Roles at University of Washington

My position at the University of Washington is atypical and is worth describing for context. I serve in two primary roles.

As Associate Director of the eScience Institute, I lead a $4M/year organization advancing the use of data science techniques and technologies across science and engineering, with emphasis on cloud computing, scalable data management, machine learning, and visualization. My team consists of six direct reports (four PhD-level data scientists with degrees in various fields and two program managers who in turn supervise three administrative staff) and four research scientists who split their time with other departments. Our alumni include two UW CSE PhDs who now work at Google and Amazon respectively. I oversee a postdoctoral program with 18 fellows across campus, and I developed and oversee an "incubator" program for "stretch" research in data science. We've run the incubator program four times, one of which as a "Data Science for Social Good" program in Summer 2015. In this context I also led the design and development of a new Masters Degree program in Data Science, coordinating input from six departments on campus and serving as the initial Program Director and Faculty Chair. My payroll title for this role is *Principal Research Scientist*.

As Affiliate Associate Professor in Computer Science & Engineering (and as a member of University of Washington Graduate Faculty), I lead a computer science research group in scalable data management where I advise typically 4-5 students and 1-2 postdocs and research scientists. I am Principal Investigator on a portfolio of grants from federal and foundation sources, averaging about $800k / year; this portfolio is independent of the eScience Institute funding. We typically (but not exclusively) source our projects through customer-driven collaborations with scientists and engineers via the eScience Institute. Our approaches are based primarily in database systems, but we sometimes work in machine learning, systems, visualization, and HCI. My payroll title for this role is *Affiliate Associate Professor*.

Research Statement

My group is interested in making data professionals dramatically more productive, especially when working at scale. Performance is an important part of this: in the context of analytics, better performance can mean higher quality results, and therefore less work tuning and selecting models or acquiring new data. But more broadly, we are dissatisfied with the current experience of working on significant analytics projects: there are a huge number of tools, all with some combination of steep learning curves, extreme brittleness, and narrow applicability. We think in terms of democratizing access to state of the art algorithms and methods -- we want to shorten the gap between the forward edge of computer science, statistics, and applied math and the daily practice of analytics professionals.

Why is this important? Our belief is that soon there will be no jobs that do not require acting as a "data professional." The big data hype curve is having very positive effects: there is a new appreciation for quantitative rigor across business, government, and research, and data-driven decision making is steadily replacing hunches and traditions in these contexts. But we need "all hands on deck" -- we need to mature beyond the idea that a "priesthood" of data scientists is either necessary or sufficient. We need a diversity of tools and approaches to match the diversity in backgrounds of the people working with data.

We find that the models and languages of databases are important ingredients, especially in the long tail of users. Higher level abstractions, data-aware optimization, portability across platforms, and near unlimited scalability can all improve productivity and performance and drive new applications.

But these "good ideas" of the database community are too often trapped inside specific and aging implementations. We need to "jailbreak" these ideas and show how they can be adapted to the modern data science workflow.

I briefly describe two flagship projects in this space; see my website (http://homes.cs.washington.edu/~billhowe/) for information on these and other projects from my group.

SQLShare (http://uwescience.github.io/sqlshare/)

In the SQLShare project, we wanted to build one system to test two ideas: 1) That a simple web interface could improve uptake of SQL databases among non-experts, and 2) such an interface could serve as an "instrument" to help us better understand the kinds of tasks and datasets we need to support. The project was motivated by our observation that relational databases are underused in data-intensive projects, despite an obvious applicability. Researchers and practitioners needed easy data transformation at scale for tabular data, and were chronically struggling with script-and-file based solutions. But the overhead of installation, configuration, schema design, data ingest (a step that remains terribly painful in practice) were perceived as limiting factors. So we asked "What is the simplest possible system that can support SQL as a day-to-day analysis tool?"

To answer this question, we hosted a database in the cloud and designed a simple interface that reduced the "data workflow" to just three steps: upload data, write queries, and share results. An important contribution was to accept "weakly structured" data: vaguely rows-and-columns, but perhaps exhibiting inconsistent types within a column, missing or incorrect column labels, missing and incorrect data, and even gross structural errors (e.g., wrong number of columns). These weakly structured datasets are all happily tolerated on ingest, such that SQL itself can be used to make "repairs" by casting data types, filtering bad values, and renaming columns. We also emphasized view creation as a first-class citizen: whenever you save a query, you are implicitly creating a view that can be shared, reused, and

built upon. SQLShare emerged as a management layer for “virtual datasets” -- instead of sharing scripts, you shared the view, which allowed better provenance and transparency, better control over versions, better scalability, and usually a more concise expression of tasks.

We have operated SQLShare continuously for many years and have attracted hundreds of science users including many SQL novices. We described several use cases in a series of papers from 2010 - 2014. In a SIGMOD 2016 paper, we report on our analysis of the multi-year workload we collected. We found that SQLShare supported fundamentally different use cases from conventional databases. Datasets had very short “lifetimes” (i.e., the time between first and last query that accessed that dataset), more complex structure and content, and more complex query patterns than those of a conventional science database used as a control (the Sloan Digital Sky Survey workload). Since the SQLShare workload consists of thousands of hand-authored SQL queries from real users, we anticipate that the released workload dataset will be incredibly useful to the database research community.

Myria Middleware (http://myria.cs.washington.edu/)

While SQLShare represents a new “delivery vector” for database technology, it inherits at least two of the weaknesses of the underlying technology: questionable scalability to multi-terabyte datasets and poor expressiveness for complex analytics. Moreover, there has been a “Cambrian Explosion” of big data systems in recent years; researchers need better access to these tools as well, not just conventional relational databases.

The Myria project (led by PIs Suciu, Balazinksa, and Howe), consists of three main thrusts: theoretical models of parallel computations (led by PI Suciu), a new backend engine leveraging these ideas (led by PI Balazinska), and a “middleware” layer called RACO providing language and optimization services across multiple backends (led by PI Howe).

We have embraced the term “polystore” (see the BigDawg demo at VLDB last year) to describe the middleware problem. Modern big data ecosystems consist of multiple platforms co-existing and co-operating. The benefits include specialization for particular data types and algorithms (e.g., linear algebra, graphs, images), but at the cost of exploding complexity for users. Tasks need to be written and re-written many times to even evaluate the suitability of certain platforms. These systems have strongly overlapping capabilities; the decision procedure for which combination of systems to use for any particular task is unclear. The expertise required to effectively use even one of these systems is high; professionals who can effectively use many such systems are rare indeed.

To manage the complexity of these polystore ecosystems and study the fundamental differences between various systems and models, we are developing a new optimizing compiler for iterative relational algebra queries that can integrate multiple backends, including our own MyriaX backend, Spark, SQL-based systems, an MPI-based C++ distributed computing engine called Grappa, serial C programs for fast “small data” applications, and more. Key to our

approach is to generate and reason about a *polystore execution plan*: a distributed, multi-platform, and potentially iterative orchestration plan designed to insulate applications from the complexity of a heterogeneous big data ecosystem.

With additional funding from the Department of Defense and Intel (who both report significant investment in polystores in their own environments), we are exploring adding new services around security, cataloging, view management, and integration with cross-system resource allocation frameworks such as REEF/YARN.

This project include the language, optimizer, and web interface of the Myria project itself, and has been deployed in a production for multiple years. The code is available online (https://github.com/uwescience/raco).

We see Myria and RACO as an expansion of the thrust began by SQLShare: simplifying access to advanced data management and analytics technology, and delivering this technology in new contexts and markets.

Other Projects

- I have also done extensive work studying new capabilities for big data platforms including iteration (VLDB 2010 and 2012, Datalog 2.0 2014), workload analysis (VLDB 2013, SIGMOD 2016), skew management (SIGMOD 2012, VLDB 2012), and data pricing (VLDB 2012, PODS 2012).
- Higher up the application stack, I led one of the first projects in visualization recommendation with VizDeck (CHI 2014) which continues in partnership with Jeff Heer in Voyager (InfoVis 2015).
- With my postdoc Seung-Hee Bae, I have developed scalable algorithms for high-quality graph clustering on massive datasets (ICDMW 2013, Supercomputing 2015, TKDD 2016).
- With my student Poshen Lee and collaborators in scientometrics, I have used machine vision tools and techniques to develop new approaches for extracting information from figures in the scientific literature (ICPRAM 2015, BigScholar 2016).
- I've also worked deeply in domain-specific big data projects in Astronomy (Astronomical Society of the Pacific 2011, SSDBM 2010, In-Memory Data Management and Analytics 2015) and environmental microbiology (SSDBM 2013, IAAI 2015, Bioinformatics 2016, Computing in Science and Engineering 2013).