Dan Suciu Speaks Out on Research, Shyness and Being a Scientist

Marianne Winslett and Vanessa Braganholo

Dan Suciu

https://homes.cs.washington.edu/~suciu/

Welcome to ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today we are in Snowbird, Utah, USA, site of the 2014 SIGMOD and PODS conference. I have here with me Dan Suciu, who is a professor at the University of Washington. Dan has two Test of Time Awards from PODS as well as Best Paper Awards from SIGMOD and ICDT. Dan's Ph.D. is from the University of Pennsylvania.

So, Dan, welcome!

Thank you.

Can we put a price on individuals' privacy?

Oh, that's tough. Today users give away their private information to Internet companies like Google or Yahoo for free, but this has to change. Users need to have control over their private data. So, it's not clear how this would happen, but at the University of Washington, we've started a number of projects that look into how to price data. In particular, we've looked at how to cover the entire continuum between free and differentially private data and paying the users the full price for access to their private data if they're willing to release that. It is still way too early to see what the right business model would be to allow users to monetize their private data. So far, we do studies in academia. We are still waiting for somebody to come up with the right business model.

We need to train the future scientists [..] in a way that will help them develop their careers for many years. For that, we need a good combination of both theory and practice.

I know it's a hard question but what can you tell me about how we should price it? What kind of methodology should we use?

Clearly, we need to allow users to opt-in and there are some technical challenges here because sometimes the user's decision of whether to opt-in or not might actually reveal something about their private data. This used to be one of the most technically challenging problems in pricing private data. In addition to that, the problem that we did address and that seems more within our reach is how to adjust the price according to the amount of perturbation that we add to the query. So, if the data analyst wants to have very precise data, then he would pay to have the perturbation removed. If he is willing to cope with differentially private data, then he would not pay, and then he would get differentially private perturbed query answers.

So what is my email address worth?

Ah, I think that kind of depends on you, and I don't think your email address is worth in isolation. Maybe it is, but you're part of a larger population. The question is, what is it worth to the analyst to get statistical information about that population? There is a game between users who would like to be compensated for access to their data and the analyst who would like to pay for access to aggregated data over a larger population. So, I don't have a good answer to this. I think that there are developing techniques that would allow the users to choose their price.

It's like there are two categories. There are the ways to contact me and then there are facts about me.

These are indeed different kinds of private information. I don't think we have thought too much about how to distinguish between these kinds of information. So nope, I can't tell you much more about this.

Channels vs. attributes so to speak. Okay! What about the temporal element?

Temporal element in data... you got me here. We know the techniques to deal with temporal data. Maybe you're asking about archiving data and how to keep it for a long time?

Well once something is out there, it's out there.

Ah, how to retract data?

Is that the solution? If I change my mind, update it or whatever?

That is again, a difficult technical challenge. I know there has been research, not inside the database community, but in the operating systems community. There was a project run by some of my colleagues called Vanish[1] that allowed people to produce data that would completely vanish from the cloud after a few hours. So the idea is that you can send an email to your friend, but this email is private, and then it will completely disappear. Every trace of this email will completely disappear from the cloud after a few hours that your friend has read this email. It's a technically challenging problem. Now we also see legislation in Europe that tries to force companies to remove data. I think the jury is still out for what the right model is -- if the data should be kept forever and how much should the users have control over when their data is being deleted.

[1] The VANISH project. https://vanish.cs.washington.edu

Differential privacy also gets more difficult when there's a periodic release of aggregated information compared to a one-time release.

That's a major limitation of differential privacy. They talk about a privacy budget. You can only ask as many queries as is allowed by a privacy budget. There is no story of what happens after this privacy budget has been exhausted. So, then the analyst should theoretically never have access to the data because he/she has exhausted the privacy budget. There is some hope if you also take into account data churning. The data is never static. It always gets updated. Old data is being removed, and new data is being added. So, then your privacy budget would maybe be renewed, but this is still work in progress, and this is not something we do in our group. In our group, we try to solve the problem by allowing users to place a price on this data and that will simplify the privacy budget because now the budget is really a monetary budget (as much money as you have). This is how much you can get access to the private data.

Ok great. Tell me about the work that you received the Test of Time Awards for.

The Test of Times in PODS? There were two papers. Both were about XML. The first was on type checking XML transformations[2]. Here the question that we asked was: If you are given the schemas for your input data and for your output data, can you automatically check if a given XML transformation will indeed map every input data conforming to the input schema to an output data that conforms to the output schema? It turns out this is decidable. You can do this for a non-trivial fragment of XPath. Today XQuery does it differently. It uses type inference, which is not a complete decision procedure. We had a complete decision procedure for a more restricted fragment. So, this was one work.

The other[3] was for a very simple and fundamental problem, which is, you're given two XPath expressions and you need to check if they're equivalent. They may be syntactically different perhaps, but are they actually semantically equivalent? We looked at the tiniest fragment of XPath that is mathematically very elegant to define. That fragment only has "*" (wildcards), "//" (slash slash) and predicates (the "[]" in XPath). What was funny was that for any combination of two of these features, it was known before that checking equivalence could be done efficiently in polynomial time. What we showed is that when you throw in all three features, then the equivalence problem becomes co-NP-complete. And that was an interesting insight because it tells you that XPath is as difficult to check for equivalence as arbitrary conjunctive queries, for example.

Interesting. So both of those papers were on XML, and after that, you moved to working on privacy and then to probabilistic data and from probabilistic data to data markets. How do you choose your next topic and the timing of the move?

This is actually quite hard, but as researchers, we need to watch technology trends and application pulls. The world changes. So in the 90s, the web was new, and for the first time, people discovered that they could share data. They could exchange data. XML was actually designed by people working as a document community. So, they designed XML thinking of it as a document. I think the database community should be credited for showing how XML should be thought of as data and as a data exchange format. The research problems that emerged were fun, but they were not particularly deep. I would say that they are largely solved by now.

But in the meantime, we got new challenges. People started to realize that by exchanging data and having access to data, you need to worry about data privacy. Also, much of the data is uncertain, so data privacy and probabilistic databases emerged later as new challenges for data management. The technical questions underlying these challenges turn out to be much harder. None of them is solved. I actually have my doubts whether privacy can ever be solved in the way in which the academic papers present it. I am a little bit more hopeful that adding a price to the data might lead to a practical solution. For probabilistic databases, they are equally technically challenging, but at least here we're not the only ones looking for solutions. The knowledge representation community and the machine learning community are very hard at work at trying to solve the same challenges we face in probabilistic databases, which is probabilistic inference.

There aren't any commercial probabilistic database systems yet. Do you see a killer app that will bring them into practice?

This is a good question. The most interesting application that I see is that of information extraction. There is a lot of data out there, but it is not in a query-able format. Information extraction is by now a mature field that takes unstructured data and converts it into

[2] Tova Milo, Dan Suciu, Victor Vianu. Typechecking for XML Transformers. PODS 2000, pp. 11-22.

[3] Gerome Miklau, Dan Suciu. Containment and Equivalence for an XPath Fragment. PODS 2002, pp. 65-76.

some structured format, but the output is often probabilistic. The information extraction tools are never 100% accurate, and they produce relational data that is probabilistic. A large example that I know of is Google's Knowledge Vault, which is separate from Google's Knowledge Graph. So, Knowledge Vault has about two billion tuples (two billion triples), and they're all probabilistic. The probabilities of these triples are pretty much uniformly distributed in 0-1. So you can find almost certain triples, and you can find almost junk triples inside, but there's a lot of rich information in that data that, hopefully, with techniques from probabilistic databases, we can query. There are other applications beyond that like record linkage, de-duplication, querying anonymized data that can be viewed as a probabilistic database. Actually, the other day, I was listening to a SIGMOD talk[4] that described a very unexpected application of probabilistic databases. They wanted to do bootstrapping, and they realized that in order to do bootstrapping for some SQL queries you don't need to resample hundreds or thousands of times, and re-run the same query over and over again. Instead, you can view the original data as a probabilistic database and then simply run the query as if you were running it over a probabilistic database. It's a very interesting and unexpected application of probabilistic databases, and I think we'll see more of these in the future.

Someone commented to me that in each of your projects you combine theory and practice perfectly. Is this always one of your goals?

Yes, it is. I believe that in academia this should be a major goal. We need to train the future scientists, and we need to train them in a way that will help them develop their careers for many years. For that, we need a good combination of both theory and practice. I also find that the most difficult theory questions are those that are grounded in practice and that the most interesting systems are those that have a strong theoretical component. I can justify research that is purely theoretical or purely systems oriented, but the most interesting ones that I saw are those types of research that combine both.

Does Big Data have any formal foundation?

Ah, Big Data does need a formal foundation, but I don't think that the jury has settled yet. The industry describes Big Data through the three V's: high volume, high variety, high velocity, but this does not set the research agenda for our community. We have dealt with these qualities of the data since the beginning of databases. We have dealt with volume. Variety means essentially semi-structured data and other data formats, and velocity, well, data streaming has addressed velocity for more than ten years now. So, we need to look for different attributes of Big Data that justify and inform our research. In my view, these attributes would be: (i) massive parallelism -- how do we do query computation over hundreds or thousands of servers; (ii) extending query languages, adding recursion, or adding some capabilities to deal with linear algebra in addition to standard relational query processing over large data; and (iii) the third attribute would be less well defined, but people should look for some kind of techniques or tools that would allow users to explore the data, maybe to understand it, maybe to find explanations, maybe to approximate query processing over very large data, or maybe produce graphs quickly and analyze and interact with those graphs. So these are the three attributes that I think should inform researchers on Big Data in our community.

[...] the most difficult theory questions are those that are grounded in practice and [...] the most interesting systems are those that have a strong theoretical component.

For that second one: adding linear algebra. It sounds like you're moving a little in the direction of SAS or other statistical packages. Are you talking about turning it into a data mining type of thing? And you also said on top of SQL. You didn't say on top of MapReduce.

No, I really meant on top of SQL, on top of relational languages. I think they are here to stay. People have tried to replace them for many years. They are founded in something very principled, which is first-order logic and they're just irreplaceable. So, to this, we need to add the capabilities that people need for large-scale analysis today, which is dealing with linear algebra, doing some machine learning, or some statistical processing. It's interesting, some of the research papers, in fact, the best paper in SIGMOD this year (2014), specifically addresses data management issues

[4] Kai Zeng, Shi Gao, Barzan Mozafari, Carlo Zaniolo: The analytical bootstrap: a new method for fast error estimation in approximate query processing. SIGMOD 2014, pp. 277-288.

in System R[5]. So here we go. We feel the need to integrate linear algebra operations in data management.

What kind of data exploration tools do you have in mind?

We put our money on causality and explanation. We want to allow users to ask questions like "why does this graph have a dip here?" Or "why does this graph increase when I was expecting it to decrease?" The system should hopefully either find some records in the database that are most likely to explain that behavior or find some predicates, some groups of records (some classes of records) that explain that behavior.

Can we differentiate between correlation and causality?

There is a huge debate about this, and essentially the consensus is that you cannot deduce causality from correlations, but people have moved past this general principle. Judea Pearl wrote a very influential book[6] about 15 years ago on causality, and essentially the approach is that you would write down the causal path in your problem. In Judea Pearl's book, causal paths are like graphical models, but in databases, it is even easier to see what they are. They are like foreign keys. Every foreign key can be thought of as a causal path. If you remove the record to which this foreign key points, then implicitly you have said that you have removed the record that has a foreign key to that record. So, keys and foreign keys already give us some causal paths for free. Can we use these and maybe some other forms of causal paths to find explanations to observations on the data? That's where we are heading our research.

How does query optimization change when you're running over thousands or hundreds of thousands of servers?

That's a very interesting theoretical problem because traditional query processing has defined query complexity (the cost of running a query) in terms of disk IOs. A query that uses fewer disk IOs is better than one that uses more, or a query plan that is cheaper in terms of disk IOs is better than another query plan. But now, when we use hundreds or thousands of servers in order to do a complex data analytics computation, we often do this because we want the entire data to fit in main memory. So, disk IO is no longer the main bottleneck, but the new bottleneck is the communication. How much data do we need to communicate in order to compute that query? It's a completely new metric, and this requires some interesting research to understand the inherent lower bounds on the communication complexity for computing the various queries. This is where we have done our research, and some of that research is complemented by algorithms that other people have found. For example, there is a class of algorithms that was described by Afrati and Ullman in a paper[7] about four years ago, and they are good matches for the lower bounds that we find using our theoretical analysis.

So, the query optimization crowd worried about communication back in the old days of distributed databases, when that was the highest cost for answering a query. So, for the second time around, is it fundamentally different?

I think it is fundamentally different if you run your query on 10 or 20 nodes or whether you scale it up to 500 or 10,000 nodes. You would ask the question: If I increase my number of servers from 500 to say 10,000, will I see a benefit? Well, we need to understand how the amount of communication depends on the number of servers involved in answering that query. We don't have these answers yet. Actually, I would claim we start to have these answers as a result of this theoretical work. These are the kinds of questions that are new on this framework: the dependence on a large number of servers that we did not have before in the 80s when the first parallel databases were developed.

Although the HPC community has always thought of the cost of computation as whatever you need to do in memory, and the communication, and if you have to read stuff from disk which they have to avoid like the plague, there's nothing new under the sun. But it's a new problem. They don't run parallel database queries -- different types of analysis.

So, the relationship is interesting because linear algebra is mostly about dense matrices or dense vectors. Databases are all about sparse matrices. Every binary relation can be thought of as a very sparse matrix. It's interesting to see that the techniques developed by the HPC community, they are absolutely the best when they deal with dense matrices. If you try

[5] Ce Zhang, Arun Kumar, Christopher Ré. Materialization Optimizations for Feature Selection Workloads. SIGMOD 2014, pp. 265-276.

[6] Judea Pearl. Causality: Models, Reasoning and Inference. Cambridge University Press, 2nd edition, 2009.

[7] Foto N. Afrati, Jeffrey D. Ullman: Optimizing joins in a map-reduce environment. EDBT 2010, pp. 99-110.

to blindly apply joins to do a matrix multiplication, the performance will be way beyond that developed by people in the HPC community. Conversely, when you're dealing with sparse data, the join processing techniques that we know in our community are just unbeatable. They are the best. It's interesting to combine these two, and probably we just need to add these two different solutions together into a unified system.

Rejections from SIGMOD and PODS don't mean that your work is lousy, it perhaps means that your work is ahead of its time. Eventually, good work will be published.

Sounds interesting. Three of your Ph.D. students have been runners-up or recipients of the ACM SIGMOD Dissertation Award. I want to know how you make that happen!

I want to know too! I was just lucky to work with very talented people. Some of my current students are equally talented, so I just feel lucky to have them and look forward to working with more people like them. I really don't have a recipe. I think it's just a matter of finding the right people.

So how do you find those people that you hire?

Um, to my shame, I should acknowledge that they found me. I did not find them. So, if I look back, I'm not good at recruiting people. I think I try to get people enthusiastic about the research that I'm doing and that is where my role stops, and it's very difficult. I really don't have a recipe. I also have very limited data points. I have only had maybe 15 Ph.D. students in my entire career. I don't know if this is an accurate number, but that would be the ballpark, and every case is different.

So, you're not involved in the admission space. They find you after they've been admitted.

I am involved in the admission space, but I'm a very lousy predictor of the performance of a student just based on their application.

It can be very hard for a shy young person to get out there and talk to others, especially at a big event full of strangers, like SIGMOD where we are today. But networking is important for almost any kind of career. You just mentioned a minute ago that you've been so busy meeting with people you haven't had time to hike in our beautiful surroundings here. So how have you dealt with this issue of overcoming shyness with strangers in your career?

I'm personally very shy, and I had difficulties at the beginning of my career. My advisor helped me a lot, and I think it's the duty of any advisor to help their students get over their shyness. Also, it gets better once you have research results. When you have a paper that you really care about, then you are really eager to tell the world about it, and then the shyness is less of an impediment. You just go out there and tell people about your great results. It's just things that people have to cope with. With help from the advisor and with…

How did your advisor help you?

I'm trying to remember… I think he just introduced me to people. It wasn't terribly effective, but maybe in 1 out of 10 cases the person he introduced me to would be actually interested in my research, and that is when I would take the opportunity to explain my research. I should also mention that explaining one's research is really important. I see this with my students all the time. They don't learn from me how to best explain their research because I know what their research is and I value it. They need to go out and try to explain it to other people, and then they discover that the language that they use and the way they explain their research is often the wrong way to do it. So then they need to adjust it and try again with the next person to explain their research. So, it's a very important experience to try to advertise your own work to other people and find the best way to describe it.

Do you have any other words of advice for fledgling or midcareer database researchers?

I think the question that anyone should ask is why am I in the game? Do I do science because I want to be a scientist? Or do I do science because I like to do science? In the first case, if you do science just because you enjoy the status of a scientist then maybe this is the wrong thing to do. Maybe you should search a different career and become a much more famous person in industry or in some other kind of career where you can make much more money than you can do in science. If you really care about the scientific questions, then this is the right place to be.

Also, ignore all those rejections. Rejections from SIGMOD and PODS don't mean that your work is lousy, it perhaps means that your work is ahead of its time. Eventually, good work will be published, and then it will be much better recognized if you stuck with your ideas despite the fact that the community had a difficult time accepting those ideas.

Did you ever find yourself have to wait a long time before something finally got accepted?

Oh yes, it was very difficult to publish papers on probabilistic databases. The community was just not ready to accept this model. So, I had a low point especially when I was advertising and when I was working on the early papers on probabilistic databases.

Is there any reason that the community turned a corner or it just happened? Did something happen to make it move in that case?

I think two things happened. One is that our community got more used to accepting the need to cope with probabilistic data. And the second is that we found a better way to explain the connection between the probabilistic data model and other probabilistic models that had been around for a longer time, like graphical models and more recently, statistical relational models.

From all your past research, do you have a favorite piece of work?

Yes, I do. That is still within the area of probabilistic data. It is a dichotomy theorem that tells you that every union of conjunctive queries can either be computed in polynomial time over a probabilistic database or it is #P hard to compute it over that probabilistic database. The reason why I'm really fond of this result is because it is inevitable. You need to know this if you want to do any kind of probabilistic inference. I mean this not only for database researchers but also for researchers in the knowledge representation as they become aware now of this result. It is simple, but not obvious at all. The criterion that makes a query to be computable in polynomial time not obvious, it's difficult to guess it, but once you see it, it's actually quite simple and straightforward. Moreover, the result itself was very hard to prove. It took us four years to prove the theorem, and I'm not proud that it took us so long, but I'm really happy because of the result in the end.

And where did that appear?

It appeared in the Journal of the ACM in 2012[8].

Was there a conference version before that?

There was a conference version in PODS 2010[9], but the JACM version is much more complete.

[…] the question that anyone should ask is why am I in the game? Do I do science because I want to be a scientist? Or do I do science because I like to do science?

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

Ah, I would hack. I enjoyed hacking in the past, but I didn't have any time to do hacking in the last several years. I would play more with these new fancy tools: IPython, Python, R, or Matlab. I would try to extend my knowledge of data management in the new direction.

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

I would learn more linear algebra. When I learned linear algebra in high school and in college, I was already passionate about programming, and I saw no connection between linear algebra and the cool things you could do with programs. But now linear algebra is a centerpiece of computer science and modern research, and I think the database research community needs to adapt and integrate linear algebra in its research.

Thank you very much for talking with me today.

Thank you, Marianne.

[8] Nilesh N. Dalvi, Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. J. ACM 59(6), pp. 30:1-30:87 (2012).

[9] Nilesh N. Dalvi, Karl Schnaitter, Dan Suciu. Computing query probability with incidence algebras. PODS 2010, pp. 203-214.