Building an Enhanced DocGraph Dataset using Mortar (Hadoop) and Neo4J

"The average doctor has likely never heard of Fred Trotter, but he has some provocative ideas about using physician data to change how healthcare gets delivered." This was from a recent Gigaom article. You can read more details about DocGraph from Fred Trotter's post. The basic data set is just three columns: two separate NPI numbers (National Provider Identifier) and a weight which is the shared number of Medicare patients in a 30 day forward window. The data is from calendar year 2011 and contains 49,685,810 relationships between 940,492 different Medicare providers.

You can read some excellent work already being done on this data here courtesy of Janos. Ryan Weald has some great work on visualizing geographic connections between doctors here as well.

The current DocGraph social graph was built in Neo4J. With new enhancements in Neo4J 2.0 (primarily labels), now was a good time to rebuild the social graph, add in data about each doctor, their specialties and their locations. Finally, I've added in some census income data at the zip code level. Researchers could look at economic indicators to see if there are discernable economic patterns in the referrals.

In this series of blog posts, I will attempt to walk through the process in building the Neo4J updated DocGraph using Hadoop followed by the Neo4J batch inserter.

Building the import documents.

One of the goals of the project was to learn Pig in combination with Hadoop to process the large files. I could easily have worked in MySQL or Oracle, but I also wanted an easy way to run jobs on large data sets.

My friends at Mortar have a great platform for leveraging Hadoop, Pig and Python. Mortar is the fastest and easiest way to work with Pig and Python on Hadoop. Mortar's platform is for everything from joining and cleansing large data sets to machine learning and building recommender systems. Mortar makes it easy for developers and data scientists to do powerful work with Hadoop. The main advantages of Mortar are:

- Zero Setup Time: Mortar takes only minutes to set up (or no time at all on the web), and you can start running Pig jobs immediately. No need for painful installation or configuration.
- Powerful Tooling: Mortar provides a rich suite of tools to aid in Pig development, including the ability to Illustrate a script before running it, and an extremely fast and free local development mode.
- Elastic Clusters: We spin up Hadoop clusters as you need them, so you don't have to

predict your needs in advance, and you don't pay for machines you don't use.
- Solid Support: Whether the issue is in your script or in Hadoop, we'll help you figure out a solution.

Data Sets

One great thing about this data is that you can combine the DocGraph data with with other data sets. For example, we can combine NPPES data with the DocGraph data. The NPPES is the federal registry for NPI numbers and associated provider information.

To create the data sets for ingest into Neo4J, we are going to combine Census Data, DocGraph Data, NPEES database and the National Uniform Claim Committee (NUCC) provider taxonomy codes.

Pig Scripts
Using Pig scripts, I was able to create several data files that could then be loaded into Neo4J.

Running the Pig Code in Mortar
In the next post, we will look at using Mortar's framework to run the Pig jobs.