Above the Clouds: CouchDB and Me

Dean Nida
CS 410/510 Cloud and Cluster Data Management

Decisions Made

Document Model

The decision to use the document model was by far the most influential as this impacted our choice in databases and the way we implemented our queries. I personal like the document model as JSON is becoming pervasive in web applications and is easy to work with and human readable.

CouchDB

We selected CouchDB as it uses both the document model and map reduce which we wanted to get experience with. CouchDB is simple with only a handful of features. This made it easy to set up and administer. The time we saved not having to do extensive configuration we used to experiment with different document formats and query options.

Design of Documents

CouchDB uses map reduce and has limited ability to handle dependencies across documents. Joins are implemented by placing a link to the dependent document as a field in the document being queried. Both documents will be returned and the client program is responsible for the join. In order to get our reduce functions to do as much useful work as possible we simply placed the dependent information in each document. This tradeoff meant each record from the freeway loop data also included the station ID and station length. This was done as we parsed the cvs files to JSON.

We also reduced the size of the data set. The original freeway loop data file was 700MB and including all of the overhead from the JSON format would have been well over a gigabyte. This was excessively time consuming and looking at the queries we noticed that a large number of records had detectors that we didn't have associated stations for in the station file. As these records wouldn't have been included in any of the results we parsed them out reducing the size of the dataset to about 100MB without losing records our queries would return.

Iris Couch

Iris Couch is a Platform as a Service that offers instances of CouchDB. We initially planned on using it only as a testing platform to allow our entire team to work on queries at the same time as we waited to get permissions to Amazon EC2, however, we decided to use it as our production database. We found that the amount of configuration required to deploy

CouchDB to EC2 was cutting into our testing and development time and that Iris Couch was robust enough to handle all of our needs. Like our decision to use CouchDB in the first place our Decision to use Iris Couch allowed us to focus on learning map reduce and experimenting with the data model.

Lessons Learned

Thinking in parallel

The nature of cloud computing penalizes dependencies across records as these dependencies require either multiple different map reduce jobs or a client program to resolve them. Initially we tried to use multiple different document types and do joins in our reduce steps and found that this was extremely difficult and was working against the map reduce paradigm. Once we decided to included all of the required data in each document we significantly reduced the complexity of our solutions.

Rereduce

CouchDB has multiple reduce steps. The results of the first reduce are sent to a rereduce. Initially we didn't understand how this worked and continued to get erroneous results. We realized that the results must be calculated at both the reduce step and the rereduce as it is not easy to determine if the rereduce will run.

What happens when you delete your production database on accident.

CouchDB make all the aspects of database administration easy. This includes making it easy to delete an entire database after too many hours of debugging code. Thankfully, CouchDB also makes it easy to replicate a database and keep a backup handy.

Contributions

The complexity of this project meant no one person could work in isolation on a component and each step had to be returned to and revised based on what we learned along the way.

Data parsing

I wrote several python programs to parse the data into JSON files. I had to revise these several times as we had to change the document model after becoming more familiar with map reduce.

Reduces

I wrote most of the reduce functions and had to experiment and test extensively to understand the rereduce functionality.

Graph Traversal

We determined that CouchDB and map reduce would not be able to implement graph traversals. Our initial plan was to simply return the records to the client program and implement it there. After looking at the data we determined that if all of the required records where handled by a single reduce that it might be possible to implement a very basic graph traversal. I wrote this and constructed the required document format.

Distaster Recovery

After several days of writing and testing our queries on a test dataset we where ready to load the full dataset into the database. I wanted to remove the records from the test dataset to reduce duplication. Being tired and not paying attention I dropped the test database forgetting that the queries where just documents in the database I had deleted several days of our work. Lucky, CouchDB's replicator is easy to use and for once in my life I had an up do to date backup and had the database restored in less than five minutes. This was a clear demonstration of CouchDB's resilience and ability to recover from failure.