EPISODE 387

[INTRODUCTION]

[0:00:01.1] JM: Quantum computing is based on the system of quantum mechanics. In quantum computing we perform operations over qubits instead of bits. A qubit is a vector which can take on many more values than zero or one. The technology used to implement quantum computers is advancing such that it has its own Moore's law, but it can also leverage the classical advancements of Moore's law. If classical computing advances at the exponential pace of two to the N, quantum computing advances at the pace of two to the two to the N.

Quantum computing will advance technologies in ways that will take us by surprise. If things feel like they are moving fast now, just wait until developers have access to quantum processing units. Machine learning, simulated chemical synthesis and NP complete problems are ripe for quantum computers.

Vijay Pande is a partner at Andreessen Horowitz and a board member at Rigetti Computing, a quantum computer company. In this episode, we explored what software engineers today need to know about quantum computers and some of the application domains that developers will be working on as quantum computers become available. I enjoyed this episode. I certainly plan to do many more shows on quantum computing because I barely understand it.

[SPONSOR MESSAGE]

[0:01:31.8] JM: Developers love Heroku, it's the fastest way to deploy an application on scalable infrastructure. I know this from experience as I have built more software products on Heroku than I can remember. When a project I am building with Heroku starts to get more serious, I use Heroku Flow together with Heroku CI. Heroku Flow is a full continuous delivery solution for Heroku. With Heroku Flow, a push to GitHub spins up an instance of my application called a review app. I contest the new instance, the review app, and discuss it with my team and if it works I can approve the changes and push it to master where the tests will run through Heroku CI before being pushed to production. Heroku Flow brings the developer experience of Heroku together with continuous delivery.

Check it out now at Heroku.com/flow. You can also check out the interview that we did with Andy Appleton from Heroku about Heroku Flow and continuous delivery. If you want to try a simple platform for deploying scalable apps with continuous delivery check out Heroku Flow at Heroku.com/flow.

[INTERVIEW]

[0:02:49.6] JM: Vijay Pande is a partner at Andreessen Horowitz. Vijay, welcome to Software Engineering Daily.

[0:02:54.3] VP: Hi. Great to be here.

[0:02:56.1] JM: Today we're going to talk about quantum computing, and I want to start off by giving some people a soft introduction because I don't think we can make too many assumptions about the audience. What do software engineers who don't have any background in physics need to know in order to understand quantum computing?

[0:03:16.3] VP: Quantum computing is a fundamentally different paradigm. You think about the paradigm shifts we've had to deal with, like going from a regular von Neumann CPU to, let's say, message passing, or threads, or even GPUs, or TPUs. Each one of those are major changes where you have to take algorithms and I think usually the secret to success in those generational shifts is not just taking the exact same algorithm but really rethinking the algorithm. At times, rethinking the problem. In doing so, that's how you really can make huge gains.

In my case, with folding in home the distributed computing project that I led at Stanford, we had to really completely rethink the algorithm because we went from running on massively parallel HPC, high performance computing to millions of computers connected by the internet. It's a very different type of paradigm.

Here, it's different still, because we call it quantum computing. I think to really reset your thinking about this, what is really happening is more like a quantum experiment, like a physics experiment that's being run that you can automate and set up and take advantage of. Much like

— I don't know if any of you have ever sort of seen analogue computing and the idea there that it's sort of just physics is doing the calculation. The equation is how can you be clever enough to take advantage of this?

The beauty unlike analog computing is that so much of the paradigms for regular computing come over, but if you start thinking about this as a quantum experiment, it will sort of set your expectations for how different this will be and for the fun and the challenge of doing this completely different type of computing.

[0:04:55.8] JM: When you say analog computing, are you talking about when you go from doing computing where at the lowest level it's represented by zeroes and ones to a place where you can represent a range of values for every slot that would previously have been represented as a zero or a one?

[0:05:13.3] VP: Yeah, exactly. In analog computing, you could build electrical circuits that do addition and multiplication just by the laws of physics of circuits and you're not having a multiplier or an adder in terms of digital stuff. Analog computing just doesn't scale and so it's a very different type of beast. In that sense, yeah, it's like you're not using zeroes and ones. You're using something else.

[0:05:36.6] JM: When I talk to your partner, Chris Dickson, we were discussing a little bit about this quantum computing stuff and he suggested that a logical language other than Boolean algebra might soon become useful. A lot of our computation is based on Boolean algebra right now because at the lowest level we're doing stuff with ones and zeroes. There were some transistor technology, I think, a long time ago that you could have three different — It was ternary logic. You could have zero, one or two, because at the lowest level, transistors — The abstraction that I understand, transistors, they operate off of either a high voltage state or a low voltage state, and that's the one or the zero, but those are ranges that are just fabricated and you could just say, "Okay. Now you have a low, medium and high voltage," and then you could essentially have a wider language of expressivity with our current model. Do you have any thoughts on that history or do you have any perspective on those experiments in the non-binary transistor space of the past?

[0:06:43.3] VP: Yeah. I think there's this really interesting situation where I would tie it into Moore's law in general, which is that — Moore's law has died so many times. The first death of Moore's law is, well, we can't make individual processor cores much faster. We're going to have to have more processor cores. Some people felt that was the death of Moore's law. If you think about Moore's law as just doubling the number of transistors. Moore's law hasn't died at all. We're still getting smaller, smaller transistors. Everything is just moving along. No death there.

The next we'll say is, "Eventually —" Actually, it's amazing how far we've come in terms of just how many few nanometers these transistors are. Eventually, transistors can't be smaller than atoms and we're getting close to that. In a few generations, 10, 20 years, probably more like 20, we will get to this physical limit. Then there will be those people who'll say, "Well, then Moore's law of course is dead," but then there's going to be three dimensional transistors and that will allow increasing of transistors. I suspect that there will be tricks along the lines of what you're talking about that continue to extend our compute power. Transistors can get smaller but they could get more capable.

If you could handle four things instead of two, like zeroes and ones, you have zero, one, two, three. Obviously, that would be a tremendous addition. Also, I think given all the excitement about deep learning and neural nets, it seems natural for a revival of analog computing coming in where instead of dealing with these 8-bit floats like you would on a modern GPU or GPU or so on, an analog computing would easily be able to handle that with the precision you need and do it with many fewer transistors and fewer FBU's and all these things. I think there's going to be a really interesting trend that computer engineers have been so resourceful, and that resourcefulness I think is not going to end for a long time.

[0:08:31.8] JM: I want to give people more color on this analog computing, and I think what you mean by that is that — The notion of quantum computing is going from bits, which are ones and zeroes to qubits, which are basically vectors. A qubit represents a vector rather than an on or off state. We have a wider range of values that can be represented in the same space that in the previous paradigm might have been represented by a zero or a one. Is that correct?

[0:09:01.8] VP: Yes. That's true of quantum computing. I think that's not necessarily a thing most people lead with in quantum computing because the really cool thing is a superposition

which I'm sure we'll get to. The fact that you can do all these calculations in parallel. That is a nice extra twist that you get a more bang for your buck even per qubit versus per regular bit.

[0:09:21.5] JM: Okay. Talk about superposition.

[0:09:23.7] VP: Yes. This is the sort of kind of crazy and exciting thing about quantum computing, the heart of it. The heart of it is that if you have, let's say, n-bits, let's just say 32-bit quantum computer which his relatively small one compared to the designs that people have been throwing around. What you can do is that you can do calculations that do operate on all 32-bit simultaneously. Let's say you need to search through something that had a space of 32-bit. The number of possibilities is 2 to the 32 power, something which is about roughly about 4 billion.

Instead of having to just cycle through each one of them, the quantum computer in principle can do that in one operation instead of 2 to the 32 operations. To make analogs regular computing, the clock speed of a quantum computer is going to be a lot slower, but the number of operations that you can do per tick is dramatic.

The real kicker is this, is that — With n-bits, you can do 2 to the n. Now the question is like, "How many bits are we going to have?" The answer right now is what people have published is like 8-bits, but people are talking about 32-bits and 64-bits coming relatively soon. The question now is, "How is this number of bits going to change?"

With the technologies that people are using right now, which is just boot strapping on top of Silicon chips and so on, the number of qubits increases like Moore's law. To keep them as simple, like roughly doubling every year. If you had 8 now, next year it'd be 16, and 32, then 64 and so on. The power of the quantum computer scales like 2 to the number of qubits. This quantum Moore's law is kind of mind-blowing, and that the number or qubits goes like 2 to the n where n is, like I've said, number of years from now. The power of the machine is to the 2 to the number qubits. It's 2 to the 2 to the n. This is going to catch people by surprise, because we're used to the cadence of Moore's law which is just 2 to the n and we're used to speed-ups and you get a thousand times over 10 years roughly, or something like that. This is something where quantum computing at first will seem like a toy. You'll say like, "32 qubits. My GPU can beat that

easily." Then, "64 qubits. My GPU can beat that easily." 120, 256, maybe it's 200 qubits. That's the killer. You go from 120 to 256 in one generation. Suddenly, the quantum computer goes from a toy to devastating.

[0:11:41.7] JM: Moore's law is not exactly a scientific law. It's more like we had a bunch of — I think Gordon Moore just kind of like looked in the past and he's like, "Okay. Here's kind of some evidence that the trends are going to work this way," then he sort of stated it. Then whether it was just a fact of human nature, this is how human nature advances, or if it was the fact that by codifying into what he called law, he kind of encouraged the market to all operate in lockstep and they did all operate in lockstep which made the market operate against Moore's law. It wasn't exactly a law of nature if I remember it correctly. Is this the same case with the qubit relation to Moore's law that you're explaining, or is there actually like a scientific basis for why we would get those kinds of gains overtime?

[0:12:34.7] VP: Yeah. You're actually right, that this is more an observation and a power of human will. In the quantum case, we are already just boot strapping off of existing Silicon technology so that you don't actually have to do anything all that new. The size of the gates and so on are huge compared to transistors. That part is kind of where it seems to be guaranteed in terms of just the number of gates.

The good news there is that's not something that has to be proven out necessarily. You have to do other things. You have these gates to have this sense of coherence which itself is a challenge, but that's not a challenge in sort of Silicon process engineering necessarily.

The part though that is just fun to think about is like why has Moore's law been a sort of an empirical fact. Let's take it away from a law. An empirical fact over so many decades. Part of it is a testament to a human will and engineering and market forces and so on. A part of it is that there are just certain technologies that have this capability of the exponential increase. If you can just make some percent better of its original. If you can make it 10% better or 50% better. That alone gets you the exponential — That's just math, if something increases in capability by a faction of itself. There are a lot of technologies that work this way. Computers work this way. Actually, things that have nothing to do with computers work this way. Genome sequencing, the cost of genome sequencing was going like Moore's law until it stopped. Actually, it stopped

because it's now going faster than Moore's law. It's a testament to the fact that, often, things can get better by a fraction of it rather than just adding something.

If your bank account increased by 30% every year, your bank account will grow exponentially. If your account increased by $30 a year, it would grow linearly. It's that kind of math where something can increase by a fraction. Its existing capability is what makes this possible.

[0:14:25.6] JM: I'm still trying to wrap my head around this topic by doing a fair amount of research on it and I'm trying to figure out to what degree I can draw a mapping between traditional — Computing as I've known it since I kind of started looking into programming, computer science and quantum computing. I'm trying to figure out where the analogies breakdown and where they're supported. If every qubit is a vector, it has a range — It has a wide range of values instead of a logical zero or one, like a bit, can we put the entire value of an opt code into a single qubit? Because an idea of an opt code is this kind of this instruction. Is that an analogy that would make sense?

[0:15:12.0] VP: Kind of. I think the qubits are more like the registers, and so you don't put an opt code into register, obviously. You have something else that's reading the instructions and then operating on your registers. This would be almost like your accumulator in a CPU.

If you think of it that way, it's a lot like another accelerator. It wasn't that long ago, the first computers I programmed didn't even have FPUs, right? You have to have this separate chip, like an 80, 87 as your FPU and it works seamlessly, but then things like GPUs and so on are not nearly as seamless. This is something else where it's just sitting, connected to your computer. I think the future of things, at least for a while, are going to be these hybrid machine where you have a GPU on your classic computer, you'll have a QPU on your class computer. You're going to offload stuff to it. The qubits are your accumulator and then you send instructions that are the opt codes that operate on these qubits. Doing various physics things on it, but it's the equivalent of going through opt codes.

[SPONSOR MESSAGE]

[0:16:18.0] JM: Your enterprise wants to adopt containers but you aren't sure how. CoreOS will help you along your journey to a containerized architecture. CoreOS are the container experts trusted by Salesforce, eBay, Ticketmaster and other world-class organizations. Go to softwareengineeringdaily.com/coreos to find our top five episodes about containers and Kubernetes as well as a whitepaper about migrating an enterprise to Kubernetes with CoreOS. They've hosted, attended and spoken at many shows about containers and Kubernetes because those technologies are the future of the web. That's why CoreOS built Tectonic, an enterprise-ready Kubernetes platform. At softwareengineeringdaily.com/coreos you can learn about how containers can make your organization run more efficiently.

Thanks to CoreOS for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:17:26.3] JM: The things that makes sense to offload to the quantum processor on your computer, what are those going to be?

[0:17:34.6] VP: Yeah. Anything where your algorithm really can be massively scaled by doing a lot of things in parallel. Anything right now that's NP-complete that fits in the number of qubits, that's the first-off no-brainer. If you had an order and algorithm, it's generally probably not the case that a quantum computer will speed it up much, but order and algorithms are rare and already quite fast and probably the same for even an analog n.

If you have an analog n algorithm, I doubt it will be all that useful. Anything either exponential. Exponential is where the obvious wins are. We tend to — The finding is we tend to, as a program, you will just not even think about exponential algorithms because that would be the dumb solution usually to things. There are certain problems we just can't event tackle because we just reject anything that's NP-complete as being just un-scalable.

What this does is it opens the door to NP-complete being the standard way to approach a lot of things. That's the easy one. You don't have to be NP-complete to be a pain in the ass. You could be n to the 8^{th} or n to the 6^{th} and that still is bad enough, large enough n that this is worth doing. That's, I think, a good place to start. There are many type of algorithms that people have

been proposing that are classic computer science algorithms that would be interesting to bring over in machine learning, in cryptography, in all these things. We can talk about them in more depth, but I think a lot of the first things that will come about will be to use quantum computing to calculate quantum mechanics, which is itself, if you do it right, is NP-complete and a mess and this thing could just do it naturally because it's quantum in its very own nature.

[0:19:10.3] JM: Why wouldn't we just have the QPU replace all of our processors, because all the computation that I work with, like I go on to my Twitter feed, or I open up Searchlight on my Mac and the recommendations that these systems provide to me, they're great, but I'm sure they would be better if they were kind of going in with the assumption that Napsac was solvable or that best match, all these different NP-complete problems that you learn about in your algorithms courses. Why wouldn't QPUs replace everything?

[0:19:51.6] VP: I think if you think about modern computing, it's typically going to be a mash-up of many different types of devices. Just take a look at a phone. A phone is a regular CPU and a GPU, sometimes a DSP. All these different things thrown in together. For the phones, it's obvious that you'd want to do this because of just power consumption that's going to handle that.

Even if power was an issue, there's reasons why our laptops or our desktops have CPUs and GPUs. There are just some things CPUs will always do better, and so there is that. Finally, on top of that, when I said this is a physics experiment, a quantum experiment, these things are hard core at last at the moment. They are run at 4 milli-Kelvin. One Kelvin is already pretty freaking cold. Freezing is 273 Kelvin. What 4 milli-Kelvin is you need liquid helium, liquid nitrogen which is about as cold as most people ever see. It's 73, 77 Kelvin. It's something where — This is not something you would have in your garage, like a 4 milli-Kelvin fridge. It's something akin to the early days of [inaudible 0:20:57.0] where you would have it somewhere safe.

Nowadays, maybe the better analogy is the cloud. You typically would have racks of cloud servers in your garage. You could, but the heating-cooling would be ridiculous. Why would you bother? Similarly, quantum cloud is just obvious given how we do computing today and would naturally slide into that.

[0:21:18.5] JM: Yeah. There's another show I did with your partner, Peter Levine. We talked about this idea that he kind of talked about where you get the cloud moves to the edge because of cars and drones. I wonder if, to decrease latency, you could have a drone that manages to have the right temperature properties floating around so you have lower latency to hit the drone and get the answer to your NP-complete problem.

[0:21:48.8] VP: Yup. Peter is very prescient in his observation there. I think something that we've all seen is this back and forth. People had main frames and many computers allowed departments to have their own vexes, so we didn't have to share the government, sort of the company mainframe where the sort of the university mainframe. Everyone had their vexes, but then they became centralized, so people got workstations and they became — There's this back and forth of centralization-decentralization that I've seen over and over and over again. The cycles maybe are 15 years or something like that.

Right now, the cloud is king and I think Peter is most likely I think. I bet he's spot on with the edge is coming up especially for power and latency on these things. I think it seems pretty safe to say the pendulum will eventually swing back because you just won't have a quantum processor on your drone as you talked about or at least not for a while. That will open up interesting possibilities.

Actually, I'll just leave a one thing which is cute. I think that the temperate in space. It's pretty cold in space. I think the cosmic — My career background is something on that order. 4 milli-Kelvin might be still a little warm for space. We should Google this right now. There will be really interesting applications in other places. Maybe not on your drone, but on your satellite is not crazy.

[0:23:04.7] JM: Interesting. Let's talk a little bit more about how to build a quantum computer. Can you just give an overview for what the necessities to — Obviously, okay, you talked about cold temperate, but just give an overview of what components go into a quantum computer.

[0:23:22.1] VP: Yeah. Quantum computing, the idea of it has been around for multiple decades. As a remember as a grad student, I was at MIT, that's when Schur's lemma came and everyone got excited. This was early 90's. That's a long time ago.

The problem has been how to build this coherent quantum computer with multiple qubits. This process of where you can sort of do 2 to the n only works when these qubits are coherent, where they're in one quantum state. There are machines that have lots of qubits but they are not fully coherent. The challenge in building this is how can you build it such that the [inaudible 0:23:59.1] isn't lost. It always will get a eventually lost, but isn't lost so long that you can't do some calculations while still coherent.

This requires — Those successful teams I think will have a full stack approach where it's not just the quantum software engineers and not just the chip designers, it's the fridge. It's getting signals in an out of the quantum computer. It's building the interface between the class computers and quantum computer. It's understanding the process engineering. It's making the chips. That full stack approach is going to be more relevant here than in other places, although if you look at how Apple built the iPhone, they are from chips to UI. That's not unprecedented, but I think this is a place where it may be really critical.

[0:24:42.8] JM: How are the different corporate players, like Google and IBM and Rigetti, who you just invested in, how are these different players attacking the problem differently?

[0:24:54.2] VP: For disclosure. I'm on the board of directors for Rigetti. We invest in them. Rigetti is a startup. Google and Microsoft used to be startups. They are certainly no longer startups, and IBM as well are all in this mix. It's very exciting and heartening that these A+ players view this space as being sort of critically important. I think it reflects that this is a special time. We'll see lots of different approaches. If you look at what Microsoft has publicly talked about what they're doing. They have interest on software and some interesting ideas on hardware. IBM and Google have been pushing hardware and will do software. Rigetti really wants to do this full stack approach where they go all the way from chips through computers, through software.

The full stack approach with rapid iteration, specially rapid iteration that can some ways be best facilitate by a startup I think will have huge advantages. It's still hard to do things in startups. Startups are the hardest companies to build and run and so there's not going to be without challenges. I suspect there'll be pros and cons of each approach.

Finally, much like computers, it's not like the world was dominated by one computer maker. I think there will be many different players in this ecosystem as it evolves. Especially, maybe the biggest difference is that in computers you had primarily one architecture, like the 8086 versus the 6502 versus the Z80. They had different instruction sets, but they weren't radically different in terms of the thought process for programming them. It was just how many registers did you have. How many bits and so on.

These thing will be a bit more different. Having the diversity of players, it will be super interesting to see how these all shakes out and to see which designs are most flexible versus most parsimonious and cost efficient. There'll be probably certain applications that will need one type of machine versus another type of machine. With all these players involved it seems guarantee that something great will happen. I think it still remains to be seen how it all shakes out.

[0:26:49.2] JM: Is there a good answer to the why now question? Has something changed in supply chain economics or scientific discovery that has made quantum computers now buildable despite the fact that we've known that this technology could exist or could be buildable for a while?

[0:27:06.9] VP: Yeah, I think there are several things. This is, by the way, the hardest thing to get right. It may not be now, but I'll give you evidence for the case for why it could be now. Part of what the key shifts that we always look for is shifts for something going from science engineering. You can't schedule science. Einstein didn't have on his calendar, "Today, I'll come up with relatively." You don't have a roadmap for these things. It's a very different type of skill. It's about discovery. It's about creativity and so on. With engineering, it's not guaranteed. It's extremely difficult and engineering has its own issues.

Engineering is a little much more roadmapable and it's something where we know how to scale engineering in ways that you just can't even scale science. At the heart of the shift that we're seeing now is this shift from science to engineering. Especially engineering that can take advantage of process advantages in other spaces. The fact that you can boot strap on top of what people do for building chips, that's a pretty dramatic thing. A combination of things like that where the challenges are engineering challenges, building on technologies which are already fairly established, it doesn't mean it's guaranteed by any means, but it certainly changes the style from, fingers crossed, maybe we'll figure out something scientifically to we have to engineer something. The way we've engineered other things in the past, not everything works, but it's a different game.

[0:28:32.5] JM: I don't know how much Rigetti has talked about this publicly, but are there any strategies that you've seen for how the company is aligning itself management-wise, or is it finding out what kinds of teams they need. Do we need a software testing system? Do we need — I don't know. Is it just like building Intel, for example? Are there historical analogs that you can look at for how to construct the team and how to manage engineering?

[0:29:02.5] VP: I think the challenge there is that you have — With their full stack approach, they have to have a team doing fab. They have to have a team doing chips. They have to have a team building the computers. They have to have a team programming the infrastructure code. Ideally, some people helping with examples and application code. That full, full, full stack since is a huge challenge from a management perspective, but that is the opportunity. That's the sort of the intriguing part of it.

The upside is that of doing it this way, obviously, is that you can have advances that you come out on the software side influenced how do you design the architecture of the chips. Process engineering advances can affect how you build a chip and so on. All of these things is really sort of puts us — Puts Rigetti in a very different place. We'll see how it shakes out. That's one of the things the guys are very excited about that.

[0:29:57.7] JM: Rigetti built this thing called Forest, which is I think it's a tool for bridging the gap between classical computing and quantum computing. Explain what Forest is and why it's useful.

[0:30:13.6] VP: Yeah. Forest is the name for a suite of different software tools. There is Quil and Pyquil, which is the programming language. There's grove which is a repository. There's multiple parts in it. I think the part that's especially important to start off with is Quil. Me loving Python, Pyquil is a very natural one. The fact that you can program a quantum computer in Python, especially quantum computer hybrid and interfaced everything else in Python should hopefully facilitate the rapid development of new codes.

Another key part of forest is the quantum virtual machine. You might not have access to a quantum computer right. There are only a lucky few people on the planet that do. Everybody can have access to the QVM can start coding right now for code that will run on a quantum computer. It's a unique opportunity. You can manage it like the early days of homebrew computing with the first PCs and Bill Gates coming up with basic interpreters and operating systems and so on. He had to have the hardware in his hands to be able to program this. This is somewhere where the next kid, the next quantum Bill Gates out there doesn't need to buy one or have his parents buy him one. It's something where you just need to access to the software, is you need a laptop and an internet and you can start rolling.

That's, I think, part of the real opportunity. Enrolling for quantum computing to run on Rigetti. In principle, Quil can run on everything. It could run on any quantum computer. It's opening the door into this new world.

[0:31:41.4] JM: Do you think that's important to work on right now, the higher level interfaces for it? To me, I look at this, and I'm like, "It seems like if it's still so early, why wouldn't you just kind of put all your chips on developing the hardware and the full-stack?" Isn't it too early to start developing the higher level interface?

[0:32:06.2] VP: Higher level interfaces are going to be important because it's easy to do, unlike in the early days of processors. It's easy to build something like Pyquil, and the ability to abstract on top of it would really greatly facilitate the building applications. In this way it's actually different in the early days of personal computing, because you don't have all the infrastructure we already have to build on top of computing. Also, if you go back to sort of what we're talking about in terms of how quantum computing will catch people by surprise, that at first

it will be ridiculously slow compared to the classical one, and then within one generation it goes from ridiculously slow to killer applications to feeding all.

With that in mind, you want to be programming this before that generational shift happens. Ideally, even starting to think about when those generation shifts happen, because it's not going to be the same for every application. For some applications, maybe 200 qubits will be the point where we have what we call quantum intercept, where the quantum computer would surpass the classical one. For some applications, it might be a thousand. For some it might be 10,000. Understanding where that is is something that we want to figure out now and the teams that get out ahead of this and have codes ready for that next generation. They're the ones that are going to likely become the big winners rather than the ones that wait until the machine has passed quantum intercept and then start deciding it's time to learn about quantum computing.

[0:33:34.5] JM: You have a background in biology and you talked about your experience working on folding at home. What excited you about the intersection of biology and quantum computing?

[0:33:47.5] VP: This could be a podcast at its own rate and so I'll try go — I should mention by the way that my undergrad and graduate degrees are in physics. My first intellectual love is intersection between physics and computing. In this sense, this is like right down the line for that. Also, my appointment at Stanford is in chemistry, so something where I've spent a lot of time around chemistry and biology and physics. That's where this gets very interesting because it's a beautiful synergy of all three. It's a physics calculation. It's using quantum mechanics. Where I suspect the first applications will be is in the chemistry of biological molecules.

I can unwrap that. Biological molecules do all these amazing things. The most dramatic things they might do is act as enzymes. Enzymes catalyze reactions that things that on its own without the protein, the enzyme protein doing something. It could literally take a million years for this to just happen just by the laws of physics. The protein can catalyze it such that this happens in seconds or milliseconds.

That's a pretty amazing thing and we would love to harness that type of power. Catalysts are used in engineering applications all over the place in chemical engineering, but it's very difficult

to design enzymes the way evolution has and the way biology has. If we can simulate them — Especially, simulate them with sufficient accuracy, because we can simulate them now but the accuracy part is the kicker. Simulate them with sufficient accuracy, that's where we can actually start having a sense of rational design. Because we can't do that, so much of biology is done empirically. To make an analogy, when we design a bridge like the Bay Bridge here, that wasn't done empirically. People understand the physics of it. You can do calculations. You build a bridge. You know it's going to work.

My favorite analogy here is if you build a bridge empirically, the way you do most biological things, like drugs, build a whole mess of bridges. You build a million bridges. Find the one you think works the best and then start doing empirical tests, like you put mice on the bridge to see if the mice survive, and they put people on the bridge, and it goes in a sort of — If you think about the engineering risk and empiricism, it's a radical shift.

The ability to really start to engineer things that we can't do any other way because we just can't do the calculations, that would transform these industries and that's what at hand here. The reason why it's possible with a quantum computer is that the most accurate calculations are NP-complete. It's not 2 to the n. It's actually n-factorial, which is basically n to the n, even worse. That's the way to do the most accurate calculations. There are approximations people use, but those approximations have themselves huge limitations that are keeping us from using these things. I think that could be probably one of the first killer applications, the ability to truly accurately engineer quantum devices, enzymes, anything in that space.

[SPONSOR MESSAGE]

[0:36:40.6] JM: The process of troubleshooting bugs can be very tedious and inefficient for developers, especially are they are pushing more and more code to production. The unlucky developer assigned to trouble shooting bugs make it bombarded with error alerts and spend countless hours figuring out which errors to address first. Digging through logs and reaching out to other engineers.

Bugsnag improves the task of troubleshooting errors and makes it a much more enjoyable and less time-consuming process. For example, when an error occurs, the application team can get

notified via Slack. See all the diagnostic information related to the error and identify the developer who committed that code. Bugsnag's integration with JIRA and other collaboration tools makes it easy to assign and track bugs as they're being fixed.

Try all features free for 14 days at bugsnag.com/sedaily. Development teams can now iterate more quickly and improve software performance. To get started, go to bugsnag.com/sedaily. Get up and running in three minutes. Airbnb, Lyft and Shopify all use Bugsnag to monitor application errors. Try all features free for 14 days at bugsnag.com/sedaily.

[INTERVIEW CONTINUED]

[0:38:10.2] JM: This is something I don't completely understand. What I know about drug discovery is, computationally-wise, you take a simulated environment of the human body sitting in a computer somewhere and you do a bunch of kind of guided brute force tests for how some compounds might bind to certain receptors and how that would affect the simulated human body that you have sitting in a computer. Part of the reason that these things are not super effective, because we don't understand the pathways in the human body well enough to actually have accurate simulation. You can correct me if I'm wrong. I'm just trying to understand how we can actually — Even if we have massive increase in compute, how that's actually helpful if we don't have an accurate model that we're simulating against.

[0:39:01.9] VP: First off, designing of enzymes for, let's say, engineering applications on is very different than designing of drugs. In terms of the empiricism, they're different in terms of the process for some of the reasons that you just talked about that biology is complicated, the body is complicated. When you run an enzyme you're not doing it in a human body. You don't have to worry about toxicity when you're using it to generate a chemical and so on. That's done in some big vat and some industrial process. You don't have to worry about all these other parts. That's a key difference here.

A second one actually is that almost no drugs are designed with computation these days, or at least where the people involved in designing a drug say that it would have been impossible without the computation. Computation plays a huge role, but it's not the star. It's much more of a support role. This is shifting, and this is a whole another podcast to talk about which is, let's say,

machine learning engineering in drug design which is something that I've worked on a lot and then we've made investments in and I have tons of friends in this space. The compute and machine learning engineering in drug design I think is shifting. It has some ways to go. Actually, this could be very helpful in that shift. That's almost like a very different other game.

[0:40:06.7] JM: Got it. Okay. Yeah, because I guess the main issue is the unknown pathways. If you're just designing an enzyme to do — I have no idea what an application. I'm making better steel or something. That's not something where you have to worry about the collateral damage of some other chemical pathway, because you know lives are going to be lost in that kind of trial.

[0:40:29.2] VP: That's right. Without going to deep into this, some things where pathways don't matter. Let's say, if you come in with an antibiotic, you just have to kill the bacteria. It's very relatively easy for drugs to kill. That's part of the reason why drugs are so toxic. Killing is easy. Antibiotics kill bacteria, but not humans. That not human part if the tricky part.

It's a lot harder to cure Alzheimer's or to cure depression. That's where the pathways are complicated and the biology is complicated. There will be other ways to address that, but, yeah, probably not by these.

[0:41:00.9] JM: We did a show a while back about the storage potential of DNA. There are some people from Microsoft who are working on this. Have you looked into that topic at all?

[0:41:12.8] VP: Yeah. I've been a huge fan of. I've looked into — We've seen some companies. We have not made any investments in it yet. DNA is kind of fun to think about as a storage medium. I don't know if you've ever used tape backups. I've used them my whole career. Tapes are pretty flaky. Apart from that, if you have a take backup that I would say is more than 7 years old, that change that you have the hardware to read it is pretty low.

Frankly, I don't know if you have any cassette tapes on you. Even now, I don't even know where I would play a CD. I guess I have to go into my car to play a CD. What's different about DNA is that for this foreseeable future, I cannot imagine a world where are not able to read DNA, because DNA is so important to human beings. Human beings would have to be in some sort of

sci-fi beings of light like space before we stopped caring about DNA. Even then, we may be care about the DNA for the animals around us and so on. The read part of DNA is universal. The storage density is tremendous. The bandwidth for reading DNA is increasing faster than Moore's law. Writing is coming up to speed. All that is exciting.

Intriguingly, regular storage is also still moving like Moore's law too. It's a fun battle to see. I think what I suspect you'll see is that the first applications for DNA storage will be for long term. Storage where you want this to be around for four years or something like that and you have a lot to store and it goes sits in a minus 80 fridge somewhere. I could see that to start. I don't think it's going to be something where you have your backup USB drive will be a DNA drive. That might be quite a while. For certain application, I bet it's not that far off.

[0:42:56.7] JM: What are the enabling technologies that you've got your eye on that would make this space work?

[0:43:04.3] VP: Yeah. The one key part of it is reading. Alumina, and now Oxford and other people are having some really cool — and Thermo Fisher have some very powerful technologies for reading. Reading looks like it's in good shape. Writing is also — There are several companies, several startups that are involved in writing and doing some exciting things there. The question is when does those curves sort of catch up with the progress happening on backups. I think the thing about backups is that you have to think about the total cost of backups. The fact that most backup systems, the tapes are actually in the facility after two years even if you haven't messed with them, they're read back in and wrote to new tapes or preserving them and also that preserves for hardware generations and so on.

If you include all the cost of that, I think what I'm looking for is where the crossover makes DNA competitive and I don't think we're there yet, but it's a battle between two very powerful technologies. It's clear that the DNA one will win on density and once that starts to become an issue I think then we'll start to see I think a lot more uptick.

[0:44:04.9] JM: Okay. I want to talk a little bit about modern venture capital strategy, because I think as there's more potential for these ideas that for a long time are just preposterous or insane, now — I did a show recently with Geoff Ralston who is a Y Combinator guy and it's just

like the idea that there's almost nothing that is too preposterous to at least consider at this point, but these companies that are not just like SaaS companies. They're not just easy to understand, consumer products. I think that the discussion of how and why to finance these is kind of interesting. How do you think about financing for a quantum computer company?

[0:44:54.1] VP: This is a deep question and there's a couple of different ways to think about it. First off is that, in general, in investing in venture investing, our goal is to look for things that have world-changing, from an investor point of view, high data returns. These are things that we'll typically have to be pushing the edge.

On the other hand we want to do things that are not going to be ridiculously capital-intensive. A company that can become, let's say, $100 billion company but needs $99 billion of investment. That's probably also not necessarily the best investment. Something you think there's the hope. Even if a fair bit of capital comes in it's not going to be ridiculously capital intensive.

Finally, hopefully, there're ecosystem players that are interested in this space. In quantum computing, certainly, ecosystem of players, like IBM, Google, Microsoft now. Would Facebook be interested? Quantum machine learning could be very natural for them to want to be running in their own infrastructure. For any of these big players, it's not crazy to think about Amazon and AWS wanting in time to have a quantum cloud. Any of these players could become interested.

From a startup perspective, if you can really push powerful technology, you might IPO and partner with these guys. You might get bought by them. There are lots of different possibilities. There just has to be the sense that there can be progress. Ideally, progress and monitor not just by scientific progress, but revenue along the way such that you can build a big company the way some of the greatest tech companies have been built.

[0:46:21.4] JM: Now, in venture stage investing, you're typically not ever investing with the goal of a small acquihire type of liquidity event, but there was this acquisition of Deep Mind recently. Deep Mind got to a very strong financial position despite not having any product. The financial position that they got to was essentially based on this mote that they had of knowledge and talent and it made it into just this incredible accuihire, because there was so much upside in just the potential of the team. I know this is not traditionally a way to make a venture investment, but

do you at least think about that as a protection against some of the downside risk, because there's such a mote in terms of the intelligence of a quantum computer team that even in the event that there are some sort of issue in building a fab or whatever else, you probably have some downside protection in the fact that the company would get acquihired for a ton of money. Does that factor in to the way that you invest at all?

[0:47:32.0] VP: Yeah. I think you're spot on there in the sense that, A; we don't invest with the intention for acquihires. That's very much a plan B, or C, or D. What excites us is how this really could change the world and build a huge company.

With that said, I think as an investor, if you can do a super high beta investment, super high beta company where there is downside protection, if you really generally believe on both sides of that, that becomes almost a no-brainer for investing. Part of that no-brainer though obviously has to be that this has to be this A+ team and all these other parts have to be there, but that does help mitigate some of the financial concerns.

[0:48:08.7] JM: What do you think about the idea of basic science these days? The Deep Mind example, again, is they're kind of doing something that looked like basic science. They didn't really have an idea of what product they were going to build. They just kind of took some money and we're like, "Okay. We're going to build general artificial intelligence in 20 years." Is that a viable strategy? If you're trying to start a company, can you just say, "Look. I'm really good at this field. We're just going to do some basic science and kick the can down the road in terms of what product we're going to build." Are there enough things that look like basic science that have enough long-term upside that this is a reasonable strategy?

[0:48:47.2] VP: That's not a strategy that I would suggest, and I think there are not that many examples beyond Deep Mind that you can list. I think they had a fantastic team and have a fantastic team and a very natural wire in Google. That might be more of a perfect storm than something that is every day.

In general, your goal really should be to build something great, not to build something acquirable. Those are the things that we're looking for to invest. There are lots of ways to build companies and there's no right or wrong way to build a company.

In terms of the stuff that I'm looking at investing is something that has the potential to be the next Google, not necessarily just to be acquired by Google.

[0:49:25.2] JM: What role do governments play in quantum computing?

[0:49:28.1] VP: This is where there are a lot of interesting applications, applications in cryptography. Quantum cryptography is a super-rich area. There are a lot of applications that naturally overlap with aspects that government usually takes care and handles.

I think much like in other areas, I think we'll see governments in national labs being early customers of quantum computing broadly in many companies. It's often that early adoption. People forget about DARPA and ARPAs role in the internet and we can go on and on and on. That can really help spur innovation that really then becomes consumer innovation later.

I think there's all these different places. I think, also, one thing we'll see is to the extent that there's basic research that needs to be done, let's say in terms of algorithms or especially new types of algorithms, new types of computer science, much like there's new types of algorithms and new types of computer science being developed for regular computers too. That would be natural for the Natural Science Foundation and other governmental agencies to fund. There's many different roles for government to play as a customer, as a funder, and I suspect we'll see all of that.

[0:50:32.8] JM: If I'm a quantum computer programmer or want to be quantum computer programmer today. What do I do? Where do I start and what are the kinds of applications that I might be incentivized to try to build on quantum computing infrastructure today?

[0:50:49.8] VP: A very natural place to start would be to go to the Rigetti homepage and check out Forest, because all of that is just laid out for you. There's, Pyquil is super easy to get started with. You can go through examples. It would feel — I'm not sure what an analogy is. It's like learning scikit-learn or something like that. It's another Python tool that can do really powerful things, because with Python you could connect everything else and connect to your existing

[inaudible 0:51:14.5]. That would probably the natural thing and start playing with the Hello world equivalence.

Really, that will get you started, but that's obviously just the beginning. The next step is to really be thinking about what the opportunities are. What are the spaces to go after? What are the new algorithms to do? Either from the point of view of a basic researcher or from the point of someone that wants to develop code in open source or from the point of view of someone who wants to develop a company.

[0:51:39.2] JM: Okay, just to close off, I would love to know what's changing in your thesis for biology investments? Things at the intersection of biology and computer science with the — We have all these changes going on, like CRISPR. There's obviously all of the work going in to neural interfaces. Who knows how far along that stuff is, or maybe you know. Is there anything that's changing really rapidly and how you think about these investments? Maybe how things have changed relative to where your thoughts were two years ago?

[0:52:12.9] VP: We started thinking internally about this intersection between biology and compute in 2015 and we launched our first bio fund at the end of 2015, then investing in that. A lot of investing almost — This is maybe hack analogy, but it has a deep analogy with surfing that if you're a little too far in front of the wave you don't get to ride it. If you're too far behind it, you don't get to ride it. Timing is very very important. I think we did well with the timing of that fund and me coming aboard and us really expanding the firm to really go big into this space.

I think the next steps are realizing that machine learning and compute is part of a bigger picture. It's part of the picture that we referred to earlier about engineering versus empiricism. If you think about all the things that we love about tech investments versus, let's say, very very traditional biotech investments that tech investments can be capital efficient, can be typically engineering versus empiricism, may not have a ton of regulation, all those tons of things like Airbnb and Lyft that certainly have to be very thoughtful and working with regulators.

These are the things that we attribute with the tech side of investing and we're starting to see companies in biology and healthcare that are tech companies in all those ways and can be built up with small amounts of capital and so on. Machine learning was a very natural place to start.

In a natural intersection, I think we're seeing more and more. I think one version of this is the fact that there's deep connections with biology in terms of reading and writing and coding biology, but I think this concept of engineering biology, the way we engineer circuits and computers, that's emerging and I think that's going to be part of this bigger trend. You listed CRISPR. There's stem cells. We can list all these advances in biology that themselves — Forgetting about any sort of computer engineering themselves are super fascinating and exciting, but they really enhance and catalyze this perspective of how we can really push everything forward.

[0:54:04.9] JM: Indeed. Vijay, thank you so much for coming on Software Engineering Daily. It's been a real pleasure talking to you.

[0:54:09.2] VP: Yeah, great talking to you too.

[END OF EPISODE]

[0:54:13.5] JM: Look for a job more efficiently with Indeed Prime. Indeed Prime flips the typical model and lets you find a job more efficiently even while you're busy with other engineering work or coding your side project. You simply upload your resume and in one click you get immediate exposure to companies like Facebook, Uber and Dropbox. The employers that are interested will reach out to you within one week with salary, position and equity upfront.

Don't let applying for jobs become a full-time job itself. With Indeed Prime, the jobs come to you. The average software developer gets five employer contacts and an average salary offer of a $125,000 through Indeed Prime. It's 100% free for candidates. There are no strings attached.

Sign up now at indeed.com/sedaily. Thank you to Indeed Prime for being a repeat sponsor of Software Engineering Daily. If you want to support the show while looking for a new job, go to indeed.com/sedaily.

[END]