Homework 10
(Maximum 50 points)
Due beginning of class Monday November 16, 2015

1. (50 points) [Sorting programming: running time comparison] Compare the running time of insertion sorting, Shell sort, quicksort, and LSD radix sort using each of the provided six files of seven-digit random numbers. The files are named by the number of random numbers in them, i.e., r_10, r_100, r_1K, r_10K, r_100K, and r_1M.

- For Shell sort, use the Sedgewick's increment sequence: $h_k = 9 \times 4^k - 9 \times 2^k + 1$ (k = 0,1,2,…) for odd positions and $4^k - 3 \times 2^k + 1$ (k=2,3,4,…) for even positions, that is, 1,5,19,41,109, 209, 505, 929, 2161, 3905, …, ($\leq \lfloor N/2 \rfloor$).
- For quicksort, use the median-of-three strategy to pick the pivot. Cut off when the file size is small (e.g., 5) and finish off with insertion sorting.

Show the measured running time of each algorithm in a table of the following three columns: algorithm name, file size, and running time. You can use the static function "System.nanoTime()" to measure the elapsed time in nanoseconds. In addition, show the running time of each algorithm in a graph with respect to the file size and state your observations on the growth rates of the running time as the file size increases. (Draw the graphs in a base-10 log scale.) Submit your source codes via Blackboard. You can use the program codes from textbook or any other source as the base as long as the source is acknowledged.

Note, if you are using the program codes from textbook:
- The textbook Shellsort program uses Shell's increment sequence. Do not use it but use Sedgewick's sequence instead.
- The textbook radix sort assume input keys are character strings, while the input files used in this exercise have digit strings (i.e., decimal integers). So, in your radix sort program, extract one digit at a time (from the least significant digit first) from the input decimal number. This extraction will introduce some overhead, which will work against the speed of radix sort when compared with other sorting algorithms, but it will not affect the linearity of running time complexity.

Last modified: November 4, 2015