Research Statement

We are witnessing the increasing use of warehouse-scale computers to analyze massive datasets quickly. Examples include a scientist who post-processes simulation results on a high-performance computer or an enterprise that sifts through IoT data by renting 100,000 CPU cores for an hour for $500. These new analytical needs motivate us to investigate **how to design a data management platform for warehouse-scale computers**, with a focus on two core challenges: interoperability and scalability.

Thrust #1: Interoperability. Database systems have enjoyed widespread adoption thanks to their sophisticated data storage, data retrieval and data analysis capabilities. Today, however, users are embracing a more diverse set of tools for storage and analysis of massive datasets. For example, many scientific datasets contain images (arrays) in file formats like HDF5, FITS and NetCDF which are analyzed using deep learning frameworks like TensorFlow and PyTorch. Using a database system for such workloads is impractical, as this would entail continuously loading and saving large volumes of data. Instead of using a database system, users increasingly sift through massive datasets by writing code that manipulates data in files. This violates the revered data independence principle and precludes systematic optimization of these pipelines. Our research has contributed systems for managing massive scientific datasets that are interoperable with existing scientific data management tools:

- Our SIGMOD'14 paper [1] introduces SDS/Q, an *in situ* query engine for processing SQL queries directly on data in the HDF5 file format. The experimental evaluation shows that querying data in their native data format avoids data conversion overheads and shortens the time to answer the first query from hours to minutes.
- Our ICDE'18 paper [5] introduces ArrayBridge, a system that enables SciDB to query data in the HDF5 file format using its array-centric query languages (AQL and AFL). In addition to querying data, ArrayBridge allows SciDB to quickly materialize data in the HDF5 format by introducing a non-materialized view capability inside the HDF5 library to decouple I/O concurrency from compute concurrency. Experiments on a production facility show that ArrayBridge scales linearly to hundreds of nodes. At this scale, loading a 43TB HDF5 array dataset in SciDB would have taken weeks, while ArrayBridge can manipulate the same dataset in minutes.
- Our ICS'19 paper [8] identifies I/O optimization opportunities in heterogeneous clusters. A heterogeneous cluster, for example, may have a handful of powerful "large memory" servers running an in-memory key-value store such as Redis, and many cheap servers running a distributed file system like Hadoop HDFS. The key contribution of the paper is formulating the I/O optimization problem as a joint consolidation and placement problem after observing the parallel data access pattern of the workload. The evaluation finds that I/O optimization can increase performance by $200\times$ over directly reading small arrays and is $4\times$ faster than naive consolidation and placement methods on the ASAS-SN supernovae detection workload.

Thrust #2: Scalability. Warehouse-scale computing is quickly embracing new hardware. At the local node level, multi-core and many-core CPUs with hundreds of gigabytes of memory are becoming the norm. At the datacenter level, high-performance networks are displacing slow Ethernet in part due to their remote direct memory access (RDMA) capabilities that bypass many layers (and inefficiencies) of the software stack. The biggest cloud vendors even develop bespoke network interfaces, such as the Amazon SRD datacenter protocol that is partly implemented in hardware in Amazon's proprietary Nitro chip. Despite these advances, foundational data processing operations do not scale to the unprecedented concurrency of warehouse-scale computers nor do they fully exploit modern hardware. Examples include data shuffling

across plan fragments that triggers an inherently unscalable all-to-all communication pattern; hash-based joins that partition both inputs and do not fully overlap communication and computation; parallel aggregations that are oblivious to congested links in non-uniform network topologies. Our research has contributed algorithms that are cognizant of the topology, the data placement and the RDMA capability of a datacenter network to avoid scalability bottlenecks:

- Our SoCC'15 paper [2] considers how to optimize queries that join multiple tables in a memory-resident database. A hardware-conscious cost model surprisingly finds that join orders with pipeline-breaking operations (left-deep trees) can sometimes achieve higher performance than join orders that never materialize intermediate data (right-deep trees). An extensive experimental evaluation corroborates this model-driven insight.
- Our EuroSys'17 paper [4] designs data shuffling algorithms that use RDMA to transfer data at line rate. The experimental evaluation reveals that one can improve performance by as much as $4\times$ over other data transfer mechanisms, such as MPI. A unique feature of our data shuffling algorithms is the ability to forego transport-level guarantees such as message ordering for higher performance, when the query plan permits it.
- Our VLDB'19 paper [7] proposes a new parallel aggregation algorithm that considers the network topology and the data distribution to avoid unscaleable all-to-one communication patterns during parallel aggregation. Instead of always transmitting data directly to the destination, our algorithm performs partial aggregation along arbitrary network paths that end at the destination node. Although this approach transmits more data overall, a carefully chosen aggregation plan can significantly reduce the amount of data that is transmitted over the slowest network link, thereby improving performance. The research challenge is where and when to perform partial aggregation in a complex network. The evaluation shows that although the problem is SSE-hard, a simple greedy scheduling algorithm outperforms naive repartition-based parallel aggregation by as much as $3\times$ even in small clusters.

Synergistic activities

Disseminating data management research to the broader computer systems research community

Recent advances in computing require a wider perspective of computer systems research. In particular, it is essential to broadcast significant research results from the data management community to bigger audiences. Towards this goal, I have been a regular contributor to the ACM SIGARCH Computer Architecture Today blog series to communicate the breadth and depth of the data management discipline and its relevance to the broader computer systems research. The Computer Architecture Today series publishes weekly posts on different thematic clusters in computing research, spanning storage technology, programming languages, networking and novel applications. My articles highlight recent data management research and draw parallels to broader research themes in other areas of computer science and engineering. The series has proven an effective medium for disseminating and exchanging ideas, as it has generated more than 170,000 views and reaches more than 1,400 email subscribers.

Integrating education and research

I have engaged in the following activities to promote the integration of research and education:

1. **Introductory book on transaction processing with modern hardware.** One of the foundational problems in data management has been how to process transactions with high concurrency. The last decade has seen a resurgence of interest in this intricate problem, and more than a hundred papers have been

published on this topic in leading data management venues. The fast pace of research output makes this problem difficult to approach for students and junior researchers. With this audience in mind, I have co-authored a book on transaction processing on modern hardware that classifies and distills this new body of work [9].

2. **Curriculum design for the Data Analytics undergraduate major.** Proficiency in analyzing large datasets is an emerging workforce need that Ohio State decided to address through a new Data Analytics undergraduate major. As part of the mandatory curriculum of the major, I have designed "Data Management in the Cloud", a new course that teaches data management techniques for massively parallel computing infrastructure to a data science student audience. Topics include fundamentals of query optimization, partitioning and replication, and data placement in heterogeneous storage hierarchies. I have developed new lectures, labs, homeworks and exams about the systematic organization and analysis of data on high-end computers that are based on our research.

3. **Disseminating pedagogical experiences and practices.** Data Science is a multi-disciplinary subject that combines analytical expertise with technological knowledge. Instructors need to develop new methodologies to teach deeply technical material to data scientists that do not have a formal computer science background. I have been disseminating pedagogical experiences and practices from teaching an advanced course to Data Analytics majors in multiple fora, including the workshop on "Undergraduate Data Science Pedagogy & Practice" at UC-Berkeley in July 2018.

Looking ahead

Our future research plan will investigate the following open challenges in managing data at scale.

1. **Henosis: an integrated runtime for data science on massive datasets.** A natural representation for data in many analyses is that of an array (tensor). This abstraction has been embraced by high-productivity statistical and numerical computing environments such as R and MATLAB, but unfortunately these systems are limited to processing datasets that fit in a single node. In comparison, scaleable data processing systems rely on an outdated design of a bytestream-oriented interface (POSIX) to data storage which is agnostic to the optimization opportunities that array-centric computing presents. We argue that it is necessary to adopt an array-centric data access interface to storage for data independence: application developers will reason in terms of accesses to arrays and a data management runtime will intercept and evaluate these accesses as queries on an array view. Henosis will hide sophisticated data management capabilities behind the HDF5 array access interface and will access datasets that are fragmented into different files and formats by performing conversions in transit. A consolidation and placement algorithm will track the workload access pattern to determine the most efficient storage backend and representation for each array chunk. Dynamic re-chunking and co-location will decouple disk I/O from data propagation between data producers and data consumers, and allow for selective data materialization. Finally, semantics-driven optimizations will tailor I/O to the workload through chunk-oblivious work assignment and coordination.

2. **Hyperscaling data analytics with network-aware algorithms.** Data analytics relies on a few key operations to process and summarize massive datasets such as aggregation, union, join, and sorting. However, these operations are currently performed in a manner that is oblivious to the underlying network topology, which is often complex and heterogeneous. Existing parallel computation models predominantly focus on the cost of communication. However, the emergence of fast networks means that balancing and minimizing the computation is equally important for high performance. We will pursue an end-to-end investigation of how we can model, design and deploy network-aware algorithms for fundamental data

processing tasks. To achieve this goal, we will first develop a parallel model that can jointly capture the cost of computation and communication. Using this model, we will build algorithms with provable theoretical guarantees and we will consider the practical implications of implementing these algorithms in a prototype network-aware parallel data management system.

3. **Near-data computing in the datacenter with RDMOs.** A fundamental problem with scaling data management to a datacenter is the latency of a remote access: even in the fastest networks, accessing remote memory is at least $10\times$ slower than accessing data in local memory. Making matters worse, many simple manipulations require a sequence of operations, which results in multiple round-trips over the network. Our key idea is that the networking hardware shall support the dispatch of simple data processing logic in one "unit" to a remote computer which will be executed near the physical point of data storage in one round-trip. We term this unit of computation a Remote Direct Memory Operation, or an RDMO. Specifically, an RDMO is a short sequence of reads, writes, and atomic memory operations that will be transmitted and executed at the remote node without interrupting its CPU, similar to an RDMA operation today. Unlike an RPC call, an RDMO cannot invoke program functions, initiate system calls or issue additional RDMO requests. Unlike a network ISA, an RDMO cannot execute arbitrarily user-defined programs. The hardware architecture community has extensively explored near-data computing as a mechanism to bring computation closer to local memory. We posit that RDMOs will be the enabling technology to bring near-data computing to remote memory in the datacenter.

Selected publications

[1] S. Blanas, K. Wu, S. Byna, B. Dong and A. Shoshani. Parallel data analysis directly on scientific file formats. In the *ACM International Conference on Management of Data (SIGMOD)*, 2014.

[2] F. Liu and S. Blanas. Forecasting the cost of processing multi-join queries via hashing for main-memory databases. In the *6th ACM Symposium on Cloud Computing (SoCC)*, 2015.

[3] Y. Yuan, K. Wang, R. Lee, X. Ding, J. Xing, S. Blanas, X. Zhang. BCC: Reducing False Aborts in Optimistic Concurrency Control with Low Cost for In-Memory Databases. In *Proceedings of the VLDB Endowment (PVLDB), 9(6)*, 2016.

[4] F. Liu, L. Yin and S. Blanas. Design and evaluation of an RDMA-aware data shuffling operator for parallel database systems. In *12th ACM European Conference on Computer Systems (EuroSys)*, 2017.

[5] H. Xing, S. Floratos, S. Blanas, S. Byna, Prabhat, K. Wu and P. Brown. ArrayBridge: Interweaving Declarative Array Processing in SciDB with Imperative HDF5-Based Programs. In the *34th IEEE International Conference on Data Engineering (ICDE)*, 2018.

[6] D. L. Quoc, I. E. Akkus, P. Bhatotia, S. Blanas, R. Chen, C. Fetzer, T. Strufe. ApproxJoin: Approximate Distributed Joins. In the *9th ACM Symposium on Cloud Computing (SoCC)*, 2018.

[7] F. Liu, A. Salmasi, S. Blanas and A. Sidiropoulos. Chasing similarity: Distribution-aware aggregation scheduling. In *Proceedings of the VLDB Endowment (PVLDB), 12(3)*, 2019.

[8] D. Kang, V. Patel, A. Nair, S. Blanas, Y. Wang and S. Parthasarathy. Henosis: workload-driven small array consolidation and placement for HDF5 applications on heterogeneous data stores. In the *33rd ACM International Conference on Supercomputing (ICS)*, 2019.

[9] M. Sadoghi and S. Blanas. Transaction Processing on Modern Hardware. In *Synthesis Lectures on Data Management*, Morgan & Claypool Publishers, 2019.

[10] F. Liu, L. Yin and S. Blanas. Design and evaluation of an RDMA-aware data shuffling operator for parallel database systems. In *ACM Transactions on Database Systems*, 2019.