15-849(C) Parallel Computing 25 Sept 98
Homework Assignment 1

Due Date: Monday, Oct 5

Problem 1.

Let's say you have been hired by FastSim Inc. and they would like to buy a 64 processor parallel machine and would like you to help them figure out what machine to buy. The three choices are a Sun Enterprise server, an SGI Power Challenge and a Cray T3E. Let's assume they all cost about the same. Please summarize the pros and cons of the three machines (in a page or less). In particular consider the various features we discussed so far in class (e.g. network parameters, shared vs. distributed memory, processor speed). You can base your decision on the web pages that are given off of the course home page, and any other information you find on the web, or elsewhere. You can work in groups on this question. For all machines you can assume the latest fastest processor they offer.

Problem 2.

Sometimes it is useful to emulate one network with another since this would allow us to use an algorithm designed for one network on another network. For example we might emulate a 8x8x8 mesh on a 512 processor hypercube by assigning one processor of the mesh to each processor of the hypercube. We call the network being emulated the *guest network* (e.g. the mesh above) and the network used for the simulation the *host network* (e.g. the hypercube above). We assume that each edge of the guest is assigned a fixed path in the host. The interesting question to ask for any guest and host is how efficient is this emulation. We use the following measures to measure efficiency.

Dilation The maximum amount we must stretch a guest edge in the host. For example if all neighbors in the guest are 3 away from each other when emulated on the host, then the dilation is 3.

Congestion The maximum number of edges of the guest that are routed on any edge of the host.

A. What is the best dilation and congestion for mapping an 8x8x8 toroidal mesh on a 512 processor hypercube.

B. If we map a n node hypercube on a $\sqrt{n} \times \sqrt{n}$ mesh can you give a lower bound on the congestion and dilation? Hint, consider the bisection bandwidth and latency of the two networks.

Problem 3.

Consider the following notation and definitions:

- $T^*(n)$ – the running time of a sequential algorithm on a problem of size n
- $T_p(n)$ – the running time of a parallel algorithm on a problem of size n using p processors.
- $T_\infty(n)$ – the running time of a parallel algorithm on a problem of size n using an unlimited number of processors.
- The speedup $S_p(n) = T^*(n)/T_p(n)$ – how much faster a parallel algorithm runs on p processors than a sequential algorithm on one processor.
- An algorithm is *scalable* if $\lim_{n\to\infty} \frac{T_\infty(n)}{T_s(n)} = 0$. This basically says that for "large enough" problems we can always get more speedup with more processors.
- Work $W_p(n) = PT_p(n)$, where P is the number of processors – the total number of cycles (or operations) used by a parallel algorithm.
- A parallel algorithm is *work efficient* if $W_p(n) = O(T_s(n))$

A. If a parallel sorting algorithm runs in $O(n)$ time on $\log n$ processors,

1. is it scalable?
2. how much work does it require?
3. is it work efficient?
4. what is the speedup over a sequential algorithm?

B. Consider an algorithm that has two parts that need to be run one after the other. The first part can be implemented in parallel and has running time $T_p(n) = O(n/p + \lg n)$. The second part is not parallel and runs in time $T^*(n) = \sqrt{n}$. The best sequential implementation of the first part requires $O(n)$ time. When we look at the whole algorithm,

1. is it scalable?
2. how much work does it require?
3. for what range of p is it work efficient?
4. if the second part required $T^*(n) = n$ time, would the whole algorithm be scalable?