Analyzing HHS Data with Mortar

I'm starting a new series on analyzing publicly available large data sets using Mortar. In the series, I will walk through the steps of obtaining the data sets, writing some Pig scripts to join and massage the data sets, adding in some UDFs that perform statistical functions on the data and then plotting those results to see what the data shows us.

Recently, the HHS released a new, improved update of the DocGraph Edge data set. As the DocGraph.org website says, "This is not just an update, but a dramatic improvement in what data is available." The improvement in the data set is that we know how many patients are in the patient sharing relationship as shown in the new data structure:

FirstNPI, SecondNPI, SharedTransactionCount, PatientTotal, SameDayTotal.

Again, from the DocGraph.org website, "The PatientTotal field is the total number of the patients involved in a treatment event (a healthcare transaction), which means that you can now tell the difference between high transaction providers (lots of transactions on few patients) and high patient flow providers (a few transactions each but on lots of patients)".

Data Sets
HHS released "data windows" for 30 days, 60 days, 90 days, 180 days and 365 days. The time period is between 2012 and the middle of 2013. The number of edges or relationships is as follows:

Window	Edge Count
30 day	73 Million Edges
60 day	93 Million Edges
90 day	107 Million Edges
180 day	132 Million Edges
365 day	154 Million Edges

NPPES
The Administrative Simplification provisions of the Health Insurance Portability and Accountability Act of 1996 (HIPAA) mandated the adoption of standard unique identifiers for health care providers and health plans. The purpose of these provisions is to improve the efficiency and effectiveness of the electronic transmission of health information. The Centers for Medicare & Medicaid Services (CMS) has developed the National Plan and Provider Enumeration System (NPPES) to assign these unique identifiers.

NUCC Taxonomy Data
The Health Care Provider Taxonomy code set is an external, nonmedical data code set designed for use in an electronic environment, specifically within the ASC X12N Health Care transactions. This includes the transactions mandated under HIPAA.

The Health Care Provider Taxonomy code is a unique alphanumeric code, ten characters in length. The code set is structured into three distinct "Levels" including Provider Type, Classification, and Area of Specialization.

The National Uniform Claim Committee (NUCC) is presently maintaining the code set. It is used in transactions specified in HIPAA and the National Provider Identifier (NPI) application for enumeration. Effective 2001, the NUCC took over the administration of the code set. Ongoing duties, including processing taxonomy code requests and maintenance of the code set, fall under the NUCC Code Subcommittee.

Why Do I run Mortar
I use Mortar primarily for four reasons:

1. I can run Hadoop jobs on a large set that I can't run on my laptop. For example, my MBP laptop with 8GB ram and an SSD can work with 73M records. However, when I start joining records and running large number of map reduce jobs, I am going to run out of temporary space on the hard drive. I can't fully analyze the data sets.

2. It is easy to develop on my laptop and then deploy to a large Amazon cluster with a single command. The framework gives me Pig and Hadoop on my laptop with no configuration, version control through Github and the 1-button deployment. Mortar integrates easily with Amazon S3 so I can store these large data files in an S3 bucket and not worry about running out of space.

3. For debugging Pig scripts, Mortar has incorporated lipstick. Lipstick shows what the pigscript is doing in real time, provides samples of data and metrics along the way to help debug any issues that arise.

4. Cost. Using Amazon spot instances, I can process the data on a large cluster for less than the cost of a grande cup of coffee. Check out my invoice.

I ran an eight-node cluster for over an hour to process 73M records creating quartiles across three groups for $1.12. You can't beat that.

For more details on the Mortar framework, check out their website, their help pages and their github examples page.

Data Processing
The rest of the post will discuss processing the data sets. The HHS data sets are CSV files that have been zipped and stored on the docgraph.org site. In order to get information about the provider, I used the most current NPI data set (also known as the NPPES Downloadable File). This file contains information about the provider based on the National Provider Index (NPI). It is also a CSV file. Finally for the taxonomy, I used the Health Care Provider Taxonomy code set.

Each of these data sets are CSV files so there isn't any data modification/transformation needed. I downloaded the files, unzipped them and uploaded them to an Amazon S3 bucket.

The desired data structure was to create a single row consisting of the following data elements:

referringHealthCareProviderNPI,
referringHealthCareProviderState,
referringHealthCareProviderTaxonomy,
referringHealthCareProviderOrgName,
referringHealthCareProviderDocName,
referringHealthCareProviderNUCCCode,
referringHealthCareProviderNUCCType,
referredToHealthCareProviderNPI,
referredToHealthCareProviderState,
referredToHealthCareProviderTaxonomy,
referredToHealthCareProviderOrgName,
referredToHealthCareProviderDocName,
referredToHealthCareProviderNUCCCode,
referredToHealthCareProviderNUCCType,
sharedTransactionCount,
patientTotal,
sameDayTotal

This data structure allows me to perform various groupings of the data and derive statistical measures on the data.

The Pig code to create the structure is shown in the following gist.

Using Pig Functions and a UDF
For this blog post, I wanted to take a look at creating quartiles of different groups of data. I wanted to group the data by referring state and taxonomy, referred to state and taxonomy and finally referring taxonomy and referred to taxonomy.

For the median and quantiles of the data, we will use the Apache DataFu library. DataFu is a collection of Pig algorithms released by LinkedIn. The getting started page has a link to the JAR file which needs to be downloaded and registered with Pig. The statistics page shows us how we can use the median and the quantiles function. Both functions operate on a bag which we easily create using the Group function.

Once registered, we calculate the statistics as follows:

At this point, the code is ready to go and we can give it a run.

Running in local mode
One of the great things about Mortar is that I can run my project locally on a smaller data set to verify that I'm getting the results I'm expecting. In this case, I added a filter to the DocGraph data file and filtered out all records where the SameDayTotal was less than 5000. This allowed me to run the job locally using this command:

mortar local:run ./pigscripts/localdocGraphLargeSameDayTotal.pig -g 0.12

This job reads the data from my local machine, runs in about 15 minutes and writes out the results to my local machine.

Running on the full dataset
When I am ready to run this on the full data set, I can simply launch it as follows:

mortar jobs:run ./pigscripts/localdocGraphLargeSameDayTotal.pig -g 0.12 --clustersize 5

While running, I can use Mortar's lipstick to visualize the running job as shown below:

Once the job completed after about an hour, the output format looks like this. From here, we can take this data and make some box plots to look at the data.

Closing Thoughts
Leveraging frameworks like the Amazon and Mortar allows someone like myself to perform large scale data manipulation at low cost. It allows me to be more agile and able to manipulate data in various ways to meet the need and provides the beginnings of self-service business intelligence.