Workshop on Hot Topics in Cloud Computing (HotCloud '09)

San Diego, CA
June 15, 2009

Summarized by Alva Couch (couch@eecs.tufts.edu) and Kiran-Kumar Muniswamy-Reddy (kiran@eecs.harvard.edu)

Cloud computing remains a "cloudy concept" for many people. The first USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '09) brought together academic and industry researchers to discuss late-breaking results and current trends in cloud computing. As in other "hot topics" conferences, HotCloud papers defined a problem and discussed a possible solution and preliminary results. Results ranged from performance of specific management strategies to designs for new components of cloud infrastructures. Full papers discussed upcoming research plans in detail, while short papers described an interesting idea worthy of further study. HotCloud '09 included 13 full papers and eight short papers, resulting in a day packed with new ideas and future challenges.

The workshop discussed several distinct kinds of clouds that are distinguished by the kinds of services that they provide to clients:

- Software as a Service (SaaS): clients gain access to specific software functions (e.g., gmail, Google Maps).
- Platform as a Service (PaaS): clients gain access to individual virtual machines: (e.g., Amazon Web Services, Eucalyptus).
- Infrastructure as a Service (IaaS): clients gain access to networks of (perhaps physical) machines (e.g., virtual data centers).

The kind of cloud determines the boundaries between a client's responsibility and the cloud provider's responsibility. In SaaS the client uses the application as an exterior entity. In PaaS the client must load an operating system instance into a virtual machine, while in IaaS the client might have to choose, deploy, and manage provisioning software that in PaaS is part of the service.

Clouds and cloud applications can exhibit (or lack) *elasticity*, the ability to dynamically adapt to changing use patterns by provisioning and decommissioning resources and virtual

instances of servers. In SaaS elasticity is completely invisible to the client; in PaaS the client must enable elasticity by providing images of virtual instances suitable for replication; in IaaS the client may be responsible for ensuring elasticity by choosing, deploying, and managing an elasticity application.

One motivation for "pushing an application into a cloud" is to reallocate responsibilities and risks from client to provider. Clouds can be characterized by the kinds of risks the provider assumes:

- **Compute clouds** provide computational power on demand. The provider assumes responsibility for availability and reliability of compute servers.
- **Data clouds** provide data persistence and preservation, where data can include file systems or databases. The provider assumes responsibility for data availability, integrity, and persistence.
- **Service clouds** provide and ensure function of a specific service. The provider assumes all responsibility for providing the service.

Of course, many cloud infrastructures provide all of the above.

Ensuring security and privacy for cloud data is more difficult than ensuring security and privacy in non-cloud infrastructure. Several security and privacy threats repeatedly arose at HotCloud, including:

- **Malicious use of privilege:** The maintainers of the cloud have administrative privilege and thus clandestine access to client data that they do not own.
- **Exploitation of co-location:** Malicious client applications can discover confidential information about other clients whose cloud functions happen to be co-located on the same physical devices, by employing back-channels, including shared use of memory, I/O, cache, or even address translation buffer behavior.
- **Limits of legal protection:** The Stored Communications Act (SCA) provides less legal protection against subpoena for cloud data than for data stored on self-owned hardware.

Thus, cloud clients may assume implicitly that providers are mitigating risks that may be beyond the providers' capabilities to mitigate. Many presenters assumed that all data in a cloud is public, sidestepping these difficulties, while others specifically considered the difficulties of keeping data private.

Finally, there was much discussion and controversy over eventual versus strong consistency in data clouds. In distributed database theory, a database exhibits *strong consistency* if changes to the data store are reflected immediately in subsequent queries, and *eventual consistency* if it is possible that changes will not be reflected in queries until a later time. Data clouds can likewise exhibit either strong or eventual consistency. While financial transactions such as purchases usually require strong consistency (so that the customer sees a purchase record immediately after a purchase), eventual consistency is usually acceptable for the results of a crawler or Web search. But this is a controversial issue: eventual consistency is "fun for computer scientists," but difficult to handle in practice, and leads to bugs in applications.

CLOUD PLATFORMS AND ARCHITECTURES

Full Papers

- ***Open Cirrus™ Cloud Computing Testbed: Federated Data Centers for Open Source Systems and Services Research***
  Roy Campbell, Indranil Gupta, Michael Heath, and Steven Y. Ko, University of Illinois at Urbana-Champaign; Michael Kozuch, Intel Research; Marcel Kunze, KIT, Germany; Thomas Kwan, Yahoo!; Kevin Lai, HP Labs; Hing Yan Lee, IDA, Singapore; Martha Lyons and Dejan Milojicic, HP Labs; David O'Hallaron, Intel Research; Yeng Chai Soh, IDA, Singapore

Open Cirrus is a cloud computing testbed with 11,000 cores, global services, and an open source stack, with nine sites and a planned size of 20 sites. Objectives of Open Cirrus include providing a vendor-neutral testbed for cloud technologies, collecting realistic traces of workload, and exposing the research community to realistic enterprise requirements. Infrastructure for Open Cirrus includes Tashi-provisioning software from Intel, as well as Hadoop for programming. Open Cirrus is intended to serve as a testbed for metrics of success for cloud computing and thus to inform the decision of whether to lease or own cloud infrastructure. As Open Cirrus is an international infrastructure, challenges include issues of privacy and legality. Users of Open Cirrus must develop separate service agreements with each of the nine international sites. Data privacy is difficult to guarantee when private data is hosted at foreign sites.

- ***Nebulas: Using Distributed Voluntary Resources to Build Clouds***
  Abhishek Chandra and Jon Weissman, University of Minnesota

Nebulas are a form of cloud computing based on voluntary cooperation and inspired by the success of edge-node computing infrastructures such as SETI@home. Voluntary, loosely coupled clouds based on an edge-node computing model seem to have several advantages, including an estimated two orders of magnitude cost difference between SETI@home and Amazon Elastic Compute Cloud (EC2). Nebulas, unlike clouds, implement elasticity through use of excess resources on volunteered distributed hosts. This leads to low cost, at the price of lower potential performance and higher volatility due to dynamic variation in resource availability. Challenges include coping with heterogeneity during deployment, fragility and churn, and data privacy. Threats to privacy arise both from privileged users on the volunteered hosts and from back-channels through co-location of Nebula services.

- ***Towards Trusted Cloud Computing***
  Nuno Santos, Krishna P. Gummadi, and Rodrigo Rodrigues, MPI-SWS

Trusted cloud computing refers to a situation in which data in the cloud—both computed and stored—remains private and protected from data leaks. One threat to data privacy is that cloud administrators have privileged access to virtual machine instances but do not own data contained in the instances. The cloud provider must be trusted to provide physical security and to limit physical access to cloud infrastructure. Software support for trust—which is effective only in the presence of physical security for cloud hardware—includes secure booting and remote attestation of state (i.e., some proof that privacy is being maintained). Challenges for trusted cloud computing include building trusted virtual machine monitors (VMMs) based on key infrastructure provided by trusted platform modules ("TPM chips"), and providing facilities for secure service migration without potential exposure of private data.

- ***The Case for Enterprise-Ready Virtual Private Clouds***
  Timothy Wood and Prashant Shenoy, University of Massachusetts Amherst; Alexandre Gerber, K.K. Ramakrishnan, and Jacobus Van der Merwe, AT&T Labs—Research

A virtual private cloud is an "Infrastructure as a Service" (IaaS) cloud mechanism whereby enterprises can augment in-house computing resources by renting remote computation and storage infrastructure transparently, securely, and flexibly. For IaaS to be practical, legacy applications have to be able to execute in the cloud without being specifically aware of where they are executing. Current cloud mechanisms for IaaS are difficult to secure if applications are not aware that they are running in a cloud, including firewall configuration. A virtual private cloud (VPC) establishes secure connections between owned and cloud infrastructure using dynamically configured layer-2 or layer-3 multi-protocol label-switching (MPLS) virtual private networks (VPNs). Advantages of VPCs include no requirement for end-node configuration and ability to transparently migrate existing applications to the cloud. Challenges for VPCs include the need for virtualized routing infrastructure and for a mechanism to make traditionally static VPN allocation dynamic, perhaps through Border Gateway Protocol (BGP) signaling. The audience expressed concern that the enterprise is giving the cloud provider's administrators privileged access to their owned infrastructure via VPC connections, thus increasing the risk of data leaks.

Short Papers

- ***Private Virtual Infrastructure for Cloud Computing***
  F. John Krautheim, University of Maryland, Baltimore County

One way to improve data privacy in a cloud is to utilize public-key cryptography to secure private information within virtual machine instances. A locator bot (lobot) is a virtual cloud appliance that stores an instance's private keys and manages the instance that utilizes those keys, thus allowing applications inside the instance to access encrypted resources. Lobots are created by a Private Virtual Infrastructure Factory (PVI factory). Challenges of creating lobots include how to measure and validate the security of the fabric in which the lobots execute, as well as protecting against object reuse during object shutdown. Private data leakage due to co-location of malicious clients might remain a problem due to persistence of in-memory copies of decrypted data.

- ***Refactoring Human Roles Solves Systems Problems***
  Jeremy Elson and Jon Howell, Microsoft Research

The success of cloud computing depends on decomposing the task of cloud deployment into human roles with clearly defined and minimal interfaces. In the same way that the software industry decoupled the user from the software developer, new roles in cloud implementation have the potential to decouple parts of the cloud implementation process with positive results. The "hardware wrangler" builds the hardware infrastructure for a cloud, while the "software integrator" chooses the software and versions to execute on that hardware. Inappropriate (or perhaps a better term is "over-specified") interaction between cloud client and integrator leads to "DLL hell" in which desired configurations are impossible to deploy, while inappropriate interaction between application developer and integrator can lead to vertical "stovepipe" architecture with minimal reuse. Challenges include limiting interactions between roles so that system administration of the result remains practical.

ELASTIC CLOUDS AND RESOURCE MANAGEMENT

Invited Short Presentation

- ***GENI and Cloud Computing***
  Harry Mussman, BBN Technologies

The Global Environment for Network Innovation (GENI) is an NSF project in support of experiments in network design. While GENI is not itself a cloud infrastructure, GENI encourages cloud researchers to build clouds on top of the GENI infrastructure, which is deeply programmable at a network level to support networking protocols other than the Internet. GENI is being developed by 29 teams, both academic and industrial, and an initial version will be available for initial experiments in 2009 and fully operational by 2010. GENI asks the cloud community to become involved by communicating specific needs for cloud research to the GENI developers.

Full Papers

- ***ElasTraS: An Elastic Transactional Data Store in the Cloud***
  Sudipto Das, Divyakant Agrawal, and Amr El Abbadi, University of California, Santa Barbara

The Elastic Transactional Data Store (ElasTraS) is a data storage mechanism that adds distributed transaction processing capability to a key/value data storage mechanism. Distributed transactional storage is implemented via a hier-

archy of transaction managers. "Owning transaction managers" own key/value mappings, while "high-level transaction managers" communicate with the "owning managers," serve as points of contact, and enhance performance through caching. Challenges include optimal (geographical) distribution of "owning" and "high-level" transaction managers.

- ***Reflective Control for an Elastic Cloud Application: An Automated Experiment Workbench***
  Azbayar Demberel, Jeff Chase, and Shivnath Babu, Duke University

A reflective elastic application is a cloud program that can manipulate its own resource requirements based on detailed knowledge of resource availability. Reflective applications can adapt to resource availability, e.g., by deferring computation until resources are more available, and opportunistically exploit excess resources, e.g., by completing deferred computations when resources are more available or cheaper. To understand the needs of reflective applications, the authors created an experimental workbench that can measure effects of various resource allocations on behavior and performance. The output of the workbench is a visualization of the "response surface" that depicts the relationship between input resources and resulting performance. Response surfaces can be efficiently calculated via sampling methods that interpolate response in areas where behavior seems to vary predictably. The audience questioned whether this approach is cost-effective, because of the relatively high cost of experimentation in a production environment.

- ***Toward Cloud-based Collaboration Services***
  David Banks, John S. Erickson, and Michael Rhodes, Hewlett-Packard Labs

Fractal is an open source cloud-based collaboration platform for public information. In Fractal, multiple "tenants" share a common cloud and contribute information that Fractal can coordinate. Fractal streamlines interaction between cloud information spaces through "extensions" that execute whenever data is modified and automatically relate data from different sources. Extensions can create cross-references between spaces, including citation, author, and location lookup, as well as automatic metadata extraction from documents. Extensions are customized for each tenant. While "privacy" is not considered, "content pollution" is a problem; tenants should not be able to alter the behavior of other tenants' content. Challenges include defining the appropriate notion of isolation for tenants, at the physical, virtual, and data levels.

Short Papers

- ***Colocation Games and Their Application to Distributed Resource Management***
  Jorge Londoño, Azer Bestavros, and Shang-Hua Teng, Boston University

The financial feasibility of renting cloud infrastructures can be improved if cloud clients collaborate (or perhaps collude) to share resources. In a market where providers provide fixed-size instances (in memory, storage, and computational speed), co-location by collaboration between cloud clients provides financial benefit. For example, two customers might realize that their applications will "both fit" inside the same virtual instance of some specific service provider. Such co-location can be modeled as a strategic game. The general case of this game has no guarantee of stability, but considering processes (applications) alone leads to a guaranteed (and stable) Nash equilibrium state in which no player can improve personal financial benefit by relocating. The authors propose that because this co-location game has a stable result, this kind of co-location should be supported by location services that help customers find partners, as well as infrastructure to enable migration.

- ***Virtual Putty: Reshaping the Physical Footprint of Virtual Machines***
  Jason Sonnek and Abhishek Chandra, University of Minnesota

Virtual putty refers to a scheme for optimizing the mapping of virtual applications to physical resources. Each physical machine is described in terms of resources and location. Likewise, each virtual instance has a footprint that includes its static resource needs, dynamic resource utilization patterns, and conflicts with other instances. By matching these footprints against one another, one can efficiently utilize physical resources and lower the cost of operations. Challenges include determining parts of the application footprint that are difficult to observe, e.g., dynamic resource utilization. Someone questioned whether the detail in the footprint actually does better than a simple greedy mapping algorithm and asked whether obtaining footprint data might be too expensive to be cost-effective.

- ***Statistical Machine Learning Makes Automatic Control Practical for Internet Datacenters***
  Peter Bodík, Rean Griffith, Charles Sutton, Armando Fox, Michael Jordan, and David Patterson, University of California, Berkeley

Current methods for resource allocation in clouds use simple performance models trained offline, or watermark methods such as increasing resources when a utilization watermark has been met (e.g., "when CPU utilization is greater than 70%, add a core"). It is possible to do better than these methods with a statistical model of behavior, learned dynamically via online experimentation. Based on measurements of end-to-end latency and its variance, a control policy simulator evaluates different policies and tunes model parameters to optimize a reward function. In the case of tuning feedback gain, the model tuning process is shown experimentally to closely approximate optimal behavior.

PANEL

- ***Future Challenges to Cloud Computing***
  Moderator: Amin Vahdat, University of California, San Diego

  Panelists: Garth Gibson, Panasas/Carnegie Mellon University; Stefan Savage, University of California, San Diego; Ben Sigelman, Google; Rich Wolski, University of California, Santa Barbara/ Eucalyptus Systems

Garth Gibson, "RAID for Clouds"

Garth Gibson questioned whether we have "the answer" to storage in clouds. He pointed out that common storage methods triplicate every file block, resulting in a 200% storage overhead. He suggested a strategy, called DiskReduce, that replaces duplicates with parity blocks to implement distributed RAID 5. Repair of defective blocks is a background task deferred to times when storage is otherwise idle. The strategy is tuned to perform optimally for realistic file-system contents, where he estimated that 58% of files use eight blocks or less, and 25% of files fit into a single block.

Gibson noted that the true necessary complexity of a storage stack is an unsolved problem and suggested that developing a definitive understanding of complexity in storage is a challenge problem worthy of the Turing Award.

Gibson noted that infrastructure management is now in its third generation. The first generation involved clustering with Beowulf and condor. The second generation introduced virtualization via VMware and Xen. The third generation introduced elasticity. In Gibson's opinion, the fourth generation will reintroduce time-sharing, in service agreements for response time.

Gibson also mentioned several general cloud challenges, including refining the business model, balancing eventual consistency of data against buggy code that needs strong consistency, and a need for testing at scale. We need expensive resources that we can safely "crash and trash."

Stefan Savage, "Are Cloud Privacy and Security Possible?"

Stefan Savage concentrated on security and availability issues in third-party computing. While Infrastructure as a Service (IaaS) clouds leave primary responsibility in the hands of clients, other models of cloud computing assign responsibility for computing and storage to some third party. Implicitly, a cloud client trusts a cloud provider to provide privacy, as well as storage availability, integrity, durability, and retention limits. The cloud provider trusts cloud clients to act in compliance with "acceptable use" policies and to pay promptly and without contest. There is an implicit (and perhaps unfounded) expectation that the cloud provider will monitor clients for appropriate behavior.

Data privacy is a severe problem. A partial solution is "opaque" storage that is encrypted on disk, but key distribution and management remains an unsolved problem. Aside from technical issues, the Stored Communications Act (SCA) grants third-party data less protection than data stored at a first-party site, and it is unclear whether the privacy mechanisms available in clouds are sufficiently strong to satisfy regulations (e.g., HIPAA and PCI). Much less is known when cloud and customer are in different countries.

In a technical sense, Savage noted that data privacy is threatened not only by privileged access, but also by the existence of side-channels through which one customer can determine the transient state of another, e.g., determining transaction volume by observing the timing of cache or memory flushes, or even via the observed behavior of block translation buffers. This gives one customer real-time information on the state of another that can lead to a competitive advantage.

Durability of storage has both technical and legal aspects. How does one "prove" that storage is durable? What happens in case the cloud business fails? In a recent case, a cloud provider deleted 4% of customer information irretrievably, and the customer had no recourse. A year later the company went out of business, and in transferring their data to another company, one-half of all customer information was irretrievably lost with no customer recourse.

Another ambiguity is what is meant by availability. How do you know your provider is a good "steward" of your data? Cloud providers offer "availability zones" but no one knows what they mean. Meanwhile, lack of availability is reimbursed as cost of the service, rather than the cost of the business loss due to lack of availability. There may be a role for the insurance industry in mitigating the risks that arise from this disparity.

Another ambiguity in cloud hosting is the nature of retention. How does a customer know that deleted data is really gone? Supposedly deleted data can be subpoenaed, and the courts have not supported Fifth Amendment rights for encrypted data.

Cloud computing has inherent risks for both client and provider. Clients risk corruption/subversion of VM images, problems of jurisdiction, and inability to verify the privacy of cloud data. For providers, cloud infrastructure is a cyber-criminal's dream world, with plenty of ambiguity and anonymity behind which to hide. What could be more ideal for the cyber-criminal than paying for a huge amount of untraceable computing infrastructure with a stolen credit card?

Ben Sigelman, "The 'Elephant in the Datacenter' and Cloud Monitoring"

Ben Sigelman discussed the problems of monitoring clouds. The "elephant in the data center" is that clouds are actually quite difficult to use. Infrastructure degrades and changes over time, developers move on, and performance of distributed applications is counterintuitive when one understands only the serial version. The failures we observe are only the subset that is visible, making troubleshooting very difficult. These are all evidence that the building blocks we are using for monitoring are wrong. Programming languages haven't

adapted. The time spent on seemingly trivial tasks is alarming.

Recent work at Google on monitoring includes distributed "always-on" event tracing, correlated with low-overhead counters for performance monitoring and accounting. Selected events are traced end-to-end, and request-response times can be broken into components and analyzed in detail. Implementing this kind of monitoring requires standardization, including ubiquitous IPC/RPC mechanisms and control-flow libraries. Monitoring is best considered an independent platform in the cloud.

Challenges to cloud monitoring include needs for standardized APIs for monitoring data, as well as ex post facto accounting. Azure/AppEngine-like systems should expose detailed performance info for APIs. For accounting purposes, we do not know the cost of a write until after the write occurs.

Rich Wolski, "The Self-Owned Open-Source Cloud"

Rich Wolski discussed the role of open source in clouds and the relationship between open source self-owned clouds and the current "retail sales" model of cloud service purchase.

Current public clouds are based on a "retail sales" model that quite literally employs the same infrastructure to rent CPUs as to buy DVDs. Public clouds are dependent on customer self-service and a concept of "quality-of-service" that is misnamed a "service-level agreement." Accountability between customer and provider is based on e-commerce. A customer with a problem is treated like a customer who is dissatisfied with a material purchase.

Meanwhile, management models for clouds are just as valid in the self-owned data center as in the cloud, and upcoming challenges in data assimilation from ubiquitous sources, multi-player gaming, and applications for mobile devices require a new level of infrastructure that is present in clouds but not present in current self-owned data centers.

One solution to this problem is the self-owned, open source cloud. Eucalyptus is one of the first enabling technologies for creating one's own clouds. Eucalyptus (an elastic utility computing architecture) is a Linux hosting service that is simple, extensible, commodity-based, and easy for system administrators to install and maintain. Using Eucalyptus, one can emulate first-generation cloud services such as Amazon Advanced Web Services easily and quickly.

Intended uses of Eucalyptus include cloud research, as well as homogenization of existing self-owned IT infrastructure. It is not intended as a replacement for commercial cloud services, but, rather, as an open prototyping environment that enables research and open source development.

Challenges of clouds include federation, privacy, cost, and storage. Federation is a policy mediation problem. "Private" clouds are actually hybrid clouds with both private and public information. Cost of cloud services is increasingly becoming a "first-class" object, in the sense that algorithms are measuring cost and reacting directly. We have not seen "the" cloud storage model yet.

A short discussion followed, in which several questions were raised. Is it even more difficult to have a testbed than to set up a cloud? Panasas never tested hardware at anywhere near the scale that people are purchasing. Cost and incentive models are hard to understand. If you do not believe this, try teaching a cloud computing course to undergraduates. They do not understand that they are spending money until they "see the bill" for what they did during the course. What is the Eucalyptus business model? When one starts a venture-backed company, one bases one's model on serving the enterprise. Eucalyptus will develop and sell customizations that enable enterprise needs.

STORAGE CLOUD AND APPLIANCES

Full Papers

- ***In Search of an API for Scalable File Systems: Under the Table or Above It?***
  Swapnil Patil, Garth A. Gibson, Gregory R. Ganger, Julio Lopez, Milo Polte, Wittawat Tantisiroj, and Lin Xiao, Carnegie Mellon University

Data-intensive scalable computing (DISC) systems, intended to process and store massive data sets, have built their own distributed file systems (e.g., Google File System, Hadoop Distributed File System [HDFS]). By contrast, cluster file systems such as the Parallel Virtual File System (PVFS) have been used to run larger-scale workloads by the High Performance Community (HPC) community for about a decade. The authors explore how to evolve the file system API used by the HPC community so that they can be used for DISC workloads. The authors propose extending traditional cluster file systems to expose block layout to applications, thus allowing applications to co-locate computation with data. The authors built a lightweight shim layer that connects Hadoop and PVFS. Through this shim, they added three functions: read-ahead, co-location of compute with data, and exposing file block layout to applications. Their experiments show that PVFS with the shim layer performs comparably to HDFS. Second, most DISC systems use databases with weaker semantics than traditional databases to store and query metadata. The authors propose a mechanism for using the file system with a filtered directory scan to provide similar functionality.

- ***CloudViews: Communal Data Sharing in Public Clouds***
  Roxana Geambasu, Steven D. Gribble, and Henry M. Levy, University of Washington

Currently, most Web services store and process their data in their own data center. For example, Flickr and Picasa have similar interfaces, but both of them reimplement the software stack from the ground up. With the advent of public cloud services, however, Web services can "rent" themselves to each other, which is made easier by sharing data among

co-located services. CloudViews is a storage system that is designed so that services running on a cloud can share data with each other. CloudViews provides database-style views for data sharing between applications. For example, in CloudViews, a Flickr-like service might create a view that shares photos to an automatic photo tagging service but not the ownership information of the photos. The challenges in such a service include providing a scalable protection mechanism, query admission control, and QoS for resource allocation. A member of the audience pointed out that views are good for read-only data and another member asked how CloudViews shares metadata between services. The author replied that both these issues are good material for future research.

- ***Cloud Analytics: Do We Really Need to Reinvent the Storage Stack?***
  Rajagopal Ananthanarayanan, Karan Gupta, Prashant Pandey, Himabindu Pucha, Prasenjit Sarkar, Mansi Shah, and Renu Tewari, IBM Research

MapReduce workloads are generally executed on Internet-scale file systems, such as Google File System (GFS), that do not provide a POSIX interface. The authors explore the suitability of traditional cluster-based file systems for such workloads. In particular, they compare HDFS (an open source implementation of GFS) with IBM's GPFS cluster file system. Compared to GPFS, HDFS provides larger data blocks (on the order of 64MB), allows applications to co-locate computations with data by exposing block locations to applications, and provides data availability in case of node and disk failures.

To verify that they could bridge the gap between HDFS and GPFS, the authors modified GPFS to expose the block location information to MapReduce applications. Second, directly increasing GPFS block size to match that of HDFS is not feasible, as GPFS internally uses block size to perform prefetching. Instead, the authors introduce a new construct called a metablock, which is basically a consecutive set of (smaller) blocks of a file that are allocated on the same disk. The small blocks are used internally by GPFS to perform accounting, prefetching, etc., whereas the larger logical metablock is exposed to MapReduce applications. With these changes, the performance of the modified GPFS and HDFS are comparable. Further, the authors ran experiments to confirm that metablocks do not hurt the performance of GPFS for traditional applications. Thus, clustered file systems, enhanced appropriately, can provide the best of both the traditional applications and MapReduce workloads.

Short Papers

- ***Constructing and Managing Appliances for Cloud Deployments from Repositories of Reusable Components***
  Matthew S. Wilson, rPath, Inc.

The usual way to deploy applications is to start with a base image, install applications, snapshot the image, and then spin up new instances from snapshots. However, these snapshots are hard to move from one provider to another. Automation tools can help, but they require a new setup for each cloud environment. Instead, Matthew Wilson proposes handling software configuration management via a version control system. Dependencies between software components are encoded by grouping components with the components that they require. Once all software is managed and grouped under version control, one can build deployment images from these groups. One member of the audience asked how many companies are using their system. Wilson replied that companies can use their rPath software to do this or can use their rBuilder free online service. About 17,000 projects are using the service, and 50 companies have downloaded rPath. Another audience member asked whether they changed the operating system. Wilson replied that the operating system is changed as little as possible.

- ***Maximizing Efficiency by Trading Storage for Computation***
  Ian F. Adams, Darrell D.E. Long, and Ethan L. Miller, University of California, Santa Cruz; Shankar Pasupathy, NetApp; Mark W. Storer, Pergamum Systems

The authors argue that instead of storing data that is not frequently accessed in the cloud, it can be more cost-efficient to regenerate data on demand. For example, instead of pre-generating various formats of photos (BMP, jpeg, tiff, etc.), it might be more efficient to store photos in the most frequently used format and regenerate other formats on demand. To enable regeneration of data, one needs to record the inputs, processes, and provenance needed to regenerate the data. The decision whether data should be stored or regenerated is determined by cost-benefit analysis. The factors to consider in this analysis include data semantics (i.e., should the exact same data be regenerated or will any data generated by the same process suffice), the cost of regenerating data, and the cost of computing in the cloud in the future.

MAP REDUCE AND CLOUD APPLICATIONS

Full Papers

- ***Mochi: Visual Log-Analysis Based Tools for Debugging Hadoop***
  Jiaqi Tan, Xinghao Pan, Soila Kavulya, Rajeev Gandhi, and Priya Narasimhan, Carnegie Mellon University

Current debugging tools present debugging data at the wrong level of abstraction to be useful in debugging clouds. Mochi, instead, expresses MapReduce program execution in terms of the high-level operations "Map" and "Reduce." It extracts views of node behavior with SALSA, correlates execution traces, and creates a conjoined representation of control and data flow. Control flow consists of the order in which operations are executed, while data flow indicates how the output of one operation is used as an input to others. This conjoined representation is visualized in a number of ways using the R statistics system. The "swimlanes" visualization shows the extent of map and reduce operations

in time, so that wedged operations can be detected and addressed. "Realized execution paths" provide a statistical depiction of time spent in each processor state, while data flow depictions show how map and reduce functions relate to one another.

- ***A Common Substrate for Cluster Computing***
  Benjamin Hindman, Andy Konwinski, Matei Zaharia, and Ion Stoica, University of California, Berkeley

NEXUS is a common substrate level that allows several cloud frameworks with differing semantics to co-locate in the same cloud. It can also be used to run several versions of the same framework in one cloud. NEXUS is extremely lightweight and attempts to be a "microkernel" for serving cloud stacks. Performance experiments for a logistic regression machine-learning algorithm show that running Hadoop on top of NEXUS is negligibly slower than running Hadoop alone, but that running the same application on NEXUS alone is several times faster. Since microkernels were not successful, an audience member wondered, why do the authors expect NEXUS to be successful? By the time microkernels were introduced, there were a number of well-established players in the operating systems space, but the cloud is still young and can be changed.

- ***Using Proxies to Accelerate Cloud Applications***
  Jon Weissman and Siddharth Ramakrishnan, University of Minnesota, Twin Cities

A proxy can be utilized to speed up access to cloud services by having superior location or access to relevant resources. In a PlanetLab experiment, proxies were utilized to access 30 commercial Web services. Response times for 70% of queries were improved by proxying, with a 20% performance improvement on average among these. Proxies excel when a cloud application accesses multiple others, which can happen due to specialization of computing infrastructure or data store, distributed data mining, and mash-ups, among others. Open questions include whether to proxy and why, where to optimally locate proxies, and how to select a proxy from those available. The ability of a proxy to cache results or perform local computations has not been explored.

Short Papers

- ***DryadInc: Reusing Work in Large-scale Computations***
  Lucian Popa, University of California, Berkeley; Mihai Budiu, Yuan Yu, and Michael Isard, Microsoft Research, Silicon Valley

A Dryad job is a directed acyclic graph representing data flow in a distributed computation, where each vertex is a computation and each edge represents data flow. A Dryad job or set of jobs often involves redundant calculation of the same result several times. "Identical computation" (IDE) caches and reuses results of repeated computations, while "incremental merging" (MER) employs a user-crafted computation that incorporates new data into the results of a previous computation. The cost-effectiveness of IDE and MER depends on a time/space tradeoff and whether computation time or cache space is more expensive in context.

- ***Towards Optimizing Hadoop Provisioning in the Cloud***
  Karthik Kambatla and Abhinav Pathak, Purdue University; Himabindu Pucha, IBM Research Almaden

Hadoop has hundreds of configurable parameters. Current tools like Hadoop on demand and Cloudera are laborious to use when parameter tuning. One alternative is controlled experimentation. Trying a distributed grep with 1, 4, 8, 16, and 24 map nodes shows diminishing returns after use of 8 map nodes. Thus one can determine an appropriate number of map nodes by direct experimentation. Someone questioned the value of such a method given that such experiments would have to be done "at scale" and expensively in order to guarantee sufficient accuracy.