Andrew Chien Speaks Out on the Impact of Current Trends in Architecture

Marianne Winslett and Vanessa Braganholo

Andrew Chien
http://people.cs.uchicago.edu/~aachien/

Welcome to ACM SIGMOD Records series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today, we're at the 2017 SIGMOD and PODS Conference in Chicago. I have here with me Andrew Chien who's a professor at the University of Chicago. Andrew is an ACM fellow, an IEEE fellow, and a fellow of the American Association for the Advancement of Science. Before joining Chicago, Andrew spent five years as the vice president/director of Intel Research. Andrew's Ph.D. is from MIT.

So, Andrew, welcome!

Nice to be here.

You're from outside the SIGMOD community, a highly successful systems researcher. From your outside perspective, what words of wisdom do you have for us data and information researchers?

That's a very interesting question, Marianne. It's a high bar for an outsider, actually.

Well, let me narrow it down a little bit. Can you tell us what, from your perspective, either we've done wrong so far or important problems we haven't given enough attention to yet?

So, yeah, I think the SIGMOD community, actually, is an amazing, dynamic community, and as I look across computer science, several of the great strengths of the community is its deep attachment both to applications as well as to underlying technology and how that changes the game periodically. I also think – I mean, it's pretty obvious to everyone that the growing importance of data and computing systems writ large puts the SIGMOD community sort of in the driver's seat for all kinds of secular change in computing systems.

So, the strengths of the community, I think, are evident, and if I had any advice to give, what I would say is that it seems to me that the SIGMOD community is unique in Computer Science in that it has thought of computation and data in an integrated form. That is, you talk about queries and computation and transactions with a data model, with a schema, with some notion of the structure of the data. Then, you talk about consistency of those kinds of data collections and so on, again in that integrated view of consistency with respect to transactions or computations or workloads or even applications. I actually hope that and would wish that the SIGMOD community would actually try and take some of its learnings and not straightforwardly transliterate them into other fields, into other areas of computer science, but I think that the community has a lot to contribute to the broader space of computing systems.

I spent many of my years as a computer architect, and I can tell you that the issues of data locality, data orchestration, efficient location of computation, efficient parallelization are all fundamental problems in computing systems of every type today, and I can't think of any community that understands better how to combine data organization with consistency with computation than the SIGMOD community. So, I think there are vast contributions to be made by the community.

I think I understand what you're getting at, but can you give me an example of something you could imagine us deciding to do?

I don't know if it's a collective thing. It's sort of can some members of the SIGMOD community that have this knowledge decide to forge out and create new kinds of computational systems that go a bit further from just the data part up into – you're starting to see this, I guess, in certain kinds of – learning systems. Perhaps some examples of systems that are data-centric but actually are much more encompassing than just the data management tasks, so reaching out to include much more complex forms of computation, much more compute-intensive kinds of data transformation and analysis as an integral part of what you might think of as the data management system.

Does Spark Streaming count, or Storm?

I think those systems are examples of movement in that direction. I don't know that they go very far. What do I mean by that? I think that, as I understand those systems – we're fortunate to have an expert in those systems at Chicago now (we have Mike Franklin there now) – there's more of a divide, actually, between the parts of the system around which you have strong properties of consistency or the like and the part that's doing the computation.

I think those systems, as I understand them, are behaving mostly like integration platforms, rather than trying to extend the semantics and the properties and the capabilities and analysis of the underlying database system or data management system up into those computational domains.

So, you'd like to see stronger, for example, consistency guarantees or fault recovery properties in other types of systems as well. Are you thinking more up to new applications or down, putting them at lower levels of the system?

I think it's both directions, actually. I think that you'd like to have those nice properties evident at higher application layers, as you suggested. I think it's also true one of the great successes of the database community is concurrency management as embodied with this notion of transactions, data orchestration, very efficient data movement, and aggregation of different operators, standard query across the memory hierarchy and understanding how to organize that data and that movement and how to represent that data.

Those kinds of things, I think, are in their infancy in computer architecture and in systems, and when you look at the proliferation of nonvolatile storage all over these systems, new kinds of memories – and I'm happy to talk more about that – the opportunities to do

intelligent things are vast, but the frameworks and the understanding for how to do things that are not just locally intelligent but globally intelligent, I think, are lacking.

> ***[…] the issues of data locality, data orchestration, efficient location of computation, efficient parallelization are all fundamental problems in computing systems of every type today, and I can't think of any community that understands better how to combine data organization with consistency with computation than the SIGMOD community.***

What future trends in the lower levels of the system stack in the hardware world do we need to know about that we don't already know about?

Gosh, let me try to address that by rattling off some of the things that I think that are exciting that are happening in the hardware world right now. I think that, starting from several years ago, it's become increasingly clear that specialization is the order of the day. So, what does specialization mean? Well, I don't mean GPUs. GPUs are now a 10-year-old specialization trend. But now, I think you're starting to see pushed both from the bottom – that is in mobile devices – but also from the top – large-scale cloud systems – customization of architectures for higher performance, for energy efficiency, for density, lower latency.

So, that's beginning in the forms of things like FPGAs (Field Programmable Gate Array), Microsoft's Catapult project, and Google's TPU (Tensor Processing Units). TensorFlow processing recently got a lot of press. But it's also true that anyone working in a large-scale vertical application now has the means and the capability and the economics, actually, to do architectural specialization.

So, what does that mean? I think you can expect to see accelerators for almost any focused, large-scale transformation you might want to do. There are the simple things like compression and crypto, which everyone is aware of, but you might imagine indexing, certain kinds of parsing tasks, certain kinds of data representation tasks. Transformations will be accelerated by new kinds of architectural features in the future. That has implications for software. That has implications for query optimization, perhaps, and the costs of different operations, and perhaps it has implications in the long-run for how people formulate their applications.

Do you mean they're accelerated by being put on FPGAs and TPUs, or do you mean some other way of accelerating?

Oh, I view FPGAs as a halfway house. It's a way of balancing the cost of putting custom silicon into these systems, yet preserving some of the breadth and flexibility by allowing it to be reprogrammed. I think what you're going to see over the next couple of years is more and more custom silicon, hardwired silicon, being put onto these chips as SOCs, as instruction set extensions. Oracle has already done some interesting things in their Sonoma series of processors, Sparc-based, that didn't get, I think, a huge amount of press because that's a narrow kind of exposure these days.

But I think those things are happening because they give benefits of 10x, sometimes 100x efficiency in those tasks, and that's just getting too big to ignore, and we have so much silicon. The latest chips being released have 20 billion transistors on computing chips, so there's a lot of room to put interesting stuff on there.

So, for researchers, unless you're in a company who happens to produce these specialty chips, how would one do research on that topic?

It's a good question. I think FPGAs are a vehicle for doing research on these kinds of things, and there are systems available like the NSF Chameleon system that has a set of FPGAs deployed where you could do experiments. Amazon and the other cloud guys.

Whoa! The FPGA is no problem, but the actual custom hardware is going to be way faster than the FPGA. That was what I was getting at.

I think that's a difficult challenge. So, when I was at Intel, the timescale from conception for new architecture features to them appearing in silicon in products was four to five years, but increasingly, in this new world, we're seeing that distance being more like 18 months, 24 months, something like that. So, I think it's possible to work with folks with FPGAs and with simulation, and then, shortly thereafter, with actual hardware deployed at scale in these cloud centers or the like.

So, I don't have any magic bullet that makes it possible to have this time machine and deal with these future hardware systems today before they exist, but if there's any encouragement, it's that the timescales are much shorter. It isn't the time from when you're a graduate student to when you're a full professor when the ideas actually get out there. It's more like two or three SIGMOD review cycles.

Okay, sounds great.

There's a couple of other things on that front. I think there's a number of other trends that are happening that also are disruptive and play to this idea that folks who've been thinking about computation coupled with data coupled with efficient data movement have a lot to contribute.

First, the memory hierarchies in these systems are getting much deeper, and that's despite the fact that processor clock rates aren't getting any faster. What's happening is that multi-core requires more and more bandwidth, and in order to meet these bandwidth needs, people are moving to exotic stacked DRAM kinds of technologies.

Perhaps you've heard of HBM or HBM2 or HBM3. These are stacked DRAM technologies that allow you to get into the terabytes range of memory bandwidth. These represent a super-fast but small memory hierarchy. So, you might have 16 gigabytes, 30 gigabytes, those kinds of numbers, and then, beyond that, you can have the super-large DDR kinds of DRAMs, terabytes or those kinds of things, but you're not going to have terabytes of this super-small, fast memory.

So, if you're thinking memory hierarchies go away, they probably don't go away. What you're seeing is one grows, and that means that that gives birth to another smaller one that's above it. These memory hierarchies not only have bandwidth differences, so you might have terabytes per second of bandwidth in this small stacked memory hierarchy. That might actually mean that at the DDR level or in this future persistent 3D XPoint or other kinds of nonvolatile memory, you might have even less memory bandwidth there. You might have 100 gigabytes per second or less because the introduction of a new tier usually for architects is an excuse to reduce bandwidth and performance at lower layers.

So, I think that's a challenging problem. Beyond that, of course, we have the widespread acceptance of SSDs and flash-based things. So, you've got at least three or four interesting tiers of memory hierarchy that seem to be here for the foreseeable future.

Beyond that, there are all kinds of opportunities to do interesting things between them because the gap isn't so large. I think the storage management community, the data community, has been, for many years, to a degree, shaped by the large gap between disk and DRAM.

Not anymore.

And now we have flash and NAND.

We went main memory a while back because key customers tend to have problems that fit in main memory, so you've got to rethink everything from scratch. So, we went to the app route. And caching and buffering is always a popular topic, but I don't know that we've gone for quite as many layers as you've been talking about yet. I don't know.

Okay, well, that's good to hear. So, that's one dimension of what's happening. Another thing that has happened is that in this push to create the DRAM replacement, the hardware technology community has produced a whole bunch of different kinds of memories, and some of them, actually, are quite a bit faster and more reliable and more persistent than some of the technologies being pushed as DRAM replacements.

So, just to hold out a few examples, there's MRAM technologies and other kinds of exotic memory types that, while not cheap enough that you would ever dream of replacing all of your SRAM or all of your DRAM with those technologies, they can be used in spots and different special functions.

So, I know there are some researchers looking at this, but I think there's probably a larger opportunity to exploit those kinds of special memory technologies for narrower uses, perhaps certain kinds of locking structures, certain kinds of logging structures, things related to performance-critical aspects.

Do you think that, in the nodes in the cloud, there'll be little bits of these specialty memories sprinkled around?

I think that's likely, and I think it's also likely you may see little bits of the specialty memory actually integrated into compute chips in the future, so that would be another way it could become generally accessible.

I was going to add one more thing about memories. There's a lot of exciting stuff happening in memory systems, perhaps more change now than there has been in decades. Another thing that's happening is this vision around disaggregated servers or this looser association between memories and CPUs. For a long time, we've had this traditional notion of either pizza boxes – processor, DRAM, and maybe some disk or maybe some other kind of storage and scale out in the cloud – and now, increasingly, you have fat nodes and other kinds of pairings of quantities of DRAM and network-attached storage and other kinds of things.

Sort of the natural and logical extension of that that's being enabled by super-high-speed networks and interconnects is disaggregated resources in the cloud. So, you can imagine large nodes that are close to memory only. You can imagine nodes that are primarily compute and coupled to large shared pools of memory that might be shared amongst multiple different domains of processing.

So, this raises a bunch of really interesting questions about how do you manage irregular memory performance, distributed memories of different capacities, various associations of computing, and all the data locality problems are different now in this space. And I think that we're perhaps well-prepared from the distributed systems and cloud software side to think about those problems, but understanding how to do it well, I think, requires this integrated data model and computation view that has been one of the cores of the database community for years.

Super. What major trends at the application level might have escaped our attention?

Gosh, I'm loathe to presume that they've escaped this database community's attention, but some of the things I think that are big changes, compared to what I hear about the database community focused on, is there's this explosive growth around how data relates to society, that is, geographic, national boundaries, regulatory boundaries, different commercial proprietary boundaries, and so on, and it seems like that's increasingly a fundamental aspect of function and performance of these systems. And while I don't pretend to know everything that's going on in the database community, it seems to me that those aren't traditional foci of the community and I think are important challenges for how this all goes forward.

The second area that I would comment on is there's excitement around IoT – Internet of Things – self-driving cars, any other kinds of network or edge devices that might be high data-rate sources. So, there are a couple different ways to think about those systems. I hear a lot about people thinking about the cloud side or the server side element of those problems, "After I've uploaded a lot of that data, how do I do analytics on it? How to do real-time analytics on it? How do I store and organize it and so on?" But the reality is that for lots of those systems, large fractions of the data will never make it to the cloud. One of the funny secrets of the sensor systems is that most video cameras don't bother capturing, actually, most of their data. Certainly, the self-driving cars, where they're talking about data rates of many gigabytes per second while the vehicles are in operation, imagine multiplying that by tens of millions of these vehicles. It's not hard to figure out that you can't actually afford to capture, network, and store all of that stuff for very long, if it ever gets to the data center.

So, there's a bunch of interesting challenges about how do you do distributed and streaming data analytics, how do you deal with collections that are fundamentally asymmetrically distributed, and how do you host applications in some reasonable programming and performance-tuning model across those very, very complicated kinds of infrastructures. And then, you've got all the security and privacy and governance kinds of questions we talked about before.

You should realize that the process of research is to make a contribution to the intellectual direction of the community. If you're completely aligned with the intellectual direction of the community, you're not making any contribution to the vector. Maybe a small contribution to magnitude, but not direction.

We had some very interesting conversations with a provider here in Chicago that turns out to be the host of a lot of self-driving or connected-car kinds of applications. It's the company that was formerly called Navteq, bought by Nokia and then sold to an alliance of German auto manufacturers. Very interesting company. And the challenge they have is how do they service these different companies that are fundamentally competitors, and of course they all have their own data. They all would benefit from some kind of data pooling because some of the companies are exotic, high-end kinds of firms that don't sell tens of millions of cars a year and don't have the same coverage. Others are that, but don't necessarily have the high-end sensing platforms in their vehicles for economic reasons.

So, those kinds of companies and venues have all of these problems today, and I can tell you from talking to them, they don't have good systems solutions to address the privacy needs, the sharing needs, and the vertical aspects of those systems.

Do you have any words of advice for fledgling or mid-career researchers?

That's an interesting challenge. I guess what I would say is what I've learned over the years is I'd encourage them not to pursue any fads. There are lots of fads.

So, how do you know you're not pursuing a fad? I think you need to make sure that the trends that your research ideas or research direction depend on are fundamental, and you have to figure that out for yourself because the fads won't tell you that. And if you look hard and try to understand what the fundamentals are that are driving the importance of a set of research ideas or research direction, when you formulate your research problems and contributions, you're likely to get resistance from the community.

This is not necessarily a bad thing. You should realize that the process of research is to make a contribution to the intellectual direction of the community. If you're completely aligned with the intellectual direction of the community, you're not making any contribution to the vector. Maybe a small contribution to magnitude, but not direction. So, it's not a bad thing if you initially get resistance, and you have to make the case, and you have to make a justification of the problems and so on that you work on. But I think that if you're persistent at it, then that's the way, actually, to make a long-term impact.

And my experience has been we've worked in areas, and I have to confess that for me personally, at times, we had strong convictions about where things were gonna go, and we gave up too soon. And I think that in hindsight, when the community finally came around to those thoughts and ideas, we could have made a larger contribution if we had stuck with it and really built up that critical mass of both understanding as well as evidence and prominence in that area to drive the community forward.

If you magically had enough extra time to do one additional thing at work that you're not doing now, what would it be?

There are a hundred things, but the thing I miss the most, and actually the reason that I decided to come back to the university, was the opportunity to have more time to be hands-on with technology. So, for me today, that means experimenting with new systems that have been put out. That means writing a little code. I never get to write as much as I would like. Maybe it means doing a little hardware design for me. But designing is as much about exploring all of the exciting new things that the community and the industry is producing and understanding what's possible as it is about designing or coding per se.

If you could change one thing about yourself as a computer science researcher, what would it be?

I think I would say, and it's related to the comment I made earlier about advice for young researchers, it would be patience. One of the things I've learned over the years is that you can often figure out a problem – first, you make sure it's a real problem – and produce academic-quality kinds of ideas and solutions and proof. And then, if you build on that, you have a larger mass of proof. But it takes a long, long time for those ideas or systems to actually find their way into large-scale use. The idea is to find their way into the center of the community and so on.

So, I think that we're all, in the research community, very, very smart, quick people, and we need to understand that there's a big gap between understanding at an intellectual level and acceptance and broad dissemination. So, one improvement for me would be to have the patience, actually, to see these things through, to make sure that inventions that we have or good ideas that we have actually have the maximum impact they can have.

Thank you very much for talking with us today.

Thank you for having me.