ABOUT SKIP LIST:

Skip List data structure makes the searching within ordered set of elements efficient. It contains the elements arranged as a list with each node having a hierarchy of pointers to the elements at a particular distance from the node.

|  |  |
| --- | --- |
| **OPERATION** | **EXPECTED TIME** |
| Adding an element | O(log n) |
| Removing an element | O(log n) |
| Finding an element | O(log n) |

SKIPLIST VS JAVA’s TREEMAP:

|  |  |  |
| --- | --- | --- |
| **INPUT** | **SKIPLIST\_ TIME(msec)** | **TREE MAP\_TIME(msec)** |
| a1.txt | 13 | 10 |
| a2.txt | 11 | 10 |
| a3.txt | 17 | 13 |
| a4.txt | 36 | 29 |
| a5.txt | 90 | 101 |
| a6.txt | 267 | 268 |
| a7.txt | 299 | 348 |
| a8.txt | 1836 | 2317 |
| a9.txt | 1792 | 2359 |
| b1.txt | 8 | 8 |
| b2.txt | 10 | 9 |
| b3.txt | 13 | 16 |
| b4.txt | 28 | 33 |
| b5.txt | 92 | 106 |
| b6.txt | 252 | 278 |
| b7.txt | 334 | 374 |
| b8.txt | 2235 | 2716 |

From the comparison, we can see that the skip lists work as efficient as TreeMaps in java.