18.435/6.848 Theory of Parallel and VLSI Computation	**Fall 1997**

Problem Set #1

Due: Wednesday, September 16 — *Prof. Leighton*

Reading (for lectures 1–3): Sections 1.1–1.2 of the text.
(for lectures 4–6): Sections 1.3–1.6 of the text.

Problems (from lectures 1–3):

1. Consider two algorithms for solving a problem of size M, one that runs in M steps on an M-processor machine and one that runs $\sqrt{M}$ steps on an M^2-processor machine. Say that we need to run the algorithm X times and that we have a P-processor machine. Determine the conditions on X,M, and P for deciding which algorithm is best. (1.4)

2. An **Up/Down Counter** is a counter which takes input sequence $x_1, x_2, \ldots, x_N$, where $x_i \in \{+1, -1\}$ is input on the ith step. After seeing each x_i, the counter outputs "0" if $\sum_{j=1}^{i} x_j = 0$, and "$\neg 0$" otherwise in the $(i+1)$th step. Show how to implement an up/down counter on a $\log N$-cell linear array in the bit model where each cell has $O(1)$ bits of memory. Here, N is the maximum length of input sequence. The ith input is delivered to the 1st cell of the array at step i.

*3. Show how to sort N $\log N$-bit numbers in $O(N)$ bit steps using the network shown in Figure PS1.1. (Hint: Use the solution to Problem 2 and the approach of bucket sort.) Can you show that the time bound must be $\Omega(N \log N)$ for sorting N $2\log N$-bit numbers on this network?

*4. **The Firing Squad Problem.** Given an N-cell linear array of bit processors (each with $O(1)$ bits of memory), show how to coordinate them so that they all enter a special state simultaneously at step $10N$. No inputs are allowed other than the clock ticks and a start signal entered into the left end of the array. (There is nothing magical about 10; any sufficiently large constant will do.)

5. Show that any connected N-node network with E edges and maximum node degree d can simulate a $2E+1$-node linear array with slowdown $\leq d$. (Hint: You may use the fact without proof that any connected graph has a double Eulerian Tour, that is, a cycle which traverses each edge of the graph exactly twice.) Use this fact and the firing squad solution (which you may now assume) to extend the result of the previous problem to work for any N-cell bounded-degree network (The Firing Mob Problem). Make sure that every node of the network fires at the same time. Why did we need to assume that $d = O(1)$?

*6. Prove that the bisection width of an $M \times N$ mesh is at least $\min(M, N)$ for any M and N. (1.14)

7. If each leaf of an N-leaf complete binary tree has $O(\log N)$ storage, show how to compute an $N \log N$-input prefix problem in $O(\log N)$ steps on the tree. Explain briefly how to implement addition of two $N \log N$-bit numbers on an N-leaf complete binary tree in $O(\log N)$ time. (1.24,1.38)

8. It is easy to compute the exclusive-or of N bits in N steps on an N-cell linear array. Show how to compute the exclusive-or of N bits in $2\sqrt{N}$ steps on a linear array. (You are allowed to input one bit per processor per step.)

9. Show how to compute the N-input prefix problem on a $\sqrt{N} \times \sqrt{N}$ array in $O(\sqrt{N})$ time where each processor has constant amount of memory. Show how to make it faster and more efficient by running in $O(N^{1/3})$ time on an $N^{1/3} \times N^{1/3}$ array where each processor has $\Theta(N^{1/3})$ memory.

10. Let $s_1, s_2, \ldots, s_N$ be a secret key (known to sender/receiver), let $m_1, m_2, \ldots, m_N$ be a message and let $x_i = m_i + m_{i-1} + s_i \pmod 2$ be their encoding algorithm with $m_0 = 0$. The encoded message will be of the form $x_1, \ldots, x_n$. Notice that computing this requires $O(1)$ steps on N processors. Design an efficient decoding algorithm. How fast can you make it using N processors?

Problems For Extra Credit

**1. The greatest common divisor (GCD) of two integers u and v is defined to be the largest integer d such that d divides both u and v. Show how to compute the GCD of two N-bit numbers in $O(N)$ bit steps on a linear array. Hint: This problem is tricker than you might first think! It may help you to observe that $\mathrm{GCD}(u,v) = 2 \cdot \mathrm{GCD}(u/2, v/2)$ when u and v are even, $\mathrm{GCD}(u,v) = \mathrm{GCD}(u/2, v)$ when u is even and v is odd (and symmetrically), and $\mathrm{GCD}(u,v) = \mathrm{GCD}((u \pm v)/4, u) = \mathrm{GCD}((u \pm v)/4, v)$ when u and v are both odd (choosing the sign so that 4 divides $(u \pm v)$).) You may give a high level description of the algorithm and proof, but make sure that it really works in $O(N)$ bit steps. (1.50)

$$2. Describe a parallel algorithm for the GCD of two N-bit numbers that runs in $o(N)$ steps using a bounded-degree network with $N^{O(1)}$ processors.

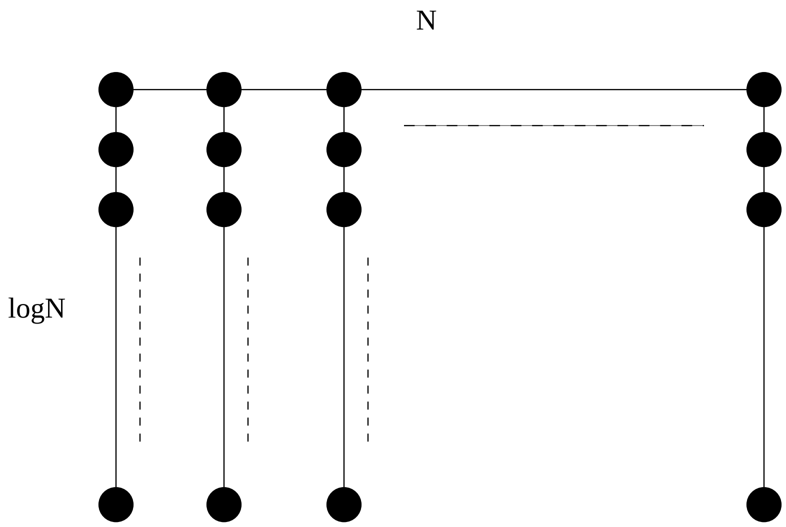

Figure PS1.1

Figure PS1.1: