Homework 2

Cyrus Ramavarapu

29 September 2016

Problem 1:

A

Since T_0 cannot be parallelized and must finish before T_1, its time is fixed. However T_1 can be parallelized over the p processors give. As a result it will take $\frac{T_1}{p}$ time. The speed up can then be calculated by taking the ratio of the time the program will take serially over the time it will take if implemented in a parallel manner. This is given by the following equations.

$$S = \frac{T_s}{T_p}$$

$$S = \frac{t_0 + t_1}{t_0 + \frac{t_1}{p}}$$

B

Analysis of the previous speed up equation shows that the time required to run a parallel implementation depends inversely upon the number of processors. As a result, limiting the number of processors towards positive infinity will give the speed up if an unlimited number of processors is available.

$$\lim_{p \to \infty} S = \frac{t_0 + t_1}{t_0 + \frac{t_1}{p}}$$

$$S = \frac{t_0 + t_1}{t_0}$$

$$S = 1 + \frac{t_1}{t_0}$$

C

If the serial portion of the program, T_0, is defined as $a = \frac{t_0}{(t_0+t_1)}$, this value can be substituted into this equation and simplified.

$$S = 1 + \frac{t_1(t_0 + t_1)}{t_0}$$

$$S = 1 + \frac{t_1}{a}$$

Problem 2:

A deterministic algorithm for an $N \times N \times N$ mesh with wrap around links can be modelled on the X-Y algorithm looking at each index individually while also considering the distance to the target node to determine if it should pass forward or backward.

If $x_c < x_d$ *and* $abs(x_c - x_d) > x_{mid}$: send towards lower x
send towards higher x
If $x_c > x_d$ *and* $abs(x_c - x_d) > x_{mid}$: send towards lower x
send towards higher x

If $y_c < y_d$ *and* $abs(y_c - y_d) > y_{mid}$: send towards lower y
send towards higher y
If $y_c > y_d$ *and* $abs(y_c - y_d) > y_{mid}$: send towards lower y
send towards higher y

If $z_c < z_d$ *and* $abs(z_c - z_d) > z_{mid}$: send towards lower z
send towards higher z.
If $z_c > z_d$ *and* $abs(z_c - z_d) > z_{mid}$: send towards lower z
send towards higher z

Problem 3:

A:

Assuming the processor and all instructions are received perfectly, the processor can fetch 2×10^9 cache blocks a second. Each cache block is 4 bytes. In order to achieve maximum performance, the memory bandwidth will have to be $4 \times 2 \times 10^9$ or 8 Gb/s.

B:

If there is no prefecthing, the cache will empty every 4 processor reads, or equivalently, additions. This will lead to a delay of 80 clock cycles while the cache line gets refilled or approximately 20 wasted clock cycles per addition. Since the maximum possible performance was 2 GFLOPS, this gets taken down by a factor of 20 and becomes $\frac{1}{10}^{th}$ of what it was.

Problem 4:

A:

To map a $N \times N$ $2-D$ torus into a $N \times N$ $2-D$ mesh, a ring can be made from each row and each column in the mesh. The minimal congestion will be 2 due to the extra wrap around link. The minimal dialation will be N-2 because of the link that wraps around reduces the distance.

B:

To map a 9 pointed start into a $N \times N$ $2-D$ mesh, the lines connecting the central node to the corner nodes be removed and replaced with two edges connecting the corner node to its neighbors. The load will be 1; however, the congestion will be N and the dialation will be 2 because connected to its neighbor, but corners are not connected to the center.

Problem 5:

The transmission time is give by the following equation.

$$T = I + n/b$$

$$T = 5 \times 10^{-7}\,\mathrm{sec} + 32b/1 \times 10^{8}\,\frac{MB}{\mathrm{sec}}$$

$$T = 820\ nsec$$