CS 1501 Assignment 2: Hashing Schemes

By: Andrew Beers

Email: [acb75@pitt.edu](mailto:acb75@pitt.edu)

Scheme Comparison

All results for this assignment were run multiple times for each alpha value, this includes both probing and timing results, it is possible due to the size of data collected that some patterns were not recognized, this is what I observed with the results collected.

According to the timing results of linear probing, separate chaining, and double hashing, it appears that separate chaining is the fastest of the three hashing schemes. This is based on the growth of time as the table filled. As the table had less and less empty spaces, the timing both double hashing and linear probing increased drastically, which separate chaining remained fairly quickly.

When calculating the number of probes taken to find a value successfully, separate chaining is the scheme that uses the fewest probes, while linear probing uses the most. Conversely, when calculating which hashing scheme used the fewest probes when unsuccessfully finding a value, separate chaining was the most efficient, while linear probing was the least, using the most probes.

Difference in Successful and Unsuccessful Search

It is important to consider the difference between a successful and unsuccessful search, particular to this example, there were many unsuccessful searches, so when timing analysis comes into play, the hashing schemes should be tested. According to the results, all three hashing schemes were similar for lower alpha values, when the table was less filled. However, when the table filled up, the find time for linear probing increased, this is because for an unsuccessful find in a mostly filled table, there are many clusters to deal with. Double hashing ran into this same problem, but to a lesser degree. Once again separate chaining performed the best of the three schemes, this is because when searching for a value that has not been hashed yet, an unsuccessful search will only take one probe, and less time. This is different from linear probing and double hashing, and the fundamental idea behind open and closed addressing.

Impact of Changing Table Size

I observed little to no change in any operations when the table size was changed. This was not surprising to me because when hashing keys we are using a modulo operation and looping around our array if we overflow, and as long as the tables are the same percentage full, I made the assumption that the tables would act similarly in their number of probes. My results did not contradict this hypothesis in any obvious way. Timing may have been affected by the sizing, as it seemed that the time perhaps increase along with the table size, however I did not see this often enough to say for sure that it is a pattern.

Consistency of Results

The results I collected were consistent throughout hashing schemes, meaning that each hashing scheme had the same amount of finds for the same data, which was expected. As far as the consistency of data collected, the probing results seemed feasible to me, although not always directly on the theoretical results, they did not differ enough to make me think my algorithm was acting incorrectly.

The consistency of timing analysis was slightly off. I notice spikes in timing when starting a new hashing scheme, and sometimes when beginning a new alpha value. These spikes are on the order of 10s or 100s of nanoseconds, so I am not too worried as these are very small measurements of time that could be related to the code itself, or possibly my computer.

Seen Probe Results Versus Theoretical Probe Results

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DH Probing |  |  |  |  | Alphas |  |  |  |  |
|  | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 0.95 | 0.98 |
| Success Seen | 1.382 | 1.444 | 1.604 | 1.831 | 2.035 | 2.375 | 2.956 | 3.614 | 4.567 |
| Success Theory | 1.188 | 1.277 | 1.386 | 1.527 | 1.719 | 2.011 | 2.558 | 3.153 | 3.991 |
| Fail Seen | 1.704 | 2.072 | 2.498 | 3.148 | 4.078 | 5.835 | 10.882 | 20.733 | 49.295 |
| Fail Theory | 1.428 | 1.666 | 2 | 2.5 | 3.333 | 5 | 10 | 20 | 50 |

Defining a probe to be the access of a location, I ran the program Assig2A.java to calculate the number of probes each hashing scheme did for both successful and unsuccessful finds. I then compared the number of probes used to the theoretical value depending on the alpha value, the fullness of the table. The results for linear probing and double hashing compared to the theoretical results are listed below. Along with the seen results of separate chaining.

Table 1 Double Hashing

It can be seen that the amount of probes for a successful search in the program increased consistently along with the theoretical values, however the amount of observed probes is also consistently higher than that of the theoretical value. My explanation for this occurrence is that the double hashing function being used in this program is not optimal, so some worse results are observed. A similar trend can be seen for the number of probes in an unsuccessful search, however this trend falls off when the table gets full. This can be explained by getting duplicate values when searching, in the next section.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LP Probing |  |  |  |  |  |  |  |  |  |
|  | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 0.95 | 0.98 |
| Success Seen | 1.21348 | 1.3364 | 1.467575 | 1.76145 | 2.16825 | 2.81187 | 4.81146 | 7.061 | 12.24984 |
| Success Theory | 1.214 | 1.333 | 1.5 | 1.75 | 2.166 | 3 | 5.5 | 10.5 | 25.5 |
| Fail Seen | 1.5105 | 1.8692 | 2.45498 | 3.5503 | 5.8867 | 11.9203 | 35.6585 | 83.386 | 247.80136 |
| Fail Theory | 1.52 | 1.888 | 2.5 | 3.6055 | 6.055 | 13 | 50.5 | 200.5 | 1250.5 |

Table 2 Linear Probing

The results of probing seen from linear probing were comparable to the theoretical values. In the amount of probes for a observed successful search are nearly identical to the theoretical value for alphas up through .7, however then the observed value is less than the theoretical value. This caused by duplicate values, elaborated on in the next section.

Unsuccessful search probes for linear probing follow a similar patter as double hashing unsuccessful searching. The observed average number of probes are similar to the theoretical values until an alpha value of .7, but then the table doesn’t seem to get much more full, due to duplicates, so the value of search probes does not increase as much as it theoretically should.

Table 3 Separate Chaining

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SC Probing |  |  |  |  |  |  |  |  |  |
|  | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 0.95 | 0.98 |
| Success Seen | 1.15749 | 1.21106 | 1.246789 | 1.32608 | 1.367 | 1.3968 | 1.4792 | 1.495 | 1.52 |
| Fail Seen | 1.297 | 1.404 | 1.507 | 1.607 | 1.709 | 1.81 | 1.911 | 1.9642 | 1.993 |

Separate Chaining did not have a listed theoretical growth value, but compared to the other searches on probing, it faired quite well, as mentioned before.

Impact of Duplicate Values

Duplicate values in this experiment were a factor because since they were not handled in a way that another number was drawn until it was not a duplicate. The tables were never guaranteed to get as full as the specified alpha value. Since the theoretical times are based off of specific alpha values, there was a difference from theory to results as the tables tested were not comparable to the theoretical tables because of duplicates. The tested tables would never be higher than the alpha values, and for higher alpha values, they could be significantly lower.

Timing Results

Results Table for Timing Analysis

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | Alphas |  |  |  |  |  |
| Timing in ns | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 0.95 | 0.98 | AVG |
|  |  |  |  |  |  |  |  |  |  |  |
| LP | 56.75 | 64 | 70 | 84 | 90.75 | 107 | 442.5 | 829.75 | 2752.5 | 499.6944444 |
| SC | 83.75 | 86.75 | 87.75 | 90.25 | 93.5 | 91.75 | 97 | 95.5 | 101.75 | 92 |
| DH | 90.75 | 96.25 | 104.75 | 113.5 | 128 | 142.5 | 149 | 298.5 | 602.5 | 191.75 |

The table above indicates that separate chaining is the fasted overall of the three hashing schemes, while linear probing is likely the slowest. Linear probing is likely the slowest because, even though it is actually the fastest of the three hashing schemes through an alpha value of .7, it does not age well, and ends up with very large search times for higher values, we see here why resizing is key with linear probing.

Double hashing is slow in the beginning compared to the others, and this is expected because of the overhead of running h2(x) instead of knowing the constant increment already, and in the case of separate chaining, there is one probe on a unsuccessful search, instead of many, so double hashing is the worst in sparser tables.

Conclusion

The three hashing schemes tested in this experiment were not unrealistic to use, however I personally would never use linear probing out of the three tested. It is very similar to double hashing but performs worse in all categories. That being said, separate chaining performed the best in all categories according to my data.

While linear probing and double hashing require less work to internally store, separate chaining will perform better on search times and probes, this is why if I was using a hashing scheme, I would choose separate chaining.