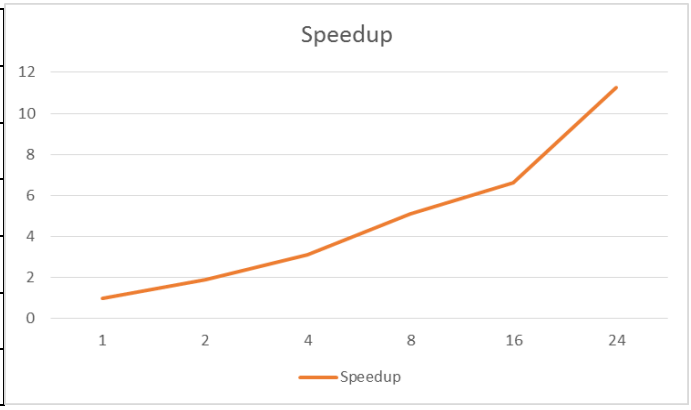
HW2

1. sol

Threads	Runtime	Speedup
1	1027.39s	1
2	544.11s	1.89
4	332.52s	3.10
8	201.46s	5.10
16	154.72s	6.64
24	91.27s	11.26



$$S(n) = \frac{1}{(1-P) + \frac{P}{n}}$$

As rough estimation, substitute the 5 records, $P = \frac{1}{5}\sum_{k=0}^{4} P_k \approx 0.922$

I.e. approximately 7.8% of the code is strictly serial.

$S(\infty) = \frac{1}{(1-P)+\frac{P}{\infty}} = 12.82$, T(min) = 80.14s

2. sol

Suppose we have a k-issue machine, assuming T_i is the time taken on one processor; For the cases no more than k instruction per cycle, and more than k instruction per cycle,

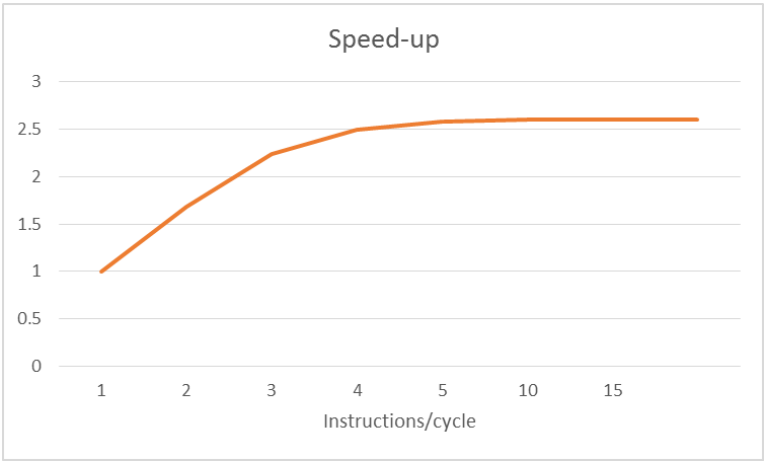
$$S_{sub} = \sum_{i=0}^{k} i * f_i \quad ; \quad S_{sur} = \sum_{i=k+1}^{\infty} k * f_i$$

For the cases,

Thus, $S_{sub} = \sum_{i=0}^{k} i * f_i + \sum_{i=k+1}^{\infty} k * f_i$

Substituting the data in the left graph of P1.7

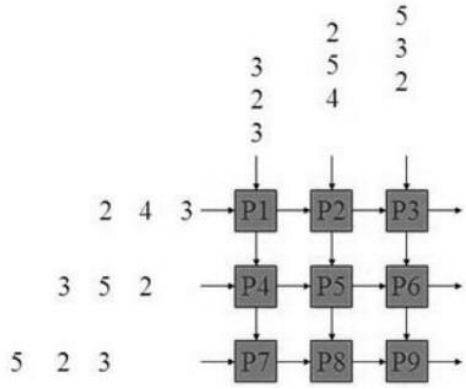
K-issue	Speedup
1	1
2	1.68
3	2.24
4	2.5
5	2.58
10	>2.6
15	>2.6
∞	>2.6



3.

Systolic array architecture seems a combination of parallelism and pipeline. The core of this architecture is “clocked parallelism”, with high throughput but low memory bandwidth.

For example, it will be very high efficient using a n*n systolic array, for which each computation unit has an adder and a multiplier, to realize the multiplication of two matrix, just as shown,



Thus, systolic array is very fit for those kind of applications like image processing and other two-dimensional data processing.