Aditya Parameswaran Speaks Out on Human-Powered Computation

Marianne Winslett and Vanessa Braganholo



Aditya Parameswaran
http://web.engr.illinois.edu/~adityagp/

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, Aditya Parameswaran, who is an assistant professor at the University of Illinois at Urbana-Champaign. Aditya received the 2014 SIGMOD Jim Gray Doctoral Dissertation Award for his thesis entitled, "Human-Powered Data Management". Aditya's PhD is from Stanford, where he worked with Hector Garcia-Molina.

So Aditya, welcome!

Thank you! I'm happy to be here!

Tell me about your dissertation

My dissertation is on Human-Powered Data Management as the title says.

The question is: how do you process large quantities of unstructured data (images, videos and text) with the help of humans? So here, we are talking about using crowdsourcing.

The goal of my dissertation was to figure out the fundamental primitives underlying the techniques you would use to process data with humans. So we figured out that there is a fundamental trade-off in this space, a trade-off between cost (you do need to pay humans if they help you process data), accuracy (humans make mistakes and you do need to take that into account) and finally, latency (humans take a lot of time). So there is a three-way trade-off that naturally appears in this setting. Given this three-way trade-off, our focus was on designing fundamental data processing algorithms, or rather, revisiting fundamental algorithms for data processing. Things like sorting, finding the max, filtering, they all have to be revisited under these new assumptions. The second goal was on how to use these algorithms in data processing systems. So we built a database system that uses humans as a data source (just like any other data source), and also a crowd-powered search engine, that uses humans to process data for you.

We have been recently using Amazon Mechanical Turk in my group, and I see jobs posted there – tasks that involve using a search engine to look up something and to rank the results. Are those coming from you guys?

It is possible! It is possible. The tool that we built for crowd-powered searching, which we call DataSift, starts with a query that a user might issue. This could contain images, for instance, "give me cables that connect to a socket that I took a photo of using my iPhone", and so here is a query that contains some rich information. Ordinary search engines cannot deal with these types of queries. As a result, we need to rely on humans as an integral part of the computation. So my system, DataSift, figures out the right way of decomposing this query into the set of small tasks that are done by humans, as well as automated tasks, which are done by the algorithm, and then combining the two to give accurate results.

How would you decompose the cable/plug question?

The workflow that we found to work well in this scenario is a workflow that we call "Gather-Retrieve-Filter-Retrieve-Filter". It's a mouthful, but the underlying idea is the following… You start by asking the crowd for fully textual reformulations of this query. In order to be able to use a traditional keyboard search API, you do need text. If you have images, there is no way you can use a traditional keyboard search API. So you ask the crowd for textual reformulations of this query. Maybe they may give you "this is a USB socket", or a more complex socket than I probably would not be able to identify. So they give me these textual reformulations. Starting from these textual reformulations, I (as Datasift) go and retrieve a few items that correspond to these textual reformulations using my keyword search API, so this is an automated step. Once I retrieve those items, I can then have humans evaluate those items to see whether they satisfy the query or not. Maybe if it indeed was a USB socket, "USB socket" would be a great keyword to retrieve things from and the items that you retrieve are all likely to be correct so people would say, "Yeah, those are good answers". On the other hand, if you start with a wrong answer, and you have people coming up with "three plug-pin sockets" (I just made that up), you're likely to get the wrong answers and as a result the crowd workers are going to identify that "Hey, these are returning wrong results. You should probably not use this". Starting from that, I can go back and reweight my reformulations. I can focus on the reformulations that gave me the most mileage from the sampling phase. Maybe I might focus on the USB sockets rather than the three plug-pin sockets. So once I narrow down on the reformulations that give me the most bang for the buck, I can retrieve a lot more items for that and once again have humans evaluate those items to finally compose the results for my query.

Most companies […] would not be willing to admit that it's actually humans in the background doing work for you.

Okay! I have some questions for you. How much duplication? How much would you have to pay? How long does it take? And how many duplicates would you

need to have a fairly high confidence for a query like the one we're talking about?

You have a workflow, which contains the six components. Some of them are automatic and some of them are crowdsourced. So internally for some of these components, specifically the filter component, we have developed algorithmic techniques that tell us when to ask additional questions, like how to trade-off between cost, latency, and accuracy. It will tell you, "Hey, look, there is a lot of disagreement about this item, based on the answers that we've gotten so far, so you should probably ask an additional human". On the other hand, there may be cases where you arrive at an agreement between the workers very quickly and therefore you do not need to ask additional humans. These individual operators in this workflow are optimized in the sense that given certain parameters, they optimize the others. Now, in terms of the overall budgeting, you start by having the user of such a system specify the amount of money they want to spend on this workflow. So, along with a query I provide my credit card and I say, "Hey, use $2". For queries that require domain knowledge, I may certainly be willing to spend those $2 to decompose a query into small units of work that are then answered by humans and then get results for the query. You could certainly pay a lot more, so you could get results faster, but when you pay around $2, you end up getting results within half-an-hour… that is the number that we typically see.

So you have to be willing to wait a while. You have to be willing to be a little patient. Basically, the crowds can help you in the cases when it's either a really hard query that you don't know the answer to or when the query requires so much labor that you're not willing to put in, and your time is more valuable. Presumably, this could be a useful building block for an apartment search engine. I have spent hours searching for apartments and if I could just specify what I want and the crowd could figure out…

> ***[…] don't rule out any options at the onset and be strategic.***

You'd like that one!

Yup! So these are the sorts of use cases -- anywhere it requires domain expertise or hard labor. Those are cases that the crowd can help you with.

How much of that half-hour spent is waiting for people to pick up the task versus formulating it? How does that half-an-hour break down?

I've had this happen many times. I would start off with a complicated query that I wanted to issue on Google search. I would start by posing that query and often would find that even with the first five pages of results, I didn't get anything useful. I would then reformulate it and formulate a different query and then go through the same process with that, and keep repeating the process until I get to the results that I want. Oftentimes it takes me half-an-hour or more; I would rather spend that $2 and have someone in the crowd help me out with it.

Okay so most of that time is spent with the person doing what you would have done.

Potentially, in the case where it is a labor-intensive task… In the case where it is a domain specific task like a "USB socket", I may not know the answer myself. Someone who may not be very electronically savvy might want to use a service like this just because they would be able to get answers that they don't already know.

That would be great!

This could be a solution to local IT support.

Yes! Instead of "what would Google say?" which is the best answer to give, it would be, "what would Amazon Mechanical Turk say?"

Yup!

Very good! What industrial impact do you see your dissertation work is likely to have?

I'm glad you've asked that question. We are currently conducting a survey of a number of companies that use crowdsourcing at a very large scale. As it turns out, a lot of companies use crowdsourcing at a large scale. So companies like Microsoft, Google, Facebook, all of them use crowdsourcing at a large scale and they are often ashamed to admit it because it is their secret sauce. Most companies prefer when the clever technology they are using is an algorithm, or better hardware, or something like that, right? They would not be willing to admit that it's actually humans in the background doing work for you. So a lot of companies use crowdsourcing as their secret sauce.

Secret sauce for what kinds of work?

Google, for instance… I'm probably missing out a lot of use cases, but [they use crowdsourcing as a secret sauce] for almost anything that requires training data. Any scenario where you require training data for your machine learning algorithm, that's a scenario where you could use crowds. So for content moderation, for spam detection, for search relevance, all of these are use cases for crowds. Oftentimes, they make subtle tweaks to the algorithm, and they have to then evaluate the results using crowds. So that is like a verification step rather than a training step. Each of these companies use crowds at a very large scale and that's what we've been discovering when we've been talking to these people. In fact, a lot of them are certainly trying to optimize for the tradeoff between costs, latency and accuracy, but some of them have not even gotten the basics right. So the techniques that myself and my collaborators have developed could certainly benefit these companies because they are doing this at scale and if they use optimized plugins or the algorithms that we've developed, they could certainly get a lot more mileage from the same dollar spent. So they could get the results quicker, they could get results of a higher quality, and so on.

When you say "at scale" do you mean like millions of worker tasks per day coming from these places? What does "at scale" mean nowadays?

So I don't think I'm allowed to talk about how many tasks these companies pose, but at the very least it is millions of tasks every week…and a lot of companies do even more. It suffices to say there's a lot of crowdsourcing being done and often for a lot of these companies this is at a scale larger than Mechanical Turk. Mechanical Turk is a toy example for them. These guys actually use outsourcing firms in India, Philippines and so on, and these outsourcing firms are middlemen. They will then hire employees who come and work for them 9-to-5, doing these micro tasks day in and day out. So given that you have these workers in-house, you have the ability to track their progress, and you have the ability to incentivize (bonuses that you could provide to the guys that are doing well). So it's a different set-up, but that's how some of these companies operate at scale.

Very interesting! Is there something that you know now that wish you had known earlier in your grad school and post-doc career?

When I started my PhD, I was hell-bent on either doing a startup or joining a research lab. Those were my only two career goals when I stared my PhD. Very soon I realized that my heart lay in research rather than doing a startup, at least at the start. I was passionate about getting to the bottom of things rather than dealing with management and dealing with all different issues that come up when doing a startup. By year two, I was all for joining a research lab. Year three was when Yahoo Research collapsed, and that's when the bubble burst for me. Yahoo Research was the place to go because there were a lot of really smart people, they had access to real problems, they had access to real data, and they were given the freedom to do whatever they wanted. That was how it was back then. The bubble burst because this is not a sustainable model. If you do not contribute back to the company, then it's not going to work out in the long run. That was when I started thinking and introspecting as to whether I really wanted to be in a research lab or would I rather have students of my own, to leave a legacy, to champion an area, to have a research vision, to have people working on fragments of that research vision, and moving the field forward with something that I can truly call my own rather than my company's or my team's. That's when I started thinking seriously about academia. It was not until year three when I started thinking seriously about academia.

If I had to do it all over again, I would have not eliminated that as a potential career goal at the start. So in the first two years, I was just having fun as a grad student. Not having fun in the sense of not doing work, but I was having fun working on all sorts of problems. I was going for breadth rather than depth. I was collaborating with people at Yahoo. I was collaborating with people at Microsoft. I was having collaborations with folks not in my research group but in other research groups at Stanford and I was having fun. But this was not getting me deep into a research topic such that I could have something substantial and meaty to say during my job talk. And that is something that happened more along the way. I wish I had figured this out a little earlier and mentally prepared myself a little earlier. I am not sad with where I am right now, but I would have mentally prepared myself to be an academic a little earlier.

I don't know about that. Your dissertation won that prize, so there's got to be meat there and we hired you! So I hope that was a happy end to the job hunt. I'm not sure that you really needed to start thinking pre-professionally any earlier than you did, but still, that's your advice and it stands.

So at the very least, maybe I got here by chance, because I got onto an area that was relatively unexplored and therefore I was lucky. But at least to others I would suggest that they don't rule out any options at the onset and be strategic. Have fun, have a lot of collaborators, have fun collaborating with a lot

of smart people, but be strategic and think long term at the very least. So it worked out for me at the end, but I wouldn't have expected it, right? In year three and year four I was panicking because I didn't know what was going on. I didn't have a dissertation topic, and that's when I chanced upon this exciting new field and very quickly we published a number of papers on it. If that had not happened, I don't think I would have landed an academic job.

Well thank you very much for talking with me today.

Thank you so much!