563

James Ballard

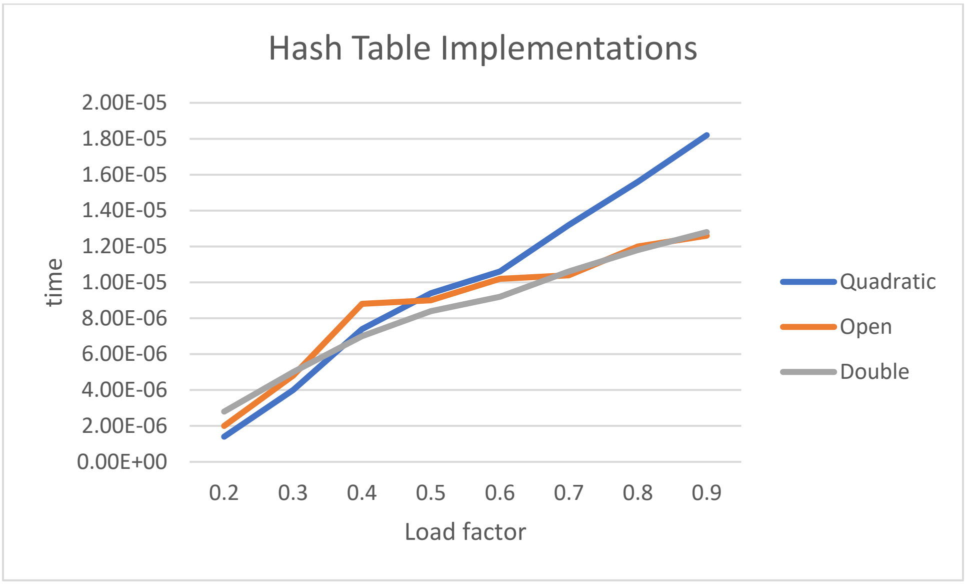
Hash table lab report

The purpose was to test 3 different types hash tables double hashing, quadratic probing, and open hashing and see how the insertion time changes with as more data is loaded in the hash table. To do this we set a couple of variables the first and most important was the size of each hash table had to be constant which we set as the large prime number 600,011 since the numbers inserted would range from 1 to 2^(21) we also set p of the double hash equal to 5. The variables that no change to prove this experiment was the amount of elements add to the hash table, and what these elements would be. For the amount of elements insert to the array we set K=60,000 and made an array of size K*loadfactor (where load factor was looped from .2:.1:.9) the array was then populated with random nums. In order for the array to account for the variance provide by random nums, we sent 5 unique sets of random in to the array by modify srand() (0,1,2,3,4) and then filling the array. Once we had populated the array with the set of random numbers using the timer class we measured the time it took for all numbers in the array to be insert in to the respective hash table after which we output the time and emptied the hash table we procced to do this for all three hash tables. We outputted the result ./hash_table_test >>data.txt which gave use the following results.

load factor	srand	0	1	2	3	4	average
0.2	quadtric	2.00E-06	1.00E-06	2.00E-06	1.00E-06	0.00E+00	1.20E-06
	open	1.00E-06	2.00E-06	2.00E-06	2.00E-06	2.00E-06	1.80E-06
	double	2.00E-06	2.00E-06	1.00E-06	2.00E-06	2.00E-06	1.80E-06
0.3	quadtric	3.00E-06	3.00E-06	3.00E-06	3.00E-06	3.00E-06	3.00E-06
	open	3.00E-06	3.00E-06	4.00E-06	4.00E-06	4.00E-06	3.60E-06
	double	3.00E-06	4.00E-06	3.00E-06	4.00E-06	4.00E-06	3.60E-06
0.4	quadtric	4.00E-06	4.00E-06	1.40E-05	5.00E-06	1.40E-05	8.20E-06
	open	1.70E-05	5.00E-06	5.00E-06	6.00E-06	5.00E-06	7.60E-06
	double	5.00E-06	5.00E-06	6.00E-06	6.00E-06	5.00E-06	5.40E-06
0.5	quadtric	5.00E-06	5.00E-06	5.00E-06	1.50E-05	1.60E-05	9.20E-06
	open	5.00E-06	6.00E-06	7.00E-06	7.00E-06	7.00E-06	6.40E-06
	double	7.00E-06	6.00E-06	6.00E-06	6.00E-06	7.00E-06	6.40E-06
0.6	quadtric	3.30E-05	6.00E-06	1.60E-05	1.70E-05	1.70E-05	1.78E-05
	open	1.70E-05	1.70E-05	8.00E-06	7.00E-06	7.00E-06	1.12E-05
	double	7.00E-06	7.00E-06	7.00E-06	7.00E-06	7.00E-06	7.00E-06
0.7	quadtric	7.00E-06	1.70E-05	1.90E-05	1.80E-05	1.90E-05	1.60E-05
	open	9.00E-06	8.00E-06	9.00E-06	9.00E-06	9.00E-06	8.80E-06
	double	8.00E-06	7.00E-06	8.00E-06	7.00E-06	7.00E-06	7.40E-06
0.8	quadtric	8.00E-06	8.00E-06	8.00E-06	9.00E-06	1.00E-05	8.60E-06
	open	9.00E-06	9.00E-06	1.70E-05	9.00E-06	1.80E-05	1.24E-05
	double	8.00E-06	9.00E-06	8.00E-06	9.00E-06	8.00E-06	8.40E-06

0.9	quadtric	1.00E-05	1.90E-05	2.20E-05	2.10E-05	2.10E-05	1.86E-05
	open	1.00E-05	9.00E-06	1.00E-05	1.00E-05	1.10E-05	1.00E-05
	double	1.00E-05	9.00E-06	1.00E-05	9.00E-06	1.00E-05	9.60E-06

Then I plotted the load factor to the averages of the 3 hash types



So from this it seems the Quadtric is the least affective in terms of inserting numbers as N increases and it looks like double and open hashing functions operate the same for large n but Double is better for small N. note the data was originally stored in data2.txt