CS 267 Midterm - Spring 1994

1) Consider the system of linear equations

$$\begin{bmatrix} 1 & & & & \\ a_2 & 1 & & & \\ & a_3 & 1 & & \\ & & \ddots & \ddots & \\ & & & a_n & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ \vdots \\ y_n \end{bmatrix}$$

where the a_is and y_is are given, and we want to solve for the x_is.

1.1) Write a (very short) program fragment to compute all the x_i's serially. What is the running time? Use $O(\cdot)$ notation. Your answer should be a function of n.

1.2) Show how to compute all the x_is in $O(\log n)$ time using parallel prefix. Ignore the possibility of over/underflow.

Now consider the following system of linear equations

$$\begin{bmatrix} 1 & & & & & \\ a_2 & 1 & & & & \\ b_3 & a_3 & 1 & & & \\ & b_4 & a_4 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & b_n & a_n & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ \vdots \\ y_n \end{bmatrix}$$

where the a_is, b_is and y_is are given, and we want to solve for the x_is.

1.3) Write a (very short) program fragment to compute all the x_i's serially. What is the running time? Use $O(\cdot)$ notation. Your answer should be a function of n.

1.4) Show how to compute all the x_is in $O(\log n)$ time using parallel prefix. Ignore the possibility of over/underflow.

1.5) Describe generally how you would solve a linear system in parallel with nonzeros only on the diagonal and in the first m subdiagonals.

2) Explain what the following program fragments do. Sort them in order of increasing running time on the CM-5, as best as you can determine. Explain your ordering, counting the number of messages in the network, if possible.

2.1)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[i][MYPROCS] = b[MYPROCS][i];
```

2.2)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[MYPROCS][i] = b[i][MYPROCS];
```

2.3)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[i][MYPROCS] := b[MYPROCS][i];
sync()
```

2.4)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[MYPROCS][i] := b[i][MYPROCS];
sync()
```

2.5)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[i][MYPROCS] :- b[MYPROCS][i];
all_store_sync()
```

2.6)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[MYPROCS][i] :- b[i][MYPROCS];
all_store_sync()
```

2.7)

```
static int a[PROCS]::[MAXPROCS],b[PROCS]::[MAXPROCS]
    for (i=0; i++; i< PROCS)
    a[i+MYPROCS mod PROCS][MYPROCS] :- b[MYPROCS][i+MYPROCS mod PROCS];
all_store_sync()
```

3) Your task is to design a parallel particle simulator. There are two types of particles, red jellybeans and green jellybeans, which move in 3D according to Newton's laws. Red jellybeans are attracted to green jellybeans by an inverse square law, and green jellybeans are repelled from other greenjelly beans by an inverse r^6 law, so the forces between distant green jellybeans are insignificant. There are many more green jellybeans than red jellybeans. Sktch an efficient parallel implementation of this system, and explain the design choices you make. Discuss data layout and communication.