A. M. Turing Award Oral History Interview with
Richard ("Dick") Edwin Stearns

by Dan Rosenkrantz

Albany, New York

November 15, 2017

Rosenkrantz: Hello, I'm Dan Rosenkrantz. It's November 15^{th}, 2017. I'm going to interview Dick Stearns, who won the Turing Award in 1993 with Juris Hartmanis in recognition of their seminal paper which established the foundation of the field of computational complexity theory.

To start, can you please tell us where you were born?

Stearns: Yes. I was born in Caldwell, New Jersey on July 5^{th}, 1936. The place I was born is just a few houses down from where President Grover Cleveland was born.

Rosenkrantz: Can you tell us about your family background?

Stearns: Yes. My father was born in Matawan, New Jersey, was brought up in Caldwell, New Jersey. His father graduated from the University of Pennsylvania. He eventually went into the ministry and became a minister of the First Presbyterian Church in Caldwell. He married the girl next door named Mary Jeter. My father's name is Edwin I. Stearns.

My mother was born in Newark, also raised in Caldwell. Her father was born in Iowa, studied engineering at the University of Iowa, came east to work for Edison Electric Company, which eventually became GE, and then RCA. He married Clara Kemper, who was born in Elizabeth, New Jersey, and lived in New York City. And my mother's name was Winifred Scales.

I have a younger brother Robert who is retired from the Army Corps of Engineers, and a sister Dinny who … her last job was at Williams College where she was director of technology.

Rosenkrantz: Okay. Growing up, when did you first become interested in mathematics and science?

Stearns: Well, I should say more about my father. My father studied chemistry at Lafayette College, then got a master's degree in chemical engineering at RPI, and he went to work for American Cyanamid Company in Bound Brook, New Jersey. He soon discovered that if he wanted to get further along, he should get a PhD. So the same year that I started kindergarten, he started studying for a

PhD at Rutgers. So it's always been almost assumed that a PhD is your academic goal. My mother was a chemistry major at Swarthmore College. She was the only woman chemistry major there. She learned a lot of mathematics, which she was able to discuss mathematics with me as I was growing up. My father was an amateur ornithologist, well-known in New Jersey.

So there was a lot of exposure to science within my family as I grew up through grammar school and middle school in North Plainfield, New Jersey. There were several books in our home library on mathematics. One was Courant and Robbins' *What is Mathematics?* and a book by Gamow on *One Two Three . . . Infinity*. At some point, they also acquired a little book by Williams called *The Compleat Strategyst*, where they discussed two-person zero-sum games.

I remember a few instances with my mother regarding mathematics. One instance, there was a fad in the school at one point where you had this… they called a "Sixteen Puzzle," where you slide pieces around on a four-by-four grid. There was a booklet that listed different things you could do, like the numbers in sequence or the odd numbers then all the even numbers. Then the last one was called "Impossible." There were some people who claimed they had gotten the "Impossible." The only real way of getting that is cheating, take all the pieces out and put them back together in a different order. Well, my mother explained to me that that was impossible because you could only do even permutations. So at that point, I learned something about permutations.

Another incident that comes to mind is I was interested in probabilities, having rolled dice and moved pieces around a game board. So I asked my mother about probability and she had a book which explained the elementary principles of probability. That was my background through middle school.

Then we moved from North Plainfield to Plainfield, because they wanted a little bigger house since my sister had come along, and I went to Plainfield High School. My education there was great, particularly in mathematics. I remember the algebra teacher I had as a freshman, she always insisted that you always explain what you're doing in terms of operations. So, if you were solving an equation for example, you're supposed to say, "Well, now I'm subtracting something from both sides of the equation," and that sort of discipline instilled some appreciation.

Then after that, I took advanced math for three years under a teacher named Art Smith. He covered all the topics that one would… trigonometry, geometry, a little bit of calculus. So I was well prepared. Not only that, but found a great interest in math, but I was also well prepared for when I got to college.

Rosenkrantz: Was math your favorite subject in high school and were there any subjects that you hated?

Stearns: Math was my favorite subject. My only thing I really didn't like was spelling. There was no course in spelling, but in English, they give a list of spelling words to learn, a hundred words and you're supposed to get 95 of them. I took that test and got close to, a little over 95 I think. I tried to take it again to get a higher score, but I got a worse score, so I had to take it a third time. So yeah, spelling was my least favorite.

Rosenkrantz: Was there any course you hated in college?

Stearns: Yes. German was my downfall. [laughs] I really struggled with that, *[0:10:00]* even had to repeat it a semester. When I took the second year, there was a book with scientific… chapters on science, and one of the things that was supposed to be on the test was to translate a passage from that book. Well, I spent hours and hours until I knew sort of the translation by heart.

Anyway, I got through the second year of German actually with B's, but it with a great struggle.

Rosenkrantz: After high school, why did you decide to attend Carleton College?

Stearns: Well, that's another instance where chance events occurred. I knew that I should go to a small liberal arts co-ed college, since I was very introverted and I thought that would be best for me. And, of course, my mother had gone to such a college, Swarthmore. Had nature taken its course, I probably would have gone there. But my father got transferred to Chicago, and my uncle had come over, as he often did, and they said, "Well, maybe you should think about something in the Midwest." My uncle suggested Carleton College, so I sent away for the information. Although the distance been Carleton in Northfield, Minnesota and Chicago was still quite a ways, I was attracted to it. And, after a great struggle, and I had never actually had a visit to Carleton, but something about it appealed to me and I decided to go there. So sight unseen, I picked Carleton and went there and it was a great experience for me.

Rosenkrantz: Can you describe your experience at Carleton?

Stearns: Yes. A number of things come to mind. First thing, during my freshman year, I got acquainted with another student, Roger Kirchner, who was also very interested in math and a top-notch student. We became best friends. We had a lifetime friendship, which turned out to be important in several ways later on.

When I got to the end of my sophomore year, I had to pick a major. Well, because of the courses I had taken, I had two options, one to be a chemistry major and one to be a math major. I was a little bit concerned whether as a mathematician I could make a living. I went to the head of the department and asked him how I could make a living, and he said yeah, he knew somebody who'd got a job and basically said yes, I could. Of course, math was my first love, so I signed up to be a math major.

That summer after my sophomore year, I wanted to take a book home from the math library over the summer to read. I picked out von Neumann and Morgenstern's books on *Theory of Games and Economic Behavior*. I was really impressed, by the way, starting from very scratch, they'd built up a model of games. I thought that's kind of the way math should be carried out. You think if you carefully work out your model, hopefully reflecting as much as you can of the real world and then work from there. So I felt very enlightened by that book.

Well, they had an arrangement where, your senior year, you could write a thesis, or a paper, do something of that ilk, and if you succeeded in doing that, you would graduate "with honors". Kind of an alternative was doing well on your final oral exam and you would graduate "with distinction". Well, I thought graduating with honors and having written something would be better, so I did something involving graph theory and Arrow's paradox.

Arrow's paradox says, given preferences of voters, there is no fair way of coming to a group decision. I analyzed and said, "Well, what if we have just three people and their preferences amongst three candidates was equally likely? What is the probability that there was an outcome agreeable to all the three people?" Well, the math department – probably the chairman, Ken May – said, well, I should write that up and submit it to the *Math Monthly*. So I did that and that became my first mathematical publication.

Rosenkrantz: Did any of your college friends also pursue careers in math or computer science?

Stearns: Yes. My friend Roger went to Harvard to get a PhD. There were two other very good math students in the class. One of them also went on to Harvard. The other one went on to medicine. The number of math majors at that time was something like 11 or 12. The others did not go for PhDs. I think some of them went into math education.

Rosenkrantz: Why did you choose Princeton for your graduate study?

Stearns: Well, that was due to another chance event. The chairman of the department at Carleton, Ken May, knew John Kemeny, who was at Dartmouth. He wrote a famous textbook and was instrumental in setting up the computer system at… Well, he basically invented the language BASIC. But anyway, he came to visit Carleton and he described the math department at Princeton in such glowing terms that persuaded me that that's where I ought to go.

Rosenkrantz: Who was your PhD advisor at Princeton and what was your thesis topic?

Stearns: Well, *[0:20:00]* my thesis advisor was Harold Kuhn, who did a very famous piece of work in game theory on games with perfect recall. Now at Princeton, of course Princeton happened to be the place where game theory… the origin of game theory from von Neumann and Morgenstern… Of course, von Neumann by that time had passed away, but Morgenstern was there in the economics department. Just in the Princeton bookstore, there were a number proceedings on game theory. I was interested, so I read all that stuff. That made me want to study game theory for a thesis, so I asked Kuhn to be my advisor and he agreed.

But he did not suggest a topic. He said, "Well, you ought to do something that you're… find out what you're good at and do that." He gave me good advice on writing a thesis and what style to use in expressing theorems and to include a lot of English in your statements rather than math. And he introduced me to game theorists as they came through.

Well, another chance event. Bob Aumann was visiting Princeton in something in connection with Morgenstern. He had an office in the… There was a consulting firm called Mathematica, and they gave him an office. His office was there. It's a little bit fuzzy on exactly how he was associated with Princeton. Anyway, he had some suggestions as to what to work on. I picked one of the topics he suggested, which was *Three Person Cooperative Games without Side Payments*, the objective being to find all of von Neumann–Morgenstern, solutions to those. After he got me started, he had made sure I understood the first paper that he had written, and I was able to improve one of his proofs, which satisfied him that I understood the material.

So I went and worked on that problem. After I had solved it, I told him, "Well, now I have the answer." He said, "Alright. Well, explain it to me." We set up a time at the university where there was a blackboard and things, and I started explaining it to him. Well, I had only a few pages of notes on this thing, but after discussing it already for an hour, I wasn't done, so we scheduled a second hour. Well, about the end of that hour, I discovered there was one case I hadn't covered, so we met for the third hour and I fixed that case, I analyzed that case, and he was satisfied.

I think that was really an examination on my thesis, because although Kuhn was helpful particularly in some of the early chapters and spotted a couple spelling mistakes which had to be changed… In those days, you made your copies with carbon paper, so certain words you have to erase. I was making three carbon copies because I had to hand in two copies of the thesis, I wanted one for myself, and I wanted one for Bob Aumann. So I had to make all those spelling changes.

So yeah, I passed and then I passed the exam, the committee. Well, later on, when I was going over the thesis to write the paper on it, I discovered one page where the subscripts and superscripts were all missing. Because in those days, to make a subscript or a superscript, you had to go back and turn the roll on the typewriter up or down to make them. So I realized that maybe… [laughs] maybe nobody had read that far into the thesis. Of course, now having been in academia, that doesn't surprise me, because I certainly haven't read every, checked every line of every thesis I've ever been presented with.

So yeah, it was luck Aumann there to be a mentor. Of course, Aumann later got the Nobel Prize in Economics, so I can say I was mentored by a Nobel Laureate.

Rosenkrantz: How did you wind up going to General Electric and what was your job title when you joined?

Stearns: Well, there's another case where chance, a chance event altered the direction of my life. GE Research Lab, particularly under Dick Shuey, tried to get students to come and work over the summer with the hope that they would like it well enough so that when they got out, they would come back. So the GE recruiter went to Harvard. My friend Roger was looking for a summer job, so he interviewed with them, and he said, "By the way, you want to hire Dick Stearns as well." Without having sought a summer job, I was invited up for an interview and accepted the job.

Now the way it worked is that the different members of the staff there would make proposals to say, "Well, you ought to work on… Here's something you'll work on." One of the proposals was by Hartmanis. Now Hartmanis had written a paper called "The State Assignment Problem, number I" and he said, "Well, look, this was involving finite-state machines and partitions with a substitution property," and he said, "Well, I know you can have in effect the input partition is one way and the output partition in another way, and they fit together." So he said, "Look at this."

So I did look at that and I made different examples in order to learn more about the problem. I came up with the idea of partition pairs, which essentially is … the example he had given me was that. But I also came up with the idea that you could have a lattice called the MM-lattice, which would describe the set of all these pairs. So you're going beyond just describing the pairs themselves. That became the paper known as "State Assignment II." *[0:30:00]*

Well, that was before I'd started my thesis, so I had to get it at the end of the next year. Shuey called me on the phone and said, "Come back for another summer." I said, "Well, I think I'll be out." So they offered me a permanent job. I went up there on… It was Memorial Day in 1961. I remember it was a very hot day. I was still in the process of typing it up. I mean I got tired of typing, so I thought, "Well, let's go for this job." I was typing up while I was there. Well, I didn't realize it, but it turned out to be a little concern of the people having not actually finished the PhD, still having to write it up. But there was nothing in the job description that said, "Well, this is conditional on finishing your PhD." But anyway, I typed it up as planned and was able to… They had a fall graduation. I don't think they have that anymore, but I was able to get my thesis in in time still to graduate in 1961.

Of course, it was natural, having started on sequential machines, that we continued our work on that. We had several papers on it, which I won't go into, but finally we decided we would put the material together in a book. That was this book on the *Algebraic Structure Theory of Sequential Machines*. Well, it didn't turn out to be very practical, but it was quite enjoyable and rigorous, and we got many people who had read it said it was their first rigorous presentation of something having to do with this field. Anyway, it went out of print fairly soon, but that's how I got there and how we worked on sequential machines.

Rosenkrantz: Did you interview with other places for a permanent job? And why did you decide to go to GE?

Stearns: Well, first of all, I didn't think I wanted to go into academia. Somehow teaching, at least at that time, was not attractive. I did an interview at Bell Labs. They decided there was no place for me, even though I had two things under my belt, the state assignment work and the game theory work. Someone from Philco asked me to interview, which I did, but that was nothing of… I told them I wasn't interested. So that was the only job offer I had. I was really interested in going there, so that's what I did.

Rosenkrantz: After your work in sequential machines, you did your work which led to your Turing Award with Juris Hartmanis. How did you get started working on complexity?

Stearns: Well, Juris had read Shannon's work on information theory. He thought there must be a reason why things are hard. Maybe something like that might be the foundation for complexity. Well, and we thought that maybe trying to study the transition from finite-state machines to context-free languages to Turing machine maybe, kind of studying that transition, maybe that would be helpful. And, I don't know, we had some preliminary stuff on Turing machines. Not really complexity and I've forgotten what it was. Juris is speculating that we had just rediscovered some things about recursive sets or something. I mean I don't know exactly. We were thinking of complexity classes as the set of all things that you could do in a certain amount of time. In a sense, that's an easiness class, things that you can do in a time, but obviously we're not going to call our stuff "computational easiness."

Anyway, a paper came out by Yamada on real-time countable functions. And that was really the missing piece, because basically he was talking about now what would be called … yes, time-constructible functions. Somehow, we had the idea of what a complexity class was and now we saw that in effect a computer could, particularly a Turing machine could time, keep track of, basically keep track of time. Like when n^2 time-… when 1^2, 2^2, 3^2, and 4-… keep track of those time intervals. That's what we needed in order to do the diagonalization. That was the final breakthrough. At first, we used his model. His model was to put out an infinite sequence. So we developed a hierarchy based on infinite sequences.

Well, there was also at that time the language recognition problem, where… See, at that time there wasn't a standard even to work from. Well we could show that the same thing applied to the language recognition problem. That's how we put the thing together. Then of course we wanted to get that out to the computer science community, and many of the what we would call theoretical computer scientists attended a conference called Switching Circuit Theory and Logic Design. So 1964, we presented our results to that group. Of course our paper, our published paper came out subsequent to that. We also presented it at an IFIPS conference, which is another way of reaching people. Having aptly named it "On the Computational Complexity of Algorithms," a catchy title, people started working on it.

Rosenkrantz: Can you elaborate any more on how you went about developing your results?

Stearns: Well, I think I have kind of elaborated on it … *[0:40:00]* The hierarchy theorem says that "If we give you a little more time," and the one we worked out was saying, "Well, if you square the amount of time, then you can do something more that you can't do in the given time." Now as I say, that was all done with a diagonal proof using Yamada's idea of real-time countable in order to switch simulations from simulating one Turing machine to simulating another Turing machine.

Rosenkrantz: How did you hear about winning the Turing Award?

Stearns: I was at home and Dick Karp called and said, basically told me I had won the award. Now, as you may recall, they had submitted the nomination the year before and been encouraged to repeat, resubmit it. So it wasn't a total surprise. But he said the only condition was that I'd have to go to Phoenix and give a talk and to accept the award. Of course, I readily agreed to that.

The one thing he didn't tell me was that the ACM wanted to make the press release before this became generally known. So in addition to telling my family, trying to explain to them what this award was, I told the colleagues at the university and the university put out their press release before the ACM did.

Rosenkrantz: What was the impact of winning the award and how did your family react?

Stearns: Well, the family was pleased. The award carried a monetary award of $25,000, which was a big improvement from earlier days, but now it's worth a million dollars. Only one of the local TV stations picked up on the announcement and did anything about it. They sent somebody over to the university to take a brief picture of me and they wanted me to sit at a terminal. So I sat at the terminal. Of course, what are you going to do at terminals? So I sent a message to the faculty saying, “Well, they’re here taking my picture.” Anyway, that appeared briefly at the end of the nightly news. I remember the announcer said, “Well, it has something to do with algorithms.” *[laughs]*

So, well, I had done the work purely for the pleasure. Doing mathematics is a pleasure. That’s why I do it. Proving theorems is the ultimate rush for a mathematician, so I thought that would be satisfaction enough. But then once people, once the ACM said, “Well, he’s very smart,” so other people thought, “Oh, he must be smart.” I was raised to the rank of Distinguished Professor at the university. I got an alumni award from Carleton as an outstanding alumni. Luckily, my friend Roger Kirchner had gone back to Carleton as a teacher, so I just told him, “Well, nominate me,” and he did that for me.

So it continues. More and more people are impressed. As time goes on and the prize now grows, more and more people hear about this, it turns out that a lot of people know about the award. My wife is quite happy to explain to people that I won the award, so I don’t have to do that myself. There’s now what’s called a Heidelberg Laureate Forum where winners of the Turing Award and the top math prizes are invited to Germany every year and meet with students, they call young researchers who have applied and been accepted to that program. You go there and try and influence them. They want to have their picture taken with you and you feel like a rock star.

Another related incident. When last summer we went down to see the eclipse of the sun and we took a plane to Charlotte, where we went on to Clemson, my daughter decided to go with us and she had a different seat on the plane down there than we did. Well, she got talking to the woman next to her who was in computers and mentioned my name and computational complexity, and she was thrilled to have me on the plane and wanted to take a picture with me. She said, “This was better than having Taylor Swift on the plane.”

So they’re just rewards. Yeah, they’re just rewards that come with the recognition that I never thought of, just a bonus on top of having actually done the work.

Rosenkrantz: Can you tell us about your work with Fred Hennie?

Stearns: Sure. I mean the hierarchy theorem that Hartmanis and I developed said that, well, in order for us to prove that there’s something that can be done at one time and not another, that you would have to… that if you squared the one function, then it would have more time. Well, that didn’t really seem to be the best one could do. Maybe it would seem unlikely that things you could do in time n were the same as things you could do in time n^2 … or sorry, sorry… in time n^2 was the same as you could do with time n^3. I thought, yeah, there ought to be something better.

Well, in our proof, we simulated… you had to simulate Turing machines. And of course the Turing machine we constructed to do things in the one time that you couldn't do in the other had to have a limited number of tapes, but it had to be prepared to do things that Turing machines with more tapes could do. In order to do that, we were putting all the information onto a single tape from the machine we were simulating. That led to an n^2, because with each move you would have to… Well, there were actually two ways you could do it. One, after each *[0:50:00]* step of the Turing machine you were simulating, you could readjust the tape copies of the tapes on the one tape so that the places where the heads were lined up on the head-square of the tape, of the single tape.

The other method is that you would mark the location on the tape where the head was, so after each move you'd simply change the marker where the heads were. Well, of course with many tapes, simulating many tapes, those markers would get farther and farther apart. That's where they require n^2. The other method of moving the tapes, moving all the tapes, that was also n^2, because the amount of information on the tapes… each time you made a move, the amount of information you put on the tape might grow. So we had two methods of doing the simulation.

I thought to myself, "Why not… Maybe there's something in between." What does that mean? That means you readjust the tapes but you don't readjust the whole tapes. You just take a portion of the information and shift that back into place. I said, "Well, let's break the tape into zones of square size. One zone of 2^2 and 3^2 and 4^2. And …go… clean up the information in one zone whenever necessary."

Well, cleaning up meant shifting a lot of information at once. So I had the idea of the second tape, so I could take a block of information and shift that on the tape. Anyway, that worked out to be *f* to the three-halves power over *g* would be enough to guarantee you can do more. That was an improvement over n^2. I thought, "Well, maybe we can do better than that." So I said, "Well, maybe we can have the zones be of n^3 size, and then break those up in n^2 size, and we would work, use the n^2 method to clear up the zones of cubic size."

Anyway, that worked out to be … not three-halves but four-thirds, and it became apparent then that this generalization, you could well say, "In addition to that, have zones of a power to 5." So we said, "Well, why not use as many zones *[laughs]* as you can on the limit." Now the question is "What on earth does that mean time-wise?"

Well, Fred Hennie had been hired by Dick Shuey for the summer and he had an office next to mine, and he was clever at counting things. I went into his office about the end of the day, saying, "Look, I have no idea. Here's how I want to do it, but I have no idea how to… what the time this is taking." So he went and took that home with him at night and he came back the next day, then said, "Well, see, your method takes (log n)^2, but here's how it should be done." He had worked out the zone should be exponential in size. He supplied that piece in a very clever… with something I never would have… don't think I would have thought of.

He was able to finish off what I had started. Our work was really sequential. We never sat down together and said, "Let's solve this problem," but operating sequentially, we came up with the solution, which enabled now the present form of the hierarchy theorem, which says that f log f over g was enough to give you more time. That meant that in particular something you do in say n^2 could add a little bit n^2, you know, n to the 2 plus epsilon, that's already enough to give you more. So every… that kind of smoothed things out and made it into a … changed an ugly theorem into a pretty theorem.

Rosenkrantz: Your work on space complexity with Hartmanis and Phil Lewis has been influential. How did that work come about?

Stearns: Well, Phil Lewis was pretty insistent that we should look at space as well. At first I kind of had a feeling it was trivial. But then we came up with an important idea. That is the idea that we would put the input onto a separate tape from the work and we would only count the space on the work tape.

So, after we decided not to count the space on the input tape, then all kinds of things opened up, because particularly then you could talk about space which was less than n. Particularly you could talk about log n space, which has proven to be a very useful concept. One of the things we proved was that given log n^2 space, one could decide whether a string belonged to a given content-free grammar. The method we did that is that we in a sense guess… not guess but try segments of the tape, which was roughly one- to two-thirds the size of the input, and analyze that segment of the input, then, if that turned out to originate from a nonterminal, then analyze the rest of it. Anyway, that proof became very important because Savitch picked up on that idea and used it to show that nondeterministic polynomial time… space… that polynomial space, deterministic polynomial space and nondeterministic polynomial space were the same.

We also did some stuff below log n and down to log log n. The only strange thing about the classes below log log n is that somehow you need the assistance of what's actually written on the tape in order to do your analysis.

Rosenkrantz: You developed an algorithm to determine whether a deterministic pushdown machine recognizes a regular language. What led to that result?

Stearns: Well, when we were doing our space complexity work, *[1:00:00]* I wanted to show that recognizing that a … that a pushdown machine, if it was non-regular, would have to count, and therefore log *n* space would be required.

The proof went like this: If you do a certain thing, the pushdown stack would get too much stuff on it that we would then have to count. In a sense, it was a constructive proof. The proof was finally say, "Well, then if it's not regular, then this machine has to count."

Anyway, when I went to the IFIPS conference, Seymour Ginsburg was there and was asking us what we knew. He said, "Well, do you know, can you test a deterministic pushdown machine to see if it's recognizing a regular set?" I instantly knew that this was an implication of that proof.

So the test then is to run the construction of the proof, which gives you a finite automaton, which is a candidate … a candidate for the finite automaton that recognizes the context-free language. So all you have to do then is test that candidate against that, with the grammar. Within the construction is involved several levels of exponentiation, because you have to account… because of epsilon moves, you have to worry about whether the information on the stack is going to go "poof" and disappear before you're forced to count it. So while you push down the stack, it has to have certain properties that won't disappear, "poof." Anyway, it's a … both the proof was difficult and … the time of the algorithm was remarkably large.

Rosenkrantz: Let me change topic here. Can you please tell us about your first exposure to computing?

Stearns: Yes. I'd never seen a computer until I got to General Electric. There was no computer when I got there, but they soon got the time-sharing system, may even have tied into the time-sharing system at Dartmouth where you ran the BASIC program. And it was run on a Teletype. You'd type in your program on a Teletype and the output would show up on the roll, and it was of course a long time for the Teletype to print something out, so you had the ability to just look at a portion of your program. You'd say, "Well, show me lines 400 to 500," and it would print that out so you didn't have to look at your whole program.

That was my first experience. Soon they added FORTRAN to that time-sharing system, or GE added the FORTRAN. There were certain problems where I had to use FORTRAN because… I think it was because I wanted to write files on the disk, and the BASIC language didn't have that capacity. So for no other reason than wanting to use that capability, I had to write a FORTRAN program. Then later, the ALGOL compiler was developed by Dan, Phil, and some programming help. That was put on the system and I switched over to programming in ALGOL.

And I would write programs. Probably the most interesting use I made of the computer was a certain game theory problem, when I was interested in questions regarding what they call the kernel of the game. One would have to test a certain vector of payoffs to see if it was in the kernel. You could easily make a mistake doing that by hand, so I wrote a program which I would then experiment and put a description of the game in there and I would experiment with different vectors to see if they were in the kernel.

Well, then there was a question of whether, if the vector you were testing was not in the kernel, could you do something to make it in the kernel or get it closer to the kernel. So that was easy to do. I put it in a loop where I'd put in the vector and figure out it wasn't in the kernel and then figure out the biggest change I could make in the vector which would make it closer to the kernel. That led to the game theory paper I wrote on "Convergent Transfer Schemes."

I found from time to time the computer is a useful tool for working on examples, or as a computer aid in research.

Rosenkrantz: When did you first think of yourself as a computer scientist and how did that come about?

Stearns: Well, at the time when I came to GE, there was nothing called a "computer scientist." I took on the title of Mathematician. They asked me what title I wanted. That was the title I wanted, Mathematician. Well, it turned out that the group of people who were interested in the work were what we'd now call theoretical computer scientists, although the people would think of themselves as electric engineers or mathematicians. There was just discussion around whether there was such a thing as computer science and, if you went to academia, would you dare be in a department called "Computer Science" or would you want a joint appointment with some other department?

Well, then computer science departments started showing up more and more. Juris Hartmanis of course went to Cornell early and he started a computer science department there. So I start thinking of myself a mathematician/ computer scientist. At some point, I joined the ACM, Association for Computing Machinery. That is a symptom of, an indication of starting to think as a computer scientist. Later I dropped my memberships in the math societies. I guess that completed the transition, *[1:10:00]* although I still think of myself as a mathematician/computer scientist, because all I really do is prove theorems.

Rosenkrantz: How did you get involved in parsing, syntax directed translations and attributed translations?

Stearns: Well, context-free languages was a hot topic at that time. General Electric was getting into the time-sharing business, and learning languages, one would have to learn syntax. There were some interesting papers around about parsing and several kinds of parsing had been invented, one being the LR(k) parsing that Knuth had developed. We thought, it made us think about parsing, and naturally when you parse, you have to do something… If you're really running a compiler, you have to do something in addition… in addition to just recognizing that thing has been correctly written. So you want to take actions while you're parsing. Well then, if you're taking actions while you're parsing, you'd like to take those actions as early… the earlier you can take an action, the better.

So we came up with a concept we called TD languages, for "top-down." When we submitted that paper to Knuth as editor, somebody said, "You ought to call that 'LL parsing." Since he was the editor, that's what it's been known ever since.

Well, so then first then, the syntax-directed translations was really about taking actions to make outputs. A compiler takes actions of course, all kinds of actions as part of the compiling process. The idea of attributes was already around. Knuth had a paper on attributes. We saw that we'd want to… We asked the question, "How are we going to pass those… How are we going to actually do the computation of these attributes while the parsing is going on?" That led to our paper on attributed translations.

Rosenkrantz: The textbook on compiler design written by the two of us and Phil Lewis was an early book on compilers. Can you tell us what led to this book?

Stearns: Well, basically we had a lot of… we had developed a theory, as discussed, and the development of the ALGOL compiler, I don't remember if that preceded the book. But anyway, you know a lot and you want to write a book. This is the book here. We intended it as a textbook. We found that if we put it in the IBM series that they were willing to sell the book less than if it was sold as an academic book. That's why we published it in this particular series. Oh, it took a long time to write the… it took several years to write the book, because I don't know if it was a little bit painful to write a book or because we were distracted by other things.

Rosenkrantz: How did you get involved in FORTRAN compiler and what was that experience like?

Stearns: Okay. Well, at that time, GE was getting into not just time-sharing computing but they were planning to manufacture and sell computers. For some reason, they wanted a new FORTRAN compiler for their computer. So the main work was going to be done in Phoenix with their computing division. But we agreed that we would write a front-end for the compiler. So the people in Phoenix said, "We want such-and-such for the next pass of the compiler," so we put our theory to the test to write the front-end, because we did like to support the company when we could. They sent somebody from Phoenix to work on the front-end as I designed it. It turned out that that was the coldest winter ever remembered in Schenectady with piles and piles of snow.

But anyway, we did it, and there's some hardships in doing it. According to the FORTRAN, spaces don't count, so you have to look way ahead to know what kind of statement you're working on. They also wanted to be able… when you wrote a Boolean expression, they wanted to be able to rearrange the terms to what they thought may be better. And a third peculiar thing was with certain… they're called "inline functions" in FORTRAN where you make a single statement to define a function, and they wanted those to be treated like macros. So there were a number of problems there which weren't strictly parsing, but we got through it.

But in the meantime, General Electric sold their computing business to Honeywell. By the time the project was finished, it was a Honeywell project. Of course, since Honeywell is no longer making computers, I don't know whatever happened to that compiler. But there were certain strange things built into it. For example, because its built-in functions were being treated like macros, I found a way almost to call them recursively. Not quite, but they would call themselves to a limited depth. One could test. If one ever found a FORTRAN compiler, you can test it on that example to see if it behaved that way.

Another thing we did was that complex numbers, they were written in left parentheses, comma, right parentheses. We had it so that you could actually put expressions in for… I mean the design only called for putting constants in for the complex numbers, but we treated *[1:20:00]* the construction of a complex number as an operation. So if one ever finds a compiler and wonders about it, one can try writing a complex number using variables instead of constants and see if it goes through.

Rosenkrantz: At GE, the two of us together with Phil Lewis worked on the travelling salesperson problem. Can you tell us how that work started?

Stearns: Well, I guess there are two things. One is that some components of GE had problems, which were travelling either… correctly modelled by travelling salesman problem, not actually sending salesmen around. For example, there was a problem with drilling holes in a sheet of metal. The quicker you could get from one hole to the next, the better you were. And there were several colleagues at the research lab who had sorting algorithms … uh, not sorting … algorithms for travelling salesman, including nearest neighbor and some of the others. We decided it would be fun to analyze them, so we did a paper on analyzing the travelling salesman problem. Originally, we were calling it "travelling salesperson problem," but the referee found that objectionable, so reverted to "travelling salesman problem."

Rosenkrantz: At GE, the two of us also worked with Phil Lewis on concurrency control in database systems. Please tell us what led to your interest in this topic?

Stearns: Well, databases, concurrency control and databases was of interest to the company. One particular database need they had was in regards to their atomic energy program, because they're required to keep track of … strict government regulations on keeping track of where the nuclear fuels are. So they were very interested in a robust concurrency control. Our main paper on that was to show that to be consistent, if the database program was both reading and writing, then they would have to be serializable, and there was an interesting exception for programs which didn't write. I think we also had some other ideas in some conference proceedings. Phil Lewis pursued that much further, eventually wrote a book on concurrency control, whereas we moved on to other things.

Rosenkrantz: Along with your work in CS, you were continuing to make contributions to game theory. Can you tell us about this?

Stearns: Yes. Of course my thesis work on *Three Person Cooperative Games without Side Payments*. There's an interest side story on that. When I came to GE, that's exactly what they had done. They had colluded with two other companies, Allis-Chalmers and Westinghouse, as far as bidding on contracts was concerned. Sometimes one company would make a lower bid and sometimes the other companies would make a lower bid. That was cooperation and there were no side payments. I think they actually used the phase of the moon as a random device to decide who was doing what. So it was kind of a joke that they hired me at just that time.

Well, anyway, beyond the thesis, the next thing I did was… Well, my thesis was on von Neumann and Morgenstern's solutions. There was always a question of whether von Neumann and Morgenstern's solution would always exist. I found an example without side payments where there was no von Neumann–Morgenstern solution. The original question was for "What about games *with* side payments?" Well, I mean the solution concept was already in trouble because, in general, there were games where the solutions were too abundant and where any kind of a compact set could be part of a solution.

So it was in some sense already on the way out. People were looking for alternatives. But that was the start of the downfall and it was destroyed by William Lucas, who took the same ideas and found a game with side payments that had no solution.

After that, I did my work on convergent transfer schemes, which I already discussed.

Then some computer scientists were doing some work, consulting work for the Arms Control and Disarmament Agency. The main contractor was Mathematica, the consulting firm in Princeton. They were going to gather some computer scientists, some game theorists together for a couple weeks just to work on their problems. One of the problems that Bob Aumann and Mike Maschler worked on solving … was on repeated games of imperfect information. They had shown that yes, they knew how to solve the games if one of the players had the perfect information about the other player. So they knew, they had strategies for the player with knowing whose basic type was a secret and the one who didn't.

But the question when I joined this group, which was not the first meeting, was "What do you do if both players have imperfect information about the other player?" Anyway, that was still open at the end of that particular session. So I took that back to GE with me to think about it, and thinking about the problem top-down and thinking, "How would you show there's no solution?"

Well, I had the idea that, *[1:30:00]* since the game was repeated forever, sort of the first hundred times or so forth wouldn't count due to… I mean whatever payoff you got then, since you're interested in what happened in the limit, you'd make it all up. Sort of the beginning didn't count. I thought, "Well, one player should wait until he gets all the information about the other player, and then make his…" See, the thing is the one player may have a different… maybe one of several types, and the idea if you can make moves to conceal your type, which means doing the same thing regardless of which type you are or you can make moves that are more favorable to the type you happen to be, was you make the moves more favorable to the type you have to be, the other player can start figuring out which type you are and play accordingly.

Anyway, my idea is that it's sort of like picking the larger number. What is the strategy for picking the largest? You and I are each supposed to pick a number and see who has the larger number. Well, there is no optimal strategy because obviously if I know what probabilities you're using on different numbers, I'll just pick a number much larger than what you're picking and vice versa. So not all games have solutions.

Then, well, if I was going to wait out the information, then how could… somehow I'd have to measure the information. I wasn't sure how to do that, so I went to Phil Lewis, into his office and said, "Well, I want something like…" I told him vaguely what I wanted, and he made a suggestion and that suggestion worked. But somehow I never gave him credit for that. He deserves credit for providing me with that idea. Anyway, I was able then to follow through on that idea, with one player waiting out, now having them… can measure the information for the other player and understood what it meant to get all the information that that player was going to provide. Anyway, it showed that there was no solution for that case.

Well, still, that went into a report to the Arms Control and Disarmament Agency, where very few people knew about it. The proceedings were passed around among a few people, but it was not generally known. Then at some point much later, in fact I think it was around 1985, Aumann and Maschler decided to take that information and put it together into a book. They invited me to in a sense provide a chapter, basically what I'd written for the Arms Control and Disarmament Agency, make that a chapter in the book. But that wasn't a big enough contribution to be considered an author, but here's the book here. If you read the fine print, it says, "With the collaboration of Richard E. Stearns." I don't have my glasses on, so I think that's what the fine print says. So it's kind of a semi-author. Anyway, this book did win the Lanchester Prize for book of the year, so that's the story of how I almost won the Lanchester Prize.

Anyway, continuing with my contributions to game theory, some game theorists were proposing that to measure a strategy in a repeated game, they were asking, supposing an automaton is playing that, is moving the strategy, and let's call the number of states that that automaton uses as the measure of the difficulty or complexity of the strategy. Well, in my view, that was not really a good measure because there are very simple ways to describe some non-regular sets. For example … if you took the strategy that after every thousandth move you'd go left instead of right, well, that's a very easy strategy to describe and to write a simple little program to do it. Namely, you count up to a thousand and repeat. But that would take a thousand-state machine. So I thought as an alternative that one should say, "Alright. Instead of a finite-state machine, we'll use an automaton with a bound on the amount of tape." It would be like a linear bounded automaton but with no input, so not exactly linear bounded, but…

Anyway, and that, the measurement and the amount of space you needed to carry out the strategy, you could carry out a very complicated strategy at a very limited amount of space. I mean the difficulty with that, which of course is obvious, is that the algorithm might be exponential in the amount of tape you were using. But that measurement actually resembled the Kolmogorov complexity of single sequences in that if you change the tape alphabet, or did things like that, the measure is independent of the… I mean you could write the strategy down on the tape and the complexity was basically independent of the number of symbols you were using, in the same way that Kolmogorov complexity is invariant in respect to the language you use to describe it.

Let's see, that contribution and that … I gave that talk out in Stanford. I had it written down and that became a Stanford Report, even though I was out there for one week and I had presented that result at a game theory conference, but it never came out as a publication. I've also recently been writing a paper on games with imperfect recall, essentially showing that *1:40:00]* if you have a little bit of imperfection, there's still good strategies for it. It's not a big cliff when you go from perfect recall to imperfect recall, but it's only as the amount of the imperfection so to speak grows that difficulty in describing strategies grows.

Rosenkrantz: What influenced your move from GE to the University of Albany?

Stearns: Well, the first thing was that the culture at GE was changing. When I first got there, all the money came from in effect taxing the components of the company and then they would say, “Go do something interesting.” But more and more, they were asking you to find components of the company that could support your work or asking you to get government contracts. The freedom was slowly being whittled away. I think that happened not only at GE but on research labs around the country.

Okay. Second of all, I thought, now that I’ve done all this work, I have something I can contribute to teaching. The prospect of teaching now became more interesting. Also, I did teach a database course at RPI for a semester to verify that I could teach and have some feeling as to what teaching would be like.

Rosenkrantz: Was it compilers or databases you taught at RPI?

Stearns: No, it was databases. I also taught compilers some time earlier at Union College.

Then the university offered me a very attractive position as a leading professor with a salary which was essentially the same as what I was getting at GE, and I wouldn’t have to grovel my way up the academic ladder from assistant, associate, that route. And it was easy to switch to the university because it was an easy commute from where I was living. And of course Dan, you were there, which was attractive. It looked like the department there was up and coming. For all those reasons, I switched.

Rosenkrantz: At U. Albany, your work with Harry Hunt introduced the notion of power indices for hard problems. This was a precursor to subsequent work by other researchers on notions such as the exponential time hypothesis. Can you tell us how work on power indices came about?

Stearns: Well, in general, Harry Hunt’s research, three things that are very important to him. One is to prove things with using as few assumptions as possible so it has the widest possible applicability. The second is to make your reductions as efficient as possible, that is as small a size as you can, so that the implications for hardness are stronger. And third, try to make your reductions go onto natural subsets that might occur in practice, so you’re really proving something hard. And this work was kind of in line with that philosophy. We defined the power index of a function. For example, 2^n was called power index 1 and 2 to the square root of *n* was called power index ½, and so forth.

Then we noted that if you assumed that the power index of SAT is 1, which we call the “satisfiability hypothesis”, then according to the size of the reduction you made, you could make conclusions about the complexity of the thing you were targeting. For example, the partition problem. The reduction from SAT to the partition problem, as originated by Karp, is n^2 in size. That is, given the SAT problem, it maps it onto a partition problem, which is n^2. That means there's a match between the size of the reduction and its implication for hardness, and the algorithm that we have. I mean there's no wiggle room there unless you can do SAT better.

The suggestion we made is that if the conclusion you draw from your reduction doesn't match the best algorithm you know, then there's room for improvement, either improving, improving your look for a faster algorithm or look for a better reduction. Those are the two ideas of using that notion, one to talk about the implications of your reductions according to size and the other suggests where you should look for improvements either in the algorithm or the reduction.

Rosenkrantz: At UAlbany, you also worked with Harry Hunt on algorithms for combinatorial problems by exploiting subproblem independence. Can you describe how this work came about?

Stearns: Well, again, looking for cases and situations where the structure of the problem gives you improvements over the general algorithm. And we looked at satisfiability, and we looked at it as an algebraic problem. That is, you want to evaluate the sum of products over a ring and the conditions that would make it go through, that is, which would make dynamic programming go through are distributive and commutative laws. Then we called it "channelwidth" because that is the amount of information you need to pass from one subproblem to another.

Now these same ideas can be attributed to treewidth, but our thinking was that these things were more naturally described as sum-of-product problems. Of course, treewidth became… people published problems in treewidth by saying, "Oh. Well, for a bounded treewidth, this is polynomial," which is actually a consequence of the idea *[1:50:00]* that the problem can be solved by nonserial dynamic programming. Of course, if you say this problem can be solved by nonserial dynamic programming, that's not as sexy as polynomial for bounded... Perhaps we could have gotten more notice by presenting it that way.

Anyway, this algebraic approach did lead to some other, a couple of other papers. One is we showed how this could be extended to quantified formulas. The second, we showed how, if you have something that can be done nondeterministically in sublinear space and in linear time, that that can be explained entirely in terms of channelwidth. That is, all those problems could be reformulated as SAT problems with limited channelwidth.

Rosenkrantz: You also worked at other problems with Harry Hunt, such as the complexity of simple Boolean formulas and using algebra to characterize the complexity of problems. Can you tell us about this work?

Stearns: Well, again, that fits into Harry's basic philosophy of making sure that your reductions go to natural subsets. In the case of the Boolean formulas, we considered equality of two Boolean formulas and containment, and the results were of the form "Well, the difficulty is hard even if the variables only appear once on each side, and one side is a CNF and the other side is a DNF of… two levels of CNF… of DNFs." So the idea that these are the simple formulas, ones that are more likely to occur in practice. And the other work, again trying to achieve the most generality for results, they all kind of fit in that rubric.

I should add that during the time I was chairman, this was very helpful to me to work with Hunt. Otherwise, I might have stagnated.

Rosenkrantz: Can you talk briefly about your PhD students at UAlbany and the nature of your interactions with them?

Stearns: Yes. There were a couple students that come to mind which I worked with Harry Hunt advising. One was Venkatesh, or "Venky" for short. He did things like investigating planar problems and why that might be easier and how treewidth would apply to those. The other student I worked with Harry on was Madhav Marathe, who was interested in, among other things, the complexity of hierarchically presented problems. The nature of the interactions, Harry did a lot of the interacting with these students and I was mainly following his lead.

There were other PhD students at the time which I served on their PhD committees and who worked with combinations of faculty members. I know you and Harry had several students.

Then there was Tom O'Connell much later, who was interested in game theory. He worked on some implications of limited information and mechanism design, showing that even under very simple conditions, mechanism design led to some very hard problems.

Rosenkrantz: Can you tell us about the courses you taught at UAlbany?

Stearns: Let's see. I taught the theory course many times, both to graduates and undergraduates. I did teach discrete structures once or twice before it got so big. I taught databases and algorithms. I also taught the introduction to computer science for non-majors, and that was a giant course. I figured I should contribute along with other faculty members at teaching something large. The advantage of that course was that it was taught in the language BASIC, which was the language I knew, so I didn't have to learn the latest version in the ALGOL, C, Pascal, that sequence.

Rosenkrantz: For some time now, you've been working on the topic of discrete graphical dynamical systems. Can you discuss this work?

Stearns: Yes. This is a project that our former student Madhav Marathe has been leading, getting financial support for. I was invited to join as sort of a subcontractor on this work. Then it leads to some challenging problems like "What are the consequence of the nodes in this network, that they have certain kind of functions? What are the implications of those functions? What happens in the course of operations as these networks proceed from one time to another. What can you say about the overall flow of information and the changes to the configurations?"

Rosenkrantz: What other problems have you been working on since you retired from UAlbany in 2000?

Stearns: Well, I mentioned that I was currently working on, I'm writing a paper on games with imperfect recall, which… It involves… I've always been kind of interested in how information relates to game theory. There's difficulty in dealing with these things in that the… the information is valued. You might say two pieces of information *[2:00:00]* which are very similar are … that not so… The distinction is not important, so having to … I finally sort of figured out how to handle this problem where strategies have to be correlated and you like to have … I guess it's another way of putting it – how much correlation you need for a particular game in order to have an optimal strategy.

Rosenkrantz: You published a paper with your father. Can you tell us how this came about?

Stearns: Yes. My father was an expert on color. He worked for American Cyanamid in the branch that produced dyes. In particular, the problem of matching colors. That is when you take a swatch of paint to the store and say, "Match this." How is that done? In fact, he wrote a book on that subject and he also actually did write a system, an early system to do this for Benjamin Moore, the paint company.

There's a certain need for interpolation and extrapolation in the calculations. That is, when I take the swatch to the store to match the color, they sample it. It maybe would take 20 samples at different wavelengths. Well, the method, the standard method, maybe the supposedly accurate method would require that you have a lot more measurements. So you had to take these measurements from the measurement you've taken and you have to extrapolate what that function looks like. By "the function," I mean really the spectrum of the color sample that you're looking at.

So we had discussions on how to do that interpolation and that extrapolation. It's not very deep mathematics, but it's quite useful.

Rosenkrantz: What's your opinion about the P versus NP question?

Stearns: They are definitely different. I guess the question is when will we ever get a proof that they are different? I am very… I mean I wonder if we ever will get a proof. I mean if I give you a Turing machine for example and say, “This is solving satisfiability,” how would you know? I mean if you want to show that no such machine exists in polynomial time, I say, “Well, supposing you do have a machine.” Well, what can you ever say about such a machine? I mean the Turing machines are really inscrutable, so try to say anything about what an arbitrary machine is doing is basically undecidable. Of course, depending on how you phrase the question.

The problem really gets back to Turing machines. You have a nondeterministic Turing machine and you want to make a Turing machine which does the same thing. That means to some degree the Turing machine has to anticipate what the nondeterministic… Somehow it has to condense the operations of the nondeterministic machine to some degree. I don’t know how that… I don’t see that being done. So it’s a very hard problem and may never be solved.

Rosenkrantz: Can you tell us about the theory conferences of the ’60s that evolved into FOCS and STOC in later years, and more generally how did conferences come to play such a large role in the culture of computer science?

Stearns: Well, at the outset in the early ’60s, people who were doing theoretical computer science was a small community. I mean you can argue a little bit about “What is theory?” and sure, you can point to… people say, “These people doing AI, they weren’t part of the conference,” and so forth. But the conference really, which was then called Switching Circuit Theory and Logical Design, was the place where people interested in theory got together and exchanged ideas. At that time, the proceedings that were being put out were really almost full papers. The program committee would select the papers maybe half would get in or something, I don’t remember statistics. But once the proceedings got out, that was practically a publication in the sense that most people who were interested had access to it. For example, Cook’s paper was never published outside the conference proceedings.

Well, as time went on, these things got bigger and bigger and more topics became… there were more papers on the same topic. For example, languages. There were more and more papers on languages, a lot more than could possibly be accepted, so the special interest group POPL was spun off of that. The databases got… when they grew, that was also spawned off.

Well, there I should say, besides the switching theory and automata theory, Switching Circuit Theory and Logical Design, which changed names a couple times until it became FOCS, the Foundations of Computer Science, a second conference was started so that there would be a conference in the fall and one in the spring, the other one being STOC. One was sponsored by the ACM and the other by the IEEE, the engineering society, to accommodate more papers. Those two conferences were really the spawning ground *[2:10:00]* for a lot of spin-offs from that. In some sense, the development of computer science can be measured by the growth of these conferences and the spin-offs they produced.

Rosenkrantz: What are your views on the development of computer science?

Stearns: Well, I have a theory that somehow things are driven by the hardware available. When I first started out at GE, we were working on a Teletype. After a while, we had little terminals which would display characters. But with development of more and more computing, more sophisticated terminals, then something like Windows became possible. I mean suddenly there was enough computational power to do some very sophisticated graphics. I think as now things in artificial intelligence which seem to be things that are computational-heavy and now can be done. So in some sense, we're being driven by the hardware. Of course, it's obvious larger and larger problems are being handled … and lessened … These are not as amenable to theoretical analysis as some of the other things. I mean they have programs to learn to play games. The programs learn and no one really understands what they've learned, only that they're playing well.

So I don't have a grand vision as to what happens. I mean there are developments that really amaze me, like when I Google something and things instantly… how often it comes up with relevant papers and references.

Rosenkrantz: Do you have any advice for young computer scientists?

Stearns: Well, the first piece of advice is to do what you love. When I was at the Heidelberg conference, there was a young researcher going around asking people to… she had something like an autograph book, saying, "Write down… I want your signature and I want your advice." You know, "Say something." I said, "Do what you like," "Do what you love," or something. She said, "Well, that's what everybody's saying." Sure enough, some of the… I asked her, "Show me what some of the others wrote," and they wrote some of that.

Second thing is picking the right model. And the third thing is to think top-down. Now earlier in this interview, I mentioned the work with Hennie and how I started thinking, "Well, how can I take these two ideas, these two approaches we have and merge them," and gradually that idea was refined. You should do what you love, be sure to work on the model to get the right model, and think top-down.

Rosenkrantz: Looking back, what turning points and major decisions affected your life?

Stearns: Well, I mentioned things as we kind of went along here, that things just happened to *me [chuckles]* kind of without too much planning. Suddenly thinking about Carleton. Having the visit of Kemeny, which persuaded me to go to

Princeton. My friend Roger suggesting to GE that they hire me. And things like that. So I don't… The only real decision I made I guess was to switch from GE to SUNY.

Rosenkrantz: What have been your interests outside computer science?

Stearns: Several I could talk about. First is in dancing. When I started at graduate school, there was a folk dance group, a very active one. I tried that out and found it good exercise, stimulating, and sociable. And I continued. When I got to Schenectady, I continued folk dancing with a group there. After dancing there a year, a young woman caught my eye, came, and after a year's courtship, I married her. Meaning my wife Charlotte, of course. We married in '63, and so that's 54 years we've been married. So that was a very happy occurrence.

As the folk dance group kind of faded, we started doing contra dancing as well, and finally mainly contra dancing. After that, we got interested in English country dancing, which I don't know if you've watched a film, a Jane Austen film like *Pride and Prejudice*. You'd see that kind of dancing there. Now that I've slowed down a bit, I can't make moves as fast, the contra dancing is now out, but I can still do the English dancing because it's slower and emphasis on grace and being in tune with the music rather than stomping around, and we've gone back to some international folk dancing. So that's been a lifetime hobby and joy.

Another activity, when I was young, like in high school and even before that, I got very interested in chess. My father took me down to a chess club when I was in seventh grade, and I played in that chess club for… I was the only kid in the club, but I did that for six years and became somewhat proficient. I did far less of that when I got to Carleton and even less, hardly at all, very little since then.

In the meantime, I'd been playing bridge at home. At Princeton, I started playing at a bridge club there. I've been playing bridge fairly regularly since then for a number of years. We were just in a small group of couples who got together. Since then, I've started playing in club games as well, gone to some of the national tournaments and achieved the rank of Silver Life Master, where if you know the scheme of things, it's not really very far up on the different ranks. *[2:20:00]*

Then in the early '70s, when McGovern was running for president, I got kind of interested in politics and sort of figuring I should do something when I was invited to join the Democratic committee, which I did. And got involved in some political infighting there and became chairman of the party. I was party chairman in Niskayuna well, first for four years and then later for another stint of a little longer. I've always felt it important that, you want to do something for society's benefit, you ought to work on that.

Let’s see. Then I guess one final thing, when my son was in the Scouts and the scout leader wanted to run a pinewood derby… Pinewood derby is when you make cars and they go run down a track by gravity and you try to pick a winner. The scout master, Marc Borom, asked me, “Well, how should we do this?” I decided to develop some schedules with the idea that instead of playing the knockout or something, that all the cars should race the same number of races. So we had a schedule for that and Marc Borom then wrote up the whole thing as an article in a scouting magazine. Since then, that’s become known as the Stearns–Borom method of doing car races. If you Google “Stearns” and “pinewood derby,” you’ll find it. Another community which seems to appreciate some things I’ve done.

Rosenkrantz: Okay, Dick, I think that’s all the questions I planned on asking. Is there anything else you want to say?

Stearns: Well, I want to thank you, Dan, for agreeing to be the questioner on this and for all the good times we had at the research laboratory and working together at SUNY. It’s been a great run we’ve had together and I very much appreciate that.

Rosenkrantz: Thank you, Dick.

[end of recording]