Problem 1. Assume that each word of your machine has 60 bits. Assume that you can multiply two n-word numbers in time $4n^2$ with a standard algorithm. Assume that you can multiply two n-word numbers in time $11n^{\lg 3}$ with a "fancy" algorithm. For each part *briefly justify* and *show your work.*

(a) Approximately how large does n have to be for the fancy algorithm to be better?

(b) Approximately how many bits is that?

(c) Approximately how many decimal digits is that?

Problem 2. Assume that you multiply the two two-digit numbers 63 and 25 using the method that does only three atomic multiplies. Show that steps of the algorithm on this example.

Problem 3. Consider the following recurrence (for the time of some algorithm).

$$T(n) = 4T(n/5) + 3n + 1, \qquad T(1) = 2 \ .$$

(a) Calculate $T(25)$ by hand. Show your work.

(b) Use the tree method to solve the recurrence exactly, assuming n is a power of 5.

(c) Use the formula to calculate $T(1)$. Show your work.

(d) Use the formula to calculate $T(5)$. Show your work.

(e) Use the formula to calculate $T(25)$. Show your work.

(f) Use Mathematical Induction to prove that your formula is correct in general.

Problem 4. Recall the following selection-sort-like algorithm (from Homework 3): In the first iteration, find the smallest and largest values and put them at the beginning and the end of the list, respectively. In the second iteration, find the second smallest and second largest values and put them next to the beginning and next to the end of the list, respectively. In the third iteration, find the third smallest and third largest values and put them into their proper locations. Etc.

(a) Write the pseudo code for this version of selection sort, where the outer loop is done by *recursion.* As before, assume that you have available the method `MinMaxIndices(A,p,r)`. Make your algorithm efficient with respect to number of comparisons.

(b) Write a recurrence for the (exact) number of comparisons.