Cloud computing is undergoing a fundamental shift, stimulated by an exponential growth in data and users and an increasing desire for elastic, fully-managed services. In the emerging wave of cloud computing, users no longer need to manage virtual machines with fixed ratios of CPU cores, memory, and storage to run their applications. Rather, cloud providers are offering fully managed compute and storage services, backed by disaggregated resource pools. The main benefit for users is that cloud providers automatically scale and allocate resources to achieve both high performance and high resource efficiency for applications.

My research enables this new wave of cloud computing through practical software systems that address two fundamental requirements. The first requirement is fast access to remote data between compute and storage resource pools, which are connected over a network. I designed a system, ReFlex, that enables fast, predictable access to Flash storage on any remote server in a cloud facility with spare capacity and bandwidth. The second requirement is intelligent control of storage resource allocations to meet the performance demands of applications while minimizing resource usage (i.e., cost). My system, Selecta, leverages machine learning to derive near-optimal resource allocations for cloud applications. I have unified fast access to remote data and intelligent resource management in a new cloud storage service, Pocket. Pocket automatically allocates and scales resources across multiple nodes and storage technologies to enable an emerging class of cloud applications, namely interactive analytics, to share data efficiently. The system leverages ReFlex for fast access to Flash storage.

My approach to research, which I have demonstrated throughout these projects, is rooted in analyzing real and relevant classes of applications. I value developing a thorough understanding of application characteristics and the performance bottlenecks that existing systems impose. This kind of analysis guides the design of new systems I build to ensure that my work addresses real needs. For example, the ReFlex project stemmed from my study of production workloads that use Flash storage at Facebook. I also build novel systems to enable new classes of applications. I strongly believe in making my research widely accessible and implementable in real systems. To this end, I have made my work open source and I have designed systems that run on public cloud platforms. My work is seeing adoption in industry, which I find highly rewarding.

ReFlex: Fast Access to Disaggregated Storage

Designing cloud servers to have the right balance of compute, memory, and storage resources is difficult because applications have unique, and often dynamic, requirements for each resource. I studied real applications at Facebook and found that applications vastly underutilized Flash storage capacity and bandwidth on servers due to imbalanced resource requirements [1]. Low utilization leads to high cost as server hardware cost makes up a significant portion of total expenses for cloud operators.

Disaggregating storage from compute resources allows cloud operators to improve utilization by independently scaling resources. I explored the benefits of disaggregated cloud storage designs during my internship at Microsoft Research [2]. In a disaggregated cloud environment, applications remotely access storage devices that have spare capacity and bandwidth, regardless of their physical location in a cloud facility. However, disaggregating modern Flash storage devices presents several challenges. Traditional operating systems introduces high overhead as they were originally designed with the assumption of slow network and storage devices. Furthermore, since write operations take signif-
icantly longer than read operations on Flash technology, providing performance isolation between read and write requests from multiple clients sharing a Flash device is another important challenge.

My system, ReFlex, enables applications to access remote Flash storage with nearly identical performance to accessing Flash that is directly attached to the application’s CPU [3]. ReFlex is a software system that runs on a remote Flash storage server and bypasses traditional operating system software for network and storage processing. The execution model for request processing in ReFlex is inspired by my work on the IX operating system, which showed that we can dramatically improve network processing efficiency by making network controller queues directly accessible to applications [4, 5]. While IX focused on networking, ReFlex closely integrates network and storage processing and also introduces a novel I/O scheduler to provide predictable performance on shared Flash storage.

By efficiently forwarding data between network and Flash device queues, ReFlex serves up to 850,000 storage requests per second per CPU core. This is an 11× improvement compared to a Linux server implemented with standard I/O libraries. ReFlex provides throughput and tail latency performance guarantees for up to thousands of clients sharing a Flash server by using a scheduling algorithm that takes into account Flash device characteristics and client performance objectives. In contrast to approaches that rely on specialized networking hardware to achieve high performance, ReFlex is a purely software-based solution that works on commodity hardware in public clouds. ReFlex is highly adoptable as it requires no changes to the host operating system.

ReFlex is already having impact. The system serves as a fast Flash storage tier on public clouds for the Apache Crail distributed data store, which enables efficient data sharing in widely used data processing frameworks, such as Spark. Broadcom is also integrating ReFlex on a system on chip to provide quality of service guarantees to tenants sharing a Flash device over the network.

Selecta: Automating Cloud Resource Management

As cloud computing shifts from providing virtual machines to offering fully managed services, cloud providers are faced with the challenge of controlling how resources are allocated to applications for performance and cost efficiency. In addition to allocating the right amount of compute resources, the choice of storage is often essential, particularly for data-intensive applications. Choosing between different storage technologies, interfaces, and locality is difficult. Automating compute and storage resource management is particularly difficult for cloud providers since they have no upfront knowledge of user application characteristics.

I have shown how we can apply machine learning to select appropriate resource allocations for applications. My system, Selecta, recommends a near-optimal resource allocation for a target application by predicting the application’s execution time across a variety of cloud resource configurations [6, 7]. The system learns by leveraging performance data collected for applications that were previously run across a variety of compute and storage configurations. Using the latent factor collaborative filtering technique, Selecta makes fast, accurate recommendations with sparse training data. Collaborative filtering works by uncovering latent similarities across jobs and resource configurations.

Cloud providers can use Selecta to find resource allocations that minimize the execution time of a job, minimize the cost of running a job, or maximize the performance of a job within a budget. For minimizing job execution time, Selecta has a 94% probability of recommending a configuration that performs within 10% of the true highest performance configuration. For minimizing job execution cost, where cost is a function of resource pricing per unit time and the job execution time, Selecta has an 80% probability of recommending a configuration that achieves within 10% of the true optimal
cost configuration. My evaluation of Selecta across various compute and storage configurations for
data analytics applications also revealed several key insights to guide the design of future storage
systems, such as the need for fine-grain allocation of storage capacity and bandwidth.

**Pocket: A Cloud Storage Service for Serverless Analytics**

My most recent work shows how combining fast access to data and intelligent control of resource
allocations in a fully managed cloud storage service enables new applications. *Serverless computing*

is a fully managed compute service that is becoming increasingly popular, enabling users to quickly
launch thousands of light-weight tasks in the cloud, as opposed to entire virtual machines. Though
initially designed for simple applications that execute a single task in response to an event, several
emerging application frameworks are leveraging serverless computing to exploit massive task parallelism
and achieve near real-time performance for more complex jobs, such as data analytics.

When studying the characteristics of serverless analytics applications, I found that a key challenge
is efficiently managing temporary data that tasks generate and exchange between different stages of
execution in an analytics job [8]. Since serverless computing tasks are short-lived and any data that
a task stores locally is lost when the task terminates, serverless analytics jobs share data using dis-
tributed storage solutions. However, popular storage systems used to share data in serverless comput-
ing applications today (e.g., Amazon S3 and Redis) either introduce significant performance overhead
or they require users to manually manage expensive storage clusters.

Using my work on fast access to disaggregated storage as a foundation, I designed and imple-
mented *Pocket*, a fully managed cloud storage service for efficient data sharing in serverless analyt-
ics [9]. Pocket’s design spans three different planes, which can each be scaled independently: i) a data
plane that stores data objects across multiple nodes and storage technologies, ii) a metadata plane
that routes client read/write requests to the right storage nodes, and iii) a control plane that decides
which resources to allocate for each job and how to autoscale cluster resources based on utilization.
The system leverages optional hints, which can be provided by application frameworks or users, to
determine a job’s latency, throughput and capacity requirements. Pocket automatically assigns a cost-
effective resource allocation for a job. Storage nodes leverage a variety of technologies (e.g., DRAM,
Flash, and disk) with different performance-cost trade-offs and run optimized software for network
storage processing. For example, ReFlex provides fast access to Flash for Pocket clients.

Pocket reduces the average time that serverless tasks in a video analytics application spend on
ephemeral I/O by over 4× compared to S3, a popular object storage service operated by Amazon and
used by serverless computing applications. Pocket offers applications similar performance to a popu-
lar memory-based data store, Redis, while offering automatic resource management and saving close
to 60% in cost. These cost savings come from intelligent data placement across storage technologies
and Pocket’s pay-what-you-use cost model as a fully managed cloud storage service.

**Future Research**

Cloud computing is undergoing unprecedented growth as users migrate their data and computations
to the cloud, datasets continue to grow, and new applications emerge. Meanwhile, the cloud’s hard-
ware landscape is rapidly evolving with numerous accelerators (e.g., GPUs, TPUs, and FPGAs), new
storage technologies (e.g., phase change memory), higher bandwidth networks, and hardware sup-
port for secure enclaves. I am interested in innovating across layers in the cloud system stack, from
application frameworks to hardware interfaces, to expose hardware capabilities to user applications with clean, easy to use abstractions. In future work, I plan to design systems for emerging cloud applications that address their growing needs for high scalability, performance predictability, resource efficiency, and privacy in the cloud.

**Storage for machine learning:** Machine learning (ML) jobs are an important class of applications migrating to the cloud. While significant work has focused on specializing hardware and software for ML computations, managing data is becoming a bottleneck. As organizations collect massive amounts of data, storing and ingesting data at this scale poses several challenges. One challenge I am interested in is finding the right balance between data disaggregation (to support large scale) and data locality (for high performance). Distributing data across nodes enables arbitrarily large datasets to be stored and processed with high parallelism, yet optimizing locality is important because massive datasets are difficult to move. One approach to minimize data movement is to move computation closer to the data. For example, new Flash storage devices with processing capabilities are emerging. I plan to investigate near-storage computing for ML, which requires coordinating distributed data processing, determining which computations are best to support in storage hardware, and designing a new interface to storage that is both easy to use and sufficiently expressive. I am also interested in automating fault tolerance decisions (i.e., encoding and replication) for distributed data by inferring the utility of data objects from computation graphs.

**Opportunistic cloud computing:** Cloud facilities commonly have idle resources, as providers need to reserve spare capacity to accommodate load variations. One way to make cloud computing more resource efficient, and hence significantly cheaper for users, is to run an opportunistic cloud computing service on temporarily idle resources. I would like to design an application framework that automatically scales jobs based on the amount of slack resources available and gracefully handles resource revocations. A key challenge is providing bounded performance guarantees to applications on unreliable infrastructure. Another challenge is designing a programming model that abstracts failure recovery from users while offering sufficient transparency to tune performance trade-offs.

**Performance debugging:** As cloud services become fully managed, how should users reason about the performance and cost of their jobs? For example, if a job experiences high latency, how does a user determine whether this is due to their code or how the cloud provider has scheduled tasks? I would like to design intuitive interfaces for users to express their performance requirements to cloud providers and for cloud services to help users reason about the performance of their jobs. I am interested in enabling parts of a computation to be replayed at low cost and applying machine learning on execution traces to detect anomalies and infer dependencies for performance debugging.

**Privacy in the cloud:** Privacy is a prevalent concern for cloud users. I believe that efficient computation on encrypted data will be increasingly important. One approach is to leverage hardware support for secure enclaves, which protect application code running inside an enclave from malicious code outside the enclave, including operating system code. I would like to help programmers secure their applications with reasonable performance trade-offs with a systematic approach to identify the minimal code base that should run inside an enclave. Since hardware support for enclaves is currently limited to general purpose CPUs, I would also like to co-design software and hardware to provide a uniform enclave abstraction across diverse hardware, including accelerators.

I am fascinated by the research challenges that arise as a result of the evolving application and hardware landscapes in the cloud. With the rapid growth of cloud computing over a wide range of application domains, I believe that operating systems research in this area offers exciting opportunities to have real impact at a broad scale.
References


