Distributed Hash Tables on Adhoc Networks

Prasanna Gautam

October 22, 2010

Introduction

In an age where decentralization and parallelization are the key, distributed hash tables (DHTs) have had a dramatic impact on the decentralization and scalability of peer-to-peer applications. In essence, a DHT is a system designed to harness the storage and network resources of a large number of computers by providing a hash-table interface where *(key,value)* pairs are distributed across a large number of nodes—*keys* represent hashed objects that map to *values* that we are interested in [6]. Some of the most common uses, apart from academic research projects have been in peer-to-peer file transfer protocols like Bittorrent's distributed tracker where peer nodes are distributed across a network and can join or leave a group of connected peers at any given time. More recently, projects like Cassandra and Bigtable have emerged to provide structured storage systems using DHTs [2][5]. At the hardware level, wireless networks provide a similar communication strategy known as mobile ad-hoc networking (MANET) where the communicating mobile devices do not rely on preexisting infrastructure such as routers or access points for communication. Instead, each node is responsible for routing its data to other nodes. My project aims to explore the use of DHTs on adhoc networks to answer questions about the practicality, reliability, integrity and consistency in such networks. My previous experience has demonstrated testing and measurement of ad-hoc networks in the field is difficult at best . I want to leverage pre-existing network simulation frameworks like NS-2/NS-3, OPNET, GloMoSim, NetSim to accurately model DHTs on ad-hoc networks and use this knowledge to build my own simulation environment that will augment existing simulation infrastructures or if necessary construct a new simulation environment from scratch based on the requirements of the project.

Description

A DHT layer shields many difficult issues including fault-tolerance, object location, scalability, availability, load balancing for the distributed application designer[8]. Existing algorithms like "consistent hashing" implemented in DHTs have been proven to be fault tolerant and able to adapt to changing network topologies by efficiently partitioning a keyspace among a distributed set of nodes to provide an additional overlay network that connects nodes for efficient key lookup[4].

Here, by keyspace I mean the set of all the possible hashes that can be generated by a hashing function. For SHA1, the keyspace will be a set of 2^160 binary numbers. If a data like text, say this file's source, is entered, first the hash of the content is generated. The file will then be stored as (a4c74fef8068aa8e3cdad3e89e28642b24a4fcce, {content of proposal.tex}). In cases where the data can change, some unchangeable metadata can be used as content for the hashing function. If the data is bigger than a certain size, it can be partitioned into multiple chunks. These chunks can be passed as a linked list elements to the participating nodes. An element would be a tuple like (hash of previous chunk, hash of next chunk, hash of this chunk, data of this chunk, hash of the full data). These elements can then be replicated across nodes for availability. Various options have been explored to implement DHTs in MANETs using a proximity-aware DHT called Pastry[7] and an on-demand MANET routing protocol named DSR[3]. Based on my experience working on the Random Walk Gossip algorithm implementation in POSIT[1], I have realized that testing mobile ad-hoc networks consistently in the field can be very challenging thus requiring a detailed simulation framework for experimentation and measurement. My goals are to design a simulator for experimenting with DHTs on Ad-hoc networks and to explore DHT topologies and their impact on power consumption of the mobile devices . That said, I will be exploring and understanding various existing simulation frameworks to come up with a workable model to run my tests. The most important questions I aim to explore and answer are:

- How power efficient can a DHT be on an Adhoc network?
- What kinds of DHT topologies are preferrable?
- What are the effects of node "churn" (i.e., the adding or removing of nodes on the network)?
- What is the best way to deal with network partitioning?

The simulator would be written in clojure, a lisp dialect running on the JVM. My main motivation is that given that I have a rather short time frame of a year to complete this project, I want to be able to test the various components of the applications as independent of each other as possible. Using a functional language makes it much easier to write as I can test individual functions separately. I can also use a variety of existing Java and C libraries when necessary to simulate code that's much closer to the actual system code or when efficiency is an issue. The architectures of interest are primarily Android phones, the ADP1 and the Nexus one, and ad-hoc network implementations in Linux on laptops. I'll run a battery drain test on my targets with various stress levels to get an idea of the network usage with respect to power for the phones and run the simulations around those parameters. The end result of this project would be a simulator that does the network simulation that lets us create DHTs on a large number of randomly generated nodes, observe progress on each node and report and record the results. The variables that are relevant include: loss of data when a certain percentage

of nodes drop, power consumption on average of each of the algorithms, throughput of the network, and availability of a key/value pair on a network over time. Along with this, I want to explore these conditions in various network topologies and network conditions. This will mean testing on topologies like fully-connected, mesh, ring, tree on network conditions like the nodes being close together, separated by large distances, dropping in and out frequently, etc. and observing the variables mentioned above.

Timetable

- 30th September - Finish research on frameworks, relevant papers, required devices, libraries
- 15th October - Get experimental data about the devices being tested
- 22nd October - Finish preliminary architecture of the simulator
- 29th October - Report on existing frameworks
- 12th November - Technical Paper Presentation
- 7th December - Prototype of the simulator for implementing basic DHT network on simulated ad-hoc networks
- 15th December - Project report and findings from the semester

Budget

Since books on this topic are rather sparse in the library, I'd like to order a few books. Most of the network simulators I've mentioned are either open source or free under academic license but getting software might be necessary.

Conclusion

With this project I'm aiming to get a better understanding of the state of Distributed Hash Tables on Adhoc networks and test the practicality of those networks. I'll also learn about running network simulation and testing hypotheses about various DHTs on Adhoc network implementations with them. In the end, I want to have a working simulator to test and observe various network conditions by simulating real network hardware and conditions as well as possible.

References

[1] M. Asplund, T. deLanerolle, C. Fei, P. Gautam, R. Morelli, S. Nadjm-Tehrani, and G. Nykvist. Wireless ad hoc dissemination for search and rescue. In *7th International Conference on Information Systems for Crisis Response and Management*. ISCRAM, May 2010.

[2] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E. Gruber. Bigtable: A distributed storage system for structured data. In *IN PROCEEDINGS OF THE 7TH CONFERENCE ON USENIX SYMPOSIUM ON OPERATING SYSTEMS DESIGN AND IMPLEMENTATION - VOLUME 7*, pages 205–218, 2006.

[3] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *MOBILE COMPUTING*, pages 153–181. Kluwer Academic Publishers, 2002.

[4] David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin. Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 654–663, New York, NY, USA, 1997. ACM.

[5] Avinash Lakshman and Prashant Malik. Cassandra- a decentralized structured storage system.

[6] Moni Naor and Udi Wieder. Novel architectures for p2p applications: the continuous-discrete approach. *ACM TRANSACTIONS ON ALGORITHMS*, 3(3):50–59, 2007.

[7] Antony Rowstron and Peter Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems, 2001.

[8] Himabindu Pucha Saumitra, Saumitra M. Das, and Y. Charlie Hu. How to implement dhts in mobile ad hoc networks? In *Student poster, the 10th ACM International Conference on Mobile Computing and Network (MobiCom 2004), September-October*, 2004.