EPISODE 1518

[INTRODUCTION]

[00:00:00] ANNOUNCER: This episode is hosted by Sean Falconer. Shawn's been an academic founder and Googler. He has published works covering a wide range of topics from information visualization, to quantum computing. Currently, Sean is Head of Developer Relations and Product Marketing at Skyflow, and host of the podcast Partiality Redacted, a podcast about privacy and security engineering.

The classical computing power has doubled every two years, a pattern known as Moore's Law. However, the ability to fabricate more and more transistors in a computer chip is saturating as we are approaching atomic dimensions. Quantum computing is a promising technology to take us beyond this. A quantum computer uses qubits to run multidimensional quantum algorithms. Quantum computers are faster and can solve problems that are beyond the reach of even the most powerful classical supercomputers.

Dmitri Maslov is the Chief Software Architect at IBM's quantum computing division. He joins the show to discuss how quantum computing is different from classical computing. How to get started in this space? How would quantum computing impact current encryption standards? We also went through practical applications, quantum supremacy and quantum algorithms.

[INTERVIEW]

[00:01:18] SF: Dimitri, welcome to the show.

[00:01:19] DM: Yes. Hi, Sean. Thanks for having me here.

[00:01:22] SF: Yeah, it's really great to see you. You know, we met now over 20 years ago. I was just a snot-nosed kid desperate to talk to anyone about AI and quantum computing that would listen to me back then. And you were nice enough to take pity on me and spend some time answering my emails. I thank you for that. I think you're probably one of the smartest guys I

know. I'm very excited to have you on the show. But let's start by having you introduce yourself to the listeners.

[00:01:45] DM: Yeah, thank you very much. And thanks for the reminder. I distinctly recall you taking a class that I taught. And it's interesting that you asked me questions about quantum computing back in the days when I myself was not interested in quantum computing yet. It was, I believe, year 2001. And I became interested in quantum computing after that. And I recall not being able to answer your questions.

And now, 20 years down the road, I'm in a far better position to actually accomplish that. Alright. For the audience. Yes, my name is Dmitri Maslov. I work at IBM Quantum in the capacity of quantum software architect and research staff member. And my primary areas of interest evolve around all things quantum circuits and quantum compiling, which is an area that studies ways to efficiently implement quantum algorithms on quantum computers.

[00:02:41] SF: You've mentioned that you weren't actually doing research and had in depth understanding into quantum computing back when I asked you those questions. Can I take credit, essentially, for the career path that you have now spent 20 years of your life dedicated to as initially sparking that interest in quantum computing?

[00:02:58] DM: I mean, with all due respect, I don't think I can give you that kind of credit. But I can explain how I came to work in quantum computing, and perhaps maybe give an argument that I really had no choice but to work in quantum computing.

The story starts with being a young kid, literally 10 to 12-years-old. And in my experience, that in this world, you need to start thinking about what you would like to become in terms of your career early in life. And to illustrate this point, the question of what would you want to be when you grow up is not too uncommon to be heard in either school, or coming from parents.

And I personally distinctly recall that when I thought about what I wanted to do back when I was, say, 10 to 12-years-old, I give it a thought and I realized that I would like to build computers. And the year was 1987 or so by then. And unfortunately, or so I thought, computers have already

been built by then. I felt I was completely out of luck and needed to find something else to do with my life and career.

Then somewhere around here 2002, which is after 2001, when, I believe, you took the class that I taught. When I had interest in the classical computation in the ways of computing Boolean functions, think of it as a theory of computing, I stumbled on an area of reversible computation. And reversible computation is somewhat different from standard computation, but it is similar enough that it allows working on it without needing any further background.

To clarify, reversible computation is same as usual computation except you have to make sure that the output of a computation contains enough information about it to be able to reconstruct what the inputs were. Say, Boolean, and this irreversible. Because if you know the output, say, it's zero, then it is unclear what the two input values were. I mean, they could have been zero and zero, zero and one, or one and zero. Lots of options, and you cannot choose the right one.

And reversible computations, in turn, brought me to the area of quantum computing, since all quantum computations are inherently reversible. In a nutshell, it's kind of the combination of the interest in computing and in the circuits, and doing something novel that effectively brought me to the only place it could have brought me to.

[00:05:27] SF: Mm-hmm. Yeah. That's really interesting that you, I think, had this desire to build computers, and then realize, “Well, you know, a lot of people have done that.” And then you sort of lead yourself to reversible computing. And then that led to quantum computing. And I think that's a good place to kind of jump off with kind of starting to dig into what is a quantum computing.

I think, when it comes to software engineering, in my experience, and this is biased towards some types of engineering being considered sort of more hardcore than others. Backend sometimes is considered more hardcore than frontend engineering. Or doing bit twiddling of bootloader on an operating system requires more hardcore skills than something else. And personally, I think there's complex challenges in any area of engineering. They require different skills. But I think I'm willing to accept that quantum computing might be the most sort of hardcore that you could possibly get.

Just to kind of level set with the audience, because I think this is probably an area that most people will have little to no personal experience with, what is quantum computing? And then how is it different than conventional computing?

[00:06:32] DM: Yeah, yeah. I agree with you that we have to start with basics and discuss the basics first, and then maybe talk a little bit about the engineering. First thing is first. To explain how a quantum computer is different from a conventional classical computer, let us first take a look at the conventional computers.

A conventional computer is your regular desktop, laptop or phone. But that's not the key. The key that I would like to highlight is that whichever device it is that you're using and calling a regular classical computer, it operates on the laws of Newtonian mechanics. This goes to say that old classical computers could have been purely mechanical, as in operating by the means of levers, gears, belts, and other mechanical objects. And the only reason why we have electronic as opposed to purely mechanical computers is because electronics can be made small reliably. And because electronics is a lot faster at switching than mechanical computers.

But rest assured, your regular computer could be built with mechanical components and it obeys the laws of Newtonian mechanics. And this highlights the core difference between conventional and quantum computers. Quantum computers, in contrast, are based on the laws of quantum mechanics. Intuitively, this is the reason why quantum computers are different from an advantageous to classical computers.

Let me explain. Different because Newtonian and quantum mechanics are different physical theories. Should the laws and formulas of one were possible to rewrite in terms of the other theory, they'd be one physical theory. But they're not. They have two different things. So, they must be different. And indeed, they are.

In fact, so much so that the computational paradigms based on the two physical theories end up being different. The second part is why quantum computations or quantum computers are advantageous to classical ones. But say, not the other way around.

And this is easy to explain on an intuitive level as well. Say, quantum mechanics deals with tiny objects. And quantum mechanics becomes Newtonian mechanics when objects become too large. This means that Newtonian mechanics is just poor-quality quantum mechanics, if you want to think of it this way. And thus, a classical computer is a poor-quality quantum computer.

Or think of it this way. Say, quantum mechanics underlies the construct of the world. This is how the world operates on the very basic level, and thus its methods can be used and underlie the behavior and capabilities of all objects at the higher level of abstraction. And Newtonian mechanics is at a higher level of abstraction compared to quantum mechanics. Similar to object-oriented programming, I guess. And nature has to compile everything that it does down to the level of elementary transformations, and they're all quantum. So, nature really speaks the quantum language.

Let me throw there say that this explanation of the difference between classical and quantum computers, there are some good and bad news, depending on how you choose to interpret. And the news is that it seems unlikely that there will be a yet another computational model discovered soon, because we seem to be running out of physical theories to build the new kinds of computers on.

There is, of course, the relativity theory to explore. But the hopes or dim if you're single bit of information is of the size of a planet. I mean, you can do better than that. And the good news is that it seems as if, a long time from now, we will only have two types of computers; regular computers and quantum computers. And it seems as if there is no room for yet another type of computer. And that's a good news, because less to learn.

[00:10:40] SF: How does someone get started in this space? Is this something that you need, essentially, like a physics background and a PhD to work in quantum computing today because of where it isn't in terms of its history of development is still very early days? And there's a lot of sort of more research level work to be done?

[00:10:59] DM: Right. Right. It's interesting that you asked this question, because I understand this is for the benefit of the audience. And this is a great question at that. But let me clarify, Sean and I are actually co-authors. We wrote a paper about quantum computers in the year 2008.

And as far as I can tell, Sea didn't know quantum computing back then. He was a PhD student working on other topics.

And my point is, and please correct me if I'm wrong, or be the witness, it took you maybe a couple of hours, if even that, and I'm willing to bet it was under an hour, to learn basic concepts of quantum computing to start working in a useful way on quantum computers.

[00:11:42] SF: Yeah, I think that's reasonable.

[00:11:43] DM: Yeah. All right. And to further illustrate this point that I'm trying to make, let me say the following. Basically, in order to start working with quantum computers, you don't need much of a background. In fact, if you're a software engineer, you can learn concepts very quickly. I believe what you need to learn is the three things. One is the data structure. Second is what kinds of transformations are possible? And third, what constitutes the measurement? In other words, how do you extract the information from the data structure that you computed? The data structure is unitary vectors in complex space of dimension to the VM. And that gets a little bit technical. But that's where kind of the "complexity" is.

The transformations are unitary matrices of size two to the n by two to the n. You multiply vectors by matrices. It's a linear algebra. And your outputs need to be measured, and measurements are probabilistic and projective. There is furthermore a straightforward formula that is easier to write than say. So, I'm not going to say it. In any case, well, here you go. Once you know what the data structure is, what the transformations are in the measurement, you already know quantum computing.

[00:13:00] SF: Yes. It seems like even though, fundamentally, sort of the underlying construct of quantum computing is different than a classical computer, someone who's trained as a software engineer, there are transferable skill sets to move essentially from classical computing to quantum computing in a reasonable amount of time if this was something of interest to people working in conventional software engineering.

[00:13:22] DM: Yes. I absolutely agree with that. And moreover, I would like to highlight that the knowledge of physics is not even necessary, in my opinion. So, what you need to know is a little

bit of linear algebra and classical computation. And this is a good enough start to work in this area. It's very easy. I mean, this world is 100% logical and obeys the various mathematical laws.

[00:13:46] SF: On the picture, sort of what a quantum computer looks like, what does it actually look like? Is this anywhere familiar to what classical computer looks like? Are we talking about something that's like drastically different?

[00:13:59] DM: Alright. This question, I guess, can be interpreted in various ways. But let me consider one possible interpretation literally what it looks like. If you look at it, what does it look like? And then to answer this question, I'd like to first mention that there are various technologies on which a quantum computer can be built. This includes two leading technologies being the superconducting circuits and trapped ions. And these are technologies that to date resulted in the construction of most capable **[inaudible 00:14:28**] prototype, but universal programmable quantum computers. In other words, kinds of devices that can pass for a computational device.

For superconducting circuits technologies, what you see from the outside is that the illusion of refrigerator. Think of it as a huge barrel, maybe the size of a human, roughly speaking. Once you take the outer protective shell down, inside you will see a set of layers looking like blades, brass-colored. Lots of brass wire is between the various plates sitting at various levels. From the outside perhaps, it looks a little bit like a chandelier. And by the way, this is how experimentalist often refer to this structure. I mean, it literally looks like a chandelier.

And quantum chip, it's an integrated circuit. And it sits on the lowest and coolest level of this chandelier. And the quantum chip itself is maybe of the size of five millimeters by five millimeters. It's very tiny. Right. This is superconducting circuits technology.

In trapped ions technology, it looks different. The computation is being done by ions suspended in free space. I mean, it's one possible model. The ion chain itself is located in a vacuum chamber of the size of, let's say, a basketball, which sits on top of or near an optics table. And optics table in size is comparable to a pool table. And the quantum computer itself is a set of several or dozens many ions of a particular species, say ytterbium, which would be a setup that I'm most familiar with, that are suspended in free space in a vacuum chamber. And they're too

tiny to see with the naked eye, although you can probably force them to emit enough light to be detectable by a naked human eye by shining lasers at those ions.

And the reason you need the lasers and the reasons why this basketball size vacuum chamber sits on top of the optics table is because optics and lasers are used in order to induce the gates. In other words, to perform the transformations on this quantum computer that consists of ions. It looks very different. And other setups would look even differently.

[00:17:05] SF: Right. It sounds like if you think about like classical computing in like the history there, original computers were size of these, like, essentially a room. And then you know, we've gone a long way to now essentially walking around with a computer in our pockets. But it sounds like with quantum computing, in one model that you mentioned, you're dealing with like a human-sized drum, the chandelier type of setup. And then another one, you have something that's like the size of a pool table. Do you think that as this technology evolves, the size of the actual physical quantum computer will continue to shrink the same way that has with classical computing? Or is this just like an impossible thing to achieve using this type of technology?

[00:17:48] DM: I think the size will likely shrink specifically as far as the optics table is concerned. I think a better way would be to package this optics table into an integrated photonic device or photonic circuit. It becomes a lot smaller by then. Maybe it becomes slightly bigger than the size of a regular desk. But it's kind of, in a sense, beyond the point, and in the sense that I don't believe that this is necessarily important.

And the reason for that is what quantum computers are good at, at the highest level is solving the problems where the input is small, the output is small, but the computational space or the amount of computation that you need to do, say, classically is large. And because of that, the communication cost in order to set the input and read the output is relatively low. Therefore, rather than carrying a quantum computer in your pocket, supposed we are ever able to create quantum computers this small, you may instead just wirelessly access a quantum computer that is sitting somewhere in the lab and get it to do the computation and then you get the output. Because both the input specification and the output are small. And modern networks will be able to efficiently communicate this amount of information. And the benefit of that is you never need

to take care of the maintenance of a quantum computer. And the size of a quantum computer, in a sense, doesn't matter.

[00:19:25] SF: Yeah. I mean, even with conventional computing, we're essentially reaching that world where most of what we're doing is essentially interacting with computers over a network. And our laptops are really just a place to house our browser, that is then you're connecting with the greater world of the cloud.

You mentioned a little bit about what a quantum computer is good at. What are specifically some examples of like a practical application that someone can kind of like understand the types of computation that quantum a computing excels at versus something like a classical computer?

[00:20:03] DM: Right. In terms of the practical applications, kind of for the future, it may be difficult to tell because we do not yet have practically useful quantum computers. And the entire quantum computing area is furthermore relatively young, which is to suggest or give a hope that perhaps we're not yet well-positioned to judge what the impact may be.

This said, I have my own preferences and views, or beliefs, if you will. And I believe that quantum computing would be most impactful in the fields related to materials, research, chemistry, and I guess in a very long time, perhaps even drug design. And the reason is the core capability of quantum computing. In other words, what is quantum computing good at?

And you don't even need to know or understand quantum computing very well to buy the following simple argument. And the argument is quantum computers are good at simulating quantum mechanics, which makes perfect sense. And somewhat less trivial is the notion that we do not know how to simulate quantum mechanics efficiently on a regular classical computer. But it turns out that we don't know.

If that is indeed the case, then you would think that quantum computers would be of interest in all areas where quantum mechanical computations need to be done in order to find the desired answer to a computational problem. And what are those? Suppose if you have a chemical compound, and you may use a quantum computer to compute whether it conducts electricity or does not by simulating the properties of that molecule. And what laws do the molecules obey?

They’re small. So, that will be the laws of quantum mechanics. That's why a quantum computer would be particularly efficient at computing the conductivity property of complex materials.

And why is it useful? Well, it may offer a solution to the problem of finding room temperature superconductors. I'm not saying it will. I'm saying it may. And if that is indeed the case, why do we care about room temperature superconducting? Well, one of the reasons to care about that would be because power loss during the transmission and distribution of electricity to consumers, meaning the trip electricity takes between the power station and your home, is about a 10% loss.

You may say 10% is not much. But imagine everyone in the world getting a permanent discount of 10% on your electric bill in perpetuity. Suddenly, it doesn't seem all too insignificant. There are other examples of practical likely applications of quantum computers. Let me mention a few more. For instance, developing better chemical reactions. Developing better batteries. And I mean, what is a better battery? A better battery is one that has higher capacity. It charges faster. And it offers more recharging cycles.

And quantum computers may help with all those. But there is even more to say about the batteries alone. And to explain, let's think about what's a battery. A battery is a storage of energy. Not unlike a hand grenade. And I would say that the major difference between a battery and a hand grenade is how fast it releases the stored energy. And in particular, we don't want the battery to release the energy all too fast. And yeah, that can happen. And it is not something that we would like to happen.

To illustrate how it may happen, and not that I recommend that you verify what I'm saying, and I hope you trust me. But if you punch a hole in a battery, a fire may erupt. And this is an example of a fast release of energy. And we obviously don't want our batteries to get set on fire. Where does quantum computer come helpful in terms of solving this problem?

A sufficiently powerful quantum computer can predict by computing what reaction happens if the energy starts changing in the dynamic system set up by the combination of the various materials when you puncture the battery. And you can calculate how fast does the energy spread in the mix of various materials. And by doing this calculation, you would know if the

battery is safe when damaged or it's not. A quantum computer, other than helping to create a better battery in terms of the higher capacity, more recharging cycles, charging faster, can also help to create a safer battery. And this seems like an important enough topic.

[00:24:56] SF: MM-hmm. I think one of the things that you hear a lot If you follow any news related to quantum computing, especially even mainstream things outside of like research publications is the relationship between quantum computing and encryption. And does essentially quantum computing make all sorts of conventional encryption that we use today breakable? Because suddenly you can do things to compute in this very large space very, very quickly using a quantum computer.

[00:25:27] DM: Yeah, indeed, quantum computers make some of the standard cryptographic protocols such as RSA and ECC unsecure against a quantum attack. And this is because both RSA and ECC are the kinds of cryptographic protocols that are based on a belief of the computational difficulty of solving a certain problem.

To be more specific, this problem is the discrete logarithm problem over certain kinds of abelian groups. And it turns out that this is precisely the kind of a problem that a quantum computer can solve well. Thus, the original premise of the difficulty that the security was based on no more holds in the presence of a quantum computer, which renders protocols such as RSA and ECC insecure. Well, now, discrete logarithm over abelian groups is just one problem that you can base the security of a cryptographic protocol on.

And there are other problems for which we believe a quantum computer will not be able to offer an efficient solution. For instance, consider the lattice shortest vector problem. The problem is as follows**:** Given a basis in a discrete lattice over some multidimensional space, find the shortest vector in that lattice. And it appears to be a difficult problem to solve even for a quantum computer. We don't know for sure, much like we didn't know with RSA and ECC. But nobody was able to develop an efficient algorithm. And so, well, maybe there does not exist an efficient algorithm, either classical or quantum, to solve the problem of finding the lattice shortest vector problem. You can base the security on the assumption that this problem is difficult to solve and develop the respective public key encryption and key establishment algorithms based on this problem.

In fact, recently, the National Institute of Standards and Technology, NIST, ran a competition to determine best post-quantum cryptographic algorithms, and identified that those based on lattice shortest vector problem as the finalists. What does it mean in practice? It most likely means that should the need arise to revise the cryptographic protocols use, such as RSA and ECC, then chances are the underlying encryption protocols will just change. And the user, as a user, you're unlikely to even notice this because the secure communication will still happen. It's just that the underlying algorithms and underlying software will be different, which will be fixed with a patch that will install itself, I don't know, in maybe minutes at most. Yeah, this is the fact on the encryption and cryptography.

[00:28:24] SF: Yeah, that is comforting as someone like myself who works in the security and privacy space. As you mentioned, for something like conventional RSA, ECC, the way that they're set up is something that, in theory, quantum computing can essentially make new in terms of being able to break those encryption algorithms.

But given that we understand that, why is there so much research about like proving quantum advantage? Isn't it clear that, for certain types of problems, there is an advantage for quantum computers versus regular classic computers? Why is this a debate in the research world?

[00:29:04] DM: I mean, it's both clear and unclear. As far as the clear part is concerned, there is the so-called bells inequality, or CHSH game, if you will. Let me formulate what it is. It's a cooperative game where the two players, Alice and Bob, are given two random Boolean inputs, S and T. And they're required to come up with the beats A and B respectively and using no communication, such that the Boolean product of S and T equals to exclusive or of A and B. And to quickly remind, exclusive or of A and B, think of it as modular to addition. In other words, it's equal to zero if both A and B are equal to either zero or one. And it's equal to one if A equals zero, b equals one, or A equals one and b equals zero. And this exhaust all the possibilities.

This game can be won if both Alice and Bob have a quantum computer, like, "quantum computer" more precisely if they share parts of an entangled state. And this is a very constrained type of advantage, but it is a demonstrable type of advantage. And in fact, as of a few days ago, there was an announcement of a new Nobel laureates in physics. And the Nobel

Prize was issued in part for the discovery and the formulation of the CHSH game that I described, right?

It's not that you would like to play this game as a person. But it is a game that allows that can be won if you have a “quantum computer”. You really need a very simple state. It's not a full-fledged quantum computer. Meaning, a quantum computation, minimal quantum computation, already allows you to win this game.

But like I said, there are more examples. And let me say what those examples are. Say, let's consider a computational problem with black boxes. And suppose your task is to learn some property of a black box. And for certain kinds of problems of this kind, a quantum computer with quantum black boxes can learn those properties faster than a classical computer and provably faster.

But then, many of those algorithms, blackbox algorithms, are not very practical. You can ask an even harder question, to say, you can say that let's look at the white box computations, which is what one may normally think of when they think about the computing. And then formally speaking, within that model of computation, which is a completely unrestricted model of computation, we do not know if quantum computing is better. We just don't. But then we also don't know if P equals NP. In other words, if the computational complexity of finding a solution and verifying it are of the same complexity or not. This said, there are plenty of circumstantial evidence that both quantum computations are better than classical computations, and that P is not equal to NP. But like I said, formally, we don't have an exclusive proof.

[00:32:38] SF: I see. Yeah. And then there's obviously – Oh, well, maybe not obviously, but there is this proceeds or performance advantage of quantum computing, at least for specific class of problems over a classical computer. Are there other potential drivers for adoption, like greater energy efficiency or something like that?

[00:32:57] DM: I can think of the greater energy efficiency advantage. In a sense, a greater energy efficiency advantage may come twofold. One, as a result of different asymptotic complexities for solving certain problems. And suppose if it takes exponential effort to solve a problem on a classical computer, and only polynomial effort to solve on a quantum computer,

then at a certain size of the input, a quantum computer will be a lot more efficient energy wise independently of the energy cost per a unit of computation, just because exponent will eventually overtake any polynomial. So, that's one possible answer.

A second answer is, well, technically speaking, a quantum computer is reversible. And due to the reversibility, it does not lose the energy due to the loss of information. There is a physics principle that when you lose a single bit of information, it costs you some energy. There may be some advantage coming from that principle. But of course, not as big as the advantage coming from the different asymptotic complexities of the algorithms are responsible for solving different problems.

Other than energy, there is a potential computational advantage that comes from having to use less space. Say, consider the following problem. Say your input is a read-only memory, much like a CD-ROM, for those who are old enough to remember those. And suppose you have this memory, and you only have a single bit of memory that you can actually remember and rewrite.

And here's the question, can you compute the majority of three bits to quickly remind the majority of function of three bits equals one When majority of the inputs are one. In other words, if two or three inputs are equal to one, then it outputs one. And if zero or one inputs are equal to one, then it outputs zero.

And suppose, in this computational model, you can only use standard two input, single output gates. And the question is, can you compute the value of this function if you have only a single bit of memory that you can write over? The answer is, classically, you cannot, because there is not enough memory to accomplish this task. And let me try to convince you that this is the case. Indeed, consider the computation of majority for two bits. And then you compute the majority for two bits, and then you add the information on the third bit in order to compute your final answer. What do you need to know about the first two bits in order to be able to compute the majority of function?

Well, you need to know three distinct pieces of information. You need to know if the first two bits reside in the states zero and zero, because then it is guaranteed that the output of the majority is zero. That's one case that you need to distinguish. The second case that you need to

distinguish is if both beats are ones, because then majority is guaranteed to compute the one independently on the value of the third bit. And the third scenario is the first two bits contain values 0, 1, or one zero. And then the answer depends on the third bit. In other words, you need to remember three pieces of information. And a single bit does not offer you this opportunity, because a bit can only store two pieces of information.

However, if you look at the quantum bit, then it turns out that a quantum bit can be made to store more than two values simultaneously. And as a result, a quantum computer with only a single quantum bit of memory is able to compute the majority of three bits. And this shows that quantum memory is more powerful than classical memory, as it can store more things. Moreover, this is not just theory, as this behavior was explicitly demonstrated in an experiment, a quantum computational experiment.

There is slightly more depth to this kind of experiment. And I'm trying to illustrate something about quantum computation here. According to Polyvagal's theory, the amount of information that you can learn about an N qubit quantum state is limited by N bits. You cannot learn more than a single bit of information from a quantum bit. However, the ability to compute majority using one bit of memory means that in fact, so long as you do not look at your memory. It contains more information than it does once you look at it. That may not make very much sense. But it's true.

It furthermore highlights an important principle of quantum computing that makes it so difficult, and it is that not everything that doesn't make sense is untrue. In other words, there are things that don't make sense but they are true, which by itself kind of makes sense. And this is because your intuition is based on common sense, right? And your common sense is based on your daily experiences. And all of your daily experiences on Newtonian mechanical. If quantum mechanics doesn't make sense, this is good. Because if it did make sense, it will be called Newtonian mechanics. And it's not. I hope you're with me on that argument.

[00:38:58] SF: Yeah, I think everything gets a little wacky when you start to dig into quantum mechanics. And I agree, I think that if you think that you understand that it makes sense, then you might actually be misleading yourself and not actually know enough about it to realize that you don't understand it. And it doesn't necessarily make sense.

I want to talk a little bit about one of the sort of results that made headline news in the last couple of years, which was in 2019, Google claimed that they had achieved quantum supremacy. And I believe their results have actually since been passed by a regular supercomputer. But what was it that Google claimed to have achieved at that time?

[00:39:36] DM: They claimed to have achieved supremacy, as you mentioned. And supremacy, as originally defined by Preskill in his 2012 paper, I have a quote here in front of me. So, I will read the quote. And this is the entire sentence, and it goes as follows, “We therefore hope to hasten the onset of the era of quantum supremacy when we will be able to perform tasks with controlled quantum systems going beyond what can be achieved with ordinary digital computers.” Yeah. That was the Google's claim to have achieved the supremacy.

[00:40:15] SF: Mm-hmm. And you and your colleagues at IBM had cast on that achievement at the time. Why was it that you had written and come out publicly that they – and challenged, essentially, the claim of quantum supremacy from Google?

[00:40:30] DM: Right, because we didn't believe that Google reached the supremacy in terms of its original definition, or even supremacy in a weaker sense. And let me clarify. Basically, what happened is IBM demonstrated that Google's computational experiment, which is an excellent experiment. It's indeed very interesting. But it can be simulated classically in two and a half days.

However, Google claimed that their experiment can be simulated – If you wanted to simulate it classically, then the classical computation would take 10,000 years. And the difference between the 10,000 years estimate and two and a half days estimate made, I guess, such a big difference in the IBM size. And moreover, I believe that by the time that Google wrote the paper, it could have been figured out that it's possible to perform a simulation in just seven days.

I guess the IBM's contribution was reducing seven days to doing it two and a half days, which is roughly a factor of three improvement. But even discarding IBM's improvement, there was a simulation that Google could have referred to, which would only take seven days on the largest known classical computer that would simulate their quantum experiment. But this said, quantum

experiment took far less than these. I believe it took only a few minutes. There is still some sort of weak supremacy, but not as stated in the paper. And that was the part that was criticized.

[00:42:03] SF: Right. Okay, it makes sense. Yeah, substantial difference between 10,000 years and seven days, or even two and a half days. One of the areas of research that you mentioned that you're interested in is quantum compilation. What is a quantum compiler? Fun to kind of try to think about that and put it in terms of maybe conventional engineering.

[00:42:25] DM: Right. You can think of it in terms of a set of software tools that translate the algorithm of interest into machine executable code, if you're willing to think of it this way. Or you can think of it as a set of methods, approaches and techniques to express quantum algorithms as short circuits directly executable on quantum computers.

However, you look at it, the key of compiling is much like classical compiling in quantum compiling is to express a quantum algorithm of interest using the smallest number of computational resources, whichever those resources are. It can be the number of gates. It can be the circuit depth. It can be the number of qubits used and so forth.

[00:43:13] SF: I see. And in terms of expressing a quantum algorithm, how does someone do that? Are they essentially creating a quantum program that consists purely of like laying out gates? Or are there abstractions in terms of like programming languages for designing a quantum algorithm?

[00:43:31] DM: A few levels of abstractions that I can think off, in the sense that a one level of abstraction can be the one where you think of a computation in terms of the elementary gates, such as the CNOT gate, which is controlled NOT gate. An analog of classical modular to addition of Boolean numbers. Or you can think of quantum algorithms in terms of the higher-level objects such as, for instance, integer adders, multipliers, the quantum Fourier transform, or other sorts of specialized transforms.

But this said, the number of those logical layers, the different layers, is very small. And this goes to illustrate two points. One, quantum computation is, in a sense, in its infancy. We don't yet have very complex to express quantum algorithms that require you to have multiple, maybe

internal representations or very complex software. In fact, all the algorithms, quantum algorithms that I can think off, can be expressed in terms of quantum programming languages using maybe a page or two of text. And now that I'm thinking about it, frankly, I haven't seen an algorithm that needs two pages of text to be expressed as a quantum program. I think I've seen only those expressible in a single page of text. But let's see a few pages.

[00:45:02] SF: Yeah. Pretty small.

[00:45:05] DM: Yes. Very small. Very short. Yep.

[00:45:08] SF: And what are some of the major challenges that people are working on in quantum computing today? And I guess, like looking back on, you've been in the space for 20 years, what are some of the major breakthroughs that have happened during your time working in quantum computing?

[00:45:21] DM: I would say the major challenge in a single word is fidelity, as in gate fidelity, which is the same as same reliability of quantum gates. It's a huge challenge, in the sense that if you give me a quantum computer with the gate fidelity of one, that is absolutely perfect gates. Then it will be useful today in order to solve practical problems. Because we have enough qubits. We just need the gate fidelity to be higher. And improving the gate fidelity is notoriously difficult.

[00:45:58] SF: That has to do with essentially quantum noise. Is that right?

[00:46:00] DM: Yes, with noise. Yes, correct.

[00:46:03] SF: Why is there so much noise that happens inside the computation of a quantum computer?

[00:46:09] DM: Well, noise in quantum computations is nothing other than the result of the interaction that you have with your quantum computer, or the interaction of a quantum computer with the environment. And you may say, “Well, let's completely remove it.” As in, let's perfectly isolate a quantum computer from the environment. And let me say that, sure, you could do that.

But then your quantum computer would be perfectly isolated from your ability to program it. And every time that you want to program and run computations on it, you need to talk to it. And this means interact with it. And it's the process of the interaction with the quantum computer that introduces the noise.

[00:46:56] SF: I see. I guess, sort of looking ahead to the next five to 10 years, where do you think quantum computing is going? Is this something that – is this problem something that is going to be – you feel would actually we're going to make essentially significant improvements to in the next five to 10 years? What do you think, from 10 years from now, we'll be talking about in the quantum computing world?

[00:47:19] DM: Frankly, I have no idea. What I can say is, when I joined IBM, which was in the year 2019, I came to IBM from the National Science Foundation, where I was a Program Director, and I had a tenured government job. So, I gave up my tenured job, and the standard job was outside the quantum computing industry, in favor of a non-tenured job in quantum computing. And I guess my time horizon back then, and now as well, is five years. In the sense that it is my belief supported by personal action, not just words, and remaining to be the case to date. That five years down the road, quantum computing will still be here and it will still be important. And hopefully it will be more advanced. How much more so? I have no idea. But my own career depends on the answer. So I wish I knew the answer. So, I wish I knew the answer.

[00:48:19] SF: In some ways, it's probably exciting to work in a space where you don't know the answer, though. Because you didn't spend all your time sort of building up expertise and dedicating your career to the space to necessarily know everything from day one. It’s the journey and exploration that you're probably excited by.

[00:48:37] DM: Yeah, that's right.

[00:48:38] SF: Dmitri, I want to thank you so much for coming on the show. It was great to see you after all this time. I appreciate you taking the time to share your knowledge and expertise with the Software Daily listeners. And I think this is a topic that comes up in the news all the time and sort of mainstream media. I think it’s really interesting to kind of dig in and really dive into

the details of what's going on in this role beyond just sort of sensational headlines that you might see in your news feed. So, thanks so much.

[00:49:07] DM: Yeah, there is one more note that I would like to add maybe concerning the discussion about the noise that I kind of struggled to answer a little bit at first. Let me explain why quantum computers are difficult to build, and difficult much more so than classical computers.

Imagine I asked you to build a one-bit classical computer. And what you could do is take a coin called the head zero on tables one, put it flat on the table and flip it by hand to perform one-bit computations. There're your entire one-bit classical computers. Easy. And a quantum one qubit computer is exactly the same. But you're now the size of the planet. And this is a fair comparison, because the relation between the sizes of a human to an elementary particle is comparable to the relation of the size of a planet to a coin.

And now, imagine the kinds of problems you will have while operating a coin if you're the size of the planet. Some of those may include not being able to even see the coin, for it being too small to see. Then maybe squishing it into the table, which lies on along with the table. And the house of the table is in. Possibly, the entire city the house is in. And also, if you're the size of the planet and barely touch the planet the coin is on, the seismic wave can flip the coin or throw it out from the table. And this introduces an error to your computation. And these are precisely the kinds of problems that one has to deal with while building quantum computers. But I think those problems can be solved eventually, because solving them does not fundamentally contradict the laws of physics as we know them. The question is most likely that of time, as opposed to the possibility.

[00:51:04] SF: Yeah, it's amazing. I mean, I think this is – even though you've spent 20 years working in space, there's still so much to – it's such a rich lately. It's very, very much early days and sort of just scratching the surface of what can be done. I think this is – I'm glad that you kind of stopped down to share this analogy. I think it's the perfect analogy to kind of really articulate the size of the problems, the planet sized problems that people have to try to figure out in the quantum computing world. Thank you so much.

[00:51:33] DM: Yeah. Thanks a lot for having me here.

[END]