Microprocessor & Computer Architecture (µpCA)

UE19CS252

Dr. D. C. Kiran

Department of
Computer Science and Engineering

Microprocessor & Computer Architecture (μpCA)

Unit 5: Advanced Architecture

Dr. D. C. Kiran

Department of Computer Science and Engineering

Microprocessor & Computer Architecture (μpCA)

Syllabus

~~Unit 1: Basic Processor Architecture and Design~~

~~Unit 2: Pipelined Processor and Design~~

~~Unit 3: Memory~~

~~Unit 4: Input/Output Device Design~~

Unit 5: Advanced Architecture

~~Need for High Performance Computing~~

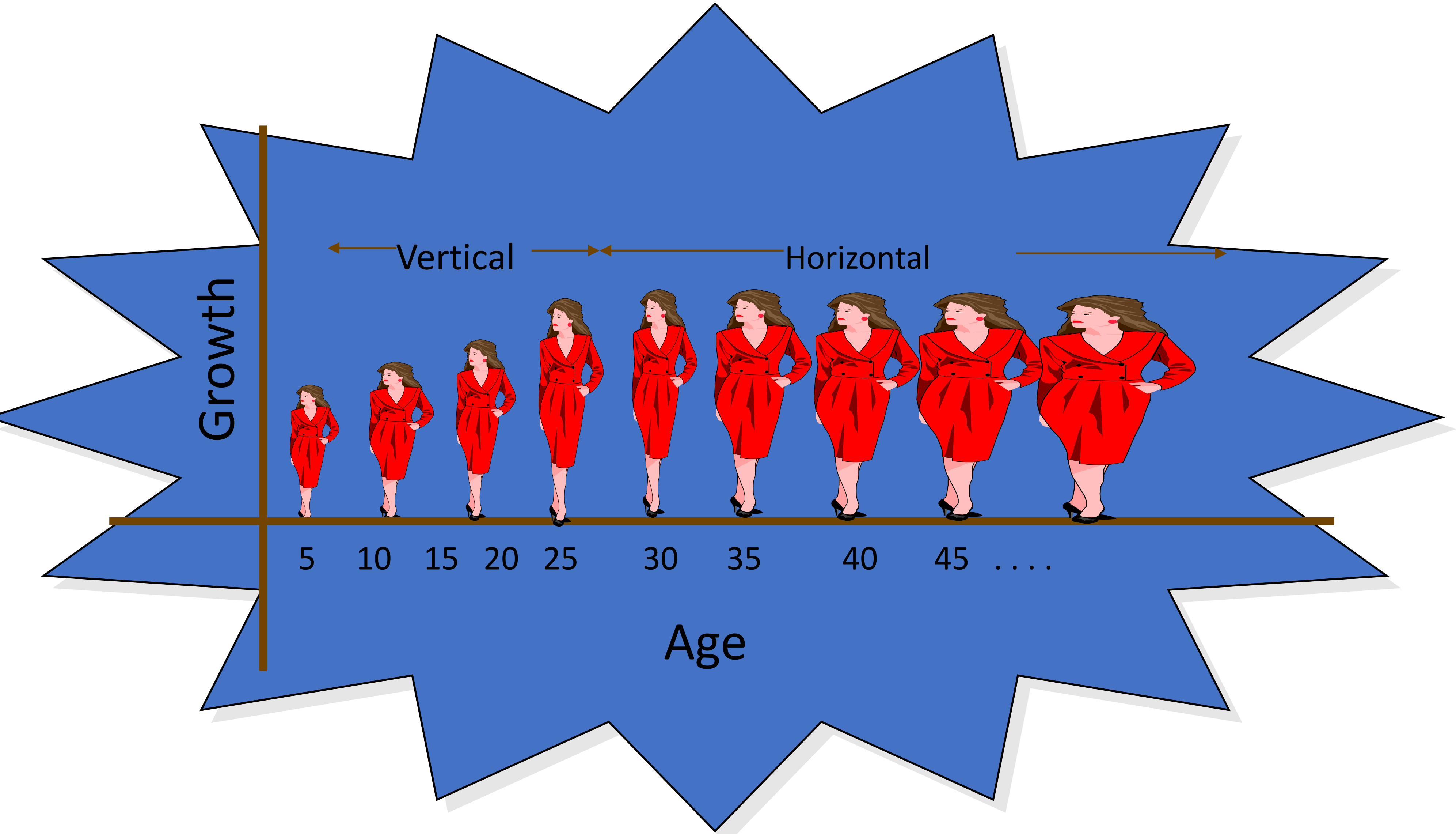
~~Classification of Parallel Architectures~~

~~Shared Memory Vs Distributed Memory Programming Paradigm.~~

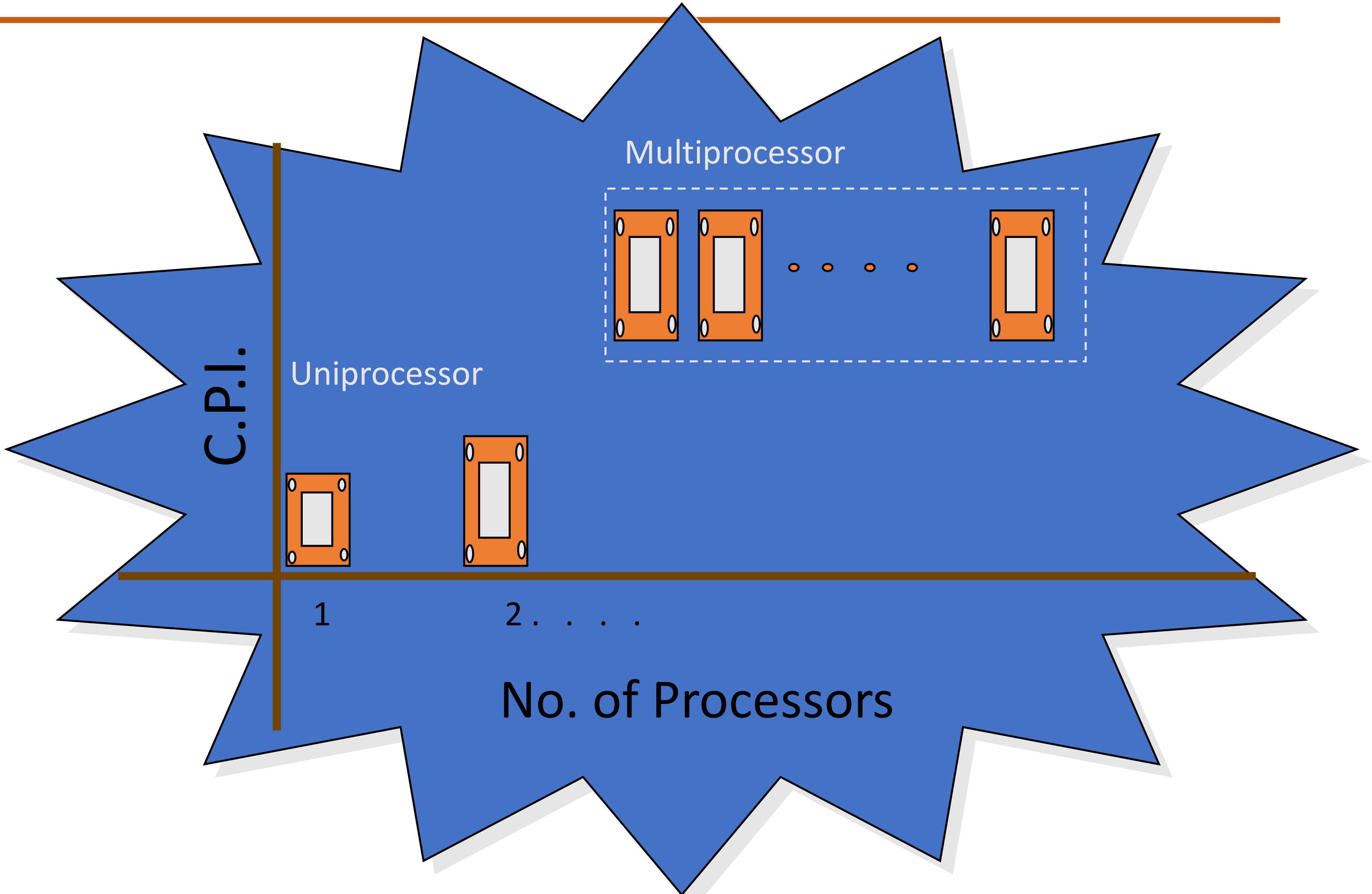
~~Bird Eye View of Parallel Architectures~~

Parallel Processing

Advancement in Parallel Computing

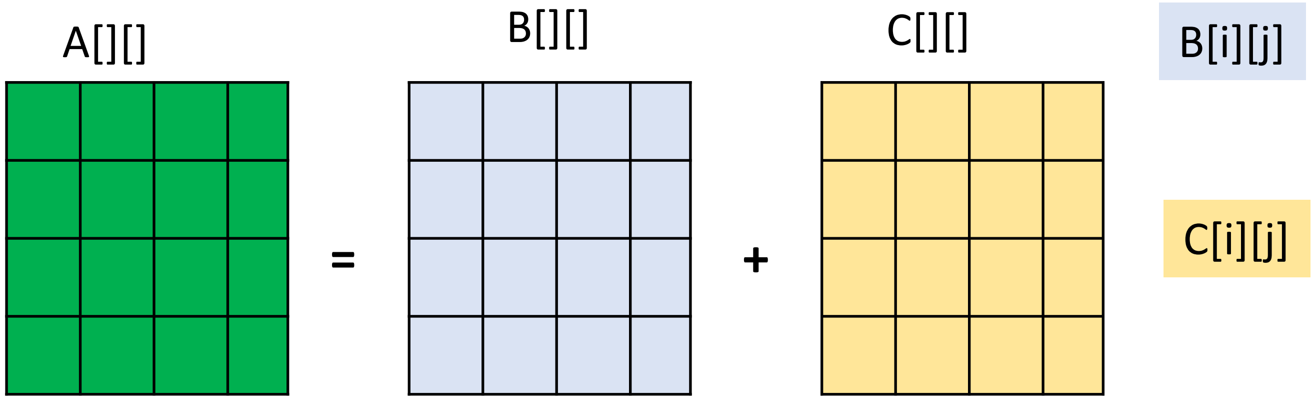


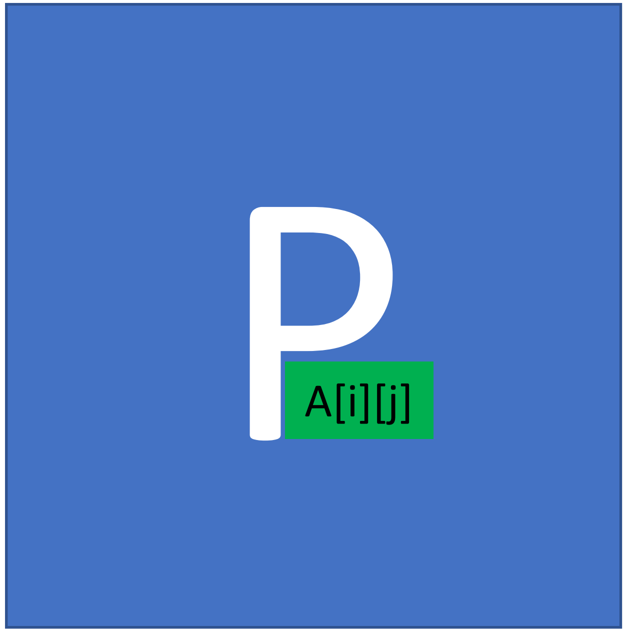
Advancement in Parallel Computing



Adding Two Matrix

```
for(i=0;i<row;i++)
  for(j=0;j<col;j++)
     A[i][j]=B[i][j]+C[i][j]
```



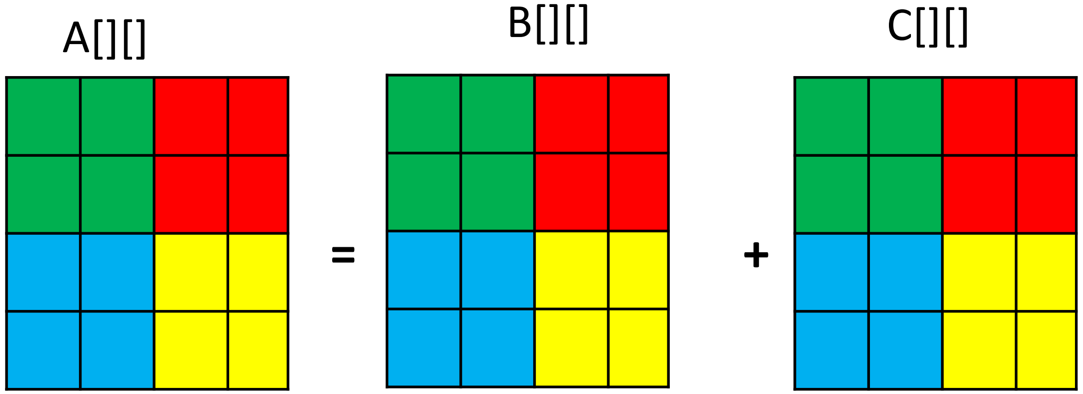


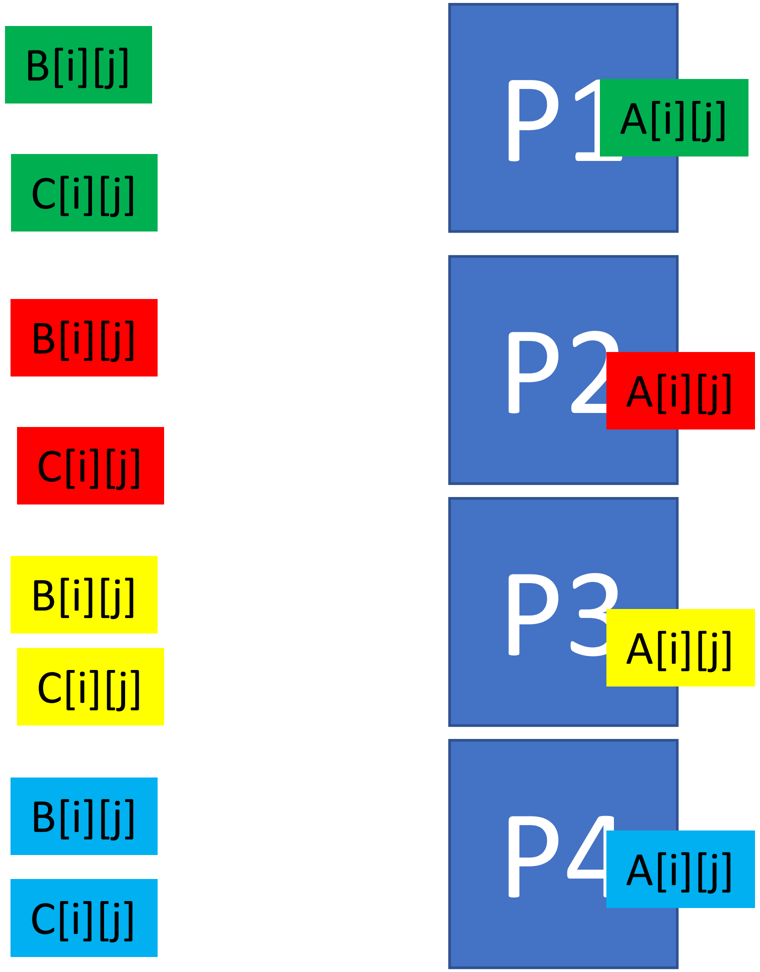
Uni Processor

Adding Two Matrix

How to Utilize 4 Processors to perform Matrix addition?

```
for(i=0;i<row;i++)
   for(j=0;j<col;j++)
      A[i][j]=B[i][j]+C[i][j]
```





Speedup- Parallel Architecture

- Speedup is the most often used measure of parallel performance
- If
  - T_s is the best possible serial time
  - T_n is the time taken by a parallel algorithm on *n* processors
- Then
  - $$Speedup = \frac{T_s}{T_n}$$

Example:

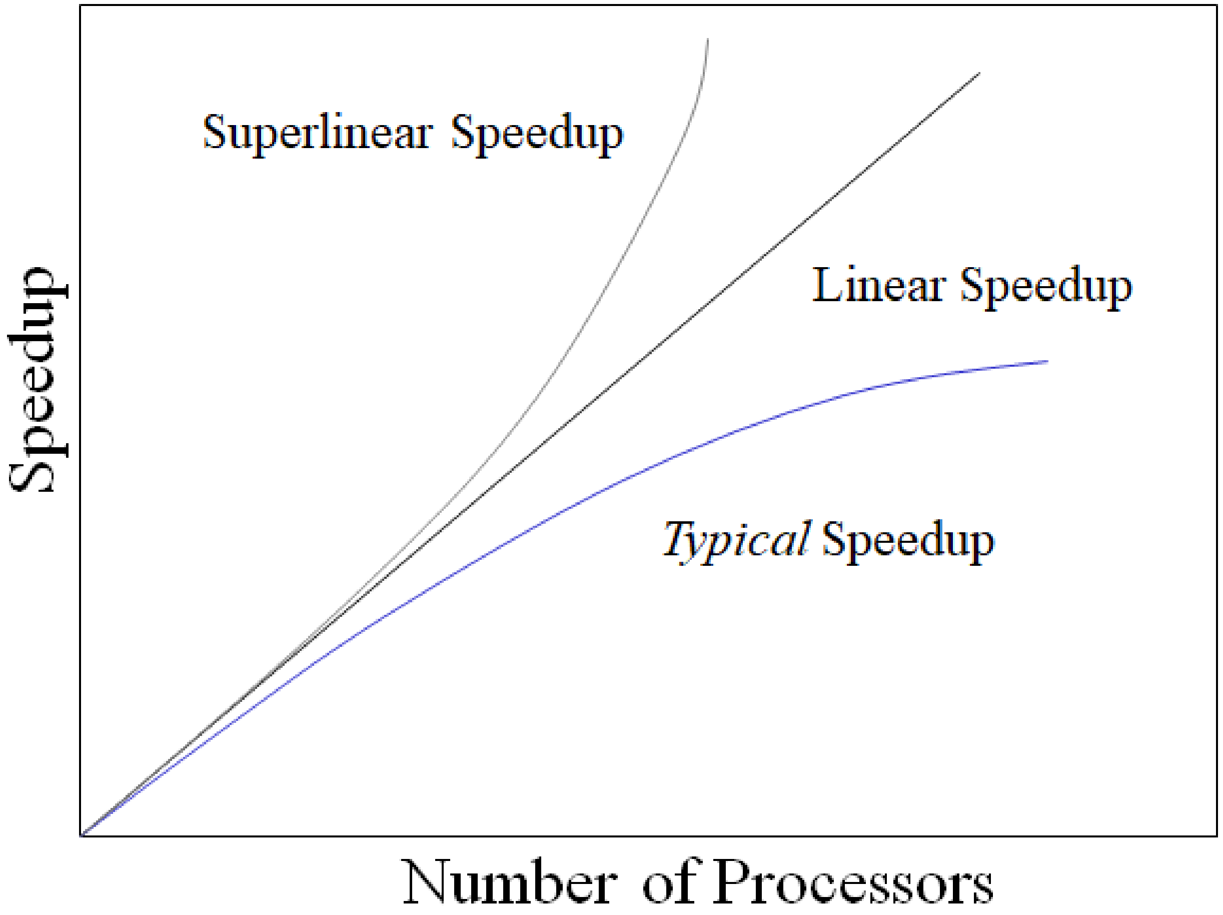
If T1 is time taken to execute program on 1 Processor.

Then T1/N is time taken to execute program on N=4 Processor

If T1=1 then TN= 0.25

Thus the speed up is 4

Speed up vs Number of Processors



Is it worth?

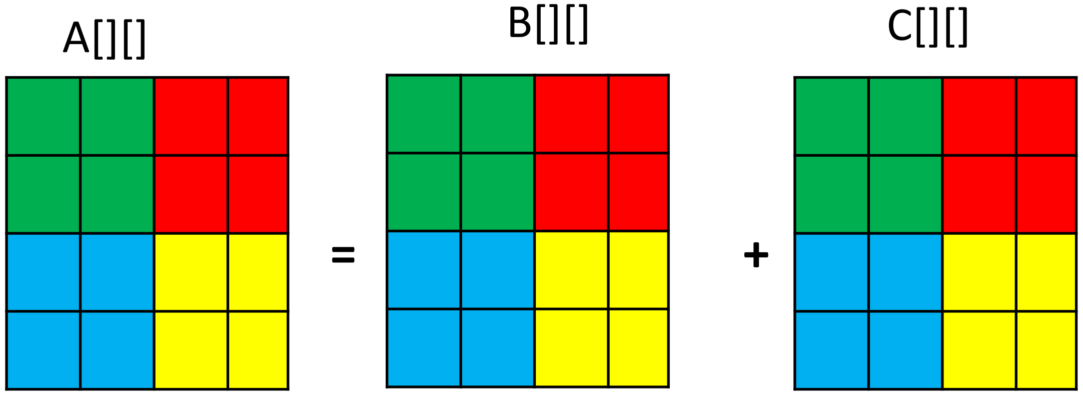
Example

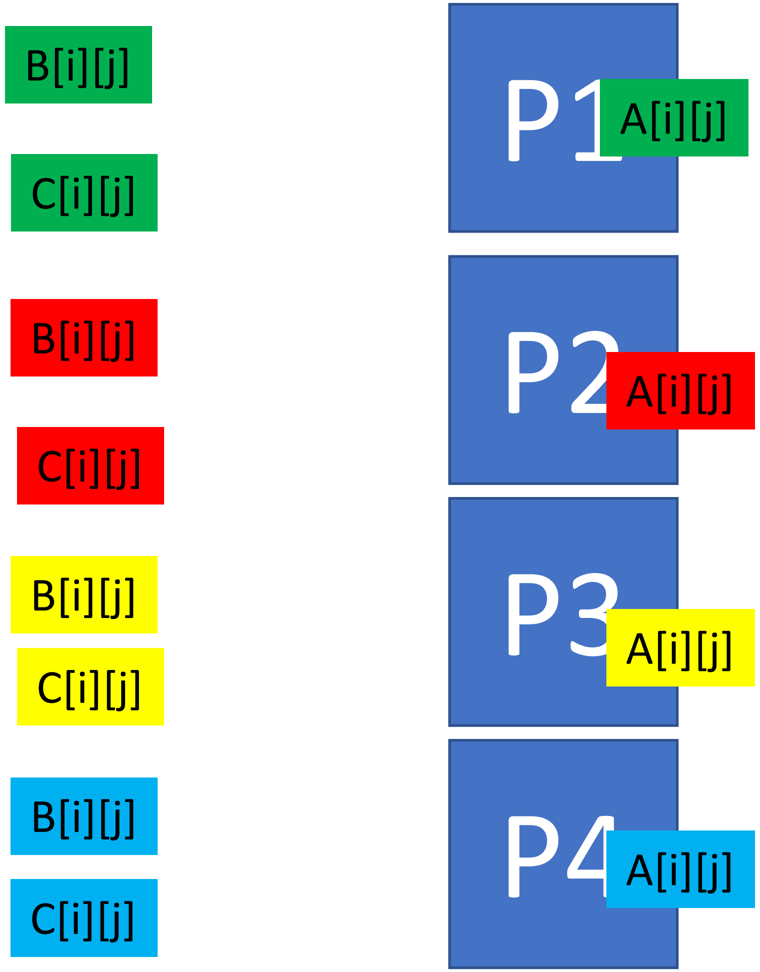
Processors	Time(secs)	Speedup	Efficiency
1	76	1.00	1.00
2	38	2.00	1.00
4	20	3.80	0.95
5	16	4.75	0.95
6	14	5.42	0.90
8	11	6.90	0.86
9	10	7.60	0.84

Adding Two Matrix

How to Utilize 4 Processors to perform Matrix addition?

```
for(i=0;i<row;i++)
   for(j=0;j<col;j++)
      A[i][j]=B[i][j]+C[i][j]
```





Everything is not Parallel
However, 4 Sequential operation in each Processor

Irony of Parallel Computing

Too Many Cook…… ☹

Design Issues:

Partitioning: Splitting to Smaller Problem
Mapping: Distributing to Multiple processor
Communication: if Required (Depend on Topology)
Consolidating : The Final result

What About it?

Problem :

```
for(j=1;j<n;j++)
        A[j]=B[j]+C[j]+A[j-1]
```

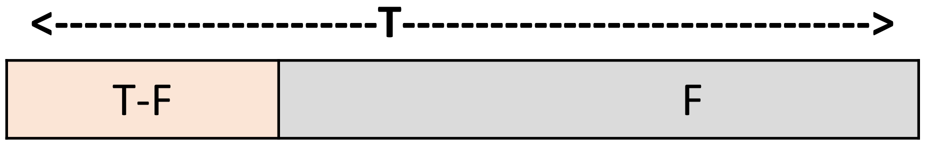
The Above program cannot be fully parallelized due to dependency between the Instructions

Truth of Parallel Execution

A program (or algorithm) which can be parallelized can be split up into two parts:

- **A part which cannot be parallelized**
- **A part which can be parallelized**

- T = Total time of serial execution
- T-F = Total time of non-parallizable part
- F = Total time of parallizable part (when executed serially, not in parallel)



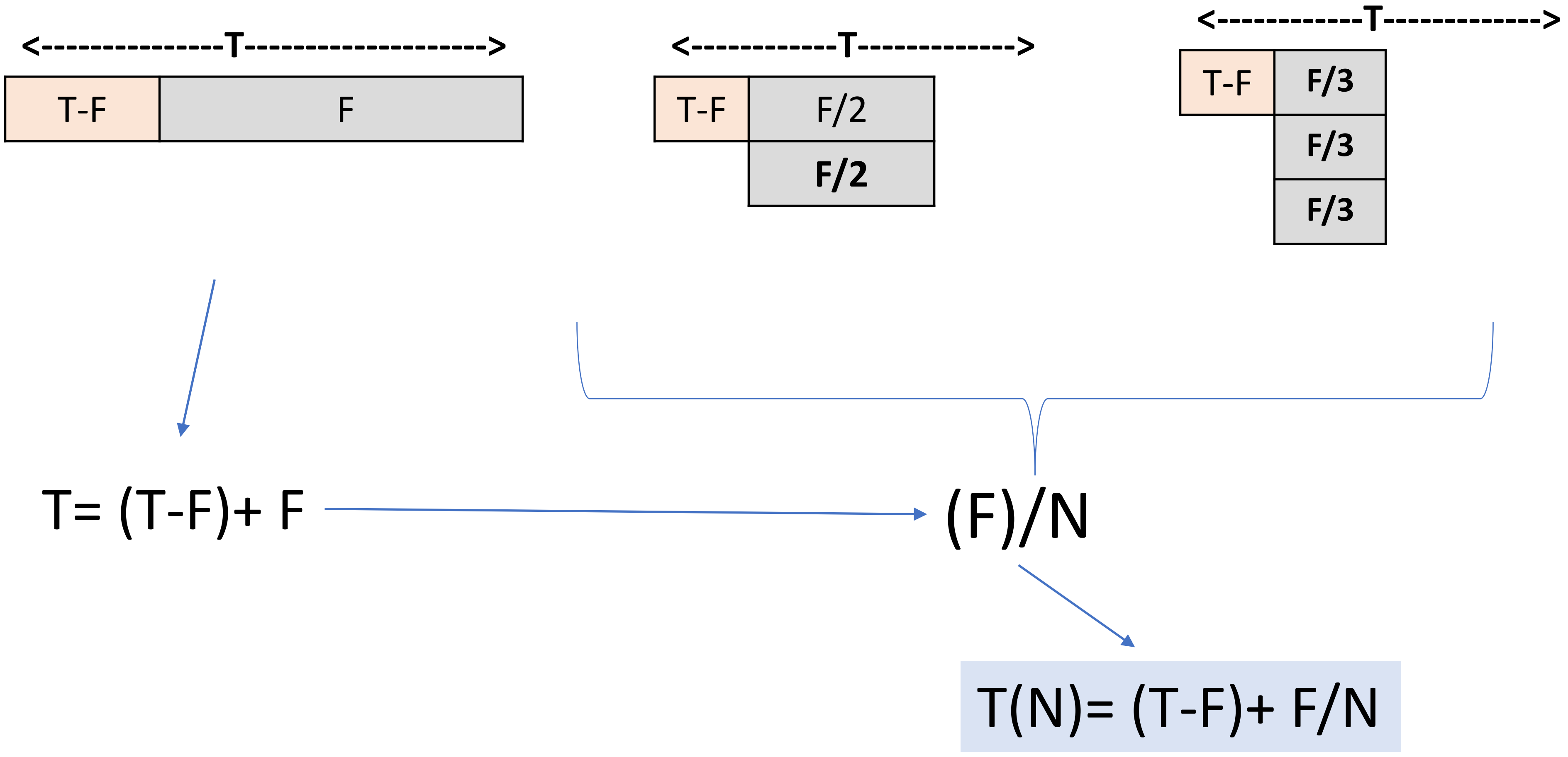
<---------------------T------------------------------->

T-F

F

T= F+ (T-F)

Truth of Parallel Execution


<---------------T-------------------->
T-F
F
<------------T------------->
T-F
F/2
F/2
<------------T------------->
T-F
F/3
F/3
F/3

T= (T-F)+ F

(F)/N

T(N)= (T-F)+ F/N

Example

The total time to execute a program is set to 1. The parallelizable part of the programs consumes 60% of the execution time. What is the execution time of the program when executed on 2 processor ?

Solution:

The parallelizable part is thus equal = 0.6.

Time for non-parallelizable part is 1-0.6= 0.4 .

The execution time of the program with a parallelization factor of 2 (2 threads or CPUs executing the parallelizable part, so N is 2) would be:

$$\begin{aligned} T(2) &= (1-0.6) + 0.6 / 2 \\ &= 0.4 + 0.6 / 2 \\ &= 0.4 + 0.3 \\ &= 0.7 \end{aligned}$$

Example

Making the same calculation with a parallelization factor of 5 instead of 2 would look like this:

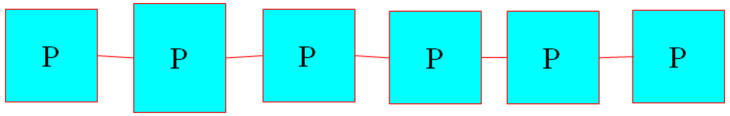
$$\begin{aligned} T(5) &= (1-0.6) + 0.6 / 5 \\ &= 0.4 + 0.6 / 5 \\ &= 0.4 + 0.12 \\ &= 0.52 \end{aligned}$$

Conclusion, by increasing the number of processing unit will not contribute in improved execution time. Instead, pralllelization in the program need to be improved.

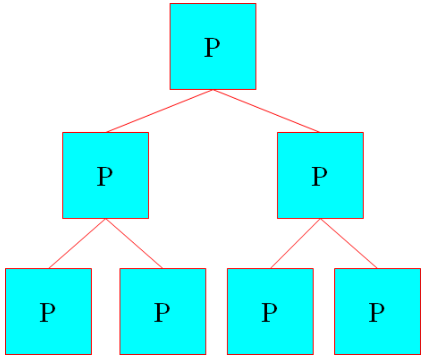
i.e writing parallel program make sense

Issues with Communication Time & Topology

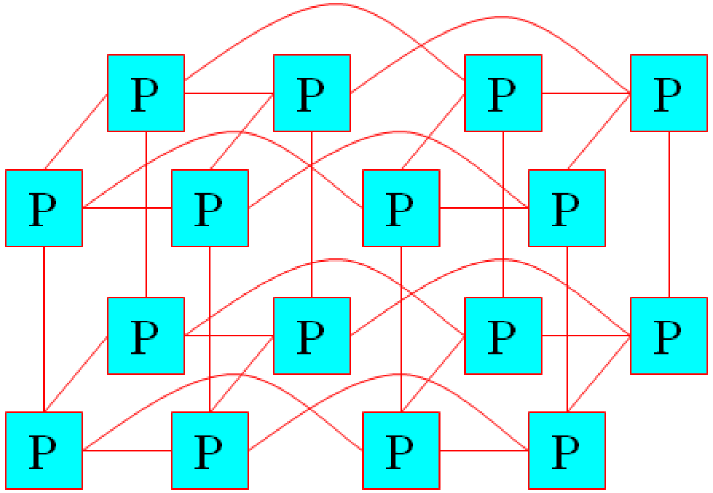
Different Parallel Computing Topology?



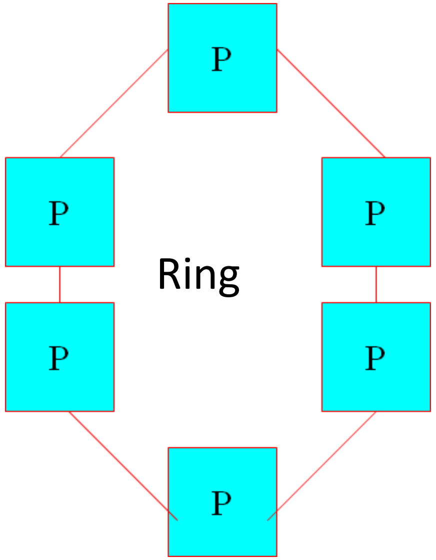
Linear



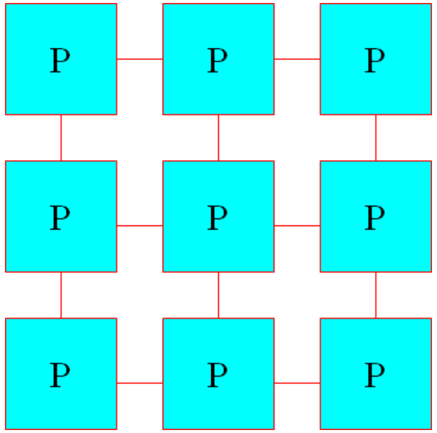
Tree



Hypercube



Ring



Mesh

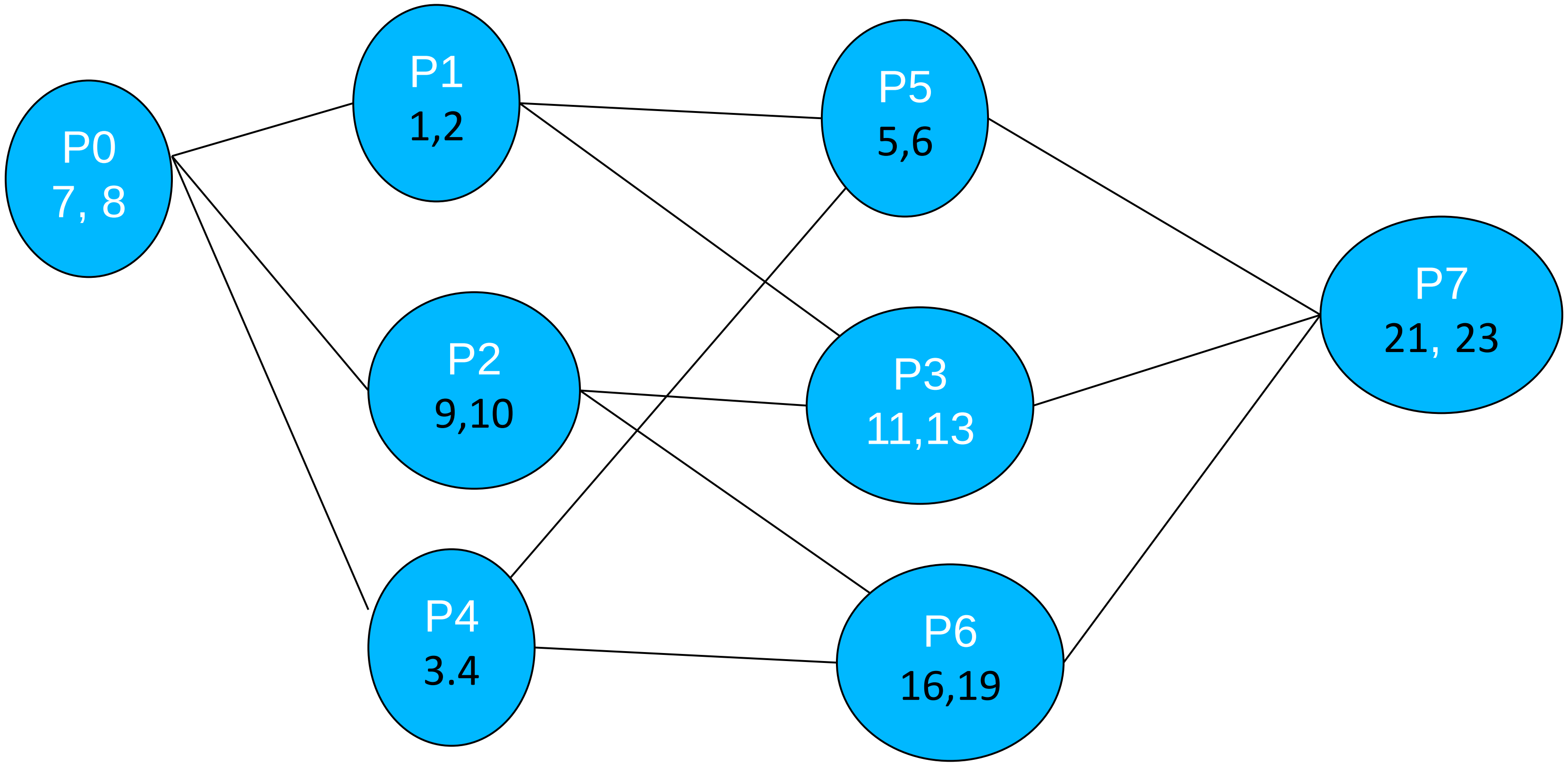
Challenges!!

- Different (Parallel) Programming Skills required as Topologies changes.
- Communication Cost (Time) changes according to Topologies.
- Through understanding of the Hardware is required to write program to utilize the computational power fully

Summation (Hypercube SIMD)

Input

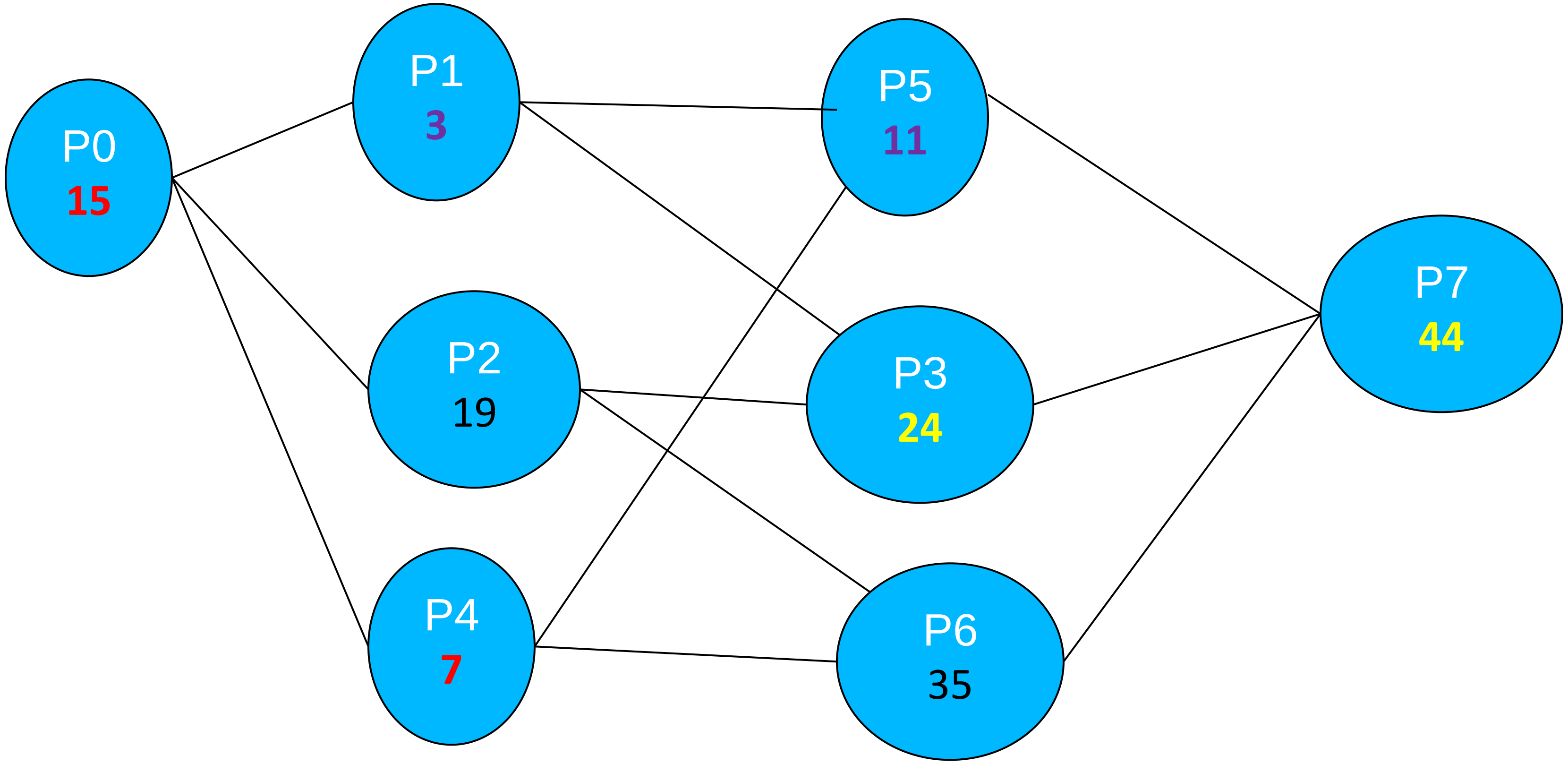
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (Hypercube SIMD)

Input

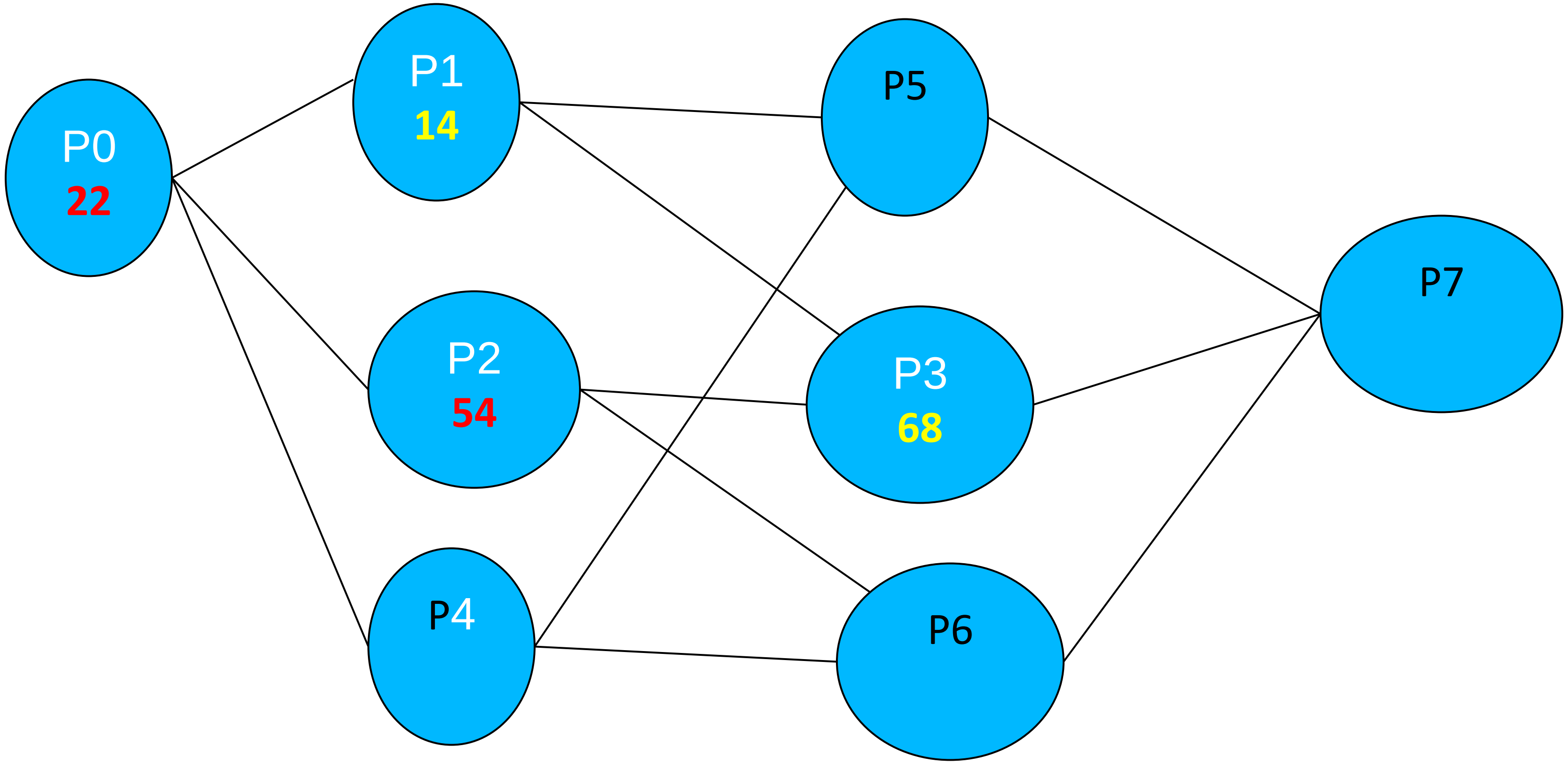
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (Hypercube SIMD)

Input

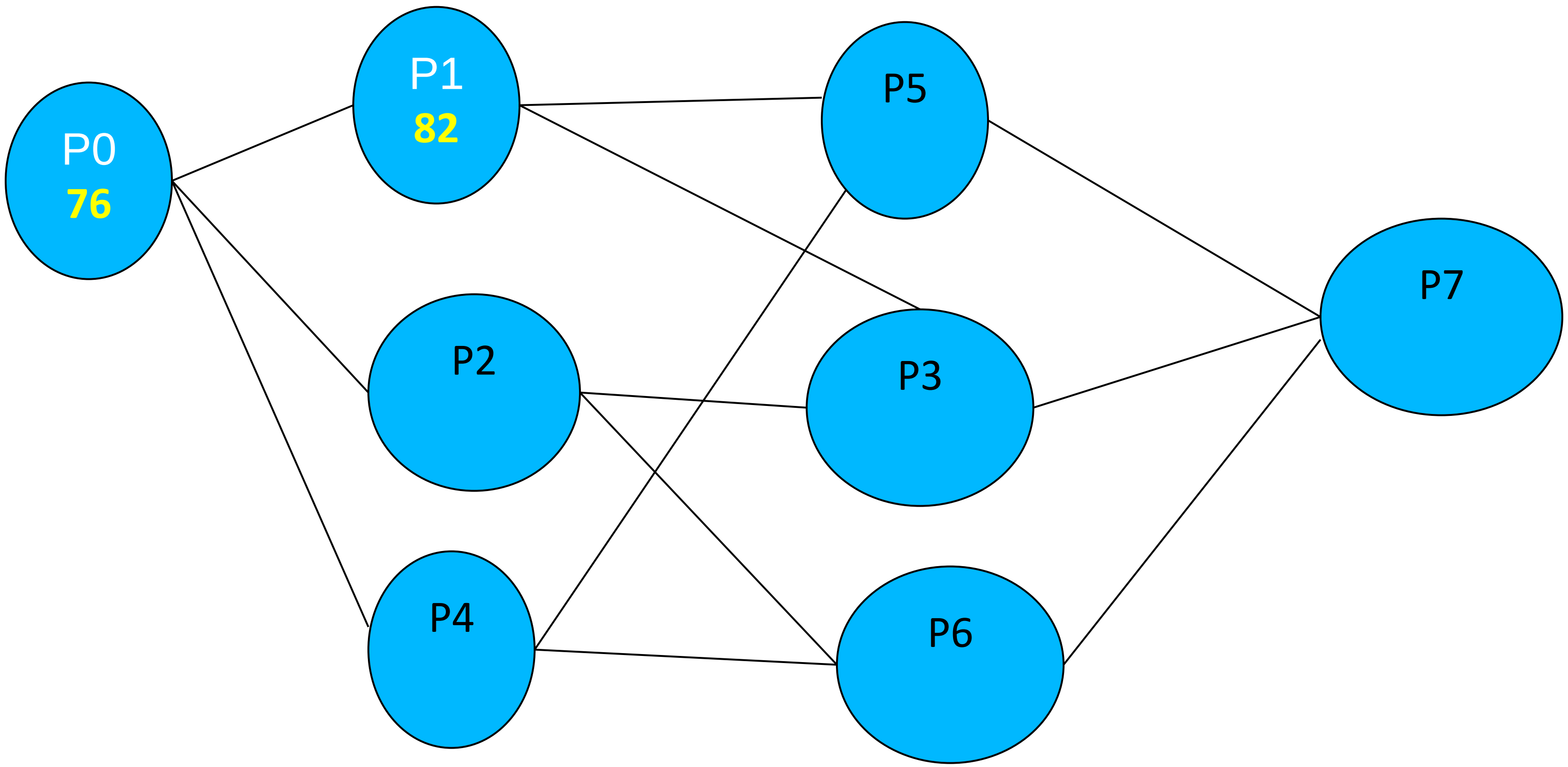
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (Hypercube SIMD)

Input

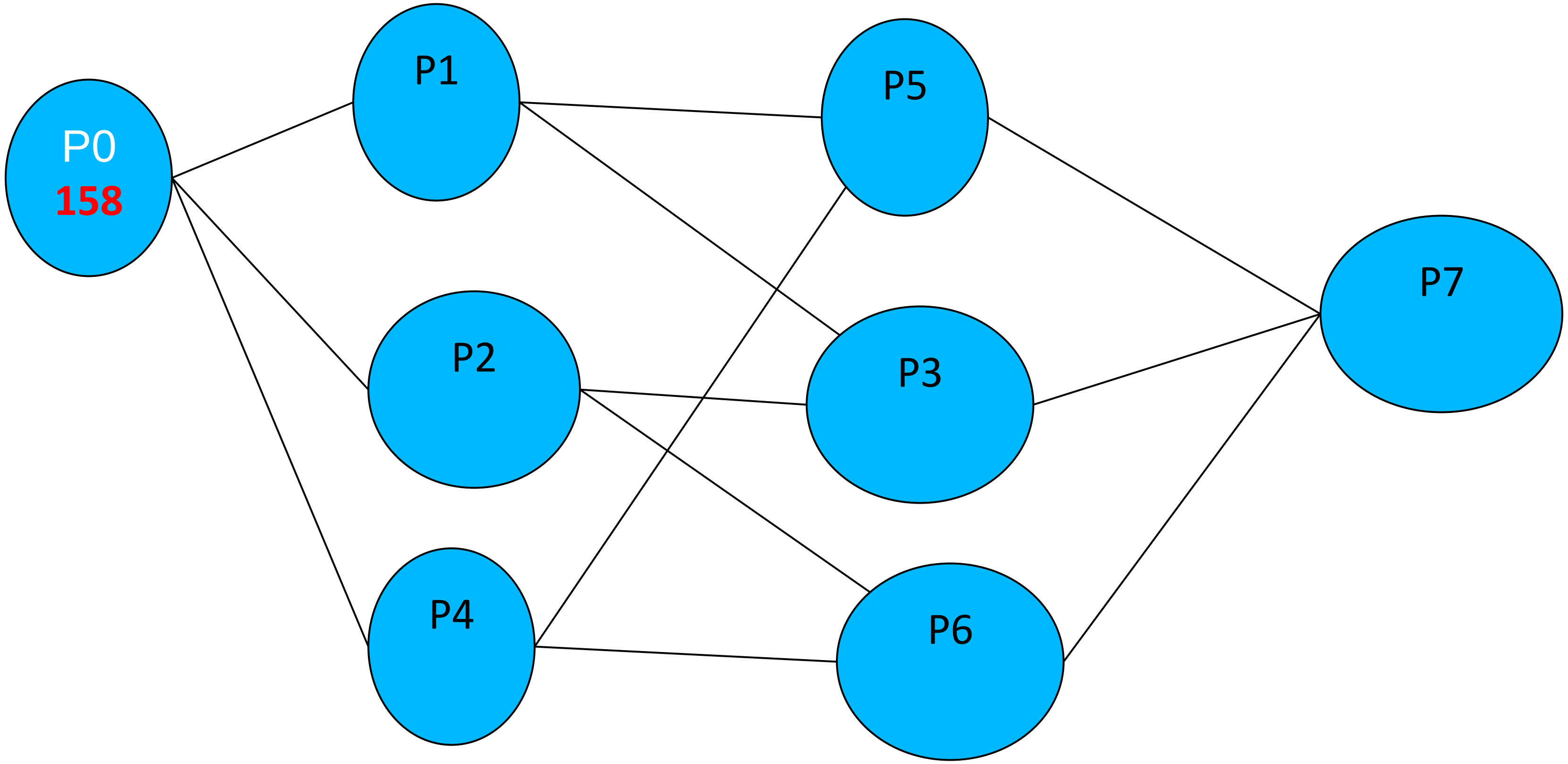
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (Hypercube SIMD)

Input

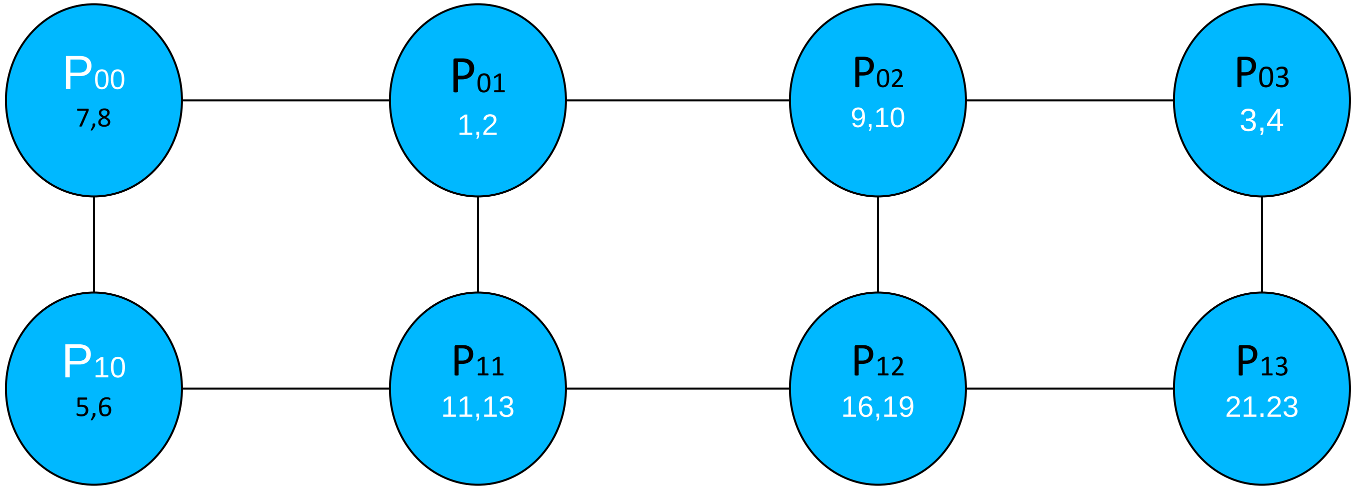
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (MESH SIMD)

Input

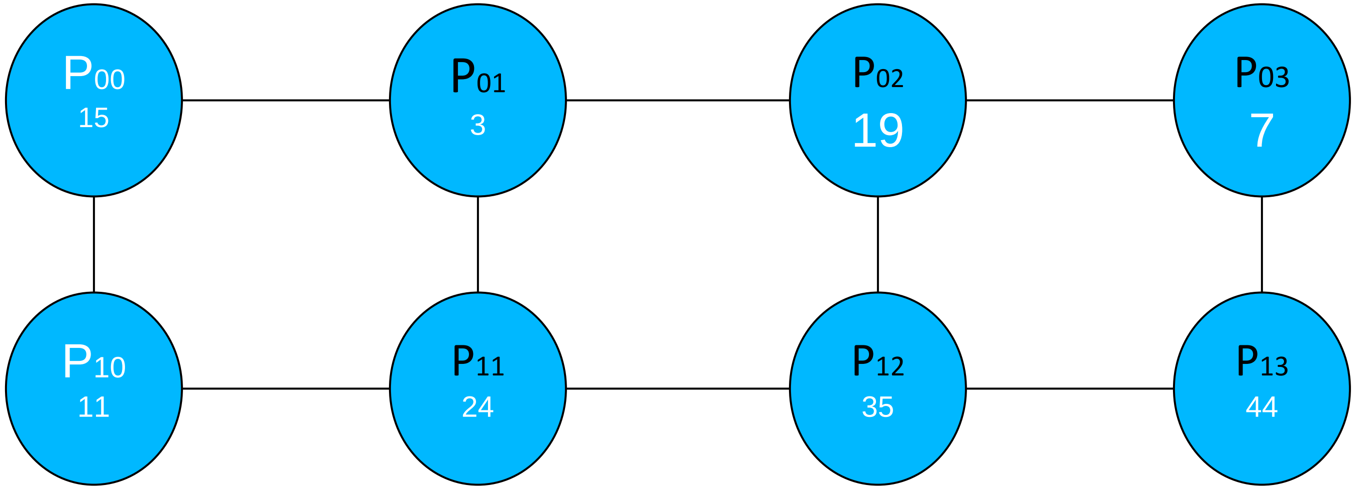
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (MESH SIMD)

Input

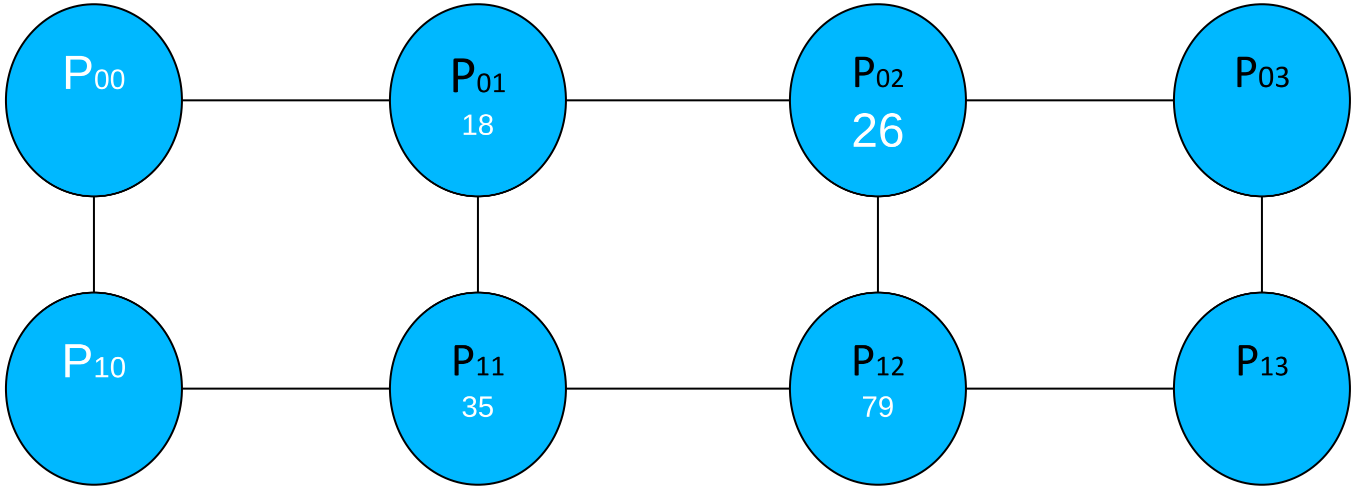
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (MESH SIMD)

Input

7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23

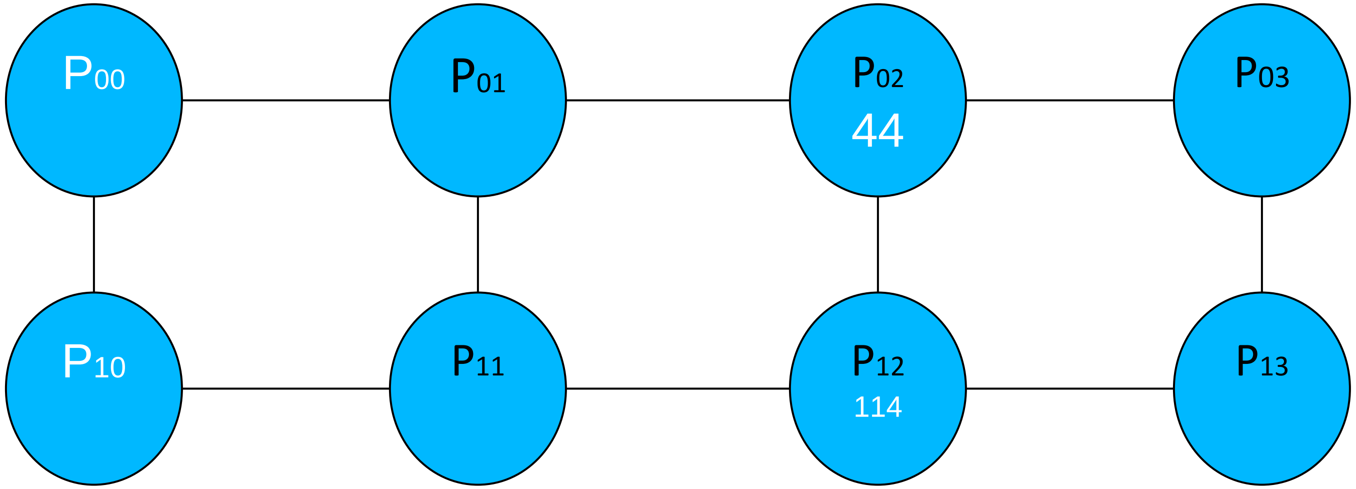


Microprocessor & Computer Architecture (μpCA)

Summation (MESH SIMD)

Input

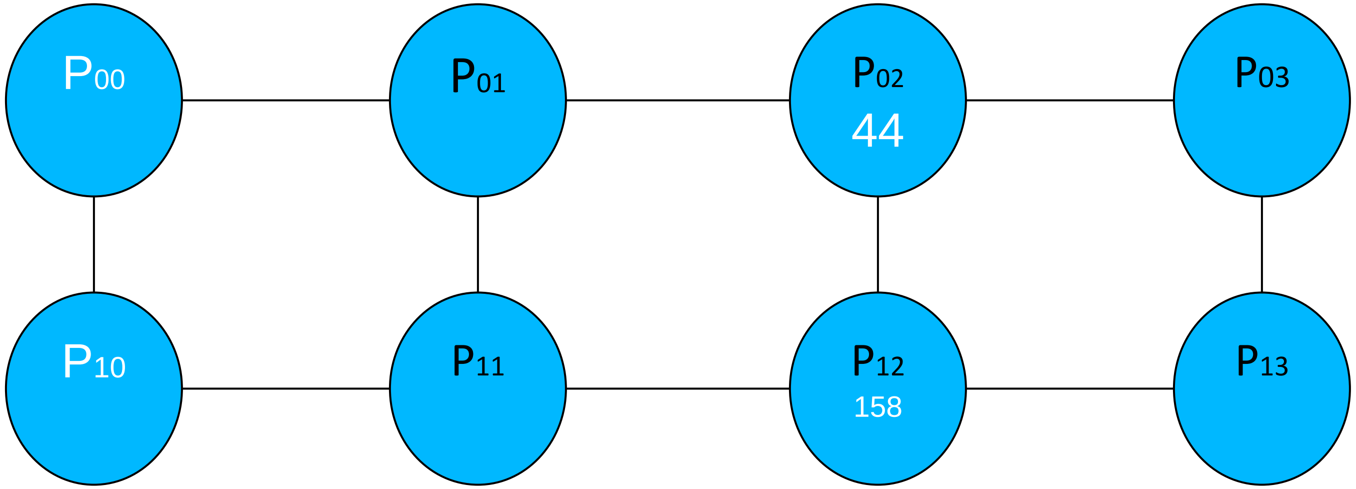
7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Summation (MESH SIMD)

Input

7	8	1	2	9	10	3	4	5	6	11	13	16	19	21	23



Next Session

Amdahl’s Law
&
Gustafson’s Law

PES UNIVERSITY ONLINE

THANK YOU

Dr. D. C. Kiran

Department of Computer Science and Engineering

dckiran@pes.edu

9829935135