the rate of client requests remains bounded even when a server comes up after a long outage.

Security. Volunteer computing poses a variety of security challenges. What if hackers break into a project server and use it to distribute malware to the attached computers? BOINC prevents this by requiring that executables be digitally signed using a secure, offline signing computer. What if hackers create a fraudulent project that poses as academic research while in fact stealing volunteers' private data? This is partly addressed by account-based sandboxing: applications are run under an unprivileged user account and typically have no access to files other than their own input and outputs. In the future, stronger sandboxing may be possible using virtual machine technology.

Future of Volunteer Computing

Volunteer computing has demonstrated its potential for high-throughput scientific computing. However, only a small fraction of this potential has been realized. Moving forward will require progress in three areas.

1. Increased participation: The volunteer population has remained around 500,000 for several years. Can it be grown by an order of magnitude or two? A dramatic scientific breakthrough, such as the discovery of a cancer treatment or a new astronomical phenomenon, would certainly help it popularity. Or, the effective use of social networks like Facebook could spur more people to volunteer. Another way to increase participation might be to have computer manufacturers or software vendors bundle BOINC with other products. Currently, Folding@Home is bundled with the Sony Playstation 3 and with ATI GPU drivers.

2. Increased scientific adoption: The set of volunteer projects is small and fairly stagnant. It would help if more universities and institutions created umbrella projects, or if there were more support for higher-level computing models, such as workflow management systems and MapReduce. Two other factors that would increase scientific adoption are the promotion of volunteer computing by scientific funding agencies and and increased acceptance of volunteer computing by the HPC and computer science communities.

3. Tracking technology: Today, the bulk of the world's computing power is in desktop and laptop PCs, but in a decade or two it may shift to energy-efficient mobile devices. Such devices, while docked, could be used for volunteer computing.

If these challenges are addressed, and volunteer computing experiences explosive growth, there will be thousands of projects. At this point volunteers can no longer be expected to evaluate all projects, and new allocation mechanisms will be needed. For example, the "mutual fund" idea mentioned above, or something analogous to decision markets, in which individuals are rewarded for participating in new projects that later produce significant results. Such "expert investors" would steer the market as a whole.

Biography

David P. Anderson is a research scientist at the Space Sciences Laboratory at the University of California-Berkeley.

Clouds at the Crossroads

Research Perspectives

By Ymir Vigfusson and Gregory Chockler

Despite its promise, most cloud computing innovations have been almost exclusively driven by a few industry leaders, such as Google, Amazon, Yahoo!, Microsoft, and IBM. The involvement of a wider research community, both in academia and industrial labs, has so far been patchy without a clear agenda. In our opinion, the limited participation stems from the prevalent view that clouds are mostly an engineering and business-oriented phenomenon based on stitching together existing technologies and tools.

Here, we take a different stance and claim that clouds are now mature enough to become first-class research subjects, posing a range of unique and exciting challenges deserving collective attention from the research community. For example, the realization of privacy in clouds is a cross-cutting interdisciplinary challenge, permeating the entire stack of any imaginable cloud architecture.

The goal of this article is to present some of the research directions that are fundamental for cloud computing. We pose various challenges that span multiple domains and disciplines. We hope these questions will provoke interest from a larger group of researchers and academics who wish to help shape the course of the new technology.

An Architectural View

The physical resources of a typical cloud are simply a collection of machines, storage, and networking resources collectively representing the physical infrastructure of the data center(s) hosting the cloud computing system. Large clouds may contain some hundreds of thousands of computers.

The distributed computing infrastructure offers a collection of core services that simplify the development of robust and scalable services on top of a widely distributed, failure-prone, physical platform. The services supported by this layer typically include communication (for example, multicast and publish-subscribe), failure detection, resource

usage monitoring, group membership, data storage (such as distributed file systems and key-value lookup services), distributed agreement (consensus), and locking.

The application resource management layer manages the allocation of physical resources to the actual applications and platforms including higher-level service abstractions (virtual machines) offered to end-users. The management layer deals with problems related to the application placement, load balancing, task scheduling, service-level agreements, and others.

Finally, we enumerate some cross-cutting concerns that dissect the entire cloud infrastructure. We will focus on these issues: energy, privacy and consistency, the lack of standards, benchmarks, and test beds for conducting cloud related research.

Energy

Large cloud providers are natural power hogs. To reduce the carbon footprint, data centers are frequently deployed in proximity to hydroelectric plants and other clean energy sources. Microsoft, Sun, and Dell have advocated putting data centers in shipping containers consisting of several thousand nodes at a time, thus making deployment easier. Although multi-tenancy and the use of virtualization improves resource utilization over traditional data centers, the growth of cloud provider services has been rapid, and power consumption is a major operating expense for the large industry leaders.

Fundamental questions exist of how, where, and at what cost can we reduce power consumption in the cloud. Here we examine three examples to illustrate potential directions.

Solid-state disks (SSDs) have substantially faster access times and draw less power than regular mechanical disks. The downside is that SSDs are more expensive and lack durability because blocks can become corrupted after 100,000 to 1,000,000 write-erase cycles. SSDs have made their way into the laptop market—the next question is whether cloud data centers will follow [14]. Can we engineer mechanisms to store read-intensive data on SSDs instead of disks?

Google has taken steps to revamp energy use in hardware by producing custom power supplies for computers which have more than double the efficiency of regular ones [12]. They even patented a "water-based" data center on a boat that harnesses energy from ocean tides to power the nodes and also uses the sea for cooling. How can we better design future hardware and infrastructure for improved energy efficiency? How can we minimize energy loss in the commodity machines currently deployed in data centers?

In the same fashion that laptop processors adapt the CPU frequency to the workload being performed, data center nodes can be powered up or down to adapt to variable access patterns, for example, due to diurnal cycles or flash crowds. Some CPUs and disk arrays have more flexible power management controls than simple on/off switches, thus permitting intermediate levels of power consumption [13]. File systems spanning multiple disks could, for instance, bundle infrequently accessed objects together on "sleeper" disks [9]. More generally, how should data and computation be organized on nodes to permit software to decrease energy use without reducing performance?

Privacy Concerns

Storing personal information in the cloud clearly raises privacy and security concerns. Sensitive data are no longer barred by physical obscurity or obstructions. Instead, exact copies can be made in an instant.

Technological advances have reduced the ability of an individual to exercise personal control over his or her personal information, making it elusive to define privacy within clouds [5]. The companies that gather information to deliver targeted advertisements are working toward their ultimate product: you. The amount of information known by large cloud providers about individuals is staggering, and the lack of transparent knowledge about how this information is used has provoked concerns.

Are there reasonable notions of privacy that would still allow businesses to collect and store personal information about their customers in a trustworthy fashion? How much are users willing to pay for additional privacy?

We could trust the cloud partially, while implementing mechanisms for auditing and accountability. If privacy leaks have serious legal repercussions, then cloud providers would have incentives to deploy secure information flow techniques (even if they are heavy-handed) to limit access to sensitive data and to devise tools to locate the responsible culprits if a breach is detected [17]. How can such mechanisms be made practical? Is the threat of penalty to those individuals who are caught compromising privacy satisfactory, or should the cloud be considered an untrusted entity altogether?

If we choose not to trust the cloud, then one avenue of research is to abstract it as a storage and computing device for encrypted information. We could use a recent invention in cryptography called fully homomorphic encryption [10]; a scheme allowing the sum and multiplication (and hence arbitrary Boolean circuits) to be performed on encrypted data without needing to decrypt it first. Unfortunately, the first implementations are entirely impractical, but beg the question whether homomorphic encryption can be made practical.

Another approach is to sacrifice the generality of homomorphic encryption. We can identify the most important functions that need to be computed on the private data and devise a practical encryption scheme to support these functions—think MapReduce [7] on encrypted data. As a high-level example, if all emails in Gmail were encrypted by the user's public key and decrypted by the user's web browser, then Gmail could not produce a search index for the mailbox. However, if each individual word in the email were encrypted, Gmail could produce an index (the encrypted words would just look like a foreign language) but would not understand the message contents.

The latter case implies that Gmail could not serve targeted ads to the user. What are the practical points on the privacy versus functionality spectrum with respect to computational complexity and a feasible cloud business model? Secure multiparty computation (SMC) allows mutually distrusting agents to compute a function on their collective inputs without revealing their inputs to other agents [19]. Could we partition sensitive information across clouds, perhaps including a trusted third-party service, and perform SMC on the sensitive data? Is SMC the right model?

Consistency

In a broad sense, consistency governs the semantics of accessing the cloud-based services as perceived by both the developers and end users. The consistency issues are particularly relevant to the distributed computing infrastructure services (see Figure 1), such as data storage.

The most stringent consistency semantics, known as serializability or strong consistency [11], globally orders the service requests and

presents them as occurring in an imaginary global sequence. For example, suppose Alice deposits $5 to a bank account with the initial balance of $0 concurrently with Bob's deposit of $10 to the same account. If Carol checks the account balance twice and discovers it first to be $10 and then $15, then no user would ever see $5 as the valid balance of that account (since in this case, Bob's deposit gets sequenced before Alice's). In the database community, this type of semantics is typically implied by ACID (atomicity, consistency, isolation, and durability).

Intuitively, supporting serializability requires the participants to maintain global agreement about the command ordering. Since cloud services are typically massively distributed and replicated (for scalability and availability), reaching global agreement may be infeasible. Brewer's celebrated CAP theorem [2] asserts that it is impossible in a large distributed system to simultaneously maintain (strong) consistency, availability, and to tolerate partitions—that is, network connectivity losses.

Researchers have looked for practical ways of circumventing the CAP theorem. Most work has so far focused on relaxing the consistency semantics; basically substituting serializability or (some of) the ACID properties with weaker guarantees. For instance, it does not matter if Carol and Bob in the example above would see either $5 or $10 as the intermediate balances, as long as both of them will eventually see $15 as the final balance.

This observation underlies the notion of eventual consistency [18], which allows states of the concurrently updated objects to diverge provided that eventually the differences are reconciled, for example, when the network connectivity is restored.

Apart from eventual consistency, other ways of weakening consistency semantics have looked into replacing single global ordering with multiple orderings. For instance, causal consistency [1] allows different clients to observe different request sequences as long as each observed sequence is consistent with the partial cause-effect order.

Weaker consistency semantics work well only for specific types of applications, such as cooperative editing, but do not easily generalize to arbitrary services. (Just imagine what would happen if withdrawals were allowed in the bank account example above.) Moreover, semantics that are weaker than serializability (or ACID) tend to be difficult to explain to users and developers lacking the necessary technical background.

Yet another problem is that for certain types of data, such as the meta-data of a distributed file system, it might be inherently impossible to compromise on strong consistency without risking catastrophic data losses at a massive scale. The possible research questions here would have to address questions such as can we produce a comprehensive and rigorous framework to define and reason about the diverse consistency guarantees. The framework should unify both weaker and stronger models and could serve as a basis for rigorous study of various consistency semantics of cloud services and their relative power. It should be expressive enough to allow new properties to be both easily introduced, for example by composing the existing basic properties, and understood by both developers and consumers of the cloud services. It should also help to bridge diverse perspectives on consistency that exist today within different research communities like the database and distributed systems communities.

Although it is well understood that a cloud architecture should accommodate both strongly and weakly consistent services, it is unclear how the two can be meaningfully combined within a single system. How should they interact, or what implications would such a model have on performance and scalability?

Current approaches to supporting strong consistency primarily focus on isolating the problem into "islands" of server replicas. While beneficial for scalability, such an approach creates an extra dependency on a set of servers that have to be carefully configured and maintained. Can we make strong consistency services that are more dynamic and easier to reconfigure, providing a simpler and more robust solution?

Standards, Benchmarks, Test Beds

Technical innovations are often followed by standards wars, and cloud computing is no exception. There is a plethora of cloud interoperability alliances and consortia (for example, Open Cloud Manifesto, DTMF Open Cloud Standards Incubator, Open Group's Cloud Work Group). The largest incumbents in the market are nevertheless reluctant to follow suit and have chosen to define their own standards. Whereas the strategy is understandable, the lack of interoperability may have adverse effect on consumers who become locked-in on a single vendor. The worry is that clouds become natural monopolies.

The Internet was built on open standards. The question is whether clouds will be as well. Making cloud services open and interoperable may stimulate competition and allow new entrants to enter the cloud market. Customers would be free to migrate their data from a stagnant provider to a new or promising one without difficulty when they so choose. Can the smaller players leverage their collective power to lobby for an open and flexible cloud computing standard that fosters competition while still allowing businesses to profit? Or can this be accomplished by the larger companies or governments? What business models are suitable for an open cloud? On the technical side, could users switch between providers without needing their support, for instance by using a third-party service?

Different cloud providers often adopt similar APIs for physical resources and the distributed computing infrastructure. For instance, MapReduce and Hadoop expose a similar API, as do the various key-value lookup services (Amazon's Dynamo [8], Yahoo!'s PNUTS [6], memcached [4]). Other components have more diverse APIs, for instance locking services like Google's Chubby [3], Yahoo!'s Zookeeper [16], and real-time event dissemination services. The broad question asks what components and interfaces are the "right" way to provide the cloud properties mentioned previously. A more specific question is how we can compare and contrast different implementations of similar components. For instance, how can we evaluate the properties of key-value stores like PNUTS and Facebook's Cassandra [15]?

The most appealing approach is to compare well-defined metrics on benchmark traces, such as the TPC benchmark for databases (www.tpc.org). How can we obtain such traces, or perhaps synthetically generate them until real ones are produced? Also, consensus benchmarks enable researchers outside the major incumbent companies to advance the core cloud technologies.

Developing distributed computing infrastructure layers or data storage systems is a hard task, but evaluating them for the massive scale imposed by clouds without access to real nodes is next to impossible. Academics who work on peer-to-peer systems (P2P), for example, rely heavily on the PlanetLab (www.planet-lab.org) test bed for deployment. PlanetLab constitutes more than 1,000 nodes distributed across nearly 500 sites, making it ideal an ideal resource for experimental validation of geographically networked systems which sustain heavy churn (peer arrivals and departures). The nodes in the data centers underlying the cloud tend to be numerous, hierarchically structured with respect to networking equipment, and face limited random churn but occasionally suffer from large-scale correlated failures.

PlanetLab's focus on wide-area networks is suboptimal for cloud platform research, unfortunately, and the same holds true for other similar resources. A handful of test beds appropriate for cloud research have made their debut recently, including Open Cirrus from HP, Intel and Yahoo!, and the Open Cloud Testbed. We encourage other players to participate and contribute resources to cloud research, with the goal of providing a standard test bed with open-access, at least for academia, including researchers from underrepresented universities. Who will create the future "CloudLab"?

How to Get Involved

Students and researchers who are interested in shaping cloud computing should consider participating in the LADIS (**www.cs.cornell.edu/projects/ladis2010**) or HotCloud (**www.usenix.org/events/hotcloud10**) workshops, or the upcoming Symposium on Cloud Computing (SoCC: **http://research.microsoft.com/en-us/um/redmond/events/socc2010**). Large industry players are currently driving the research bandwagon for cloud computing, but the journey is only beginning. A concerted multi-disciplinary effort is needed to turn the cloud computing promise into a success.

Biographies

Dr. Ymir Vigfusson is a postdoctoral researcher with the Distributed Middleware group at the IBM Research Haifa Labs. His research is focused around distributed systems, specifically, real-world problems that embody deep trade-offs. He holds a PhD from Cornell University.

Dr. Gregory Chockler is a research staff member in the Distributed Middleware group at the IBM Research Haifa Labs. His research interests span a wide range of topics in the area of large-scale distributed computing and cloud computing. He is one of the founders and organizers of the ACM/SIGOPS Workshop on Large-Scale Distributed Systems and Middleware (LADIS). He holds a PhD from the Hebrew University of Jerusalem.

References

1. Ahamad, M., Hutto, P. W., Neiger, G., Burns, J. E., and Kohli, P. 1995. Causal memory: Definitions, implementations and programming. *Distributed Comput. 9*. 37-49.
2. Brewer, E. 2000. Towards robust distributed systems. In *Proceedings of Principles of Distributed Computing* (*PODC*).
3. Burrows, M. 2006. The Chubby lock service for loosely-coupled distributed systems. In *Proceedings of the 70th USENIX Symposium on Operating Systems Design and Implementation* (*OSDI'06*). USENIX Association. 335-350
4. Danga Interactive. memcached: A distributed memory object caching system. **http://www.danga.com/memcached/**.
5. DeCandia, G., Hastorun, D., Jampani, et al. 2007. Dynamo: Amazon's highly available key-value store. In *Proceedings of the 21st ACM SIGOPS Symposium on Operating Systems Principles* (*SOSP'07*). Association for Computing Machinery. 205-220.
6. Cavoukian, A. 2008. Privacy in the clouds. White Paper on Privacy and Digital Identity: Implications for the Internet. **http://www.ipc.on.ca/images/Resources/privacyintheclouds.pdf**.
7. Cooper, B., Ramakrishnan, R., et al. 2008. PNUTS: Yahoo!'s hosted data serving platform. *Proc. VLDB Endow. 1*, 2. 1,277-1,288.
8. Dean, J. and Ghemawat, S. 2008. MapReduce: Simplified data processing on large clusters. *Comm. ACM 51*, 1. 107-113.
9. Ganesh, L., Weatherspoon, H., Balakrishnan, M., and Birman K. 2007. Optimizing power consumption in large-scale storage systems. In *Proceedings of HotOS*.
10. Gentry, C. 2009. Fully homomorphic encryption using ideal lattices. In *Proceedings of the ACM Symposium on Theory of Computing* (*STOC'09*).
11. Gray, J., and Reuter, A. 1993. Isolation concepts. In *Transaction Processing: Concepts and Techniques*, chap. 7. Morgan Kaufmann.
12. Hoelzle, U. and Weihl, B. 2006. High-efficiency power supplies for home computers and servers. Google Inc. **http://services.google.com/blog_resources/PSU_white_paper.pdf**.
13. Khuller, S., Li, J., and Saha, B. 2010. Energy efficient scheduling via partial shutdown. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms* (*SODA*).
14. Narayanan, D., Donnelly, A., Thereska, E., Elnikety, S., and Rowstron, A. 2009. Migrating server storage to SSDs: Analysis of tradeoffs. In *Proceedings of EuroSys*.
15. Ramakrishnan, R. 2009. Data management challenges in the cloud. In *Proceedings of ACM SIGOPS LADIS*. **http://www.cs.cornell.edu/projects/ladis2009/talks/ramakrishnan-keynote-ladis2009.pdf**.
16. Reed B. and Junqueira, F. P. 2008. A simple totally ordered broadcast protocol. In *Proceedings of the 2nd Workshop on Large-Scale Distributed Systems and Middleware* (*LADIS'08*). ACM.
17. Smith, G. 2007. Principles of secure information flow analysis. In *Malware Detection*, Christodorescu, M., et al. Eds., Springer-Verlag. Chap. 13, 291-307.
18. Vogels, W. 2008. Eventually consistent. *ACM Queue 6*, 6.
19. Wenliang, D. and Atallah, M. J. 2001. Secure multi-party computation problems and their applications: a review and open problems. In *Proceedings of the Workshop on New Security Paradigms*.