CS 373: Combinatorial Algorithms, Spring 2002

Homework 1, due January 29 23:59:59, 2002

Name:	
Net ID:	Alias:
Name:	
Net ID:	Alias:
Name:	
Net ID:	Alias:

Starting with Homework 1, homeworks may be done in teams of up to three people. Each team turns in just one solution, and every member of a team gets the same grade. Since graduate students are required to solve problems that are worth extra credit for undergraduate students, **grad students may not be on the same team as undergraduates.**

Neatly print your name (first name first, with no comma), your network ID, and a short alias into the boxes above. **Do not sign your name. Do not write your Social Security number.** Staple this sheet of paper to the top of your homework.

Grades will be listed on the course web site by alias give us, so your alias should not resemble your name or your Net ID. If you don't give yourself an alias, we'll give you one that you won't like.

Required Problems

1. **[10 Points]** (Based on CLRS 27.1-2 and 27.1-4, or CLR 28.1-2 and 28.1-4)

   (a) **[5 Points]** Let n be an exact power of 2. Show how to construct an n-input, n-output comparison network of depth $\lg n$ in which the top output wire always carries the minimum input value and the bottom output wire always carries the maximum input value.

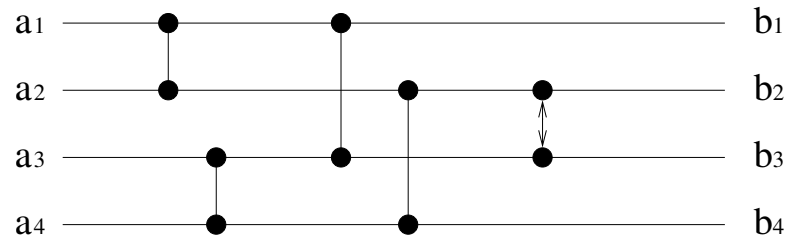
   (b) **[5 Points]** Prove that any sorting network on n inputs has depth at least $\lg n$.

2. **[10 Points]** (Based on CLRS 27.1-8 or CLR 28.1-8)

   Suppose that in addition to the standard kind of comparator, we introduce an "upside-down" comparator that produces its minimum output on the bottom wire and its maximum output on the top wire.

(a) **[6 Points]** An n-input sorting network with m comparators, is represented by a list of m pairs of integers in the range from 1 to n. Thus, a comparator between the wire i and j is represented as (i, j). If $i < j$ then this is a regular comparator, and if $i > j$ it is an upside-down comparator.

Describe an algorithm that converts a sorting network with n inputs, c upside-down gates and m overall gates into an equivalent sorting network that uses only regular gates. How fast is your algorithm?



Suppose in the comparison network above, all comparators are regular ones except the rightmost one, which is an upside-down comparator. This comparison network is represented as the following: $(1, 2), (3, 4), (1, 3), (2, 4), (3, 2)$.

(b) **[4 Points]** *Prove* that your algorithm is correct (i.e., it indeed outputs a network that uses only regular comparators, it always terminate, and the output network is equivalent to the input network).

3. **[10 Points]** (Based on CLRS 27.2-5 or CLR 28.2-5)

Prove that an n-input sorting network must contain at least one comparator between the ith and $(i+1)$st lines for all $i = 1, 2, ..., n-1$.

4. **[10 Points]** (Based on CLRS 27.5-1 and 27.5-2, or CLR 28.5-1 and 28.5-2)

The sorting network SORTER$[n]$ was present in class (it is also shown in CLRS Figure 27.12 or CLR Figure 28.12), where n is an exact power of 2. Answer the following questions about SORTER$[n]$.

(a) **[5 Points]** Give a tight bound for the number of comparators in SORTER$[n]$.

(b) **[5 Points]** Show that the depth of SORTER$[n]$ is exactly $(\lg n)(\lg n + 1)/2$.

5. **[10 Points]** (Based on CLRS 27.5-3 or CLR 28.5-3)

Suppose that we have $2n$ elements $< a_1, a_2, ..., a_{2n} >$ and wish to partition them into the the n smallest and the n largest. Prove that we can do this in constant additional depth after separately sorting $< a_1, a_2, ..., a_n >$ and $< a_{n+1}, a_{n+2}, ..., a_{2n} >$.

6. **[10 Points]** (Based on CLRS 27.5-4 or CLR 28.5-4)

   Let $S(k)$ be the depth of a sorting network with k inputs, and let $M(k)$ be the depth of a merging network with $2k$ inputs. Suppose that we have a sequence of n numbers to be sorted and we know that every number is within k positions of its correct position in the sorted order, which means that we need to move each number at most $(k-1)$ positions to sort the inputs. For example, in the sequence 3 2 1 4 5 8 7 6 9, every number is within 3 positions of its correct position. But in sequence 3 2 1 4 5 9 8 7 6, the number 9 and 6 are outside 3 positions of its correct position.
   Show that we can sort the n numbers in depth $S(k)+2M(k)$. (You need to prove your answer is correct.)

7. **[20 Points]** (Based on CLRS 27.5-5 or CLR 28.5-5)

   We can sort the entries of an $m \times m$ matrix by repeating the following procedure k times:

   1. Sort each odd-numbered row into monotonically increasing order.
   2. Sort each even-numbered row into monotonically decreasing order.
   3. Sort each column into monotonically increasing order.

   (a) **[8 Points]** Suppose the matrix contains only 0's and 1's. We repeat the above procedure again and again until no changes occur. In what order should we read the matrix to obtain the sorted output ($m \times m$ numbers in increasing order)? Prove that any $m \times m$ matrix of 0's and 1's will be finally sorted.

   (b) **[8 Points]** Prove that by repeating the above procedure, any matrix of real numbers can be sorted. [Hint:Refer to the proof of the zero-one principle.]

   (c) **[4 Points]** Suppose k iterations are required for this procedure to sort the $m \times m$ numbers. Give an upper bound for k. The tighter your upper bound the better (prove you bound).