**CMPUT 291 Project 2 Report**

**By Omar Almokdad, Gemma Marcinkoski, Wai Yi Low**

Implementation Techniques & Analysis

B-TREE & HASH:

The implementation of the B-tree file, and the hash file was basically identical in terms of the navigation, and algorithms used to retrieve the data, the only change was in the way the database was opened to write to or parse. The B-tree file called the opener function in the bsddb3 library with the B\_TREE flag, while the Hash file called the function with the HASH tag. This tag is what tells the database how to structure the data.

The BTREE and HASH databases gave similar response times to each other in the three different scenarios: searching by key, searching by value, and searching by range with lower and upper limit, as seen in the tables below.

Our first method, searching by key, generated the results from both databases is in average of about 200 microseconds with insignificant variance. Searching by key took the shortest time among the three scenarios because it uses the key, which is the main factor in how the data is structured.

For the other two methods, searching by value and searching by range, generated results in average of over 100,000 microseconds. That is because a sequential search is required for both of them, while the data is not structured sequentially

INDEXFILE:

The implementation of the index file was a little bit different than the other two. For the index file, as the database was created, a second database was created along-side it, with the values set up to have their own index with a B-Tree structure. This allows both searching by key, and by value to be more efficient, and avoids sequentially searching the database in these two cases.

Experimental Results (Time taken to execute in milliseconds)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **B-Tree** | Key Search | Data Search | Range Search | # of Entries in Range |
| Test 1 | 159 | 109851 | 114220 | 3776 |
| Test 2 | 338 | 103225 | 122203 | 3830 |
| Test 3 | 183 | 103806 | 109262 | 3900 |
| Test 4 | 269 | 112935 | 110991 | 3773 |
| Average | 237.25 | 107454.3 | 114169 | 3819.75 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hash** | Key Search | Data Search | Range Search | # of Entries in Range |
| Test 1 | 183 | 119030 | 136416 | 3776 |
| Test 2 | 177 | 121662 | 133407 | 3830 |
| Test 3 | 180 | 122166 | 131969 | 3900 |
| Test 4 | 314 | 128582 | 130380 | 3773 |
| Average | 213.5 | 122860 | 133043 | 3819.75 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Index** | Key Search | Data Search | Range Search | # of Entries in Range |
| Test 1 | 204 | 196 | 12334 | 3776 |
| Test 2 | 308 | 195 | 11934 | 3830 |
| Test 3 | 193 | 323 | 13095 | 3900 |
| Test 4 | 314 | 200 | 11050 | 3773 |
| Average | 254.75 | 228.5 | 12103.25 | 3819.75 |

Summary of Results

|  |  |  |  |
| --- | --- | --- | --- |
| Queries | B-Tree | Hash Table | Index File |
| Key Search | 237 | 214 | 255 |
| Data Search | 107454 | 122860 | 229 |
| Range Search | 114169 | 133043 | 12103 |