Ian Riley
04/22/16
CS 2123 - Girgis

Homework 3 Results

This report is an analysis of three separate hash table implementations. The three implementations are that of separate chaining, quadratic probing, and double hashing. All three implementations were written in the Go programming language. The hash tables were used to store single and paired words from five separate input texts. The input texts are *Huckleberry Finn*, *Much Ado About Nothing*, *Tom Sawyer*, *Innocent Adventuress*, and *Prodigal Village*. The sizes of the texts range from 10^4 to 10^5 in magnitude. The rough estimate for the speed of the executing processor is 10^9 operations per second. The executing machine is a MacBook Pro with a 2.7 GHz Intel i-5 quad-core processor with 8 GB of 1867 MHz DDR3 memory.

According to the results that were gathered, separate chaining had the smallest average time. This was expected since, compared to the other implementations, separate chaining has the fewest collisions and requires the least amount of hash functions to be computed. Unlike the other implementations, which compute a new hash every time a collision occurs, separate chaining only ever computes one hash per key. This is very important since strings are actually expensive to compute a hash for. This is also why double hashing had the largest average time. Double hashing uses two expensive hash functions to compute the hashes for each string.

Book title	*Much Ado About Nothing*	*Prodigal Village*	*Innocent Adventuress*	*Tom Sawyer*	*Huckleberry Finn*	Average Time
Size	26247	31183	42634	76118	116954	---
Separate Chaining	2.150s	3.407s	6.349s	17.702s	27.615s	11.445s
Quadratic Probing	3.600s	6.056s	10.990s	30.372s	44.843s	19.172s
Double Hashing	3.877s	6.148s	11.372s	32.298s	48.029s	20.345s