithm, and one of the interesting things that we discovered -- almost on day two of the project, not literally, but just after we got Lederberg's algorithm running in LISP and ran off a few tests -- was that even for the second simplest amino acid, Threonine, there were structural isomers of Threonine that we generated simply. There were only 32 of them, I think, the number was, and therefore it's not a big deal. But a whole bunch of them were not present in the Beilstein Handbook of Known Organic Structures. The reason is, no one ever thought of them. No one ever thought of…

**Nilsson:** They were structurally possible.

**Feigenbaum:** They were makeable by any graduate student in any laboratory. No one had done it. It was not in the Beilstein Handbook. So that was an interesting thought: that even without intelligence, we were able to "discover" some things that weren't in the literature. But then the spaces… I said Threonine, second simplest amino acid, was 32 things, but as soon as you start adding some more carbon atoms with branching factor of four, some more nitrogens with three, and so on, you start adding atoms -- you start getting up into huge spaces. Well, AI people are comfortable with huge spaces.

**Nilsson:** Huge, meaning what, tens of thousands?

**Feigenbaum:** No, not tens of thousands. Ten to the something-or-other. Huge, exponential, big spaces. But that doesn't scare AI people. AI people come out of a background where legal moves in chess are ten to the hundred and twentieth for a game, roughly speaking, plus or minus a few orders of magnitude, maybe ten orders of magnitude. We know what to do with that. What we do is prune the space and we steer our search through the space. So that's what we began to look for: how can you prune, how can you steer? Well, the pruning and steering, to make a long story short, comes from knowing a lot about chemistry. And you can either know about chemistry in general, like, for example if you string out two oxygens together you're going to make a peroxide, and peroxides are just inherently unstable. So, you'll blow up the instrument if you do that. If you start juggling these atoms in the structures, if you get an oxygen connected to an oxygen, don't bother.

**Nilsson:** Throw it out. The machine would have blown up, and it didn't blow up, so throw it out.

**Feigenbaum:** Yeah, exactly. So what you do is you get rid of not just that one, you get rid of a whole subtree, just like you do in chess. And then the other has to do with knowledge that's specific to the empirical discipline of mass spectrometry. Some things break up in certain ways and not in other ways, and you got data, and the data can be used to--

**Nilsson:** And chemists know how they break up.

**Feigenbaum:** Our chemists did. Our chemists were -- that's another interesting story -- our chemists were Carl Djerassi, a famous man, inventor of the chemical behind the birth control pill. Later one of the world's most respected mass spectrometrists, which is the phase he was in when we knew him. He was also doing commercial adventures along the side. I think he was the founding CEO of Syntex or close to it, maybe the CEO when Syntex moved to Palo Alto, to Stanford Industrial Park. And then he started another company. Anyway, a renaissance guy. He was a leading mass spectrometrist of the time and the author, or the editor, of a big handbook of empirical mass spectrometry. What I mean by that, it's not theoretical. Not looking at the quantum theory and deciding what will break inside an atom, but looking at what did break when you bombarded this thing. So Carl and his team knew a huge amount about mass spectrometry. By deliberate strategy, we began to show them -- when I say them, I mean Carl particularly -- the early "results", by which I mean really poor results, by intention. We didn't want to make this very good. We wanted to show Carl something really poor. And Carl would say, "That is really stupid." Our answer, already planned is, "Carl, what is it you know about mass spectrometry that we don't?" That was it. So we got Carl to work. Carl got hooked on the project, and he laid out a series of what you might call knowledge adventures. Maybe he opened up the minds, maybe that's a better metaphor, for ketones, and ethers, and sulfur-bearing compounds, and a whole bunch of them that he laid out for us. We systematically worked our way through that, while elaborating the underlying AI method, which is simply a generate and test method. It was generate and test with a prediction -- we had a thing called a predictor, which would predict for those molecules that passed the test, what mass spectra would result from those molecules, and what did we actually have.

**Nilsson:** Oh, you had a model of what the mass spectrogram would do.

**Feigenbaum:** That's right, we had a virtual mass spectrometer.

**Nilsson:** Now, were these during the days in which DENDRAL was only handling the acyclic?

**Feigenbaum:** Yes, so all of this was acyclic molecules. The original DENDRAL algorithm by Lederberg was a tree-generating algorithm. Cycles were really hard, because cycles brought in all kinds of mathematics that we didn't--

**Nilsson:** Cycles, like, for example, a benzene ring would be a cycle?

**Feigenbaum:** Yeah, that's right, closed rings. Computer scientists are, or were at the time, pretty good at tree structures, pretty bad at cyclic graphs, although there were a few people that were great at it, like Bob Tarjan, who was on the faculty here. Now, the discipline is way ahead of where it was then. We actually hired a mathematician on sabbatical at the time, but he liked the project so much he just stayed, that was Harold Brown, tenured professor at Ohio State. Came and worked with us, along with some graduate students, Larry Masinter and a couple other graduate students, and Georgia Sutherland doing the programming. We worked out the cyclic generation strategy. Georgia actually did one that was… Well, let me go back. The Lederberg acyclic algorithm generated the space completely but irredundantly, and that's magnificent. That says that every candidate that could possibly be in that space was in that space, and it never occurred twice. Georgia did one for cyclic molecules, which didn't have that nice property. We could accidentally ignore some, and we could accidentally duplicate the generation of some. The duplication wasn't as bad as the ignoring. But Harold, and Larry, and the others, worked out the details of the other algorithm. Then we were back to the property, the coverage property that we wanted. [We] kept adding in the chemical knowledge. We did a series of experiments. For me, the critical series of experiments started around 1966. I reflected on it in the, roughly speaking, the spring of 1968, trying to ask myself, what did we really do? This was because Donald Michie was having a conference in Edinburgh: Machine Intelligence Workshop something-or-other.

**Nilsson:** One of his series?

**Feigenbaum:** One of his series. He had heard about DENDRAL from me or somebody.

**Nilsson:** He had been to Stanford visiting.

**Feigenbaum:** Maybe that's why. He wanted a paper on it. We hadn't really published in the AI literature up until that time, although we had given a talk to AI people at Carnegie Mellon. Josh Lederberg and I flew back to Carnegie Mellon at Herb Simon's invitation to give a talk in '67. I thought, gee, we better make this a good paper. So, what had we really done?

**Nilsson:** We'll get to that next.

**Feigenbaum:** Yeah, we'll get to that.

END OF TAPE 3

**Feigenbaum:** The question was: what is it that we had really done? Well, I put together... essentially laid out the experimental evidence starting sometime around '66 and ending around the time I mentioned, early '68, going from the first of the chemical molecules where we began to add in Djerassi's knowledge, all the way to the most complicated ones that we were doing at the time. I also wanted to make sure that the AI community read the details of this. Not just a high level gloss, but I wanted to give them the actual structures and the actual detailed data. So the 1968 paper in Donald's volume is, historically, for me, a very important paper, but I'm not sure how much the AI community was able to get through it because it was just full of what looked like a lot of chemistry. These experiments amounted to titrating in more and more and more and more and more and more and more knowledge. The more you did that, the smarter the program became. That is, in terms of smart, I mean the more relevant became the answers, the right answer showing up. We didn't often -- in fact, probably never -- gave the chemist only one answer. We gave the chemist a few answers. But the better the answers got over time from titrating in a lot of knowledge.

The message of that paper was: “in the knowledge lies the power.” I knew I was using Bacon's words, but I didn't really mean the same orientation as Bacon. Bacon was talking about political power, but I was just using the phrase as a catch phrase. “In the knowledge lies the power.” Which means we had pretty much of a vanilla flavored generate-and-test process, not that much different from what had come out of the first decade of AI's work. That didn't bother me at all. I'm a person who believes in building on the scientific results of the past, rather than having to reinvent everything new in your generation. So I was pretty happy with that. Gee, those old ideas really work, and we just had to tune them a little bit, like with the predictor, to get it to work a little better. But I was very happy with that. The difference was, we were learning how to represent the knowledge of chemistry. And we were learning how to represent it in a very nice way, a very high level symbolic way, where you could actually see exactly what the knowledge was. It was: if the molecule was in this shape, that implies meaning the mass spectrometer will do this, and it will break it up into these pieces. Or, if the molecule contains O-bond-O, then throw away those candidates because they're unstable.

**Nilsson:** So you noticed a kind of a monotonic increase in the quality of the results as the experiments went along and you added more knowledge. More chemical knowledge.

**Feigenbaum:** Yes.

**Nilsson:** Not more search...

**Feigenbaum:** No, not more search. That's right. We didn't have any fancy search things at all, except the pruning and steering. Steering meaning, if I have some left, what do I look at first? What are the priorities for looking at things?

**Nilsson:** So this "in the knowledge is the power" then informed subsequently AI research that you and your group did?

**Feigenbaum:** Yeah. That was the big idea. In my career, that is the huge, "Ah ha”. “Oh, my God, that's what it is." That wasn't the way AI was being done previously. If you looked at GPS, for example -- which was the hot project of, I don't know, the early '60s or something, for AI people -- it was a brilliant piece of work. But, when it got to the knowledge part, Newell and Simon had allocated it to a relatively small operator difference table which had maybe in it, I don't know, depending on the domain, two dozen facts, or something like that, in the table.

**Nilsson:** But in the “Donald plus Gerald equals Robert” problem, the cryptarithmetic problem, they didn't go out and interview experts on solving cryptarithmetic to find out any particular heuristics that they used.

**Feigenbaum:** No, they didn't. But it is true that Herb Simon read enormously in chess while Newell and Shaw were programming up the chess model. I sometimes think of Herb Simon sitting there at his desk, with all of his books, as the first knowledge engineer. He was mining it out of these books, which is fun. Alex Bernstein was mining it out of his own head for the chess program.

**Nilsson:** To get a little more information on DENDRAL: the input would then be, typically, a mass spectrogram and perhaps the chemical formula?

**Feigenbaum:** Yes. No. The input-- yes. Uh huh.

**Nilsson:** And then the output would be some structure, or maybe a group of structures. And usually the structure that an expert chemist would come up with for that mass spectrogram and that formula would be among that small list...

**Feigenbaum:** That's correct.

**Nilsson:** ...which could have been a list of millions, but you were able to pare it down with all of this knowledge to a list of half a dozen or whatever?

**Feigenbaum:** That's correct. Incidentally, I was calling that “hypothesis formation”. That is, single spectrum. Which is, my dataset today, to my answer today. My answer forms a hypothesis to explain that data. The theory was the generator of the knowledge. ut we didn't have a theory program at the time. We did later, in Meta-DENTRAL.

**Nilsson:** Now, what were the size in, say, molecular weight, of some of the more complex compounds that the DENDRAL in those days could handle?

**Feigenbaum:** Oh, boy.

**Nilsson:** I mean, they were big compounds.

**Feigenbaum:** Yeah.

**Nilsson:** They weren't proteins, they were...

**Feigenbaum:** No, no, no, no, no.

**Nilsson:** Not that big.

**Feigenbaum:** But let's say… In my memory sticks the fact that C-23 was the largest one that we ever tried to handle. Remember, we were working on 1/Nth of a PDP-10 or PDP-20, which is a trivial amount of computing compared with what even our desktop delivers today.

**Nilsson:** Yeah.

**Feigenbaum:** So we were hesitant to get into very large combinatorial spaces. I think C-23 was the largest one we tried, but I'm not sure about that. It may have been a good deal smaller than that.

**Nilsson:** Let's move on to some of the follow-on to the DENDRAL years were, especially now armed with this important idea that knowledge is extremely important. You also coined the phrase, "knowledge engineering". Can we talk a little bit about what was done under the general banner of knowledge engineering? I guess, Meta-DENDRAL came along. You wanted to learn some of these rules.

**Feigenbaum:** Let me separate meta-DENDRAL from what I would call mainstream knowledge engineering. Meta-DENDRAL was the culmination of that dream of the early to mid-'60s that had to do with theory formation. The conception was that you had a problem solver -- call it DENDRAL -- that took some inputs and produced an output. In doing so, it used a layer -- which I didn't give a name to but Newell later called the knowledge level -- of knowledge that it was using to steer the search and prune the search and so on. That knowledge got in there because we interviewed people. How did the people originally get the knowledge? How did Djerassi get that knowledge? By looking at thousands of spectra. So we wanted a program that would look at thousands of mass spectra and infer the knowledge of mass spectrometry that DENDRAL could use to solve individual hypothesis formation problems. And we did it. But only for a narrow class of compounds. A class of compounds called ketoandrostanes. What we used was a generate-and-test model for that, where, since we had several years of experience with representing the knowledge of DENDRAL, namely the knowledge necessary to solve that problem, we knew what its form was. We could then generate objects of that form exhaustively and irredundantly, and we could prune the space by using now thousands of spectra to prune that search space. So that's what we did. What fell out of that was a handful of rules for ketoandrostanes. I can't remember exactly how many, but we were able to publish that new knowledge of mass spectrometry in the Journal of the American Chemical Society, without giving much credit to -- only in a footnote -- that a program, meta DENDRAL actually did this.

**Nilsson:** It should have been the author.

**Feigenbaum:** The triumph there was, at the symbolic level, we were able to do something which had been a dream: to kind of come up with a new piece of science that you could actually publish. It wasn't special relativity, it certainly wasn't general relativity, it was just a little thing that we were able to publish. But, still, it was some science. Then the disappointing part of it was… Well, the good part was, after that Tom Mitchell cleaned it up. Meta-DENDRAL was oriented toward a specific goal that had to do with mass spectrometry. Tom Mitchell, with his version spaces, cleaned it up into a more general thing that other people could use, in concepts other people could use. I thought that was the beginning of something. Turned out that was the end of something. Nobody followed up on that. The rest of the stuff turned out to be statistical machine learning theory. What does that have to do with…? Well, anyway, to me, that was a tremendous disappointment.

**Nilsson:** When you're thinking about meta DENDRAL and the problem that it was involved with, namely trying to come up with scientific ideas, the knowledge that scientists have -- well, that itself is a problem. Could you interview scientists to ask them what heuristics, what knowledge, what meta knowledge they had about coming up with… How did Djerassi decide to… Are there any heuristics that would have helped Djerassi learn all of those rules?

**Feigenbaum:** There are two answers to that. The first is, absolutely yes. Absolutely. That leads me to say something quite general which I'll say in a minute. But specifically for meta DENDRAL, we actually did talk to them about how do you… if these were the alternatives, how do you prove that space of possibilities? We did get some good ideas from them. But the general point that I'd like to make, from what you just said, was that I am not at all a believer in this thing called tacit knowledge, which has a history in philosophy and it comes up, oh, I don't know, ten times a year, you'll see it in various writings of people who think they're writing about AI. I just don't believe it. That it's tacit while you're not interviewing the person, it's tacit while… Yeah, sure, the person uses it and that person doesn't even know they're using that knowledge. But you could interview that person in the context of specific situations and dig it out.

**Nilsson:** Dig it out.

**Feigenbaum:** And there's no reason you can't dig it out. That's what knowledge engineering is all about.

**Nilsson:** You can declarativize it in some way or another.

**Feigenbaum:** Yeah. What I called knowledge engineering, Donald Michie at one point called knowledge mining, which has a better feel to it. You're actually chipping away in the mine [mind?].

**Nilsson:** During these days of the DENDRAL project and up through meta DENDRAL, during some of the early part of that period, you have this other job of being computer center director. Do you want to say anything at all about that, how you happened to get involved in that job and what effect it might have had on your research?

**Feigenbaum:** Yes. In was the summer of 1965 through, roughly speaking, the end of 1968 or maybe early '69 -- '68 probably -- [when] Paul Armer, who had been head of the computer department at Rand, took over my job. I recruited him to take over and make it a professional, not a professorial activity. George Forsythe had been director of the computer center. When he started the computer science department, he noticed that he couldn't do both, especially since the computer center was about to transition into a new generation of machines, and expand. He really didn't have the time to do that. So he asked around about people who were interested in that role. For reasons that I just don't understand today, I said I was interested in that role. It may have been -- let's see, that was 1965 so I was 29 years old -- it may have been that it was time for me to try out managing something. Or maybe it was time to try out being a big shot, or who knows what? I'm not sure exactly what it was. But I got into a big management role, managing a budget of two to two and half million dollars a year, and big equipment that had to run all the time and it would be a financial disaster to Stanford if it didn't, and we had to sell a lot of time. Oh, plus the fact that IBM was in a very unstable state during the early days of the software of the 360. We were promised a timesharing system. I knew, after a year, that we weren't going to get that timesharing system, because I had a thesis student named Norm Neilson who did a thesis simulating, under my direction -- he was an operations research student -- he simulated the activity of the proposed 360/67 timesharing system. We could show in simulation, that it wasn't going to work. We tried to convince IBM to change the system. They wouldn't believe a simulation, and they didn't. It was a disaster, and we had to switch to a 65 with another kind of software that I think Stanford is still using, Wylbur. Anyway, it was a tough time. And we did not only one, but we did three. We did the 360/91 at SLAC, where Bill Miller was running the SLAC computer operation. He delegated that job of running the big computer to our center. Then we did the medical school one, where we hired Gio Wiederhold to do the timesharing system. That was a big one. Did it detract from my research? The answer is absolutely yes. Looking back, I wouldn't do it again. Research time is so precious and the rewards are so huge that it's silly to fritter it away on managing something.

**Nilsson:** Well now we are what I would think of as ending some of the early days at Stanford, and transitioning into some of your middle days at Stanford, in which you were involved in various projects, HASP [?] and the various expert systems. Do you want to lead off by talking a little bit about some of them?

**Feigenbaum:** Well, I want to get back to answering your question about the rest of knowledge engineering. I said, let's put meta DENDRAL aside, and then we started another set of experiments. Partly opportunistically, that is, partly based on what problem came up and who wanted to do what. We did a rather systematic search for… Back up. One swallow doth not a summer make, and one DENDRAL doth not a solid piece of science make. You have to try it out on other things. We had good connections into the medical school, with Lederberg as being a co-PI on all of our work. We started searching in different areas of biomedicine for relevant problems. For example, one came up with the whole question of drug interactions. Well, it turned out that drug interactions were an interesting problem, but it could be solved in another way with big table lookups and there wasn't any advantage of solving it. No virtue and a lot of difficulties.

**Nilsson:** Not a sandbox for working on induction.

**Feigenbaum:** Yup. But it turned out that the person we contacted, as we did this search for that particular problem, was a professor of pharmacology named Stanley Cohen. Stanley Cohen eventually became world famous as the co-inventor of the first methods of genetic engineering with Boyer. In the middle, from the time we knew him in drug interactions to the time he became this world famous guy, he was interested in AI. He particularly worked with one of our students, Ted Shortliffe, M.D./Ph.D. student, on AI methods for doing diagnosis and therapy of bacterial infections, where Stan was the person whose knowledge was being mined. Plus, not just Stan. It was Stan plus other physicians in that area, in the infectious diseases area of the medical school, and that led to MYCIN. So it was a combination of Stan's interest and Ted's interest in doing that. Ted knew the language of doctors, as a medical student.

**Nilsson:** He was a medical student at the time?

**Feigenbaum:** A medical student at the time. He also knew computer science ideas, and he was interested in AI, so that was an opportunity. Then Larry Roberts was directing the Information Processing

Techniques Office at DARPA, one of the successors of… Well, after Bob Taylor, Larry Roberts took over. And Larry was facing a… Larry knew about the DENDRAL success, and Larry abstracted it in the form of: you have a spectrum, you got an answer.

**Nilsson:** They got a problem like that, huh?

**Feigenbaum:** Forget the details. Yeah, I have a problem like that. I have large numbers of spectra coming from hydrophones buried deep down out on the Continental Shelf. They are sending signals back to the shore. Those signals are being processed by Fast Fourier Transform and minicomputers into frequency distributions over time. I have beams of them, and there are lots of beams, and from that I have to decide what kind of enemy submarine is out there in the ocean. In an ocean that is just roiling with sound. So that when you try to listen to it with your ears, you just don't hear anything. It's just one mess of sound, and a Soviet submarine trying to be as quiet as it can. And that's like DENDRAL? I mean, DENDRAL was clean data, couldn't be better, nice black and white lines of molecular data, and a nice generating algorithm for generating candidates. And here's Larry saying: “same problem, isn't it?” Anyway, he wanted me to work on this problem, and we did. We said, okay, let's give a try. It was a classified problem so it was done off campus at Systems Control Incorporated. Incidentally, for those of you who are watching this program 50 years from now, the reason the classified work went off campus was student riots that took place during the Vietnam War, which, believe it or not, for those of you 50 years from now, was a war that took place in the late 1960s in which millions of young people in the United States were very upset with this war. They were rioting and destroying campuses. Stanford, yes, did get destroyed, but also didn't want the rest of it to get destroyed. So they unloaded classified research by (a) deciding it shouldn't happen on campus and (b) selling SRI, Stanford Research Institute, to itself, to become SRI International, and get all the classified work that was on campus sent to SRI. So our work was done off campus. It was the problem of tremendous amounts of hydrophone data. There is a single answer. Not a single answer, but a small number of answers. It's a Soviet submarine of type XYZ traveling at so many kilometers per hour passing by Cape Mendocino right now. And here's the reason I believe that: long explanation that came from the inferential argument. And the knowledge came from people who were experienced in doing that. They knew the big intelligence books that had all the information in there about what these submarines had on them, and what noises they made. They had experience on what neighboring stations told each other along the coast. They had experience with what the U.S. navy did along the coast, so that you could subtract out our own noises from the ocean. And what kind of animal noises there were in the ocean, and all the other stuff that you would want to account for.

**Nilsson:** That's probably the results from an hour ago or ten minutes ago, which would have some effect, probably, on what the results now would be.

**Feigenbaum:** I guess so, yeah. Right. Because sound propagation is what you're talking about?

**Nilsson:** And tracks, you know, a particular submarine that might have been identified off the Mendocino coast ten minutes ago was not going to be off San Diego...

**Feigenbaum:** Right. So there was a lot of that kind of knowledge in the program, too. That is, the submarine is a physical object. What can it do and what can't it do? The ocean itself is a physical object.

If you're tracking this submarine and you have a hypothesis as to what you're tracking and where it's going and somehow the sound shuts off, well, don't worry about it. Don't junk your hypothesis right away, because the submarine may have just gone behind a sea mount and oh, by the way, we know, from the past, Soviet submarines love to hide behind sea mounts. They don't just pass the sea mount, they stop behind the sea mount, just to get you confused. So just wait awhile, and then decrement the credibility of your hypothesis over time. After an hour, you can say, well, I don't believe that any more; I'm going to form a new hypothesis. But otherwise predict the track just where the submarine was going.

**Nilsson:** This certainly was a hypothesis-formation problem, an induction problem, of the sort you were interested in anyway.

**Feigenbaum:** Yeah. To my great surprise, I couldn't believe how successful it was. I just didn't believe… That was the hardest problem I had ever faced. I couldn't believe that we could do this, but we did it.

**Nilsson:** But you used a particularly interesting AI technology in that...

**Feigenbaum:** So the technology-- so, yeah. I tried the generate-and-test strategy. You know, you give a guy a hammer and he hammers one nail with it, the next nail, he's going to use the same hammer. Well, the same hammer -- generate-and–test -- didn't work. Because what are you going to generate? What are the candidate solutions to this thing? Submarines of any kind anywhere?

**Nilsson:** There's no legal move generator.

**Feigenbaum:** Yeah. There's no legal move generator. We had to have a piecemeal opportunistic solution, grabbing onto any data or any inference that you could grab onto, and building up an inferential or symbolic complex of… Man, how to say this. What you believe in at what layer of abstraction. Like, what do you believe about the fleet? What do you believe about this particular submarine? What do you believe about the machinery in this submarine? What do you believe about the lines that you're seeing in the spectrum? Well, fortunately, right at that time, something just like that had popped up at Carnegie Mellon in the speech understanding project that was being run with DARPA money. It was called Hearsay. It was a product of Lee Erman and Raj Ready and Victor Lesser and Rick Hayes-Roth, which designated such layers and a problem-solving method for working between the layers. Now, the layers for them were things like the pragmatics layer, and the semantics layer, and the lexical layer -- that is, the word layer -- and the phonetics layer, eventually terminating down here in the signal itself, the segments of the speech sound wave. The inference method was top-down and bottom-up all happening intermixed. If you knew something about the lexical layer, that is, the words, you could then infer something about what phonemes would be in those words, particularly if you have a dictionary. It gives you pretty much the pronunciation and you can infer exactly what the phoneme should be from that. And from the phonemes, you could predict lower levels of what the sound should look like, and you could work top down that way. But you could also work bottom up. If you saw some patterns in this signal, you can say, ah, that's this kind of a phoneme -- phoneme wasn't the right word for that level -- but it popped up into the phoneme level, and then from there into the lexical level. You could make hypotheses going bottom up. When you mix the two, you get a very powerful mix of expectation-driven reasoning by looking top down, and inductive reasoning looking bottom up, [which] makes a very powerful problem-solving method. Since you're dealing with a serial computer, you can organize all of this activity into an agenda, saying what's the priority? What should you look at first? What next, and what next?

**Nilsson:** And when you had a parallel computer, some of these things could be done in parallel.

**Feigenbaum:** And we tried that in the '80s with our parallel blackboard in our project. Anyway, we modified the Carnegie Mellon idea. Carnegie Mellon actually didn't use their idea in the speech understanding product that they produced. They used another idea called HARPY, which was another kind of search idea: beam search idea using dynamic programming as a selection mechanism. But we weren't constrained by the real time problem that they had with speech understanding. We had forever to make these calculations, because the submarine was moseying along from Russia to the U.S. It was thinking it was going fast but going through the ocean is a pretty slow process, so we had forever to make these calculations. So we could use the whole Hearsay type mechanism, and it worked really well. Like you just said, that, when it came time to-- when DARPA was thinking about parallel computing, we thought of parallelizing Blackboards and we did export the methods. We abstracted some methods from the classified work that we did, packaged them up into a piece of software called AGE, and that was the Blackboard framework of choice for several years until basically what took over was a couple modifications of it done by Barbara Hayes-Roth called BB-1 and BBK in our laboratory. AGE was done by Penny Nii, who was the chief engineer of the sonar signal processing project over at Systems Control.

**Nilsson:** And Blackboard -- not only was it successful with the sonar processing, but it was used in many other applications, right?

**Feigenbaum:** Again, like I said before, one swallow doesn't make a summer. You've got to test this thing out in various ways. We wanted the toughest possible test, outside of the sonar context. So we found a problem. I had known some people in X-ray crystallography at UC San Diego by virtue of my having served on an NIH committee with a person, an x-ray crystallographer, and I knew about his inferential task. I knew what he was trying to do in inferring the three dimensional -- notice geometry not topology -- the three dimensional structure of proteins in crystals from the x-ray crystallographic data -- the electron density that you got from doing x-ray crystallography on the protein crystals. Many of you know about this because you've read Watson's book, "The Double Helix" -- I think it was by Watson, not Watson and Crick -- Watson's book on "The Double Helix", in which one of the British scientists has done that kind of crystallography on what eventually turned out to be the DNA. Watson basically stole the pictures from Rosalind Franklin, who should have gotten the Nobel prize along with Watson and Crick but she died early. She had done the x-ray crystallography in her lab of the DNA that gave Watson and Crick the idea of how this geometry worked. We were trying to do that with our program, CrysAlis. The people who we were working with, I believe, were the first people to have done the structure of hemoglobin. There weren't too many of these structures that had been inferred at the time, because it was really a tremendous amount of work to do this, and a lot of work crystallizing the protein. Anyway, we used a Blackboard model to do in two ways. One was to analyze the electron density itself. That was very close to the data. The other was to infer… You got partial results by analyzing the physical data, and part of the results by working up and down the chemistry. So we had intersecting Blackboards, in which one Blackboard had the levels of chemical abstraction, the other Blackboard had levels of crystallographic abstraction, and, at some points they would intersect. Where the crystallography was making some inferences [and] the chemistry was also making the same inferences, we could believe in those nodes because both were telling the same story. That was intersecting Blackboards. That was the first and only time that's been used.  Then Blackboard became a little industry.  There was one of the UMass Amherst students working with Vic Lesser started a Blackboard company, selling Blackboard software during the era of expert system construction.

**Nilsson:** It's used quite a bit throughout computer science.

**Feigenbaum:** Yeah.

**Nilsson:** As a matter of fact, as you and I both know, it has resurfaced -- some of the ideas of top down and bottom up simultaneous working -- has resurfaced a bit in some of the recent work on brain modeling.

**Feigenbaum:** Jeff Hawkins published a book on… Basically, the book starts out by saying AI hasn't done much, and all of the next chapters are bringing in AI results and showing how they apply to the brain.  I'd appreciate it if Hawkins would actually reference the AI stuff a little more.

**Nilsson:** So we had several of these things going on during this period, Ed.  There was the meta DENDRAL work, the work on sonar, the work on bacterial infections with Ed Shortliffe.  Also you were beginning to think a little bit about molecular genetics and that led to some interesting programs also, right?

**Feigenbaum:** The project that that led to was called MOLGEN.  For years, I had been asking my colleague Josh Lederburg, how come you have been recommending to us that we work in the chemistry or medical areas when, in fact, you are a geneticist -- in fact, a Nobel prize winning geneticist?  How come we're not working in your area where you, yourself, can be the expert?  In fact, I humorously codified a law, called Lederberg's law, which is: everything is automatable up to but not including what I do.  But, eventually Josh said, yes, let's do it.  And that had to do with the invention of the genetic engineering method of Boyer and Cohen.  Josh's view was, that starts a new era.  And if it starts a new era, we can get in on the bottom.  That is, if we have to do knowledge accumulation, knowledge engineering, don't start where you have a huge pile of knowledge, like in genetics.  Start where you have almost no knowledge, which is genetic engineering.  That got us into MOLGEN.  There were two theses done in MOLGEN.  First of all, MOLGEN was the first computer science project anywhere in what is now called computational molecular biology.  There were some other computer efforts in genetic departments at places like the University of Wisconsin, where geneticists were doing also what we did, which is initially searching of genetic sequences for interesting information about the occurrence of basis in loops and isomorphisms and things like that, homomorphisms.  That was a set of tools that was extremely valuable, and on our network timesharing system [we] soon had hundreds of users in the biological community.  Those tools were called SEQ for sequence analysis, and it led to the formation of a company called Intelligenetics, which offered those services for sale to the biological community.  In addition, there were two significant Ph.D. theses involved in the planning of experiments in genetic engineering.  One of those theses was what you might call “plan by recipe”, very much similar to what Roger Shank was telling us ought to be the way that we looked at AI problem-solving and planning.  The other was a much more basic view of building up a plan from the bottom up, in very similar ways to the way SRI had been doing it and other people in planning.  The theses of Mark Steffic[ph?], which was the one closer to the traditional methods, and the one by Peter Friedland, which was more skeletal planning, planning by recipe or story.

**Nilsson:** Okay.

END OF SESSION 1

## Session Two: June 27, 2007

**Nilsson:** Ed, I'd like to have you talk a little bit more about molecular genetics -- computational molecular genetics, especially the MOLGEN Project -- but as I recall I think you were Department Chair during that time. Do you want to say a little bit first about being the Department Chair of Computer Science at Stanford, and then we can talk a little bit about MOLGEN.

**Feigenbaum:** The Department Chair role that I played was essentially a service role. It lasted for five years. It was a three year term renewable once. I was asked to renew and therefore I should have been Department Chairman for six years. In fact, the job was so vexing and so distracting to me that I actually quit at the end of five years. Now, I can give you the reason, the nominal reason that I gave for quitting, but even in the middle of the five years I took two one-quarter sabbaticals, one to Tokyo and one to Paris, in order to relieve the pressure. Ullman was Acting Chair at one time, and McCarthy was Acting Chair during the other period. I've been looking back at why I said “yes” to being Department Chair. It turns out, I think, to be a lot more complicated than I once thought. I think partly it had to do with the problems the department was facing while Bob Floyd was Department Chair. Floyd was a friend of mine, absolutely wonderful guy, a genius at computer science. He just was not really a great Computer Science Department Chair. So we had gotten into a bit of a mess, and it needed some cleaning up. I'm not sure why I thought, or the university thought, I was pretty good at cleaning up things. Maybe because I had some managerial experience with running the computer center. That was one of the reasons I took it on. I think the other reason was that that job had a luster associated with it, which had to do with George Forsythe, who started the department. He was simply superb, like maybe the best Chairman you could ever imagine. So it had a luster, and maybe a little bit of that luster would rub off on me. Maybe that's what I thought. Well anyway, we were in… So we had a problem. Bob Floyd had planned the new space that we moved into, from being scattered around to being moved into a portion of a building in the Quad. And I say that with all bitterness, because we should have had the whole building. Instead, the provost, who happened to be one of our colleagues, Bill Miller, at the time made an elaborate compromise, which I think partly had to do with the doner, to give part of the building to some Psychology Department laboratories. So we were basically lacking a floor and a basement of a building, and we were far too crowded going into the building. That was a big problem for the Chairman. How do you shoehorn the department, which is being brought in from numerous locations, including the Stanford AI Lab on Arastradero Road, how do you shoehorn them into space which is totally inadequate, and pretend that you're bringing the department together? Well, the people that you're dealing with aren't ordinary people. They are some of the world's greatest scientists, people like Don Knuth and John McCarthy. And Knuth needs space for his brand new computer-driven printers, for example. Or we're trying to build a systems group. Systems groups take a lot of space. The HPP [Heuristic Programming Project] was growing by leaps and bounds, and yet I couldn't use my role as Department Chairman to exhibit favoritism to my own laboratory. So eventually we actually had to move off campus, simply because there was no way to grow. So that was a big problem. Accomplishing anything as Department Chairman, as you know Nils, from your role as Department Chairman, is extraordinarily difficult in the Computer Science Department. The Chairperson is the handmaiden for a bunch of highly ego-driven, often charismatic people who really run the department, namely the professors. I set out a goal for myself. I may have told this story before, but let me tell it again. I got my PhD in three years when I was a PhD student of Herb Simon’s at Carnegie Mellon. I started by thesis research basically on day one, and at the end of three years I had enough to qualify as a thesis. I passed my exams in the middle of it all. Allen Newell did something very similar as a graduate student. He graduated a couple of years earlier than I did at Carnegie Tech, but he did something very similar. He did the logic theory work with Simon, and then he passed his exams and he was out. In our department our average student was taking five and a half years to graduate, and the Stanford norm, if you look in the administrative guide, is four years for a PhD, not five and a half. I was determined to organize the process so that the students had to pass certain hurdles at certain times, and they would get out of there in four, not three, but four. And I worked, and worked, and worked at it. I must say, when I say I, I don't mean just me. It was me and Dennis Brown, a young person who had been a student of mine and a graduate student for a very long time. He himself was one of these people. He became Assistant Chairman of the Department, organizing the educational program, and organizing this process, among many other things Dennis did. When I finished in 1981, when I finished prematurely in 1981, the average time for a graduate student to get out of our PhD program was five and a half years. Zero progress in five years. So it's a rather discouraging kind of job.

There were some good things associated with it. I was thinking of one actually this morning. I was listening to an ad on the radio, and the ad was from a car dealer, and the car dealer was Fletcher Jones Junior. Fletcher Jones Senior was one of the founders of Computer Science Corporation CSC, and when he died he gave money to the Computer Science Department for a Chair. I had the honor to recommend and then appoint Don Knuth to that first Computer Science Chair. So there were some happy things associated with it. I found that, in retrospect, there was only a limited amount of attention one has in the world, and the older one gets the less amount there is of that, because other things become more important in life. In the limited amount of attention that I had, looking back, I should have been paying more attention to some of the excellent things that were going on at the HPP, including this MOLGEN Project that we were developing, rather than spending my time trying to reduce the average graduate student stay from 5.5 to 5.5.

**Nilsson:** Yet the MOLGEN Project's pretty successful, so do you want to say a little bit about that? You obviously had some roles in that, and some excellent graduate students.

**Feigenbaum:** To follow on from the previous question you asked me last time about MOLGEN, we were right near the end of the session and I didn't have time to expand on a couple of things, and I would like to expand on that. First of all I'd just like to repeat that the idea of moving into genetics itself was a longstanding idea, where we kept asking my colleague Joshua Lederberg, when is it the right time to start working in his field so that he could be our expert? At one point he said, "Now is the right time." It was a short time after Cohen and Boyer had invented the basic techniques for genetic engineering, and it looked like we can get in on the ground floor of that discipline. Remember that our mantra, then as now, is that "In the knowledge lies the power." So if we didn't have the knowledge of genetics behind us, we would have no power in this effort. And the knowledge of genetics was vast by roughly 1973, 1974 -- all through that period in fact from the '60s through into the '70s. The knowledge of genetics went back a hundred years, at least. So Lederberg was quite right in pointing out we should get in on the ground floor of something new, namely genetic engineering, not all of genetics.

**Nilsson:** That's just when they were figuring out, as you mentioned, Cohen and Boyer, splicing.

**Feigenbaum:** That's correct.

**Nilsson:** The techniques for taking DNA apart and reassembling.

**Feigenbaum:** Yeah, exactly. The basic techniques of genetic engineering was one of the two most valuable patents that Stanford ever had. The other being John Chowning's FM synthesis that he did at the Stanford AI lab, which was something that just about drove me out of the AI lab because his FM Synthesis experiments were soaking up so much computer time on the DEC machines out there. Anyway, two things in the MOLGEN Project that I want to be specific about. The first was, I want to talk about a strategy that we had -- we evolved -- from the 1960s up to that point, that point being roughly 1976, for finding and fascinating an expert. Now I use that term because a Berkeley friend of mine, who at one point made a lot of money in the computing business and decided to go off and just write a book, wrote a really best selling book called, "How to Find and Fascinate a Mistress." But for our work, finding and fascinating an expert collaborator in some other discipline was truly important. They are busy people. They have little incentive to collaborate with you, unless what you're doing for them is giving them some productive edge, some leg-up on their science. It's implausible to think that the AI part of our project would deliver the cutting edge right away, because we have a lot of work to do. We have research to do before we get there. So we have to provide some kinds of tools that will find and fascinate an expert short of the hypothesis formation, or problem solving, or planning, or whatever that AI people know how to deliver eventually. We did that, of course; in Dendral we did it with structure elucidation, that is, we did it with the Dendral algorithm and its symbol manipulation and its notation, long before we delivered any of the Heuristic Dendral to the chemist. And we did it with the doctors, when we did some of the early medical applications like MYCIN, PUFF. Well, in this case we did it with one of the arrows in our quiver that at that time was pretty specific to AI people, namely, the ability to do computational symbolic manipulation easily. We not only had a lot of experience with it, going back all the way to the IPLs in the '50s, but we had some superb tools to do it. At that time we had Interlisp running on the DEC machines. Probably there was no programming environment ever designed and developed better than the Interlisp environment at that time. So we were really powerful in that area. And what was the molecular geneticists' problem? The problem was sequencing was coming along. It was primitive at the time compared to what it is now, but sequences with thousands of bases were coming out. Now we're talking about millions of bases, billions of bases. At that time it was thousands beginning to pileup. But even thousands are too much for the human eyeball to scan and look for what amounts to mathematical regularities, or sequence regularities. So we offered, as a tool, sequence manipulation. If you read in all these bases from the sequences, what are what we now would think of as the primitive things you could find, can calculate from that? Loops and homologies, and other kinds of patterning things in the sequence. In the '80s such methods took a huge leap forward and contributed perhaps more than any other thing to the actual sequencing of the human genome, which was more or less considered impossible until this brashness of immense amounts of computation were thrown at the problem. At that time it was still pretty primitive. But it wasn't all that primitive. It wasn't all that unhelpful. Since we were connected to this also very primitive thing called the ARPANET, via the goodness of DARPA that let our machine -- the only non-DOD sponsored machine on the ARPANET -- people could dial into so called

TIPs around the country, which allowed them, by phone, to access machines on the ARPANET. We were one of those machines. The machine being called SUMEX-AIM. A-I-M stands for “AI in Medicine”. They could come in and use our sequence manipulation routines, which were called SEQ, was the name of the package. Before long we had something like 300 users coming in over the ARPANET to the SUMEX machine, which at one point was a PDP-10. It later graduated into be a PDP-20, but it never was a major resource. But we had 300 people coming in from biology departments around the country, from pharmaceutical companies that wanted to use this new tool. They didn't come in trying to use the AI things we were doing. Nobody said, "I really have got to have Mark Stefik's planner to plan some new experiment in molecular genetics." No, that was way too easy at that point for humans to do themselves. It was not a task that boggled the human mind, whereas sequence analysis was right at the edge of a task that boggled the human mind, even for those very small sequences.

I want to say two things more about that. One is that we had a couple -- either one or two and I can't remember which -- competitors in this area, academic competitors. One definitely was at the University of Wisconsin in, I think, the Genetics Department, or Molecular Biology, or something. It was a group that was doing sequence analysis, not using Lisp, but using something much simpler like C, I think the routines were written in. Then there may have been a second one, and it may have been at Yale. I'm not sure about that. The details of that are written out in a history of this project, MOLGEN, that was done by Professor Tim Lenoir at Stanford in collaboration with two participants in the MOLGEN Project, Professor Brutlag in Molecular Biology, and Dr. Peter Friedland who did one of the two thesis on the AI side of this project. They collaborated with Tim Lenoir on the historical record and they have all of this written down. The second thing is that once you notice that you have 300 users, and your machine (a) can't stand it, and (b) you're tying up a lot of ARPANET traffic with this interaction and a lot of the bandwidth of the SUMEX machine with this interaction, it gives rise to thoughts that there might a commercial aspect to this. You might be able to sell this service. So a group of people, the two biology collaborators, Dr. Kedes, Larry Kedes, and Dr. Brutlag, along with myself and Peter Friedland, formed a Silicon Valley startup company, which was called Intelligenetics.

**Nilsson:** This was after Friedland had his PhD finished, I guess?

**Feigenbaum:** Yes.

**Nilsson:** Or somewhere around.

**Feigenbaum:** I'm sure it was. I'm sure that was the case, but it was anyway 1980. To essentially rent a PDP-20 from DEC and offer this service, and also they offered a national service which was called GenBank. Now GenBank is run by the government, but at that time GenBank started out as being run by a private enterprise, which was this little company that was going to do this molecular biology service. This Intelligenetics as a company service didn't do very well. There were two reasons. One was that the Wisconsin group… Well, first of all, the academics, who saw that they had to pay for this service were not very happy with that, since they could get similar things from the University of Wisconsin and one other place, I think, for free. As soon as you had to pay one dime for it, they gravitated away from paying anything for it. And secondly the government didn't really want… There was an effort in the National Library of Medicine to bring the database inside the National Library of Medicine where they thought it belonged, rather than in a private company, which seemed a little bizarre. If this were going to be a national resource, then it should be inside the great library of the country for medicine and biology, the National Library of Medicine. Incidentally on whose board of trustees I served from in the late 1980s. A wonderful, wonderful institution. So anyway, Intelligenetics basically had no role. It sold itself to an industrial laboratory who thought that they might want to use the technology, and it changed its business to selling expert system development software, namely some version of the object-oriented software called KEE, Knowledge Engineering Environment.

Now I want to go back to the theses, the AI theses. I think there's a very interesting paring there that needs to be looked at, or at least needs to be mentioned. It doesn't need to be really looked at again by AI people, but it needs to be mentioned, because I think there's a big lesson there. Stefik's thesis looked like much more the kind of thesis that would be included in what you might call mainstream AI. It was fairly advanced ideas about planning from basic knowledge of the domain of discourse, which in this case was genetic engineering experiments and molecular biology. Even at that time that could be understood very well. Friedland's thesis took a different approach. It wasn't quite mainstream, but it was the approach that Roger Schank was telling the AI field that it really should take, which is what you might call the story-based approach, or the recipe-based approach. In many ways it's similar to what Roger was telling us we should do in the form of case-based reasoning. It wasn't exactly the same as case-based reasoning, but it was very similar to Schank's argument when he used the Chinese Restaurant Script.

**Nilsson:** Scripts, uh-huh.

**Feigenbaum:** I believe Peter called them experimental recipes, not Scripts, but it basically was a Scripts idea. And that was, in the end, very powerful. That to have those planning islands that constitute the elements of the Script, or the recipe in Friedland's case, was a very powerful method. I think AI has way under-exploited both the Scripts method specifically, and case based reasoning in general. But also it hasn't paid enough attention to Roger Schank's total view of episodic memory, of what we know as being encapsulated in stories. We tend to be -- we, you and I and others like us -- tend to be atomists, and the atoms -- or maybe we're not atomists, maybe we're molecularists -- but in any case the molecule or the atom is a logic expression. Whether it's expressed in first order logic, or a higher order logic, or whether it's expressed in an informal if-then rule case, it is a unit of knowledge, a piece of something.

**Nilsson:** A small unit basically.

**Feigenbaum:** Small unit. And Roger was saying that's not the way we as humans store our knowledge. We store our knowledge as strings, story strings, no matter what we're looking at. Like this whole event that we're into, I'm going to remember because you're here interviewing me, and we're talking, and it's a story, and things came up, and you asked an interesting question, and I responded—. We AI people, for a long time, simply looked down on that idea, and Roger simply didn't push it consistently enough. He certainly wrote enough about it.

**Nilsson:** Was that related to Minsky's Frames and his slots and fillers, except the frames would be very big?

**Feigenbaum:** No, no, not at all. Minsky's Frames idea is in fact the atomist idea. In fact I want to say right here in this oral history, and maybe I've said it before, that -- this is with all due admiration for Marvin who I respect and believe is one of the world's great AI scientists, and in fact one of the world's great computer scientists -- but the Frames idea was not a new idea. The Frames ideas went all the way back to list structures, actually the attribute-value part of list structures, back in IPL. The IPL list structures had a name, a symbolic name, just like the Frame had. It could have a separate name if you wanted to put as an attribute on the list, a special name, and it could have then a character string as the name. It could have any kinds of attributes and values, which you might think of as just facets of the Frame, and fill in the value of the facet. When Frames showed up in the so called “Frames paper”, people like Newell, and myself and Simon looked at it and said, "What's new about this?" It's just another way of talking about it. It's a great name. Actually it wasn't a great name, because it confused something. It confused the so called Frame problem with a data structure called Frames. So that was actually a kind of a disastrous choice of name. In any case Schank's stories, is not at all the attribute-value, or list-of-values approach that you find in the so-called Frames. For example, the KEE system, that we actually elaborated and built up a homebrew one called UNITS at Stanford in the mid… Mark Stefik and Peter Friedland really did the key programming on that, and it later became KEE. It got converted by Richard Fikes, who was Vice President for Research and Engineering of Intellicorp, into Knowledge Engineering Environment. The basis of that was, that was a Frame based environment. But if Roger Schank looked at that he wouldn't see anything about stories in there. He would see anything about recipes in there.

**Nilsson:** I think that Jerry Feldman at Berkeley and his new book, "From Molecule to Metaphor," talks a little bit about structures that are somewhat like these Scripts in terms of understanding natural language. So maybe they're coming back a little bit, we'll see.

**Feigenbaum:** Actually I'm glad you mentioned that, because I didn't even know about Jerry's book.

**Nilsson:**These were great AI dissertations, Stefik and Friedland's. Friedland went on, as you say, to help found Intelligenetics. There are other companies, also beginning at the same time, and one of them involves work that had started way back with the MYCIN Project and then EMYCIN. Do you want to say something about that?

**Feigenbaum:** Yeah, sure. Let me just give you the transition path, because it's very illustrative of the way that we were working. MYCIN was really the PhD thesis product of Ted Shortliffe, closely supervised by Bruce Buchanan, and with the domain expert being Stan Cohen, later famous for the Cohen/Boyer invention in genetic engineering. Ted was an ideal student to do this, because again, the mantra, "In the knowledge lies the power." You can pull AI algorithms out of the air all day long, but if you don't have the domain knowledge to make them work, you can't do the experiment. Ted was ideal. He was a computer science student getting his PhD, but at the same time he was a medical student getting his MD, and his head was getting filled up with all this knowledge. Furthermore he had the close collaboration of one of the faculty members. I'm sorry, I said one. I don't just mean one. There was really a team of faculty members at the medical school helping Ted and Bruce do that work. MYCIN turned out to be a very powerful knowledge-based system for, as you know, providing diagnosis of blood infections and their antibiotic therapies. Well, in good computer science fashion, they and the rest of group had the idea of extracting the computational essence of MYCIN. What if you wanted to use MYCIN to do something other than blood infections? Can you save a lot of programming by extracting the computational essence of MYCIN and using it for another purpose? That's an act of abstraction. Bill van Melle took it on as a

PhD thesis. Very powerful idea. Bill came up with a thing called EMYCIN, which stood for Essential MYCIN, or Empty MYCIN. We never really decided which, but it was EMYCIN. I decided to do a very quick test, personally, of how powerful EMYCIN was. Penny Nii had just come back from Systems Control where she had been working on the HASP Project, and I had a colleague up at Pacific Medical Center in San Francisco -- the old Stanford Medical School -- I had from a colleague from one of these NIH panels that I was serving on, a review panel. We had talked quite a lot about the kind of work that I do, and I'd been consulting [with] him on looking for a problem. In fact the problem became Larry Fagan's thesis eventually, Ventilator Manager. That was the problem. Again, a very powerful application. It was another one of these induction problems, where the streams of data were data pouring from instruments in the intensive care unit, and the induction was, "What's the patient's situation now? Is the patient in trouble? Is the patient going to die, blah, blah, blah?" But anyway, another thing that my friend recommended was pulmonary disease diagnosis, and recommended an expert there, Bob Fallet. Penny and I made some trips up… Well, Bob came down to Stanford first, and we spent a very long afternoon and evening with pizza in the middle, or Chinese food, or something, doing the first knowledge acquisition pass at Bob Fallet for lung disease diagnosis based on instrument data from a machine called a Spirometer. Then subsequently Penny and I went up there every week for the next two or three weeks, culling more, pulling out more and more cases from his file, adding knowledge until the point that it looked like diminishing returns. At that point we had a very good pulmonary disease diagnosis… I'm sorry, we had a very good program for doing the diagnosis from the Spirometer data, and printing it out in an English language form next to the instrument.

**Nilsson:** So the knowledge was in rules that EMYCIN could use?

**Feigenbaum:** Yes, and we used EMYCIN. Hence within a month I personally had, Penny and I had, shown the EMYCIN concept to be valid and powerful. We didn't have to do much to it. We were able to feed some working information back to van Melle for his thesis. But basically it was the right idea.

**Nilsson:** So was van Melle's thesis involved with the pulmonary diagnosis?

**Feigenbaum:** No, no, no. That was just simply what Penny and I did with Bob Fallet. Melle's thesis was EMYCIN, the program. When the news about EMYCIN hit the world, lots of people wanted it. We began to send out, literally, hundreds of tape copies of EMYCIN, and EMYCIN manuals, working documents. It was almost a full time business of a secretary and of the SUMEX staff to ship out these tapes. Probably Ted or Bruce Buchanan have a list of all the people -- or maybe Tom Rindfleisch had a list -- of all the people we sent these to, but it was really large. Because everyone was beginning to get interested in expert systems, and this was a really simple way to get into it. Well, that looked like that was another commercial opportunity. Why keep SUMEX busy? Why keep a secretary at Stanford busy when a Silicon Valley company could develop that software, make it industrial strength, deal with the servicing of it, etc., etc.. So a software company was born called TeKnowledge, whose ostensible goal it was to basically migrate EMYCIN into the commercial domain, and make it industrial strength, and sell it and apply it. TeKnowledge, in fact, didn't go down that path exactly. It ran into a significant intellectual property dispute issue with the Stanford Office of Technology Licensing, and decided in the end to do what so many other companies have done, which is simply to reinvent it on their own, off campus. Not use EMYCIN, but do their own thing. It cost them a million dollars to replicate that piece of work. It was definitely not worth it. By the time that product came out Osborne was selling a modest variation of it for 99 bucks for to run on the early PCs or Osborne machines. There's no way you're going to make enough money to recoup that investment. TeKnowledge, however, did have very large amounts of investment from industrial sponsors, and also many, many industrial contracts to do applied work based on that model.

**Nilsson:** So they helped other people build expert systems in-house?

**Feigenbaum:** Yes. Yes they did. They also did a constraint-satisfaction piece of software which apparently was the first. It was patented, and TeKnowledge won a patent dispute with another AI company, another one started by Stanford undergraduates. It came to a patent dispute, and TeKnowledge actually won that patent dispute in terms of building constraint-satisfaction systems similar to things like you find in the DEC program for manufacturing of mini-computers that was so well known at the time.

**Nilsson:** That was R1?

**Feigenbaum:** Well, R1 was a version of it. That was a Carnegie Mellon version of it, yeah.

**Nilsson:** Well, perhaps we'll quit here.

END OF TAPE 1

**Nilsson:** Ed, the companies that you talked about, namely IntelliGenetics, IntelliCorp, Techknowledge, can all be thought to have formed from this mother, SUMEX. Can you say a little bit about what SUMEX was?

**Feigenbaum:** Yes. SUMEX was… I can describe it as a computing facility, or I can describe it as a project. As a computing facility, it was the main DEC-equipment-oriented computing facility of the Heuristic Programming Project and, later, the Knowledge Systems Laboratory. As the project grew and changed its name to an umbrella name, SUMEX was its main computing facility. That's no different than the Stanford AI Lab having its own computing facility. We had one, except it had a separate name, because it was a separately funded project by the National Institutes of Health. It was funded as a national resource to assist computing in artificial intelligence, apply to medicine and biology. That's where the A-I-M comes in, Artificial Intelligence in Medicine. The SUMEX stands for Stanford University Medical Experimental System dash A-I-M, Artificial Intelligence in Medicine, AIM. AIM had its own steering committee, which regulated the national part of the resource dispersement. That committee had people on it, like Saul Amarel. It was chaired for a while, at least, by Don Lindberg, who, at the time, was collaborating with us and Amarel on expert systems in medicine from the University of Missouri. Now, of course, for the last 30 years or so, he's been director of the National Library of Medicine. A couple things about SUMEX. First of all, it was run by an absolutely superb first-rate computer science and technology person, Tom Rindfleisch. Later, Tom, because his information processing role was so excellent at the medical school, and so admired, he became the head librarian at the Lane Medical Library, even though he was not trained as a librarian. He was brought in to kind of modernize the Lane Library into the information processing age. But that was after the SUMEX Project was no longer funded by NIH and basically went away.

**Nilsson:** So the funding of all of this research, then, shifted from DARPA to the National Institute of Health.

**Feigenbaum:** The funding for the Heuristic Programming Project was and has been all along very strongly funded by DARPA. But the computing facility that we use was funded by the NIH. And then DARPA did this nice thing -- Licklider did -- of allowing us to be on the ARPANET. SUMEX itself… You'll recall that the Stanford AI Lab people, in many of the things that have been written about them, are credited, correctly so, with being highly innovative in the computing facility they were running, for SAIL [Stanford Artificial Intellgence Laboratory]. Similarly, SUMEX was highly innovative. It had extraordinarily good people working for it, both in the development of software, in making sure that the service was first-rate. In those days, timesharing systems could be flaky, especially evolving timesharing systems, like the TENEX System, as it evolved -- DARPA paid for it and BB&N was working on it -- or Interlisp system. The people at SUMEX made sure that this was all done very reliably. But in addition, there were some remarkable contributions. Like, for example, we're sitting here in the Computer History Museum. If you walk down that hallway, you'll see a picture of Bill Yeager. Bill Yeager is credited as being the inventor of the router. That idea was essentially lifted from Stanford and Bill Yeager and SUMEX and the computer science department by the two people who founded CISCO. Then grew up the mythology that they had somehow invented it. Well, the historians here worked on the problem, and they didn't. Bill Yeager invented the router. Similar things happened in the context of SUMEX. So SUMEX is a real landmark in computing facilities.

**Nilsson:** Now, some other very important things happened about that time. And that, besides the companies, involves books that were written. You were involved in several of them. For example, The Handbook of AI, a very important set of volumes. Do you want to say some things about that?

**Feigenbaum:** Yeah, I think one should. I'm going to talk about The Handbook of AI. And then let me also mention the book called Fifth Generation. Did you have any other books in mind?

**Nilsson:** Well, there was Rise of the Expert Company.

**Feigenbaum:** Oh, yeah, okay. That's five years later. Okay. Let me talk about all three of them then. Somewhere in the early to mid-1970s, it occurred to me that essentially one had to change the scale by which AI was being practiced. It was being practiced up til that time by essentially a handful of people, who were usually graduates of Minsky at MIT or, occasionally one of McCarthy's graduate students, like Raj Reddy, who then went to Carnegie Mellon and graduated a few people, or Newell and Simon's students, like myself. And nurtured in a few places, like SRI. This was changing, gradually. We had Saul Amarel developing a reasonable capability at Rutgers University, and a few other places where AI was springing up. But I didn't see that as being… If we really ever wanted it to get to where we were going, namely the ultra-intelligent machine, the complete model of human information processing, or whatever grand vision we were going toward, we weren't going to get there in this piecemeal fashion. I was very encouraged by what was happening, but wanted to just push it. In effect, there was no sourcebook for everything that AI knew. Well, that was also true at the time of Diderot and his encyclopedia. He wanted to get everything in there. And I wanted to get everything into one piece of work, which turned out to be The Handbook of Artificial Intelligence. It wasn't called The Encyclopedia of Artificial Intelligence. I think there are some reasons why "handbook" was a wrong choice of language. For example, Science Magazine would not review it because they never review handbooks, even though it really was an encyclopedia. The way to accomplish this was to harness very large amounts of the smartest talent around, which is Stanford graduate students, and set up a course in which they were reviewing the techniques and methods and history and accomplishments of AI for themselves. But the little packages that they would produce were packages which became chapters or sections of chapters of this enormous encyclopedia, the Handbook. The plan for that was laid out in an elaborate… Actually, it wasn't ad hoc. It was really laid out in advance. And that's not to say that the outline didn't change over time, because I'm sure it did. I'm sure there are versions of it. But it really was there. And it was, "Let's do this, and then let's do this, and then let's do this. Next year, we'll hold the same seminar and we're going to do this and this and this and this and this. And we'll hire some people in the summer, and they'll do this, this, this, this and this." To the extent that this needed funds, partly to hire people and partly because we wanted and benefited by the editing skill of Diane Kinerva, the best book editor I've ever seen; we hired her. All of that was paid for by the DARPA contract, which, like I said in the past, wasn't accounted for on a dollar-by-dollar basis for, "What are you contributing to the defense effort?" It was, "What are you contributing to science? And if it's good, we'll do it." That partly funded The Handbook of Artificial Intelligence infrastructure. And then, of course, probably the single most important person in the whole project and source of effort was Avron Barr, a very skilled writer. Really knew AI. Just excellent, in general, in organizing the literature and making it comprehensible to people. So, in a sense, I was the overall organizer and the editor of what Diane and Avron were doing with the student work. Eventually we did a three-volume work. It seemed like we weren't quite finished yet, because AI moves on. So we thought of a volume four. Unfortunately the kind of effort that organized volume three, I just couldn't bring myself to do. That was started in '76. Volume one, I think, was published in '81 and volume two and three in '82 and '83, somewhere around there. It was just overwhelming effort. So we didn't organize volume four, which was an attempt to bring the other ones up-to-date, in exactly that way. We organized it as chapters that people would write. For example, probably the single best piece on blackboard systems is Penny Nii's chapter in handbook four of The Handbook of AI. There are many other articles which were bringing fields up-to-date. For that one, Avron and I recruited help. And that was Paul Cohen, who was a Ph.D. student at the time, and was willing to help us, and did a wonderful job in organizing those chapters. Now, The Handbook of AI had an enormous influence. It was everywhere. It was on the Library of Science Book List. It was one of those things you would get as a freebie if you joined the Library of Science Book List. You could order that as your set of volumes that you would get for free. It sold a lot of copies. It contributed a lot of royalties. The royalty income fed back, in large measure, to the Heuristic Programming Project to fund more research. It seemed, from what I have heard from others, who have come to me over the years, it made a big difference in people's decision as to whether to go into the field or not. Now, they had something they could read. It was aimed for them. It was aimed at the Scientific American reader, not at the AI jargon audience. And they could think about AI.

**Nilsson:** It's still a valuable source to find out about what some of the programs and ideas were during those years.

**Feigenbaum:** So let me talk about The Fifth Generation. The community of potential practitioners was not the only community that needed to be reached. There is a much larger community of people out in the world who are generally interested in science, technology and its effect on society. For example, that community includes some, but not all, senators in the U.S. Senate, or Representatives, or members of the executive branch who would have to fund either research or development or education in artificial intelligence. And a much broader community of influence outside of Washington. A New York audience. A broadly-spread industrial audience. That particular audience was quite interested, at the time, in the surge of Japan into the world's arena of technology and business. Just about that time, the Japanese organized a project to essentially do the knowledge-based systems portion of the AI enterprise, but do it in a style different from the style that we were accustomed to in this country. For one thing, they wanted to do it in the "I-am-not-LISP style". Because they had been kind of faulted in the past for being imitators, in this case, they weren't going to be imitators. So they chose Prolog, because it was European and they didn't care if they imitated Europeans. But they were going to do Prolog. They were going to do it as a machine language. Instead of having the normal machine language, you would have Prolog as the… or, at least, a layer from which it was very easy to compile ordinary Prolog into that layer. In addition, in order to get consensus in Japan for such a large expenditure, namely of about $1 billion at the time -- and that was a lot of money for 1980, 1981 -- they were going to include parallel computing in their initiative. So that the Prolog machine would indeed be built out of parallel computational elements. They decided to do it in a new style with the creation of a special laboratory for this project, rather than farming the funds out to companies, as had been done with other national projects. And it was under the direction of a person who was a very quiet but charismatic leader, namely Dr. Fuchi of one of the government laboratories, ETL. I saw that plan early, because by that time I had been in contact with Japanese quite a lot. I saw that plan early and decided, "That is really an interesting plan, and that is worthy of note. Let's write about that plan. Let's write about the Japanese intentions to essentially close the gap with Americans in computation. And let's do it in a way that talks about our own effort, talks about the successes of the DARPA Program, the network, and AI in general, and so on." Well, I had been delivering lots of lectures about this. It was a time of many, many lectures, going around the country. It ended up being far too many lectures. My doctor, at one point, told me to stop traveling so much, it was hurting my health. But I was able to sit down with a friend of mine, a person whose writing I respect a great deal, very old friend of mine, Pamela McCorduck. Pamela had written a great book on AI, early AI history, called Machines Who Think, now in its 25th anniversary edition. I was able to convince Pamela to be a coauthor, be the writer of this book, sit down and listen to me give these lectures to her, and then let's have lots of conversation about it. Pamela and I spent dozens, maybe hundreds, of hours together, she listening, writing down stuff, asking questions, being a knowledge engineer, bringing the human element into the picture, which of course I, as an AI techie, would tend to disregard. But she would make sure it's there, and satisfying the key thing about about stories. I remember reading, somewhere, someone saying, "You know, it's okay to be wrong as long as you're not boring." Pamela ensured that we weren't boring, and she really made it an interesting book.

**Nilsson:** Sold a lot of copies, right?

**Feigenbaum:** It sold a ton of copies. By the time it was finished with its authorized and unauthorized translations around the world, it was about 500,000 copies, which is very big. Even books like this that sell 25,000 are thought to be pretty big sellers. It was in hardback. It was in paperback. It was in translations in many different languages, including the Chinese, who translated it without authorization and then just distributed it widely.

**Nilsson:** So it got into many languages.

**Feigenbaum:** Yeah. So that really caught people's attention. I had visits from… For example, Senator Bingaman, of New Mexico, and his staff came to talk to me about what are the issues, what could be done, were the Japanese a real problem in this, was the U.S. falling behind. I can't take credit for the entire current of activity. But there was a current that resulted in the formation of an inter-company industry consortium called MCC, in Austin, Texas, led by former Admiral Bobby Inman, to push the United States ahead in various areas of technology. And it had an excellent AI group, started by Woody Bledsoe of the University of Texas and including Doug Lenat, who eventually, when MCC folded, he eventually spun off his own company to do what he was doing there.

**Nilsson:** You think it had an influence also in helping Bob Kahn get the so-called Strategic Computing Program started at DARPA?

**Feigenbaum:** It had a huge influence in that. It got congressional staffs behind that. I can't express enough admiration for Bob Kahn's plan, his strategic plan, called Strategic Computing, a DARPA plan which, even today, is regarded as a model of a DARPA planning activity. It included applied AI, that is, more expert systems aimed at the military context. It included research in knowledge representation and advanced methods in knowledge-based processing. But it also included lots of work in parallel computing, which was also a theme of the time, not started necessarily by the Japanese  Gordon Bell started a company called Encore, which was a multiprocessor computing company. And there were several others, Stardent-- Ardent, Stardent, after the merger. Bob's program included all of that. There's a very interesting book by a historian on the Strategic Computing Project, which I think does not do justice to the deep insights of Bob's plan. I think that book was written by a historian with an agenda. The agenda was to show that Bob Kahn was really satisfying the needs of his friends in various portions of the DARPA community, and not seeing the vision. But The Fifth Generation book definitely helped sell that project. So expert systems kept tumbling out of companies, out of laboratories, in medicine, in engineering and so on. In 1987, your friend and mine, Peter Hart, one of the early pioneers of AI, came to my office and said, "Ed, you need to do a book similar to the book that Peters had done out of the Stanford Business School," which was a collection of tales of excellence in business. Do you remember the name of the book, Tom Peter's book? It was a huge bestseller, but it was a series of case studies of excellence in businesses. Peter said, "How about doing a series of case studies of excellence in expert systems?"

**Nilsson:** In business.

**Feigenbaum:** Yeah. I took Peter seriously and got going on that. I had the help, again, of Pamela McCorduck. In this particular instance, I got the help of Penny Nii also to help me do interviews. We had so many interviews to do that I and Penny did a lot of them. Pamela did some of them. Pamela did, again, a great deal of the write-up. And that was the book The Rise of the Expert Company. The only thing wrong with that book… That was a excellent book. The only thing wrong with it was its title. I think that's a pretty dumb title, The Rise of the Expert Company. It sounds something like The Rise of the Third Reich, or something like that. I think we could've done way better on a catchy title.

**Nilsson:** What was the book about?

**Feigenbaum:** What we did was find great stories of companies that had been applying AI. For example, the IBM effort to apply expert systems. The whole program at IBM was led by Herb Shore, and by the IBM management who were fully behind it. They actually had an expert system center, on Page Mill Road, in Palo Alto. That was one of the stories. DEC's story of configuration, configuring computers, minicomputers, was another one of the stories. DuPont's different approach, led by Ed Mauler, a DuPont chemist. Mauler had the idea that you didn't really need $10,000 pieces of software and $10,000 minicomputers. What you needed was smaller pieces of software and personal computers and make the individuals in the company their own knowledge engineer. They didn't need a knowledge engineer. They're smart enough to be their own knowledge engineer. Give them the right tools. Give them a little bit of training. That was the Mauler story. We also had a terrific chapter there, chapter three, which is the best summary, in a concise way, of what is an expert system. That's the chapter you would hand out to people if they just wanted to know. That's kind of a Scientific American article on what is an expert system.

**Nilsson:** Now, do you think that a book like that, talking about how expertise and ability to marshall expertise in companies, and how it should be done, contributed to the demise of expert systems companies who would sell expert systems? Now the companies themselves could imagine that they could build their own expert systems, and it got diffused way into the companies. In a way, that made it look as though you don't need to have an AI company doing this for you. You could do it right in the company.

**Feigenbaum:** Well, there's some truth in that, and the Mauler story that I told you about is a good example of that. But there was still major, major applications of big size, that needed teams of people who were skilled at the art. There are several reasons why the expert systems companies didn't survive. I can give you some reasons that relate to companies that I helped to found. I can't, in fact, give you a reason why Peter Hart's company didn't succeed. Nothing could've been run better than Peter's company, and the product was superb. I can't explain why that company didn't succeed. But I know why some things, like, for example, Teknowledge…. There was pressure to bring in a CEO early on. The CEO was a very good CEO for starting the company, for getting it going, organizing a team. Well, we had the team, actually. We had 20 founders. A large number of them left the HPP and just went to work for Teknowledge. We even got people from SRI and other places. Like, from the Prospector Project, I think Georgia Sutherland came over to work at Teknowledge. But the person who was able to raise the money wasn't a very good company builder. Therefore, the company really was mismanaged and basically went down the tubes. Spent too much money on things that they shouldn't have spent too much money on. IntelliCorp, which was the morphing of IntelliGenetics, had a very excellent applications group under John Koontz. You know John. He's one of our former students at the Heuristic Programming Project. But the company had in mind a productization strategy, rather than a system building strategy, because it was, quote, "well-known," unquote, to business people, who were the managers that we brought in, that you made a lot of money on product. You didn't make a lot of money on services. That turns out to be incidentally quite a big fiction. But at that time, it was strongly believed. So IntelliCorp turned down the gain on helping people who had big problems to solve. And they turned up the gain on spending money on productization. Well, the products just didn't sell. As soon as the R&D groups got finished with the products, learning how you do this, they would re-implement things in more conventional languages. Partly because their IT people, so-called information systems people, the much feared and dreaded information systems people in large companies, insisted that they get out of LISP and they get into more conventional things.

**Nilsson:** Then the personal computer came along, and a lot of these things did not need big computers. It needed big software packages. Well, right around that time also, Ed, I forgot exactly the year, you were a co-recipient of ACM's prestigious Turing Award. Can you say anything about that, and maybe a little bit about some of the things you said in your Turing Award speech?

**Feigenbaum:** Actually, we're now moving from the '80s into the '90s. The Turing Award was a 1994 award, which was actually granted in a ceremony in 1995 and was joint with Raj Reddy at Carnegie Mellon. The award citation was oriented toward applications of artificial intelligence, which is an interesting thing in itself I'd like to come back to at the end of saying something about the speech. The speech, of course, is published, well, it's online anyway. Anyone can get it on the web, and it's in the communications of the ACM. So there's no need to go into great detail about it. I just want to say two things about it. I wanted to highlight a theme that I thought was broadly applicable over the entire space of AI applications. Maybe, to use Roger Schank's kind of terminology, it's one of AI's great stories. I mean, it's a story we tell, not a story of accomplishment. It's a story we tell of what we're doing. It has to do with a thing called the What-How Spectrum. Those of us who are old enough, all started out at the "how" end of the spectrum. The first things I programmed were 005, space, 0100 -- for cell 100 -- 0105, for jump to cell 105. 005 might've been "clear and add into the accumulator". That's the ultimate "how". That's the "how" that the logic circuits in the computer understand. Well, we quickly moved away from that to things like, "Well, you don't really want 005. You want CLA for clear and add." And then we quickly moved -- not so quickly -- but by 1956 and '57, we had moved away from that to writing algebraic expressions. We computer scientists had started to write algebraic expressions for the ordinary computational things. That was Fortran, and Perlis' IT compiler, and several others. In AI, of course, the IPLs started to do the same thing. Then LISP carried it to a beautiful conclusion for symbolic expressions. We were moving away from the "how" end of the spectrum toward the "what is it you want to do" end of the spectrum. Now, a Fortran algebraic expression is not exactly what it is you want to do in mathematics, but it's a lot closer. It uses the same terminology. It uses the same mental framework. And you can kind of understand why you're doing algebra. Then a computer program, called a compiler, translates that expression into how it shall be done, step-by-step, by a computer. Well, AI is really a series of steps toward the "what" end of the spectrum. The ideal AI program is one that accepts a user's goals in the context of a user's workspace, needs, understanding the various dimensions of user habits, and does a kind of super-compilation, which you and I would call problem solving, to figure out how to translate from user goals all the way down into how it is the computer should carry that out. That would be the ideal AI program. In fact, we're not all that far away from that. It happens we're doing this lecture -- for those of you who are watching it way in the future -- we're doing it in 2007. Recently I went over to SRI, where they gave me a demo of an integrated AI system that they have called POW for office assistant tasks. It's amazing. I mean, it's remarkable how much of user needs and user understanding one can carry out and infer, even today. To bring this long story about the Turing Award to a conclusion, at the time that I wrote this -- remember, it was a '94 award but it was awarded in '95, when I had to give my speech. I was chief scientist of the U.S. Air Force at the time. I was deeply concerned with the problems of software and software development, because that's really what I was hired to do as chief scientist of the air force. We can get into that more in another question. But anyway, what I did was try to apply the What-How spectrum to the idea of more automatic program development. Now, this is not new for A.I. In 1970, we started to have meetings with Bob Alsar [ph?] and Cordell Green and lots of other people on automatic programming. Then Cordell started a company called Kestrel Institute, whose goal was to do automatic programming. Bob Alsar had a long series of projects at ISI, on that. But we never really got that far. We never really got to the point of expressing what it is we would like a system to do for us, and having the program do the reasoning that builds the system. So my talk was oriented toward, "Here's the concept. Let's do it more for software."

**Nilsson:** Perhaps before the end of the ‘80s, as you were doing this, coming off of the work that led to these companies and the HPP, you got interested in parallel computing. Do you want to take a minute or two to talk a little bit about that?

**Feigenbaum:** Yeah, I would like to talk about it, and I have a feeling that we better delay for a moment.

END OF TAPE 2

**Nilsson:** So, Ed, you had some work toward the late '80s, I think, about parallelizing Blackboards, and you developed an interest in parallel computing. Let's talk a bit about that.

**Feigenbaum:** The thoughts about parallelization was initially a technology push. It originated in our lab in 1976. I believe it was roughly '76, let's say plus or minus a year, but it's easily checked in my archives, under the code… the keyword would be HYDRA. Now HYDRA was a term which was used in other universities, computer science departments, for different things. Other machines were HYDRAs. But in any case, our idea was a HYDRA which would be a multi-headed, multi-processor machine. The technology push was Lederberg and I saw that these Intel chips that were being produced around 1976 were no longer toys. They were significant machines and by Moore's Law they were going to become more significant. We began to ask the question, "Well, what would happen if we had a computer organization of that sort?" Remember we're existing now in 1976 in an era in which we were time-slicing a relatively small machine: at that time, I believe, it was just a PDP-10 and then later a DEC-20. So we're not getting a lot of computing in timesharing. So the idea of getting a boost by parallelization of these inexpensive but more and more capable devices seemed attractive. Okay, that's step one. But nothing happened. We just coasted along doing what we needed to do. Nothing seemed to be demanding it of us at the moment. We seemed to have sufficient computing to do what we needed to do. Then came the plan for the Japanese Fifth Generation Project that said they were going to view parallelization as a goal. It basically was going to be AI and parallelization together. We thought to ourselves, gee, you know, we've been thinking those ideas for five years now. This is HYDRA. Maybe we ought to be interested in that. But just at the time we had had some successes, quite a run of success, with the Blackboard model. And the Blackboard model was intrinsically parallelizable, the so called knowledge sources. I won't go into them here in this interview, but anyone who wants to go back to, let's say, Volume 4 of the Handbook of AI, look up Penny Nii's article, or her AI Magazine articles on it, will see that the knowledge sources are essentially a parallelization concept.

**Nilsson:** She thought of them as running asynchronously.

**Feigenbaum:** That's right.

**Nilsson:** All writing independently of each other, and reacting to what others wrote.

**Feigenbaum:** Exactly right. And the only question that in fact came up later as a technical question was, do we centralize the Blackboard? That is, do we literally have a Blackboard centralized where everyone writes, or do we distribute the Blackboard to all the processors? Again nothing happened, until Bob Kahn structured his Strategic Computing Program, which essentially encouraged -- both allowed and encouraged -- AI to be merged with parallel architectures, structures. So we put in a proposal of that sort, and that proposal was funded. We were enabled to do this kind of work, and I just want to make a few quick comments about it. Being enabled to do the work is not the same as having the human resources to do it. In order to get the human resources to do it we found an industrial affiliates company, namely Digital Equipment Corporation, that was interested both in AI, because they had had a success in the AI, in the expert system area, and they were interested in parallel computation. And they provided us with a computer architect. His name is Bruce Delagi. Bruce did a wonderful job, stayed with us for many years until the project ended, and then he worked either for DEC or someone else in Silicon Valley after that. First of all we developed two different models, in our usual experimental way. One is not good enough. Look at other alternatives. We developed two different ones, CAGE which stood for Concurrent AGE. That's pretty obvious. That's the AGE System moved into a parallelization structure mode. Then there was another one called POLIGON, which dealt with distributive Blackboards, and a set of other techniques. Many papers have been published on CAGE and POLIGON, so it's not appropriate to go into it here. But I do want to say that several things. First of all, except for Bruce Delagi, we were all software people. I've never done a hardware thing in my life, other than plugging cables together in my hi-fi system. I'm a real software guy, and so are all the other people. My decision was, no way we're going to touch hardware. This is going to be software simulation parallel computer research. Then we can make lots of changes fast, big changes fast, without worrying about the expense of building all these machines, which I didn't know how to do anyway. Well, turns out that is one of those big mistakes you make. Not that I could have done it, but the world, the community of people who were interested in parallelization, simply did not respect software models of parallelization. They were absolutely insistent that you build hard things, like John Hennessy was doing, like Anoop Gupta was doing in our Computer Systems Lab. They had to be things you could see and touch. That was what a parallel computer was. It wasn't a computer simulation of a parallel computer. So we lacked credibility. Even though the ideas were quite successful, and we applied them to some hard problems that were actually, believe it or not, militarily related problems. It happened that we knew some of those that we could sanitize. And the results were excellent. They were excellent, but they were only excellent in software simulation, and who trusts those software guys? They could be manipulating those numbers, you know, who knows? Anyway, that's number one problem. Number two problem is, Moore's Law didn't go away, and Moore's Law meant that every, roughly speaking, every three years you were getting a factor of ten out of normal computing. And wow, that's a lot. That's like every three years you're getting ten more computers, and we were only working on a low degree of multiprocessor parallelism. What role did multiprocessors have when Moore's Law is chugging along? And that particular phenomenon is the one that killed all these multiprocessor companies like Encore, Ardent, Stardent, whatever. They never made it into the big time. Now, final point about all this is a long term vision issue. Or to put it another way, why DARPA, or somebody else ought to be funding long range research. Thank you, Bob Kahn, for thinking about long range research. It turns out that now we are running out of Moore's Law. Now you see the appearance of multicore computers. Like Apple, common is two core, and they're introducing four core computers, and maybe next year six core computers. And the PC world already has its six or eight core computers. Well, now you're talking! Now these are the numbers that we were experimenting with. Now you can do software that would exploit what is common in the computing world. How many years are we into this? We started that parallel computing research in 1985, roughly speaking, getting the money from DARPA. We're now 20 plus years into it, and now we need to dig out those CAGE and POLIGON papers and re-implement that stuff. That is the right stuff.

**Nilsson:** I hope somebody will do that. Now, right about that time, I think it was in 1986, you were elected to the National Academy of Engineering. Do you want to say something about what the citation was ,the reason that you were elected? Did that summarize some of the work that you had done up to that time?

**Feigenbaum:** Nils, to be honest with you I don't remember. Do you happen to have it there?

**Nilsson:** No.

**Feigenbaum:** Anyway, I don't remember and that's something we'll put into the transcript. I just don't remember. I can give you though a feeling behind it. I mentioned that the Turing Award Citation had to do with applied AI. When I got started on this in the early work that we talked about last time -- starting in 1962 with what I wrote about inductive inference and then formulating a problem area to work in, and collaborating with Lederberg, and hiring Buchanan-- we had absolutely no idea that we were working on applied AI. We were putting together a sandbox in which we could experiment with these inductive processes. Then it turned out, yes, we had stumbled on a mother-lode. If you got the knowledge out of the experts, and put them to work with these AI methods, indeed these became useful from an applications point of view. We did so many of them that eventually I got known as a person who did applications. However, I tried to keep in mind all the time, throughout the years and decades that we were not an application shop. That we were always on a vector toward something bigger in AI. It's great if the applications come off of there, and I love useful things, and at the very heart of me I'm an engineer. I have to admit that. Being in the National Academy of Engineering is a much more valid thing than my being in the National Academy of Science, because I feel like an engineer who builds things. I'm glad to have these applications, but that wasn't the main route that I felt I was on.

**Nilsson:** Now, there were a couple of other activities that you were involved in, in the '90s. One was that you visited the Stanford campus in Kyoto and taught courses there. Another you became, as you mentioned a little bit earlier, Chief Scientist of the Air Force. So let's say a little bit about each of those two activities.

**Feigenbaum:** Okay. Let me slide into that by talking a little bit about the period of, roughly speaking, 1990, late '80s and 1990. By that period of time the head of steam that we had built up over parallel computing was kind of spent, and it was necessary to both renew the DARPA application for more research in parallel computing, and invent some new ideas. By that time I really didn't feel like doing that any more. I just didn't feel like. Okay, I had done parallel computing, it seemed like a good idea at the time and then it didn't seem like such a good idea by 1989 or so. I didn't want to write that DARPA proposal nor even think of new ideas for it. That's number one. Number two is, I had run out of gas on writing DARPA proposals. There's a time when that's appropriate, and a time when you've had enough of it in your life, and you just don't want to do it anymore. So I told the group, "Hey group, most of you are highly prestigious senior scientists in your own right. Write the proposals for what you want to do, and I'll be the PI for them, but I don't want to write those proposals." Well, within one year I had lost Harold Brown to go to Hewlett Packard -- HP Labs -- because he didn't want to write any proposals. Within two years I had lost Penny Nii, because she didn't want to write any proposals. Barbara Hayes-Roth did stick around writing proposals, and she did, in fact, do several interesting things thereafter. But what it amounts to is that with this lack of sustained funding the lab was kind of falling apart. This team effort was falling apart. Plus the fact that we had been collaborating with, and later hired, Rich Fikes to continue the Knowledge Systems Lab in his own image, that is, along the path that he wanted to lead it, which had to do with the How Things Work Project, which I had gotten started, but he was greatly helpful, and then took it over. The ontology work, the hiring of Tom Gruber to work on ontologies. All of that shifted the direction, and Rich Fikes took on the job of writing those government proposals. So there I was, when the phone rings and it's Jim Gibbons on the other end, Dean of Engineering. "Well Ed, I have a problem. We're in a project, Sloan Foundation funded project, with the Business School -- School of Engineering and the Business School -- but it looks like the Business School is really running it, and it looks like the School of Engineering is playing no role in it. Could you go over there and look at it, because it doesn't look like it's going so well." So I went over there, and sure enough it really wasn't going so well, and in two respects. One was a leadership problem, which was solved a year later by Bill Miller taking over the leadership of that project, and then Bill did a wonderful job for the rest of Sloan grant. The other was I noticed that they weren't doing anything whatever about software, like software didn't exist. Like this was Silicon Valley, meaning silicon, and discs, and Ethernets, and things like that. So I started the software subproject of the Stanford Computer Industry Project, which is the official name of the Sloan-sponsored project. It was an industry-oriented project looking at a piece of the computer industry as a whole. I didn't have a major research focus at the time, so that was good enough for one. But as I began to define that project I had a commitment; I had signed up for maybe a year and a half in advance, or two years in advance, to go teach at the Stanford Japan Center in Kyoto in the spring of 1993, when they have the engineering students over. Spring quarter is for engineering students, both undergraduate students at the junior or senior level, usually senior, and then graduate students. Anoop Gupta was also teaching that quarter also, so it was a very interesting. Yumi Iwasaki was there, obviously. So it was a very good quarter. But when I got there I said to myself, "It's ridiculous to teach a course that I could teach on campus to these same people, at the Stanford campus, why bother to teach it in Kyoto? Let's do something unique to Japan. Well what's unique to Japan? Well, let's give it a try. Let's do a software study of the Japanese software industry as a trial run of a later study of the American software industry." I got the support of Professor Imai at the Stanford Japan Center Research Section, and funding from him, basically, to take students around with me on lots of interviews of various kinds of companies in the Japan software industry. We put together a very good… The students, of course, wrote reports on individual visits, and integrative reports for their term papers. They had a great time doing it. One of my students in that process was Jim Mynall's [ph?] daughter. She had a great time. Candace, was a wonderful help. Then later on I wrote a paper on it called "Japanese Software: Where's the Walkman. " Meaning, to use another phrase in English that might have been more appropriate in English, "Where's the beef." There's nothing here. There's no there there in Japanese software. That was the gist of the article and it was published in a January 1996 volume called the Future of Software. There was an article on the future of Japanese software.

When I came back there wasn't a heck of a lot of time to pursue the American side study, because the phone rang again. This time the phone rang and it was Professor Sheila Widnall of the Department of Aeronautics and Astronautics of MIT, a person that I knew from the American Academy of Arts and Sciences. I was on a committee that she was supposed to chair, but it hadn't met yet, but still Sheila and I barely knew each other. She called me to ask if I knew of anyone who was interested in filling the job of Chief Scientist of the Air Force, because she was about to go to Washington to take some kind of a role which she couldn't talk about on the phone. Dd I know of anyone who wanted to do this, and by the way, was I interested in doing this. Well, it happened that I knew a lot about this job, because I had a friend in the Aero/Astro Department at Stanford who had had this job in 1965 and who had told me over the years what a great job it was. Not only personally interesting, but it's a good national contribution. I thought about it briefly, discussed it with my wife, because it involved a move to Washington, and called back Sheila and told her yes, I was interested in doing that. Well, it turns out the job that Sheila was going to was Secretary of the Air Force in the Clinton Administration under Bill Perry, who was Defense Secretary --- I'm sorry, he was Deputy Defense Secretary and later moved up to being Defense Secretary. When I went to the Pentagon to look over this job, and for them to look over me, I asked if I could have an appointment with Bill, because I knew Bill from Stanford contexts. Bill was very busy, of course, but he took time to talk to me about this, and he said, "Ed, can there possibly be a better time? I'm here, Anita Jones is DDRNE, Sheila Widnall, a professor, she's Secretary of the Air Force, basically your boss. Could there possibly be a better time?" And there really wasn't a better time. It was a fabulous time.

**Nilsson:** Great time in your career also.

**Feigenbaum:** Yeah. My job was to be a window on science for the Chief of Staff of the Air Force. Actually I worked for the Chief, I didn't work for Sheila, for Sheila Widnall. I worked for the Chief of Staff of the Air Force. My job was to do quick and special studies for them, to be a window on science if anything important came up, to point them in the right direction, to do liaison with the Scientific Advisory Board of the Air Force, which is a high level group of 60 excellent volunteers that do studies for the Air Force -- kind of a National Research Council devoted strictly to the Air Force. To do liaison with other science establishments in Washington, like the National Science Foundation Director and staff, the Commerce Department's Assistant Secretary for Science and Technology. To do visits to lots of Air Force bases to find out what's really going on. In fact my first boss, first Chief of Staff that I worked for, before they changed, was a guy who told me, "I want you out of this building as much as possible. You don't learn anything in this building." That's the Pentagon. Basically he was right; you don't learn anything in the Pentagon. You have to really go outside to do the job right. So I spent a lot of time on the road visiting Air Force facilities, flying in Air Force planes.

**Nilsson:** What did your doctor say about that?

**Feigenbaum:** I, yeah, right, that was… But it was fun. I guess it was good for my psyche, or something. Oh, I also had a lot of liaison with other services. For example, the Army had actually established an expert systems center in the Pentagon. I was able to go down to the basement of the Pentagon and actually help them with specific projects, look at what they're doing, recommend things to them and to the Air Force. I was able to do things jointly with the Navy as well. The Navy brass was just down the hall from where I was. I just found it a terrific opportunity. It was very interesting to be inside the beltway, frankly. Bill Perry once said, "The beltway, inside the beltway," something like, "The beltway surrounds..." I can't think of this exact thing, but it's, "The beltway is 50 square miles surrounded by reality. " Washington D.C. is a very special place. It is the politics city in the same way that Detroit is the auto city, or Silicon Valley is the chips place.

**Nilsson:** And New York's financial.

**Feigenbaum:** Yeah, New York financial, right. Hollywood, entertainment.

**Nilsson:** You were on leave from Stanford at that time, and then you came back to Stanford. You were involved in teaching courses?

**Feigenbaum:** Well, I came back in '97, and let me contrast the situation. When Sheila Widnall went back to MIT at roughly the same time, plus or minus a few months, I can't remember exactly, but when she went back to MIT she was given the title of an Institute Professor, which means she didn't have departmental responsibility. She had broad ranging responsibilities all over the organization. Stanford doesn't have that. Carnegie Mellon has it. Raj Reddy and Al Newell were Institute Professors. For Stanford it's, "Oh, you're back. You going to teach introduction to AI in the fall?" Well, when you get off a big stage, and you go back to teach what you were teaching 30 years ago from new notes that you have to carefully prepare, because things have moved along, it's not the most enticing thing in the world. Plus the fact that if you want to get a research program moving again you had to go write these proposals, which I said in 1990 I wasn't going to do any more of. All of that pointed in the direction of: don't really need this job. Let's become a Professor Emeritus. I can do what I want, don't have to teach, don't have to write research proposals. You know that very well.

**Nilsson:** Well, before you retired, you were also appointed to a Chair at Stanford, the Kumagai Professorship, right?

**Feigenbaum:** Yes. I don't remember what year that was, but I think you were the first Kumagai Professor.

**Nilsson:** I think so.

**Feigenbaum:** And I think I was the second Kumagai Professor.

**Nilsson:** Now we're both Kumagai Professors Emeritus.

**Feigenbaum:** Emeritus. Mr. Kumagai is a person whose family made a fortune in the construction industry in Japan, and he had the good sense to donate a substantial chunk of that fortune to funding a professorship at Stanford in the Engineering School. It was originally aimed at Civil Engineering, but Dean Gibbons reoriented it toward Computer Science. I believe Jean-Claude Latombe is now the active, alive Kumagai Professor.

**Nilsson:** Now after 2000, after retiring, you also wrote something about the Japanese entrepreneurial situation.

**Feigenbaum:** Okay, let me tell you. That carries on from the previous study. In 1996, when I finished that study of the Japanese software industry, and published it, there was… How do I put it? The result of the study in '93 and '94, which was later written up in '96, was to uncover a mystery that needed to be looked at. The mystery was why is the startup situation for software companies in Japan so bad? Where's their Oracle? Where's their Microsoft? Where's there anything? Have you ever heard-- what's your favorite piece of Japanese software on your MAC? Nothing. Right? Or on your PC? Nothing. So there's a problem there. When I looked around in the Japanese software industry for databases, for example, the databases were Oracle, Sybase or Informix, at that time. There were some inhouse things from Fujitsu and Hitachi, and so on, but as far as companies are concerned, there were no Japanese database companies. There was a Japanese word processing company named Just Systems, J-u-s-t, but it was getting clobbered by Microsoft, and Microsoft had just moved in to clobber the NEC operating system, which had been the operating system that had first been the one widely used on Japanese PCs. So that looked like a really open issue. What's going on here? There's a phenomenon that needs an explanation. In 1999 I went back to the Sanford Japan Center for a month, and under the tutelage of Professor Imai, who was Director of the center, and a major economist in Japan, and funding from them. I looked into that question more deeply. Turned up a lot of reasons why the entrepreneurial situation in Japan is not good. Later wrote those up in a monograph. Not a long monograph. The monograph is available on the internet. If you're way out in the future you may have to go to the Internet Archive to look this up 50 years from now, but if you search Google under “Stanford Japan Center Feigenbaum, Brunner” -- Brunner was my coauthor, a student of mine -- you'll find a manuscript for a book called “The Japanese Entrepreneur: Making the Desert Bloom”. It's six chapters of diagnosis about what the problems are with Japanese entrepreneurism in high tech areas, and one chapter of therapy that involves a concept called “Special Economic Zones”, which was particularly popular at that time in 2002, because the Chinese had been using it successfully. That book was translated in a big hurry, because there was a debate in the Japanese Legislature, the Diet, on Special Economic Zones. This book was translated in a big hurry, published in Japanese only, in December of 2002, so that it could play a role in this debate on Special Economic Zones. An English language version of this book has never been published, because it's available, it has been available from day one, on the Stanford Japan Center web site.

**Nilsson:**Did the Japanese follow up on any of these therapy suggestions?

**Feigenbaum:** No, they didn't. There was a small flurry of interest that resulted in my going to Japan a couple times to give seminar series and lecture series, but in effect it was like throwing a pebble into the pond. It makes a small wave that ripples out to the edge, and then it dies, and that's it. The total effect of it was close to zero. I believe the reason has to do with the fact that the… Everyone agreed that the diagnoses were correct. The therapies were, in my later opinion, counter-cultural. That is, the Japanese didn't want to adopt those particular kinds of therapies. I kept telling them, "You don't have to. You can remake these therapies any way you want that fits the need. Just look at the diagnoses and remake the therapies." But no one bothers to remake the therapies. The government has done about all that it can do right now in loosening up things to foster entrepreneurism, and things are better, considerably, now. I'm about to get involved with a project in which we're going to do an update of this book with Japanese help.

END OF TAPE 3

**Nilsson:** Ed, let’s conclude by looking back and trying to summarize a bit about what you’ve learned through all of these research projects, and work that you’ve done in artificial intelligence. And also take the opportunity to look a bit at the future as to where you think AI is, where it is going. I had asked some questions, for example, questions about what do you think how well have we done on trying to solve the induction problem, and where are we toward getting the ultra-intelligent machine. That gave you pause to think about some things, and you did. What are your thoughts?

**Feigenbaum:** Well, Nils, I liked the questions a lot, and I knew they were really important questions. I always do better when I write out the answer to questions. First of all, I am not as wordy as I am otherwise. And secondly, I say things better. So I actually wrote out some answers. This isn't a lecture, so I do not want to really read from what I wrote, but I hope the audience excuses me if I glance down at my notes from time to time.

The first question that Nils asked me is: am I satisfied with what I and my team -- by which I think he means the Heuristic Programming Project -- have done on the induction problem through many sandboxes, which is what we were referring to as experimental environments. I made a reference a while back to the book "Computers and Thought." This is a book that appeared in 1963. Except for the proceedings of the British conference on Mechanization of Thought Processes, it was the first book that AI students could get their hands on to read papers in our field. And here is what I wrote in the introduction to this book. Page 8. "Inductive inference. Artificial intelligence currently is strong on deductive inference, weak on inductive inference." By deductive inference I really meant more like the puzzle-solving end of AI. "Yet in the melting pot of everyday of intelligence, induction is certainly the more significant ingredient. One way of looking at the problem is that we need programs that will in some sense induce "internally stored models" (quote unquote) of external environments, models from which the program can make valid and useful predictions of future environmental states. Looked at in another way, this is the problem of hypothesis formation by machine." So you could see that that phrasing was on my mind. "It is the general pattern recognition problem." I am going to come to it later why I do not believe that anymore. "And today", this is 1962 when I wrote this, "we know very little about this crucial problem." So that's what I said that. Now, there were a few people working on pattern recognition at the time. There is a pair of them, whose paper appears actually in "Computers and Thought", Oliver Selfridge primarily, and Jerry Deneen had done Pandemonium. Pandemonium was an induction program of a very simple variety, much more appreciated today, actually, than it was at the time. Since then, a good deal of AI research has followed that particular path, that is, a path of induction from date, induction from examples. One of the more significant early works that impressed me was the environment modeling that was done inside the mind of Shakey the Robot at SRI. The intention of the designers was that Shakey would have a model of the environment. And although Nils says that it didn't have a fully dynamic model of the environment, it did have enough to plan its activities and maneuver through that environment. That was 1972, I think. Now our group, the Heuristic Programming Project, focused on inducing symbolic models -- symbolic models, that's going to be important later on -- of particular domains of discourse. Now, when I say a domain of discourse, why all that funny language? Because I don't have any other way to say it. It's an area of work that you are doing, like if you are doing work in medicine it is some part of medicine. If you are doing civil engineering, it is some part of civil engineering. It's some domain of discourse. If we're talking about the activities of the Computer History Museum, it's about being a museum. It's about history.

**Nilsson:** Collections, and all of that.

**Feigenbaum:** Yeah. On these, of course, we had some quite significant results, and a lot of them we have discussed in this interview. My favorites happen to be Meta-DENDRAL first of all, and HASP, but I'll mention why. Meta-DENDRALl, with the follow-on done by Tom Mitchell for his thesis, called "Version Spaces", represented a milestone, for me. A milestone on the path that I had intended to be on all along, namely the path toward the future of machines that would produce scientific theories for mankind. And I still believe in that path. I still would like to be on that path. If I had another career, I would still be on that path. So Meta-DENDRAL was an exciting event, because we were actually inferring the rules of nature from massive amounts of data in mass spectometry.

**Nilsson:** In a symbolic form.

**Feigenbaum:** In a symbolic form. Yes. Now HASP, the sonar signal understanding program, was important for me as a landmark of technical achievement in applied AI because essentially the task was so difficult. I wish I had here an acoustic rendition of those ocean noises and play them for you and see if you can figure out if there's an enemy submarine in there. And also because the so-called blackboard framework that Nils and I have talked about here, the blackboard framework for induction was greatly elaborated, although it wasn't invented for this purpose. It had been invented and then not used at Carnegie Mellon. It was greatly elaborated and then sent out into the computer world as a software package for wide use, the package called "AGE." However, in Nils' question, there was an implication that induction was the focus of the work we were doing. I just want to mention that the Heuristic Programming Project did a lot of things that were not on that path. For example, the user interface work that was so important in MYCIN, where MYCIN was able to interact with a doctor in more or less natural medical English. Or the explanation systems that went along with MYCIN. Or the teaching systems that Bill Clancy was able to convert MYCIN into a teacher of diagnosis of blood infections and therapy of blood infections. Experiment planning work in MOLGEN is much more along the lines of planning work in AI, and not induction. There was no induction part of that. Now, in addition to us, there was a small effort of modeling of scientific behavior done at Carnegie Mellon University under Herb Simon that involved several students of Herb Simon's, among them Pat Langley. Those models were made, were constructed to be, deliberately simple, because that is actually the way Herb viewed those early scientific discoveries of Kepler and some other early scientists -- as being very simple, human information processing involved in those. This work of Simon and his students didn't get very far, because the AI community, as a whole, viewed the models as not simple, but simplistic, and therefore not worthy of further pursuit. So that particular line of attack at the question of scientific discovery was abandoned. But I am remembering just sitting here right now, remembering the last time that I... I used to visit Herb Simon from time to time in Pittsburgh, as he got older, just to make sure that I was in touch with Herb. I remember sitting in his living room and he was reading a book on the life of one of the,…I can't remember which of the great early scientists he was reading. He had written a paper on the thought processes that he believed the scientist was using, and had submitted that for publication to some journal, and had gotten a really critical review back of that article from someone who had just finished a book on the history of this scientist and said, "You are wrong. And here's the evidence that you are wrong." So Herb went back to read this other material to see what he was wrong at. And even in his older years, he was still concerned about this problem, the problem of scientific discovery.

In the 1980s and 1990s, much attention was focused on learning. By learning I really mean computational learning, machine learning. In the earliest days of AI, we were calling this kind of work machine learning. For example, when Arthur Samuel was doing an early version of this work for his checker playing program, it was called machine learning. Of course, that also was an inductive task. Lots of behavior resulting in the tuning of the weights in a certain formula that attributed quality to positions, both in checkers and chess. So all of machine learning is basically machine induction. Some people in these early days, like for example 1983-85, weren't quite happy with calling it machine learning. They were calling this kind of work knowledge discovery. And I really liked that term, because I had already made my conclusion back in 1968, that's the "knowledge is power" hypothesis, which later I rephrased the knowledge principle that "in the knowledge lies the power". What you were discovering… Excuse me. What you were learning that would assist higher levels of performance at a later time, which is what we mean by effective learning, was really knowledge. You weren't learning processes. You were learning more knowledge. You were learning what was relevant, and what was a good rule, and what was a good weight, and things like that. So I really liked the term "knowledge discovery." In 1972, as usual -- sorry, not as usual -- as became common over the years, every so often DARPA needed a report to make the case that AI research was valuable. Fortunately for AI, DARPA support has continued from 1963 right up to the present day, basically undiminished. Not diminished. It's really been great. But they do require these reports from time to time saying that we are doing cutting edge work. And in 1972, Newell called me up and said, "Ed, it's your turn." Because Newell had been doing that kind of thing before that. So I wrote a report. It's in my archive, if anyone wants to look at the whole report. It's not a published report. But for that report, I tried to rename the "machine learning" term to be "knowledge acquisition", because I had already come to the conclusion that learning involved -- it's like Meta-DENDRAL -- learning involved the acquisition of new knowledge. That didn't stick. It became used as a term to describe the human process that was involved in knowledge engineering. So when a human interviewed another expert and got the knowledge out, what Donald Mickey would call a mining process, became known as "knowledge acquisition". Carly Scott and others even wrote a whole book on that subject of how you do that. In the end, the term "machine learning" is the one that stuck. The Machine Learning Society, the Machine Learning Journal. It's the term that stuck. Now this was induction research, the '80s and '90s work on machine learning. It's truly massive, in terms of number of people involved. It was induction research on a scale that hadn't ever been seen before in AI. But the inductions were largely statistical inductions, not the induction of symbolic models.

**Nilsson:** How about inductive logic programming? Was that an exception to that, then?

**Feigenbaum:** No. Inductive logic programming is a good example of the induction of symbolic models. Yeah. Actually, I had completely forgotten about it, I don't know why. Maybe it just didn't stick because it didn't have many applications. But you're absolutely right. Inductive logic programming is a good example. Now it's really a disappointment to me... In fact, I heard the first, big talk on this by Tom Mitchell, one of our former students who had done such a wonderful job on cleaning up Meta-DENDRAL for us. And my response in the question-and-answer period was, "Tom, is that all there is?" Come on. There's got to be more than that. That's not what Einstein is doing. He's not doing statistical learning on the phenomena of nature. He's thinking through things about nature. And that's what all great scientists are doing. They're thinking about structures, and in symbolic expressions. The statistical models just led to what I consider to be weak theories of various phenomena based on statistical aggregation and correlation. Now, that's not to say that they're not useful. In fact, we all learned a lot about the learning based on aggregation from Nils Nilsson's book, which was very early in AI, the book on learning machines. But there has to be more than that, in my view. So the answer to Nils' question is that our group, the HPP, it did pathbreaking work in a large, unexplored wilderness, which is the wilderness of all the great theories we can have. Particularly in my view the beautiful wilderness of scientific theories that we could have. But that most of that beautiful wilderness today remains largely unexplored. So Nils, if you are asking, am I happy with where we have gotten in induction research, the answer is absolutely not. I am very glad we did a few key steps that people will remember. But I'm not happy.

**Nilsson:** We aren't there yet, but you got a good start.

**Feigenbaum:** I want to comment that in this "Computers and Thought" thing I talked about the general pattern recognition problem. I don't believe there is a general pattern recognition problem. Let me put it another way. I don't believe that there is a general pattern recognition problem. I believe that pattern recognition, as most of human reasoning, is domain specific. That I believe in domain specific cognitive acts that are surrounded by knowledge of the domain, and that includes acts of inductive behavior. So I don’t really put much hope in general anything in AI. In that sense I have very much aligned with Marvin Minsky’s view of… I don’t know what. Marvin called it a “society of mind”. Some people have called it mind, a kitchen sink model of mind. But I’m very much oriented toward a knowledge-based model of mind.

**Nilsson:** The other question, I do not know how much you want to go into this, Ed, is what more needs to be done? What are some of the big problems that need to be solved in AI? Where are we in AI these days?

**Feigenbaum:** Once again, I’d like to answer that by cutting through a lot of detail and relying on a very simple model of what I think is going on. The model was a consequence of the work we were doing, particularly during the MYCIN days of the HPP, where it led specifically to a diagram that was first drawn up on the blackboard by Randy Davis -- I remember exactly the moment -- which had a box in it called the knowledge base, and another box called the inference engine. And then there was a user that these two would interact with each other. The inference engine was doing inferences over the space of expressions in the knowledge base, and the user was being fed results, or monitoring this interaction. What we found, long after Randy drew the first diagram of it, was that these are not boxes of equal size. The box called the “knowledge base” was enormous. Everything, almost everything, was in the knowledge base. And almost nothing was in the inference engine, except a little guy there called Aristotle. A little modus ponens. A little backward chaining, forward chaining. A little bit of that goes a long way. And not much of anything else is needed. Now, of course, you could argue with that and say, well, the blackboard model is a lot more complicated than that. Yes, it is. True. But it is really a combination of a little bit of Aristotle going up and a little bit of Aristotle going down, and some very intricate knowledge representation work that’s done at the different inferential levels. So what does that amount to? What’s in that big box, the knowledge base? How’re we going to get that box to be really big? And when I say really big, I do not mean the 400 or 800 rules of MYCIN, or the 400 rules of PUFF, or the 5,000 or 50,000 rules of the DEC knowledge base for constructing computers. I mean something much more like Jack Meyer’s 500,000 piece of knowledge, knowledge base for internal medicine. Or maybe 50 million. I don’t know exactly what these numbers are; I’m sure they are in the hundreds of millions to billions of things that people know. But how’re we going to get that? Well, I think that the only way to get there is the way human culture has gotten there. There are a number of significant ways in which we differ from animals, but one way is that we transmit our knowledge via cultural mechanisms called texts. Used to be manuscripts. Then it was printed texts. Now it’s electronic text. But we put our young people through a lot of reading to absorb the knowledge of our culture. To boil it down to some bumper sticker language, you don’t go out and experience chemistry. You go out and you study chemistry. We need to have a way in which computers can read books on chemistry and learn chemistry. Or read books on physics and learn physics. Or biology. Or whatever. We just don’t do that today. Everything is hand crafted. Everything is knowledge engineering. The human mind transforms it into symbolic expressions. We will be forever doing that unless we can find out how to read from text. Reading from text in general is a hard problem, because it involves all of common sense knowledge. But reading from text in structured scientific domains, I don’t think is as hard a problem. I think that’s a critical problem that needs to be solved.

I think in the little box called inference methods, we have a few that have not gotten explored enough in my view. Again, if I were to have another career, I’d do way more exploration of one in particular. Well, of course, I’ve already said that I thought the blackboard model was superb, and it hasn’t gotten enough exploration, either. But there’s another model that we used in DENDRAL, which goes back to the early days of AI, called generate-and-test. On the one hand it sounds too simple. Namely, that you have a combinatorial generator that can generate candidate solutions, and that the way you get rid of the very large numbers, the exponentially large numbers, of candidate solutions that you have, is by using heuristics, which are either general purpose heuristics, or most likely, almost surely, there will be domain specific heuristics. People have shied away from using generate-and-test, because they talk about finding needles in haystacks. And they see immense haystacks. The search for them -- they'll never find needles in those haystacks. Well, first of all, if you use heuristics right, you might be able to prune away most of the haystack, which is what we kind of demonstrated in dendral. But there may also be not just needles in the haystack, there may be diamonds in the haystack, which are things that because of the combinatorics of generate-and-test, people have never thought of before. It's just too deep and too complicated. It reminds me of that game 2 of Kasparov versus Deep Blue, where Deep Blue was doing mostly a generate-and-test with some chess heuristics to prune the space. But it was able to do tens or hundreds of millions of paths in that search. And it happened to find a diamond. It played that move, and Kasparov was completely spooked. He was spooked for the rest of the match. He gave an interview in which he… if you go back and look at the interview, there is one point in which he said, "It was like looking into the mind of God." Now I don't think Kasparov is a believer in God. But he was using that metaphor as a way to capture something. It was a diamond. It was a beautiful piece of thinking.

Now, I think we also have to couple people into this search, to assist this search, partly because the knowledge to prune this big space -- we just simply won't have it in the computer for a long time to come. We're going to have to let people's powers of association help to prune the space. Doug Lenat did this when he was doing his work on Eurisko. He was able to find basically the diamonds that cracked open a major national game that he was involved in playing, where he won the championship two years in a row because he was steering what amounts to a generate-and-test thing. Every night he would come in, he would use the machines over at Xerox PARC, and he would steer the search into certain directions. He found niches in the game, which were diamonds in that space.

The last thing I wanted to mention is that we have to keep in mind something you don't hear much about in AI. You hear about it everywhere else in computer science, but not in AI. From the time that some of these great early works were being done -- whether they were Newell and Simon's works, or the works of our group, or the SRI work, or others -- we have advanced something like, depending on what is at your command these days, somewhere between 10 to the $6^{th}$ and 10 to the $8^{th}$ -- six or eight orders of magnitude -- more computing to bring to bear on the problem than we had before. Now, we don't know if that's significant. The reason we don't know is that 10 to the $6^{th}$ or 10 to the $8^{th}$ does not sound like a lot in spaces that tend to be 10 to the $100^{th}$ big. But, on the other hand, getting 10 to the $6^{th}$ up on moves in chess beat the world's chess champion, and we didn't think that was possible. We just don't know.

Finally, I want to say a few words about the blackboard framework before we go on to our next question. If you look at the structure of the blackboard, the blackboard is organized into levels of abstraction. At the bottom is the data itself. At the top is the grand overview, the overriding theory. At every level you have more detailed expressions of the abstraction at that level. Well, that's almost a description of scientific theory. Scientific theory is rooted in the evidence that we measure from nature, but at every level we have abstractions. We can view it at the particle level. We can view it at the atomic level. We can view it at the molecular level. We can view it at the more engineering levels.

**Nilsson:** As a matter of fact, going down from one level to the one below is often called reductionism, right?

**Feigenbaum:** Okay. You got it. I think the blackboard model has a tremendous amount to say about modeling of scientific inference, and we have yet to discover that. The blackboard model was virtually abandoned in the early 1980s, and hasn't really been much used. It was used for a military problem in the late '80s, early '90s. I tried to get it used a lot more in military contexts when I was at the Air Force. But I think we ought to look more at that.

**Nilsson:** Maybe we could conclude the interview, Ed, by looking at the AI culture as it exists today. You had mentioned how important culture is. Is the AI culture today set up? Is it ready to look at and try to achieve the ultra- intelligence machine? Are people focused on that? Or have the AI culture somehow become a little more interested in things other than what I have called in the past, the "eye on the prize"? Where do we stand with regard to AI research today?

**Feigenbaum:** I am personally one of the world's greatest champions of the prize, and so was very, very happy when you and McCarthy started to give these seminars and lectures, and when you published your paper on the "eye on the prize". Because, guess what? No one else is talking that way. Now, Minsky is left, and he still talks that way. Newell and Simon are dead. I don't know of anyone else who goes around talking about that great prize. Now what is the prize? Well, it comes in two varieties, both of which would have made Newell and Simon very happy. One is the ultra-intelligent computer, the phrase of which was first used by I. J. Good, a statistician. Incidentally, I. J. Good had worked with Turing during the Bletchley Park days of decoding the Ultra and other codes in World War II. I. J. Good coined the term "ultra intelligent computer." And that just means AI doing its thing to the utmost. When we finish our job, we will have invented the ultra intelligent computer. That's what we're trying to do. Whether it is a 100 years from now, or 200 years from now. But that is what we are trying to do.

The other flavor it comes in is a very complete model -- I say very complete; that doesn't mean we'll ever get everything; I hope we do, but I'm not predicting it -- of an information processing model of how the human mind works. And when I say the human mind, I do not mean the human brain. I mean the symbolic processing entity. What Newell and Simon called "elementary information processes". The elementary manipulation of symbolic expressions, symbols, lists, and so on. So we have both views. Now if you talk to the people like some of my... Nils and I have a very distinguished colleague at Stanford. Younger, way younger than we are. Talking about this, you get the response: "Sorry. The vision of the founders" -- that means people like me and McCarthy, Newell and Simon, Nils, Charlie Rosen, people like that – "Sorry, that's what got us into trouble". Now, first of all, there is a view that somehow we got into trouble. In fact, that even reflects itself in a recent conference, the AAAI Conference, of "what went wrong and why", as if something went wrong. But the feeling is, "Look, we're just part of computer science. Don't give me this AI stuff. We're just part of computer science, and we make progress one paper at a time". Now, I am not exaggerating. I really mean that. They really believe that "one paper at a time" thing. "I got to get six per year. I'm not after the big prize. There's no such things as the big prize". Well, of course, we know that there really is. And when major things are accomplished the world cheers, like the Stanford AI labs' driverless car that went 200 kilometers in the desert without a driver. That did win a giant prize, and the world cheered. In my view, the thing that we call AI, or that maybe is better called computational intelligence -- computers doing intelligent things -- is the manifest destiny of computer science. That is the end of the road. If you're going to use, which I love to do, the Lewis and Clark metaphor of starting out at St. Louis and slogging your way through the Missouri valley, and up the mountains to wonderful places that are difficult to achieve, the goal is the ocean. AI is the goal. You can stop along the way. You can have a wonderful life in Salt Lake City doing whatever, graphics, or you can stop in Denver and do databases. But the goal is AI.

Now AI has a tendency to spin off little things that sit in valleys. Nils' metaphor is valleys. For example, if you look in "Computers and Thought", Oliver Selfridge had a key paper. He and Deneen had a key paper on distinguishing X's from O's. That was the birth of optical character recognition. By 1964, that is, one year after "Computers and Thought" came out, optical character recognition was its own named field. And that has happened time and time and time again in AI. Computer vision. Expert systems. Computational linguistics. Carnegie Mellon, for example, which has a School of Computer Science and therefore a highly flexible approach to setting up departments, has their own Department of Machine Learning -- as if that's different from AI. They don't have a Department of AI, they have a Department of Machine Learning. So it's happened again and again and again.

Next thing I want to say is that I do believe that the most effective way to make progress on the ultra-intelligent computer, which happens to be my engineering slant in this whole game -- used to be the psychology slant, but the engineering slant -- is to study the way people behave. Not necessarily because one wants to publish papers in Psych Review, or any of the other psychology journals, but because we have excellent examples of intelligence walking around us all the time. They are easy to study. There are billions of them to study. We as scientists, in my view -- at least for myself, and I am pretty sure it holds for most of AI -- are better at discovering things than we are at inventing things. Inventing is hard. Finding mechanisms by discovery, looking at complex human behavior, and modeling that behavior, is easier. Now, that's not to say that we should be happy with what we find. Why? Because evolution may not have put together the best thinker. Evolution may have put together the best hunter. Maybe the best eyes. Maybe the best hearing, we have inherited from animals. But not necessarily the best thinking. So as soon as we uncover principles from human intelligence, we can improve upon those principles.

**Nilsson:** So perhaps on that note we should conclude, and thank Ed for his valuable recounting of some of the things that he has done in his life and his ideas about where we might go in the future. I hope very much that people have an opportunity to look at this and reflect on it.

**Feigenbaum:** Thank you, Nils.

END OF INTERVIEW