CSCI 4452.2 2015 Assignment #2

Show all your work.

(1) (a) Use gprof to perform a runtime analysis of the BVP_SOLVER-2 software (BVP_M-2.f90, BVP_LA-2.f), available at the CSCI Moodle course webpage, using the AMR1_20.f90 driver software, also available at that website. On the cs machine, an appropriate compiler is the gfortran compiler. When you compile the BVP_LA-2.f source code file, use the -w switch to supress warnings.
Discuss what the analysis shows. If you were interested in improving the efficiency of this software by investigating the algorithms employed within the various modules which make up the software, which modules should you focus upon? If you could improve the performance of each of the top four most time-consuming modules by 50%, what would be the effect on the overall performance of the solver?
(b) Examine and discuss the output from the solver. How many different meshes are considered and how many subintervals does each mesh contain. How many Newton iterations are performed using each mesh? What is the estimated defect of the numerical solution associated with each case, if a numerical solution was obtainable? Is that solution acceptable? What is the defect of the final numerical solution? What is the user tolerance?

(2) (a) For the list, [13, 1, 12, 3, 9, 5, 2, 11, 10, 8, 6, 4, 7], use the tournament tree method to determine the largest element in the list. Show the tournament tree that is obtained.
(b) Use the approach discussed in class for the efficient determination of the second largest element in the list. Recall that this involves tracing back from the root to the leaf level, replacing all instances of the largest element with $-\infty$, followed by a "repair" of the tournament tree. Show the tournament tree that is obtained from this second step.
(c) Use the approach of part (b) to efficiently determine the third largest element in this list.
(d) Suppose you had a list of size N where N was 10 million, and suppose that each comparison takes one microsecond on a given computer. If you were to use a standard method for finding the five largest elements in the list (where each such search costs about N comparisons) how long would it take for the computation to complete? In contrast, using the tournament tree approach (with repairs) how long would the computation take? Explain your answers in detail.