Code Together Podcast
Episode 4: Bridging Performance Abstraction Layers in Data Science
Host: Nicole Huesman, Intel
Guests: David Liu, Intel; Peter Wang, Anaconda

Nicole Huesman: Welcome to *Code Together*, an interview series exploring the possibilities of cross-architecture development with those who live it. I'm your host, Nicole Huesman.

The first industrial revolution was powered by coal and steam, the second by electricity, and the third by electronics and IT. The fourth, networking and the exchange of information between human and machine, is fueled by AI, machine learning and data science.

Today, I'm happy to welcome two guests to the program who are no strangers to data science. David Liu, AI solutions engineer at Intel. Hi David.

David Liu: Hi, Good to hear from you.

Nicole Huesman: And Peter Wang, CEO and co-founder of Anaconda. Welcome, Peter.

Peter Wang: Hi. Thank you for having me.

Nicole Huesman: Peter, let's dive in. It's an exciting time for Python. Stack Overflow just reported Python as the number one language among developers in data science. As a true pioneer of the language, can you talk about the evolution of Python. Why has Python been ascendant, particularly in data science?

Peter Wang: The interesting thing about Python is that it's one of the very few popular mainstream languages in use today that can actually trace its legacy to the idea of being a teaching language. In the last 30 years or so, as software development has become its own profession, we viewed programming languages as being an expert tool for software developers. But early on, it was definitely viewed that, here are these powerful thinking machines and we should teach everyone how to do it. So the creator of the Python language, Guido van Rossum, made a language that was easy for people outside of just traditional software development communities. And then as Python became more popular, I think in the late nineties and early 2000s, we found this community that's adjacent to professional software development, they had started adopting Python, and that is the numerical and scientific computing community. And those people have been programming always as part of their jobs, but none of them are professional software developers. So Python started getting organic adoption in that space, started producing better and better tools, and started becoming really quite powerful for doing lots of big data crunching, large numerical simulations. By about the 2010 timeframe, I and some of my colleagues who are working in this space, we saw that the time was right for Python to go into tackling business data analytics and being used by a broader community outside of what had traditionally been sort of at the niche of so-called technical computing. And so we made a push for that. So I started the PyData organization just almost as a branding exercise. The tools are the same as what we use in the scientific Python community, but a lot of people don't think of themselves as scientists, but if you say PyData, they think, “Oh! Data science or doing data analytics. Well, I'm a data analyst, I could probably use it for that. And we just happened to be at the right time that we had sort of a peak of big data. People had put a lot of data into data lakes and Hadoop and things like that. You could use a few lines of Python to ask very interesting questions on terabytes of data. So I think the reason it's gotten to this point is just a series of maybe accidents, but they're happy accidents and they all lined up to now make this easy-to-use language that many people—not just software developers—can use, and they can ask questions of data that are then executed in very high-performance fashion.

Nicole Huesman: And David, you're really no stranger to Python either. Can you talk about some of the new developments in this area and how they benefit data scientists?

David Liu: Yeah, no problem. So closer to about 2013, I really started working in this space, and from a data science perspective and machine learning perspective, the tools started reaching their peak efficiency. A lot of the technologies that have been really pioneered through Python have started from the needs that Peter just spoke about. So some of the new things that Intel has been really pushing in this space to really help with these types of initiatives are the Intel Distribution for Python, which is built on Anaconda and Conda. The Intel Scalable Dataframe Compiler, which allows you to compile down a dataframe in Pandas or other types of numerical arrays into LLVM calls that are based through Numba. We have our AI Analytics Toolkit, and XGBoost optimizations. And a lot of the vectorization capabilities that have come through the newer processors have been put front and center for the optimization work that the teams at Intel have been working on to give to the users of Python and NumPy and SciPy, and a lot of the other data science specific frameworks. So really—because Python has been the front interface to machine learning, data science and AI—Intel has been very heavily invested in enabling that ecosystem with the optimization work, or through our partners.

Nicole Huesman: Peter, with Anaconda's long history in data science, can you talk a little bit about the business implications and what you're seeing happen in the industry?

Peter Wang: I think businesses are starting to wake up to this idea that the era of retrospective-looking reporting analytics, that era is drawing to a close. And the businesses that succeed moving forward are going to be the ones that are best able to do predictive analytics. Heading to the new world order, everything is online. Data sets, consumer sentiment, weather patterns, logistics. Everything is changing all the time, and that creates an opportunity. So if you, as a business, don't lean into that, don't lean into more advanced sense-making, more advanced prediction, your competitors will, and they will eat your lunch, right? So I think every business now, what I'm seeing differently now than when we started 10 years ago, is that businesses actually are starting to say, well, okay, no one's going to make the argument that we can just sit still. Everyone is leaning into this. Businesses are seeing that this really can't happen unless you have digital transformation, unless you have business transformation. And that's very promising to me.

Nicole Huesman: Absolutely. There's this massive scale and this need for low power, performance-intensive computing. It's almost like back to the future, back to the past, right? So can you talk about where performance really matters again?

Peter Wang: When it comes to machine learning and the analytics and the numeric stuff, well, it starts to matter a whole lot because the kinds of questions you ask, they're all numerically intensive. If you do the right thing numerically, it can be an order of magnitude. It can be 10X better or faster. And it comes back to that competition aspect and the competitive advantage. If you and your competitor both have similar amounts of budget, but you have software developers that don't know how to harness the best technologies and the best software approaches and your competitor does, they're going to be able to ask questions either 10 times faster or get results with 10 times better fidelity, or use a lot less money to get the same results that you're getting. And that does start to matter at scale. So I think that's the kind of thing that machine learning is one of those things where you could throw infinite amounts of money at it. So you have to start caring about the efficacy of your dollars per hour. Really, your results per gigawatt or kilowatt that you put into this thing. So yeah, I do say that performance matters again. So I'm really excited about that. That's just a super cool thing to be able to work together with the Intel folks to make our compilers better, to make the whole stack better. That's just a really cool thing.

Nicole Huesman: So David, can you talk a little bit about how Intel is responding to these needs?

David Liu: Going back to, you know, this concept of performance matters. When we started looking at different problems within the various domains and algorithms that are prevalent in machine learning, one of the things that we came to the conclusion of is that new types of accelerators and architectures might need to be explored in order to access this type of performance. And so, you know, from the new instruction sets such as Vector Neural Network Instructions (VNNI) or FPGAs, really that exploration, you started seeing it in Intel's portfolio where we're starting to really explore what else is out there. How can we get these other types of workloads faster? And you're starting to see us experiment with a whole slew of different architectures within a portfolio. And that's been kind of a reactionary part of this explorative process. We need to get farther, we need to find the performance again, we need to really dig down and see what's out there. That's why Intel's movement towards this increased portfolio size has been, for that particular reason. And then how do you get it into the hands of the people?

Nicole Huesman: Yeah, absolutely. And Peter, can you talk about Anaconda's current focuses in this area and how you're investing and inventing at the deep part of the stack?

Peter Wang: Yeah, yeah. We have a lot of cool stuff going on. So, you know, I think I mentioned some of the compiler work that we've been doing. And Intel's helped provide resources there, and we're trying to make it so that data scientists can express the fullness of the questions and the transformation they want to have on their data without being restricted just to some of the bulk operations that some of the libraries, you know, currently support. So with our Numba compiler, with some of the optimization work that's been done by the Intel folks, we're now able to give a tool to your data analysts and data scientists to really go much deeper to be able to transform and write loops and write different kinds of expressions that will optimize down and run really fast on Intel hardware. One of the things that we see right now that's kind of problematic, that I see at least, is that people are generally aware there's, you know, higher levels of performance available, but they see them as being kind of exotic. And so they will tend to stay close to the familiar API, some familiar libraries until they run into some ceiling of performance and then say, "Oh, okay, now I've got to go and learn this other thing," right? And so people have a reticence to learn these new things because, well, frankly, they're busy and their time is worth money to their employers, right? So you can't really fault them for that. But what we like to do is, you know, get people so they don't feel like they have to go to C++, or have to go and learn some exotic new things. They can use Python, use some of the bridging layers that we're working with the Intel teams on, so that they can keep building with Legos. They don't have to go and make exotic 3D printed custom things. They can keep working with Legos, but we're sort of getting them titanium Legos. We're giving them Legos that have some real juice to them. So they can use the API that they're familiar with, they can use the tools that they know and love, but they can just get performance. I won't say for free. There's always a little bit of work, but it's not a lot. It's not like throwing everything you know away, and learning a whole new tool set. So I think some of the things that we're doing with parallelism, with Dask, some of the things that we've been doing on the Numba side. You know, we have to work with folks like Intel to make this work well. It's simply too much work for a company like us to take on and working with the Intel engineers, they can surface those pieces from the ground up and we can meet them in the middle and say, well, here's what we need from the top down. Let's meet in the middle and provide nice sets of intermediate interfaces so we can lower those computations down and have them be scalable, have them be performant. That's just a really neat thing. It is a team effort.

Nicole Huesman: Let's talk a little bit about how Python plays with things like DPC++ and oneAPI when we talk about heterogeneous computing and cross-architecture.

David Liu: One of the more interesting aspects of being in the Python space, or being a developer or user of it, is finding ways of actually getting Pythonic access to hardware. And that's historically been a very difficult aspect. You either have to directly call it out, or you have to find ways that packages are built with that driver already built in, or you're depending on the run times to get there. And with heterogeneous computing that just amplifies the challenge, right? So now you're not just looking at CPU and GPU, you're looking at FPGAs, ASICs, whatever else is coming down the line. And one of the links, if you've ever worked really in this space of enabling these types of frameworks, is you really have to drop down to the C++ level. In this case, the DPC++ that one API is providing is that link to get the performance from that heterogeneous hardware. And that's been one of the more interesting pieces that I've been looking at, as both a developer and a user, is where, as a developer, can I actually make this available for packages, and how can I then give this out to other data scientists or other scientists, in general, to be able to access this hardware and Peter's titanium Legos? Like how do you build a titanium Lego from these bits?

Nicole Huesman: Peter do you want to talk a little bit about that?

Peter Wang: One of the things that I would say about that is that with DPC++ and the oneAPI stuff, what Intel has done there is something I think that essentially any hardware vendor that's to be successful moving forward is going to have to do because I know I talk about Python a lot for obvious reasons, but it's not just about Python. It's really anyone who's programming at a higher level, and it could be JavaScript, right? It could be Python, it could be certain classes of C++ developers, even. They're just at a certain level in the abstraction space. And right now, because performance matters again in computing, all of the chip manufacturers are looking for ways to develop more innovation and innovate in places like high performance, low power—in-the-field kinds of things. So we talk about IoT, you can talk about edge computing, all these things. All of those different kinds of hardware profiles will lead to an explosion of complexity and interfaces, and you run the risk that you make all these chips and nobody can program for them. So once you have this variety and heterogeneity, it has to kind of come up and roll up into some kinds of plateaus of abstraction. And so, I think that what Intel is doing with oneAPI makes a lot of sense, and we're trying to build a bridge from that level of abstraction to the next level of abstraction, which is where the next plateau is, right? Which is kind of that Python NumPy layer. So I think that it's sort of a natural feature of the evolutionary landscape and the innovation landscape.

Nicole Huesman: So Peter, as you look forward, what are you most excited about.

Peter Wang: I feel like we're just still getting started with all this Python stuff. Even though, technically, the scientific computing and emerging landscape of Python is 20 years old at least, and certainly the Python data stuff is 10 years old, I feel like we're just getting started. Because where I think the stuff gets really beautiful is when the average person in business can reach for these kinds of tools. They're not afraid to ask questions about their data and about the models. They're not restricted just to spreadsheet thinking. They're not restricted to just looking at a PowerPoint saying, "I guess I'll accept that chart at face value," right? I look forward to having data literacy in the hands of the vast majority of people because I think it's a deeply empowering thing, and that's made possible by, you know, the work of everyone up and down the stack. People at Intel, people like us, the large ecosystem of open source innovators out there. It's really about technology empowering people to make better decisions and to feel more empowered and to do better thinking.

Nicole Huesman: David, how about you?

David Liu: Well, actually, I do have a question for Peter in this case, actually.

Nicole Huesman: Okay, great.

David Liu: Well, you know, in talking about this future, so you're talking about data literacy and kind of a productivity first focus, right? When you see the different types of hardware being developed out there—like, some of the ones I've mentioned today, FPGAs, ASICS and GPUs, right? Where do you see the average user really trying to put their head space when working on a problem. Will they be looking at the problem in conjunction with hardware or will it be an abstraction layer that's higher? Where do you see that workflow coming to?

Peter Wang: So I think where that workflow ends up in the long, long term is that we have to be able to synthesize up from the hardware and expose the costs and the time—so, the dollars and the wall clock tradeoffs of certain kinds of operations. And being able to surface that as an option for the user. You know, I think now people generally assume you're given a language, you're given a framework, you're given a computer, and you ask questions that fit in the computer, right? It's like, I give you a car and then you can only go so far in the car. I think people in the future, all sorts of computing will be available to them. All sorts of typologies of compute and types of processors will be available to them. So then it's like people are looking at a selection of cars and a road atlas and saying, “Which car do I want to drive down which roads to get to where I want to go?” That's going to have to be the thing. And it's a different kind of mindset than right now where it's everyone, I have a work laptop or I have access to this cluster, and that's all I've got. We're already seen people move out of that mode. They're renting supercomputers by the hour, right? They've been doing that for years now, so I think we're going to get people thinking about the data, the questions they want to ask, and they also need to surface up the cost of compute—time as well as budget—put it all together, and make a plan. And this can be not so onerous of an overhead as this. It could be something much more dynamic, right? But something like that, I think is going to be the mode people are going to have to be in. Just because the opportunity space involves all of these things, right? If you want to spend more money, you can get certain things faster. If you want to spend less money, then you have to work a little harder, right? That's the mode people are going to be in.

David Liu: You know, this concept of a data parallel Python, I think that was tossed around a few times from various PyCons, SciPys, other types of events. Peter, I was really wanting to see your ideas, especially on the topic of oneAPI and with all the different types of DPC++ optimizations that we have available. Where do you see that intersection coming to? Is it close? What needs to be pioneered to get something like that off the ground? Is it mature enough for that?

Peter Wang: Well, I think we have data parallel Python at different levels of the parallelism and complexity tree. It's not unified. And I think that the unification is going to be difficult within the confines of the existence, syntax of the language. And that's a big statement. That's a mouthful. So let me just say that again. As long as Python looks like what it does, there's only so much you can do with data parallelism.

David Liu: So the API being more restrictive and the NumPy mentality of making a raise, or is it just the productivity level?

Peter Wang: No, I think the concepts are fine. I think that within each of the sort of zones of data parallelism that we have, we can make some improvements for sure. And we do some unification across them. That's going to involve compiler work as well as data structure work, and we'll get there. For me, the promised land is something like a combination of a minimal Python-ish thing that looks like Python, it looks like Julia, it's all syntaxically about the same. Some of the concepts that are data-intensive that come from LINQ, or Language-Integrated Query, you know, the stuff from Microsoft, maybe some ideas from F# there even, and Julius functional. I mean, there's a lot of overlap in the ideas. When you get to the data parallelism stuff, it looks a lot like functional programming, right? And then, the beautiful set of ideas that have been lost in the mainstream that come from Arthur Whitney, from K and from the CAP native programming folks. So the new stuff he's working on, that's where I think the next big revolutionary change will come in. I think evolutionarily, Python, there's spaces that we will evolve into, but we're burdened by our success. There's just millions of lines of code that works right now. You can't break with that syntax very much, right? So I think that Python itself and data parallel Python as a concept, we'll get to certain levels, but I think there is sort of a ceiling of evolution. The tree will only go so far, as long as it's a tree that's made of wood, right? I think that's kind of where we are. The fundamental limiting stress tensor of Python is the syntax. And once you change the syntax enough, it's no longer Python, right?

Nicole Huesman: Excellent. Thank you both.

Peter Wang: Thank you. This was fun!

Nicole Huesman: Data science is such a quickly evolving space. We look forward to having you both back on the program. For all of you listening, thanks so much for tuning in. Let's continue the conversation at @oneapi.com. Until next time, thanks for listening.