Set Functionality over DHT Systems

Nuno Lopes and Carlos Baquero
Departamento de Informática, Universidade do Minho

January 2005

Abstract

For the last few years, Peer-to-Peer systems were created with the purpose of distributing information across a very large set of hosts. The scalability demands of such systems lead to the design of Distributed Hash Tables algorithms[1, 2, 3, 4], which are capable of locating an object anywhere on the system given a key value using logarithmic (or constant) message hops while maintaining logarithmic state at each host. Nonetheless, DHTs are not able to search for objects but only to retrieve them using keys.

Although DHT algorithms locate data objects efficiently, they do not provide any data consistency when using replication. Data replication is made on a "best-effort" way in which data may become inconsistent or even lost. An atomic data access DHT extension which maintained replica consistency using a state machine replication technique was proposed in [5] but required the system to have a stable host membership in order to progress. This requirement is not feasible on wide area systems, like the Internet, where hosts have small uptimes and are under constant churn[6]. Another protocol was proposed in [7] for supporting atomic mutable data in DHTs which is based on the Paxos consensus protocol, but its results have not yet been published.

While pursuing efficient search functionality on top of the DHT layer, we implemented a simple set functionality. Our algorithm uses a distributed B-tree to store set data over several hosts. This tree algorithm allows us to perform load-balancing on peers while maintaining logarithmic access to data items. We use any generic DHT algorithm as a simple storage layer, accessing data blocks through a simple key-based routing operation. This operation, $route(key, message)$, is capable of delivering a message to the host responsible for the key using the DHT scalable properties.

Our presentation will discuss the issues in implementing a tree-based algorithm over a distributed large-scale key-based routing system. We will also discuss how fault-tolerance (through block replication) influences both data consistency and availability.

References

[1] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable Peer-To-Peer lookup service for internet applications," in *Proceedings of the ACM SIGCOMM'01 Conference*, pp. 149–160, 2001.

[2] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms*, (Germany), 2001.

[3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," in *Proceedings of the ACM SIGCOMM'01 Conference*, pp. 161–172, 2001.

[4] F. Kaashoek and D. R. Karger, "Koorde: A simple degree-optimal hash table," in *Proceedings of the 2st International Workshop on Peer-to-Peer Systems (IPTPS'03)*, (Berkeley, USA), February 2003.

[5] N. Lynch, D. Malkhi, and D. Ratajczak, "Atomic data access in distributed hash tables," in *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, (Cambridge, USA), March 2002.

[6] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz, "Handling churn in a dht," Tech. Rep. UCB//CSD-03-1299, The University of California, Berkeley, December 2003.

[7] A. Muthitacharoen, S. Gilbert, and R. Morris, "Etna: a fault-tolerant algorithm for atomic mutable dht data," tech. rep., Massachusetts Institute of Technology, 2004. http://theory.lcs.mit.edu/ sethg/pubs/Etna.pdf.