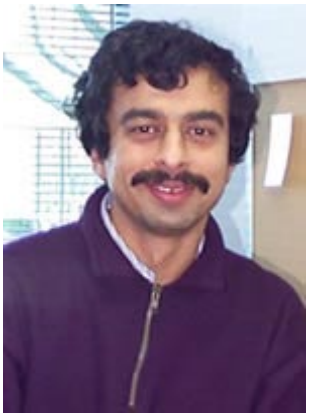
Surajit Chaudhuri Speaks Out

on How Data Mining Led Him to Self-tuning Databases, How He Does Tech Transfer, Life as a Research Manager, the Fragmentation of Database Research, and More

by Marianne Winslett



http://research.microsoft.com/~surajitc/

Welcome to this installment of ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today we are in Istanbul, site of the ICDE 2007 conference. I have here with me Surajit Chaudhuri, who is a research area manager at Microsoft Research. Surajit's current research interests lie in self-tuning databases, data cleaning, and text. Surajit is an ACM Fellow, and he received the SIGMOD Contributions Award in 2004. His PhD is from Stanford. So, Surajit, welcome!

Surajit, your PhD work was in database theory, then you switched to query optimization, then to self-tuning database systems. What has led you to become more practical over the years?

When I started at Stanford in database theory, I really liked everything I learned as a student of initially Gio Wiederhold, and then Jeff Ullman. It was a very educational experience, but I realized that I was not going to be as good as Jeff Ullman or Moshe Vardi as a database theoretician. So I started looking at more practical problems, and when I joined HP my job also demanded that. Slowly I migrated more towards systems work. I think that was good, because I don't think I am as smart as the database theoreticians!

That's very flattering to the database theoreticians. In fact, I can't think of a more important topic in core database management systems in the past decade than the need to make them self-managing. What are your thoughts on this area?

It is interesting how I got started on this topic. At that time, data mining was a popular topic in SIGMOD and VLDB. One of the questions that came up when I was thinking about data mining was how to evaluate your data mining solution and show that it is very good, that it mines high quality information that you can really use. It seemed to me that knowledge of the underlying domain is critical to be able to evaluate the quality of a data mining approach. The only domain I knew was the domain of *database systems*. So, to me, the interesting question was how to

leverage data mining in the context of database systems. Specifically, one of our design decisions was to use information such as a workload to determine what the right physical database design is.

Wait a second, this is completely the opposite of what I would have thought you would say. I thought you would have said that since you were at Microsoft, you knew that the single biggest piece of cost of ownership is paying for the database administrators to run the system. But that wasn't what got you started.

It is true that my motivation to start was a combination of factors, and the cost of database administration was certainly part of it. But I was also interested in the technical work that was being done in data mining, which certainly prompted my going in that direction. I got tremendous support at the beginning from talking to product groups, specifically the SQL Server product group, about the possibility of automating physical database design. The choice of physical database design as the place to try out our new data mining ideas was certainly motivated by the SQL Server group's push to make their systems self-managing. But it also came from my curiosity to see how data mining technology can help database systems. I think this is still an underworked area. Although I started from a data mining angle, our solution for the physical database design tool turned out to be quite different from what you see published in the data mining literature. We really turned our attention to how we can best solve the physical database design problem. The solution and architecture that emerged aren't anything like traditional data mining, although we used bits and pieces of data mining ideas, such as frequent itemsets.

Did the product group suggest the problem of automating the choice of which indexes to create?

We all know that how well queries perform depends not only on the query optimizer, but also on how good the physical database design is. Very smart people in the database community have worked on query optimization for many years. But physical database design was largely overlooked, with no one having really looked at it for a very long time, even though it has tremendous potential impact on the performance of queries and updates. Physical database design was already a well known problem, but researchers weren't focusing on it any more. I wanted to try to solve that problem, even though it was old and to someextent out of fashion.

You have been a manager at Microsoft for many years now. I would expect that the higher you go in the hierarchy, the less time you have for doing technical work. How would you describe the tradeoffs there?

Microsoft has a surprisingly flat management hierarchy, which means that I don't have to manage a budget number very carefully or any such thing. I do need approval from higher-ups to hire a person and so on, but it is a relatively simple process. The "management tax" is relatively small. Much of the credit for this goes to Rick Rashid for setting up Microsoft Research to be very flat, and for ensuring that it has a high degree of academic flavor in terms of its organization. Also, I do work directly with the researchers in my group. It is expected that I be technically involved, while still having some degree of management oversight of the two groups that I am in charge of. Working with the researchers, mentoring them, and helping them think through a problem has worked well, but it is true it does cut into my own research time.

My feeling is that being very disconnected from the technical work is dangerous because there are not too many jobs in research for such managers. So if not for anything else, but just for survival reasons, it is good to be technically involved. If you really want to do management, you

should go out in the product world and do management. If you want to stay in the research world, it is best to have a significant technical component in what you do. I try to maintain that.
You haven't listed any pluses to being a research manager. Are there any pluses? You said the minuses weren't as bad as one might think.

The pluses are that being a research manager does give you some ability to shape the research agenda. Much like faculty in a university, you get to set some research directions, and of course the process is not strictly top down. Often, researchers bring very wonderful ideas to you, and you get a chance to listen to those ideas, and support them. And sometimes you yourself may have an idea, and the research group can think it through. So you do have the ability to push an idea a little further than if you didn't have access to some resources. The research management experience can also be useful for other eventual career paths outside of research.

Your SIGMOD Contributions Award was for your work on CMT, the conference paper submissions management tool. What led you to create CMT, and do we still need it today?

For the 1999 SIGMOD, I was writing a paper on self-tuning histograms with Ashraf Aboulnaga. My wife was waiting downstairs in the car, and I told her that we had to send out this paper. She was waiting for me to print the paper and come downstairs to hop in the car so that we could drive 40 minutes to the FedEx office at the Seattle airport, which was the only FedEx dropoff point with a Saturday pickup. I thought how silly that was – I was producing something on the computer, then driving half an hour to drop it off! We had to be able to do better than that!

The opportunity came in 1999 when I was the program co-chair for the ACM KDD conference. I used that as an opportunity to ask my managing director of research, Dan Ling, for a contractor, so that we could build an online conference submission tool that would be useful for the rest of Microsoft Research. And as the KDD conference paper submission process moved along, we built the CMT code base, which served us for many years. Then recently, when I was the program chair for SIGMOD 2006, we built a completely new version of the code base, which has now replaced the old version.

Can anybody with good SQL programming and web development skills today build a tool like CMT? Absolutely. Building CMT was my opportunity to do something good for the research community, and at the same time learn what it takes to write a web service application, to deal with operational issues, and to work with the data center in a small way. Of course, building CMT was not research in the traditional sense, but it was my exposure to other interesting things, and I enjoyed it. But I don't claim that I created a piece of software that no one else could have built.

You manage both research and technology transfer people. How does that work?

When I came to Microsoft Research, I told Rick Rashid that I worked in the systems area and sometimes it takes more than one person to develop a system. He said that if I showed him the evidence that an idea is interesting, I can potentially hire one additional person. If I made more progress with those two people and found that I needed more people, then I could demonstrate the wider scope of the idea, then we could talk about additional resources again.

I believe in that approach. I like the model where we start things small, then see if we are successful. When we have a good idea in my group, we work on it until it is credible – not just in a research sense, but also to ensure that it has a certain robustness that makes it worthwhile for

the product group to potentially be interested. Then we engage with the product group carefully. If that goes very well, we work out with them a plan, and our researchers and some of the developers in research work with them to build the product (or a product feature). For example, in the case of self-managing systems (the AutoAdmin project), we worked directly with the product group to put our index tuning research and Database Tuning Advisor in the SQL Server code. We really enjoyed the experience.

Technology transfer is tricky because it has an impact on what we tell our researchers and how we evaluate them. I don't want to swing the balance too far to one end and tell my group members that they are totally unsuccessful if they don't do some degree of technology transfer. That would send them a strong signal that even if their research idea is not good, even if it is like a pure development project, that is okay because they will still get a lot of kudos and recognition for technology transfer. The other extreme would be to tell them that they should maximize their number of publications, in which case it would be really easy for them to work on smaller incremental improvements, which potentially have no impact on Microsoft's business. So I try to balance the two. It is tricky; I don't know if there is a good formula for it. I try to keep an eye on the robustness of each idea: is it a very fragile idea with too many loopholes? I try not to encourage that kind of work.

Why hasn't that approach to technology transfer been more widely used?

Technology transfer is not always easy, because you take a certain amount of risk. At performance review time, I ask the folks in my group a stock question: if you, for some reason or other, were unable to work at Microsoft, what would you do? Would you go to a faculty position, or would you work in a product group? I very much want them to stay in my group, but this binary-answer question helps them to reflect on what they are doing. Some of them want to make sure that they have the opportunity to go to a faculty position, and for them it is very important to keep an eye on publications. On the other hand, those who would go to the product group should gain experience in dealing with product quality code. So there is a complex tradeoff between where the individual wants to be in life, and what the organization gets back in return. I really can't comment on what goes on in other successful industrial research labs, such as IBM; but I think it is a tricky balance, and you have to be careful and patient to pull it off. I try to do my best.

Do other groups do that at Microsoft?

Yes, Microsoft Research takes both research publications and technology transfer into account in evaluating contributions. When we produce something that could transfer to products, I want it to be something that we can also write papers about in a first rate conference, such as SIGMOD, VLDB, and ICDE. To me that is very important.

I have heard that 80% of your researchers were interns at Microsoft before they were hired. Why is that?

The intern program is important not just for our group, but for all of Microsoft Research, and all of Microsoft in fact. The intern program is important enough that until his retirement from Microsoft, Bill Gates used to invite the interns to his house every summer. Internships let us engage the students and work with them, and internships are our opportunity to get to know the people who may be future candidates for full time positions. And the interns can find out what kind of place Microsoft Research is. The intern program is our opportunity to simultaneously evaluate students and to attract them. And it is also their opportunity to evaluate us, and see

whether Microsoft would be a good place for them for the long term. It turns out that except for maybe one person at this point, all the researchers that I have hired have interned with us one summer, and some of them have interned multiple times. I am very happy about that; it gives you great confidence when you hire them. In addition to having information such as reference letters and publications, it is a great way to know how they will work with us. Of course, we also look out for great talent even if they have never interned with us before.

What is your ratio of interns to hires?

In my research group, we try to get about 6 to 8 interns every year. We can't hire at the same rate as we get interns.

India has become very hot in information technology. If you were finishing your undergraduate degree in CS in India today, what would your next step be? Would you still head off to Stanford?

I greatly enjoyed my experience at Stanford, and I learned a lot about research and about life. So if I had to do it over again, I would probably still go to Stanford. But it is true that a person graduating from an Indian school today, let's say an Indian Institute of Technology, might not be immediately attracted to a PhD program, because there are many more alternatives today. David DeWitt and Jeff Naughton mentioned to me that the University of Wisconsin gets many fewer applications from students at the IITs than they used to. This is because the students have opportunities to work for Google, Microsoft, Yahoo!, IBM, world-known system integrators, and other multinational companies in India. There are also many interesting local ventures that they can work for. So students graduating in India today certainly have many more opportunities than there used to be. I think we will still get a subset of the very good students from India to come and do PhD programs, but fewer than before.

What do you think of the state of the art in query optimization?

This is one of my favorite topics. Every now and then I get the urge to start a new project to rebuild the query optimizer component of relational database systems. I think today's commercial query optimization is incredibly sophisticated. They have done great stuff, taking a language as complex as SQL and doing incredibly good work in optimizing large classes of queries, and really delivering on the promise of declarative specification of queries. Yet, I think that the query optimizer has also become a fragile component. The optimizers do wonderfully well for some queries, while for even some relatively simple queries, sometimes they won't do so well. I think that the main problem is the robustness of the query optimizers, and the generality of the SQL language does not help.

The tradeoffs between execution time and optimization time that existed a decade ago are also changing. The hardware is different. The cycles are cheaper. I think we have an opportunity to rethink query optimization and shift the balance between query optimization and execution.

Would I be bold enough to start a project in this area? It is always in the back of my mind. But in such an area, you need an insight before you can get started. There has been a lot of very interesting query optimization research work, like some of the work on doing things more dynamically, such as eddies. This work is very thought provoking. I am always looking for an insight which will give us robustness and yet the ability to deal with the traditional complexities of query optimization; this is the Holy Grail of query optimization research.

What was your experience as a graduate student at Stanford?

I started as a student of Gio Wiederhold, and then I moved to Jeff Ullman as my PhD advisor and I did my thesis with him. So I got exposure to both of them, and both were great mentors. From Gio, I learned to look at broad problems that are clearly of great importance. But if you look at a difficult area, it is sometimes not so obvious what the abstract formulation of the problem is. I had difficulty as a graduate student when I looked at the problem areas that Gio would point out, some of which history has shown us to be super important – for example, Gio talked about mediators and information integration way before the rest of the community. But yet, as a graduate student, it was hard to figure out what should be my thesis topic and what exactly I should I do. That struggle was terrifying as a graduate student. Now when I am years older, I recognize that you always have to deal with that ambiguity, as a junior or as a mature researcher. From Jeff and Moshe, on the other hand, I learned that in solving a problem you have to nail down your target quite clearly. They taught me to be very precise, to separate a problem from a non-problem. We don't have to solve the hardest version of the problem, but we should determine what is the problem we are solving and what is the problem we are leaving on the table, the part for which we don't have a solution, or have just a partial solution. So I greatly benefited from interacting with both Gio and Jeff, in different ways.

I would claim that there is starting to be a brain drain from academia to the industrial research labs, with people drawn there by the carrot of access to huge amounts of real world data.

I don't know whether there is a *huge* brain drain, but you would know better than I. I think it is a very interesting question: how can the data that industry has from product usage and services usage (as in Windows Live, Google, or Yahoo!) be shared with the academic world? I think that if we could find a way to do that, it would be good for all concerned, because when many groups work on a topic, it moves the field much faster than if done by just a few organizations. Offhand, I do not have an easy answer to all the issues involved in sharing of that data, such as preserving privacy and preventing unintended usage, but I wish something could be worked out. I have suffered a bit from this data sharing problem, but from a different angle: self-managing database systems are a very interesting area, and it would be very helpful for academicians to have access to real workloads and real databases and to be able to pursue research in this area. We haven't been able to make that work as yet, but I think it is a very important problem for academia and industry to work on together and see what we can together do about it.

So when you say you haven't been able to make it work, you mean just inside Microsoft, or to the outside of Microsoft?

Many people have asked us for real workloads, some way to give them the real database, so that they can understand, for example, how to evaluate physical designs. It is important to evaluate a physical design not just on synthetic data, but also on real workloads.

And you can get that inside Microsoft?

Yes, I can.

In some companies, you cannot even get that *inside the company.*

We can use datasets that we have been explicitly given access to. So it is not that we can just walk over to a product group and ask to talk to their customers and request their databases; it doesn't work like that even for us. But even when we have gotten some of that information (perhaps in such a way that we don't have the complete information), we haven't had any easy

way to share it outside Microsoft. I think sharing data with academia poses challenges, but it will be worth focusing on how to achieve it because the lack of sharing does slow us down, and will slow us down even more in the world of services.

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

The community has changed a lot from when I was a graduate student. We had a very strong systems focus. More recently, in the last decade or so, we have seen people with a great algorithms background and more wide-ranging interests come into our field and publish in SIGMOD and VLDB. If you look at the proceedings of database conferences now, the characteristics of the problems we work on have changed.

I think that there is a great opportunity right now to make another shift, toward the web; there are increasing storage issues and what you can term loosely as query processing issues that will come up as we build future applications with web-based data. Web data is not a traditional SQL database. Yet, I think a lot of insights we had from doing systems work in databases will be very useful in working with web data. So I am looking at that field and trying to understand what we can do.

More generally, obviously we need to adapt with the times and look at newer problems. But I am concerned that in our community we have too many distinct problem definitions, and thus we also have too few unifying themes. Therefore, we do not have a good evaluation of progress as we go along. For example, if you look at the papers in areas such as data cleaning and data exploration, often researchers (including myself) propose a new problem and give a solution. This results in independent silos of problems that we are solving one year, then trying to solve a little bit better the next year. I would rather see the community identify a few important problems and then have a sense of progress that over 5 to 10 years, these research groups together helped to solve these difficult problems. This happened for query optimization and of course for OLTP, but we no longer have that unity. And we cannot have such unity all the time.

But aren't those 5 year reports supposed to be rallying themes? [For example, see http://db.cs.berkeley.edu/claremont/claremontreport08.pdf, http://research.microsoft.com/~gray/lowell/LowellDatabaseResearchSelfAssessment.pdf, or http://www.hpl.hp.com/techreports/tandem/TR-90.10.pdf.]

I think they do provide rallying themes. For example, there have been many projects on the theme of information integration, which has been identified as a key research area in many of the reports. Yet, I think there are too many different problem definitions in information integration. You may say that that is the nature of the problem; that could be true, but it is unsatisfying. Five years later, we may look back to see what set of difficult problems we solved, and I may be able to raise my hand and say that in my specific domain, or on this specific definition, I made some progress. And a group from IBM, or a group from your university, may say very similar things. Yet as a community, I feel a little dissatisfied. But that may be my own personal opinion.

How could we fix that?

It is difficult. I don't have a solution. I wish we could work a little harder on this. Perhaps the best way is for a few groups to sit down together and try to identify unifying themes more concretely. The Lowell report and the Asilomar report do that at a very high level, but I think we

need to go back to some of the broad areas identified by these reports and take them one level down. And other aspects such as a shared set of benchmarks are important.

So, for example, we need an "information integration summit" where we hammer out a set of challenge problems for information integration researchers.

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

I liked query optimization a lot, and I like the recent work on data cleaning and data exploration, and looking at text. But if I have to single out one piece of research, it is going to be self-managing database systems. I have had tremendous fun with it. I started, as I said, coming from a very different angle – physical database design – but self-managing database systems is probably still my main passion.

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

I would probably educate myself a lot more on web technology, write web applications, and try to really look at the systems issues. I do intend to do that, but I wish I had more time for that than I do now.

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

When I was at Stanford, I went to advanced OS courses and I learned a lot about that area. From working with Jeff, I had great understanding of (at least introductory) logic-based techniques. But on the algorithms side, such as randomized algorithms, I only picked up bits and pieces. My depth in algorithms is a lot lower than I want it to be. And as I said, I would like more time to build systems.

Thank you very much for talking with us today.

I am very happy to do that. Thanks for the opportunity.