MP7 Performance Evaluation

Heath Gerrald

Running the retrieve driver with random addresses, I tested all three increment types and measured the time to run the driver. My results are shown below, note that times are given in milliseconds.

Load Factor	Linear (M=65537)	Double (M=65537)	Quadratic (M=65536)
0.5	17658.234	17152.036	17577.154
0.75	22082.303	22201.753	22050.997
0.9	26099.678	25770.972	25534.695
0.95	26135.105	25885.371	25879.455
0.99	28019.861	26685.14	26698.286

My data shows that the probing type makes little difference when using random addresses. Increasing the load factor increases run time since the table will be rehashed less often and searches will take longer.

I performed another test running the retrieve driver but this time changed the building schemes. I used a load factor of 0.85 and found the average probes required for both a successful and unsuccessful search for each decrement type.

	Random	Sequential	Folded	Worst
Quadratic (M=65536)	2.43416	1	118.761	11651.8
Double (M=65537)	2.22411	1	2.1809	69.3333
Linear (M=65537)	3.84966	1	13927.5	28414.7

Average number of probes for successful searches

	Random	Sequential	Folded	Worst
Quadratic (M=65536)	7.88857	190.368	190.268	7.80473
Double (M=65537)	6.67292	17.5397	17.0941	11.6917
Linear (M=65537)	22.5881	23669.2	23727.7	23723.7

Average number of probes for unsuccessful searches

My data supports the claim that this hash table is closer to O(1) complexity class as opposed to O(n) for every build case except worst with linear probing. Sequential builds are always O(1) for successful searches for each decrement type.

My last testing was to show that performance degrades when the table size is large and the number of trials is large. For example, with linear probing and a table size of 65537, I found that the percent of empty locations marked deleted with 50,000 trials was 88.187%. This number increased to 98.2094% when the number of trials increased to 100,000. Additionally, the number of probes required for searches decreased drastically after rehash, showing that performance increases when there are less entries in the table.