Assignment 5. Due Dec 7, 2015 (In class).

1. (5 + 5 points)

   a) In a k-to-all personalized communication, k nodes simultaneously perform one-to-all personalized communication operations (k is between 1 and p) in a p-node ensemble with individual packets of size m. Show that if k is a power of two, you can perform the operation in time \( t_s \log(p/k) + k - 1 \) + \( t_w m(p-1) \). (Give each step in the algorithm along with associated runtime)

b) In a k-to-all broadcast, k nodes simultaneously perform one-to-all broadcast of m-word messages (assume that these messages cant be split). Show that you can perform the operation in time \( t_s \log p + t_w m(k \log (p/k) + k - 1) \). (Give each step in the algorithm along with associated runtime)
2. (5 + 5 points)

a) Give an algorithm for the all-reduce operation. In this case, each processor has a vector with \( m \) entries. At the end of the operation, each processor must end up with another vector of \( m \) entries such that each of the entries in this result vector is the sum of all corresponding entries across the processors in the input vectors. One simple way for this operation is to perform a reduce with a single target and follow it with a broadcast. Can you get rid of the factor of 2 in the runtime of this simple algorithm?

b) Give an algorithm for a k-reduce operation, where the result of the reduction must be available at \( k \) processors. As before, the trivial algorithm uses a reduce operation followed by a one-to-k broadcast. Can you do better?
3. (6 + 4 points)

a) Consider the message passing parallel formulation of quicksort on p processors. Assume that you have a pivot selection strategy that guarantees that neither of the two sub-lists of a list of length m are shorter than log m. With this pivot selection strategy, what is the worst-case parallel runtime of sorting n entries on p processors?

(b) (6 points) Consider the problem of sorting a list of n numbers. Initially, all numbers are at processor 0. The processor computes a pivot and splits the list into two parts. One part is sent to processor p/2 and the other part is retained at processor 0. Each of these processors at the next stage repeats this process until the leaf level processors have n/p elements. These elements are then sorted using serial quicksort. Assume best case pivot selection, what is the parallel runtime of this formulation?
4. (5 + 5 points)

a) Consider the block row-wise 1-D version of a matrix-vector product. We wish to scale the problem with increasing $p$ in such a way that the parallel time remains asymptotically constant. What is the scaled speedup of this algorithm?

b) The memory requirement of the simple all-to-all broadcast based matrix-matrix product algorithm is $\frac{n^2}{\sqrt{p}}$ per processor. (This formulation uses a block 2D partitioning -- it uses an all-to-all broadcast of matrix $A$ along the row of processors and of matrix $B$ along the column of processors. This is followed by a local matrix-matrix product). Assuming that we scale the problem to fit available memory and memory grows linearly with number of processors, what is the scaled speedup of this formulation? Comment on the quality of this speedup expression?