# **Assignment 4**

# **Splay Tree Empirical Analysis**

# **Submitted By:**

# **Saba Kanwal (18L-1860)**

**Code Details:**

Implementation of BST, AVL and Splay Trees are taken from <https://www.geeksforgeeks.org>. All of these are implemented in C++. Uniform distribution and Zipf distribution code is also taken from internet and some changes are made by me to make it working. First I generate data for tree from which we have to search and write it to a text file, so it can same for all of the above trees. And same data for search generated for both uniform distribution and Zipf distribution. And save it in to text files. For building trees data files are data-100000, data-1000000, and data-10000000 for three experiments described in assignment. For searching with uniform distribution files are search-uni-100000, search-uni-1000000 and search-uni-10000000. And Zipf distribution files are zipf-100000, zipf-1000000, and zipf-10000000.

**Execution Time:**

Clock library used for only determining the total execution time of all search operations. And then average execution time for each search operation is determined. Also the process existed time noted that includes time of all insertions and search operations.

**Specifications of Machine:**

**Processor:** Intel(R) Core(TM) i5-7200U CPU@ 2.50GHz 2.70GHz

**Installed RAM:** 8.00GB

**System Type:** 64-bit operating system, x64 based processor

**Laptop Company:** HP

**Run Time Environment:**

**Operating System:** Windows 10 Enterprise

**Programming Language:** C++

**Tool:** Dev C++ 5.11

**Compiler:** TDM-GCC 4.9.2 64 bit release

**Experiments:**

* **Experiment with n=100000 and Uniform Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 942704 | 744954 | 100080 |
| **Average comp. per search Op.** | 9 | 7 | 1 |
| **Total Exe. Time of all search Op.** | 50 milli sec | 47 milli sec | 15 milli sec |
| **Avg. Exe. Time per search Op.** | 0.5 micro sec | 0.47 micro sec | 0.15 micro sec |
| **Total rotations in search op** | 0 | 0 | 81 |
| **Avg. rotation per search op** | 0 | 0 | 8.1x10-4 |
| **Process existed time** | 3.993 sec | 3.639 sec | 3.985 sec |

* **Experiment with n=1000000 and Uniform Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 11721317 | 9154403 | 1000198 |
| **Average comp. per search Op.** | 11 | 9 | 1 |
| **Total Exe. Time of all search Op.** | 929 milli sec | 904 milli sec | 173 milli sec |
| **Avg. Exe. Time per search Op.** | 0.929 micro sec | 0.904 micro sec | 0.173 micro sec |
| **Total rotations in search op** | 0 | 0 | 202 |
| **Avg. rotation per search op** | 0 | 0 | 2.04x10-4 |
| **Process existed time** | 7.02 sec | 8.287 sec | 6.37 sec |

* **Experiment with n=10000000 and Uniform Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 140213888 | 108709496 | 10000265 |
| **Average comp. per search Op.** | 14 | 11 | 1 |
| **Total Exe. Time of all search Op.** | 15224 milli sec | 13802 milli sec | 1769 milli sec |
| **Avg. Exe. Time per search Op.** | 1.5224 micro sec | 1.3802 micro sec | 0.1769 micro sec |
| **Total rotations in search op** | 0 | 0 | 268 |
| **Avg. rotation per search op** | 0 | 0 | 2.68x10-5 |
| **Process existed time** | 47.83 sec | 53.65 sec | 41.97 sec |

* **Experiment with n=100000 and Zipf Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 50624 | 44656 | 100064 |
| **Average comp. per search Op.** | 0.50624 | 0.44656 | 1 |
| **Total Exe. Time of all search Op.** | 15 milli sec | 16 milli sec | 15 milli sec |
| **Avg. Exe. Time per search Op.** | 0.15 micro sec | 0.16 micro sec | 0.15 micro sec |
| **Total rotations in search op** | 0 | 0 | 65 |
| **Avg. rotation per search op** | 0 | 0 | 6.5x10-5 |
| **Process existed time** | 3.272 sec | 3.23 sec | 3.296 sec |

* **Experiment with n=1000000 and Zipf Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 314565 | 233784 | 1000130 |
| **Average comp. per search Op.** | 0.3145 | 0.2337 | 1 |
| **Total Exe. Time of all search Op.** | 172 milli sec | 210 milli sec | 125 milli sec |
| **Avg. Exe. Time per search Op.** | 0.172 micro sec | 0.210 micro sec | 0.125 micro sec |
| **Total rotations in search op** | 0 | 0 | 131 |
| **Avg. rotation per search op** | 0 | 0 | 1.31x10-4 |
| **Process existed time** | 5.367 sec | 6.536 sec | 5.597 sec |

* **Experiment with n=10000000 and Zipf Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **BST** | **AVL** | **Splay Tree** |
| **Total comp. in all search Op.** | 5463793 | 2523479 | 10000131 |
| **Average comp. per search Op.** | 0.5463 | 0.2523 | 1 |
| **Total Exe. Time of all search Op.** | 1644 milli sec | 2257 milli sec | 1269 milli sec |
| **Avg. Exe. Time per search Op.** | 0.1644 micro sec | 0.2257 micro sec | 0.1269 micro sec |
| **Total rotations in search op** | 0 | 0 | 132 |
| **Avg. rotation per search op** | 0 | 0 | 1.32x10-5 |
| **Process existed time** | 34.41 sec | 41.84 sec | 42.17 sec |

**Discussion:**

In case of uniform distribution, Splay tree performs better than both BST and AVL Trees. Average execution time and average number of comparisons per search operation is less than both of BST and AVL trees. But in terms of rotations, AVL trees only perform rotations on insertion or deletion. So, average rotations per search operation is more in splay trees.

In case of Zipf distribution, the performance of splay trees is reduced than others. Its execution time and average number of comparisons are more than both of the BST and AVL trees. AVL performs better than BST in terms of average number of comparisons. BST performs better than AVL in terms of time. Because in AVL trees rotations performed on insertion of nodes.