Yue Cheng Research Statement

The big data era has dramatically influenced almost every aspect of our modern lives. From social networks to enterprise business, from cloud storage to flash thumb drives, from massive-scale datacenter cluster to wearables and field sensors, the rapid growth of computing technologies has changed the way we work, do business, and entertain ourselves. One fundamental building block that enables the functioning of a myriad of services is the *data-intensive computing and storage system*. However, today's service stacks are evolving at a fast pace, and consist of a mix of complicated software and hardware. Different sub-systems, components, and tiers interact with each other. Non-holistic piece-by-piece optimizations have resulted in sub-optimal solutions, hence, inevitably dragging down the end-user experience, whereas the initial target was to provide high performance and easy of use. This becomes even more challenging as the scale of these systems increases to hundreds/thousands of machines/devices deployed at different geographic locations.

I conduct research to enable efficient and flexible (i.e., ease-of-use, ease-of-programming, and ease-of-deployment) systems for the growing data demands of modern applications running on existing as well as emerging computing platforms such as cyber-physical systems and Internet of Things (IoT). Specifically, my Ph.D. research is driven by the complexities of modern computing and data-intensive systems, and the need for more efficient and flexible approaches to manage such complexities. My thesis work targets three real-world application scenarios: (1) massive-scale web applications (e.g., Facebook web queries) that require efficient, scalable, and flexible distributed storage systems; (2) enterprise primary storage workloads (e.g., Email and file servers, etc.) that demand a high-end storage hardware plus intelligent software codesign, and (3) cloud-based big data analytics (e.g., distributed NoSQL queries) that need careful deployment planning. By performing extensive and deep analysis to understand the issues, designing rigorous models and practical tools to characterize complex workload behaviors, and building efficient systems to manage different tradeoffs and the massive volume of data, my research aims at improving the efficiency and usability at the system level with a broad focus on practical and user-centric metrics.

A key goal of my research is to have practical impact and innovative solutions for real-world problems. During my Ph.D., I have successfully applied my research methodology to uncover critical issues in distributed and data-intensive systems. My investigations have led to novel and effective approaches to solve these problems. For example, the offline flash caching heuristic [3] that I developed is, to the best of our knowledge, the first offline caching algorithm that takes flash endurance into consideration. It can be used to evaluate any online flash caching solutions, and navigate the tradeoffs between performance and endurance. My research has appeared in a number of premier conferences and workshops in computer systems and high-performance computing, including USENIX ATC, HotStorage, HotCloud, ACM EuroSys, and HPDC. Overall, I have comprehensive experience with the research processes and maintained close collaborations with enterprises such as IBM Research and EMC. In the future, my vision is to build synergistic systems solutions, which, by cohesively harmonizing emerging hardware with smart software, strive to balance efficiency (e.g., performance, resource, energy, and cost efficiency, etc.) against flexibility.

1 Fast, Efficient, and Flexible Data Services for Large-scale Web Workloads

The extreme latency and throughput requirements of modern web-scale services (an impressive list of users includes Facebook, Airbnb, Twitter, and Wikipedia, etc.) are driving the use of distributed object caches (e.g., Memcached) and stores (e.g., OpenStack Swift). Commonly, the distributed cache/storage tier can scale to hundreds of nodes. With the growth of cloud platforms and services, object storage solutions have also found their way into both public and private clouds. Cloud service providers such as Amazon and Google Cloud Platform already support object caching and storage as a service. However, web query workloads exhibit high access skew and time varying heterogeneous request patterns, which cause severe access load imbalance and unstable performance. The difficult-to-develop (and -debug) nature of distributed storage systems even makes the situation worse. *My research along this line tackles the above issues using a holistic redesign approach that cohesively combines piece-by-piece optimizations with the goal of maximizing both efficiency and flexibility.*

Fast and load balanced distributed in-memory object caching To efficiently utilize the underlying cloud resources, and to effectively handle workload imbalance in memory cache deployment, I developed MBal [8], an in-memory object caching framework that leverages fine-grained data partitioning and adaptive multi-phase load balancing. MBal performs fast, lockless inserts (SET) and lookups (GET) by partitioning user objects and compute/memory

resources into non-overlapping subsets called cachelets. It quickly detects presence of hotspots in the workloads and uses an adaptive, multi-phase load balancing approach to mitigate any load imbalance. The cachelet-based design of MBal provides a natural abstraction for object migration both within a server and across servers in a cohesive manner. Evaluation results shows that MBal brings down the tail latency of imbalanced web queries close to that of an ideally balanced workload.

Efficient, elastic, and heterogeneity-aware cloud object store Not only the access imbalance exists in web workloads, modern internet-scale web workloads exhibit heterogeneity in multiple dimensions. For instance, different types of applications may impose drastically different workload characteristics. Moreover, the situation is further complicated by the fact that datacenters hosting the storage layer are becoming increasingly heterogeneous (indeed the heterogeneity in hardware is necessary to reduce the likelihood of correlated failures in datacenters). Addressing these problems, I developed MOS [9, 5], a cloud key-value store that re-architects the traditional monolithic design by independently partitioning all the resources into fine-grained *microstores*, each attuned with a particular type of workloads. By leveraging the lightweight containers, MOS dynamically provisions microstores and exposes the interfaces to the tenants to use according to application's SLA requirement. We have implemented MOS in OpenStack Swift. Evaluation using real testbed and simulations demonstrate that MOS can effectively meet the SLAs under multi-tenancy while being resource efficient.

Providing flexibility and better programmability in distributed storage systems Distributed systems are notoriously bug-prone and difficult to implement, which I experienced when prototyping multiple distributed systems projects. I wondered what would be an easy and productive way to develop distributed systems from scratch. To solve the puzzle, I conducted a study [4], where I observed that, to a large extent, such systems would implement their own way of handling features of replication, fault tolerance, consistency, and cluster topology, etc. To this end, my colleagues and I designed and implemented IOLITE [2], a universal and flexible ecosystem that handles the "messy plumbings" of distributed systems by synthesizing a series of reusable and efficient components of distributed system techniques. Using IOLITE, developers only need to focus on the core function (SET or GET logics) implementation of the application, and IOLITE will convert it into a scalable, and highly configurable distributed deployment, following a serverless fashion.

2 Cost-effective Flash Caching for Enterprise Datacenter Workloads

Unlike traditional hard disk drivers (HDDs), flash drives, e.g., NAND-based solid state drives (SSDs), have limits on endurance (i.e., the number of times data can be erased and overwritten). Furthermore, the unit of erasure can be many times larger than the basic unit of I/Os, and this leads to complexity with respect to consolidating live data and erasing obsolete data. For enterprise primary storage workloads, storage must balance the requirement for large capacity, high performance, and low cost. A well studied technique is to place a flash cache in front of larger, HDD-based storage system, which strives to achieve the performance benefit of SSD devices and the low cost per GB efficiency of HDD devices. In this scenario, the choice of a cache replacement algorithm can make a significant difference in both performance and endurance. While there are many cache replacement algorithms, their effectiveness is hard to judge due to the lack of a baseline against which to compare them: Belady's MIN, the usual offline best-case algorithm, considers read hit ratio but not endurance.

To this end, I explored offline algorithms for flash caching in terms of both hit ratio and flash lifespan. I developed a multi-stage heuristic [3] by synthesizing several techniques that manage data at the granularity of a flash erasure unit (which we call a container) to approximate the offline optimal algorithm, which we believe is much harder to compute. Evaluation showed that the container-optimized offline heuristic provides the optimal read hit ratio as MIN with 67% less flash erasures. *More fundamentally, my investigation provides a useful approximate baseline for evaluating any online algorithm, highlighting the importance of comparing new policies for caching compound blocks in flash.*

3 Cost-efficient Big Data Analytics Management in the Cloud

The use of cloud resources frees tenants from the traditionally cumbersome IT infrastructure planning and maintenance, and allows them to focus on application development and optimal resource deployment. These desirable features coupled with the advances in virtualization infrastructure are driving the adoption of public, private, and hybrid clouds for not only web applications, such as Netflix, Instagram and Airbnb, but also modern big data analytics using parallel programming paradigms such as Hadoop and Dryad. With the improvement in network connectivity and emergence of new data sources such as IoT edge points, mobile platforms, and wearable devices,

enterprise-scale data-intensive analytics now involves terabyte- to petabyte-scale data with more data being generated from these sources constantly. Thus, storage allocation and management would play a key role in overall performance improvement and cost reduction for this domain. While cloud makes data analytics easy to deploy and scale, the vast variety of available storage services with different persistence, performance and capacity characteristics, presents unique challenges for deploying big data analytics in the cloud. *My cloud storage tiering solution, driven by real-world workload behaviors and cloud service characterization, takes the first step towards providing cost-effective data placement support for cloud-based big data analytics using the economic principles of demand and supply for both cloud tenants and service providers.*

Cloud tenants' perspective: cloud storage tiering Addressing these problems, I developed CAST [7], a framework leveraging different cloud storage services and heterogeneity within jobs in an analytics workload to perform cost-effective storage capacity allocation and data placement. CAST does offline profiling of different applications within an analytics workload and generates job performance prediction models based on different storage services. By incorporating high-level objectives provided by tenants, CAST uses a simulated annealing solver to generate a data placement and storage tiering plan. The framework automatically deploys the workload in the cloud based on the plan. Targeting production Hadoop-based analytics workloads, our evaluation on a 400-core cloud cluster demonstrated that CAST can effectively improve the workload performance while significantly reducing the tenant's monetary cost. *This research has gained attention from industrial practitioners, since our approach is especially helpful considering such a usage scenario where cloud tenants need to periodically run analytics workload on the same dataset, which is being incrementally updated.*

Cloud tenants vs. providers: hybrid cloud object store via dynamic pricing While the use of faster storage devices such as SSDs is desirable by tenants, it incurs significant maintenance costs to the cloud service provider. To alleviate this problem, I extended CAST to further incorporate dynamic pricing by involving the provider. The resulting hybrid cloud store [6] exposes the storage tiering to tenants with a dynamic pricing model that is based on the tenants' usage and the provider's desire to maximize profit. The tenants leverage knowledge of their workloads and current pricing information to select a data placement strategy that would meet the application requirements at the lowest cost. Our approach allows both a service provider and its tenants to engage in a pricing game, which yields a win-win situation, as shown in the results of the production trace driven simulations.

4 Future Directions

I have been focusing my research on practical problems in computer systems. This has allowed me to gain deep understanding of one research area and to dig deeply to have the most impact. Looking forward, I would like to continue my focus on designing systems with high efficiency and flexibility, extend my understanding and expertise in cyber-physical systems and ubiquitous computing, and leverage my experience to mentor/teach new researchers in this area. In the following, I will discuss several future directions that I am particularly interested in.

Scaling high performance data-intensive systems by redesigning the system-architecture interfaces Massive deployment of fast storage devices (such as high-density non-volatile memory) has boosted the performance of modern datacenter applications. My prior and ongoing work has shown that there exists great potential for extending the endurance of datacenter flash storage in both single device [3] and at scale [1]. At extreme scale, performing distributed wear leveling and global garbage collecting can not only improve the overall lifespan of the flash array, but also effectively improve the overall performance. However, applications and the underlying distributed flash cluster are still segregated and there is no way application can directly manage the functionality controlled by the storage hardware, thus impacting the cost effectiveness inevitably. Observations and preliminary results demonstrate the huge improvement space of such an application-managed hardware design [1, 3]. In the short term, I plan to conduct research for a cross-layer system-architecture codesign to enable transparent scale-up/scale-out high performance storage.

Rethinking the server system design in the Era of Ubiquitous Computing Computing has expanded beyond the Internet and become ubiquitous everywhere in the physical world. A trending area is rethinking the server system design in the context of cyber-physical systems, IoT, mobile and wearable devices. Future server systems would involve complex interactions with users and require the right balance of resources such as CPU, memory, storage, and energy. To provide effective and efficient infrastructure support, I am interested in exploring the tradeoffs among all possible objectives including performance, deployment cost, reliability, easy-of-use/deployment, and energy efficiency, in the context of low-end, low-capacity, energy-sensitive and IoT-scale environments. Such studies will impact the next-generation server system software design and development, and provide best practice guidance for practical deployment in the field. Furthermore, with my experience in storage systems research, it would also

be interesting to see how the emerging next-generation non-volatile/storage-class memory techniques could benefit the next-generation server system design.

Serverless & Ubiquitous Computing infrastructure codesign Serverless computing[1], as a new, lighter-weighted computing paradigm has transformed how developers build, deploy, and manage applications. Compared to traditional VM-deployment based cloud computing, and elastic scaling services, serverless computing simply allows tenants to run fine-grained functions in a reliable, elastic, and "serverless" manner, finally delivering the "pay-as-you-go" promises of cloud computing. While still in her infancy, this new model imposes new challenges in nearly all aspects of systems. The problems become more exciting when considering a serverless & ubiquitous computing infrastructure codesign, when a large scale of IoT devices are staggered to interact with a set of developer-defined functions triggered in the serverless cloud. In the long term, I plan to (1) study and understand the behaviors of serverless computing in cloud service providers from the perspective of IoT app developers, (2) contribute to the development of open source serverless computing platform to gain a deeper understanding from within the core of the new model, and (3) build next-generation infrastructure support that aims to improve the resource/energy efficiency, usability/programmability, and performance of both IoT apps and serverless backend platforms.

Summary To summarize, my research will be geared towards building next-generation systems-level support for data-intensive applications with a focus on improving user-centric metrics such as usability, performance, and reliability. With the experience of helping my advisor with preparing research proposals (I was extensively involved in preparing 6 proposals during my Ph.D.), I understand the need for securing funding in any environment to meet my research goals. Hence, I believe that my proposal writing experience and my past work in computer systems research have equipped me with the requisite tools to explore new problems in diversified directions.

Selected Publications

[1] N. Zhao, **Yue Cheng**, A. R. Butt, and X. He. Improving flash cluster's lifetime and performance through endurance-aware offloading. Under preparation, June 2017.

[2] A. Anwar, **Yue Cheng**, H. Huang, D. Lee, and A. R. Butt. IOLITE: A universal framework for distributed key-value stores. In *USENIX FAST '17*. Under review, Feb. 2017.

[3] **Yue Cheng**, F. Douglis, P. Shilane, M. Trachtman, G. Wallace, P. Desnoyers, and K. Li. Erasing belady's limitations: In search of flash cache offline optimality. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, Denver, CO, June 2016. USENIX Association.

[4] A. Anwar, **Yue Cheng**, H. Huang, and A. R. Butt. Clusteron: Building highly configurable and reusable clustered data services using simple data nodes. In *8th USENIX Workshop on Hot Topics in Storage and File Systems (HotStorage 16)*, Denver, CO, June 2016. USENIX Association.

[5] A. Anwar, **Yue Cheng**, A. Gupta, and A. R. Butt. Mos: Workload-aware elasticity for cloud object stores. In *Proceedings of the 25th ACM International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '16, pages 177–188, New York, NY, USA, June 2016. ACM.

[6] **Yue Cheng**, M. S. Iqbal, A. Gupta, and A. R. Butt. Pricing games for hybrid object stores in the cloud: Provider vs. tenant. In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15)*, Santa Clara, CA, July 2015. USENIX Association.

[7] **Yue Cheng**, M. S. Iqbal, A. Gupta, and A. R. Butt. Cast: Tiering storage for data analytics in the cloud. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, HPDC '15, pages 45–56, New York, NY, USA, June 2015. ACM.

[8] **Yue Cheng**, A. Gupta, and A. R. Butt. An in-memory object caching framework with adaptive load balancing. In *Proceedings of the Tenth European Conference on Computer Systems*, EuroSys '15, pages 4:1–4:16, New York, NY, USA, Apr. 2015. ACM.

[9] A. Anwar, **Yue Cheng**, A. Gupta, and A. R. Butt. Taming the cloud object storage with mos. In *Proceedings of the 10th Parallel Data Storage Workshop*, PDSW '15, pages 7–12, New York, NY, USA, 2015. ACM.

[1] A cloud computing code/function execution model in which the cloud provider fully manages starting and stopping virtual machines as necessary to serve requests, and requests are billed by an abstract measure of the resources required to satisfy the request.