COMPERATIVE STUDY OF DIFFERENT DATA PARALLEL APPROACHES IN FORM OF MATRIX MULTIPLICATION ALGORITHMS IN DIFFERENT ENVIRONMENT WITH SHARED MEMORY ARCHITECTURE

UNDER THE SUPERVISION OF
Prof. Utpal Kumar Ray
PROFESSOR

DEPARTMENT OF INFORMATION TECHNOLOGY
JADAVPUR UNIVERSITY

ABSTRACT

Regarding the implementation of the ongoing Project "Comparative study of different data parallel approaches in form of parallel matrix multiplication algorithms in different multiprocessor environment with shared memory architecture " we were set-in with the indepth study and analysis of the Matrix multiplication issues. Our primary target dealt with the comparison of different typical Multiprocessor environments (IBM P-SERVERS) by employing thread for furnishing data-parallelism .

GENERAL DISCUSSION ON MUTIPROCESSOR SYSTEM

Multiprocessors and Multicomputers

- A multiprocessor has more than one processor, with common memory shared between processors.
- A multicomputer has more than one processor, with each processor having local memory.
- In either case, processors may be on a common bus (close coupled), or distributed on a network (loosely coupled).

FLYNN'S TAXONOMY

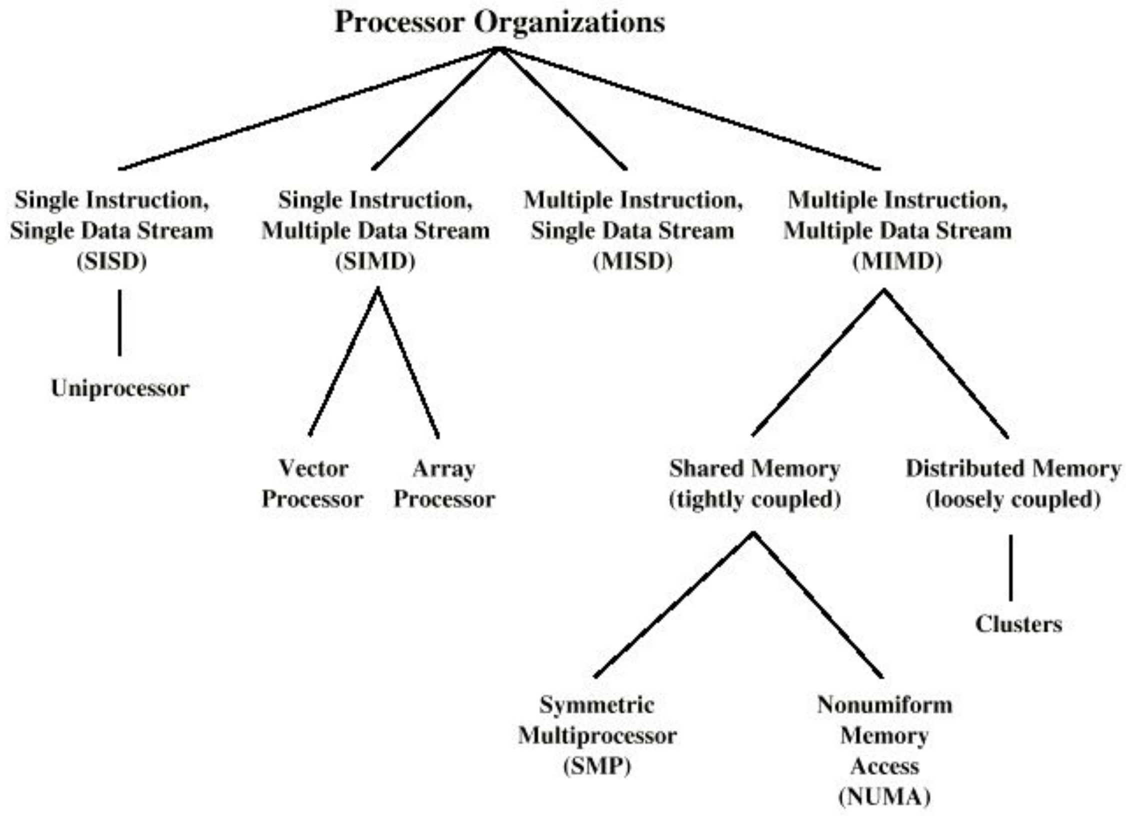
Computer system organisation described by two characteristics

Number of instruction streams
Number of data streams

Classifications

- Single instruction, single data stream – SISD
- Single instruction, multiple data stream – SIMD
- Multiple instruction, single data stream – MISD
- Multiple instruction, multiple data stream- MIMD

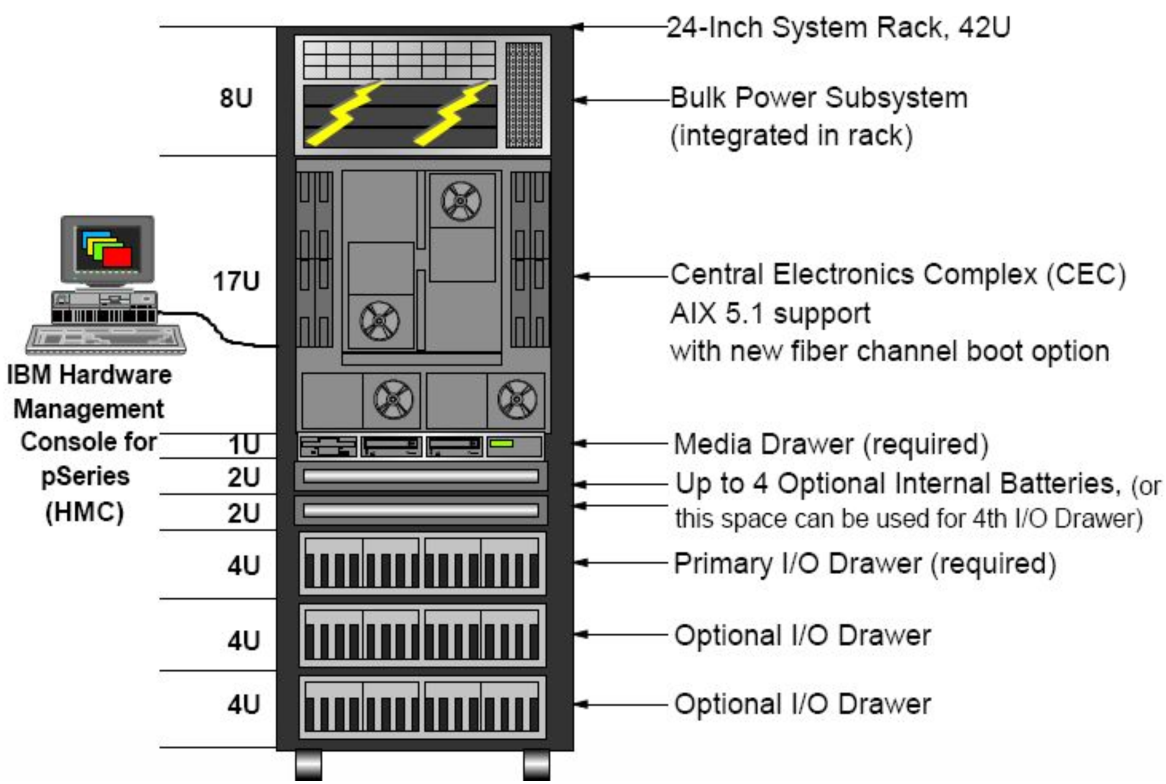
TAXONOMY OF PARALLEL PROCESSOR ARCHITECTURE



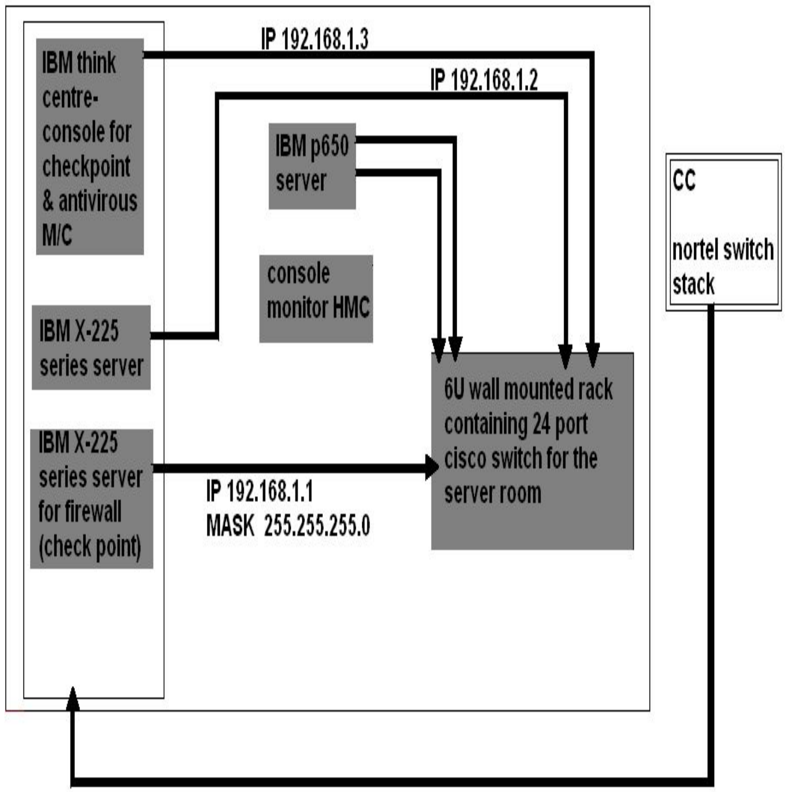
UNIX IBM P-SERVERS CONCEPT

When you need to control costs, while improving overall performance, availability and energy efficiency—consider the IBM System p™ family of servers both AIX® [UNIX®] and Linux® [Linux on POWER™ applications] operating system environments. Here's why System p is perfect platform for your mission and business critical applications and a unique server consolidation platform for your datacenter

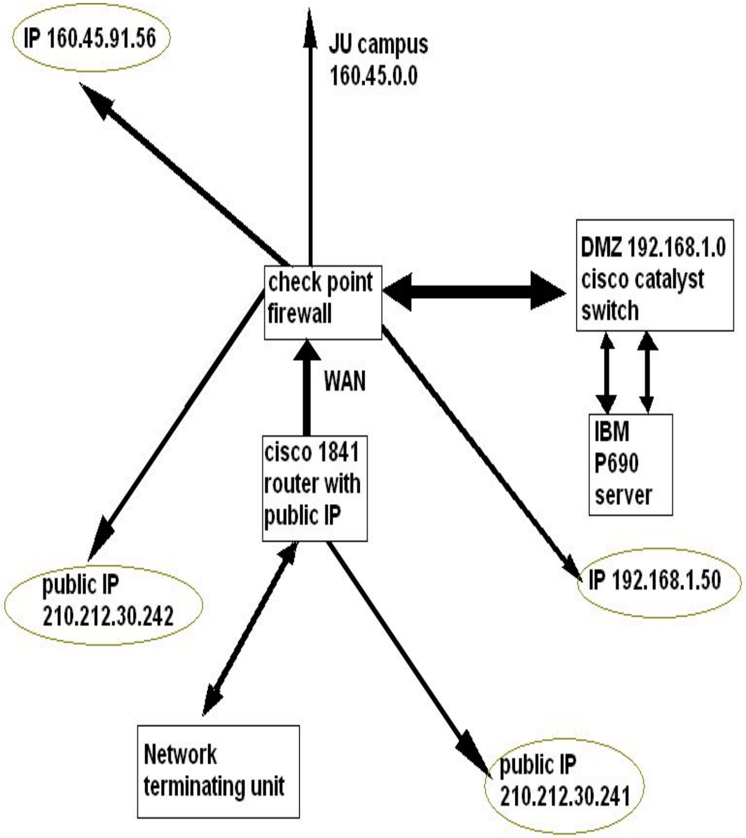
PICTORIAL DESCRIPTION



IBM p650


IP 192.168.1.3
IP 192.168.1.2
IBM think centre-console for checkpoint & antivirous M/C
IBM p650 server
console monitor HMC
IBM X-225 series server
IBM X-225 series server for firewall (check point)
IP 192.168.1.1
MASK 255.255.255.0
6U wall mounted rack containing 24 port cisco switch for the server room
CC
nortel switch stack

IBM p690


IP 160.45.91.56
JU campus
160.45.0.0
check point
firewall
DMZ 192.168.1.0
cisco catalyst
switch
WAN
IBM
P690
server
cisco 1841
router with
public IP
public IP
210.212.30.242
IP 192.168.1.50
Network
terminating unit
public IP
210.212.30.241

PARALLEL PROCESSING

Parallel computing

Parallel computing is the use of a parallel computer to reduce the time needed to solve a single computational problem. Parallel computing is now considered a standard way for computational scientists and engineers to solve problems in areas as diverse as galactic evolution, climate modeling, aircraft designing and molecular dynamics.

Parallel computer

A parallel computer is a multiprocessor computer system supporting parallel programming. Two important categories of parallel computers are multicomputer and centralized multiprocessors.

Parallel programming

Parallel programming is programming in a language that allows you to explicitly indicates how different portions of the computations may be executed concurrently by different processors

PARALLEL ALGORITHM DESIGN

Dataparallelism:

•Data Parallelism is a form of parallelization of computer code. It is meant to distribute computing across multiple processors in parallel computing environments. It contrasts to Task Parallelism as another form of parallelism.

Task parallelism:

•Task Parallelism is a form of parallelization of computer code. It is meant to distribute computing across multiple processors in parallel computing environments. It contrasts to Data Parallelism as another form of parallelism.

SPEEDUP AND EFFICIENCY

Speedup is the ratio between sequential execution time and parallel execution time

$$\text{Speedup} = \frac{\text{Sequential execution time}}{\text{Parallel execution time}}$$

The **efficiency** of a parallel program is a measure of processor utilization. We define efficiency to be speedup divided by the number of processors used.

$$\text{Efficiency} = \frac{\text{Sequential execution time}}{\text{Processors used} \times \text{Parallel execution time}}$$

AMDAHL'S LAW

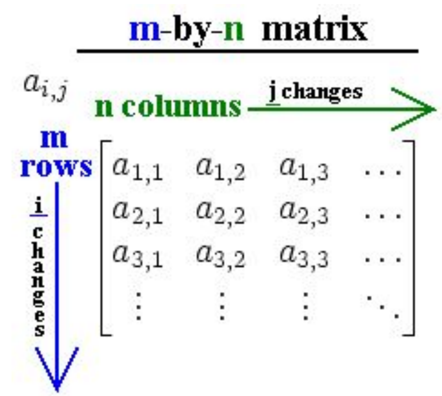
Let f be the fraction of operations in a computation that must be performed sequentially,

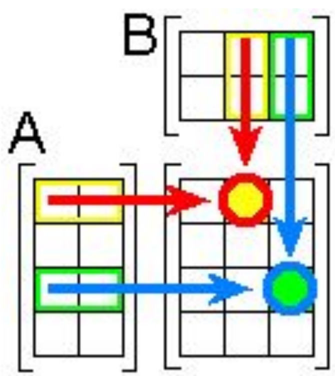
where $0 \le f \le 1$. The maximum speedup ψ achievable parallel computer with *p* processors performing the computation is

$$\psi \le \frac{1}{f + (1 - f)/p}$$

MATHEMATICAL MATRIX CONCEPT

In mathematics, a matrix (plural matrices) is a rectangular table of numbers or, more generally, a table consisting of abstract quantities that can be added and multiplied. Matrices are used to describe linear equations, keep track of the coefficients of linear transformations and to record data that depend on two parameters. Matrices can be added, multiplied, and decomposed in various ways, making them a key concept in linear algebra and matrix theory.





MATRIX MULTIPLICATION COMPUTATIONAL ANALYSIS

SEQUENTIAL MATRIX MULTIPLICATION

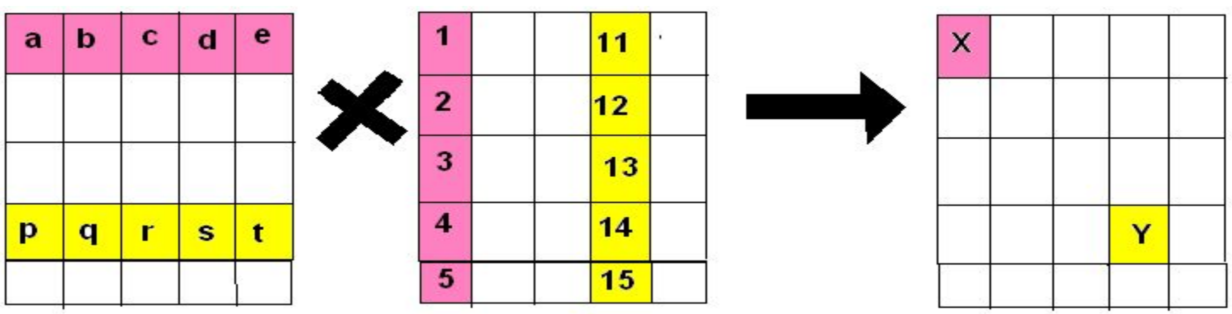
Here we implement two sequential matrix multiplication algorithms.

- 1. Iterative row oriented matrix multiplication algorithm
- 2. Recursive, Block-Oriented matrix multiplication algorithm

ITERATIVE ROW ORIENTED ALGORITHM

- The product of a n x *m* matrix *A* and an *m* x *n* matrix *B* is an *l* x *n* matrix *C* whose elements are defined by

$$c_{i,j} = \sum_{k=0}^{m-1} a_{i,k} b_{k,j}$$



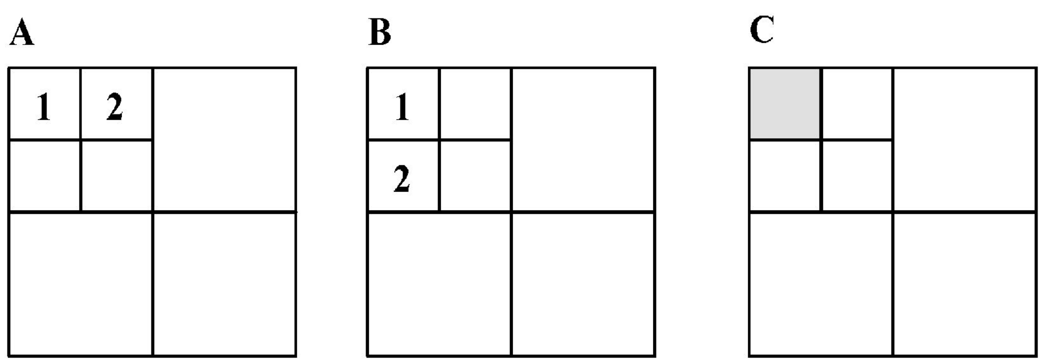
X=a*1 + b*2 + c*3 + d*4 +e*5

Y=p*11+q*12+r*13+s*14+t*

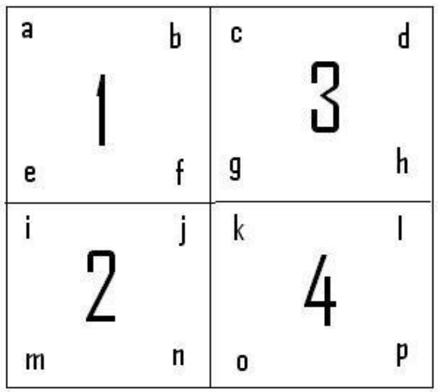
RECURSIVE, BLOCK-ORIENTED ALGORITHM

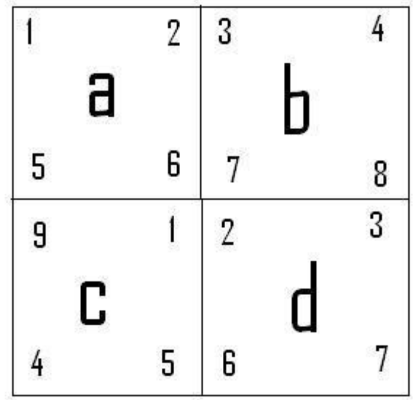
- In order to perform the matrix multiplication *A B,* the number of columns of *A* must be equal to the number of rows of *B.*
- Let's suppose *A* has *l* rows and m columns, while *B* has m rows and n columns. If we divide *A and B* into four smaller matrices

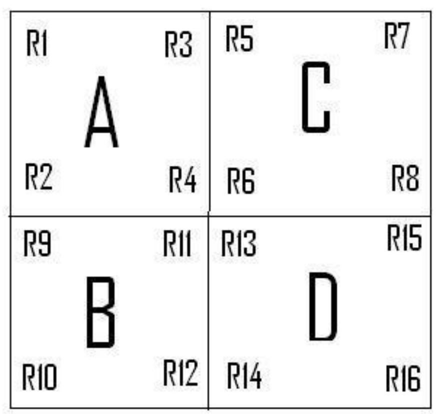
$$A = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix} \qquad B = \begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} \qquad C = \begin{pmatrix} A_{00}B_{00} + A_{01}B_{10} & A_{00}B_{01} + A_{01}B_{11} \\ A_{10}B_{00} + A_{11}B_{10} & A_{10}B_{11} + A_{11}B_{11} \end{pmatrix}$$



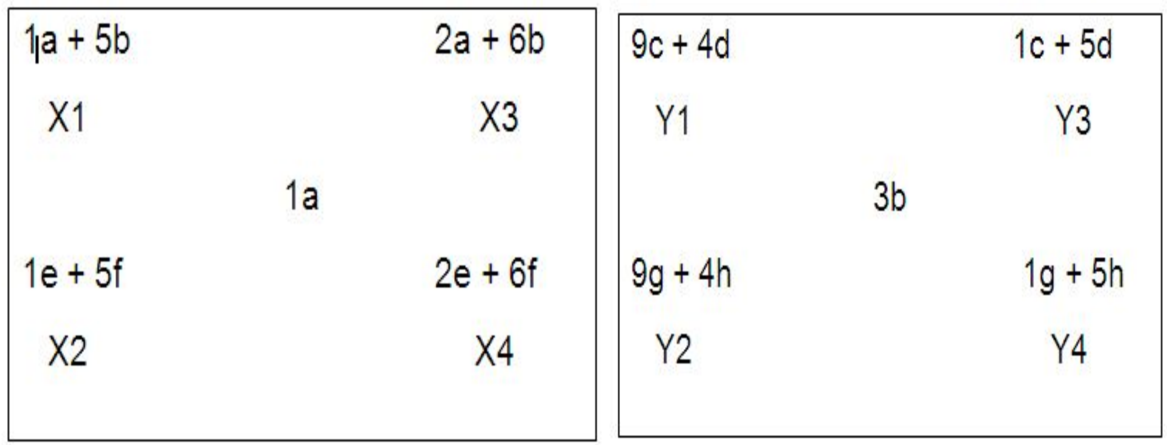
RECURSIVE, BLOCK-ORIENTED ALGORITHM







RECURSIVE, BLOCK-ORIENTED ALGORITHM



BLOCK A

R1 = X1 + Y1 R3=X3+Y4

R2 = X2 + Y2 R4=X4+Y4

RECURSIVE, BLOCK-ORIENTED ALGORITHM

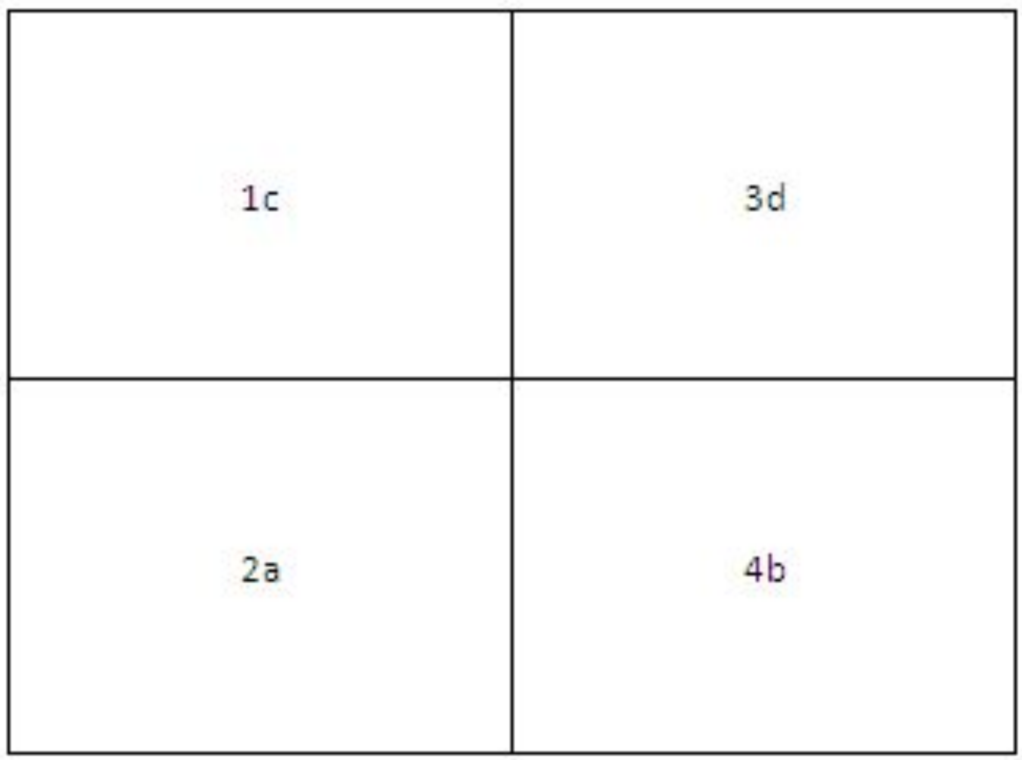
3i + 7j U1	4i + 8j U3		2k + 6l V1	3k + 7l V3
2c			4d	
3m + 7n U2	4m + 8n U4		2o + 6p V2	3o + 7p V4

BLOCK D

R13=U1+V1 R15=U3+V3

R14=U2+V2 R15=U4+V4

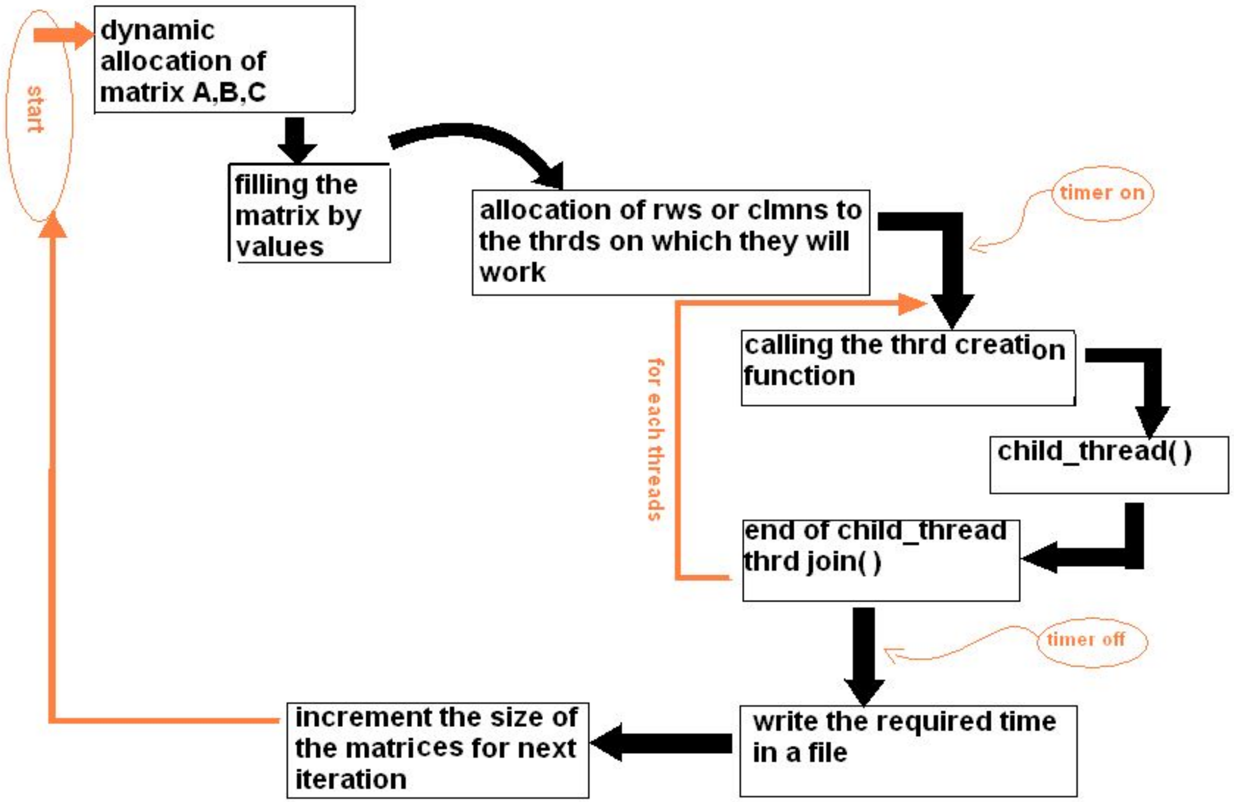
RECURSIVE, BLOCK-ORIENTED ALGORITHM



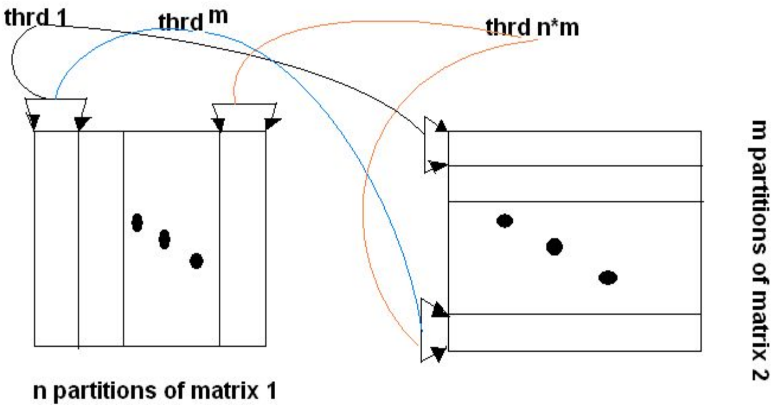
Block C

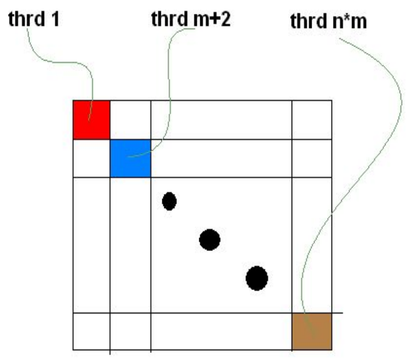
Block B

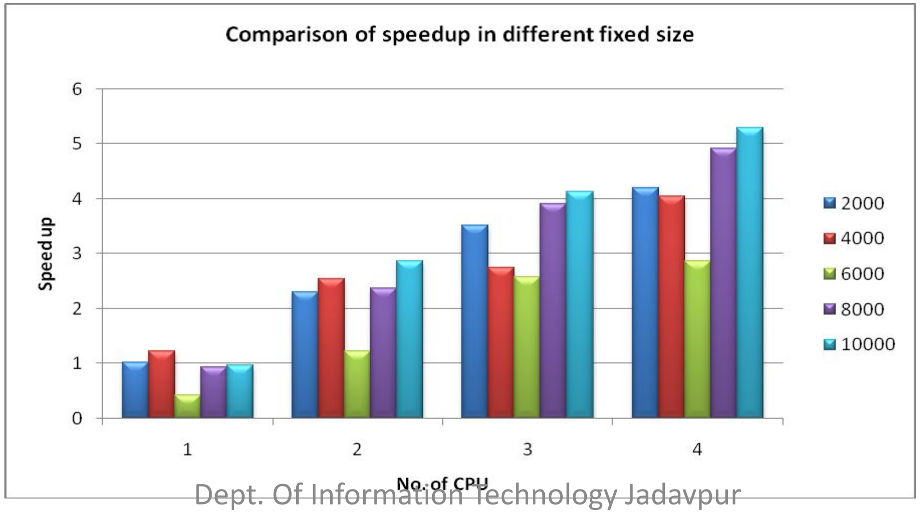
MATRIX MULTIPLICATION PARALLELIZATION



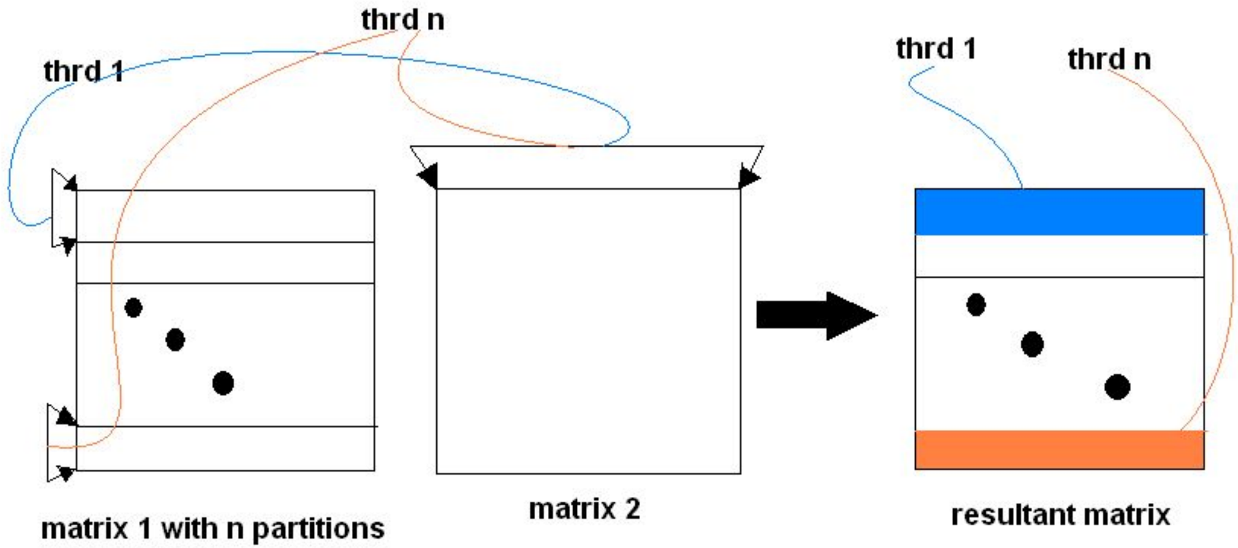
DIRECT PARTITIONING MATRIX MULTIPLICATION ALGORITHM



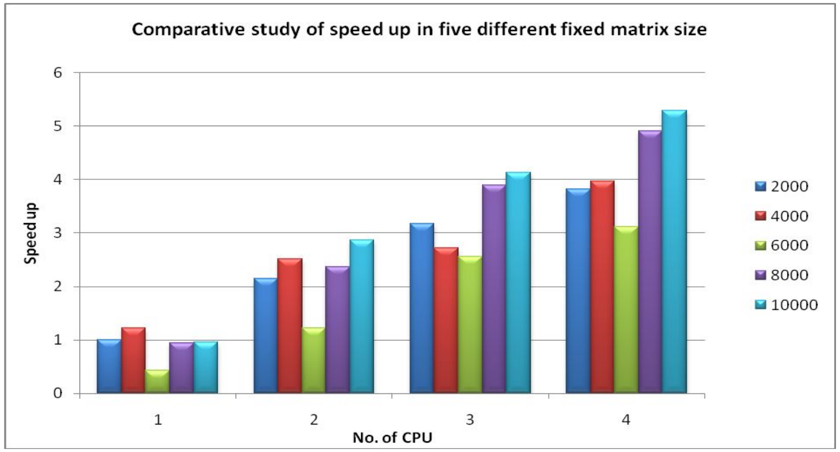


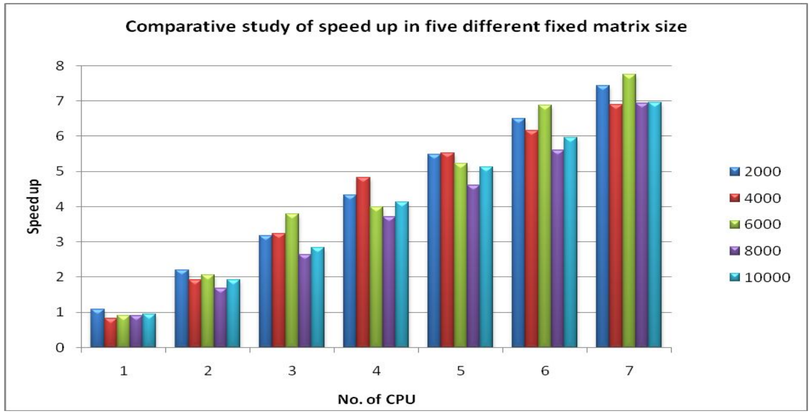


ROW ORIENTED BLOCK STRIPED MATRIX MULTIPLICATION ALGORITHM



ROW ORIENTED BLOCK STRIPED MATRIX MULTIPLICATION ALGORITHM





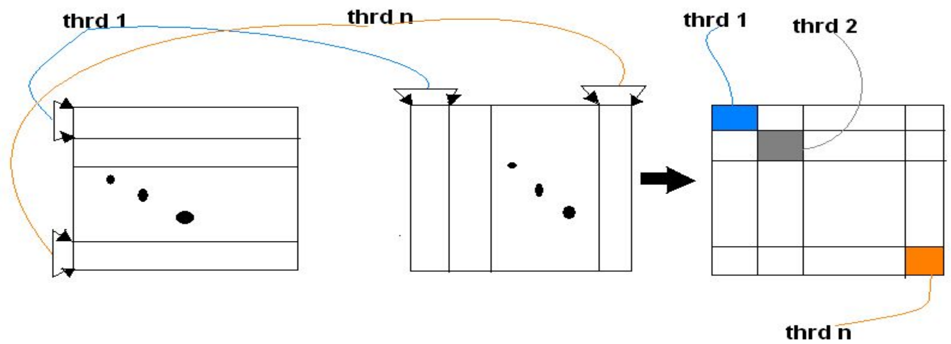
COMPARATIVE CONCLUSION

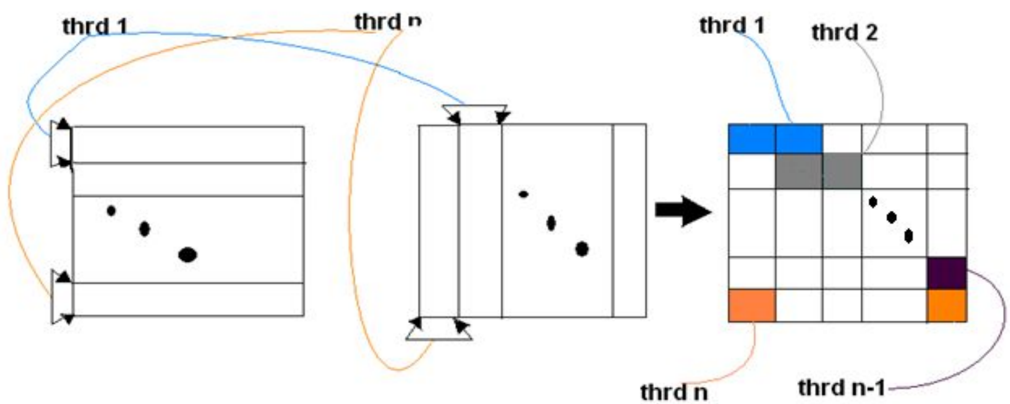
- Here we compare the row wise block striped parallel matrix multiplication algorithm in two machine IBM p650 and IBM p690 for different fixed matrix size.

- We observe:

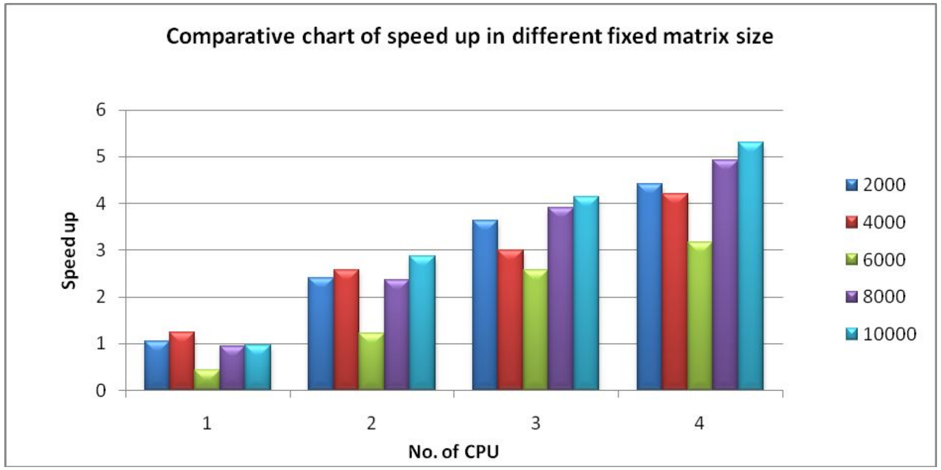
- Speed up in IBM p650 is more than IBM p690 for every considered size except matrix size 2000.
- Time taken in sec, for computational purpose is more in IBM p650 than IBM p690 except matrix size 4000.
- Sometimes we achieved super linear speedup

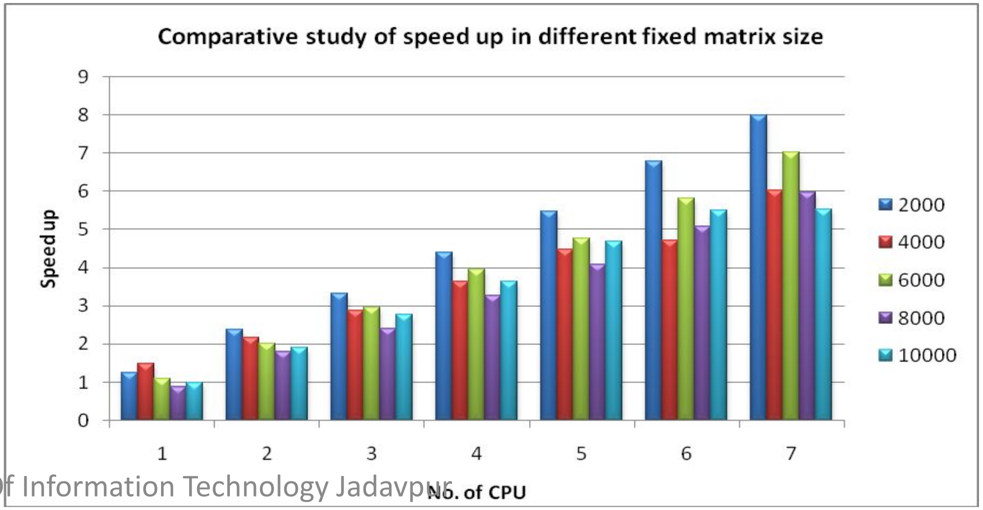
BROADCAST BROADCAST MATRIX MULTIPLICATION





BROADCAST BROADCAST MATRIX MULTIPLICATION





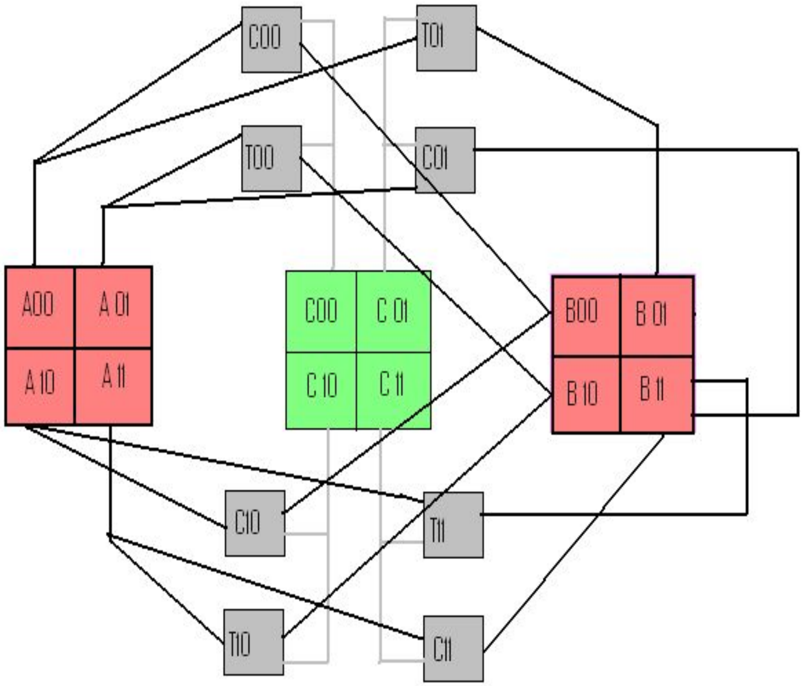
COMPARATIVE CONCLUSION

- Here we comapare the broadcast broadcast parallel matrix multiplication algorithm in two machine IBM p650 and IBM p690 for different fixed matrix size.

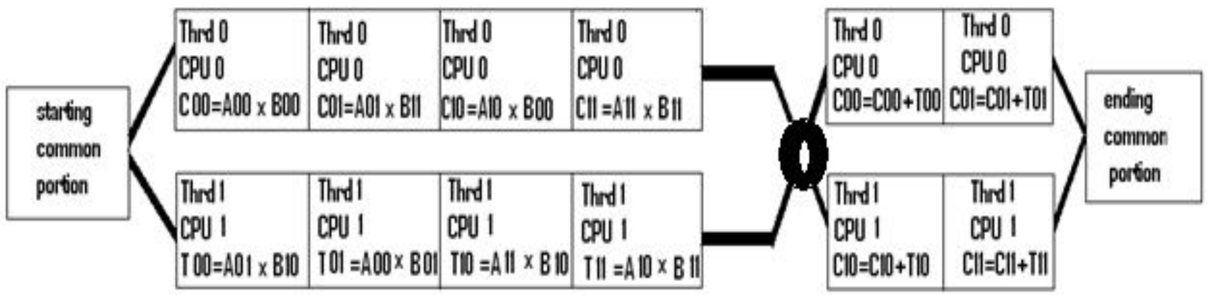
We observe:

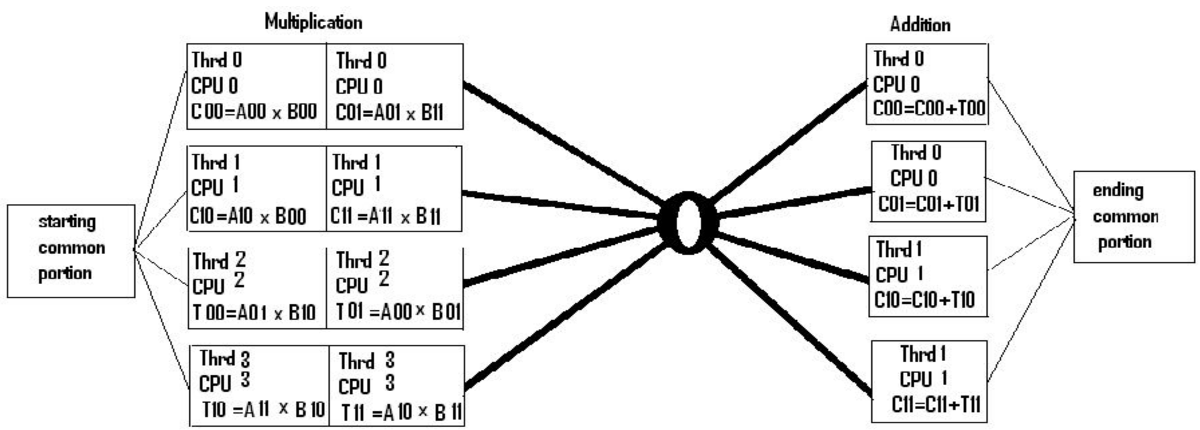
- Speed up in IBM p650 is more than IBM p690 for every considered size except matrix size 2000.
- Time taken in sec, for computational purpose is more in IBM p650 than IBM p690 except matrix size 4000.
- Sometimes we achieved super linear speedup.

DIVIDE AND CONQUER MATRIX MULTIPLICATION ALGORITHM

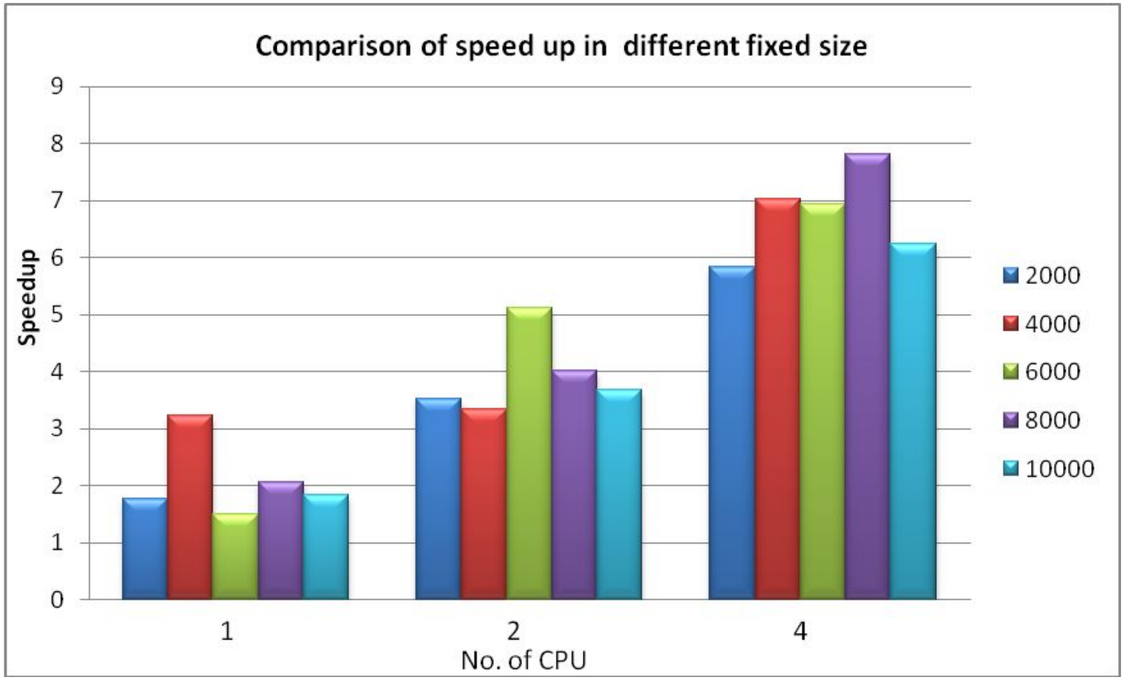


DIVIDE AND CONQUER MATRIX MULTIPLICATION ALGORITHM





DIVIDE AND CONQUER MATRIX MULTIPLICATION ALGORITHM



CONCLUSION

- The Matrix multiplication, which incorporates a huge amount of data handling and a sophisticated mathematical approach, enables us to implement the data parallelism.
- In order to parallelizing the Matrix multiplication we used the p-threads.
- Both parallel and sequential algorithms thoroughly studied on different multiprocessor machines (IBM p650, IBM p690). IBM p650 provides more speedup whereas the execution time for IBM p690 is lesser.
- Some of the obtained speedups found to be super linear.
- Around the matrix size 3000,t he required rate of time-increase Suddenly jumps up then back to normal. It may take place due to the cache hit rate falls dramatically as the matrix dimension reaches up around that size which lowers the CPU performance.
- On using single thread it has been observed, that sometimes it takes less time than the sequential approach. It may happen because when this process is assigned under a thread it gets higher priority by the system.
- For better improvement instead of taking the multiplication time in sec. We can consider the exact CPU cycles or the required FLOP.

THANK YOU

- 1.ARNAB RAY (Roll no. 032847)
- 2.TIRTHANKAR BHADURY (Roll no. 032849)
- 3. PANKAJ SAHA (Roll no. 032850)

Students of final year of B . Info. Tech.
Dept. Of Information Technology Jadavpur University