Lab Assignment 5

ISMAEL VILLALOBOS

PROFESSOR FUENTES

TA: DITA NATH

11-6-19

Introduction

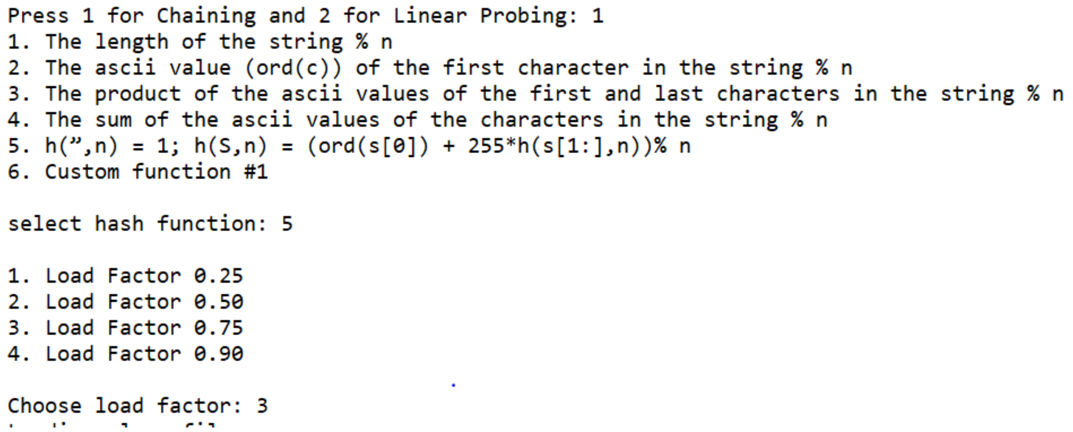
For Lab5, we were given the task to implement the same concept from the previous lab which was Natural Language Processing and finding similarities between word vectors. This time we used Hash Tables with Chaining as well as with Linear Probing, aside from using both types of hashing we used 6 different types of hash functions which included:

- The length of the string % n
- The ascii value (ord(c)) of the first character in the string % n
- The product of the ascii values of the first and last characters in the string % n
- The sum of the ascii values of the characters in the string % n
- The recursive formulation h(”,n) = 1; h(S,n) = (ord(s[0]) + 255*h(s[1:],n))% n
- Another function of your choice

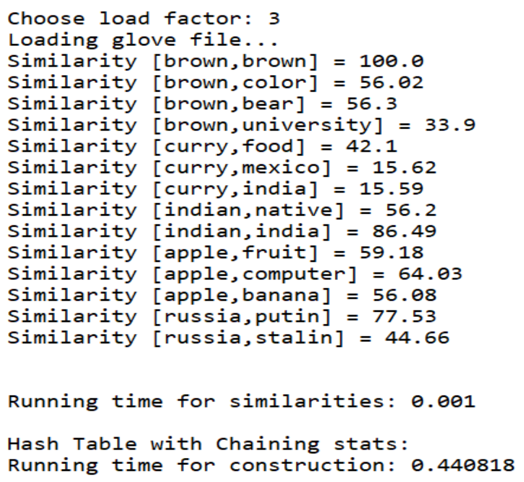
Proposed Solutions

Similar to the approach used in Lab 4 we begin by opening the Glove File provided. When reading the file, I strip each line into a word and an embedding that will be inserted in the corresponding bucket in the hash table. In order to find the appropriate position in the hash table we must implement the functions that were provided in order to find the hash value. In addition to the hash functions as options, I decided to give the user options as to the load factor for the table in order to test different table sizes.

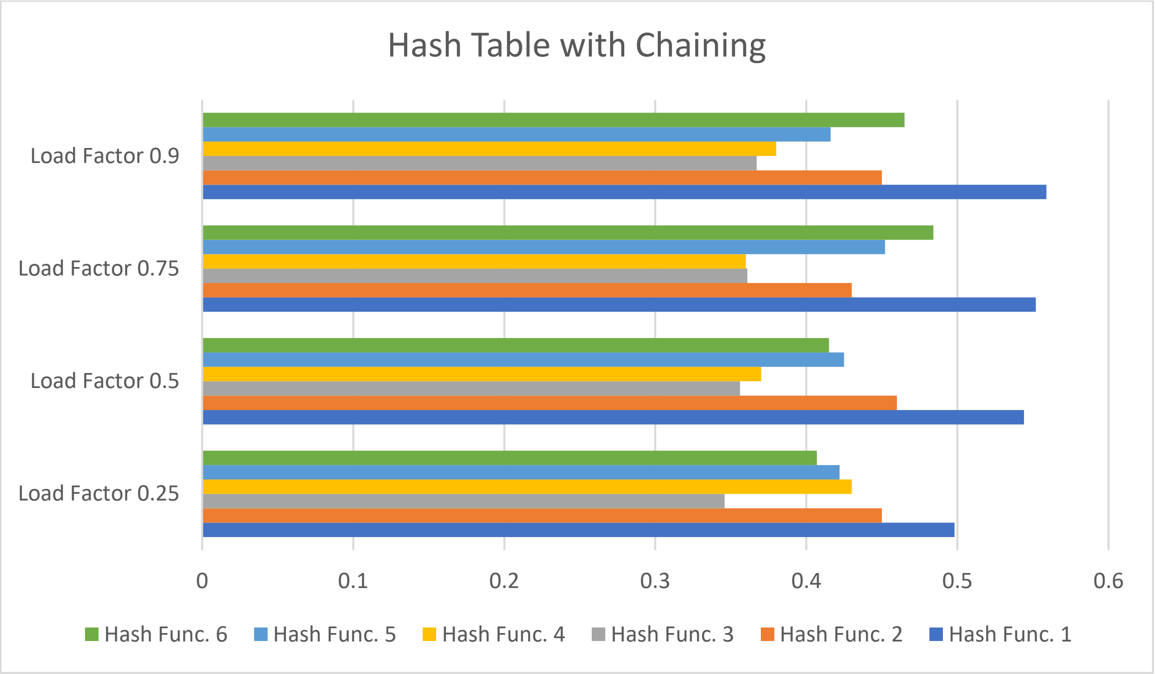
Experimental Results

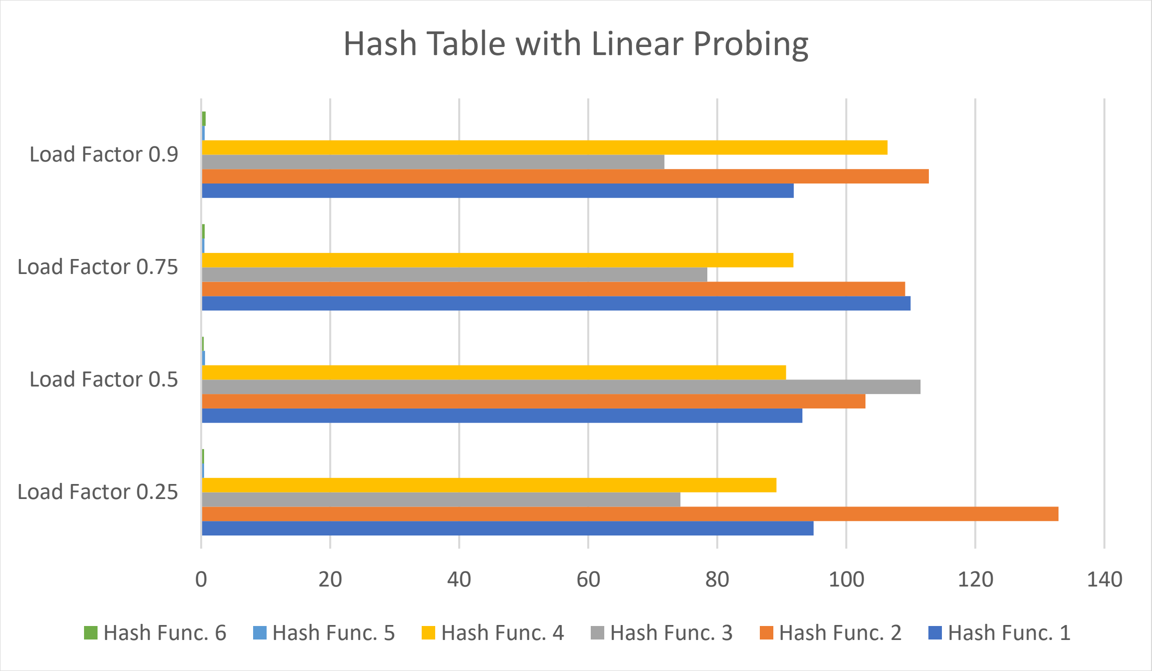


```
Press 1 for Chaining and 2 for Linear Probing: 1
1. The length of the string % n
2. The ascii value (ord(c)) of the first character in the string % n
3. The product of the ascii values of the first and last characters in the string % n
4. The sum of the ascii values of the characters in the string % n
5. h(”,n) = 1; h(S,n) = (ord(s[0]) + 255*h(s[1:],n))% n
6. Custom function #1

select hash function: 5

1. Load Factor 0.25
2. Load Factor 0.50
3. Load Factor 0.75
4. Load Factor 0.90

Choose load factor: 3
```



```
Choose load factor: 3
Loading glove file...
Similarity [brown,brown] = 100.0
Similarity [brown,color] = 56.02
Similarity [brown,bear] = 56.3
Similarity [brown,university] = 33.9
Similarity [curry,food] = 42.1
Similarity [curry,mexico] = 15.62
Similarity [curry,india] = 15.59
Similarity [indian,native] = 56.2
Similarity [indian,india] = 86.49
Similarity [apple,fruit] = 59.18
Similarity [apple,computer] = 64.03
Similarity [apple,banana] = 56.08
Similarity [russia,putin] = 77.53
Similarity [russia,stalin] = 44.66


Running time for similarities: 0.001

Hash Table with Chaining stats:
Running time for construction: 0.440818
```





Big O Notation & Runtimes

The hash table with chaining was so much faster than both BST and B-Tree in terms of speed and efficiency. However for the search function the table with chaining is faster than a B-Tree but not faster than a BST.

Hash operations usually have O(1) time with at worst being O(n)

Tree operations are O(logn)

Conclusion

With my findings I found that hash tables are way faster than any tree structures and will take that into account for future projects. This really opened my eyes to the efficiency of the algorithms and their use for Natural Language Processing.

Academic Honesty

"I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class."

Name: Ismael Villalobos

Appendix: