Catalyzing Computing Podcast Episode 15 - Interview with Melanie Mitchell Part 1

The transcript below is lightly edited for readability. Listen to "Interview with Melanie Mitchell 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 of Catalyzing Computing, I sit down with Melanie Mitchell. Melanie recently joined the CCC Council and is a Professor of Computer Science at Portland State University, as well as an External Professor, a member of the Science Board at the Santa Fe Institute. She is the author and editor of five books and over 80 scholarly papers in the fields of artificial intelligence, cognitive science and complex systems. Her most recent book is Complexity: A Guided Tour won the 2010 Phi Beta Kappa Science Book Award.

In this episode we discuss her move to AI from physics and astronomy; getting a PhD in Computer Science at the University of Michigan; and the creation of

Copycat, a computer program that could build psychologically realistic analogies; as well as common AI fallacies. Enjoy.

[Moving to Computer Science from Physics and Astronomy - 1:13]

Khari: You're listening to Catalyzing Computing here with Melanie Mitchell. Melanie, how are you doing today?

Melanie: I'm doing very well, thanks.

Khari: So can you tell us a little bit about your background? Where did you grow up and how did you get involved with computer science?

Melanie: I grew up in Los Angeles, that's where I was born. Then I went to college at Brown University in Providence, Rhode Island, and studied physics, astronomy and mathematics there, not computer science at all. But my work in astronomy was what really got me into computer science because I learned how to program in order to do data analysis in astronomy, and that kind of led me to an interest in programming and then an interest in, specifically, artificial intelligence. That was spurred by reading Douglas Hofstadter's book Gödel, Escher, Bach, which I read back in the '80s when I was in college. It got me really interested in the question of how intelligent can we make computers? How do we get them to sort of analyze their own data, if you will?

Khari: Okay, that's interesting. I think you're actually the second person to mention getting into computer science because of reading Gödel, Escher, Bach. I think Suresh [Venkatasubramanian] also brought that up [Listen to his podcast interviews here].

Melanie: Oh, yeah. I think a lot of people were very inspired by that book.

Khari: So what aspects of computer science did you use in physics?

Melanie: Well, really a lot of statistics, and it was really quite basic data analysis that I did when I was studying astronomy. In particular, I had an internship studying variable stars — we were looking at the periods of variable stars and trying to do various data analysis techniques on those. So that's what got me into programming because I started programming my own tools to do some of these analysis, but it was a very basic kind of programming and I didn't really get into more serious programming until I started working in AI

Khari: Okay. So what specifically was the move away from physics to working in AI strictly?

Melanie: Well, to be honest, when I was growing up I felt like I had this big interest in physics, cosmology, astronomy, but once I actually started working in that field in various internships, I realized that most of it was really just computing. I wasn't looking through telescopes very much. I was looking at big data files running programs on them, and so it didn't seem as interesting to me in the actual practice of working on it than it did in theory. Then I kind of drifted over to actually studying computer science. I was one of those people who never had taken courses in computer science in college and didn't get into it until after college and then ended up going to graduate school in the field. At that time you could do that. I'm sure that's a little harder now.

Khari: Yeah, it might be more difficult. So you got your PhD from the University of Michigan?

Melanie: That's right.

Khari: Do you have any interesting stories or anything from your time spent there?

Melanie: I think Michigan was one of the first universities to offer graduate degrees or any kind of degrees in computer science, back in, I think, the 1950s, I'm not sure. So they had an interesting history in Computer Science. John Holland, who was one of my co-advisors, was their first PhD in computer science from University of Michigan. His advisor was Art Burks, who actually worked with Von Neumann a long time ago, so he was like a real pioneer of computing. But the view of Burks and Holland and some of the original computer science people at Michigan was very broad. It was like computer science kind of writ large across nature, that computing was a concept that not only took place in computers, but took place in natural systems. And we should think about information processing and computing as a kind of general framework to think about nature.

That was still in force by the time I got there. Holland taught this course called “Adaptation in Natural and Artificial Systems.” That was kind of his topic of study. I don't know if you know Holland, but he was the originator of genetic algorithms. So he was really interested in computing in a very broad sense. At the time I got there, computer science was moving from the liberal arts college to the engineering college. In fact, it moved there a couple years after I got to Michigan. So it was really in a period of flux where computer science was trying to figure out what its true home was. And there were a lot of debates about that. I think it was a real incubator of some very forward looking ideas about computer science and also sort of the site of a lot of fierce arguments about what the place of computer science was in the academic world. So it was a really interesting place to be.

Khari: Interesting. We'll come back to discuss genetic algorithms more, but I'm sort of curious. You said that computer science was sort of moving from liberal

arts to engineering. And, I think, a lot of computer science departments now are in engineering schools, if that's something the school offers.

What do you think of that as a trend? Do you think computer science should align more closely with engineering or with liberal arts?

Melanie: Yeah, that's a really interesting question. I think there's always a tension in computer science about what is computer science exactly? Is it a science? Or is it an engineering field? That's a constant question and, obviously, it can be both. I think the people who come out of computer science departments that are more aligned with sort of science and mathematics versus engineering just have a different kind of philosophical view of the field.

I'm not a big fan of carving the world up into these strict disciplines and I think computer science is a really nice example of a very interdisciplinary field in that it can be very accepting of lots of different possible ways of thinking, sort of engineering-oriented or scientific-oriented or theoretical-mathematical. In practice, sometimes that makes things difficult. In my own experience with my PhD, my committee consisted of both computer scientists and psychologists because I was sort of working in more cognitive science. The computer scientists, some of them were saying, “Why is this computer science?” And the psychologists were saying, “Why is this psychology?” It was an interesting tension. I think that tension still remains. But I see the field as becoming more and more open, kind of returning to some of its earlier inspirations from natural systems. People are getting more interested in natural inspirations for computer science. But obviously, that depends on what part of the field you come from.

Khari: Right. Yeah, it definitely seems like the interdisciplinary aspect of computer science is increasingly important.

Melanie: It absolutely started out that way, as a very interdisciplinary topic. But, you know, these academic fields tend to evolve.

Khari: Do you think computer science is sort of moving back towards being more interdisciplinary and connecting more with other fields, not just strictly from the technical side of things?

Melanie: Yeah, I think it's definitely becoming more interdisciplinary as more and more people from different fields start using the tools of computer science and getting more involved. I see it in my own students: they're much more interested in many different disciplines and see that computer science, that thinking about the world in the framework of computing is actually very valuable for many fields. I think it was Jeannette Wing who introduced the idea of "computational thinking." That's kind of a buzzword, but I think that it is a really deep idea: that you think about, not just computation and computers, but also just the whole framework of computational thinking as an important addition to the conceptual tool set in many fields.

Khari: Yeah, I think you do see this on the other side, too, like in terms of data journalism and things like that. Keith Marzullo talked a lot about that [Listen to his podcast interview here]. He's the Dean of the University of Maryland's iSchool. So that's sort of the opposite flow of interdisciplinary work.

[Copycat and AI Understanding Analogy - 10:20]

So while you're at Michigan, you did your dissertation on Copycat, correct?

Melanie: That's correct.

Khari: With Douglas Hofstadter. Did I pronounce that right?

Melanie: Hofstadter, yes. So he was the author of the book that really got me into AI, and after I read his book I sought him out and told him I wanted to work with him. So he took me on as a graduate student and I did my dissertation with him as my advisor and John Holland as my co-advisor.

Khari: Ok. Copycat is a “computer program designed to be able to discover insightful analogies and do so in a psychologically realistic way.” So what does that mean?

Melanie: Right. So I mentioned that my work was sort of at the border of computer science and cognitive science. Analogy, that is being able to look at a particular situation as being essentially the same as another possibly different situation, that's really fundamental to all of our cognition. And, obviously, if you're going to get AI systems to be able to think like humans and to be robust outside of their training regime, you're going to have to enable them to make analogies.

My work was on testing out some ideas that were originated by my advisor, Hofstadter, by building a computer model of analogy making and testing it on an idealized domain of analogies that consisted of analogies between letter strings. An example might be, take a letter string “ABC,” change it to “ABD,” now make the analogous change to “p p q q r r s s.” This sounds very simple, but actually this domain captured a lot of really interesting phenomena in human analogy making. This idea of recognizing patterns in these idealized situations and events and the salient sort of similarities is still something that's an open problem in AI. So I built a program that could solve these kinds of analogies in, what we felt was, a human-like way. And we tested it against humans—that was the psychology part of my studies where I actually did studies on humans—and found that it indeed, sort of, matched to what humans were doing to some extent. It was a very open ended problem, but it used an approach that really, I think, is still relevant. And the work I'm doing now is inspired by that work.

Khari: Ok, so when you say “similar to human.”...In the example you gave “ABC” to “ABD” and then it was “p p…”

Melanie: “p p q q r r s s.” Yes. So what's the change? What's the analogous change?

Khari: My guess would be to change the double S to double T, maybe. Moving the last group up one letter of the alphabet or something like that.

Melanie: Right. You're observing a change. “ABC” changes to “ABD,” so you describe that in some way and then you apply that description to a different string. And it's not exactly the same there's some changes, and so you have to allow some of your representation of the first set of events to adapt to the new string, to the new situation. The question is, how do you do that? My program could take in any string that changes to any other string, you could give it any possible set of three letter strings and it would try and find the analogy. It wasn't always successful, but it was general in that sense, that it tried to apply the knowledge it had in a flexible way from one situation to a new situation.

Khari: Ok. So how exactly did you do that? I know in the paper, you list three major components: the slipnet, the workspace, and the code rack. What is each one of those?

Melanie: So it would probably take a while to explain in detail, but let me just try and summarize it.

[Laughter]

So if you think about the way that you perceive a new situation, say you're looking out: your eyes move around, taking in different parts of the situation. I'll take my example of “ABC,” I notice that A and B are connected by a relationship. B is the successor of A and that's some knowledge I happen to have in my store of knowledge. Then that gives me some feedback to say, maybe that kind of relationship is relevant here. So I try to apply it elsewhere. And there's this continual feedback between what you're perceiving, what you're expecting, and things that you already know. So we tried to implement this interactive architecture where your prior knowledge is continually interacting with your perceptual process, and it's a dynamic process so you're getting feedback all the time. You're trying to build a representation of the situation that you're faced with and trying to adapt that representation to a new situation.

Contrast that with a typical neural network somebody might use to classify a particular example. Most of the time, neural networks are completely feed forward, there's no feedback. Sometimes people use recurrent neural networks where there's a little bit of feedback, but the architecture we were looking at was, I think, more sort of cognitive in the sense that there was this continual dynamic feedback between the higher level knowledge and the lower level perception. That's what allowed the system to kind of zero in very quickly on what was relevant, what the relevant similarities were and what the appropriate representation should be. That's a really short account of what we did. It would take a long time to explain the whole system. But I think that's kind of the fundamental idea is this continual dynamic feedback between higher level knowledge and lower level perception.

Khari: Okay. So you contrasted this with how most neural networks work. Do you think a system like Copycat will be valuable in newer AI systems in ways that neural networks are?

Melanie: Yeah. That's the work I'm doing now is trying to use some of those ideas, in which high level knowledge representation interacts with a lower level perceptual system based on neural networks. I do think it's going to be valuable. And I think most people in AI these days are looking for something a little bit new, that’s not just built on sort of end to end feed forward neural networks. That there has to be some more cognitive style knowledge representation that interacts with neural networks that will make it more able to deal robustly with the situations that they encounter.

Khari: Right, okay. So you sort of mentioned some of the problems that neural networks have and I watched a video of a presentation you did where you discussed four AI fallacies — I should have written those down in case you don't remember — but do you know what those were and what the problems are with those?

Melanie: Yeah, let me see if I can remember. So this is kind of a more non-technical lecture on AI that I gave. The first one was that success in narrow tasks is on a trajectory to success in general tasks. One of the things that has been true of AI throughout its history is that AI systems can do really well on specific, narrow tasks that they're trained for. You might think of things like speech recognition or playing the game of chess or recognizing objects and images. But none of these systems can do anything outside the task that they were trained for, and this is sort of the definition of narrow versus general AI. Whereas, you know, we humans, for example, can apply the knowledge that we learned in recognizing a particular kind of set of objects to recognizing them in many different kinds of situations.

Neural networks have done some amazing things, they don't yet have the generality and robustness that we humans have in our perceptual and cognitive abilities. So the question is, if we can keep training neural networks for more and more narrow tasks, can we somehow put that all together to make general AI? I was questioning that assumption that a big heap of narrow intelligences may not translate into a general intelligence. Okay, so that was one fallacy. The philosopher Hubert Dreyfus said that, if you climb a tree and say you're closer to getting to the moon, you're sadly mistaken. What you have to do is come down from the tree and get on a rocket ship.

Khari: Right.

Melanie: That may be what the state of AI is right now.

Khari: So this is where something like Copycat could come into play? Sort of giving you that broader underlying strata to understanding things in not such a narrow way?

Melanie: Yeah, exactly.

Another fallacy, that I think is more from the point of view of the public watching AI, is what I called “names confer abilities.” The idea is that we call our data sets things like “common sense reasoning” or “reading comprehension” or “understanding”; and then we have some particular benchmark for, say, reading comprehension. We say, “Oh, our system has outperformed humans in reading comprehension.”

Well, actually, the data set, even though it's called reading comprehension, really isn't about reading comprehension, it tests something else. But then what gets reported in the media is that computers are better than humans at reading comprehension. I think this is something that's been a big problem in the field, in communicating what the state of the field is, and it confuses people. So that was another fallacy.

Khari: Do you remember the last two?

Melanie: One of them was that “easy things are easy and hard things are hard.” I kind of phrased these to be provocative, but the idea here was that for a long time people in the field thought that the things that were hardest for humans would also be hardest for computers, like beating a grandmaster at chess or playing Go better than any Go master in the world. It turns out that those things we've now accomplished with computers, but we haven't accomplished all the things that are in some sense the easiest for humans. Things such as learning to speak in natural language, communicating natural language, learning to describe what we see visually — things that two or three year old children can do — are things that computers find the most challenging.

In fact, it's interesting, there's a DARPA program now that's getting a lot of funding called Foundations of Common Sense. The goal of this program is to create a machine that has the same kind of developmental abilities as an 18-month-old baby. This may be really confusing to people because, you know, haven't we gotten computers that can beat anyone at chess and Go, that can do all kinds of amazing things, can navigate us through challenging terrain, and so on? How can we be trying to create a computer that has the intelligence of an 18-month-old?

[Laughter]

Isn't that a paradox? It's an interesting paradox.

The things that are, sort of, really easy for humans tend to be surprisingly challenging for computers. There was a quote from Alan Turing, a famous AI researcher that said something like, anything that a human brain can do in less than a second will be accomplished by computers very soon. I'm quoting it wrong, but that was kind of the idea/ That was an example of the fallacy that things that are easy for us, are going to be easy for computers, and I think that's actually exactly the opposite of what's true.

Khari: Do you remember what the last fallacy was?

[Laughter]

Melanie: You're putting me on the spot here. The last fallacy… Let's see. I'd have to look it up. Sorry.

Khari: No problem. I should've written it down. But, yeah, those are all interesting points.

[Outro - 23:45]

Khari: That's it for today's podcast. I hope you enjoyed it. Tune in next week where I continue my interview with Dr. Melanie Mitchell. In that episode we discuss genetic algorithms, complexity science, and the art of writing a book. Until next time, peace.