Empirical Verification of Probing Analysis for Open and Closed Hashing Algorithms

Alexander Austin
College of Charleston
66 George Street
Charleston, South Carolina, 29412
austinag@g.cofc.edu

ABSTRACT

This paper presents an analysis of average Search times (number of probes) for both Open and closed hashing (utilizing quadratic probing) functions, and thereafter provides empirical evidence gained through testing that serves to prove the hypothesis that the analysis Of the algorithms provides an accurate estimate for The amount of probes required to find an object.

INTRODUCTION

The formula used for the Open Hashing hypothesis for *S* successful searches and *U* unsuccessful searches was defined by A. Levitin[1] and presents itself as follows: $S \approx 1 + \alpha\ 2$ and $U = \alpha$, where α is equivalent to the load factor of the hash table (# of elements / # of spaces in table). The formula used for the closed hashing hypothesis was defined by D. Knuth[2] as $S = 1 - \ln(1-\alpha) - \alpha/2$ and $U = 1/(1-\alpha) - \ln(1-\alpha) - \alpha$. This paper and its experimentation set out to prove that the preceding formulas are accurate to a degree in estimating the number of probes required for searches in both hashing types. However, Knuth's formulas cannot be calculated for a load factor of one, so this paper also sets out to prove the hypothesis that for failed searches on closed hashing tables with a load factor of one, the number of probes will be equivalent to the size of the table.

METHODS

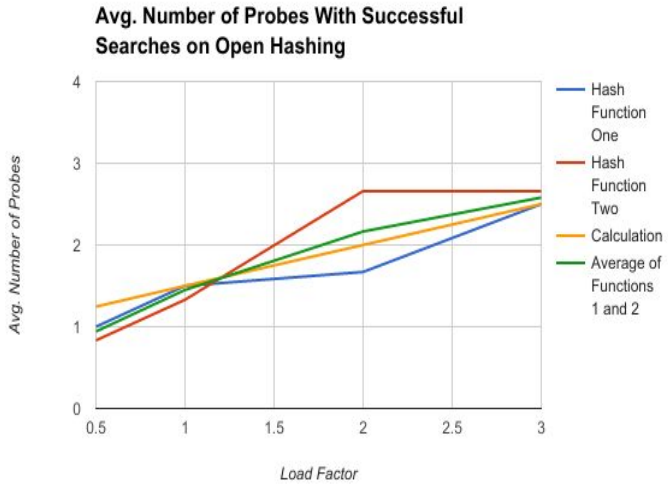
To test the hypothesis, a class for open hashing and a class for closed hashing were developed. Each utilized an array of size 1009, a prime number, in order to aid in preventing clustering. The classes were designed specifically to hash String type data values. Each of the classes contained two separate hash functions, one of which was written by the author for the purpose of the experiment (Function 1), and another that was defined by Anany Levitin in his book *Design and Analysis of Algorithms*[1] (Function 2). Once these two hashing algorithms were designed, and the classes properly defined, another class was created for testing in which sixteen tests were run, where each test ran six successful searches and six unsuccessful searches, calculated the number of probes and time taken to do each search, and an average was calculated for each.. Tests 1 through 4 were performed with open hashing utilizing function 1 with increasing load factors of ~.50 (.4956), 1, 2, and 3 respectively. Tests 5 through 8 worked similarly, except they utilized function 2. Tests 9 through 12 were performed with closed hashing with quadratic probing, and utilized hash function 1 with increasing load factors of ~.25 (.2478), ~.50 (.4956), ~.75 (.7433) and 1. The final four tests were performed in a similar manner, but used hash function two. For all tests, it was also

ensured that the successful searches were at least somewhat evenly dispersed through the table (that is, an even mix of elements that were inserted into the table earlier, and elements that were inserted later) in order to achieve averages that represented the majority of possible searches.

RESULTS

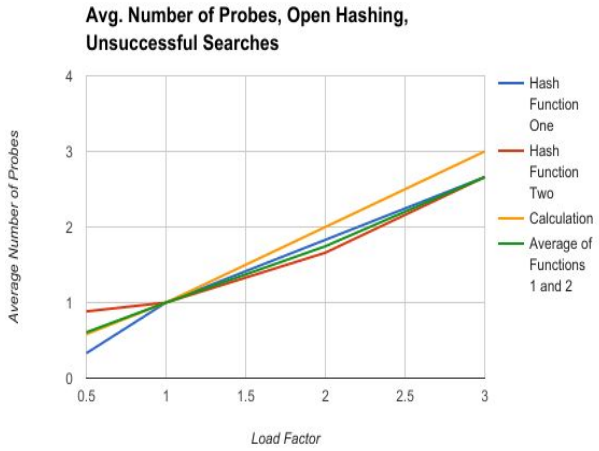
-Open Hashing

On tests of open hashing, the empirical data gathered does support the hypothesis, however there are slight deviations in the data as opposed to the calculation. For successful searches, the search on the ~.5 load factor showed that both functions were slightly lower than the predicted value. However, when the load factor goes up to one, the data values converge. When the load factor increases to two, the two tested functions diverge, one above, and one below the calculated trajectory. However, it should be noted that the average of the values of the two hash functions remains very closely in line with the calculated data. After that, the two hash functions and the calculated data begin to reconverge together, growing closer and closer with values that are only a fraction apart, as can be seen in this graph of the data.



For unsuccessful searches, the data remains more in line: keeping a constant relation to the calculated data, but is off by a small factor. The data initially starts like the data for successful searches, as they are slightly scattered, but converge when the load factor is equal to one. After that however, the data values stay slightly below the calculated prediction, but the relation appears to be constant, so it may be inferred that the testing results correlated to the predicted value, but were off by a small constant, as can be seen in the graph below.

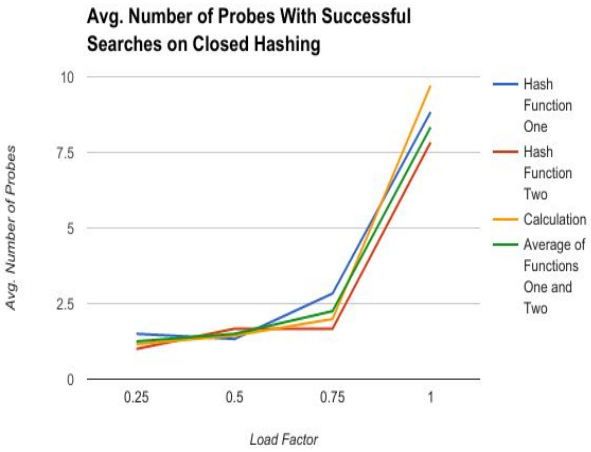
For time analysis of the open hashing, which



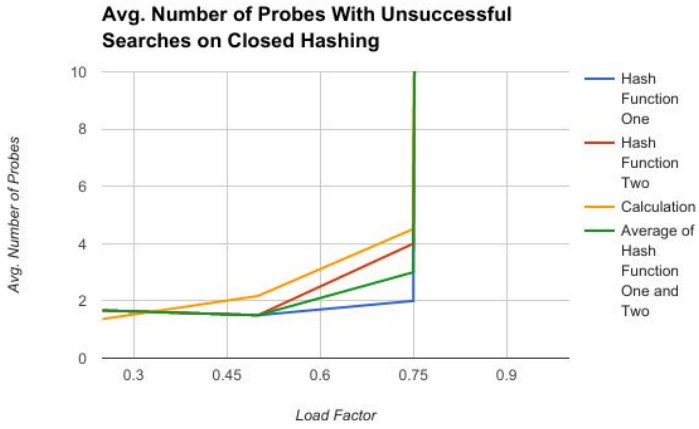
was performed using Java's nanoTime function, it was observed that often the first search of each test took far longer than the rest (between 16,000 and 22,000), but after that, there seemed to be a clear correlation between time and number of probes, excepting for some anomalies. On average, it appeared that each additional probe added an additional 800-1000 nanoseconds in time. Note however, that the nanoTime may not be precisely accurate, due to the fact that Java's clock does not count in nanoseconds.

-Closed Hashing

Tests on Closed Hashing revealed that the empirical data did support the hypothesis, but once again, similar deviations for the data show up. For successful searches, the data for the most part remained fairly consistent with the calculations, however, the functions showed similar patterns of deviation found in the open hashing, but on a smaller scale. Aside from that, and being off by a small degree at times, the empirical data follows nearly the precise trend as the calculated data, as can be seen on the graph below.



It must be noted however, that the formula cannot be used for load factors of value 1, therefore a close decimal (.99999) was used, so the calculated value may be slightly off, but nonetheless follows the pattern.
The unsuccessful searches of closed hashing yielded the largest degree of divergence from the calculated data, as both hash functions yielded an average number of probes that was below the calculated average. However, one can observe that the higher the load factor (once ~.75 is reached) the more that the data points converge, and when load factor is 1, both calculated data and empirical data are uniform, as can be seen in the chart of the data.



Timing averages also displayed a similar pattern to the open hashing tests, however, the overall time taken was typically far shorter (on average 1000 ns) except for unsuccessful searches on tables of load factor one, where the time was quite high (on average 150,000 ns)

CONCLUSION

From the empirical data gathered through experimentation, it appears that the formulas used to predict the number of probes for searches are accurate in mapping the outline of the data. Considering that in all tests, both hash functions displayed similar patterns of variation (one hash function would go above predicted value, while another would go below, and they would ultimately converge) , it may be predicted that the variation from the calculated data arises from the hash functions which were utilized, and furthermore, that if better hash functions were utilized, that the end data would be more in alignment with the predicted data. Ultimately, it appears that the hypothesis was accurate to a degree, and properly predicted general trends, but there are discrepancies between the predicted and empirical data that cannot be ignored.

ACKNOWLEDGEMENTS

Thanks go out to Dr. Anthony Leclerc in his assistance in debugging the code used for testing.

REFERENCES

1. Levitin, A. *Introduction to the Design and Analysis of Algorithms, 3rd ed.* Addison-Wesley (2012).
2. D.E. Knuth.*The Art of Computer Programming: Sorting and Searching* (second ed.), vol. 3, Addison-Wesley (1998)