Querying Large Distributed Infrastructures

Steven Y. Ko[1] sko@cs.uiuc.edu
Praveen Yalagandula[2] praveen.yalagandula@hp.com
Indranil Gupta[1] indy@cs.uiuc.edu
Vanish Talwar[2] vanish.talwar@hp.com
Dejan Milojicic[2] dejan.milojicic@hp.com
Subu Iyer[2] subu.iyer@hp.com
[1]University of Illinois, Urbana-Champaign
[2]HP Labs, Palo Alto

Recent years have witnessed a dramatic increase in the use of large-scale distributed infrastructures in many different domains. Efforts in data center consolidation lead to data centers consisting of thousands of machines at one site. Networking testbeds such as Emulab [1] and PlanetLab [2] provide researchers with access to hundreds of machines instantly. Federated computational grids like Grid2003 [4] allow scientists to perform long-running experiments with unprecedented computational power. Yet the scale of these infrastructures is continually growing as the demand for these infrastructures increases.

This ever-increasing scale of distributed infrastructures naturally comes with the challenge of *managing* these infrastructures. For example, consider a large enterprise with several consolidated data centers spread over multiple geographical areas. Each data center has a few thousand physical machines and each machine runs several virtual machines with heterogeneous operating systems. This virtualized environment adds unpredictable dynamism in the data centers since each virtual machine is created and destroyed as users come and go throughout the day. In this setting, the IT personnel who is in charge of managing the data centers face a number of management tasks such as patch management, resource allocation, license management, auditing, etc. The scale and dynamism in this virtualized environment make the tasks even more challenging.

As another example, consider PlanetLab, a wide-area shared network testbed hosted by research institutions across the world. Users of PlanetLab can access hundreds of machines instantly by creating a *slice* of PlanetLab's wide-area resources. Then they can perform any network-based experiments on top of their own slices. In this setting, the users may wish to keep track of their slices' status such as CPU utilization and disk usage, as well as to monitor their applications performance by looking at the logs from the applications.

At the heart of these management tasks lies the need for an efficient and scalable system that gives users and/or operators the ability to query the infrastructure being managed. Through this query system, users and operators can gain insights into the past and present system-wide behaviors. Thus, the query system enables users and operators to make well-informed decisions for their management tasks at hand. Table 1 shows some of the example queries for management tasks.

Existing commercial solutions, including HP OpenView and IBM Tivoli, targets precisely that ability - the ability of querying system-wide information. However, these solutions are centralized in that they use databases to gather and store system-wide information. Thus, their centralized architecture prevents the infrastructure from growing into an arbitrarily large scale, *i.e.*, they impose a fundamental limitation on scalability. Also, several existing peer-to-peer data aggregation and querying systems [3, 5, 6, 8] are either not flexible, meaning that they only support certain types of queries, or too expensive in terms of resource consumption (e.g., bandwidth and CPU consumption), since each query requires information from every

Tasks	Queries
Testbed Monitoring	Maximum disk usage of all VM hosts in the enterprise
	Average CPU utilization of hosts that run company portal
Data Center Patch Management	IP address of all workstations that need a virus update
	Hostnames of machines that are running Windows XP SP1
Resource Allocation	List of machines that consume more than 60% CPU
	List of machines that run more than one application

Table 1: *Management Queries*

machine even when the query involves only a subset of the entire machines.

Currently, we are developing a scalable and flexible querying system for large distributed infrastructures. Our system is scalable in terms of the number of nodes in the system, the number of attributes that the system can gather, and finally the number of queries that the system can deal with. Our system is flexible since we support complex queries that are more general than the types of queries that most decentralized systems support. Our solution leverages DHT overlay routing algorithms to construct multiple trees over the nodes and performs hierarchical aggregation for different attributes along different trees. Scalability is achieved with respect to queries through efficiently resolving subset queries: queries that require aggregating values for an attribute from only a subset of nodes.

We have built a prototype of our solution leveraging FreePastry framework [7] and SDIMS [8], a scalable aggregation middleware. We have deployed 500 instances of our prototype on Emulab testbed and evaluated it with several microbenchmarks and with a trace from a real data center for a wide variety of queries. Our experimental results indicate that our solution has up to 5X and 2X lower response latency than global aggregation systems in micro-benchmark experiments and trace driven experiments respectively. We are continuing to develop our solution further and are implementing our support for scalability in terms of the number of queries, and complex query processing engine on top of SDIMS.

References

[1] Emulab. `http://www.emulab.net`.

[2] PlanetLab. `http://www.planet-lab.org`.

[3] R. Huebsch, B. Chun, J. M. Hellerstein, B. T. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. R. Yumerefendi. The Architecture of PIER: an Internet-Scale Query Processor. In *Proceedings of CIDR*, 2005.

[4] Ian T. Foster et al. The Grid2003 Production Grid: Principles and Practice. In *Proceedings of HPDC-13*, 2004.

[5] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TinyDB: An Acqusitional Query Processing System for Sensor Networks. *ACM Transactions on Database Systems*, 30(1):122–173, March 2005.

[6] R. V. Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21(2):164 – 206, May 2003.

[7] A. Rowstron and P. Druschel. Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. IFIP/ACM Middleware*, 2001.

[8] P. Yalagandula and M. Dahlin. A Scalable Distributed Information Management System. In *Proceedings of ACM SIGCOMM*, 2004.