ICS 691: Parallel Algorithms
Homework 2

Due: October 30, 2014

Instructions: You may discuss the problems with other students in the class, but you must write up the solutions on your own and list the names of the students with whom you discussed each problem.

1 Pipelined mergesort (Exercise 4.22 in JàJà) - 30 pts

Prove that for any stage $s \geq 1$, and for any arbitrary node v of the binary complete tree T, $|L_{s+1}(v)| \leq 2|L_s(v)| + 4$. That is the size of the list in each node at most roughly doubles in size at each stage. (Hint: use induction on s.)

2 ShearSort: sorting on square matrices - 40 pts

Consider the following algorithm for sorting items in an $n \times n$ matrix, where rows and columns are numbered $\{1, 2, \ldots, n\}$.

- Repeat the following $\lceil \log n \rceil$ times:
  1. Sort rows in alternating order: sort each odd-numbered row in increasing order and each even-numbered row in decreasing order.
  2. Sort each column in increasing order
- Sort each row in increasing order

(a) Assuming that the rows and columns are sorted using a sorting network, prove that the above algorithm sorts the matrix in a row-major order. (Hint: Use the 0-1 principle and consider how many rows remain to be sorted after each round)

(b) Analyze the running time of the algorithm.

3 List ranking - 30 pts

Prove the correctness of Willie's pointer hopping algorithm. (Hint: Use induction on the number of iterations and consider where each node points to and the values they store.)

4 BONUS: Odd-even mergesort (Exercise 4.33 in JàJà) - 20 pts

Let $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ be two sorted sequences such that n is a power of 2. The **odd-even merge** algorithm consists of recursively merging the two "odd" sequences, $(a_1, a_3, \ldots, a_{n-1})$ and $(b_1, b_3, \ldots b_{n-1})$, into $(c_1, c_2, \ldots, c_n)$, and merging the two "even" sequences, $(a_2, a_4, \ldots, a_n)$ and $(b_2, b_4, \ldots b_n)$, into $(c'_1, c'_2, \ldots, c'_n)$. Then the sorted sequence is given by
$(c_1, \min\{c'_1, c_2\}, \max\{c'_1, c_2\}, \min\{c'_2, c_3\}, \max\{c'_2, c_3\}, \ldots, \min\{c'_{n-1}, c_n\}, \max\{c'_{n-1}, c_n\}, c'_n)$.

(a) Using the zero-one principle, show that the odd-even merge algorthm works correctly. (Hint: place sequences A and B in a $2 \times n$ matrix in column major order and consider what happens after each step.)

(b) Derive a sorting network based on the odd-even merge algorithm. State its depth (time) and its size (work).