一、 系统虚拟化

1. Directvisor: Virtualization for Bare-metal Cloud

Abstract Bare-metal cloud platforms allow customers to rent remote physical servers and install their preferred operating systems and software to make the best of servers' raw hardware capabilities. However, this quest for bare-metal performance compromises cloud manageability. To avoid overheads, cloud operators cannot install traditional hypervisors that provide common manageability functions such as live migration and introspection. We aim to bridge this gap between performance, isolation, and manageability for bare-metal clouds. Traditional hypervisors are designed to limit and emulate hardware access by virtual machines (VM). In contrast, we propose Directvisor – a hypervisor that maximizes a VM's ability to directly access hardware for near-native performance, yet retains hardware control and manageability. Directvisor goes beyond traditional direct-assigned (passthrough) I/O devices by allowing VMs to directly control and receive hardware timer interrupts and inter-processor interrupts (IPIs) besides eliminating most VM exits. At the same time, Directvisor supports seamless (low-downtime) live migration and introspection for such VMs having direct hardware access.

2. Lightweight Kernel Isolation with Virtualization and VM Functions

Abstract Commodity operating systems execute core kernel subsystems in a single address space along with hundreds of dynamically loaded extensions and device drivers. Lack of isolation within the kernel implies that a vulnerability in any of the kernel subsystems or device drivers opens a way to mount a successful attack on the entire kernel. Historically, isolation within the kernel remained prohibitive due to the high cost of hardware isolation primitives. Recent CPUs, however, bring a new set of mechanisms. Extended page-table (EPT) switching with VM functions and memory protection keys (MPKs) provide memory isolation and invocations across boundaries of protection domains with overheads comparable to system calls. Unfortunately, neither MPKs nor EPT switching provide architectural support for isolation of privileged ring 0 kernel code, i.e., control of privileged instructions and well-defined entry points to securely restore state of the system on transition between isolated domains. Our work develops a collection of techniques for lightweight isolation of privileged kernel code. To control execution of privileged instructions, we rely on a minimal hypervisor that transparently deprivileges the system into a non-root VT-x guest. We develop a new isolation boundary that leverages extended page table (EPT) switching with the VMFUNC instruction. We define a set of invariants that allows us to isolate kernel components in the face of an intricate execution model of the kernel, e.g., provide isolation of preemptable, concurrent interrupt handlers. To minimize overheads of virtualization, we develop support for exitless interrupt delivery across isolated domains. We evaluate our approach by developing isolated versions of several device drivers in the Linux kernel.

3. Learn-as-you-go with Megh: Efficient Live Migration of Virtual Machines

Abstract—Cloud providers leverage live migration of virtual machines to reduce energy consumption and allocate resources efficiently in data centers. Each migration decision depends on three questions: when to move a virtual machine, which virtual machine to move and where to move it? Dynamic, uncertain, and heterogeneous workloads running on virtual machines make such decisions difficult. Knowledge-based and heuristics-based algorithms are commonly used to tackle this problem. Knowledge-based algorithms, such as MaxWeight scheduling algorithms, are dependent on the specifics and the dynamics of the targeted Cloud architectures and applications. Heuristics-based algorithms, such as MMT algorithms, suffer from high variance and poor convergence because of their greedy approach. We propose an online reinforcement learning algorithm called Megh. Megh does not require prior knowledge of the workload rather learns the dynamics of workloads as-it-goes. Megh models the problem of energy- and performance-efficient resource management during live migration as a Markov decision process and solves it using a functional approximation scheme. While several reinforcement learning algorithms are proposed to solve this problem, these algorithms remain confined to the academic realm as they face the curse of dimensionality. They are either not scalable in real-time, as it is the case of MadVM, or need an elaborate offline training, as it is the case of Q-learning. These algorithms often incur execution overheads which are comparable with the migration time of a VM. Megh overcomes these deficiencies. Megh uses a novel dimensionality reduction scheme to project the combinatorially explosive state-action space to a polynomial dimensional space with a sparse basis. Megh has the capacity to learn uncertain dynamics and the ability to work in real-time without incurring significant execution overhead. Megh is both scalable and robust. We implement Megh using the CloudSim toolkit and empirically evaluate its performance with the PlanetLab and the Google Cluster workloads. Experiments validate that Megh is more cost-effective, converges faster, incurs smaller execution overhead and is more scalable than MadVM and MMT. An empirical sensitivity analysis explicates the choice of parameters in experiments.

4. Optimizing Live Migration of Multiple Virtual Machines

Abstract—The Cloud computing paradigm is enabling innovative and disruptive services by allowing enterprises to lease computing, storage and network resources from physical infrastructure owners. This shift in infrastructure management responsibility has brought new revenue models and new challenges to Cloud providers. One of those challenges is to efficiently migrate multiple virtual machines (VMs) within the hosting infrastructure with minimum service interruptions. In this paper we first present a live-migration performance testing, captured on a production-level Linux-based virtualization platform, that motivates the need for a better multi-VM migration strategy. We then propose a geometric programming model whose goal is to optimize the bit rate allocation for the live-migration of multiple VMs and minimize the total migration time, defined as a tradeoff cost function between user-perceived downtime and resource utilization time. By solving our geometric program we gained qualitative and quantitative insights on the design of more efficient solutions for multi-VM live migrations. We found that merely few transferring rounds of dirty memory pages are enough to significantly lower the total migration time. We also demonstrated that, under realistic settings, the proposed method converges sharply to an optimal bit rate assignment, making our approach a viable solution for improving current live-migration implementations.

5. Scatter-Gather Live Migration of Virtual Machines

Abstract—We introduce a new metric for live migration of virtual machines (VM) called eviction time defined as the time to evict the state of one or more VMs from the source host. Eviction time determines how quickly the source can be taken offline or its resources repurposed for other VMs. In traditional live migration, such as pre-copy and post-copy, eviction time equals the total migration time because the source is tied up until the destination receives the entire VM. We present Scatter-Gather live migration which decouples the source and destination during migration to reduce eviction time when the destination is slow. The source scatters the memory of VMs to multiple nodes, including the destination and one or more intermediaries. Concurrently, the destination gathers the VMs' memory from the intermediaries and the source. Thus eviction from the source is no longer bottlenecked by the reception speed of the destination. We support simultaneous live eviction of multiple VMs and exploit deduplication to reduce network overhead. Our Scatter-Gather implementation in the KVM/QEMU platform reduces the eviction time by up to a factor of 6 against traditional pre-copy and post-copy while maintaining comparable total migration time when the destination is slower than the source.

6. Securing Time in Untrusted Operating Systems with TimeSeal

Abstract—An accurate sense of elapsed time is essential for the safe and correct operation of hardware, software, and networked systems. Unfortunately, an adversary can manipulate the system's time and violate causality, consistency, and scheduling properties of underlying applications. Although cryptographic techniques are used to secure data, they cannot ensure time security as securing a time source is much more challenging, given that the result of inquiring time must be delivered in a timely fashion. In this paper, we first describe general attack vectors that can compromise a system's sense of time. To counter these attacks, we propose a secure time architecture, TIMESEAL that leverages a Trusted Execution Environment (TEE) to secure time-based primitives. While CPU security features of TEEs secure code and data in protected memory, we show that time sources available in TEE are still prone to OS attacks. TIMESEAL puts forward a high-resolution time source that protects against the OS delay and scheduling attacks. Our TIMESEAL prototype is based on Intel SGX and provides sub-millisecond (msec) resolution as compared to 1-second resolution of SGX trusted time. It also securely bounds the relative time accuracy to msec under OS attacks. In essence, TIMESEAL provides the capability of trusted timestamping and trusted scheduling to critical applications in the presence of a strong adversary. It delivers all temporal use cases pertinent to secure sensing, computing, and actuating in networked systems

7. Using Intel SGX to Protect Authentication Credentials in an Untrusted Operating System

Abstract—An important principle in computational security is to reduce the attack surface, by maintaining the Trusted Computing Base (TCB) small. Even so, no security technique ensures full protection against any adversary. Thus, sensitive applications should be designed with several layers of protection so that, even if a layer might be violated, sensitive content will not be compromised. In 2015, Intel released the Software Guard Extensions (SGX) technology in its processors. This mechanism allows applications to allocate enclaves, which are private memory regions that can hold code and data. Other applications and even privileged code, like the OS kernel and the BIOS, are not able to access enclaves' contents. This paper presents a novel password file protection scheme, which uses Intel SGX to protect authentication credentials in the PAM authentication framework, commonly used in UNIX systems. We defined and implemented an SGX-enabled version of the pam_unix.so authentication module, called UniSGX. This module uses an SGX enclave to handle the credentials informed by the user and to check them against the password file. To add an extra security layer, the password file is stored using SGX sealing. A threat model was proposed to assess the security of the proposed solution. The obtained results show that the proposed solution is secure against the threat model considered, and that its performance overhead is acceptable from the user point of view. The scheme presented here is also suitable to other authentication frameworks.

8. Dynamic VM Scaling: Provisioning and Pricing through an Online Auction

Abstract—Today's IaaS clouds allow dynamic scaling of VMs allocated to a user, according to real-time demand of the user. There are two types of scaling: horizontal scaling (scale-out) by allocating more VM instances to the user, and vertical scaling (scale-up) by boosting resources of VMs owned by the user. It has been a daunting issue how to efficiently allocate the resources on physical servers to meet the scaling demand of users on the go, which achieves the best server utilization and user utility. An accompanying critical challenge is how to effectively charge the incremental resources, such that the economic benefits of both the cloud provider and cloud users are guaranteed. There has been online auction design dealing with dynamic VM provisioning, where the resource bids are not related to each other, failing to handle VM scaling where later bids may rely on earlier bids of the same user. As the first in the literature, this paper designs an efficient, truthful online auction for resource provisioning and pricing in the practical cases of dynamic VM scaling, where: (i) users bid for customized VMs to use in future durations, and can bid again in the following time to increase resources, indicating both scale-up and scale-out options; (ii) the cloud provider packs the demanded VMs on heterogeneous servers for energy cost minimization on the go. We carefully design resource prices maintained for each type of resource on each server to achieve threshold-based online allocation and charging, as well as a novel competitive analysis technique based on submodularity of the offline objective, to show a good competitive ratio is achieved. The efficacy of the online auction is validated through solid theoretical analysis and trace-driven simulations.

9. A new cost-effective mechanism for VM-to-user mapping in cloud data centers

Attracting customers through reward programs is the primary key success for customer-oriented organizations. One of the most famous customer reward programs is applying price discount. In our negotiation-based cloud resource allocation problem, discount price is offered based on both status of a provider and behavior (or loyalty class) of a customer. That is, a resource customer who has appropriate buying behavior and negotiates for resource type instances with high necessity to sell is deserved to receive high price discount from provider. To do this, first, three customer's loyalty classes are defined and customers are classified into theses classes in terms of their previous buying behavior using fuzzy system in name FCLCDS. Second, another fuzzy system in name FNTSVMTDS is designed to determine the value of necessity to sell resource type. The outputs of FCLCDS and FNTSVMTDS are called Loyalty Class (LC) and Necessity to Sell VM Type (NtSVMT), respectively. Finally, a fuzzy system in name FDCDS is proposed to determine the discount coefficient based on both LC and NtSVMT inputs. Furthermore, appropriate times for calculating/re-calculating the discount coefficients that are applied by a resource provider to relax its counter-offers are calculated. We perform extensive simulation experiments to compare our designed negotiator in name FDMDA with MDA and FNSSA. The results show that our designed FDMDA outperforms MDA and FNSSA.

10. CrashTuner: Detecting Crash-Recovery Bugs in Cloud Systems via Meta-Info Analysis

Abstract Crash-recovery bugs (bugs in crash-recovery-related mechanisms) are among the most severe bugs in cloud systems and can easily cause system failures. It is notoriously difficult to detect crash-recovery bugs since these bugs can only be exposed when nodes crash under special timing conditions. This paper presents CrashTuner, a novel fault-injection testing approach to combat crash-recovery bugs. The novelty of CrashTuner lies in how we identify fault-injection points (crash points) that are likely to expose errors. We observe that if a node crashes while accessing meta-info variables, i.e., variables referencing high-level system state information (e.g., an instance of node or task), it often triggers crash-recovery bugs. Hence, we identify crash points by automatically inferring meta-info variables via a log-based static program analysis. Our approach is automatic and no manual specification is required. We have applied CrashTuner to five representative distributed systems: Hadoop2/Yarn, HBase, HDFS, ZooKeeper, and Cassandra. CrashTuner can finish testing each system in 17.39 hours, and reports 21 new bugs that have never been found before. All new bugs are confirmed by the original developers and 16 of them have already been fixed (14 with our patches). These new bugs can cause severe damages such as cluster down or start-up failures.

11. Model-Switching: Dealing with Fluctuating Workloads in Machine-Learning-as-a-Service Systems

Machine learning (ML) based prediction models, and especially deep neural networks (DNNs) are increasingly being served in the cloud in order to provide fast and accurate inferences. However, existing service ML serving systems have trouble dealing with fluctuating workloads and either drop requests or significantly expand hardware resources in response to load spikes. In this paper, we introduce Model-

Switching, a new approach to dealing with fluctuating workloads when serving DNN models. Motivated by the observation that endusers of ML primarily care about the accuracy of responses that are returned within the deadline (which we refer to as effective accuracy), we propose to switch from complex and highly accurate DNN models to simpler but less accurate models in the presence of load spikes. We show that the flexibility introduced by enabling online model switching provides higher effective accuracy in the presence of fluctuating workloads compared to serving using any single model. We implement Model-Switching within Clipper, a state-of-art DNN model serving system, and demonstrate its advantages over baseline approaches.

12. Rethinking Isolation Mechanisms for Datacenter Multitenancy

In theory, trusted execution environments like SGX are promising approaches for isolating datacenter tenants. In practice, the associated hardware primitives suffer from three major problems: side channels induced by microarchitectural co-tenancy; weak guarantees for post-load software integrity; and opaque hardware implementations which prevent third-party security auditing. We explain why these limitations are so problematic for datacenters, and then propose a new approach for trusted execution. This approach, called IME (Isolated Monitor Execution) provides SGX-style memory encryption, but strictly prevents microarchitectural co-tenancy of secure and insecure code. IME also uses a separate, microarchitecturally-isolated pipeline to run dynamic security checks on monitored code, enabling post-load monitoring for security invariants like CFI or type safety. Finally, an IME processor exports a machine-readable description of its microarchitectural implementation, allowing tenants to reason about the security properties of a particular IME instance.

13. More IOPS for Less: Exploiting Burstable Storage in Public Clouds

Burstable storage is a public cloud feature that enhances cloud storage volumes with credits that can be used to boost performance temporarily. These credits can be exchanged for increased storage throughput, for a short period of time, and are replenished over time. We examine how burstable storage can be leveraged to reduce cost and/or improve performance for three use cases with different data-longevity requirements: traditional persistent storage, caching, and ephemeral storage. Although cloud storage volumes are typically priced by capacity, we find that each AWS gp2 volume starts with the same number of burst credits. Exploiting that fact, we find that aggressive interchanging of large numbers of small short-term volumes can increase IOPS by up to 100× at a cost increase of only 10–40%. Compared to an AWS io1 volume provisioned for the same performance, such interchanging reduces cost by 97.5%.

14. JACKPOT: Online Experimentation of Cloud Microservices

Online experimentation is an agile software development practice, which plays a central role in enabling rapid innovation. It helps shorten code delivery cycles, which is critical for companies to survive in a competitive software-driven market. Recent advances in cloud computing, including the maturity of container-based technologies and cloud infrastructure, as well as the advent of service meshes, have created an opportunity to broaden the scope of online experimentation and further increase developers' agility. In this paper, we propose a novel formulation for online experimentation of cloud applications which generalizes traditional approaches applied to web and mobile applications by incorporating the unique challenges posed by cloud environments. To enable practitioners to apply our formulation, we develop and present JACKPOT, a system for online cloud experimentation in the presence of multiple interacting microservices. We discuss an initial prototype of JACKPOT along with a preliminary evaluation of this prototype based on experiments on a public container cloud.

15. Happiness index: Right-sizing the cloud’s tenant-provider interface

Cloud providers and their tenants have a mutual interest in identifying optimal configurations in which to run tenant jobs, i.e., ones that achieve tenants’ performance goals at minimum cost; or ones that maximize performance within a specified budget. However, different tenants may have different performance goals that are opaque to the provider. A consequence of this opacity is that providers today typically offer fixed bundles of cloud resources, which tenants must themselves explore and choose from. This is burdensome for tenants and can lead to choices that are sub-optimal for both parties. We thus explore a simple, minimal interface, which lets tenants communicate their happiness with cloud infrastructure to the provider, and enables the provider to explore resource configurations that maximize this happiness. Our early results indicate that this interface could strike a good balance between enabling efficient discovery of application resource needs and the complexity of communicating a full description of tenant utility from different configurations to the provider.

二、 容器与无服务化

1. HyScale: Hybrid and Network Scaling of Dockerized Microservices in Cloud Data Centres

ken to allow for applications to scale based on the demands of its users while still accommodating flexibility in development. Recently, microservices architectures have garnered the attention of many organizations—providing higher levels of scalability, availability, and fault isolation. Many organizations choose to host their microservices architectures in cloud data centres to offset costs. Incidentally, data centres become over-encumbered during peak usage hours and underutilized during off-peak hours. Traditional microservice scaling methods perform either horizontal or vertical scaling exclusively. When used in combination, however, these methods offer complementary benefits and compensate for each other’s deficiencies. To leverage the high availability of horizontal scaling and the fine-grained resource control of vertical scaling, we developed two novel hybrid autoscaling algorithms and a dedicated network scaling algorithm and benchmarked them against Google’s popular Kubernetes horizontal autoscaling algorithm. Results indicated up to 1.49x speedups in response times for our hybrid algorithms, and 1.69x speedups for our network algorithm under high-burst network loads. Index Terms—Docker, microservices, autoscaling, cloud.

2. Agile Cold Starts for Scalable Serverless

The Serverless or Function-as-a-Service (FaaS) model capitalizes on lightweight execution by packaging code and dependencies together for just-in-time dispatch. Often a container environment has to be set up afresh– a condition called “cold start", and in such cases, performance suffers and overheads mount, both deteriorating rapidly under high concurrency. Caching and reusing previously employed containers ties up memory and risks information leakage. Latency for cold starts is frequently due to work and wait-times in setting up various dependencies – such as in initializing networking elements. This paper proposes a solution that pre-crafts such resources and then dynamically reassociates them with baseline containers. Applied to networking, this approach demonstrates an order of magnitude gain in cold starts, negligible memory consumption, and flat startup time under rising concurrency.

3. Docker-sec: A Fully Automated Container Security Enhancement Mechanism

Abstract—The popularity of containers is constantly rising in the virtualization landscape, since they incur significantly less overhead than Virtual Machines, the traditional hypervisorbased counterparts, while enjoying better performance. However, containers pose significant security challenges due to their direct communication with the host kernel, allowing attackers to break into the host system and co-located containers more easily than Virtual Machines. Existing security hardening mechanisms are based on the enforcement of Mandatory Access Control rules, which exclusively allow specified, desired operations. However, these mechanisms entail explicit knowledge of the container functionality and behavior and require manual intervention and setup. To overcome these limitations, we present Docker-sec, a user-friendly mechanism for the protection of Docker containers throughout their lifetime via the enforcement of access policies that correspond to the anticipated (and legitimate) activity of the applications they enclose. Docker-sec employs two mechanisms: (a) Upon container creation, it constructs an initial, static set of access rules based on container configuration parameters; (b) During container runtime, the initial set is enhanced with additional rules that further restrict the container's capabilities, reflecting the actual application operations. Through a rich interaction with our system the audience will experience firsthand how Docker-sec can successfully protect containers from zero-day vulnerabilities in an automatic manner, with minimal overhead on the application performance.

4. Learning from, Understanding, and Supporting DevOps Artifacts for Docker

With the growing use of DevOps tools and frameworks, there is an increased need for tools and techniques that support more than code. The current state-of-the-art in static developer assistance for tools like Docker is limited to shallow syntactic validation. We identify three core challenges in the realm of learning from, understanding, and supporting developers writing DevOps artifacts: (i) nested languages in DevOps artifacts, (ii) rule mining, and (iii) the lack of semantic rule-based analysis. To address these challenges we introduce a toolset, binnacle, that enabled us to ingest 900,000 GitHub repositories. Focusing on Docker, we extracted approximately 178,000 unique Dockerfiles, and also identified a Gold Set of Dockerfiles written by Docker experts. We addressed challenge (i) by reducing the number of effectively uninterpretable nodes in our ASTs by over 80% via a technique we call phased parsing. To address challenge (ii), we introduced a novel rule-mining technique capable of recovering two-thirds of the rules in a benchmark we curated. Through this automated mining, we were able to recover 16 new rules that were not found during manual rule collection. To address challenge (iii), we manually collected a set of rules for Dockerfiles from commits to the files in the Gold Set. These rules encapsulate best practices, avoid docker build failures, and improve image size and build latency. We created an analyzer that used these rules, and found that, on average, Dockerfiles on GitHub violated the rules five times more frequently than the Dockerfiles in our Gold Set. We also found that industrial Dockerfiles fared no better than those sourced from GitHub. The learned rules and analyzer in binnacle can be used to aid developers in the IDE when creating Dockerfiles, and in a post-hoc fashion to identify issues in, and to improve, existing Dockerfiles.

5. Large-Scale Analysis of the Docker Hub Dataset

Abstract—Docker containers have become a prominent solution for supporting modern enterprise applications due to the highly desirable features of isolation, low overhead, and efficient packaging of the execution environment. Containers are created from images which are shared between users via a Docker registry. The amount of data Docker registries store is massive; for example, Docker Hub, a popular public registry, stores at least half a million public images. In this paper, we analyze over 167 TB of uncompressed Docker Hub images, characterize them using multiple metrics and evaluate the potential of filelevel deduplication in Docker Hub. Our analysis helps to make conscious decisions when designing storage for containers in general and Docker registries in particular. For example, only 3% of the files in images are unique, which means file-level deduplication has a great potential to save storage space for the registry. Our findings can motivate and help improve the design of data reduction, caching, and pulling optimizations for registries.

6. Evaluating Docker for Lightweight Virtualization of Distributed and Time-Sensitive Applications in Industrial Automation

Abstract—A trend, accompanying the change of automation systems and their architectures, is the virtualization of software components. Virtualization strengthens platformindependent development and the provision of secure and isolated applications. Virtualization introduces well-defined interfaces to strengthen modularity, which facilitates the scalability of applications. However, virtualization includes additional software components and layers and, thus, additional computing costs. This additional effort can conflict with the real-time requirements of automation processes. Current research lacks the investigation of the time behavior of container-based virtualizations concerning their use in real-time systems. An assessment concerning real-time applications is required to prepare it for use in industrial automation. This article examines the effects of virtualization on the time delays of a software component based on Docker containers by providing measurements on a hardware testbed in a realistic use case. The experiments indicate that Docker virtualization can meet soft real-time requirements and can be used in industrial automation.

7. Architecture for Predicting Live Video Transcoding Performance on Docker Containers

Abstract—Video can be streamed live with different applications (e.g. YouTube Live, Periscope). Typically, the video content is adapted for end users based on receiving client's capabilities, and network bandwidth. The adaptation is realized with different video representations, which are created by transcoding the original video content. When video is streamed live, transcoding has to be completed within real time constraints, which is a computationally demanding process. Particularly, live transcoding should be enabled efficiently by a content distributor to minimize resource provisioning costs. The contribution of this paper is an architecture for predicting live video transcoding performance on a Docker-based platform. Particularly, cloud resource management for live video transcoding has been focused on. A model was trained based on measurements in different transcoding configurations. Offline evaluation results indicate that live transcoding speed or CPU usage can be predicted with 3-8 % accuracy. When video is transcoded on virtual machines based on predictions in a prototype system (live), live transcoding speed prediction accuracy is within a similar range as the offline performance, but worse for CPU usage prediction (5-15 %). In most cases the specified range for transcoding speed and CPU usage can be achieved at least with a precision of 76 %.

8. A Holistic Evaluation of Docker Containers for Interfering Microservices

Abstract— Advancement of container technology (e.g. Docker, LXC, etc.) transformed the virtualization concept by providing a lightweight alternative to hypervisors. Docker has emerged as the most popular container management tool. Recent research regarding the comparison of container with hypervisor and bare-metal demonstrates that the container can accomplish bare-metal performance in almost all case. However, the current literature lacks an in-depth study on the experimental evaluation for understanding the performance interference between microservices that are hosted within a single or across multiple containers. In this paper, we have presented the experimental study on the performance evaluation of Docker containers running heterogeneous set of microservices concurrently. We have conducted a comprehensive set of experiments following CEEM (Cloud Evaluation Experiment Methodology) to measure the interference between containers running either competing or independent microservices. We have also considered the effects of constraining the resources of a container by explicitly specifying the cgroups. We have evaluated the performance of containers in terms of inter-container (caused by two concurrent executing containers) and intra-container (caused between two microservices executing inside a container) interference which is almost neglected in the current literature. The evaluation results can be utilized to model the interference effect for smart resource provisioning of microservices in the containerized environment.

9. Serverless Boom or Bust? An Analysis of Economic Incentives

Serverless computing is a new paradigm that promises to free cloud users from the burden of having to provision and manage resources. However, the degree to which serverless computing will replace provisioned servers remains an open question. To address this, we develop an economic model that aims to quantify the value of serverless to providers and customers. A simple model of incentives for rational providers and customers allows us to see, in broad strokes, when and why serverless technologies are worth pursuing. By characterizing the conditions under which mutually beneficial economic incentives exist, our model suggests that many classes of customers can already benefit from switching to a serverless model and taking advantage of autoscaling at today’s price points. Our model also helps characterize technical research directions that would be likely to have impact in the market.

10. Virtual Network Functions as Real-Time Containers in Private Clouds

Abstract—This paper presents preliminary results from our on-going research for ensuring stable performance of co-located distributed cloud services in a resource-efficient way. It is based on using a real-time CPU scheduling policy to achieve a fine-grain control of the temporal interferences among real-time services running in co-located containers. We present results obtained applying the method to a synthetic application running within LXC containers on Linux, where a modified kernel has been used that includes our real-time scheduling policy.

11. Migrating VM workloads to Containers: Issues and Challenges

Abstract—Virtualization technologies such as KVM and XEN have served the purpose of workload consolidation while providing required isolation. With the advent of Linux Docker platform, micro-service architectures have gained popularity. Adaptation of containers have changed the way, the applications are architected, developed, deployed and managed. Compared to Virtualization, containers provide lightweight alternative to co-host multiple applications on single server at the cost of isolation. This paper highlights the issues and challenges associated with migrating VM based workloads to Container platforms. A systematic analysis, illustrated through a representative application benchmark chosen from a real-life e-commerce private cloud setup is used to present these aspects. Specifically, the application workflow that is currently hosted on VMs is chosen and re-casted on to containers platform and a critical study is conducted based on resource sharing, concurrency, isolation and dependability parameters.

12. Lambda Containers: A Comprehensive Anti-Tamper Framework for Games by Simulating Client Behavior in a Cloud

Abstract—As the rapid growth of smartphones, online games are starting to execute core logic on the client-side to enable rich user interactions. This trend makes it difficult to detect illegal modifications to games. This paper presents a comprehensive anti-tamper framework for games, which detects tampering by comparing the game state information submitted by the client and the genuine state information simulated by transparently executing the genuine app on a cloud. The key technology is a novel container pool that models the execution environment of apps as lambda functions by defining an app as a first-order function and a container as a higher-order function. This system, which is a meta-level system over the existing container system, provides a functional programming model designed to statically encapsulate heterogeneous environment of apps into containers and to dynamically simulate a specific game scene by injecting the game status into containers. This system optimally executes this process by translating the meta-level operations into the parameters to conventional containers. Experimental results using a real game product showed that 110 containers run in a scalable manner on a single server with 72 logical CPU cores installed on a public cloud, without sacrificing either the response or throughput performance.

13. A Comparative Study of Containers and Virtual Machines in Big Data Environment. IEEE CLOUD 2018: 178-185

Abstract—Container technique is gaining increasing attention in recent years and has become an alternative to traditional virtual machines. Some of the primary motivations for the enterprise to adopt the container technology include its conveniency to encapsulate and deploy applications, lightweight operations, as well as efficiency and flexibility in resources sharing. However, there still lacks an in-depth and systematic comparison study on how big data applications, such as Spark jobs, perform between a container environment and a virtual machine environment. In this paper, by running various Spark applications with different configurations, we evaluate the two environments from many interesting aspects, such as how convenient the execution environment can be set up, what are makespans of different workloads running in each setup, how efficient the hardware resources, such as CPU and memory, are utilized, and how well each environment can scale. The results show that compared with virtual machines, containers provide a more easy-to-deploy and scalable environment for big data workloads. The research work in this paper can help practitioners and researchers to make more informed decisions on tuning their cloud environment and configuring the big data applications, so as to achieve better performance and higher resources utilization

14. Managed Containers: A Framework for Resilient Containerized Mission Critical Systems

Abstract— Traditional defense mechanisms are insufficient for protecting containerized mission critical systems. These systems are mostly based on cloud-based images (e.g., Docker) that need to be always-on-always-connected. High availability and data integrity become crucial to deliver their mission. Unable to guarantee uncompromisable security and given that systems will inevitably be attacked, we must change our goals to emphasize resiliency and mission survivability. This paper presents work-in-progress to create a framework for cloudbased container resiliency. Our resilient framework makes use of Linux containers to provide resiliency to services. It is designed to orchestrate and manage the container lifecycle while enforcing security and returning a service to a previous secure state in case of a cyber-attack. It achieves this by expanding upon the generic container model with additional layers that enhance security and increase auditability. We coin the term “managed containers” to refer to the enhanced containers managed by our resilient framework. In case of an anomaly, it generates a report and allows the operator to choose a resiliency strategy. In our tests, our framework is able to securely recover from a fault in less time than a pure Docker solution while protecting against the most common container vulnerabilities.

三、 分布式存储技术

1. File Systems Unfit as Distributed Storage Backends: Lessons from 10 Years of Ceph Evolution

For a decade, the Ceph distributed file system followed the conventional wisdom of building its storage backend on top of local file systems. This is a preferred choice for most distributed file systems today because it allows them to benefit from the convenience and maturity of battle-tested code. Ceph’s experience, however, shows that this comes at a high price. First, developing a zero-overhead transaction mechanism is challenging. Second, metadata performance at the local level can significantly affect performance at the distributed level. Third, supporting emerging storage hardware is painstakingly slow. Ceph addressed these issues with BlueStore, a new backend designed to run directly on raw storage devices. In only two years since its inception, BlueStore outperformed previous established backends and is adopted by 70% of users in production. By running in user space and fully controlling the I/O stack, it has enabled space-efficient metadata and data checksums, fast overwrites of erasure-coded data, inline compression, decreased performance variability, and avoided a series of performance pitfalls of local file systems. Finally, it makes the adoption of backwards-incompatible storage hardware possible, an important trait in a changing storage landscape that is learning to embrace hardware diversity.

2. SplitFS: Reducing Software Overhead in File Systems for Persistent Memory

We present SplitFS, a file system for persistent memory (PM) that reduces software overhead significantly compared to state-of-the-art PM file systems. SplitFS presents a novel split of responsibilities between a user-space library file system and an existing kernel PM file system. The user-space library file system handles data operations by intercepting POSIX calls, memory-mapping the underlying file, and serving the read and overwrites using processor loads and stores. Metadata operations are handled by the kernel PM file system (ext4 DAX). SplitFS introduces a new primitive termed relink to efficiently support file appends and atomic data operations. SplitFS provides three consistency modes, which different applications can choose from, without interfering with each other. SplitFS reduces software overhead by up-to 4× compared to the NOVA PM file system, and 17× compared to ext4 DAX. On a number of micro-benchmarks and applications such as the LevelDB key-value store running the YCSB benchmark, SplitFS increases application performance by up to 2× compared to ext4 DAX and NOVA while providing similar consistency guarantees.

3. RECIPE : Converting Concurrent DRAM Indexes to Persistent-Memory Indexes

We present Recipe, a principled approach for converting concurrent DRAM indexes into crash-consistent indexes for persistent memory (PM). The main insight behind Recipe is that isolation provided by a certain class of concurrent in-memory indexes can be translated with small changes to crash-consistency when the same index is used in PM. We present a set of conditions that enable the identification of this class of DRAM indexes, and the actions to be taken to convert each index to be persistent. Based on these conditions and conversion actions, we modify five different DRAM indexes based on B+ trees, tries, radix trees, and hash tables to their crash-consistent PM counterparts. The effort involved in this conversion is minimal, requiring 30–200 lines of code. We evaluated the converted PM indexes on Intel DC Persistent Memory, and found that they outperform state-of-the-art, hand-crafted PM indexes in multi-threaded workloads by up-to 5.2×. For example, we built P-CLHT, our PM implementation of the CLHT hash table by modifying only 30 LOC. When running YCSB workloads, P-CLHT performs up to 2.4× better than Cacheline-Conscious Extendible Hashing (CCEH), the state-of-the-art PM hash table.

4. A Cloud-native Architecture for Replicated Data Services

Many services replicate data for fault-tolerant storage of the data and high-availability of the service. When deployed in the cloud, the replication performed by these services provides the desired high-availability but does not provide significant additional fault-tolerance for the data. This is because cloud deployments use fault-tolerant storage services instead of the simple local disks that many replicated data services were designed to use. Because the cloud storage services already provide fault-tolerance for the data, the extra replicas create unnecessary cost in running the service. However, replication is still needed for high-availability of the service itself. In this paper, we explore types of replicated data services and how they can be mapped onto various classes of cloud storage. We then propose a general architectural pattern that can be used to: (1) limit additional storage resulting in monetary cost saving, (2) while keeping the same performance for the service, and (3) maintaining the same high-availability of the services and the durability guarantees for the data. We prototype our approach in two popular open-source replicated data services, Kafka and Cassandra, and show that with relatively little modification these systems can be deployed for a fraction of the storage cost without affecting the availability guarantees, durability guarantees, or performance.

5. The Design of Fast Content-Defined Chunking for Data Deduplication Based Storage Systems

Abstract—Content-Defined Chunking (CDC) has been playing a key role in data deduplication systems recently due to its high redundancy detection ability. However, existing CDC-based approaches introduce heavy CPU overhead because they declare the chunk cut-points by computing and judging the rolling hashes of the data stream byte by byte. In this article, we propose FastCDC, a Fast and efficient Content-Defined Chunking approach, for data deduplication-based storage systems. The key idea behind FastCDC is the combined use of five key techniques, namely, gear based fast rolling hash, simplifying and enhancing the Gear hash judgment, skipping sub-minimum chunk cut-points, normalizing the chunk-size distribution in a small specified region to address the problem of the decreased deduplication ratio stemming from the cut-point skipping, and last but not least, rolling two bytes each time to further speed up CDC. Our evaluation results show that, by using a combination of the five techniques, FastCDC is 3-12X faster than the state-of-the-art CDC approaches, while achieving nearly the same and even higher deduplication ratio as the classic Rabin-based CDC. In addition, our study on the deduplication throughput of FastCDC-based Destor (an open source deduplication project) indicates that FastCDC helps achieve 1.2-3.0X higher throughput than Destor based on state-of-the-art chunkers.

6. Towards Usable Cloud Storage Auditing

Abstract—Cloud storage security has gained considerable research efforts with the wide adoption of cloud computing. As a security mechanism, researchers have been investigating cloud storage auditing schemes that enable a user to verify whether the cloud keeps the user’s outsourced data undamaged. However, existing schemes have usability issues in compatibility with existing real world cloud storage applications, error-tolerance, and efficiency. To mitigate this usability gap, this article proposes a new general cloud storage auditing scheme that is more usable. The proposed scheme uses the idea of integrating linear error correcting codes and linear homomorphic authentication schemes together. This integration uses only one additional block to achieve error tolerance and authentication simultaneously. To demonstrate the power of the general construction, we also propose one detailed scheme based on the proposed general construction using the Reed Solomon code and the universal hash based MAC authentication scheme, both of which are implemented over the computation-efficient Galois field GFð28Þ. We also show that the proposed scheme is secure under the standard definition. Moreover, we implemented and open-sourced the proposed scheme. Experimental results show that the proposed scheme is orders of magnitude more efficient than the state-of-the-art scheme.

7. CROCUS: Enabling Computing Resource Orchestration for Inline Cluster-Wide Deduplication on Scalable Storage Systems

Abstract—Inline deduplication dramatically improves storage space utilization. However, it degrades I/O throughput due to computeintensive deduplication operations such as chunking, fingerprinting or hashing of chunk content, and redundant lookup I/Os over the network in the I/O path. In particular, the fingerprint or hash generation of content contributes largely to the degraded I/O throughput and is computationally expensive. In this article, we propose CROCUS, a framework that enables compute resource orchestration to enhance cluster-wide deduplication performance. In particular, CROCUS takes into account all compute resources such as local and remote {CPU, GPU} by managing decentralized compute pools. An opportunistic Load-Aware Fingerprint Scheduler (LAFS), distributes and offloads compute-intensive deduplication operations in a load-aware fashion to compute pools. CROCUS is highly generic and can be adopted in both inline and offline deduplication with different storage tier configurations. We implemented CROCUS in Ceph scale-out storage system. Our extensive evaluation shows that CROCUS reduces the fingerprinting overhead by 86 percent with 4KB chunk size compared to Ceph with baseline deduplication while maintaining high disk-space savings. Our proposed LAFS scheduler, when tested in different internal and external contention scenarios also showed 54 percent improvement over a fixed or static scheduling approach.

8. Making Application-Level Crash Consistency Practical on Flash Storage

Abstract—We present the design, implementation, and evaluation of a new file system, called ACCFS, supporting application-level crash consistency as its first-class citizen functionality. With ACCFS, application data can be correctly recovered in the event of system crashes without any complex update protocol at the application level. With the help of the SHARE interface supporting atomic address remapping at the flash storage layer, ACCFS can easily and efficiently achieve crash consistency as well as single-write journaling. We prototyped ACCFS by slightly modifying the full data journal mode in ext4, implemented the SHARE interface as firmware in a commercial SSD available in the market, and carried out various experiments by running ACCFS on top of the SSD. Our preliminary experimental results are very promising. For instance, the performance of an OLTP benchmark using MySQL/InnoDB engine can be boosted by more than 2–6x by offloading the responsibility of guaranteeing the atomic write of MySQL data pages from the InnoDB engine's own journaling mechanism to ACCFS. This impressive performance gain is in part due to the single-write journaling in ACCFS and in part comes from the fact that the frequent fsync() calls caused by the complex update protocol at the application level can be avoided. ACCFS is a practical solution for the crash consistency problem in that (1) the SHARE interface can be, like the TRIM command, easily supported by commercial SSDs, (2) it can be embodied with a minor modification on the existing ext4 file system, and (3) the existing applications can be made crash consistent simply by opening files in O_ATOMIC mode while the legacy applications can be run without any change.

9. Scalable and Adaptive Data Replica Placement for Geo-Distributed Cloud Storages

Abstract—In geo-distributed cloud storage systems, data replication has been widely used to serve the ever more users around the world for high data reliability and availability. How to optimize the data replica placement has become one of the fundamental problems to reduce the inter-node traffic and the system overhead of accessing associated data items. In the big data era, traditional solutions may face the challenges of long running time and large overheads to handle the increasing scale of data items with time-varying user requests. Therefore, novel offline community discovery and online community adjustment schemes are proposed to solve the replica placement problem in a scalable and adaptive way. The offline scheme can find a replica placement solution based on the average read/write rates for a certain period of time. The scalability can be achieved as 1) the computation complexity is linear to the amount of data items and 2) the data-node communities can evolve in parallel for a distributed replica placement. Furthermore, the online scheme is adaptive to handle the bursty data requests, without the need to completely override the existing replica placement. Driven by realworld data traces, extensive performance evaluations demonstrate the effectiveness of our design to handle large-scale datasets

10. ESetStore: An Erasure-Coded Storage System With Fast Data Recovery

Abstract—Erasure codes have been used extensively in large-scale storage systems to reduce the storage overhead of triplication-based storage systems. One key performance issue introduced by erasure codes is the long time needed to recover from a single failure, which occurs constantly in large-scale storage systems. We present ESetStore, a prototype erasure-coded storage system that aims to achieve fast recovery from failures. ESetStore is novel in the following aspects. We proposed a data placement algorithm named ESet for our ESetStore that can aggregate adequate I/O resources from available storage servers to recover from each single failure. We designed and implemented efficient read and write operations on our erasure-coded storage system via effective use of available I/O and computation resources. We evaluated the performance of ESetStore with extensive experiments on a cluster with 50 storage servers. The evaluation results demonstrate that our recovery performance can obtain linear performance growth by harvesting available I/O resources. With our defined parameter recovery I/O parallelism under some mild conditions, we can achieve optimal recovery performance, in which ESet enables minimal recovery time. Rather than being an alternative to improve recovery performance, our work can be an enhancement for existing solutions, such as Partial-parallel-repair (PPR), to further improve recovery performance.

11. Wiera: Policy-Driven Multi-Tiered Geo-Distributed Cloud Storage System

Abstract—Multi-tiered geo-distributed cloud storage systems must tame complexity at many levels: uniform APIs for storage access, supporting flexible storage policies that meet a wide array of application metrics, determining an optimal data placement, handling uncertain network dynamics and access dynamism, and operating across many levels of heterogeneity both within and across data-centers (DCs). In this paper, we present an integrated solution called Wiera. Wiera enables the specification of data management policies both within a local DC and across DCs. Such policies enable the user to optimize for cost, performance, reliability, durability, and consistency, and to express their tradeoffs. In addition, Wiera determines an optimal data placement for the user to meet their desired tradeoffs easily in such an environment. A key aspect of Wiera is first-class support for dynamism due to network, workload, and access patterns changes. As far as we know, Wiera is the first geo-distributed cloud storage system which handles dynamism actively at run-time. Wiera allows unmodified applications to reap the benefits of flexible data/storage policies by externalizing the policy specification. We show how Wiera enables a rich specification of dynamic policies using a concise notation and describe the design and implementation of the system. We have implemented a Wiera prototype on multiple cloud environments, AWS and Azure, that illustrates potential benefits from managing dynamics and in using multiple cloud storage tiers both within and across DCs.

12. Towards Unaligned Writes Optimization in Cloud Storage With High-Performance SSDs

Abstract—NVMe SSDs provide extremely high performance and have been widely deployed in distributed object storage systems in data centers. However, we observe that there are still severe performance degradation and write amplification under the unaligned writes scenario with high-performance SSDs. In this article, we identify that the RMW sequence which is used to handle the unaligned writes incurs severe overhead in the data path. Besides, unaligned writes incur additional metadata management overhead in the block map table. To address these problems, we propose an object-based device system named NVStore to optimize the unaligned writes in cloud storage with NVMe SSDs. NVStore provides a Flexible Cache Management to reduce the RMW operations while supporting lazy page sync and ensuring data consistency. To optimize the metadata management, NVStore proposes a KV Affinity Metadata Management which co-designs the block map and key-value store to provides a flattened and decoupled metadata management. Evaluations show that NVStore provides at most 6.11 bandwidth of BlueStore in the cluster. Besides, NVStore can reduce at most 94.7 percent of the write traffic from metadata under unaligned writes compared to BlueStore and achieves smaller data write traffic which is about 50 percent of BlueStore and 65.7 percent of FileStore.

13. An improved query optimization process in big data using ACO-GA algorithm and HDFS map reduce technique

Storing as well as retrieving the data on a specifc time frame is fundamental for any application today. So an efciently designed query permits the user to get results in the desired time and creates credibility for the corresponding application. To avoid the difculty in query optimization, this paper proposed an improved query optimization process in big data (BD) using the ACO-GA algorithm and HDFS mapreduce. The proposed methodology consists of ‘2’ phases, namely, BD arrangement and query optimization phases. In the frst phase, the input data is pre-processed by fnding the hash value (HV) using the SHA-512 algorithm and the removal of repeated data using the HDFS map-reduce function. Then, features such as closed frequent pattern, support, and confdence are extracted. Next, the support and confdence are managed by using the entropy calculation. Centered on the entropy calculation, the related information is grouped by using Normalized K-Means (NKM) algorithm. In the 2nd phase, the BD queries are collected, and then the same features are extorted. Next, the optimized query is found by utilizing the ACO-GA algorithm. Finally, the similarity assessment process is performed. The experimental outcomes illustrate that the algorithm outperformed other existent algorithms.

四、 并行数据处理技术

1. Pruning techniques for parallel processing of reverse top-k queries

In this paper, we address the problem of processing reverse top-k queries in a parallel setting. Given a database of objects, a set of user preferences, and a query object q, the reverse top-k query returns the subset of user preferences for which the query object belongs to the top-k results. Although recently the reverse top-k query operator has been studied extensively, its CPU-intensive nature results in prohibitively expensive processing cost, when applied on vast-sized data sets. This limitation motivates us to explore a scalable parallel processing solution, in order to enable reverse top-k processing over distributed large sets of input data in reasonable execution time. We present an algorithmic framework for the problem, in which diferent algorithms can be instantiated, targeting a generic parallel setting. We describe a parallel algorithm (DiPaRT) that exploits basic pruning properties and is provably correct, as an instantiation of the framework. Furthermore, we introduce novel pruning properties for the problem, and propose DiPaRT+ as another instance of the algorithmic framework, which ofers improved efciency and scales gracefully. All algorithms are implemented in MapReduce, and we provide a wide set of experiments that demonstrate the improved efciency of DiPaRT+ using data sets that are four orders of magnitude larger than those handled by centralized approaches.

2. Chronos: A Unifying Optimization Framework for Speculative Execution of Deadline-critical MapReduce Jobs

Abstract—Meeting desired application deadlines in cloud processing systems such as MapReduce is crucial as the nature of cloud applications is becoming increasingly mission-critical and deadline-sensitive. It has been shown that the execution times of MapReduce jobs are often adversely impacted by a few slow tasks, known as stragglers, which result in high latency and deadline violations. While a number of strategies have been developed in existing work to mitigate stragglers by launching speculative or clone task attempts, none of them provide a quantitative framework that optimizes the speculative execution for offering guaranteed Service Level Agreements (SLAs) to meet application deadlines. In this paper, we bring several speculative scheduling strategies together under a unifying optimization framework, called Chronos, which defines a new metric, Probability of Completion before Deadlines (PoCD), to measure the probability that MapReduce jobs meet their desired deadlines. We systematically analyze PoCD for popular strategies including Clone, Speculative-Restart, and SpeculativeResume, and quantify their PoCD in closed-form. The results illuminate an important tradeoff between PoCD and the cost of speculative execution, measured by the total (virtual) machine time required under different strategies. We propose an optimization problem to jointly optimize PoCD and execution cost in different strategies, and develop an algorithmic solution that is guaranteed to be optimal. Chronos is prototyped on Hadoop MapReduce and evaluated against three baseline strategies using both experiments and trace-driven simulations, and achieves 50% net utility increase with up to 80% PoCD and 88% cost improvements.

3. Improved Intermediate Data Management for MapReduce Frameworks

Abstract—MapReduce is a popular distributed framework for big data analysis. However, the current MapReduce framework is insufficiently efficient in handling intermediate data, which may cause bottlenecks in I/O operations, computation, and network bandwidth. Previous work addresses the I/O problem by aggregating map task outputs (i.e. intermediate data) for each single reduce task on one machine. Unfortunately, when there are a large number of reduce tasks, their concurrent requests for intermediate data generate a large amount of I/O operations. In this paper, we present APA (Aggregation, Partition, and Allocation), a new intermediate data management system for the MapReduce framework. APA aggregates the intermediate data from the map tasks in each rack to one file, and the file host pushes the needed intermediate data to each reduce task. Thus, it reduces the number of disk seeks involved in handling intermediate data within one job. Rather than evenly distributing the intermediate data among reduce tasks based on the keys as in current MapReduce, APA partitions the intermediate data to balance the execution latency of different reduce tasks. APA further decides where to allocate each reduce task to minimize the intermediate data transmission time between map tasks and reduce tasks. Through experiments on a real MapReduce Hadoop cluster using the HiBench benchmark suite, we show that APA improves the performance of the current Hadoop by 40%-50%.

4. Poster: Efficiently Finding Minimal Failing Input in MapReduce Programs

Debugging of distributed computing model programs like MapReduce is a difficult task. That’s why prior studies only focus on finding and fixing bugs in early stages of program development. Delta debugging tries to find minimal failing input in sequential programs by dividing inputs into subsets and testing these subsets one-by-one. But no prior work tries to find minimal failing input in distributed programs like MapReduce. In this paper, we present MapRedDD, a framework to efficiently find minimal failing input in MapReduce programs. MapRedDD employs failing input selection technique, focused on identifying the failing input subset in the single run of MapReduce program with multiple input subsets instead of testing each subset separately. This helps to reduce the number of executions of MapReduce program for each input subset and overcome the overhead of job submission, job scheduling and final outcome retrieval. Our work can efficiently find the minimal failing input in the number of executions equal to the number of inputs to MapReduce program N as opposed to the number of executions of MapReduce program equal to the number of input subsets $2N - 1$ in worst case for binary search invariant algorithm to find minimal failing input.

5. Combining Data Duplication and Graph Reordering to Accelerate Parallel Graph Processing

Performance of single-machine, shared memory graph processing is affected by expensive atomic updates and poor cache locality. Data duplication, a popular approach to eliminate atomic updates by creating thread-local copies of shared data, incurs extreme memory overheads due to the large sizes of typical input graphs. Even memory-efficient duplication strategies that exploit the power-law structure common to many graphs (by duplicating only the highly-connected "hub" vertices) suffer from overheads for having to dynamically identify the hub vertices. Degree Sorting, a popular graph reordering technique that re-assigns hub vertices consecutive IDs in a bid to improve spatial locality, is effective for single-threaded graph applications but suffers from increased false sharing in parallel executions. The main insight of this work is that the combination of data duplication and Degree Sorting eliminates the overheads of each optimization. Degree Sorting improves the efficiency of data duplication by assigning hub vertices consecutive IDs which enables easy identification of the hub vertices. Additionally, duplicating the hub vertex data eliminates false sharing in Degree Sorting since each thread updates its local copy of the hub vertex data. We evaluate this mutually-enabling combination of power-law-specific data duplication and Degree Sorting in a system called RADAR. RADAR improves performance by eliminating atomic updates for hub vertices and improving the cache locality of graph applications, providing speedups of up to 165x (1.88x on average) across different graph applications and input graphs.

6. An investigation of big graph partitioning methods for distribution of graphs in vertex-centric systems

Relations among data entities in most big data sets can be modeled by a big graph. Implementation and execution of algorithms related to the structure of big graphs is very important in different fields. Because of the inherently high volume of big graphs, their calculations should be performed in a distributed manner. Some distributed systems based on vertex-centric model have been introduced for big graph calculations in recent years. The performance of these systems in terms of run time depends on the partitioning and distribution of the graph. Therefore, the graph partitioning is a major concern in this field. This paper concentrates on big graph partitioning approaches for distribution of graphs in vertex-centric systems. This briefly discusses vertex-centric systems and formulates different models of graph partitioning problem. Then, a review of recent methods of big graph partitioning for these systems is shown. Most recent methods of big graph partitioning for vertex centric systems can be categorized into three classes: (i) stream-based methods that see vertices or edges of the graph in a stream and partition them, (ii) distributed methods that partition vertices or edges in a distributed manner, and (iii) dynamic methods that change partitions during the execution of algorithms to obtain better performance. This study compares the properties of different approaches in each class and briefly reviews methods that are not in these categories. This comparison indicates that The streaming methods are good choices for initial load of the graph in Vertex-centric systems. The distributed and dynamic methods are appropriate for long-running applications.

7. Executable schema mappings for statistical data processing

Abstract Data processing is the core of any statistical information system. Statisticians are interested in specifying transformations and manipulations of data at a high level, in terms of entities of statistical models. We illustrate here a proposal where a high-level language, EXL, is used for the declarative specification of statistical programs, and a translation into executable form in various target systems is available. The language is based on the theory of schema mappings, in particular those defined by a specific class of tgds, which we actually use to optimize user programs and facilitate the translation towards several target systems. The characteristics of such class guarantee good tractability properties and the applicability in Big Data settings. A concrete implementation, EXLEngine, has been carried out and is currently used at the Bank of Italy.

8. Pebbles: Leveraging Sketches for Processing Voluminous, High Velocity Data Streams

Abstract—Voluminous, time-series data streams originating in continuous sensing environments pose data ingestion and processing challenges. We present a holistic methodology centered around data sketching to address both challenges. We introduce an order-preserving sketching algorithm that we have designed for space-efficient representation of multi-feature streams with native support for stream processing related operations. Observational streams are preprocessed at the edges of the network generating sketched streams to reduce data transfer costs and energy consumption. Ingested sketched streams are then processed using sketch-aware extensions to existing stream processing APIs delivering improved performance. Our benchmarks with real-world datasets show up to a   8 reduction in data volumes transferred and a   27 improvement in throughput.

9. Cost-Aware Partitioning for Efficient Large Graph Processing in Geo-Distributed Datacenters

Abstract—Graph processing is an emerging computation model for a wide range of applications and graph partitioning is important for optimizing the cost and performance of graph processing jobs. Recently, many graph applications store their data on geo-distributed datacenters (DCs) to provide services worldwide with low latency. This raises new challenges to existing graph partitioning methods, due to the multi-level heterogeneities in network bandwidth and communication prices in geo-distributed DCs. In this article, we propose an efficient graph partitioning method named Geo-Cut, which takes both the cost and performance objectives into consideration for large graph processing in geo-distributed DCs. Geo-Cut adopts two optimization stages. First, we propose a cost-aware streaming heuristic and utilize the one-pass streaming graph partitioning method to quickly assign edges to different DCs while minimizing inter-DC data communication cost. Second, we propose two partition refinement heuristics which identify the performance bottlenecks of geo-distributed graph processing and refine the partitioning result obtained in the first stage to reduce the inter-DC data transfer time while satisfying the budget constraint. Geo-Cut can be also applied to partition dynamic graphs thanks to its lightweight runtime overhead. We evaluate the effectiveness and efficiency of Geo-Cut using real-world graphs with both real geo-distributed DCs and simulations. Evaluation results show that Geo-Cut can reduce the inter-DC data transfer time by up to 79 percent (42 percent as the median) and reduce the monetary cost by up to 75 percent (26 percent as the median) compared to state-of-the-art graph partitioning methods with a low overhead.

10. Data Prefetching and Eviction Mechanisms of In-Memory Storage Systems Based on Scheduling for Big Data Processing

Abstract—In-memory techniques keep data into faster and more expensive storage media for improving performance of big data processing. However, existing mechanisms do not consider how to expedite the data processing applications that access the input datasets only once. Another problem is how to reclaim memory without affecting other running applications. In this paper, we provide scheduling-aware data prefetching and eviction mechanisms based on Spark, Alluxio, and Hadoop. The mechanisms prefetch data and release memory resources based on the scheduling information. A mathematical method is proposed for maximizing the reduction of data access time. To make the mechanisms applicable in large-scale environments, we propose a heuristic algorithm to reduce the computational time. Furthermore, an enhanced version of the heuristic algorithm is also proposed to increase the amount of prefetched data. Finally, we perform real-testbed and simulation experiments to show the effectiveness of the proposed mechanisms.

11. Multi-Level Elasticity for Data Stream Processing

Abstract—This paper investigates reactive elasticity in stream processing environments where the performance goal is to analyze large amounts of data with low latency and minimum resources. Working in the context of Apache Storm, we propose an elastic management strategy which modulates the parallelism degree of applications' components while explicitly addressing the hierarchy of execution containers (virtual machines, processes and threads). We show that provisioning the wrong kind of container may lead to performance degradation and propose a solution that provisions the least expensive container (with minimum resources) to increase performance. We describe our monitoring metrics and show how we take into account the specifics of an execution environment. We provide an experimental evaluation with real-world applications which validates the applicability of our approach.

12. Efficient Operator Placement for Distributed Data Stream Processing Applications

Abstract—In the last few years, a large number of real-time analytics applications rely on the Data Stream Processing (DSP) so to extract, in a timely manner, valuable information from distributed sources. Moreover, to efficiently handle the increasing amount of data, recent trends exploit the emerging presence of edge/Fog computing resources so to decentralize the execution of DSP applications. Since determining the Optimal DSP Placement (for short, ODP) is an NP-hard problem, we need efficient heuristics that can identify a good application placement on the computing infrastructure in a feasible amount of time, even for large problem instances. In this paper, we present several DSP placement heuristics that consider the heterogeneity of computing and network resources; we divide them in two main groups: model-based and model-free. The former employ different strategies for efficiently solving the ODP model. The latter implement, for the problem at hand, some of the well-known meta-heuristics, namely greedy first-fit, local search, and tabu search. By leveraging on ODP, we conduct a thorough experimental evaluation, aimed to assess the heuristics' efficiency and efficacy under different configurations of infrastructure size, application topology, and optimization objective

13. Maximum Data-Resolution Efficiency for Fog-Computing Supported Spatial Big Data Processing in Disaster Scenarios

Abstract—Spatial big data analysis is very important in disaster scenarios to understand distribution patterns of situations, e.g., people's movements, people's requirements, resource shortage situations, and so on. In a general case, spatial big data is generated from distributed sensing devices and analyzed in a centralized way, e.g., a cloud center with high-performance computing resources. However, data transmission from sensing devices to cloud centers always takes a long time, especially in disaster scenarios with an unstable network. Fog computing is a promising technique to solve the above problem by offloading data processing tasks from the cloud to nearby computation devices. But data resolution also decreases after local processing in the fog nodes. It is necessary to investigate the optimal task distribution solutions to efficiently use computation resources in the fog layer. In this paper, we take the above research problem, and study fog-computing supported spatial big data processing. We analyze the process for spatial clustering, which is a typical category for spatial data analysis, and propose an architecture to integrate data processing into fog computing. We formalize a problem to maximize the data-resolution efficiency by considering data resolution and delay. We further propose core algorithms to enable spatial clustering in a fog-computing environment and implement the above algorithms in a real system. We have performed both simulations and experiments on a real Twitter dataset collected when Kumamoto-city suffered an earthquake. Through the simulations and the experiments, we have determined that the proposed solution significantly outperforms the other solutions.

14. Pec: Proactive Elastic Collaborative Resource Scheduling in Data Stream Processing

Abstract—In the Distributed Parallel Stream Processing Systems (DPSPS), elastic resource allocation allows applications to dynamically response to workload fluctuations. However, resource provisioning can be particularly challenging, due to the unpredictability of the workload. In addition, unlike CPU resources, bandwidth resources are often ignored in resource allocation. Moreover, resource allocation and resource placement are considered separately. In this paper, we investigate the proactive elastic resource scheduling problem for computation-intensive and communication-intensive applications, which aims at meeting the latency requirement with the minimal energy cost, and propose a dynamic collaborative strategy from the systemic perspective. Specifically, we first model a collaborative workload prediction pattern to accurately predict the upcoming workload, and construct a latency estimation model to estimate the latency of the application. Then, we design an energy-efficient resource pre-allocation method, in which the CPU frequency adjustment and the stability of resource reconfigurations are both considered. Finally, we present a communication-aware resource placement approach. Simulation results show that, compared with the reactive strategies, our strategy achieves an obviously better latency performance, and effectively avoids unnecessary resource adjustments. Meanwhile, the energy consumption is about saved by 50 percent on average, and the communication cost is maintained at a very low level of 4 percent.

15. Learning-Based Memory Allocation Optimization for Delay-Sensitive Big Data Processing

Abstract—Optimal resource provisioning is essential for scalable big data analytics. However, it has been difficult to accurately forecast the resource requirements before the actual deployment of these applications as their resource requirements are heavily application and data dependent. This paper identifies the existence of effective memory resource requirements for most of the big data analytic applications running inside JVMs in distributed Spark environments. Provisioning memory less than the effective memory requirement may result in rapid deterioration of the application execution in terms of its total execution time. A machine learning-based prediction model is proposed in this paper to forecast the effective memory requirement of an application given its service level agreement. This model captures the memory consumption behavior of big data applications and the dynamics of memory utilization in a distributed cluster environment. With an accurate prediction of the effective memory requirement, it is shown that up to 60 percent savings of the memory resource is feasible if an execution time penalty of 10 percent is acceptable. The accuracy of the model is evaluated on a physical Spark cluster with 128 cores and 1TB of total memory. The experiment results show that the proposed solution can predict the minimum required memory size for given acceptable delays with high accuracy, even if the behavior of target applications is unknown during the training of the model.

五、 数据中心和资源调度

1. Generalized Cost-Based Job Scheduling in Very Large Heterogeneous Cluster Systems

Abstract—We study job assignment in large, heterogeneous resource-sharing clusters of servers with finite buffers. This load balancing problem arises naturally in today’s communication and big data systems, such as Amazon Web Services, Network Service Function Chains, and Stream Processing. Arriving jobs are dispatched to a server, following a load balancing policy that optimizes a performance criterion such as job completion time. Our contribution is a randomized Cost-Based Scheduling (CBS) policy in which the job assignment is driven by general cost functions of the server queue lengths. Beyond existing schemes, such as the Join the Shortest Queue (JSQ), the power of d or the SQ(d) and the capacity-weighted JSQ, the notion of CBS yields new application-specific policies such as hybrid locally uniform JSQ. As today’s data center

clusters have thousands of servers, exact analysis of CBS policies is tedious. In this article, we derive a scaling limit when the number of servers grows large, facilitating a comparison of various CBS policies with respect to their transient as well as steady state behavior. A byproduct of our derivations is the relationship between the queue filling proportions and the server buffer sizes, which cannot be obtained from infinite buffer models. Finally, we provide extensive numerical evaluations and discuss several applications including multi-stage systems.

2. A Value-Oriented Job Scheduling Approach for Power-Constrained and Oversubscribed HPC Systems

Abstract—In this article, we investigate limitations in the traditional value-based algorithms for a power-constrained HPC system and evaluate their impact on HPC productivity. We expose the trade-off between allocating system-wide power budget uniformly and greedily under different system-wide power constraints in an oversubscribed system. We experimentally demonstrate that, under the tightest power constraint, the mean productivity of the greedy allocation is 38 percent higher than the uniform allocation whereas, under the intermediate power constraint, the uniform allocation has a mean productivity of 6 percent higher than the greedy allocation. We then propose a new algorithm that adapts its behavior to deliver the combined benefits of the two allocation strategies. We design a methodology with online retraining capability to create application-specific power-execution time models for a class of HPC applications. These models are used in predicting the execution time of an application on the available resources at the time of making scheduling decisions in the power-aware algorithms. We evaluate the proposed algorithm using emulation and simulation environments, and show that our adaptive strategy results in improving HPC resource utilization while delivering a mean productivity that is almost the same as the best performing algorithm across various system-wide power constraints.

3. QoS-Driven Coordinated Management of Resources to Save Energy in Multi-Core Systems

Abstract—Applications that are run on multicore systems without performance targets can waste significant energy. This paper considers, for the first time, a QoS-driven coordinated resource management algorithm (RMA) that dynamically adjusts the size of the per-core last-level cache partitions and the per-core voltage-frequency settings to save energy while respecting QoS requirements of individual applications in multiprogrammed workloads run on multi-core systems. It does so by doing configuration-space exploration across the spectrum of LLC partition sizes and DVFS settings at runtime at negligible overhead. Compared to DVFS and cache partitioning alone, we show that our proposed coordinated RMA is capable of saving, on average, 20% energy as compared to 15% for DVFS alone and 7% for cache partitioning alone, when the performance target is set to 70% of the baseline system performance. Index Terms—QoS, Cache Partitioning, DVFS, Resource Management

4. Joint Resource Allocation and Load Management for Cooling-Aware Mobile-Edge Computing

Abstract—In this paper, we jointly design resource allocation and load management in a mobile-edge computing (MEC) system with wireless power transfer (WPT), to minimize the total energy consumption of the BS, while meeting computation latency requirements. For the first time, the cooling energy, which is nonnegligible, is considered to minimize the energy consumption of the MEC system. By orchestrating the alternative optimization technique, Lagrange duality method and subgradient method, we decompose the original optimization problem and obtain the optimal solution in a semi-closed form. Extensive numerical tests corroborate the merits of the proposed algorithm over existing benchmarks in terms of energy saving. Index Terms—Mobile edge computing (MEC), wireless power transfer (WPT), computation offloading, cooling energy.

5. Towards Efficient Scheduling of Federated Mobile Devices Under Computational and Statistical Heterogeneity

Abstract—Originated from distributed learning, federated learning enables privacy-preserved collaboration on a new abstracted level by sharing the model parameters only. While the current research mainly focuses on optimizing learning algorithms and minimizing communication overhead left by distributed learning, there is still a considerable gap when it comes to the real implementation on mobile devices. In this article, we start with an empirical experiment to demonstrate computation heterogeneity is a more pronounced bottleneck than communication on the current generation of battery-powered mobile devices, and the existing methods are haunted by mobile stragglers. Further, non-identically distributed data across the mobile users makes the selection of participants critical to the accuracy and convergence. To tackle the computational and statistical heterogeneity, we utilize data as a tuning knob and propose two efficient polynomial-time algorithms to schedule different workloads on various mobile devices, when data is identically or non-identically distributed. For identically distributed data, we combine partitioning and linear bottleneck assignment to achieve near-optimal training time without accuracy loss. For non-identically distributed data, we convert it into an average cost minimization problem and propose a greedy algorithm to find a reasonable balance between computation time and accuracy. We also establish an offline profiler to quantify the runtime behavior of different devices, which serves as the input to the scheduling algorithms. We conduct extensive experiments on a mobile testbed with two datasets and up to 20 devices. Compared with the common benchmarks, the proposed algorithms achieve 2-100 speedup epoch-wise, 2–7 percent accuracy gain and boost the convergence rate by more than 100 percent on CIFAR10.

6. Network-Aware Locality Scheduling for Distributed Data Operators in Data Centers

Abstract—Large data centers are currently the mainstream infrastructures for big data processing. As one of the most fundamental tasks in these environments, the efficient execution of distributed data operators (e.g., join and aggregation) are still challenging current data systems, and one of the key performance issues is network communication time. State-of-the-art methods trying to improve that problem focus on either application-layer data locality optimization to reduce network traffic or on network-layer data flow optimization to increase bandwidth utilization. However, the techniques in the two layers are totally independent from each other, and performance gains from a joint optimization perspective have not yet been explored. In this article, we propose a novel approach called NEAL (NEtwork-Aware Locality scheduling) to bridge this gap, and consequently to further reduce communication time for distributed big data operators. We present the detailed design and implementation of NEAL, and our experimental results demonstrate that NEAL always performs better than current approaches for different workloads and network bandwidth configurations.

7. Burst Load Evacuation Based on Dispatching and Scheduling In Distributed Edge Networks

Abstract—Edge computing, a fast evolving computing paradigm, has spawned a variety of new system architectures and computing methods discussed in both academia and industry. Edge servers are directly deployed near users' equipment or devices owned by telecommunications companies. This allows for offloading computing tasks of various devices nearby to edge servers. Due to the shortage of computing resources in edge computing networks, they are often not as sufficient as the computing resources in a cloud computing center. This leads to the problem of service load imbalance once the load in the edge computing network increases suddenly. To solve the problem of "load evacuation" in edge environments, we introduce a strategy when the number of service requests for mobile devices or IoT devices increases rapidly within a short period of time. Therefore, to prevent poor QoS in edge computing, service load should be migrated to other edge servers to reduce the overall delay of these service requests. In this article, we have introduced a strategy with two stages during the burst load evacuation. Based on an optimal routing search at the dispatching stage, tasks will be migrated from the server in which the burst load occurs to other servers as soon as possible. Subsequently, with the assistance of the remote server and edge servers, these tasks are processed with the highest efficiency through the proposed parallel structure at the scheduling stage. Finally, we conduct numerical experiments to clarify the superiority of our algorithm in an edge environment simulation.

8. Energy-Efficient Parallel Real-Time Scheduling on Clustered Multi-Core

Abstract—Energy-efficiency is a critical requirement for computation-intensive real-time applications on multi-core embedded systems. Multi-core processors enable intra-task parallelism, and in this work, we study energy-efficient real-time scheduling of constrained deadline sporadic parallel tasks, where each task is represented as a directed acyclic graph (DAG). We consider a clustered multi-core platform where processors within the same cluster run at the same speed at any given time. A new concept named speed-profile is proposed to model per-task and per-cluster energy-consumption variations during run-time to minimize the expected long-term energy consumption. To our knowledge, no existing work considers energy-aware real-time scheduling of DAG tasks with constrained deadlines, nor on a clustered multi-core platform. The proposed energy-aware real-time scheduler is implemented upon an ODROID XU-3 board to evaluate and demonstrate its feasibility and practicality. To complement our system experiments in large-scale, we have also conducted simulations that demonstrate a CPU energy saving of up to 67 percent through our proposed approach compared to existing methods.

9. Scheduling Parallel Real-Time Tasks on the Minimum Number of Processors

Abstract—Recently, several parallel frameworks have emerged to utilize the increasing computational capacity of multiprocessors. Parallel tasks are distinguished from traditional sequential tasks in that the subtasks contained in a single parallel task can simultaneously execute on multiple processors. In this study, we consider the scheduling problem of minimizing the number of processors on which the parallel real-time tasks feasibly run. In particular, we focus on scheduling sporadic parallel real-time tasks, in which precedence constraints between subtasks of each parallel task are expressed using a directed acyclic graph (DAG). To address the problem, we formulate an optimization problem that aims to minimize the maximum processing capacity for executing the given tasks. We then suggest a polynomial solution consisting of three steps: (1) transform each parallel real-time task into a series of multithreaded segments, while respecting the precedence constraints of the DAG; (2) selectively extend the segment lengths; and (3) interpret the problem as a flow network to balance the flows on the terminal edges. We also provide the schedulability bound of the proposed solution: it has a capacity augmentation bound of 2. Our experimental results show that the proposed approach yields higher performance than one developed in a recent study

10. Towards Plan-aware Resource Allocation in Serverless Query Processing

Resource allocation for serverless query processing is a challenge. Unfortunately, prior approaches have treated queries as black boxes, thereby missing significant resource optimization opportunities. In this paper, we propose a plan-aware resource allocation approach where the resources are adaptively allocated based on the runtime characteristics of the query plan. We show the savings opportunity from such an allocation scheme over production SCOPE workloads at Microsoft. We present our current implementation of a greedy version that periodically estimates the peak resource for the remaining of the query as the query execution progresses. Our experimental evaluation shows that such an implementation could already save more than

8% resource usage over one of our production virtual clusters. We conclude by opening the discussion on various strategies for plan-aware resource allocation and their implications on the cloud computing stack.

11. AI4DL: Mining Behaviors of Deep Learning Workloads for Resource Management

The more we know about the resource usage patterns of workloads, the better we can allocate resources. Here we present a methodology to discover resource usage behaviors of containers training Deep Learning (DL) models. From monitoring, we can observe repeating patterns and similitude of resource usage among containers training different DL models. The repeating patterns observed can be leveraged by the scheduler or the resource autoscaler to reduce resource fragmentation and overall resource utilization in a dedicated DL cluster. Specifically, our approach combines Conditional Restricted Boltzmann Machines (CRBMs) and clustering techniques to discover common sequences of behaviors (phases) of containers running the DL training workloads in clusters providing IBM Deep Learning Services. By studying the resource usage pattern at each phase and the typical sequences of phases among different containers, we discover a reduced set of prototypical executions representing the majority of executions. We use statistical information from each phase to refine resource provisioning by dynamically tuning the amount of resource each container requires at each phase. Evaluation of our method shows that by leveraging typical resource usage patterns, we can auto-scale containers to reduce CPU and Memory allocation by 30% compared to statistics based reactive policies, which is close to having a-priori knowledge of resource usage while fulfilling resource demand over 95% of the time.

12. Resource Efficient Stream Processing Platform with Latency-Aware Scheduling Algorithms

We presented a novel platform dedicated to stream processing that improved resource efficiency by sharing resources among applications. The platform utilized latency-aware schedulers to handle stream applications with heterogeneous SLAs and workloads. We implemented the prototype in Spark Structured Streaming and evaluated the platform with pseudo IoT services. The result showed that our platform outperformed default Spark Structured Streaming while reducing the necessary CPU cores by 36%. We further compared the adaptability of the schedulers and found that one of the schedulers reduced the SLA violations by 90% compared to the default FAIR when the platform was overloaded.

13. Stratus: Clouds with Microarchitectural Resource Management

The emerging next generation of cloud services like Granular and Serverless computing are pushing the boundaries of the current cloud infrastructure. In order to meet the performance objectives, researchers are now leveraging low-level microarchitectural resources in clouds. At the same time these resources are also a major source of security problems that can compromise the confidentiality and integrity of sensitive data in multi-tenant shared cloud infrastructures. The core of the problem is the lack of isolation due to the unsupervised sharing of microarchitectural resources across different performance and security boundaries. In this paper, we introduce Stratus clouds that treat the isolation on microarchitectural elements as the key design principle when allocating cloud resources. This isolation improves both performance and security, but at the cost of reducing resource utilization. Stratus captures this trade-off using a novel abstraction that we call isolation credit, and show how it can help both providers and tenants when allocating microarchitectural resources using Stratus's declarative interface. We conclude by discussing the challenges of realizing Stratus clouds today.

14. Securing RDMA for High-Performance Datacenter Storage Systems

RDMA is increasingly popular for low-latency communication in datacenters, marking a major change in how we build distributed systems. Unfortunately, as we pursue significant system re-designs inspired by new technology, we have not given equal thought to the consequences for system security. This paper investigates security issues introduced to datacenter systems by switching to RDMA and challenges in building secure RDMA systems. These challenges include changes in RPC reliability guarantees and unauditable data-accesses. We show how RDMA's design makes it challenging to build secure storage systems by analyzing recent research systems; then we outline several directions for solutions and future research, with the goal of securing RDMA datacenter systems while they are still in the research and prototype stages.

六、 云端融合与桌面虚拟化

1. Firework: Data Processing and Sharing for Hybrid Cloud-Edge Analytics

Abstract—Now we are entering the era of the Internet of Everything (IoE) and billions of sensors and actuators are connected to the network. As one of the most sophisticated IoE applications, real-time video analytics is promising to significantly improve public safety, business intelligence, and healthcare & life science, among others. However, cloud-centric video analytics requires that all video data must be preloaded to a centralized cluster or the cloud, which suffers from high response latency and high cost of data transmission, given the scale of zettabytes of video data generated by IoE devices. Moreover, video data is rarely shared among multiple stakeholders due to various concerns, which restricts the practical deployment of video analytics that takes advantages of many data sources to make smart decisions. Furthermore, there is no efficient programming interface for developers and users to easily program and deploy IoE applications across geographically distributed computation resources. In this paper, we present a new computing framework, Firework, which facilitates distributed data processing and sharing for IoE applications via a virtual shared data view and service composition. We designed an easy-to-use programming interface for Firework to allow developers to program on Firework. This paper describes the system design, implementation, and programming interface of Firework. The experimental results of a video analytics application demonstrate that Firework reduces up to 19.52 percent of response latency and at least 72.77 percent of network bandwidth cost, compared to a cloud-centric solution.

2. Edge Computing – the Case for Heterogeneous-ISA Container Migration

Edge computing is a recent computing paradigm that brings cloud services closer to the client. Among other features, edge computing offers extremely low client/server latencies. To consistently provide such low latencies, services need to run on edge nodes that are physically as close as possible to their clients. Thus, when a client changes its physical location, a service should migrate between edge nodes to maintain proximity. Differently from cloud nodes, edge nodes are built with CPUs of differentInstruction Set Architectures(ISAs), hence a server program natively compiled for one ISA cannot migrate to another. This hinders migration to the closest node. We introduce H-Container, which migrates natively-compiled containerized applications across compute nodes featuring CPUs of different ISAs. H-Container advances over existing heterogeneous-ISA migration systems by being a) highly compatible – no source code nor compiler toolchain modifications are needed; b) easily deployable – fully implemented in user space, thus without any OS or hypervisor dependency, and c) largely Linux compliant – can migrate most Linux software, including server applications and dynamically linked binaries. H-Container targets Linux, adopts LLVM, extends CRIU, and integrates with Docker. Experiments demonstrate that H-Container adds no overhead on average during program execution, while between 10ms and 100ms are added during migration. Furthermore, we show the benefits of HContainer in real scenarios, proving for example up to 94% increase in Redis throughput when unlocking heterogeneity.

3. A Double-Edged Sword: Security Threats and Opportunities in One-Sided Network Communication

One-sided network communication technologies such as RDMA and NVMe-over-Fabrics are quickly gaining adoption in production software and in datacenters. Although appealing for their low CPU utilization and good performance, they raise new security concerns that could seriously undermine datacenter software systems building on top of them. At the same time, they offer unique opportunities to help enhance security. Indeed, one-sided network communication is a doubleedged sword in security. This paper presents our insights into security implications and opportunities of one-sided communication.

4. Locally-Centralized Certificate Validation and Its Application in Desktop Virtualization Systems

Abstract— To validate a certificate, a user needs to install the certificate of the root certification authority (CA) and download the certificate revocation information (CRI). Although operating systems and browsers manage the certificate trust list (CTL) of publicly-trusted root CAs for global users, locally-trusted root CAs still play an important role and it is difficult for a user to manage its CTL properly by itself. Meanwhile, the CRI access is inefficient, sometimes even unavailable, and causes privacy leakage. We revisit these problems by analyzing the TLS sessions within an organization. To the best of our knowledge, we are the first to analyze CTL management and CRI access on the scale of medium-sized organizations. Based on the analysis, a locallycentralized design is proposed to manage the CTLs of all users by IT administrators and access the CRI services for all users, within an organization. We apply this design to desktop virtualization systems to demonstrate its applicability, and build vCertGuard with oVirt and KVM-QEMU. In vCertGuard, the CTLs of all virtual machines (VMs) are managed in the VM monitors (VMMs). In the CTL, the self-signed certificates of publicly-trusted root CAs are properly configured, while each locally-trusted certificate chain is specified one by one. vCertGuard accesses the CRI services for all VMs, and the downloaded CRI is cached and shared among VMs. Because most TLS servers are visited by multiple users of an organization, it reduces the cost of CRI access. Experimental results of the prototype system show that vCertGuard maintains the CTLs with a negligible overhead, and significantly improves the performance of CRI access

5. EdgeSlice: Slicing Wireless Edge Computing Network with Decentralized Deep Reinforcement Learning

Abstract—5G and edge computing will serve various emerging use cases that have diverse requirements of multiple resources, e.g., radio, transportation, and computing. Network slicing is a promising technology for creating virtual networks that can be customized according to the requirements of different use cases. Provisioning network slices requires end-to-end resource orchestration which is challenging. In this paper, we design a decentralized resource orchestration system named EdgeSlice for dynamic end-to-end network slicing. EdgeSlice introduces a new decentralized deep reinforcement learning (D-DRL) method to efficiently orchestrate end-to-end resources. D-DRL is composed of a performance coordinator and multiple orchestration agents. The performance coordinator manages the resource orchestration policies in all the orchestration agents to ensure the service level agreement (SLA) of network slices. The orchestration agent learns the resource demands of network slices and orchestrates the resource allocation accordingly to optimize the performance of the slices under the constrained networking and computing resources. We design radio, transport and computing manager to enable dynamic configuration of end-to-end resources at runtime. We implement EdgeSlice on a prototype of the end-to-end wireless edge computing network with OpenAirInterface LTE network, OpenDayLight SDN switches, and CUDA GPU platform. The performance of EdgeSlice is evaluated through both experiments and trace-driven simulations. The evaluation results show that EdgeSlice achieves much improvement as compared to baseline in terms of performance, scalability, compatibility. Index Terms—Resource Orchestration, Deep Reinforcement Learning, Network Slicing, Wireless Edge Computing

6. Quality of Experience-Aware User Allocation in Edge Computing Systems: A Potential Game

Abstract—As many applications and services are moving towards a more human-centered design, app vendors are taking the quality of experience (QoE) increasingly seriously. End-to-end latency is a key factor that determines the QoE experienced by users, especially for latency-sensitive applications such as online gaming, health care, critical warning systems and so on. Recently, edge computing has emerged as a promising solution to the high latency problem. In an edge computing environment, edge servers are deployed at cellular base stations, offering processing power and low network latency to users within their geographic proximity. In this paper, we tackle the user allocation problem in edge computing from an app vendor's perspective, where the vendor needs to decide which edge servers to serve which users in a specific area. Also, the vendor must consider the various levels of quality of service (QoS) for its users. Each QoS level results in a different QoE level; thus, the app vendor needs to decide the QoS level for each user so that the overall user experience is maximized. To tackle the NP-hardness of this problem, we formulate it as a potential game then propose QoEGame, an effective and efficient game-theoretic approach that admits a Nash equilibrium as a solution to the user allocation problem. Being a distributed algorithm, QoEGame is able to fully utilize the distributed nature of edge computing. Finally, we theoretically and empirically evaluate the performance of QoEGame, which is illustrated to be significantly better than the state of the art and other baseline approaches.

7. Contextual-Bandit Anomaly Detection for IoT Data in Distributed Hierarchical Edge Computing

Abstract—Advances in deep neural networks (DNN) greatly bolster real-time detection of anomalous IoT data. However, IoT devices can hardly afford complex DNN models, and offloading anomaly detection tasks to the cloud incurs long delay. In this paper, we propose and build a demo for an adaptive anomaly detection approach for distributed hierarchical edge computing (HEC) systems to solve this problem, for both univariate and multivariate IoT data. First, we construct multiple anomaly detection DNN models with increasing complexity, and associate each model with a layer in HEC from bottom to top. Then, we design an adaptive scheme to select one of these models on the fly, based on the contextual information extracted from each input data. The model selection is formulated as a contextual bandit problem characterized by a single-step Markov decision process, and is solved using a reinforcement learning policy network. We build an HEC testbed, implement our proposed approach, and evaluate it using real IoT datasets. The demo shows that our proposed approach significantly reduces detection delay (e.g., by 71.4% for univariate dataset) without sacrificing accuracy, as compared to offloading detection tasks to the cloud. We also compare it with other baseline schemes and demonstrate that it achieves the best accuracy-delay tradeoff. 1

8. Context-Aware Deep Model Compression for Edge Cloud Computing

Abstract—While deep neural networks (DNNs) have led to a paradigm shift, its exorbitant computational requirement has always been a roadblock in its deployment to the edge, such as wearable devices and smartphones. Hence a hybrid edge-cloud computational framework is proposed to transfer part of the computation to the cloud, by naively partitioning the DNN operations under the constant network condition assumption. However, realworld network state varies greatly depending on the context, and DNN partitioning only has limited strategy space. In this paper, we explore the structural flexibility of DNN to fit the edge model to varying network contexts and different deployment platforms. Specifically, we designed a reinforcement learning-based decision engine to search for model transformation strategies in response to a combined objective of model accuracy and computation latency. The engine generates a context-aware model tree so that the DNN can decide the model branch to switch to at runtime. By the emulation and field experimental results, our approach enjoys a 30% − 50% latency reduction while retaining the model accuracy.

9. SNAP: A Communication Efficient Distributed Machine Learning Framework for Edge Computing

Abstract—More and more applications learn from the data collected by the edge devices. Conventional learning methods, such as gathering all the raw data to train an ultimate model in a centralized way, or training a target model in a distributed manner under the parameter server framework, suffer a high communication cost. In this paper, we design Select Neighbors and Parameters (SNAP), a communication efficient distributed machine learning framework, to mitigate the communication cost. A distinct feature of SNAP is that the edge servers act as peers to each other. Specifically, in SNAP, every edge server hosts a copy of the global model, trains it with the local data, and periodically updates the local parameters based on the weighted sum of the parameters from its neighbors (i.e., peers) only (i.e., without pulling the parameters from all other edge servers). Different from most of the previous works on consensus optimization in which the weight matrix to update parameter values is predefined, we propose a scheme to optimize the weight matrix based on the network topology, and hence the convergence rate can be improved. Another key idea in SNAP is that only the parameters which have been changed significantly since the last iteration will be sent to the neighbors. Both theoretical analysis and simulations show that SNAP can achieve the same accuracy performance as the centralized training method. Compared to the state-of-the-art communication-aware distributed learning scheme TernGrad, SNAP incurs a significantly lower (99.6% lower) communication cost.

10. Optimal Task Allocation and Coding Design for Secure Coded Edge Computing

Abstract—In recent years, edge computing has attracted increasing attention for its capability of facilitating delay-sensitive applications. In the implementation of edge computing, however, data confidentiality has been raised as a major concern because edge devices may be untrustable. In this paper, we propose a design of secure and efficient edge computing by linear coding. In general, linear coding can achieve data confidentiality by adding random information to the original data before they are distributed to edge devices. To this end, it is important to carefully design code such that the user can successfully decode the final result while achieving security requirements. Meanwhile, task allocation, which selects a set of edge devices to participate in a computation task, affects not only the total resource consumption, including computation, storage, and communication, but also coding design. In this paper, we study task allocation and coding design, two highly-coupled problems in secure coded edge computing, in a unified framework. In particular, we take matrix multiplication, a fundamental building block of many distributed machine learning algorithms, as the representative computation task, and study optimal task allocation and coding design to minimize resource consumption while achieving informationtheoretic security.

11. Cognitive Service in Mobile Edge Computing

Abstract—Cognitive services have revolutionized the way we live, work and interact with the world. In recent years, deep neural networks have become the mainstream approach in cognitive service, and mobile edge computing facilitates a variety of cognitive services for users by offloading computation tasks from resource-limited mobile devices to relatively wealthy edge servers. Combining the two to provide users with a higher quality of cognitive service is an issue worth researching. However, many related studies are not easy to provide fast responses because in these systems, edge servers are only used to pre-process data, and the cloud server is used to perform tasks. In this paper, we aim to study deploying deep neural network models on edge servers to provide fast services. However, a single edge server collects only a small amount of data, which results in low inference accuracy. To address this problem, we propose a cloud and edge collaboration framework. The key idea of the proposed framework is to use a cloud model to assist in training an edge model to improve the latter's inference accuracy and enable the latter to provide fast response and high-performance cognitive service. Experimental results demonstrate the effectiveness of our proposed framework.

12. Security-Aware QoS Forecasting in Mobile Edge Computing based on Federated Learning

Abstract—This paper proposes a novel security-aware QoS (Quality of Service) forecasting approach – Edge QoS PerPM (Edge QoS forecasting with Personalized training based on Public Models in mobile edge computing) by migrating the principle of integrating cooperative learning and independent learning from federated learning. Edge QoS Per-PM can make fast and accurate forecasting on the premise of ensuring enhanced security. We train private model based on public model for personalized forecasting. The private models are invisible to other users to ensure the absolute security. At regular intervals, a Long Short-Term Memory (LSTM) model is trained based on the latest private data to meet the realtime requirements of the dynamic edge environment and ensure the accuracy of prediction results. A series of experiments is conducted based on public network data sets. The results demonstrate that Edge QoS Per-PM can train appropriate models and achieve faster convergence and higher accuracy

13. Dynamic Task Offloading with Minority Game for Internet of Vehicles in Cloud-Edge Computing

Abstract—With the advent of the Internet of Vehicles (IoV), drivers are now provided with diverse time-sensitive vehicular services that usually require a large scale of computation. As civilian vehicles are generally insufficient in computational resources, their service requests are offloaded to cloud data centers and edge computing devices (ECDs) with ample computational resources to enhance the quality of service (QoS). However, ECDs are often overloaded with excessive service requests. In addition, as the network conditions and service compositions are complicated and dynamic, the centralized control of ECDs is hard to achieve. To tackle these challenges, a dynamic task offloading method with minority game (MG) in cloud-edge computing, named DOM, is proposed in this paper. Technically, MG is an effective tool with a distributed mechanism which can minimize the dependency on centralized control in resource allocation. In the MG, reinforcement learning (RL) is applied to optimize the distributed decisionmaking of participants. Finally, with a real-world dataset of IoV services, the effectiveness and adaptability of DOM are evaluated.