Lane Gramling
EECS 560
April 12, 2018

Lab 9 - Observations Report

> Organization of experimental profiling.

Each data structure’s performance analysis is recorded using a struct acting as a 2D array of the following form:

```
    buildTimes[5] = {0, 0, 0, 0, 0}
deleteMinTimes[5] = {0, 0, 0, 0, 0}
deleteMaxTimes[5] = {0, 0, 0, 0, 0}
```

Where the successive indices for each row correspond to increasing sample sizes, from 1,000,000 to 5,000,000 in increments of 1,000,000.

> Input data generating using random number generator.

All data for a given cell is generated using a random number generator, which is re-seeded and the table recreated 5 times for each cell in the aforementioned table, where the 5 results are averaged out.

> CPU time recording in C++.

The primary consideration during the implementation of time recording is minimizing the recorded sections of code to the relevant portions, so as to retrieve the most accurate time estimation for the performance analysis.

> Data recording and analysis.

Data is first generated and inserted into the given structure with M elements. This is where the build time of the structure is recorded M elements. Next, 1% of the structure’s total number of elements (0.01 * M) are deleted using deleteMin. Afterwards, 1% are deleted using deleteMax. The time for each deletion is recorded into a running sum, before being averaged out across the 5 executions for the given value of M.

> Performance comparison, observations and summary.

Both Heaps typically outperformed the Binary Search Tree, with the Heaps’ delete methods having inverse timing behaviors from each other, with the delete(Min/Max) corresponding to the Heap’s Min/Max order having the better performance on the respective Heap. This aligns with the time complexities of deleteMin() on min-Heaps and deleteMax() on max-Heaps having O(log(n)) worst-case complexities while the opposing delete methods have O(n) worst-case complexities.

> Conclusion.

The results gathered from this experimental profiling substantiate the expected results inferred from their worst-case complexities for each of the build(), deleteMin(), and deleteMax() operations on a min-Heap, max-Heap, and Binary Search Tree.