Freya Varez

Cpt\_S 223

8/31/2018

**Problem Statement:**

The goal of this problem was to test the computation time for various algorithms, most notably to insert data into an ordered, singly linked list and further calculate the median, maximum and minimum values within the newly created list. Secondary goals were to increase the efficiency of these algorithms to handle larger list sizes.

**Algorithm Design:**

The initial algorithm to create nodes and insert them into a pre-sorted singly linked list was relatively simple algorithm. The algorithm created a node for the incoming data, then iterated through the list, finding the correct space in the list of sorted nodes (sorted from smallest at the head node, to largest at the tail).

Once this list was created, algorithms were made to find the minimum, maximum and median of the values within the linked list. In order to make the respective min and max algorithms most efficient, a pointer pHead and pTail were created within the scope of the main(), recording the head and tail of the linked list respectively. Thus, finding the min and max was simply a matter of reading the indirect memory in pHead and pTail.

Similarly, for the algorithm to calculate the median of the linked list, an integer counter was recorded within the scope of main() to count the number of nodes inserted into the linked list. Thus, finding the median value was to find the node within the list (in the case of a list with an odd number of values, this value was averaged with the value).

Potential approaches that would have decreased overall time-performance would be excluding the pTail pointer, the process of finding the maximum value of the list would have then involved directly iterating times to the last value of the list.

**Experimental Setup**

Since the purpose of this experiment was simply to calculate the time complexity of the algorithms mentioned above, the design of the experiment was relatively straightforward. The experiment involved introducing data sets of different sizes, the measuring the time needed to process the data. This experiment was done through a Unix environment (run through the EECS server).

This experiment was repeated 7 times (see below for details). Note below that time\_min and time\_max are consistently 0 milliseconds, this is due to the efficient algorithm mentioned above, the time taken to find min and max is simply the time taken to access the indirect memory of pHead and pTail (constant time, almost instantaneous).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # Data points | Time\_insert | Time\_min | Time\_max | Time\_med |
| 1000 | 0 | 0 | 0 | 0 |
| 10,000 | 210 | 0 | 0 | 0 |
| 50,000 | 11520 | 0 | 0 | 0 |
| 100,000 | 56250 | 0 | 0 | 10 |
| 300,000 | 965970 | 0 | 0 | 10 |
| 500,000 | 2943860 | 0 | 0 | 20 |
| 1,000,000 | 14,538,214 | 0 | 0 | 25 |