Informatics Research Proposal
A Resource Sensitive DHT Overlay

Peter Muir
s0450876@sms.ed.ac.uk

March 18, 2005

Abstract

DHTs are a method of efficiently performing lookups on massively distributed data. Current DHTs use a logical overlay network; no account is taken of the resources available to a node. Here a proximity senistive DHT overlay is proposed as a motivating example. To evaluate the performance of overlays a simulator will be developed which can simulate both 'vanilla' networks and 'skewed' networks.

1 Background

One of the current topics in the Database community is how to store, lookup and query massively distributed data efficiently. Distributed Hash Tables (DHTs) are a proposed solution that, in general, allow data to be looked up in $\log n$ time. Numerous DHT implementations have been proposed, all of which provide the same core interface (the ability to lookup, add and remove data) but differ in the routing algorithms used. An overview of some DHTs can be found in Section 4: 'Content Based Routing using DHTs' of [5].

There are a numerous applications for DHTs whether they are used to directly lookup data or whether they are used as the basis for query processors (e.g. PIER [3]) able to query massively distributed data fast. Applications of such systems include the infamous example of file sharing (e.g. Gnutella), time-shared file storage, code-breaking and cooperative mirroring. Any improvement in DHT performance should be apparent at the application level.

Each node within a DHT is responsible for a subset of the data stored in the DHT. When a node joins a DHT it is randomly assigned a 'position' or 'zone' within the DHTs coordinate space and becomes responsible for data in that zone. When a new (key, data) pair is added to the network it is assigned to a zone using a hash of the key. To lookup the data associated with a key, the key is hashed and a routing algorithm used to request the data from the node which is storing it. The routing algorithms used vary between DHT implementations, however a few key properties are common to most of them:

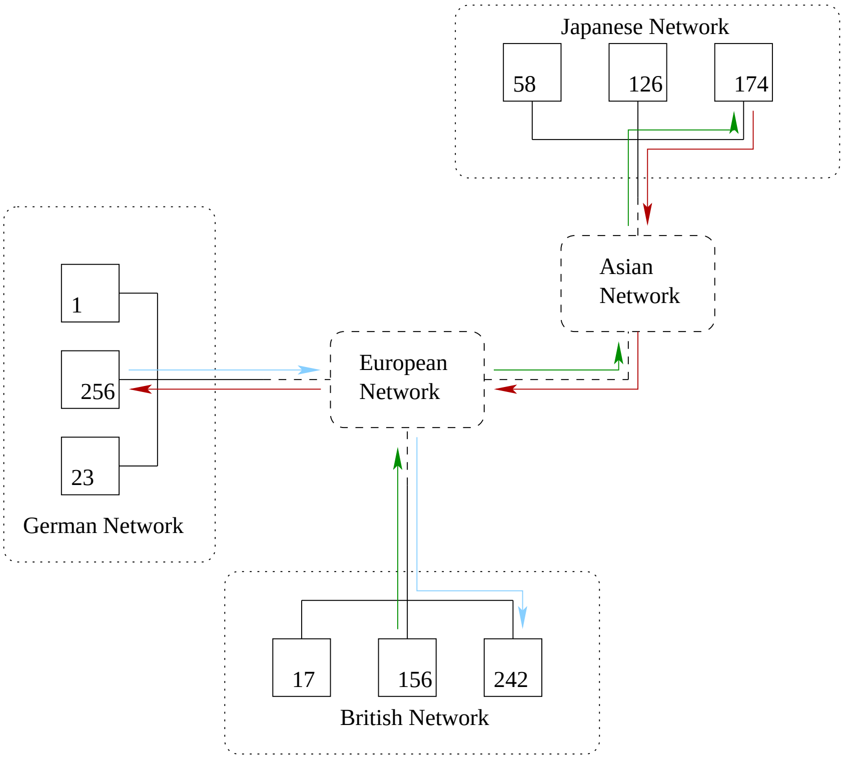
Lookup latency - the data lookup time (t) scales well (i.e. $t = O(n)$, where n is the number of (key, data) pairs)

Routing table size - the routing table (r) must scale well (i.e. $r = O(n)$), i.e. the routing table can know only about a subset of the nodes

Most DHT overlay schemes attempt a homogeneous distribution of nodes; to achieve this nodes are randomly assigned to positions.

2 Purpose

Most DHT implementations create a coordinate space such that, at a global scale, the dataset is uniformly distributed. When a node joins the DHT it is assigned a position in the space that is randomly allocated. Figure 1 shows an example network with physical links shown, and an 8-bit DHT node identifier; Figure 1(b) shows the routes that are known about. From this it can be seen that a lookup that originates at Node 17 (for example a workstation in the School of Informatics at Edinburgh University)



(a) Network with DHT node identifiers

Source	**Destination**
156	1
156	174
. . .	. . .
174	23
174	256
. . .	. . .
256	58
256	242

(b) Selected Parts of the nodes routing tables

Figure 1: An example DHT overlay network

that wants a key, data pair that is stored on Node 242 (e.g. a workstation in School of Phyiscs at Edinburgh University) might be routed via Japan and Germany.

As discussed above, most DHT schemes create overlay networks which assume that the resources available for use by each node are homogenous. This is often not true, as the above example (neighbour network proximity; Node 242 has a greater connectivity to node 17 than to node 256) shows. Other examples of resource levels which might impact upon DHT overlay structure include general link bandwidth (e.g. modem or T1), desired link load, node capability (e.g. processing power, storage space) and geographical/proximity location. The number of possible parameters are numerous and different setups will require different factors to be considered.

It is hypothesised that creating a DHT overlay network in resource sensitive fashion will improve the performance of the DHT.

Of the current DHT implementations none propose a resource sensitive overlay; however some skews to overlay networks have been proposed. CAN discusses the possibility of implementing a proximity heuristics based overlay structure but no results have been forthcoming. Pastry implements a scheme with proximity based routing, whereby the nodes it knows about are chosen to be close by.

3 Evaluation and Outputs

A system for creating resource sensitive overlays will be devised. Furthermore, a simulator will be developed to test the performance of overlay networks. It will be possible to simulate both a 'vanilla' network and one in which the overlay has been skewed.

The motivating example of a proximity sensitive overlay will be used throughout the project and it is intended that a proximity sensitive network will be created using the system developed. The proximity sensitive overlay network will be compared to a vanilla network. If time allows other resource sensitive networks will be considered.

The simulator will allow empirical evaluations of a proposed overlay through determined metrics. A DHT must route correctly, the routing table must be small relative to the number of nodes (normally number of entries $\approx \log n$) and the lookup time must be small relative to the number of nodes. It is not intended to develop routing algorithms in this project; a standard routing algorithm from the literature will be used. Many DHT designs use 'number of hops' to measure lookup time, however this assumes that internode latency is uniform which is unrealistic. More suitable metrics will be developed to measure lookup time.

There are a number network simulators that provide varying levels of abstraction, however this project needs only a simplistic view of a network and need not be concerned about the type of network being simulated. For this project it is proposed to develop a basic simulation framework that simulates both the nodes and the internode links.

3.1 Simulation Parameters

Here a list of parameters that the simulator should use are outlined. It is not intended that this should be an extensive list; it may be changed as the project is developed.

At a node:

Processing power How long a node takes to process a request

Storage Capacity The allocated storage ability of the node. A hard limit.

At a network edge:

Network topology The network structure will assumed to be Internet like and so a transit-stub topology will be used [1]

Bandwidth The capacity of the link

Latency The propogation delay of the link

4 Methods

It is intended to develop a general purpose system for designing and evaluating resource sensitive overlays.

Firstly, optimisation schemes for creating a resource sensitive overlay structure will be considered, and from this the motivating example of a proximity sensitive overlay will be developed. The

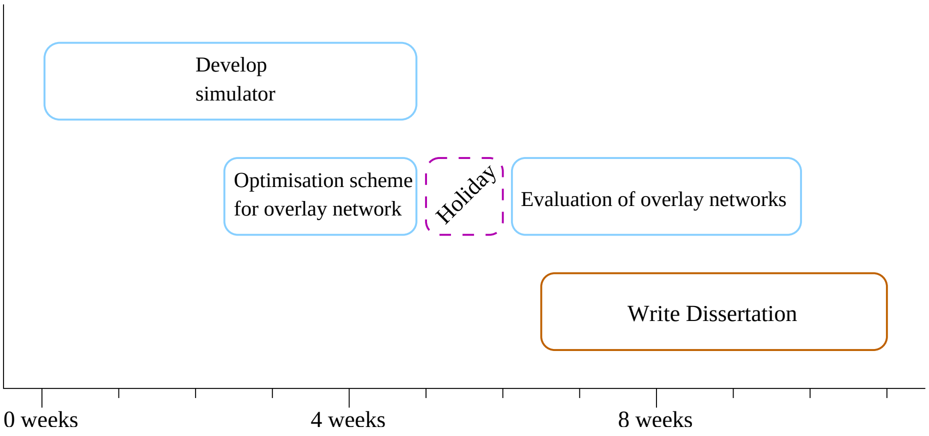


Figure 2: Proposed work plan

theoretical performace of the overlay network will be considered.

Secondly, a simulator will be developed that allows the perfomance metrics of the overlay to be studied. To evaluate any overlay it is necessary to have a comparison point and for this purpose a ‘vanilla’ overlay structure will be implemented in the simulator. The proximity sensitive overlay developed will then be implemented in the simulator.

It would be possible to simulate the entire underlying network however this is not practicable given the project time-scale. Instead a simplified message passing system with simulation steps of length s will be used:

1. Each node will be able to process p requests per simulation step. This simulates node processing capability

2. Each node can be responsible for at most h key, value pairs thus simulating storage capacity

3. Each node will have a send message queue and a receive message queue

4. The message queue will have a minimum propogation delay l; this simulates latency

5. The message queue will have a maximum number of messages per step b; this simulates the bandwidth

6. The network topology will be kept simple. A transit-stub topology will be assumed but with the links above the leaf nodes having a much higher bandwidth and a much lower latency than the leaf nodes. Therefore the predominant influence on bandwidth and latency will be from the leaf nodes own link thus negating the need to simulate the entire network. This allows for a simpler simulation. Each node will be aware of its position in the topology. It is beyond the scope of this project to develop algorithms for position finding ‘in the wild’.

It is intended that the simulator will be developed in Java using a simulation framework. This will allow emphasis to be placed on the simulation not on developing the simulator. The overlay network to be tested will be described using a Java object of a predefined type which the simulator can be instructed to load. This will allow considerable flexibility when specifying the overlay network.

There are two possible candidates, J-Sim [4] (developed at the Ohio State University) and Sim-Java [6], [2] (developed at the University of Edinburgh). Upon initial inspection J-Sim appears to be the more developed of the two simulation packages, however further evaluation is required.

The performance of the tested overlays will be compared. Both simulation packages allow statistics to be produced and have graph output packages.

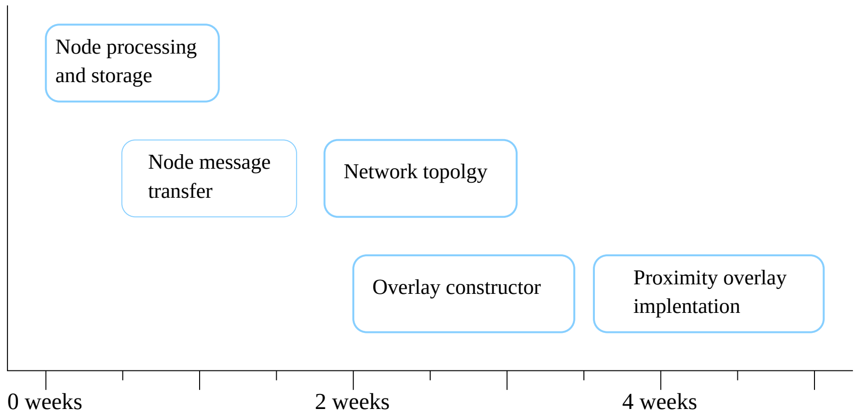


Figure 3: Simulator work plan

5 Workplan

The project has a timespan of 12 weeks. The proposed division of time is outlined in Figure 2.

The simulation development is the major component of the project, and a more detailed workplan is shown in Figure 3.

References

[1] Kenneth L. Calvert, Matthew B. Doar, and Ellen W. Zegura. Modeling internet topology. *IEEE Communications Magazine*, 35(6):160 – 163, June 1997.

[2] Jane Hilston. Modelling and simulation: Course notes. `http://www.inf.ed.ac.uk/teaching/courses/ms/`.

[3] Ryan Huebsch, Joseph M. Hellerstein, Nick Lanham, Boon Thau Loo, Scott Shenker, and Ion Stoica. Querying the internet with PIER. In *Proceedings of the 29th VLDB Conference*, Berlin, Germany, 2003.

[4] J-sim. `http://www.j-sim.org`.

[5] Peter Muir. Using relational databases in peer-to-peer networks. Informatics Research Review, 2004.

[6] Simjava. `http://www.icsa.inf.ed.ac.uk/research/groups/hase/simjava/`.