Project 1

CSC316 Section 601

2/6/15

Siddhartha Kollipara

Data Analysis

This experiment involved comparing three distributions of keys versus three different implementations of the Dictionary ADT. The data of the implementations are graphed based on the key distributions and compares three search strategies: move-to-front (MTF), transpose, and binary search.

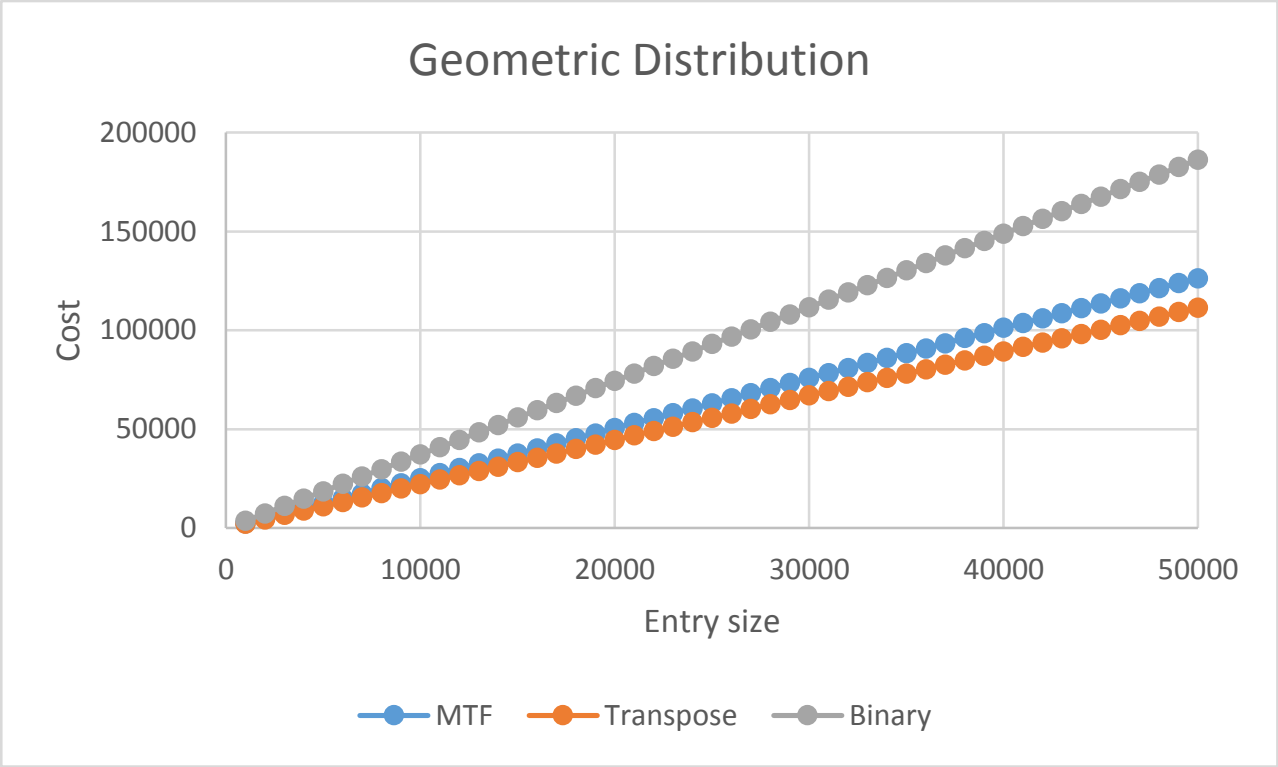

Figure 1. MTF cost: 126334, Transpose cost: 111411, Binary search: 186279, Optimal cost: 100000

As seen above, the geometric distribution exhibits a low cost variation between search strategies. All three search functions are close in relative cost. In this case transpose is the most efficient and is a bit higher than the optimal cost of 100000. MTF is next with a slightly higher cost than transpose and has approximately a 25% increase in cost over the optimal cost. Binary search is this case is the worst and is almost double the optimal cost. This is because rather than use a dictionary list sorted by frequency, it needs the dictionary sorted from the smallest value to largest.

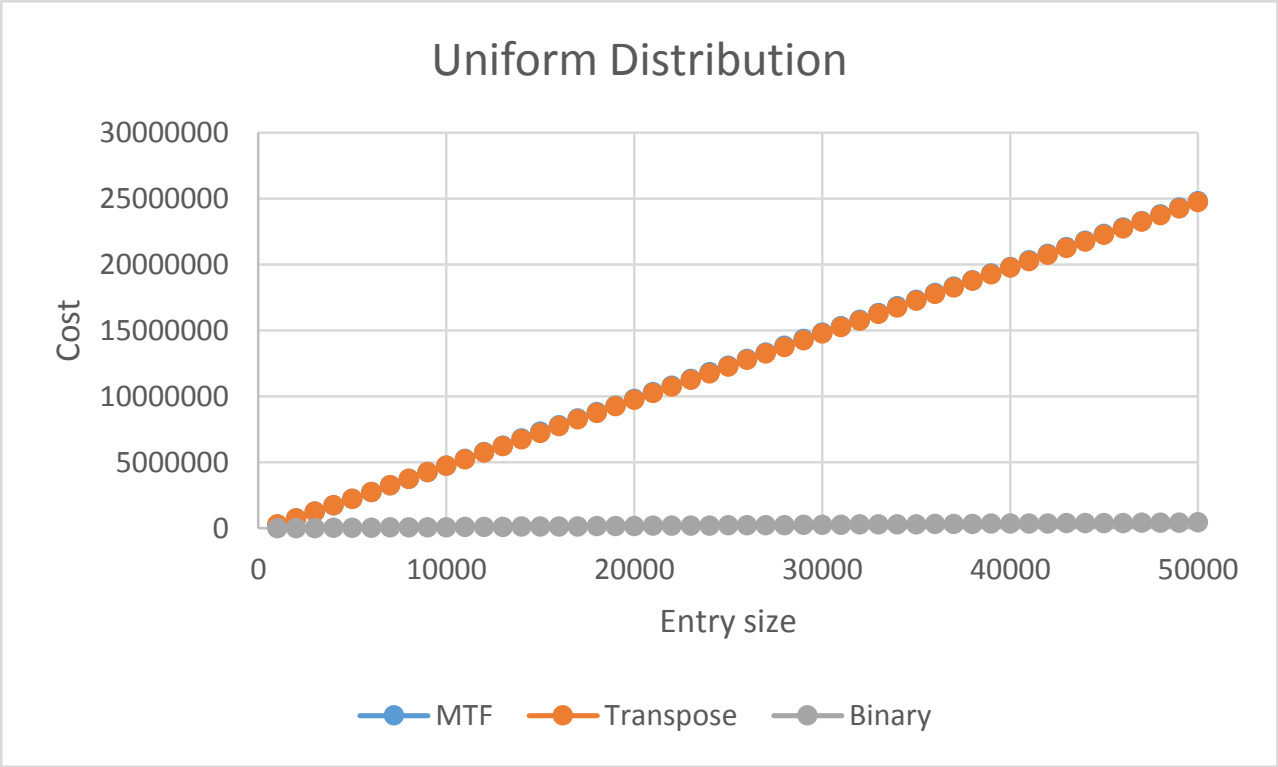

Figure 2. MTF cost: 24823163, Transpose: 24745259, Binary search: 448782, Optimal cost: 25025000

The uniform distribution requires a much higher cost to implement search functions. The most efficient implementation by far is binary search, its cost is a small fraction of the optimal cost. The cost of MTF and transpose are relatively the same and are ever so slightly lower than the optimal cost but are magnitudes higher than binary search.

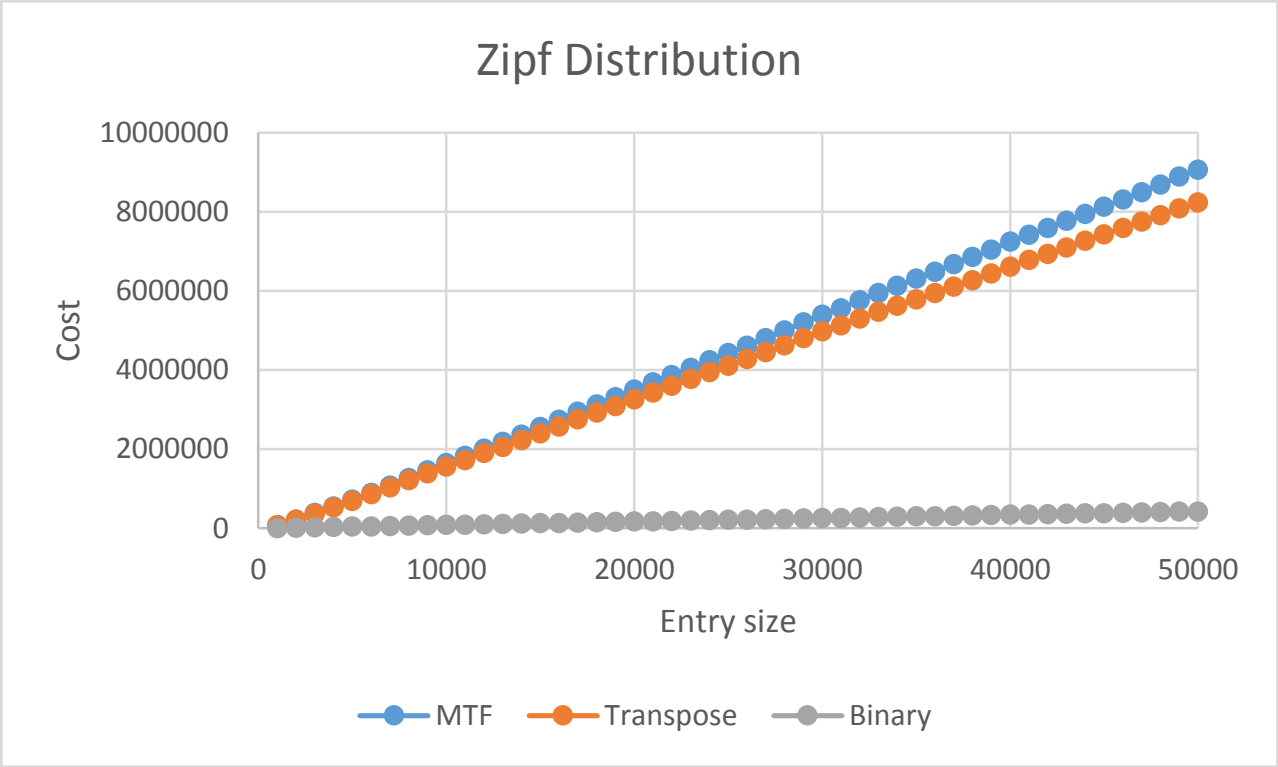

Figure 3. MTF cost: 9070405, Transpose: 8240470, Binary search: 428140, Optimal cost: 6679599.7

The Zipf distribution was best handled by the binary search, which had a significantly lower cost than the other two search functions. Transpose was the next best function but had a cost that was a bit larger than the optimal cost. MTF was the worst have the largest cost and the largest difference in cost from the optimal cost.

The distributions had very significant costs on the three Dictionary ADT implementations with geometric having the lowest cost and uniform having the largest. It makes sense that geometric would be the lowest and Zipf the second lowest since MTF and transpose are based around access frequency. Uniform has the largest cost for MTF and transpose because it is not based on having some keys more frequent than others meaning that the way the dictionary is sorted does not matter. This is the same reason why the optimal cost is basically equal to their cost since ordering does not matter for them. The optimal cost for geometric and Zipf matter more though because they are based on some keys being a lot more frequent than others. This is why the cost for MTF and transpose are greater in these scenarios.

Binary search differs from the other two methods due to its dictionary keys being sorted from least to greatest. Its efficiency is predicated mostly on the size of the dictionary rather than the distribution it is being implemented on. One could argue that the size of the dictionary is based on the distribution, which is true. Since the frequency of a key does not matter to binary search at all a radically low cost is seen for Zipf and uniform distributions compared to the other two implementations. Its cost compared to the optimal cost is not a very relevant stat due to the fact that binary search has its own way of sorting the dictionary and does not care for sorting the keys by frequency at all.

Overall, each of these search functions are useful for different scenarios. Based on the results, transpose seems to be a slightly better search strategy than MTF and both should be used with distributions that have a very high locality of reference. Binary search becomes more significant as locality of reference is less apparent. This is seen with the cost for the Zipf and uniform distributions being much lower than the optimal cost.