Phillip Thompkins
CMSC476 Spring 2014

HW5 Writeup
Document Clustering

Implementation

- *Overarching Implementation:* The code from Homework 4, particularly to handle the documents on the hash table, was kept; the same algorithm was even used. This time, instead of using words as the keys to the corpus, I used the titles of the documents.
- *Similarity Matrix:* My similarity matrices were managed by creating triangular matrices, meaning half of the matrix, including the diagonals, would be filled with zeroes, while the other half would be filled with useful information. The matrix started as an N x N matrix of nil values, and diagonal locations were ignored to prevent the program from comparing a document to itself. Cosine similarities were found by finding the words each document had in common, and multiplying the frequencies of each word before adding all of those products together. The three relevant sums, (overall, rows, and columns) were kept separate until the end, where the total sum was divided by the products of the square roots of the row and column sums.
- *Cluster Naming:* Clusters were named by concatening string names via the asterisk "*" character. For example, 343.html and 349.html would cluster together and be called 343.html*349.html, which allowed me to clearly see which documents were clustering together. However, as the clusters got larger and larger, to the point where pretty much everything was clustered together, the filenames got excessively long. I could have maybe clustered documents by the names of the most common terms or something, but that wouldn't have told me much about what was in the cluster, since "dog" could be about canines, hot dogs, or even the Elvis Presley song.

- *Other Data Structures:* Similar to Homework 4, a hash table was used to hold all relevant information for the individual documents, and the same algorithm was used.

Results

- *Most Similar Documents*: It was found that documents 349.html and 351.html were the most similar, with a score of 158.504. This makes sense, since pulling the both of them up shows that they're both similarly formatted, in the same language, and seem to start almost identically.
- *Least Similar Documents*: It was found that documents 001.html and 003.html were some of the least similar, with a score of 0, showing no cosine similarity. This makes a degree of sense, since 001.html is a political news article, while 003.html is a sparse webpage, looking like it's from a tech help forum. I expected the least similar documents to be of different languages, but I guess not.
- *Closest to Centroid*: The documents closest to the centroid (3.6779) were 312.html and 316.html, with scores of 3.6776.
- *Run-Time*: The program took a long time, nearly ten minutes, to run to completion, likely due to the fact that the triangular matrix was continually recalculated. This could have been prevented by computing the triangular matrix only once, and replacing values as things were calculated and clustered.