Vincent Lee
[redacted]
P5 Hypothesis

Sorting Experiment Hypothesis

Cocktail sort/Happy Hour
Worst case performance $O(n^2)$
Best case performance $O(n)$
Average case performance $O(n^2)$

Insertion sort
Worst case performance $O(n^2)$
Best case performance $O(n)$
Average case performance $O(n^2)$

Bubble sort
Worst case performance $O(n^2)$
Best case performance $O(n)$
Average case performance $O(n^2)$

Quicksort
Worst case performance $O(n^2)$
Best case performance $O(n \log n)$
Average case performance $O(n \log n)$

From the chart above, information gathered from wikipedia, the worst case performance for all four algorithms should be the same. Also for the happy hour sort, bubble sort, and insertion sort, the best case and average case performance should be the same. This is all theoretical and in real life I believe this will be different. From looking at the statistics on those three arrays I cannot conclude a winner. What I can hypothesize about is that quick sort's best case and average case performance should be a clear winner taking only a fraction of time since its time is O(n log n). For all algorithms besides quick sort, average case scenario time should exponentially increase with the amount of numbers being sorted. Quicksort seems most efficient through the average case performance through the numbers.

I believe that the performance will be different on different platforms, i.e. Nike the Mac's in 307, and on a few of my personal machines. Nike's performance, I believe, will be the most unpredictable. For one Nike is a multi-user system, and as Plaue also notes, handles all of the .cs.uga.edu emails. In a perfect world, we could run our sort algorithm, on Nike through a physical terminal connected to Nike, with all network connections off. That is not possible frankly. Some worst case scenarios would be for example, if a 1301 student was running an infinite loop on Nike, taking up all 4 of the Intel Xeon 2.4Ghz 10 core 20 thread processors. When running the sorting algorithm during this event would yield longer times than normal. Another problem that could occur is if a Backdoor Exploit was utilized on Nike to create a black hole email server, which would spam email accounts, and use up precious CPU time. Also since JAVA is ran on a virtual machine, students running a similar project in C++ are closer to the hardware in terms of execution, could use up more CPU time than ran on JAVA's virtual machine. The Mac's in the Boyd 307 should yield similar results if they are all running the latest version of OSX. The only problem is the Mac's do not have JAVA installed. On my personal computer there would be some variation, because my computer is not running server grade chips, and I have a lot of background process running on my Windows 7 Box. Performance, I would conclude would be quite close, as I am the only user on the machine at a particular time.

This is only a hypothesis, and to my greatest knowledge through just the O(n) time given to me, I can only guess that Quick Sort will be the fastest, and the other three will follow in terms of shortest time. I will be implementing these methods and running tests and will report back through the formal write up of my results.

Vincent