T-764-DATA – Spring 2025
Project 1: Distributed Hashing

Deadline is Feb. 9th

The primary purpose of the project twofold a) is to fully understand distributed hashing (DHT) and b) practice writing an academic paper. The former is important as we will observe, DHT as a key technique in several distributed systems, such as the noSQL systems we will read about.

Your task in this project will be to create a simple simulation model for distributed hashing, run some experiments and collect result data. The primary deliverable of this project is then an academic style paper where you give an introduction and background of the history of DHT algorithms, you explain how you pick at least one to experiment with and present the results from those experiments.

Good place to start is here *https://en.wikipedia.org/wiki/Distributed_hash_table*

1 Preparation

I will present and discuss primarily how Chord works in class but you are free to write your own algorithm as long as you can compare it with the existing work (*do not invent the wheel if you don't have to*). Make sure you understand distributed hashing and refer to other materials as needed. I would like to point out that you are allowed to investigate and/or use existing implementations of Can, Chord, Pastry or such DHT algorithms, as long as you cite them properly in your paper of course and use them to produce interesting results.

FYI: There are also interesting ideas in the Dynamo paper

2 Assumptions and Parameters

To simplify the simulation model that you make to collect data you should make the following assumptions:

1. The distributed system is composed of S servers; server i is called S_i.
2. The size of the data collection is E extents (fragments); extent j is called E_j.
3. Data is replicated with a replication factor of N copies.
4. Optimally, each server manages, on average, EN/S extents but part of your task is to define, explain and evaluate the replication policy.
5. The workload is a sequence of random updates to extent E_j.
6. Servers can be added at random, and the overlay structure should adjust.
7. In the basic evaluation, you can assume that operations have no cost; the focus is on the data distribution (the server load).
8. You will need to define some experimental parameters, like how do requests arrive? All at one server, or at random servers and does that matter?

3 Simulation Model

The simulation model is a (set of) data structure(s) and operations designed to keep track of which server handles which extents and to keep track of load distribution. A distinction should be made between the N replicas, as a newly added server must take over a set of extents at each "replication level".

You may need to make choices about how you maintain information about which server is responsible for which extent. When designing your model, consider both the initial setup phase, as well as adding servers.

The parameters S, E, N, W, I and S_m (see below for the latter three) should either be given to the model via input parameters to the program or via a setup file.

If you wish to impress and extend your model in some way, feel free to do so. Simply write about your assumptions and additional experiments in your report.

4 Experiments

The two main research questions of this project are:

a) How well is the random workload distributed between servers?
b) How well does the system scale out?

To measure workload distribution, create a system where $S = 10$, $E = 10{,}000$, and $N = 3$. Then create a workload of $W = 1{,}000{,}000$ random write (update) operations, and measure how many writes each server sees. Analyze this information statistically to understand the distribution.

Note: If you find that the distribution is bad, you should figure out the problem with your implementation and improve your simulation model, then rerun the experiment.

To measure scale out, start with the system and workload above. Then run a loop that i) adds increments of $I = 5$ servers (adjusting data structures) and ii) repeats the workload, until the number of servers is $S_m = 30$. After each step of the loop, run the same analysis as for the previous experiment.

Note: If you find that the distribution becomes bad as servers are added, you should figure out the problem with your implementation and improve your simulation model, then rerun the experiment.

Assuming the 1,000,000 random writes arrive/initiate at a random node you should also pick an overlay network (like Chord) and keep track of statistical information on how many messages are needed until the responsible node is found and the extent can be written to.

Note: To simplify the task at hand you may assume instant and perfect metadata information at every node. For example, if you assume Chord you can assume the finger tables for each node is always up-to-date.

5 Marching orders

You may work alone or in pairs but also feel free to cooperate with your fellow students on the programming / simulation task. Each of you should assume to spend at least 18-25 hours on this project, as it accounts for 15% of your grade. We suggest to use the opportunity to learn about Scala as that will be useful for Project 2 ☺.

6 Deliverables

You should deliver three files in a single .zip file and upload it to Canvas:

1) A short PDF report describing your simulation and the outcome of your experiments.
2) A zipped archive containing the code for your simulation model.
3) A short README.txt file describing the structure of your code.

The report will be the most important deliverable (70% of grade), but we will also consider your code. The report should be self-contained, in that it should contain all the information necessary to understand your motivation, simulation model, experiments and experimental results. The report MUST follow the academic paper "style" having **Abstract + Intro->Background->Method*->Results->Conclusion** as the main sections. *Method does not need to be named Method but is the section that describes your contribution/algorithm etc.