A. M. Turing Award Oral History Interview with E. Allen Emerson

by Thomas Wahl

Austin, Texas

March 4, 2019

Wahl: Welcome to the ACM Turing Award Oral History Interview with Allen Emerson, who's Professor Emeritus at the University of Texas at Austin. This is the place where we have come together here, at this beautiful campus, more precisely at the Gates Dell Complex. The day is March 4th in 2019, apparently the coldest March 4th on record in Austin, and it's a bit of a chilly day indeed.

My name is Thomas Wahl. I have the pleasure and the honor of conducting this interview with Allen today. The audience is going to be both the general public and historians of science and of computer science, and we will make sure that our questions and responses are catering to these audiences.

With that being said, Allen, it's a pleasure to be here with you today. How are you?

Emerson: I'm fine.

Wahl: Very good. Let me first say a few words about how I'm planning to structure this interview. I would like to begin with your upbringing, childhood, very early education through high school, then go on to talk about your college education and your PhD at Harvard University. From there, it will be a small step to talking about the invention of model checking, which is what gave rise to the Turing Award that brought us together here today. I would then like to talk a little bit about what you think your legacy is in computer science and the field of formal verification. I would like to talk about your relationship with colleagues, friends, competitors maybe, how they shaped you, how you shaped their careers. And then, speaking of careers, I would like to know a little bit about how you view your career in this field, any ups and downs, highs and lows that you would like to share with us, and finally talk about the person Allen Emerson, what kind of habits, what idiosyncrasies you're willing to share with us here today. Does that sound like a plan?

Emerson: Yes, it does. A good plan.

Wahl: Okay. Good plan.

Let me maybe up front quote from the Turing Award citation that is shown on the Turing Award website for you. So I quote. It reads, "Together with Edmund Clarke

and Joseph Sifakis, for their role in developing Model-Checking into a highly effective verification technology that is widely adopted in the hardware and software industries." We'll obviously have plenty of opportunity to talk about this in the interview today.

I want to begin though with a very light note. Allen, I've seen you sign letters and emails with the initials "E.A.E.," three letters, which kind of remind me of some sequence of alternating quantifiers. You usually go by the name "Allen Emerson" as far as I know, so maybe you can enlighten us first about the meaning and the origin of this first "E." in these initials.

Emerson: "I am R. Daneel Olivaw. The 'R.' stands for 'Robot.' " Seriously, the "E." stands for "Ernest." My full name is Ernest Allen Emerson II, written Roman numeral II.

Wahl: OK. Good. Now that we have clarified that, we can sort of begin with the serious part of the interview. I would like to talk first about your background, your childhood, your early education. What do you remember of your parents, your relationship with them? What kind of wisdom sort of did they impart on you? And maybe if you had any other role models in your early childhood.

Emerson: Okay. I grew up in the Oak Cliff part of Dallas in a middle-class neighborhood of suburban-style homes. My father started out as a fireman in Dallas and ended up as the Texas State Fire Marshall in Austin. And my mother was a homemaker.

In elementary school, I was savvy enough to realize that athletes were at the top of the social strata and I played football in the sixth grade. But unsuccessfully. My school team lost every game. It was not rewarding.

In high school… Have you asked about high school?

Wahl: I have asked about high school too.

Emerson: In high school and junior high school, I especially liked STEM subjects. And, in particular, mathematics … subjects. In the seventh grade, I read a book by Isaac Asimov called *The Neutrino: Ghost Particle of the Atom*. That was probably the single most impactful event of my early life. I decided that I wanted to be a particle physicist. I tried reading additional relevant physics books, but I had a lot of trouble, and I concluded that I didn't know enough mathematics. So I started reading various mathematics books. In particular there was a book called *Elementary Calculus from an Advanced Standpoint*. By the time I had assimilated this and I took high school calculus in the twelfth grade, I

already knew calculus by the time I took that course. So I was sort of tilted towards mathematics. But I also took a before-school course at 7 a.m. in computer programming. That kindled my interest in computer science.

Wahl: Did that 7 a.m. course involve actually working with computers, or was it…?

Emerson: This course taught the students BASIC programming, and then I taught myself Fortran and Algol from the Algol 60 report. At the beginning, we were on a GE Mark I time-sharing service, and then we had a time-sharing service into a Burroughs B5500 computer, I guess at the school district downtown. So…

Wahl: Okay. This gives us a glimpse where sort of the scientific interest in Allen Emerson came from. One thing I'd like to know a little bit more is what did your parents have to say about this? Were they happy with you going into apparently technical stuff? Or did they have something else for you in mind?

Emerson: Oh no, they had nothing else. Well, they were sort of religious. They were not happy when I started reading Bertrand Russell.

Wahl: Okay, okay. [laughs] And just one more thing about your childhood. Do you remember going to movies much and stuff like that?

Emerson: I think I saw *Westworld* one time, but I don't remember anything else.

Wahl: Alright. Apparently there wasn't so much about that.

Okay. Then let's maybe move on to the next educational stage, which brings us to college, and incidentally this brings us sort of back to UT, which is where in 1976 you obtained a bachelor of science in mathematics from UT. Please tell us about those years. In particular, I'm wondering whether this bachelor of science, I mean did that happen in the math department or in the computer science department? Was there any computer science education at all in this program?

Emerson: I'm pretty much self-taught in computer science and in my interest in verification. At UT, I was a math major, and *[0:10:00]* I took a number of Moore method courses, which are named after the famous mathematician R. L. Moore. My understanding is that when he was active in the '30s and '40s, the UT PhD in mathematics program was comparable to the best US departments like Princeton and Harvard. In a Moore method course, the instructor writes on the blackboard statements of theorems, but the student is solely responsible for devising proofs of the theorems, and no access to books at the library is allowed and no

collaboration with other students is allowed. Now if you're beyond a certain level of ability, I would say that's the ideal way to learn mathematics.

Wahl: Okay.

Emerson: At some point, I made this observation though. Mathematics is about 5,000 years old. It's likely that the only open problems remaining are really difficult ones. Whereas computer science is approximately 50 years old. As a new field it should be much easier to come up with… to perform important work. And I was encouraged as an undergraduate to go to grad school in mathematics, but the above considerations gave me pause. I ultimately performed my unique act of common sense when I decided to go to grad school in applied math and computer science. It's worked out pretty well.

Wahl: Very good. And we will obviously talk much more about that. Just one more thing I wanted to know about the college years in Austin. UT Computer Science obviously has been a hotspot in verification for several decades, including I think already in the mid-1970s when you were here. So I'm wondering, even though you studied mathematics back then, did you meet any of the computer scientists?

Emerson: I sat in on Woody Bledsoe's course in automatic theorem proving or… But I didn't meet the other people. Maybe Don Good.

Wahl: So the little exposure you had to Woody Bledsoe for instance, was this also relevant for your decision to study computing and maybe verification in particular? Does this incite some knowledge?

Emerson: No, it was just something to do.

Wahl: That you were curious about.

Emerson: Now when I returned to UT in 1981 as an assistant professor, that was the same time that Boyer–Moore were joining UT CS department as associate professors. I remember there was some kind of party boat and a picnic, and my wife and I sat with J Moore and his wife. But that's the extent of it.

Wahl: Okay. That was a couple of years later.

Okay. Then how about we move on to your PhD program, which you conducted at Harvard University. Before we talk about this program and what precisely you studied, I want to talk about location a little bit, because you were sort of a twenty-something boy from Texas in those days and you moved to the Northeast

with all the ramifications that this comes with. Can you tell us what this was like? Had you ever travelled to the Northeast before, or was this a completely new territory at the time?

Emerson: It was completely new to me. The overall impression was that Boston was sort of worn hard. I remember there was a secretary in another building whose husband was a grad student in physics at MIT. She asked me, "I bet you find it really hard here at Harvard given that you're from Texas." And, you know, I thought that was surprising since she was from like South Carolina or North Carolina. And there was a good Greek student, Greek-American student from the Northeast who asked me sort of sarcastically, "Do they have libraries in Texas?"

I found Harvard to be a lot of work, but it was tractable. I finished in four years, whereas most people finished in five. So, it worked out pretty well that way. I found Boston, especially Cambridge to be a much more interesting place than Texas, especially Austin. In Cambridge, you had Harvard and MIT, and in the Boston area, you had Boston University, Northeastern, Boston College, Wellesley, and 250,000 students in the Boston area, and three---now four---medical schools. I don't know of any place else in the world where you have such a concentration of intellectual talent. I found it very exciting and interesting.

I guess the bad thing about Boston was that the cold weather was really horrible by Texas standards. And the cost of living was exorbitant.

Wahl: I also want to talk about your wife. I believe you were married at the time already. And if so, did she come with you, and what did she think about this place? Was she trying to get away from it as quickly as possible? Or what was her attitude to…?

Emerson: She came in 19… late '77 or early '78, when I came in September '77. She got a job working for Harvard Medical School. More precisely, she worked for Stillman Infirmary, which was a clinic in Harvard Square for Harvard affiliates staffed mostly by Harvard Medical School faculty getting their clinical time. I remember she said to me, one time, "These Harvard Medical School faculty are every bit as smart as computer scientists." Then later she worked at the MIT medical department, which was staffed mostly by Boston University Medical School faculty getting their clinical time. Then later she worked for the chief of medicine at Boston City Hospital, which was the main teaching hospital for Boston University Medical School. *[0:20:03]* But she did say, I don't know, three years in or something, "You better finish in a year, because I'm going back to Texas, no matter what."

Wahl: And you already told us that you finish in four years as opposed to most people, so I think that that worked out, right?

Emerson: Yeah.

Wahl: Okay. Very good. But obviously I also want to talk about the actual program of study at Harvard, which was applied mathematics. And in fact that is your PhD degree.

Emerson: Yeah. It actually says "*mathematica accomodata*" – it's in Latin – but the English translation I think would be "applied mathematics." But it was really in computer science. At Harvard, well, first, I understand CS as a proper subset of applied math. Applied math at Harvard, part of the Division of Applied Science and Engineering, includes decisions and control, electrical engineering, classical applied math, and biomathematics. But at Harvard, many places actually do applied math – economics, statistics, the separate statistics department, and the School of Public Health, and the business school, they do optimization and so forth. But the pure math department doesn't do applied math, or otherwise they wouldn't be pure.

Now while I was there, that same Greek student, Greek-American student asked Derek Bok, then President of Harvard, why Harvard did not have a computer science department. And Bok replied that Harvard plans in terms of decades and centuries, and the planners are not convinced that computer science is anything but a transient phenomenon. So, Harvard has a computer science program but not a department. And I guess Harvard is still that way. You *can* get a PhD that says "computer science," but I don't know if that's English or Latin.

Wahl: Okay. That's insightful.

Emerson: It's progressed that far.

Wahl: Right. Now about the computer science education that you received there, maybe the first thing you can tell our audience, since we're also speaking to historians here and hopefully people who are watching this interview 50 years from now, what was the computing world at the time? Did you even have a computer lab there? I assume that there were no laptops and stuff.

Emerson: As I recall, we had a DEC TOPS-10 system. We were on the ARPANET – I don't remember my email. There were terminals in the machine room, not outside the machine room, and the machine room was very messy. And there were certainly no personal computers or laptops then.

Wahl: Okay. About the courses that you took on computer science–related subjects, can you maybe tell us about them, maybe a book that you read back then which might also be well known to people today? In particular, I'm curious whether the notion of program correctness came up in any of these courses.

Emerson: As far as books go, I found books by subsets of Aho, Hopcroft, and Ullman to be well written, and I found them influential. I also liked David Gries' compiler book. And, Zohar Manna had a book, *Mathematical Theory of Computation*.
I had seen Zohar Manna give a talk here (I think in Painter Hall 3.14) on the Tarski-Knaster theorem in 1975. And that was included in his book which was published about the same time. I thought that was neat.

So far as I know, there were no real courses at Harvard in verification then. I did take 6.830 at MIT, which was a course mostly about program semantics and a little about verification. But that didn't really have a great impact.

My interests are pretty much… I was self-taught from… I read a paper by Hoare, "Proof of a Program: Find," maybe in 1974, and that talk by Manna. The idea of proving a program correct seemed like a neat idea, but that wasn't, you know, what I went to grad school necessarily committed to do.

Wahl: Okay. That's insightful. Since your interest in verification was sort of a self-taught interest, that tells me that you probably didn't work on your PhD topic on day one. I guess this took a little bit of time. So can you talk about that? And also when did Ed Clarke come into the picture? When was it clear that he would be…?

Emerson: Your first year at Harvard is devoted to taking courses, and Ed arrived the second year. I think I went with him when he interviewed to the Harvard Faculty Club, which served horse steak. But I didn't… Well, at some point I guess in that second year, I started working with Ed because we were both mutually interested in verification.

Wahl: Okay. Then let's transition into your research that later led to the Turing Award. Before we talk about model checking specifically, let's talk maybe about verification a little bit more broadly and you tell us about the state of the art in verification in the 1970s. What could be done? What could not be done? What kind of gaps were people thinking need to be filled?

Emerson: I would describe it this way. The earlier work on program verification didn't work in practice. It usually involved constructing a proof using axioms and inference rules that the program satisfied a specification expressed in some kind

of logic. And it could require a lot of ingenuity to construct such a proof. Maybe you would have to come up with loop invariants or something. And people were writing 15-page papers to prove one-half-page programs. And if you tried to prove something correct but you didn't succeed, you didn't know if you simply had not been ingenious enough or you had been overwhelmed by the tedium of trying to manipulate all these formulas, or maybe the program was wrong, was not correct, it didn't meet the specification, and there was no way necessarily to tell in practice what the situation was. That's the way it was in general.

Wahl: So that obviously sets the stage *[0:30:00]* for your work on model checking. I thought this might be a good time, before we talk more about that, to remind our listeners of the Turing Award citation, which says – again, I quote – "Together with Edmund Clarke and Joseph Sifakis, for their role in developing Model-Checking into a highly effective verification technology that is widely adopted in the hardware and software industries." And I'd like to spend a couple minutes obviously with you talking about this. Maybe the first thing is if you would be so kind and tell us in layman's terms, what is a good way to understand what model checking is and what it does?

Emerson: In layman's terms, model checking is an algorithmic method of verifying correctness of nominally finite-state systems against a specification that's typically given in temporal logic. If the model checker, the model checking tool that's been implemented returns "yes," then the system is correct. If it returns "no," the specification is violated, and a counterexample is produced.

Wahl: So this is sort of where we want to go. Now maybe you can describe how, from your memories, the invention of model checking came about, in particular obviously what was your role in it, what was perhaps the role of others? How did that develop? Where did it come from?

Emerson: Joseph Sifakis is handling the European side of the exposition. And I'll talk about the North American side, and something about my interactions with Ed.

I got the ball rolling. I was responsible for the initial conception of model checking. It was a spinoff of my earlier work on program synthesis. You could ensure that whatever synthesis method you were considering was sound by applying the synthesis method to generate a candidate program or a candidate model, and then checking algorithmically that the candidate model was a genuine model of the specification given in temporal logic. And without loss of generality, if a temporal logic specification has a model, it will have a finite model of bounded size. So that is implicit there. And checking that a genuine model… checking that a candidate model is a genuine model, that motivates the term "model checking."

I had an important meeting with Ed right before he departed for POPL ’81. It was in the reading room on the ground floor of Aiken Computation Lab. Now as part of some consulting work, Ed was already investigating a proof-theoretic approach to verifying protocols using temporal logic. Ed had not thought of the algorithmic approach.

So I told Ed about model checking, which was algorithmic, and Ed made a fundamental realization or conjecture that model checking could be practical.

At that meeting, I also told Ed that I found an error in Pnueli’s POPL ’81 paper, which dealt with satisfiability and not model checking. At that time, Ed and I were both fully conversant with Pnueli’s logic, which was called “UB” for “unified branching time,” and it’s pretty much the same thing as the logic we call “CTF,” computation tree formula, and that Pnueli cited in his paper, but he had a more streamlined notation. Then CTL *[computation tree logic- ed.]* was like UB plus *until*.

So I give Ed a handout explaining the error. Ed checks the handout over and agrees to transmit it to Amir when he gets to POPL ’81. This transmission did get through because in the journal paper, revised journal paper of Pnueli’s paper, Ed and I are acknowledged.

Finally, Ed and I agree that I will work out the algorithm while he is at POPL ’81, and then when Ed returns from POPL ’81 and after visiting his parents in Virginia, Ed calls me at home and I confirm that I have the algorithm.

Oh, one other point. “Pnueli ’81” is really short for “Ben-Ari, Manna, and Pnueli, POPL ’81.”

Wahl: Right. In fact, I want to talk much more about other related works in this field at the time, but first a quick clarification. You mentioned CTL and CTF. Maybe for our audience, you can say what CTL stands for, and how is this related to the CTF logic, which probably is not as well known to most people?

Emerson: CTF has an ugly syntax, and Pnueli cited it in defining his logic UB for “unified branching time.” And CTL could be viewed as a similar streamlining of CTF, or it could be viewed as UB plus the *until* operator.

Wahl: And what does CTL stand for?

Emerson: Oh, I’m sorry. “Computation Tree Logic.”

Wahl: "Computation Tree Logic." Okay. Thank you. I want to talk a little bit about that paper which is today often cited as the one that made the term "model checking" emerge, which was the 1981 Logic of Programs paper, LOP, the conference that's today called Logic in Computer Science, with the title, Clarke and Emerson, "Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic." So it's a fairly complicated title. And the title doesn't say anything about verification, let alone model checking, so I guess people in the future will ask, and I think already today this is not clear without some context, why this paper, which introduced model checking, doesn't talk at all about verification. It sounds a little bit like…

Emerson: It does talk about verification. It has a section on model checking, but it wasn't reflected in the title.

Wahl: In the title, yes. So would you say that verification maybe came about as almost a byproduct of actually different work, almost like a positive accident in history? Or is this saying too much?

Emerson: Yeah, it was serendipitous. Well, first of all, that paper, Clarke–Emerson, LOP '81, is the paper which introduces the term "model checking" and the concept of model checking. In the full citation to the Turing Award, it was dubbed a "seminal paper." *[0:40:00]*

Why did it appear in LOP '81, Logics of Programs 1981? Well, Ed finished Cornell with one paper. And I wanted to do better than that. I already had the ICALP '80 paper that defined CTF. Model checking was implicit in that paper, but people have to be hit over the head many times before the concepts soak in. I wanted to finish with two papers, and so I encouraged Ed to use this work for Logics of Programs '81. I was happy with alphabetical order, and Ed gave the talk. Immediately afterwards, I was offered a job at Cornell.

Wahl: Maybe to round off this picture, the 1980 paper at ICALP, I understand that probably was your first publication?

Emerson: Yeah.

Wahl: Okay. Can you maybe give context and say what was the title of that? What did that say?

Emerson: "Characterizing Correctness Properties Using Fixpoints" or "Characterizing Correctness Properties of Parallel Programs Using Fixpoints." One of those two.

Wahl: That sounds like a title that is more easily connectable with model checking perhaps than the other one.

Emerson: It was implicit. But it takes time. I mean it was there. Really model checking can be defined as evaluating a temporal logic formula over a candidate model. I found maybe some yellow sheets in my handwriting with brown ink talking about one application would be to evaluate fixpoints. It's a simple idea that people kept missing.

Wahl: I guess things like that happen all the time in science. Would you---if you could do it over again, would you rebrand maybe that 1981 paper, give it a different title if you had to do it today?

Emerson: We could have done a better job, but I mean it turned out okay. I don't know why the paper had the title that it did. I don't remember. "Design and Synthesis." I will tell you, Ed frequently cited it by "Synthesis of…" He dropped the "Design." He was sloppy. By the time we got to the POPL '83 paper, which subsequently was published in *TOPLAS* '86, but the POPL '83 paper was informally identified among us as the practical paper. There we did discuss using "Model Checking" in the title, and we finally decided just to use "Automatic Verification."

So it would have been better maybe… Well, I don't know. If you used the term "model checking" for the first time, people might not have known what it was. It might have been the correct decision to have it embedded in the body of the paper.

Wahl: Yeah. This is in fact a very good transition to the next question I wanted to ask. If we move a little bit away from the immediate group that you worked in, what was the initial reception of model checking? Did people even understand what this was?

Emerson: First, I was the sole student of Ed's at Harvard that worked on model checking. Christos Nikolaou and Prasad Sistla worked on temporal logic but not model checking.

The initial response outside to model checking was confusion. People didn't know, didn't understand what it was. It was so basic, I mean that… Okay, satisfiability is checking if a formula *has* a model. Does there exist a model of a formula? Model checking is, given a model, a candidate model, does it satisfy the formula? So the definition of satisfiability, it contains the definition of model checking within it. People were sort of obsessed with satisfiability checking, and

they didn't understand it because it was too simple. Then after they did understand it, they hammered it for potential problems with state explosion.

Wahl: Yes. That is definitely a topic we will also come back to. One more thing I think that we need to talk about in this historical context is that, so this work of yours was not the first to talk about temporal logic as a way of reasoning about programs, right? There was… In fact, you mentioned Amir Pnueli already several times in what you've told us, and he had this famous paper about using temporal logics to describe the behaviors of programs. And let's maybe mention for the audience that of course Amir Pnueli is also a Turing Award winner, in 1996, for this very idea. How does that relate to your work?

Emerson: Pnueli is credited with seminal work introducing temporal logic into computer science. And I would concur. But no idea is well founded. You can always see someone else who sort of hit on that topic. In particular, the philosopher Arthur Prior, who was at Oxford, speculated in 1967 on the use of temporal logic to describe the workings of a digital computer. And in 1974, Burstall, who I guess was at Edinburgh, he proposed the sketched the use of modal temporal logic for establishing program correctness. But Pnueli convincingly demonstrated the use of temporal logic both as a specification formalism and gave a proof, a deductive proof that a program met its temporal specification. That's what Pnueli accomplished.

Wahl: Can you also talk about the role of verification in Pnueli's work, like the idea of algorithmically deciding whether programs and specifications match? Is this something that…

Emerson: Well, okay. Now…

Wahl: Was the idea of…

Emerson: …the chief weakness of Pnueli's work, although it was masterful in terms of laying out the concepts of temporal verification, nothing was precisely and rigorously well-defined. *[0:50:04]* Here's the situation. In 1977, Pnueli was at Penn, and Saul Gorn gave Amir a book by Rescher and Urquhart called *The Logic of Command*. Amir looked through it and found it to be useless. But on the book jacket, there was a blurb for another book by Rescher and Urquhart called *Temporal Logic*. That was a great book, but it was very intuitive and not rigorous. There was no definition of double turnstile or the Tarskian definition of truth. So, as an afterthought, Pnueli claimed that he could give decision procedures for some temporal logic formulas, but he couldn't really because nothing was well defined. You need the Tarskian definition of truth to rigorously give a model-checking algorithm. So Pnueli gave deductive proofs, but he didn't give

algorithmic proofs. He suggested that deductive proofs might be useful, but he makes no such claims for algorithmic proofs.

And in Pnueli's second paper on temporal logic, he again only uses deductive proofs, not the algorithmic approach. So he missed it.

And to back up, I thought about writing a paper called something like "Model Checking: A Frequently Overlooked Concept." You have the protocol people in the '70s, and they identified the importance of search for reachability analysis, to try to find an error or establish there were no errors, but they did not have temporal logic and so they could only do safety properties. Then Dijkstra in his book *A Discipline of Programming* in 1976, he talked about manual calculation of weakest precondition---which is just AF---and the weakest liberal precondition, that's AG, but he didn't contemplate coming up with an algorithm for it. And, well, Pnueli*,* again he's very intuitive but not so rigorous…Pnueli '77.

Wahl: Okay. So automation is one of the differences that you see between the work by Pnueli and your work. And we'll talk about that more. One more thing though about Pnueli versus Emerson, Clarke, etc., is this notion of branching time versus linear time. Maybe we should say a little bit about this difference for the audience, because this also gave rise to heated discussions after that.

Emerson: I was always an advocate for branching time, and so was Ed, and so was Sifakis. Branching time can distinguish between "It's inevitable along *all* futures that something eventually happens" versus "It's possible along *some* future that something eventually happens." You have explicit quantifiers over paths or futures. And in linear time, you talk about a generic timeline, that something eventually happens along that timeline and there's an implicit universal quantification over all futures. But you can't… linear time is not closed under semantic negation. There's no way to even express "There is a future such that something happens" or "There is a future, something always happens." The existential quantifier over paths is absent. But Pnueli–Lamport for example were advocates of linear time, and Ed, Joseph, and myself branching time.

Wahl: Was the branching time aspect perhaps one aspect that made it a little bit harder for people to use CTL for instance compared to LTL?

Emerson: I think a strong argument for linear temporal logic is its simplicity. I don't think it was a major difficulty. I mean people bought into branching time and CTL (and Fair CTL where the path quantifiers are relativized to fairness constraints). You would be surprised that there was always debate over this.

Wahl: Sure. Okay.

Emerson: …I respect linear-timers.

Wahl: [laughs] Very good. Do you remember maybe one of the papers that you read in ’77?

Emerson: The only paper I remember is Pnueli ’77, which I did read as a first-year grad student. But it went in one ear and out the other. It really didn’t have at the time any impact on my work around then. I didn’t realize until… Retrospectively, I didn’t realize until much later its significance.

Wahl: Okay. I would like to make a little bit of a switch and talk about the adoption of model checking in industry. The Turing Award citation states that the method is today “widely adopted in the hardware and software industries.” But I guess it’s fair to say that in those early days in the late 1970s, early 1980s you laid the foundations, but at that point I assume it was still a far cry from being applicable to industrial designs. So my question is what steps did you guys and maybe others take to make this technique more palatable for industrial applications? And perhaps you can also say where do you think are we today? Has this gap basically been closed? Is it even closer than it used to be at that time?

Emerson: Perhaps I’ve already said this. Initially, people didn’t even understand what model checking was. It was confused with and overshadowed by satisfiability. And once people understood what it was, it was criticized for not dealing with the problem of state explosion adequately.

But I should mention the POPL ’83 / *TOPLAS* ’86 paper had significant impact. People, by reading it could get a pretty good understanding *[1:00:00]* of how model checking worked, the internals, and they could apply model checking to solving their problems, either using an existing tool such as EMC or BMC or, a little later, SMV, or they could design their own tools.

So small industrial-strength examples could be handled for hardware verification and protocol verification from pretty much the early or the mid-1980s. But later on, there was significant progress by various means to limit state explosion so larger designs in programs could be handled.

Now I would say that there are actually two important issues for model checking. One is expressiveness and the other is efficiency.

Regarding expressiveness, you need to be able to express the correctness properties that you’re concerned with accurately and conveniently and perhaps succinctly. I worked on the logic CTL, Fair CTL, CTL*, and the Mu-calculus,

which provided in principle adequate expressiveness for a large number of applications. There was PSL, Property Specification Language, that was sort of derived from CTL with special operators for hardware verification. It became an IEEE-1850 standard and people used it with a lot of specially designed model-checking tools for example in hardware verification, and there are books about it.

The other issue is efficiency. One thing that refers to algorithms for model checking, linear time is better than quadratic time, and for fair cycle detection.

And the other thing is to limit state explosion. We have various strategies to limit state explosion – symbolic representation, partial-order abstraction, and compositionality. These various strategies were implemented and turned out to be significant in practice.

Wahl: If you in fact think about efficiency, the expressiveness of properties, maybe also the ability of model checking to generate counterexamples that might also be relevant for industry, do you think any one of these things was the one that gave the breakthrough for industrial use? Or is it really their combined…?

Emerson: I don't think there was any single advance that was a breakthrough. Probably the most dramatic advance was symbolic model checking using BDDs *[Binary Decision Diagrams - ed.]*. It was developed by Ken McMillan, but it built on the CTL model checking algorithm as an enabling technology, and it built on the BDD technology of Randy Bryant. The CTL model checking algorithm was Ed Clarke and myself.

Wahl: Okay…

Emerson: Now I'll also say partial order reductions can be very useful in applications for software and protocols where you have sort of an irregular state graph, but it can be very slow. And, well, that can limit its practical utility because maybe you can only get one run in in a day or in several days.

Both of these techniques, symbolic BDD-based model checking and partial order reduction have sort of limited scaling. For example, with BDDs, you could handle systems with 100, 200, maybe 300 state variables, but they didn't scale beyond that. And similarly for partial order. So these techniques were sort of heuristic. They were algorithmic given unlimited resources, but in practice they might not be able to represent or traverse the representation of the entire state graph.

I myself was always interested in techniques such as symmetry reduction for systems composed of many similar processes or parameterized reasoning about systems of size n, n a parameter, where at least in principle for many interesting cases you could establish a rigorous upper bound for the size of systems that

could be handled. Parameterized verification is undecidable in general, but for things like cache protocols, you can get good bounds.

Wahl: It sounds like from what you're saying that, since its inception in the, around 1980, a lot of the research on model checking was about curbing the state explosion problem. Is that a fair statement?

Emerson: Yeah, that's true.

Wahl: I want to shift the topic a little bit from the technicalities of model checking to talking about collaborators, colleagues, friends, maybe competitors in this business of yours. And I want to start with again the Turing Award that we're talking about here ultimately, which in your case was bestowed upon you and Ed Clarke and Joseph Sifakis, who at the time was in Grenoble in France, so reasonably far away. It's always been said that they worked independently along with somebody named Queille on ideas related to model checking. I guess many people will ask, will wonder, did you ever meet Joseph Sifakis in those days? Of course you have met him since then, but what about those early days and…?

Emerson: I met Joseph at LOP '83 I think, and I would bump into him at conferences subsequently. I have no idea who Queille is.

Wahl: Okay. About these early encounters with Sifakis, the question I guess many people will ask is were you two, the two parties basically aware that they were working on related ideas? Or did you never even talk about this maybe, if you recall?

Emerson: I had a vague recollection. *[1:10:00]* I mean I knew Joseph existed, but… He wrote a famous paper related to abstraction, "Homomorphisms between Transition Systems." But the development was really independent.

Wahl: It sounds like it took some years until this …

Emerson: Well, our work was published in Springer *Lecture Notes in Computer Science* 131 and his in 137.

Wahl: [laughs] Okay. That's fair. I obviously want to talk about you and Ed Clarke. If I am right about this, you were Ed Clarke's first student, at least one of the first? You can clarify that for us.

Emerson: I was Ed's…

Wahl: So yeah, tell us where sort of in the timeline you were and…

Emerson: I was Ed's first student. And Ed had many subsequent excellent students, especially after he went to CMU. Christos Nikolaou and Prasad Sistla were at Harvard, but they didn't really work on model checking. Christos Nikolaou worked on maybe proof-theoretic verification of certain protocols using temporal logic, and Prasad worked on the complexity of linear temporal logic for satisfiability and its fragments.

And then there were other students, in particular at CMU. Ken McMillan really had the insight with symbolic model checking. He was going to Randy's class and going to Ed's class, and he did a good job of seeing the obvious, that you could symbolically represent sets of states and relate to transition relations using BDDs and then calculate… evaluate temporal logic formulas over those representations and do it symbolically.

And David Dill, who just retired from Stanford, he did circuit verification. I remember Ed joking to me one time that Dill was going to call CTL "circuit testing logic." And Somesh Jha did work on CEGAR, counterexample-guided abstraction refinement. And there are many others, postdocs. I think SAT-based model checking may have gone back to Armin Biere.

Wahl: Okay. These were some of PhD-student colleagues and later PhDs, etc. But what about Ed Clarke himself? What do you remember about him as your advisor and…?

Emerson: I found Ed to be a great advisor. He had excellent taste and superior supervisory skills, and I would dub him the chief model checking evangelist.

Wahl: Good. How has that relationship evolved over the years? Obviously it went far beyond the PhD time itself.

Emerson: I haven't talked to Ed in some time.

Wahl: Okay. I want to talk a little about the colleagues at the University of Texas, which is where we are now and where you were as a professor. But before we do that, let's maybe sort of set the record straight here and discuss the time when you finished the PhD. You pretty much immediately moved to UT as an assistant professor. Many people here will be curious what in those almost – what? – 30-plus years ago certainly, almost 40 years ago, what the job market was? What did an interview look like in those days? What did you have to do? How easy was it to land this job right after graduation?

Emerson: When I became an assistant professor, the global job market for PhDs in computer science was very good. There were something like three openings for every fresh PhD. That didn't mean you could necessarily go to where you wanted to go, but the odds were improved. I was very happy to return to Austin, because the lifestyle is cushy, and I was happy to return to UT.

Wahl: When you started working there in formal verification I assume, I'm curious, do you remember the funding situation? Especially model checking obviously was hot off the press, was very new. Was it possible to get funding either from government or from the industry for this kind of work, or was it sort of too fresh for the sponsors?

Emerson: No, I never had trouble getting funding. My first student, Chin-Laung Lei, was from Taiwan; and he initially… okay, he had translated Apple Computer manuals into Chinese and, I don't know, made two hundred thousand. And he lived off his CDs. But the first time I got a grant, I started supporting him. And every time I applied for a grant from NSF, I got it. I think you could say that for Ed as well. So it was fashionable enough to get funding.

Wahl: Very good. Now about this question of collaborators. I think I mentioned earlier that UT obviously had been like a hotspot for automated reasoning verification, etc., for many, many years. When you joined them as a faculty member, what colleagues do you remember most and…?

Emerson: I never collaborated with any verification people here at UT, maybe not with anyone at UT on anything.

Wahl: But you had interactions with these people, right?

Emerson: Well, I'd talk to them and I have many friends and colleagues here. But I don't think we worked together.

Wahl: Okay. What about…

Emerson: I think maybe Jim Browne, the late, great Jim Browne started working on model checking in, I don't know, the '90s or 2000s. But maybe you and I went down to see his student, Natasha *[Sharygina - ed.],* while she was working on something, buffer overflow detection by model checking. I don't know. Myself and several of my students went down there to see a presentation that the student of Jim Browne's gave, and we made comments, and it came out to be more rigorous.

Wahl: Okay. But I'm thinking about not just collaboration towards papers and stuff. You must have had interesting hallway conversations with some of the people at UT at the time, right? *[1:20:00]* Dijkstra maybe…

Emerson: Oh. Well…

Wahl: They must have had an influence on you.

Emerson: That's another story.

Wahl: Anybody else at UT?

Emerson: Sometimes I would talk to Jay Misra.

So my scientific work was pretty much orthogonal to the background milieu here.

Wahl: Okay. What about colleagues outside UT and even maybe outside the United States? Do you remember any people internationally? And by "colleagues," I just mean…

Emerson: Okay. Let's back up for a minute to Harvard. I would say the most influential person, the most influential student I found to be Oded Shmueli, who's as good Israeli that's both a deep and savvy thinker. And the most influential faculty member I found to be Albert Meyer, who's very clearheaded.

And, well, there is a story. Circa 1986, when I came up for promotion, Oded Shmueli was in town at MCC, Pnueli was at UT, and I think Hoare was here. But after the department voted in my favor to recommend promotion, Oded Shmueli and his wife Deborah had Amir and Ariela and my wife and I over for dinner. It was very nice. And there was some Pakistani guy I don't remember. Yeah, so…

Wahl: A promotion celebration dinner?

Emerson: Yeah.

Wahl: Okay. Anybody else even internationally maybe that you want to point out?

Emerson: Let's see. I was really always impressed with Krzysztof Apt and Willem-Paul de Roever. I met Krzysztof maybe at ICALP '80 or maybe at LOP '81. And Amir had arranged for Willem-Paul de Roever, who was at Eindhoven then, to have me over for a summer and part of a semester. Amir cautioned Willem-Paul that "You won't see Allen during the day, [laughs] because

he only comes in at night." I had to get a special pass to get into the university at night, and I was supposed to give gin to the porters or the guards. But that was a very pleasant experience. De Roever's grad students would be like assistant professors anyplace else. They were unusually mature and smart. And I enjoyed it thoroughly.

Wahl: Can you say again, where was Paul de Roever at the time?

Emerson: He was at Eindhoven then.

Wahl: In Eindhoven also. Okay.

Emerson: Subsequently he moved to Kiel.

Wahl: To Germany.

Emerson: To Germany.

Wahl: Right. I see. Okay. And how long did you spend in Eindhoven?

Emerson: Well, I spent maybe summer of '87 and a large chunk of the spring of '88.

Wahl: Was this your first sabbatical I assume, maybe?

Emerson: They don't have sabbaticals at UT.

Wahl: Oh, they don't have sabbaticals at UT.

Emerson: You can get some sort of Dean's leave, but it's not guaranteed. You can apply for it.

Wahl: I see. I would like to move a little bit and to talk about your legacy, what you think your legacy is and perhaps what you suspect what other people might think about you and write about you in computer science books in the future. Maybe you can just first give us your impression on that. Perhaps beyond technical contributions, what would you like to remembered of?

Emerson: I am proud of my role in the invention and development of model checking, and I don't have anything to say beyond that except for some advice to budding computer scientists.

Wahl: Okay. Would you like to share that right now? What do you… Let's maybe first say, speaking of legacy and development, etc., what do you say about verification research today? Is it, is it … the questions … We talked a little bit about for instance efficiency of model checking earlier and things like that. Do you think that verification research today is talking about the right problems, is addressing the right issues?

Emerson: I think today mostly we all contend against stupidity. As the philosopher I think Franz *[Friedrich – ed.]* Schiller said, "Against stupidity the gods themselves contend in vain." There are plenty of problems – conventional verification, verification of real-time systems, verification of hybrid systems, and so forth – and we'll never run out of work.

And I think a more important topic is to contend against malevolence in terms of dealing with security. I know that there has been a good deal of work on using model checking and other sorts of verification approaches to deal with security protocols and maybe cryptographic protocols, but obviously this work is not that successful. Computer hackers prosper and you're always reading about some company's database was invaded and stolen. I don't think we have a very good handle on the security problems.

So it's highly respected work to contend against stupidity, but I think that misses the main thrust, the main issue.

Wahl: That sounds like there's definitely enough work to be done by future generations in computer science and in verification. Can you summarize maybe what you would say to those aspiring researchers in computer science, verification? What career advice might you have for them?

Emerson: First of all, I would say that you should work very hard to get as technically strong as you can. But then look for something very simple. Maybe practice identifying simplicities and imagining uses for them.

Now, I read a book entitled *A PhD is Not Enough!* that recommends *[against - ed.]* selecting a very junior advisor. The reason is a very junior advisor will tend to compete against the student and will find it difficult to give the student appropriate credit for their contributions. *[1:30:00]* It's not always the case, but that can turn out to be the case. So carefully choose an advisor. A very senior advisor might have a level of wisdom and a basket of problems that could be helpful, and a junior advisor might have the limitations, liabilities above.

Still, I would recommend formulating your own problem, don't focus on a problem chosen by your advisor, and I would say a bit of floundering around on your part

is a helpful learning experience. After you finish, assuming you finish, you're going to have to, on a fairly routine basis, come up yourself with new problems and solve them or contribute to their solution. So you might as well start doing that from the beginning. Don't take your advisor's problem. You come up with it. And if you can, try to own it. On the other hand, if your advisor jumps on it and starts elaborating it and developing it, that's arguably a good thing.

If you're not so imaginative or technically strong, don't despair. If you have good social skills and you're savvy, you can try to be a manager and run a team of subordinates.

There's also room for some quirky types of creativity. I once knew a student who never really made sense to me. Not once. But he turned out to be quite successful. Once he made a comment after he finished to his company and saved them several hundred millions of dollars, and he's coauthored papers with some of the very best computer scientists in the field. So there's not a unique path to success.

One other thing I would say is play to your strengths and try to avoid having to rely on your weaknesses. That would be my advice.

Wahl: Okay. After all this advice for the next generation, I do want to come back a little bit to your career, which – and please correct me if that's a wrong assessment of mine – I would characterize as a fairly stable career. You have spent your professor career at the University of Texas until you retired very recently, and you have worked on relatively similar topics – temporal logic, verification, model checking, maybe automata theory. But still I suppose that there were some highs and lows in your career, and I also suppose the Turing Award was one of those highs. But maybe you can expand on this a little bit. What were the highlights, positive highlights?

Emerson: I have no regrets about what I did, but I would say the highpoints include, in 2008, the day I was notified that I got the TA; in 1998, when I got the Kanellakis Award…

Wahl: Can you say maybe for the camera what the Kanellakis Award was for?

Emerson: Oh. Well, it said actually for the invention of symbolic model checking. In 1983, the single exponential determinization algorithm for Büchi automata derived from temporal logic formulas, this was work with Prasad; Pnueli characterized it as a very ingenious argument. I'll stop that there.

Wahl: Okay. Good. Those were the highpoints. When there are highpoints, there are…

Emerson: That's what comes to mind.

Wahl: That's what comes to mind. Okay. What comes to mind at the lower end of that scale? Any disasters you would like to share with us? Yeah.

Emerson: I can think of one. In 1985, I went to Dijkstra's Tuesday Afternoon Club and we went over my POPL '85 paper. Dijkstra took extreme umbrage to the paper and he wrote a scathing memo criticizing me harshly. And it was devastating. But to Dijkstra's surprise, I returned the next week with a memo of my own, defending myself and counterattacking.

Wahl: A rebuttal of sorts?

Emerson: Yes.

Wahl: And did he accept it?

Emerson: Yeah. It's like I say, every cloud perhaps has a silver lining. Dijkstra and I eventually became good friends. He and his wife Ria would drop by, like, weekly and we would have interesting philosophical discussions. One time I was telling him about the advantages of model checking, and he said something like, "Sir, you are at risk of winning the argument." Dijkstra had an excellent dry sense of humor, and he was also bit misanthropic. Once I commented to him that there are like seven billion people on Planet Earth, and Dijkstra replied, "Yes. And 99.9% of them aren't worth a damn." [laughs]

Around Year 2000, Dijkstra was diagnosed with cancer, and he eventually went back to the Netherlands for a year. But while he was still in Austin, my wife went to see him in the hospital. Now the nurse knew that Dijkstra was a world-famous computer scientist, and the nurse also knew that my wife's husband --- me --- was also a computer scientist. So the nurse asked, "Ms. Emerson, is your husband also a world-famous computer scientist?" and my wife said, "Oh no." And then Dijkstra spoke up, "He will be." So that was sort of reinforcing, I guess. Yeah.

Wahl: So this low point that we originally talked about morphed…

Emerson: Yeah, morphed into… over 15 years, it morphed into something good.

Wahl: [laughs] I guess the lesson then for young people is it can sometimes take time for these low points to evolve and turn into something good.

Emerson: Yeah.

Wahl: Okay. "Be patient." [laughs]

So Allen, I think we're getting closer to the end of this interview. I would like to talk a little bit about the person Allen Emerson, the habits, idiosyncrasies, and things like that. If I may start with one suggestion, there's this rumor – I have actually never witnessed that – that you love animals and you're curious about what they can do and what they cannot do, what their sort of brain capacity is. Can you maybe tell us where this is coming from? Maybe this sort of maps us back to your childhood, but maybe not. And is there a favorite animal of yours?

Emerson: Oh, I forgot about the tropical fish. I like lizards, I like tropical fish, and I like rabbits.

I like these little green anole lizards. The scientific name is *Anolis carolinensis*, because they're Carolina anoles, another name. When I was a kid, you could go to the state fair in Dallas and buy a lizard in a small box. *[1:40:02]* [laughs] So I would keep them as pets. It turns out that, I don't know, around 1970 they had them wild in Houston. Then, I don't know what year, their range included Austin and then eventually Dallas. So over 20 to 40 years, their range has extended from the south to the north in Texas, and I suppose that's evidence for global warming. But---I think it's the first lizard to have its entire genome sequenced. There's some more work, a closely related species that is intelligent as some birds, but that's another story.

And for, like, 20 years I had pet rabbits. I got the rabbits because my wife got them when she was a science teacher in school. She brought the rabbit home on the weekend and I wanted to keep it. Now rabbits… I had lop rabbits, floppy-eared rabbits. But rabbits in general, they are of an intelligence level somewhere between that of a cat and a tree stump.

Wahl: That's a wide range.

Emerson: Huh?

Wahl: That is a wide range.

Emerson: Well, they aren't necessarily so bright, but they're cute, and they can be trained to do simple tricks. And they binky – they can run and leap and change course in mid-air, by kicking out their feet, hind feet. So yeah, I like them.

I maybe would have been happy as a herpetologist studying lizards and reptiles. I don't care for snakes.

Wahl: Do you have any animals today?

Emerson: No. I mean the green anoles are wild around our house, and this cold weather will take a toll on them. So…

Wahl: Okay. Good. I also want to talk about one habit of yours that I can say sort of much to dismay of some of the grad students that worked with you and maybe other people who worked with you, is the fact that you tended to be this very, very late person who would schedule not only meetings with his students preferably at 5 p.m., and Friday if possible, but also would schedule classes very late, make the students come here at 5 p.m. and listen. Where did that come from and how did people react to that over the years?

Emerson: I'm pretty much a night person. So was Büchi, J. Richard Büchi. I had a friend at MCC that used to be at Purdue, and he said that when he was coming in, in the morning, he would see Büchi going home after a night of work.

Well, circadian rhythms are biological, and I was reading maybe 1% of the population is like I am. But Steven Strogatz wrote, in 1986, a dissertation at Harvard Applied Math, *The Mathematical Structure of the Human Sleep-Wake Cycle*. And there's another guy, Art Winfree. He was a theoretical biologist. He wrote several books. One of them is called *The Geometry of Biological Time*. And those are milestones in the emerging field of chronobiology.

There's a fellow, Charles Czeisler, M.D., a professor at Harvard Medical School of sleep medicine, circadian rhythm disorders, and neurobiology, who studies these sorts of things. It turns out circadian rhythms are regulated by the SCN, suprachiasmatic nucleus, of the hypothalamus in the brain. Light cues and melatonin, the hormone melatonin, can be used to manipulate somewhat your schedule. I'm interested in his work, but I've never tried it on myself.

So I taught classes at 5 p.m., hoping and expecting to attract students who were on a similar schedule. I'm sorry if some of them were disappointed, but, you know, that's just the way I am. I don't really have plans to change it. It is a real inconvenience sometimes, but yeah.

Wahl: Okay. That's totally fair. But can you say since when did you adopt this kind of rhythm? Was this even the case when you were in grad school yourself?

Emerson: In grad school I would maybe get up at noon and really start working at 4p.m. to midnight, and maybe on to 4a.m. If I could sleep from 6a.m. or 8a.m. to noon, I was okay.

Wahl: Okay. Perhaps my final question that I want to ask is: we mentioned a few minutes ago that you retired a few years back, two or three years back from your professor job at UT, and you have been married to your wife Leisa for many, many years. Maybe just give us an impression of what your life looks like right now. What kind of things are you doing on a daily basis? Are you travelling? Are you reading books? Are you watching things? And also, Allen, if you don't mind, tell us if you are still in touch with the research world that you were in for so many years and are you maybe working on something these days?

Emerson: My wife is interested in travelling, and I am not so much. I have travelled plenty. But anyway, we're constrained by my mother's health, so we can't really go anyplace. I am working on some things, but I'm tardy in getting them written up for publications.

I don't read much. I think I spend too much time staring at screens. I think the Internet, the Web, television, while they're marvelous devices maybe for enabling people to, at a remote location somehow, to connect to each other, but they can also encourage… I don't know. They can discourage people from reading and maybe even thinking. *[1:50:00]* I'm trying to slowly read a book called *The Feeling of What Happens* by I think his name is "Dablasio." *[Damasio – ed.]* I prefer print.

But one thing that's not commonly acknowledged is stuff on the web for example is very transitory, and things can be changed without your knowledge. There's something called the *Handbook of Operating Procedures* for UT Austin, and I've seen it change. I mean they changed the rules and you don't have a hard copy. It's on the web so they can change the rules at their whim. I've seen stuff like this happen and you only have your memory to go by unless you had the sense to archive something. But even a digital archive, it's just, it's a pattern of bits. Only print… For the same reason secure voting involves paper stubs or something, print is apparently the only reliable way to archive stuff. I mean the ACM Digital Library, I've seen versions of an article, a model-checking technology article by, ostensibly by Leah Hoffman. Her name came and went on the electronic version. I don't know what it's in print.

So print is good. I mean if you have a hundred thousand copies distributed, you know, it would be very difficult for someone to sabotage them all.

Wahl: That's right, yeah.

Emerson: And so, yeah, I…

Wahl: Fair statement. Okay.

Emerson: So, is there anything else?

Wahl: I don't have anything to ask of you at this point. So then what remains for me to do is to thank you for your time. I think we have seen a good and interesting picture of Allen Emerson the scientist and also the person. So thanks, Allen, again for your time.

[end of recording]