# Catalyzing Computing Podcast Episode 13 - Interview with Dan Lopresti Part 1

The transcript below is lightly edited for readability. Listen to “Interview with Dan Lopresti Part 1” here.



*[Intro - 00:10]*

**Khari: Hello, I'm your host, Khari Douglas, and welcome to Catalyzing Computing, the official podcast of the Computing Community Consortium. The Computing Community Consortium, or CCC for short, is a programmatic committee of the Computing Research Association. The mission of the CCC is to catalyze the computing research community and enable the pursuit of innovative, high-impact research.**

**In this episode, I sit down with CCC Council Member Daniel Lopresti. Dr. Lopresti received his PhD in Computer Science from Princeton in 1987. In 2003, Dr. Lopresti joined the Department of Computer Science and Engineering at Lehigh University, whose research examines fundamental algorithmic systems related questions in pattern recognition, bioinformatics and security. In July 2009, he became Chair of the Department. In July 2015, he was the new Director of the Data X Strategic Initiative. In this episode we discuss his work applying computer**

**science, molecular biology, pattern recognition and voting machine security. Enjoy.**

*[Dr. Lopresti's Background - 1:07]*

**Khari: Okay, so you're listening to Catalyzing Computing here with Dan Lopresti. Dan, how are you doing today?**
Dan: Doing good. Thanks, Khari.

**Khari: Let's start with a little bit about your background. Where did you grow up and how did you decide to study computer science?**

Dan: That's ancient history. I was born in the Midwest actually, but my family moved east to New Jersey, near Princeton, when I was about 10 years old. So, if you ask me where I grew up, I would say, New Jersey. That's also where I first became involved in computing.

Like a lot of people in my generation computers started to infuse and infiltrate education about that point in time. When I was in high school, we got a couple of computers that were given to us by AT&T Western Electric. At the time, they were giving machine access to schools in the local area. These were terminals that connected to a computer that was actually at the AT&T Western Electric location. And there were no classes in computing and computer science, but a bunch of us got together and we started to play around with the machines. I think that's a very common theme, if you talk to people of my generation.

I had a math teacher who was really great, Mrs. Jackson, and she got us involved. She was willing to mentor a group of us and together we got interested in computing. My father actually worked at that Western Electric facility as well. So he helped teach me some basic programming, too. So I would say that was my start.

**Khari: Ok. So what kind of programming were you doing on these machines in those days?**

Dan: I am pretty sure it was BASIC, which is one of the first languages a lot of people learn. Then, when I went to college, I had the good fortune to go to Dartmouth where BASIC was developed. [John] Kemeny and [Thomas] Kurtz developed BASIC. Also Dartmouth was one of the very first schools to get deeply into time sharing as well.

**Khari: What is time sharing?**

Dan: Time sharing is the idea of taking one computer and allowing lots of people to use it, as opposed to a personal computer — which ironically came later but also was earlier — which is the idea that you own the machine. If you have one machine in a university and it's a huge, expensive machine and only one person can use it, then clearly that's very limiting in the impact it's going to have. So timesharing was really a revelation. The idea was that you could have one reasonably powerful computer, but dozens of people could use it simultaneously and get the impression it was their own machine. Now, we've gone back to personal computing again, but if you sort of go back in that era, timesharing was really quite a big deal.

I studied what was computer science at Dartmouth. It wasn't called that at the time. It was sort of embedded in math. A lot of it was in math and then some of it was in engineering, but mostly in math. But it was effectively a computer science degree, even though it wasn't called that at the time. I'm also of the era that a lot of people my age would have punched cards. Because I was at Dartmouth and Dartmouth was a leader in timesharing. I never punched a card. So I have not had that experience. I don't know what a punch card is. I've seen them, but I've never used them.

[Laughter]

**Khari: That's funny. So after Dartmouth, you got a PhD at Princeton, correct?**

Dan: That's right, yes.

**Khari: Could you talk about anything interesting that happened during the course of getting your PhD? Any interesting people you worked with there?**

Dan: That was a very different, very intense experience. I went directly from college to grad school. When I was in college, first of all, it never occurred to me I could get a PhD. I'll just say that right now. I mean, I looked at my professors and I had no confidence, no belief that I could be like one of them. It was just beyond the realm of imagination to me. So when I was at Dartmouth, I applied to business school. I got into a really, really good business school. And I applied to a bunch of Masters programs in computer science because that's as far as I thought I could go, but I needed money. I couldn't pay my own tuition at that point. My parents had paid for college. I didn't want to ask them to pay for graduate school, so I checked the boxes that I needed financial assistance.

Luckily, a bunch of universities realized how foolish I was, that I was qualified for a PhD. They wouldn't give me financial assistance as a Master's student, but they would as a PhD student. So I applied for Master's programs and I was accepted into PhD programs, which still kind of scared me a little bit like, “Wait a minute, I don't think I can do this.” So I said, “Well, you know, that's my best option now.” And, one of the excellent schools, Princeton, accepted me, and I went to Princeton even though it was about 15 minutes from where I grew up.

*[Parallel Algorithms and Systolic Arrays - 5:19]*

**Khari: Yes, that's the interesting path. So your thesis was on parallel algorithms for the rapid comparison of genetic sequences. Is that correct?**

Dan: That's right, yeah. So you know, again, I had the good fortune of working with a lot of really great professors at Princeton. My advisor was Dick Lipton, a very famous theoretical computer scientist, who does lots of different things. This is just when the Human Genome Project started to come out in molecular biology, and that was one of the first outside fields that made heavy use of computing and computer science. I don't remember exactly how it came about. It's also the time that VLSI, the idea that even at a University, a student or a professor could actually build a chip and have it fabricated.

**Khari: And by VLSI, you mean very large scale integration?**

Dan: That's right. That's the technology for laying down circuitry using a set of rules that were developed that took it from the realm of a professional engineer. [Carver] Mead and [Lynn] Conway, were the names of the two scientists who came up with this and the book [Introduction to VLSI Systems] is like the Bible that, sort of, started a lot of careers in computer architecture and parallel computing. They basically wrote a book that was accessible to graduate students or professors that told you what you needed to do to lay down the circuitry to design your own chips, basically, which could be high performance parallel computers or anything else.

Then, MOSIS was a service that was provided to actually allow you to fabricate these chips. So, for a few thousand dollars, you could actually submit your design and have it fabricated and get a real chip back and test it, which even now it's kind of mind boggling. I actually honestly don't know if they're still doing it, but I wouldn't be surprised if they are. Again, that was the spark that actually lit a lot of careers and it certainly attracted me.

So, I saw a problem from molecular biology. It was a problem that had a very nice, elegant solution using a technique that's called dynamic programming. It's actually the problem of comparing genetic sequences in sort of a fuzzy approximate way, where you allow for the fact that they're not going to be exact matches. And the dynamic programming algorithm is very elegant. It's order N-squared, so it's reasonably fast, but it's not super fast. But the sequences get long, that becomes an issue and then the N-squared becomes an even bigger issue. And in those days we were using mini computers, we were using, you know, DEC VAX minicomputers and things like that.

I was able to recognize that there was a way to parallelize, to take this basic algorithm for comparing genetic sequences, and develop a parallel version of it using a new architecture that was called a systolic array. It's a special purpose architecture just for this algorithm, but when rendered in VLSI — you know, built as a chip — it had the potential to be tremendously faster than a minicomputer. So I was able to take the algorithm, build the parallel architecture, do the layout for the chip, actually have the chip fabricated, and test it. We built an array of 300 processors that were actually hundreds of times faster than the DEC VAX was doing the same problem, and that became my thesis.

**Khari: OK. So is that system, that kind of systolic array still being used for genetic sequencing?**

Dan: Well, it's interesting. We did a lot of things in those days in terms of developing special purpose hardware and then more reconfigurable hardware. So that was also the early days of a technology called field programmable gate arrays (FPGA). And the idea there was that instead of actually hardwiring the circuitry, the way I was doing it using the Mead-Conway rules, you could actually program at a hardware level a technology that was called an FPGA. So we took that same algorithm and moved it from the hardware implementation — I did, I moved it onto an FPGA supercomputer architecture as well — but then what happened was kind of interesting.

What happened was the cost benefit ratio for building custom chips, and even for doing some of these more specialized things like FPGA supercomputers, got subsumed by the advances that would be made in just standard single process microprocessors. Microprocessors were being used in oodles of devices, and Intel and the other companies that were doing that were really maximizing the performances of microprocessors, really maxing them out. So it's really hard to stay ahead with what you can go buy from the local store. My special purpose VLSI system, which probably took me about a year to build, was hundreds of times faster than the VAX, but by that point, those machines were getting faster, so it became a race. It was a race that microprocessors won for a very long time until fairly recently when we started hitting some limits with clock speeds and densities and power consumption. And now it's really interesting — people are starting to turn back again to parallelisms. There was sort of a hiatus when it wasn't quite so feasible, especially in a university environment with a small team, but now parallel is becoming much, much bigger again with multicore, with Graphics Processing Unit (GPUs), with MPGA. A lot of this demand is coming from machine learning and deep learning. So we're seeing the tide turning back again in that direction.

**Khari: So you've been involved with the CCC AI Roadmap initiative. How do you think the shift to parallelism will affect the AI boom?**

Dan: Well, it's already happening now. A lot of the things that are being done now can only be done using high-performance machines and in some cases fairly specialized machines as well. It really is a race, and it's a race where it's a combination of the software, the hardware, and the algorithmic techniques all pushing forward at the same time. Organizations that can push on all of those things are going to have an advantage over someone that's, sort of, depending on two or three year old hardware technology and if they have the latest algorithms. So it's a very exciting time, but it’s also, you know, there's a lot of energy and in some cases resources being poured into this.

**Khari: Yes. So is there a reason why you picked a problem related to biology? Did you have an interest in biology or how did it come to your attention?**

Dan: Like a lot of things that happen, it's just serendipity, right? It's not like I was just chomping at the bit to do biology. You are just sitting at a table at some point and someone puts a problem up and you start saying, “h, that's kind of interesting,” you know, and then you dig a little bit deeper and it gets more interesting to you, and you dig deeper and it gets even more interesting. I can't say that there was a plan for anything that I've done. It's always been a lot of serendipity. Again, I think if I traced it back, I could probably say that there was actually one meeting — I can't remember what it was — but there was one meeting where the germ of this idea developed and then I ran with it.

**Khari: Yeah, I guess that's how a lot of great things have come together. So after your PhD you went into teaching, right?**

Dan: That's right. So I went directly from Princeton to Brown. I was in the CS Department at Brown. Again, I had this sort of imposter syndrome. I didn't quite think I was worthy of being a professor. I was a little bit like a deer in the headlights. You know, how am I going to do all this? That's also the time when a whole bunch of other life events happened too — I got married when I was at Princeton. You asked the most interesting thing that happened when I was at Princeton, like I said, getting married.

[Laughter]

Dan: My wife and I are still married and it's great. But then I went to Brown and in very short order, I started my first real job at Brown. I was teaching, I was doing research. Faculty members have to juggle a lot of things. I also bought my first house because

that was the right time to buy a house. And then I also had my daughter, all within a period of about two years.

**Khari: Right. That's a lot of life events.**

Dan: It’s a lot of things happening all at once, and only in retrospect now do I realize the reason I felt so overwhelmed: because it was overwhelming to have all of those things happen at the same time. It felt like an insurmountable situation that you're facing. And everyone, I think, has doubts about whether they're able to accomplish certain things. And at that point I remember thinking, “Is this right for me? What's going on? How can I get out of this?” But in the end, I did persevere. I eventually left Brown to help start a lab at Princeton, with my advisor, Dick Lipton, and some other Princeton grads and Princeton faculty. That was an interesting experience, too.

*[Pattern Recognition and 2D Barcodes - 12:55]*

**Khari: Okay. So what kind of stuff did you work on at that lab that you started?**

Dan: Yeah, that's where I switched. I was doing parallel VLSI and high-performance computing for Biology and things like that at Brown. Me and my grad students were doing VLSI work. The lab at Princeton was funded by a large Japanese consumer electronics company. You know, it was Matsushita, but most people would know it is Panasonic here in the U.S. and they probably weren't into high performance, parallel supercomputers and biology. It was more like the things that might affect an office environment or a consumer environment. So that's when I switched gears and started doing work in document image analysis and pin computing.

Again, interestingly, some of the same algorithmic techniques apply there as well, which I thought was pretty interesting. But document analysis is a combination of computer vision and pattern recognition. We called it pattern recognition at the time, not machine

learning, but it's basically the same techniques. Pin computing is human computer interaction. And as I said, the algorithmic methods were very, very similar as well. It was a very interesting field to be in. There was also some natural language in there as well.

**Khari: So how did those algorithmic techniques overlap?**

Dan: Well, so in the case of genetic sequences, you're looking for things that are close matches but aren't exact matches and the same thing happens when you're looking at someone's handwriting, for example. There's no way you could write the same letter exactly the same way twice, right? There's gonna be natural variation, and the same thing is true with speech. So if you look at speech or handwriting or genetic sequences or text in a language, they're all signals over an alphabet.

The alphabet, if you're talking about natural language in English, there's 26 characters in English. If you're talking about genetic sequences and DNA, it's the four nucleotides that make up DNA or RNA. If it's protein sequences, there are 20 amino acids that make up protein sequences. If you're talking about speech, the string of phonemes, but again, it's a one dimensional signal. If you're talking about handwriting, you measure it as a dynamic signal. Then it's the (X,Y) location of the pen tip over time.

All of these things are one dimensional signals and all of them you can never expect to be an exact match. You've got to sort of have tolerance for various kinds of mismatches. And that's the algorithm that I developed — you know, the parallel version when I was a PhD student.

**Khari: Can you say what kind of specific applications you were developing for Panasonic?**

Dan: One of the things I did that I think was the coolest thing...well actually there were a couple things that I did and there's a patent there as well. We did some very early work

on 2D barcodes, which I thought was pretty cool. I knew nothing about barcodes at the time.

**Khari: So is this going to the grocery store, you scan your item type of thing?**

Dan: Oh, those are 1D barcodes and we wanted to embed a lot more information. A 1D barcode is a relatively small amount. UPC codes have been around for quite a long time, but we recognized that we could maybe embed information on documents. Again, the focus there was office and consumer type devices, right? So we had this idea that we could take the information from a document and instead of just doing OCR, which is error prone, (well, it certainly was error prone at the time but has gotten better), we could actually embed that information either in a 2D barcode or actually hide it in the documents using something called steganography, which is hiding information in a way that's not perceptible to human, but a computer can recover. We also had this idea of a 2D barcode.

**Khari: How are you hiding that in the document?**

Dan: Oh, so you could actually play around with the typography. For example, the serifs, the little decorative things that hang off letters in certain fonts, you can actually play around with the serifs in a way that encodes information, but you as a reader would not even notice that the serif is missing here, which encodes a zero, this one is present here, so it can cause a one. That's one example, some of the techniques you use. You can also play around with the spacing a little bit, so that you wouldn't look at it and say, that's really odd. But encode information, hundreds of bits or thousands of bits that way.

**Khari: So then what kind of application services?**

Dan: Yeah. So we used these ideas of the barcode, the idea of embedding the encoding in the document...this is in the early days of the Web, so the idea that you could access

a document through a handle, which we now call it a URL or a UDI or something like that, was starting to come to the fore. And if you look at what a regular copier does, a regular copier makes second generation copies of something. So it's never as good as the original, right? Our idea, and I thought this is quite clever, was to embed a digital pointer to the document, either in a 2D barcode or by hiding it in the document typography, and then instead of making a regular photocopy, we actually use that to grab the original version of the document from online and print the original document.

Or you can inform the user, “Oh, by the way, there's a later version of this.” And imagine doing this just from a copier. You obviously can do it from a computer, but doing it from a copier, right? There's some patents that we had on that, and we called it the perfect copier for obvious reasons. Actually, Panasonic came out with a product based on this, which I thought was really, really cool. That's just as I was leaving the company and I was not privy to all of the commercial details, but I remember seeing on a United flight, looking at the flight magazine and there was a full page ad for the hyper documents system, which I created.

**Khari: Wow.**

Dan: And oh, by the way, there was an image of the 2D barcode, which I created and which I'm sure I still have the code that would actually read it back, as I wrote the code.

[Laughter]

**Khari: Yeah.**

Dan: I thought that was pretty cool.

**Khari: Wow. That is pretty cool. So I kind of cut you off. Could you explain a little bit more about the 2D barcode? How is that different from the 1D?**

Dan: So now we have QR codes, right? So this is before QR codes. In fact, there had been some previous 2D barcodes, but very, very few when we first did this work. The code that we came up with looks very much like what you would recognise as a QR code right now. And we had to develop a technique that was reliable for reading off a paper document, a document that had been copied or had been rotated a little bit. So you have to build robustness. It's not processed the same way as a 1D barcode is. The scanners that read UPC codes are incredibly robust, but they're dealing with a much simpler problem. So we wanted to encode hundreds or thousands of bits and as a result, you need some special design for the 2D barcode and then software that will read it. We also avoided some of the existing patents that already were present. There were a couple patents that we had to work around and we had to drop some techniques that would work around them, too.

*[Defending Against Telephone-Based Robotic Attacks - 19:01]*

**Khari: Interesting. Another thing you have a patent in is...there are a few related to methods and apparatus for defending against telephone-based robotic attacks. Can you talk about what those are? What is a telephone-based robotic attack?**

Dan: Well, I think we're all experiencing telephone-based robotic attacks right now. You're getting these junk phone calls, people trying to get you on the phone, and they can actually get through to you if you don't have ways of blocking them. So this was the adaptation of an idea that appeared a little bit earlier in the web domain called the CAPTCHA. The basic idea of a CAPTCHA is to distinguish between human users and automated users of a system. So the idea is you might develop a web service that's totally free, you don't really require registration or a credit card or anything but you want to make sure that it’s humans, that it's real people using your system, not some kind of automated bot that's taking information from you and stealing your intellectual property.

So CAPTCHAs were developed to distinguish between humans and machines by presenting a visual problem, often an OCR (optical character recognition) problem, that is hopefully easy for humans, but hard for machines to solve. The idea of this method and apparatus for defending against telephone-based robotic attacks just carries it over to the audio domain. The idea is you've got someone calling you on the phone and you want to let real humans through when they're calling you, but you don't want an automated system to get through to you. So you present the person calling you or the entity calling you with a challenge — and it's an audio challenge — that humans would be very good at solving, but that machines probably wouldn't be so good at.

**Khari: Ok, so is this like, "press one when you hear the beep or…"?**

Dan: Well, you might say, press the key below the three on the keypad. Now, that requires pretty good speech recognition, but it also requires some understanding of the real world. You have to understand the layout of the keys on a telephone keypad, right? That's the kind of intelligence that has been hard for computers up until fairly recently, so that might be an example of one of those protections, but there are a lot of other protections you could do as well.

**Khari: Okay, so these systems are still being used? Because I get a lot of spam.**

[Laughter]

Dan: I actually want to give credit for this to Jon Bentley. A lot of people listening to the podcast will realize that Jon Bentley is a very famous computer scientist and algorithm developer, one of the founders of the field of computational geometry, was on the faculty at Carnegie Mellon, then was at Avaya Labs for a very long time. And I got connected with Jon and we've done a lot of really cool things together. He's a big supporter of our department at Lehigh. And he started to develop this idea...in collaboration with him and with Henry Baird, who is another founder in the field of computing, and some others, we

came up with this idea that resulted in these patents. Now, what have they been doing with them since, that I'm not sure. It'd be really nice to have these techniques to protect us from these robotic calls that we're all getting right now.

**Khari: Yeah, it definitely seems like an area that is ripe for some new solutions.**

*[Electronic Voting - 21:56]*

**You also worked on a project related to electronic voting called the PERFECT Project.**

Dan: Yes. Electronic voting is another really interesting subject that computer sciences have gotten involved in, and some have gotten involved in a very big way. A lot has happened since the 2000 presidential election, which was controversial for what happened in Florida with respect to the punch card ballots, and then the recount in Florida, and then the rush to replace those voting systems with purely electronic voting systems, which are called DREs. And when this happened, a lot of people said, "Well, this is great. We've got computers everywhere else. Why shouldn't we be voting using computers?"

Then computer security experts said, "Whoa, whoa, whoa, wait a minute now." There's a big difference between ordering a sandwich at a local Subway sandwich shop and voting, right? We have to worry about the security of these systems. And a lot of these systems are being built using large software environments, commercial operating systems that we know have thousands and thousands of bugs in them. They're being built using hardware that could be compromised. Some of the hardware actually has network capability. A lot of it has the ability to plug in external modules and external memories. It was demonstrated numerous times that there were features in the hardware that allowed you to basically reprogram the firmware of the voting machine quite easily with just a few minutes of access to the machine.

We all know that computers are incredibly powerful machines that can do anything, including lie to you, tell you that they've recorded your votes, but actually store a different set of votes. So as a result, computer security researchers started pushing back and speaking up and saying, “We can't do it this way, this is really bad. We've got to be a lot more careful about it.” Coming from my background, obviously, I support that position; but a lot of people are saying, “For now, the safest thing that we can do is vote on paper.” And yeah, it seems archaic, seems primitive, but the fact is, voting on paper means we, as in the voter, creates an artifact that he or she has actually verified. They create this artifact, it's a physical artifact and as a result, we can scan it. We can process it the same way we read S.A.T. tests, very reliably. Then, by the way, if something does come up and we have a reason to question the results of the election, we can always go back to the original artifact, the paper that the voter marked. And as a result, have much more confidence in the result and the outcome of the election. That is a tremendously powerful premise.

I still think it's the right way to do things and I think a lot of people would agree with me. But there's still some challenges with reading paper-based ballots, right? One of the things that we should be worried about or concerned about is what are called marginal markings. For example, if the instructions say that you should completely fill in an oval, the fact is, some people aren't going to read the instructions. They're going to use an X mark. They're not going to fill in the oval or maybe they'll try to fill in the oval and they'll only get half of it. As a result, we need really powerful pattern recognition and image processing techniques to be able to deal with these situations. That’s what the PERFECT project was, to start working on some of those techniques.

**Khari: Ok. And did anything specific come out of this? Like in terms of new voting machines?**

Dan: Well, some of the methods that we developed were involved in a discussion in my local county about a month ago. The county was making a decision on voting technology. I noticed that one of the vendors was proposing a system that was actually using some things that we'd been talking about. I won't claim that we invented those ideas — you know, this was about a decade later — but those ideas are starting to permeate some areas, which is good to see.

**Khari: Yeah. On a political level, what do you think is necessary to push the idea that we need paper voting systems and not electronic voting systems?**

Dan: Well, the good news is there are a lot of people and a lot of places that are pushing this forward. There are a lot of states, and even at the federal level there are a lot of lawmakers who are understanding the case and are making the case for a switch to a verifiable, paper-based voting system. So there's a lot of laws and proposed laws that are floating around now. Different places are making different transitions. Led by the leadership in my state, we're making a move towards safer voting systems overall.

I think we were in a very weak position until recently and we're now making a transition where some of the systems, some of the counties, are actually adopting better systems. It's going to be a long process, I think. The U.S. is a very interesting place where everyone gets to make their own decision about what we use and you can't have a dictate from on high. It's probably more robust that we actually have this more distributed approach to making those decisions, but I would like to see everyone be well informed about the advantages and disadvantages and not think that paper is old fashioned.

**Khari: Right. It is interesting. It seems like, not only on a state level, a lot of voting is controlled on a county level throughout the US.**

Dan: Absolutely, yeah. It's a very interesting system we have here in the U.S. And when I give talks about this in other places — I've given talks in Europe for example — I have to explain, “By the way, you have to understand the way things work in the US, and it’s not the way it works here.”

[Laughter]

*[Outro - 00:26:53]*

**Khari: That's it for the podcast. We'll be back next week for part two of my interview with Dan. In that episode, we discuss a few of the courses he's currently teaching, the Code 8.7 Conference in using AI and computational Ssience to end modern slavery and the future of intelligent infrastructure. Until then, remember to like, subscribe, and rate us five stars on iTunes. Peace.**