Qiskit API Update

IBM Code Tech Talk

03/14/2018

https://developer.ibm.com/code/techtalks/qiskit-api-update-2/

>> And we have an update today. And we have found a research center, our scientists Ali who was on the original team that created the Qiskit and the goal is to create a scaleable stack, bringing quantum computing to the masses with a familiar stack, using Python. So Ali is going to take us through this after giving an introduction in quantum computing. And he has the team with him to answer questions. So Led Bishop who presented last year is answering questions in the chat. Don't feel shy about posting questions. I would like to hand over to Ali for this talk.

>> Thank you very much for the introduction. Thanks, everyone, for joining. My name is Ali and I work in the quantum computing group at IBM research. And today I want to show you how you can run real quantum devices. Quantum computing today is a culmination of a lot of scientific progress over the centuries. To sort of '80s maybe these logs can be used for computation and in 1990s there was an explosion of results that quantum computing can be used to solve certain problems faster. A lot of work in labs to try to build these devices. And now we are sitting in sort of a unique point in this history where in 2016 IBM started getting access to everyone worldwide access to quantum computers through a program called (inaudible) experience. 75,000 users have signed up to use these devices from all over the world. And there is more two and a half million experiments. And we are doing really serious work. More than 60 scientific papers that have been authored.

So in this talk I want to tell you how you can use these devices. So the computers sit basically in the laboratory in cryogenic temperatures and a couple of years ago only those who went to the lab could use the computers. Now as quantum developers can have cloud access to that and that's -- Qiskit is a file proof framework that has a simple Pythonic interface and will optimize them for specific back ends. Back ends could be devices or they could also be simulators. You would use a classical computer to simulate for sanity checking and results.

So in this talk I want to give you a brief overview of what quantum computing is. For a deeper dive in to the topic I recommend seeing my colleague Lev Bishop's talk where he goes in

to detail in to the topic. And I want to jump in and look at a concrete problem where quantum computers can solve the problem faster than classical computers. And then I will show you how this can be implemented in Qiskit and run on a real device and then conclude with talking a little bit about the near-term landscape for quantum computing and challenges ahead.

So quantum mechanics basically understand quantum computing and need to understand quantum mechanics. And this comes down to a few things that are very counterintuitive. And there may be things about quantum mechanics that you have to accept. So as an example of a quantum mechanical object let's consider a photon. Can have a polarization. For example, I am showing photon polarizing in a direction. Polarize in a vertical direction. When I shine a photon polarized with horizontal, you only see it in a horizontal way. If you photon vertical you receive it in the vertical direction. Photons can be polarized in any degree. But again if I shine at you a diagonally polarized photon you can only detect either as horizontal or vertical. Once it hits the measurement apparatus it will manifest as horizontal or vertical photon. For this perfectly diagonal halfway through, you see 50% of times in horizontal and 50% of times in vertical. And this is one of the amazing things about quantum mechanics. Because suppose I prepared a thousand copies of these diagonally oriented photons, the same photons. And then when I sent this 1,000 at you 500 times you see them vertical and 500 times you see them horizontally. Identified prepared photons can behave randomly when measured. It is different from the randomness of classical world. For example, you might talk about the coin being 50% tails and 50% head. It is a limitation of our modeling, if I knew exactly what force I exert on the coin. Here the randomness seems to be based in to the quantum mechanics. Second amazing thing is that in general you can't know the state of several photons by knowing the state of each of them individually.

So, for example, here I have two photons that are diagonally oriented and this is fine. Separate states that I can say diagonal state. However I could think of a separate state for -- I could assign a separate state to these pair of Qiskits and the same state, what happens is whenever you measure one of them in the horizontal and second would definitely be horizontal also. And whenever you measure one invertically and the second one is vertical. This is very -- this is very different from a classical world because in a classical world you -- knowing states individual system you can assign a system to the whole as well. These two states, one on right and one on left are very different. The one on the right can be thought of as one diagonal photon and one diagonal photon. You can't know

anything about the individual photons.

So the way I like to think about quantum computing is very purely mathematical point of view from algebraic point of view. When you apply quantum operations or gates as we call them this is like an assigned matrix of that vector. Takes it and moves it to another place. And then measurements or observations of the system are like projections on the basis. For example, if I measure this green arrow, now green state it will 50% of the time go to the South Pole. You can think of it as being vertical polarization that I showed you before or 50% of the time might go to the 0, North Pole. If I measure the blue one because it comes closer to the North Pole it will have a higher probability of going to the North Pole. You have vector and matrices and you have projections on basics. And quantum algorithm is this, you prepare a state. You apply some measurements and you can massage the states to some place that you want. And then the hope is that after you get to that good place when you measure the result of your problem faster than classical alternative, for example. For the rest of this talk you need to know about two gates. They are called the H gate and the controls CNOT gate. These are the ones that are shown in blue in this picture. So continue by describing what these are. And then we can jump in to the problem that I promised.

So suppose that I start with a Qubit in a state 0. That's -- taking that horizontal polarization to a diagonal polarization. 0, 50%, 1 sort of Qubit. Now if I had two Qubits and I have 25% of 00, 01, 001. I have an equal probability of eight possibilities.

And so this is where the powerpoint computation comes from. Every Qubit system, the potential power of the system is doubling. So there is exponential growth. But I mean things are not as simple as that. Because this -- all of this state exists in the quantum world which we really don't have access to. We can measure. If we measure this system, for example, we just collapse one of these at random. Equal probability of collapsing to any one of these states. So with 12% I got 00, 01. Creates super position and creates -- it takes a Qubit in to a probabilistic Qubit.

The second thing that the -- phase, a Qubit in state 0. If I start with a state of 1, then we go to 0 minus 1. And -- and this minus we call a phase. Again notice that from a classical point of view, both of these scenarios when I measure I have 50% of seeing what -- 50% of seeing 0 and 50% of seeing 1. So classic they are not distinguishable, but in the quantum world there are two very different vectors. One is the addition of 0 and 1 vector and one is subtraction of 0 and 1.

And then lastly we should keep in mind that H gate itself,

when I apply the H gate either of these scenarios I end up where I was in the beginning. That's the H gate. Creates operation and induce phase.

There are certain gates that I talked about is controlled NOT gate and think of it as a quantum analog or exclusive OR gate. What it does there is a control Qubit and when it is 0 will have no effect. And when control Qubit is 1, it flips the target. 0 becomes 1 and 1 becomes 0. So that's basically all we need to know.

So let's jump in to the problem. The problem is called the simulated by (inaudible) Bernstein in the 1990s and the problem is as follows: Suppose you have a black box or a person called the Oracle, and they have -- they are guarding some secret. It is in the form of a bit strength, 00, and 011. That's the one shown in red. The only way -- so your goal is to recover the secret, find out what their secret is. And the only way that you can attract with them, they only respond to queries in the form of your own input stream. And then they take your input and they do a dot product between your input and their secret. And they give you back one bit of result which is -- dark product. So because end bits to recover and every query gives you one bit of information. Classically the way you can do this, have to do minimum of end query. For example, if you design your experiment like this with each of these queries it covers one bit. So after all end queries you recover all the secret. And this is the information right now. You cannot do anything better than this. However Bernstein showed that you can actually do this on a quantum computer in only one query to the Oracle. And that sounds pretty amazing that you can do that. So let's look at how it works.

So understand that the algorithm we kind of first have to think about how a quantum Oracle might be implemented. So classical Oracle as I mentioned has an X as input and gives you X.S on the output. Output want to do that on a quantum computer. Because quantum operations have to be reversible. Every quantum computation has to be reversible. And once people said this might be a limitation on quantum computers that everything they do must be reversible, but Charles Bennett who is at IBM shows that you can do any nonclassical computation with not much of a downgrade in performance. So we take this nonreversible computation and make it reversible. And the way to do this is by padding the inputs and outputs. We create one extra called the temporary Qubit. And this is a valid quantum operation. It takes X and then gives us X.S on the output, mark X with a template and everything can be walked back again and computation can be recovered back.

So as I mentioned here black box, you want to query it in such

a way to recover the secret. So we don't need to know what's in the black box but to understand how the solution works it is very instructive to look inside the black box and think about how it might be implemented. I claim this simple circuit is implementation of this Oracle. So, for example, suppose I have an Oracle with string of 0101. Now I -- my claim is that with a simple pattern of CNOT, where you put them at 1, you can do that the same -- you can achieve the same effect. And I mean easy to see that. As I said the CNOT, what it does is you can just go by the definition exclusive or. And when the control is 1. So when I applied this pattern of CNOTs I basically get this on the output, X3.S3 and so on. If I had a different Oracle with a different secret in it then I would just change the pattern of CNOTs. For example, there is one on each of the wires, on each of the Qubits and this is encoding the same secret. This simple circuit is implementation of the Oracle. Simple pattern of CNOTs you can achieve that.

To recover the secret. And this was -- the idea of single running is a trick in quantum computation known as (inaudible) feedback. So suppose that I start with two Qubits, one in state view and one in state 1. Now wonder if the definition that I showed you before, top goes in to top Qubit plus 1 and bottom is Qubit minus 1. Multiply this out. If you apply a controlled NOT here you can just apply the definition of the controlled NOT. Whatever the first Qubit is 0 do nothing, and first Qubit is 1. Flip this. It can be factored in to this. Now notice what has happened here. The first Qubit started as having a plus here and having no phase but it kind of acquired this phase of the bottom Qubit and that's called phase base.

So now if I apply another layer of H gates, those end up being 11. So we see that this -- this phased back circuit, took the 1 from bottom and brought it up to the top Qubit. And if you noticed this thing in the middle it was very similar to what I told you the Oracle is doing. The CNOT. And Oracle is basically a pattern of these CNOTs. So I mean now you should have an idea of how this solution might emerge. So the solution is this, we take the Oracle as a black box and you sandwich it in between two layers of heta marks and put the temporary state of 1. And you apply it by the flip to it. But then the thing is two layers of Heta marks. If you move inside the Oracle again you see that as I mentioned all of these is like phased kickback circuit. Whenever there is a 1 and there is a CNOT and phased kickback, from the bottom Qubit to the Qubit 1, 2 or 3. Now if I do this and measure, so this circuitry is a measurement of Qubit. We only want to recover end bits of secret. As I measure I recovered the secret hidden in Oracle, the 1101 perfectly. So this is the most significant bit. And this is

the least significant bit. And this is a deterministic algorithm with the 100% accuracy. You don't need to perform any more queries.

Okay? So now hopefully you got an idea of how this is working. Let's go in to actual implementation. So now Qiskit I am going to write. I am going to show you. Upper notebook and it is the entire thing is Python code. Input in to package and you use it. So when you use it the -- the notebook, okay. Now it is everything is recent. So let's start. So basically we just want to go by this. We want to implement this circuit that we kind of derived.

So initially we just have to create these quantum registers and classical registers. So Qiskit that's simple. Write classic register and name and size and create them. Okay? So now let's start building the Oracle. The Oracle as I said has a secret and acts on four Qubits. 00, 11. So it is simple. I just basically say create a quantum circuit for an Oracle and do that pattern of CNOTs that I talked about. Whenever there is a 1, do a CNOT and it is a CX. So we do that and just assign the check. And we can just basically print the circuit in assembly format. And we see that -- right. So there is a header. We have the quantum and classical registers. And then we have two control Xs or controlled NOTs for the first two Qubits that go to the last temporary Qubit. We have the Oracle. Now we want to build the solution circuit around it. Basically we want to create the two layers of (inaudible) and just do one shot, one call to the query, to the Oracle. So basically I create another circuit and add the X gate first to the last Qubit. And then do my layers and call to Oracle and then at the end I measure. Execute that.

So Qiskit you can use it to automatically draw the circuit for you so you can basically have a visualization of what it -- the whole thing looks like. So for now automatically generates a visualization of the circle form and you see that that's exactly where we want it. We -- this is automatically generated but the function is the same circuit. We get CNOT on the two significant Qubits and nothing on Q2 and Q3. This is the circuit we want to implement. Automatically generated. So now let's say we want to run the program, right? So in order to access IBM -- first you need to wrap your circuit inside a quantum project object, that's the main object you use to interact with the back end. So I will just add my circuit to the quantum program object and now I have that.

Next we want to use the actual back ends IBM quantum experience you have to authenticate yourself. So whenever you register you get API key and use that key. And now when I set API now it is authenticated and I know who the job is coming

from.

So now we can execute. I want to show you two executions. One I'm going to the simulator and then I will do on a real device, okay? So the simulator is basically running on my own laptop. When I install Qiskit it comes with a simulator. When I run the circuit now on a simulator, I get a result object. And now I can show the results. You see this is the -- I ran the experiment 1 in 1,000 and typically do that. I will talk about why. But in all of them because the -- the algorithm is getting the right answer all the time. I get the correct answer that I encoded. I could change this in a different bit stream. Like 0s, 1s or 2s. So now bit stream is 0010 and execute this again. And you see that the number changed now. I covered the same bit stream. The new bit stream. And all of this happened with only one query to the Oracle contrary to the actual case where we need it. So now let's run on the new quantum computer. I'll call -- I'll run my program on a 5 Qubit kit on the cloud. The name of the chip -- now when we run a program on a real device you have to drop, enter the queue and there are a lot of users. You might have to wait a little bit.

>> Sorry to interrupt you. It is hard to read your screen. Maybe you zoomed in too much. It is very blurry.

>> Okay?

>> Yeah. There you go. Thank you so much.

>> So now this job is running on real quantum computer. And what you will expect -- you will see now is that after it comes back, the real device is contrary to the simulator that I showed you and that's the reason why we need to run the experiments many times. Even though this algorithm, even though IBM quantum physics are some of the best in the world. When we get the answers now we get the right answer again, 10 as we wanted. But now the -- there is also some unwanted answer as well. Now this has been pretty good. If you look at the results the vast majority of the results are the right ones. So this is -- this is a very simple program. I was able to solve a problem that is -- that is faster than possible on a classical computer and do it on an actual hardware. I hope you get the idea that it is simple to drive and run your own programs. And I'll talk about places where you might want to get started.

Okay. So also let's take a moment to pause and think about why this algorithm works the way it is. In the classical sense, if the classical role if I had to query in Oracle I was able to query with one number at a time, with one bit string at a time. In the quantum role because of the position I can also send a query of -- a position of queries to the Oracle. However it is not just really trying every answer simultaneously. We are exploiting the structure in the program and the problem and that

is that we were able to encode information in the state of the Qubits. If I sent the queries what the answer would be, they would be all of the answers in to a position. That's not very useful but we were able to include information in the (inaudible), something that you can't do on a classical computer with classical bits that don't have phase. And with that we are able to exploit the structure of the problem and get the correct answer. And this is what happens in all of quantum algorithms. That's why it is so hard to come up with useful quantum algorithms and they have active research area.

So in the new term basically we -- I mean we are at a stage where we have a little quantum computer where we can run new programs on as you just saw. However there are challenges, right? Basically we don't have many Qubits on the IBM cloud. There are five Qubit systems, 16 Qubit systems for free. It is hard to push that envelope and go up further. IBM has right now a 50 Qubit system on their test as well. We also don't have many gates to do that we can perform. And the reason is that gates are very noisy. And so if you try to do a lot of gates, you accumulate very fast. And as you saw in the bar graph that I showed you now because of circuit is not that big, I can get the right result pretty well.

But if there were -- if I try to do many more gates then these noise bars will become augmented and they will kind of blend in with the actual result. That's one problem. And the other problem is that Qubits do not retain their information indefinitely. If you need a Qubit, will use the information and go to unwanted state. So that's known as relaxation. I want to -- more detail about and talked about some of the challenges that we have ahead.

So again let's go back to the Bernstein circuit. This is the circuit that I showed you. Layers of Heta and Oracle in between. This is what I try to run right now. You program, just like writing a program on a classical computer. You don't want to think too much about the back -- the device. You just want to write your program logically. And then you want to -- the compiler to be smart and translate your code in such a way that you get the best performers out of it. However there are in quantum compiler things are more subtle because it is not only performance issues it is the compiler is not good. Then because of the noise it could know the difference between success and failure of a program. So let me illustrate this with an example. So this is -- circuit that we just programmed and suppose I execute on the five bit chip on the IBM cloud. The chip has certain limitations. One was important things was that the Qubit has limited activity. Whatever error, you can use CNOT in that direction. Here you can do a -- as Q2 for

control and Q4 for target. If you look -- if you try to match this program to these devices, you see that you cannot do that. You cannot find if single Qubit that can be a target for four different control bits.

Now when you write this in Qiskit as I just did and try to execute it, Qiskit under the hood transformed your circuit to try to match it and respect the connectivity constraints of the device. This is the output of two different compilers. On the top we have a good compiler that has done that transformation pretty well and created a small circuit. On the bottom where poor compiler in the process of transforming a lot of gates and not simplify a circuit. If you worked out the math all of these circuits do exactly the same thing. All three solve the Bernstein algorithm. And these two are also -- they are able to run on a device because they respect the constraint activities of a device. When you run these this is the result that you get. For group one you are able to cover an answer and effective noise is pretty mitigated. And the bad circuit all you read out is noise. You are not able to figure out what the result was. So this is why the role of software is very important here. You know, there is a lot of work in trying to find algorithms that can work with small connect circuits. More coherence of Qubits and things like that. But then this transition between the program to the actual device also need to be very smart about how we do that transition.

As you can see the difference can be success or failure of the algorithm. So actually this is the subject of a program that we recently launched at IBM called IBM Q awards. So this is one of the challenges. We call this a developer challenge and we are asking participants to design better mappers. Better ways of translating this circuit to a different given back end.

And there are $5,000 in prizes to be won. And the deadline is coming up soon. I encourage anyone who is interested to give this a shot. And we will integrate the best result that we get. The best submission in to the Qiskit framework.

So lastly let me talk about the other form of error, other form of basic error which is Qubit relaxation. So as I said Qubits do not maintain their state indefinitely. So essentially experimenters measure these values or these relax rates all the time. And if you go to IBM quantum experience website you will see for all the Qubits there is a reported time in the latest calibration on what that relaxation rate was. So you could read that from the website or you could also just design your own experiment and measure it for yourself.

Now I want to show how we could that in Qiskit. The point of this is do a very intimately experimental thing about the clinical system but just by running circuit you can find

something about the parameters of the real chip. So a very simple experiment to do that. And I will show you how to run this in Qiskit. What we do is we take one Qubit which is at state 0 and apply an X gate. And then we just wait a little bit before measuring it. So if you wait a little bit you would expect to read one. Because of relaxation when you put in a state of 1 to when you observe it, it is relaxed to the 0 state. So if you measure it soon after, let's say I done this experiment one thousand times, 920 times you will see that it is in the state that you want it, in state 1. And any time you see that it is gone to the unwanted state of 0. So if you wait a little bit longer or a little bit longer, you will see that the unwanted state gets more and more. So this is the idea. The idea is that we create a bunch of these circuits. One more maybe a thousand times and you can plot how this relaxation is happening. And we can get a factor for that relaxation. And then we can check that against IBM quantum experience.

The result is put on the IBM quantum experience website. Let's do that Qiskit live demo. So this is a -- I will restart a notebook again. And I just kick off all of this and then I can describe a little bit longer while we are waiting for the result. This is running on -- now this is running on the IBM QX5 chip which is 16 Qubit chip and I am trying to characterize Qubit 2.

>> It is hard to read that. It is all fuzzy.

>> Is this better?

>> No.

>> Better?

>> Not for me. Clyde is for you?

>> No.

>> I don't know if it is projecting or sharing your screen.

>> No, I'm not. Should I stop sharing and share again?

>> Yeah. Try that.

>> Let me see. Let me just -- .

>> Is this better?

>> Yes.

>> Relaxation characterization and I have kicked off a run and it is running now. I will walk you through what's happening. Basically then we create a quantum program circuit. And then we kind of specify what device we want to characterize. So I'll choose IBM QX5 device and one of the Qubits, I will choose Qubit No. 2 at random. Now what we can do you can actually go ahead and read the result of the latest calibration that is online. This is what I'm doing. I am making an API call to IBM queue experience and the latest algorithm the relaxation rate was measured to be 33 microseconds. I run my own benchmarking to see what this relaxation rate is. I created

a quantum circuit and this is -- as I mentioned I'm doing that -- a series of those measurements with variable weights in between.

>> It is fuzzy for me again. Is it for Clyde?

>> Yes. The screen shrunk again.

>> Maybe you want to stop and start sharing again.

>> Is this better?

>> No.

>> Let me see.

>> Just stop and share it.

>> Yep that's better.

>> Yes.

>> Better?

>> Yes.

>> Okay. Great. So this is an example of what one of the circuits that I'm running, but the Qubit 2 in excited state and wait and then measurement it. Now I have to kick off a run. Now because we are running a lot of circuits because I am trying to measure a variable points, and then plots the relaxation. So let's go and move -- as we are waiting the job has to wait in the queue. And then when it gets on the queue we have to wait for a few minutes for the job to run. So --

>> Fuzzy again. I don't know if it is your network bandwidth. I don't know what's going on.

(Talking at the same time).

>> Use the tab.

>> Is this better?

>> That's better.

>> Okay. So this is -- this is a page on the IBM queue experience website that has a lot of information about different devices. 20 Qubit device, commercial device and 16 and Qubit devices that are available for free and has information about the topology and later on information about the errors. So everything is transparent. Everyone is reported in the open. We have gate errors for each of the Qubits. And we have errors when we try to do two Qubit gates. For example, Qubit 1 and 0 and do CNOT and this is the average error rate. And then -- so now I'm looking at the IBM slide and this is the one I am running. I am running on Q2 and trying to characterize Qubit 2 and Q1 is this relaxation rate. In the latest calibration this is data we got. The latest was 33 microseconds. So let's go back and see I'm guessing it went fuzzy again.

>> Yes, it did.

>> Better?

>> Yes.

>> Okay. So again we are waiting for the results to come back. I have run a lot of circuits. And essentially what we

see is that that exactly as we are waiting more and more for the results -- sorry. As each of these circuits gets larger and larger, and you measure, then you are relaxing more and more.

Okay? So let me go back to my presentation and -- this might take a few minutes. I guess the queue is busy now but we'll come back to it.

>> Correlation between the -- return of -- the return of circuits and the fuzziness of the screen, it seems like when it is computing the screen gets fuzzy.

>> Computation is being done on the cloud. But let me -- I think whenever I switch tabs that happens.

>> Okay.

>> And now --

>> That's better.

>> I switched and I need to display. So in this presentation basically what I have been trying to do is show you there are many different things you can do using access to real quantum computer. You can run problems -- run algorithms that are solved faster than classical counterparts. And you can characterize the devices. And everything that we are doing is really -- we try and keep it in the open. So Qiskit all the code is open and we are getting a lot of feedback from the community and all -- we are also getting a lot of contribution from the community. A great way to start, we have extensive documentation on Qiskit.org on the website. We have a lot of groups, similar to notebooks that I showed. A lot of other good notebooks that do -- basically do a variety of different tasks and Jupiter notebooks. And we contribute, anyone who can contribute a good notebook of examples of tutorials to do that. We are getting a lot of contributions.

Four to seven different people have contributed code right now and essentially if you can think of code in the future for Qiskit we encourage you to report it. Public command channel to engage with the community and to announce updates and things like that. And if anyone has questions about general quantum computing you are welcome to come and ask here. So I will switch back. That's basically the end of my presentation. I think -- I'll come back and take any questions if anybody has questions.

>> It is clear, no?

>> Sorry. Yes, so let me take questions. And I think there might have been a problem here.

>> Okay. Well, thank you for the presentation, Ali. I am sure the chat, (I can't understand them). I will ask if there is any questions that remain in the chat. And you can call out live to you. Do you have any questions?

>> (Off microphone).

>> We answered all the questions.

>> Yeah, I think we were able to answer all the questions.

>> Okay.

>> That's really great. And I got a question myself. So that -- the (inaudible) that you showed was very interesting. If someone gets involved and wants some algorithms would there be some upload to try out?

>> Yes. So as mentioned these tutorials that we have, we have a lot of conical textbook examples. You can go and run the notebook and see the results and tweak it a little bit and play around and see how it affects the results. Yes. So the notebooks are very -- are a great place to start, to learn about quantum computation in general and Qiskit in particular.

>> Okay. So if somebody wants -- how -- how for somebody to think of a problem in control (inaudible) quantum computing?

>> So that's a good question. I mean designing new algorithms for solving problems is hard. Because as I -- as I showed you you have to really try to exploit some structure in the problem to find algorithms that are useful. It is a research area. If anyone wants to get in that that research area it is a great place to get in to. It is very rich and new algorithms are discovered for doing quantum chemistry, optimization, and things like that. But it is not as trivial as thinking about a problem and just bringing it up. You have to really find a good algorithm for it first.

>> Yeah. Now my last question. From the algorithm that you showed in examples that fall with the others?

>> The Bernstein algorithm?

>> Yes.

>> It is sort of created to -- specifically for it to be hard for classical computer and for quantum computer. So it is not very practical. As I mentioned I chose it because it is a very easy algorithm to understand and show you difference. It illustrates the difference between quantum information versus classical information. Now there are shortly after the Bernstein algorithm came out very useful algorithms. For example, integer (inaudible) is a very useful algorithm or algorithms for chemistry. They are very hard. There is useful algorithms. This is a pedagogical example.

>> Got it.

>> Practical to (inaudible). So I thank you for that and I would encourage people to go back to this presentation and (inaudible) by (inaudible). And so if there is anything you would like to discuss in the future, let us know. We would like to host another of those quantum computing series. So thank you. And you are welcome to come back any time.

>> Sure. Thank you very much.

>> So what we have for next --

>> I forgot to mention one thing. I will put the notebooks online so maybe you can link to them. Show the algorithm and relaxation example, you can -- yeah, you can download Qiskit and run them yourselves.

>> I am sure that people will pick up the offer. Next week we have two tech talks. They are about 20th of March, on Tuesday we have the first one that is how you to deploy an AKKA cluster to Kubernetes and then followed by (inaudible) from the lab in California. What is archived and you can deploy cluster using Kubernetes. That's for Tuesday. On Wednesday we have also a guy from Mike's team who also talk about AKKA. Tell you about how you can -- you can use AKKA to really -- to build last responses application. And all you need to know about AKKA. So -- talk to you about how you can view these applications in AKKA. So next week is AKKA week. So on the first Tuesday we can talk about how you can deploy it on the cluster and then go in to deeper dive the next day. So that's all for today. Thank you for attending. And thank you Ali for giving this series. We will talk to you next week. Thank you very much.

>> Thank you.