SEI Cyber Talk (Episode 6)

What's Going on with Near-Term Quantum Computing?
by Ritwik Gupta, Daniel Justice, and Jason Larkin

Ritwik Gupta: Hey everyone. Welcome to this episode of SEI Cyber Talk. I'm Ritwik Gupta, your host. I'm a machine learning research scientist here at the Emerging Technology Center, and with me I have Justice and Jason.

Daniel Justice: Hi, how's it going?

Ritwik Gupta: Good.

Jason Larkin: What's up?

Ritwik Gupta: So we're here to talk today about quantum computing, and quantum computing-- I've heard a lot of buzz about it, and I'm sure everyone else has too, and I know you guys are working on a project that's related to quantum computing. I guess let me start with the first question. What is quantum computing?

Daniel Justice: Right. So quantum computing is a new architecture for computing that's being developed, and the idea is that we're switching out classical bits for quantum bits. And then the question is, what's a quantum bit? And a quantum bit is very similar to a classical bit but uses quantum mechanics to have an infinite number of states, and then once you measure it, it collapses down into either a zero or one, just like in classical bits. But this explosion into infinite states gives us a couple of different unique properties that we can leverage.

Ritwik Gupta: So when you say (inaudible), you mean I have like a chip and I look at it and it collapses? Or what's going on there?

Daniel Justice: Yeah, that's exactly what's happening. I mean, there's different ways of measuring a quantum bit and observing it, but when a quantum mechanical system is working and you're not observing it, it can be in many different states at once. But then once you actually do look at it and measure it using different types of techniques, it does collapse into one or the other state.

Ritwik Gupta: Gotcha. So measurement is more of an abstract thing. That's not necessarily something you physically look at-- it could be any way of kind of probing the state of this qubit.

Daniel Justice: Yes, that's exactly right.

Ritwik Gupta: I see. So why bother? We have greater computers already. I can go play Crisis at full resolution. Why should I bother with quantum computing?

Daniel Justice: Yeah, there's kind of two main-- well, the main reason totally would be that it gives up speed-up in certain situations. Not every algorithm will see a speed-up when you

convert from classical to quantum computers, but there are several ones. So for instance, prime factorization is one where we'll see potentially an exponential speed-up and it has severe implications on cryptography. And then the other one is also the amount of space certain algorithms take up. So there are certain types of quantum automata that take up far fewer bits or qubits than their classical computer architectures require.

Ritwik Gupta: I see. So basically what we're saying is-- and tell me if I get this wrong-- is for some types of algorithm, there's some sorts of problems; depending on kind of the structure of the problem, quantum computers could give me not only speed-up in terms of time, but also the space complexity that a classical algorithm would take up compared to a quantum computer. Am I right?

Daniel Justice: Yeah, that's exactly right. Yes.

Ritwik Gupta: And so you mean that you have to create special quantum algorithms that do this too, right? I can't just take, for example, I don't know, Dykstra's and just run it on a quantum computer.

Daniel Justice: Yeah. Well, for every classical computation there is a quantum analog. That quantum analog is not necessarily faster and oftentimes it can be a little slower. But for specific quantum algorithms that we create that you cannot perform on a classical computer, you will see these speed-ups. And there is a bit of a merger as to which quantum algorithms might be replicable on a classical computer, and so it's actually furthering our understanding of classical computers also.

Ritwik Gupta: I see. That seems pretty intuitive to me. Jason, I know you guys are working on some quantum computing stuff. Why don't you tell me a little bit about what you're doing and some of the core pieces that we should know about.

Jason Larkin: Yeah, so our project is focused on applying quantum computing to software verification and validation, more specifically on combinatorial optimization for NP-complete or NP-hard problems. So we're trying to leverage the kind of things that Justice was talking about to attack that problem. Combinatorial optimization actually is found in many different applications-- machine learning and obviously for software verification and validation. So that's what we're working on right now, and we're also trying to figure out if near-term quantum computers, which are so-called noisy, intermediate-scale quantum computers, which have a small number of qubits and are noisy-- so there's high error rates-- if we can actually leverage these quantum computers before we have proper quantum error correction-- which is what you would need to run Shor's algorithm, for example.

Ritwik Gupta: I see. And so Shor's is the one that does prime factorization, correct?

Jason Larkin: Yeah, yeah.

Ritwik Gupta: So basically are you telling me that near-term quantum computing and long-term quantum computing, they present fundamentally different challenges?

Jason Larkin: Yeah, yeah, definitely. Yes, so if you think about Shor's algorithm, you want to break 2048 RSA, you need that many logical qubits, but you actually need an order of magnitude more error correction qubits. So we're really far away from having that number of qubits.

Ritwik Gupta: So you say error correction a lot. I've never heard of error correction at least in our classical CPUs or anything. What's the deal?

Jason Larkin: So we actually do have error correction in classical CPUs. You can implement an error correction scheme where you, let's say, have an operational qubit, and then you have several redundant qubits to serve as error correction. But error correction is not so critical with classical computers. We kind of have that under control. Maybe if you're in some environment which needs it more, you'll have it. But in quantum computers, we have lots of sources of errors that affect qubits. So interaction of qubits with each other, with the environment-- and this means that we have high error rates, at least right now, and error correction utilizes extra qubits and redundancy to perform this error correction, and there are schemes that eliminate it. Essentially they can push the error rates down so far that you can do a computation for sort of an extended period of time-- minutes, hours, etcetera.

Ritwik Gupta: I see. And basically I guess what you're saying is the reason why this is really relevant for near-term quantum computing is as we have really noisy chips and noisy quantum computing-- are you saying that we can't do all these fancy things that people have been promising us, like 2048-bit RSA breaking, because we're limited by the amount of error-correcting qubits we have?

Jason Larkin: Yep.

Ritwik Gupta: I see.

Jason Larkin: That's not the only factor, but that's one of the major ones. Yep.

Ritwik Gupta: That's interesting, because I always thought that now that we have a quantum computer-- and a lot of companies do-- I just thought that, "Oh, we can do all this stuff now." But I guess that's not true.

Jason Larkin: No.

Ritwik Gupta: So what can we do with near-term quantum computers then?

Daniel Justice: Yeah, so right now there isn't a whole-- well, we're starting to get bigger. As our quantum computers are starting to push up towards 50 qubits and 100 qubits, we're exponentially increasing what we can actually model and what we can do, and one of the really low-hanging fruits right now is molecular simulation.

Ritwik Gupta: You mean like chemical molecules, right?

Daniel Justice: Yeah, yeah. So we can accurately represent these molecules. We can see all the different states, and there's a couple of different simulations you can run off of that, and as you start getting into only slightly larger molecules, such as maybe caffeine, it becomes extremely difficult for a classical computer, even a supercomputer, to accurately represent these molecules. But as we start getting just maybe a couple hundred more qubits, this should definitely be accessible. Again, it's about the logical qubits versus the qubits needed for error correction.

Ritwik Gupta: So actually, that brings me to another point, is I always hear about these things that Google or someone has created a chip that has 72 qubits or something like that. When they say that, are they saying that there's 72 qubits that do actual computation, or does that include the amount of error correction qubits that they need? What's the deal there?

Jason Larkin: So I think the Bristlecone of the 72 qubits-- there are some number which are there for error correction. So I think the Bristlecone has error correction qubits built in. Most other companies are after using all qubits for logical qubits and they're more interested in finding algorithms which are able to tolerate the noise. That is, use all qubits for logical and figure out some other way of dealing with the noise.

Ritwik Gupta: Let's say I have a quantum chip. I can do some quantum computation on it. Where do I start? How do I write an algorithm? What do I do?

Jason Larkin: I mean, you have to go back, like Justice was saying about designing quantum algorithms which do some computation which has a classical analog, and then you have to-- so there's a software stack in quantum computing much like early days of integrated circuit computing, and that stack is not well developed yet. So you've got to-- most people have to start at the application layer and take that problem all the way down to bare metal, so to speak, and that process is very difficult right now. So there's a lot of development going into exactly how you do that and how you do it optimally. So when you're programming a quantum computer, there's sort of a limited amount of programmability at an application layer that would be

accessible to data scientists and engineers. But to really utilize them now, you have to be sort of a domain expert, and that has to change.

Daniel Justice: Yeah, they're definitely working towards changing that right now, because as Jason was saying, every single quantum computer, even within the same companies that are created multiple quantum computer chips, have completely different architectures. There's different connectivity between each of the qubits and this really matters on how you-- what kind of computations you can have and the efficiency of them, and each basically company that creates a quantum computer and then makes it accessible to the public, they will abstract a way that compile a transpiler from normally Python. There are other programming languages that they'll use, but it's normally Python. They'll abstract a way. A transpiler takes Python code down into a format that they can actually run on their quantum computers, which is normally accessible via the cloud.

Ritwik Gupta: So are you telling me that I could today launch my cloud editor or whatever and just kind of write a program? Like I could just use for-loops and everything, or is there a specific way I have to write a program? I've heard of things like quantum circuits and-- there are all these things that I've heard. Can I just write a program, is what I'm asking?

Jason Larkin: Sure you could, but you'd have to understand the way quantum circuits work. So I guess we could say that the abstraction layer, which is above quantum circuits at the application layer-- there's a limited number of applications which are accessible by data scientists and engineers. If you actually want to program a quantum computer, you have to understand at the circuit level, and that means understanding at the quantum computation level.

Ritwik Gupta: So is it fair for me to say that if I were to bring quantum computing to an analogous classical setting, and if I as a data scientist wanted to use that, I would basically be writing their log for ML? Is that an equivalent for quantum computing of where we're at now with the quantum circuit models?

Jason Larkin: I mean, it depends on if someone has implemented the available quantum algorithms that you're looking at. If they haven't, then you'd have to figure out how to implement it, or you'd have to go to the literature and implement it yourself.

Ritwik Gupta: I see.

Jason Larkin: Yeah, the whole field is in its infancy.

Daniel Justice: Yeah, I would say that the future, at least near-term-ish, should kind of see quantum computers as an alternative to GPU, where instead of-- once you reach an algorithm that's very difficult to do in the first place on a classical computer, right now we would go and

use a GPU to give us a speed-up, and that works for most algorithms. But then if we even have a better speed-up using that quantum computer, you might call that as a subroutine and then get the information back and continue.

Ritwik Gupta: So as far as I know, for my (inaudible) background a little bit, is GPUs are this SIMD architecture, which are really good for matrix operations. Are you saying that quantum computers can do matrix operations really well? Or are you just using an analogy that GPUs provide speed-up for certain tasks and quantum computers can also serve as this kind of co-processor basically?

Jason Larkin: Yep. Accelerator, co-processor.

Daniel Justice: That's a good way of looking at it.

Ritwik Gupta: So are you guys saying that I will never have a quantum computer-- like I can't just log into a quantum computer, right? It's going to be like a chip that's kind of sitting on the side that I can use?

Jason Larkin: So what would you mean by logging into the quantum computer?

Ritwik Gupta: Like it's not like I could just come into work and log in on Active Directory in Windows that's running on a quantum computer. It's more like a thing that--

Jason Larkin: It's like a co-processor and accelerator, and you would submit a job to that QPU. It would compute, it would send back the result, and you could read that on a server that's connected to it.

Ritwik Gupta: I see. And so long-term, what are the goals? I understand near-term, everything's noisy. I understand that there's some constraints with we have to keep it near absolute zero or really low temperatures. What are the long-term goals for quantum computing?

Jason Larkin: Well, the one definite long-term goal is to get enough qubits and enough physical qubits so that you can run error correction, pushing the coherence times to longer time periods that'll run larger problems to execute Shor's algorithm, for example. And the sort of timeline that most people feel for that to happen is about 10 years, maybe 10 to 20 years. So beyond that, I think-- I mean, there's a number of alternative quantum computing architectures, like topological quantum computing that Microsoft is working on. There are other models. There's a whole bunch of stuff in the academic literature that's coming up. So it's a lot like in the 1950s, when nobody knew that the transistor was going to emerge as the dominant medium of computation, so I think it's a lot like that. So it's hard to say what's going to happen in 10, 20

plus years. But I think everybody is aiming at being able to execute Shor's. That's the one thing for sure everyone's aiming for.

Ritwik Gupta: Shor, for sure.

Jason Larkin: For sure.

Ritwik Gupta: This is, again, very interesting. I can't say I understand all of it, but if I wanted to go and learn more about that, do you guys have any resources for the audience to go read up on this stuff?

Daniel Justice: Yeah. So we're starting to compile on our own quantum hub, which we'll link the link to that. We're starting to compile a corpus of materials that you can look at. It'll include academic papers, it'll include YouTube videos, and books, and anything else that we find helpful-- maybe some tutorials to get you started. I definitely suggest a great way to get an intro is to follow some of the main academics in the area, especially Seth Lloyd has some interesting stuff on YouTube, along with Ryan O'Donnell.

Ritwik Gupta: Cool. Jason, any resources for you?

Jason Larkin: Researchers? Oh jeez, there's a whole bunch. Scott Aaronson, Seth Lloyd, John Preskill at Caltech. I forget the guy's name at UC Santa Barbara, but he works at Google. Eddie Farhi from MIT is also at Google now.

Daniel Justice: Yeah, Farhi's very good.

Jason Larkin: I could keep going on. It's better to put a list.

Ritwik Gupta: So quantum hub is what I heard, and that's something that you guys host, so we'll put a link to that in the description and on the screen as well, and then all these academic papers, which we can link to as well.

Daniel Justice: Mm-hmm. Yeah, exactly.

Ritwik Gupta: Cool. That's really helpful. Again guys, if you guys want more information on the work that we're doing, we'll again link them in the description. If you guys have any specific questions that we can answer, please just email us at info@sei.cmu.edu. Jason, Justice, thank you guys so much. It's very informative and hopefully we hear back from the audience really soon.

Daniel Justice: Thanks very much.

Ritwik Gupta: Thank you.

Jason Larkin: Thank you, man.

Related Resources

https://quantum.etchub.xyz
https://www.amazon.com/Quantum-Computer-Science-David-Mermin/dp/0521876583/
https://www.amazon.com/gp/product/0521879965/
https://www.amazon.com/Quantum-Computing-Introduction-Engineering-Computation/dp/0262526670
https://www.youtube.com/watch?v=Z1uoz_8dLH0&list=PL74Rel4IAsETUwZS_Se_P-fSEyEVQwni7&index=1
https://www.youtube.com/watch?v=5xW49CzjhgI