Report

LBSSHE003

Results

Load Test

The LoadTest was executed with the following command (variations highlighted):

java -cp bin LoadTest dictionary.LPHashTable 3739 lexicon.txt

LP

|  |  |  |
| --- | --- | --- |
| Load factor | Table Size | Probes |
| 0.5 | 7481 | 2135 |
| 0.75 | 4987 | 5785 |
| 1.0 | 3739 | 170285 |

QP

|  |  |  |
| --- | --- | --- |
| Load factor | Table Size | Probes |
| 0.5 | 7481 | 1725 |
| 0.75 | 4987 | 3680 |
| 1.0 | 3739 | 32783 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SC   |  |  |  | | --- | --- | --- | | Load factor | Table Size | Probes | | Na | 7481 | 1336 | | Na | 4987 | 2101 | | Na | 3739 | 2750 | |

Search Test

The SearchTest was executed with the following command (variations highlighted):

java -cp bin SearchTest dictionary.LPHashTable 3739 lexicon.txt 100 10000

LP

|  |  |  |
| --- | --- | --- |
| Load factor | Table Size | Average Probes |
| 0.5 | 7481 | 33 |
| 0.75 | 4987 | 166 |
| 1.0 | 3739 | 74756 |

QP

|  |  |  |
| --- | --- | --- |
| Load factor | Table Size | Average Probes |
| 0.5 | 7481 | 23 |
| 0.75 | 4987 | 72 |
| 1.0 | 3739 | 31094 |

|  |
| --- |
|  |
| SC   |  |  |  | | --- | --- | --- | | Load factor | Table Size | Average Probes | | Na | 7481 | 2 | | Na | 4987 | 4 | | Na | 3739 | 7 |   \*I know these results don’t look right, but it is what I got and I didn’t want to manufacture my results. I have tried to look into my code to see why I didn’t get different results. |

Analysis

Load Test

Percentage difference (for a table size of 7481) between:

|  |  |
| --- | --- |
| QP & LP: | -23.77% |
| SC & LP: | -59.81% |
| SC & QP: | -29.12% |

Percentage difference (for a table size of 4987) between:

|  |  |
| --- | --- |
| QP & LP: | -57.20% |
| SC & LP: | -175.35% |
| SC & QP: | -75.15% |

Percentage difference (for a table size of 3739) between:

|  |  |
| --- | --- |
| QP & LP: | -419.43% |
| SC & LP: | -6092.18% |
| SC & QP: | -1092.11% |

Search Test

Percentage difference (for a table size of 7481) between:

|  |  |
| --- | --- |
| QP & LP: | -43.48% |
| SC & LP: | -1550% |
| SC & QP: | -91.30% |

Percentage difference (for a table size of 4987) between:

|  |  |
| --- | --- |
| QP & LP: | -130.56% |
| SC & LP: | -4050% |
| SC & QP: | -1700% |

Percentage difference (for a table size of 3739) between:

|  |  |
| --- | --- |
| QP & LP: | -140.42% |
| SC & LP: | -1067842.86% |
| SC & QP: | -444100% |

\*\*This has been calculated with the formula that was given to us in the assignment document.

Evaluation

Linear probing(LP) uses less memory than Sequential Chaining(SC). LP doesn’t have to store the links to the next item like SC does when two items hash to the same place. However, LP is slower than SC since two or more items may hash to the same place and then LP will need to look through the table one by one to find the correct item which may take a long time due to primary clustering (Many words hashing to the same function, and then finding the next available slot to full). Quadratic Probing (QP) can always insert a new item into the table (provided that the table is not already full) and no cell is probed more than twice during an access.

The difference in processing speed between LP, QP and SC:

With SC there is pointer dereferencing and with LP and QP there is the calculation of the hash function, indexing of the hash function and then there may be repeats due to collisions (SC does not have to worry about this). Therefore, SC will produce the fastest insert/search functions, due to the collisions that LP and QP face.

The technique that generally performs the best is the SC. With SC the table will never be full, even though items may hash to the same position, it will decrease the time spent looking for an item as they are all linked, i.e. do not have to look through the whole table as with LP and QP due to clustering.

SearchTest was executed with the following command (variations highlighted):

java -cp bin SearchTest dictionary.LPHashTable 3739 lexicon.txt 100 10000

|  |  |  |  |
| --- | --- | --- | --- |
| Table Size | LP | QP | SC |
| 7481 | 39.53 | 38.87 | 0.93 |
| 4987 | 27.98 | 25.74 | 0.80 |
| 3139 | 25.53 | 20.44 | 0.68 |

\*\* results obtained by using the SearchTest class, and are in seconds.

From the results it is clear to see that the best method is SC, and the method that has the most efficient storage is SC. No matter how many values hash to the same function, the item will always be in that place. Due to the fact that SC has the least probes and the most efficient storage time, it will offer the best combination of speed and efficient storage.