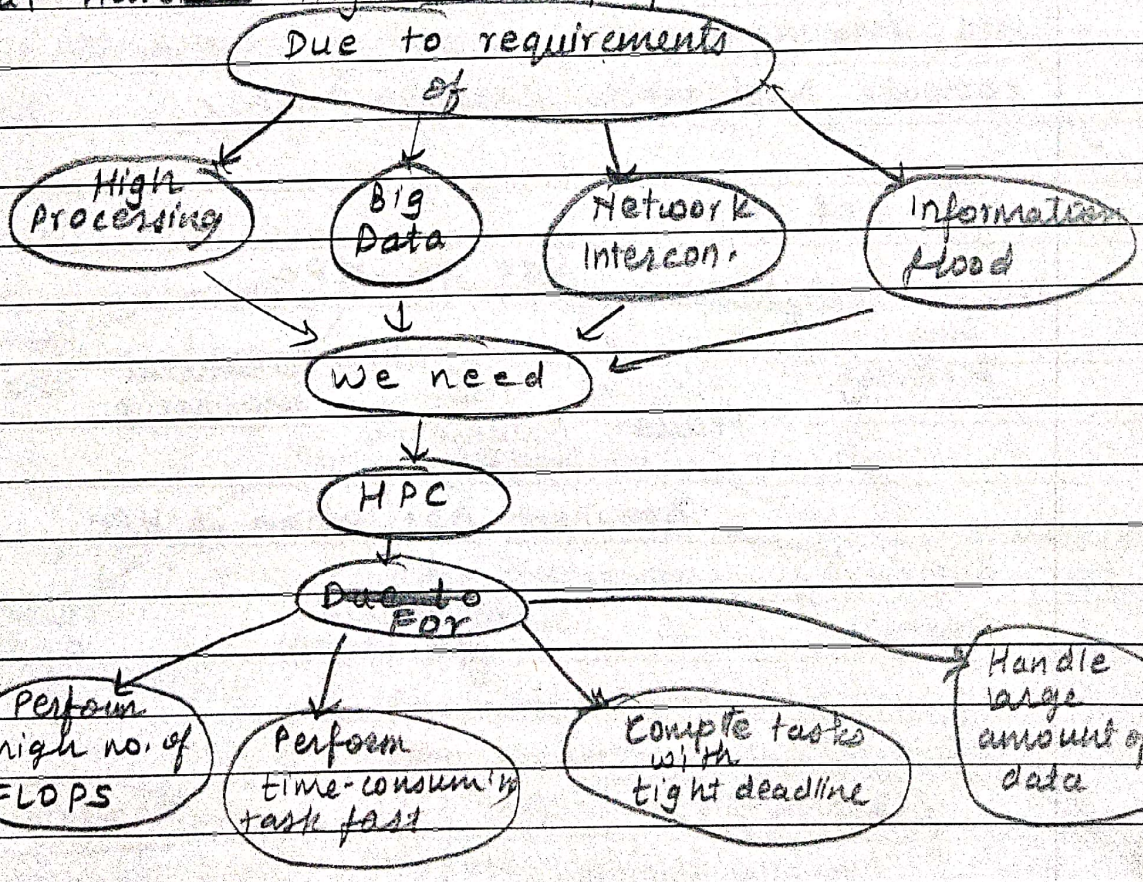
HIGH PERFORMANCE COMPUTING

UNIT 1: INTRODUCTION TO PARALLEL COMPUTING

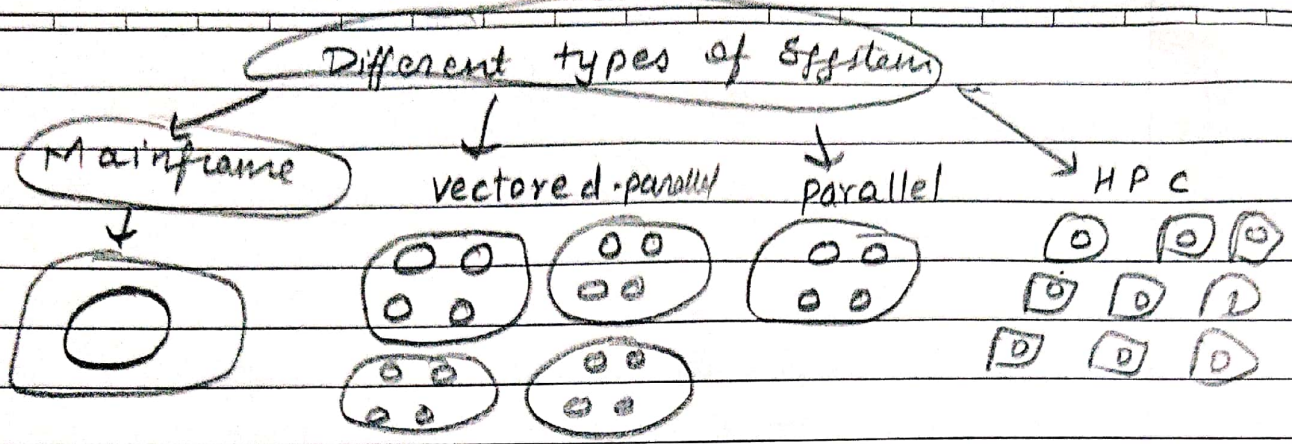
Q) What is HPC:-

- Refers to performing computation operations collaboratively on multiple computers that have higher level performance in terms of throughput.
- Keywords:- Computation collaboratively on multiple comps. that have [ ] higher level performance.



- HPC is mechanism of fast computation in parallel on computing nodes on a very fast computing network.

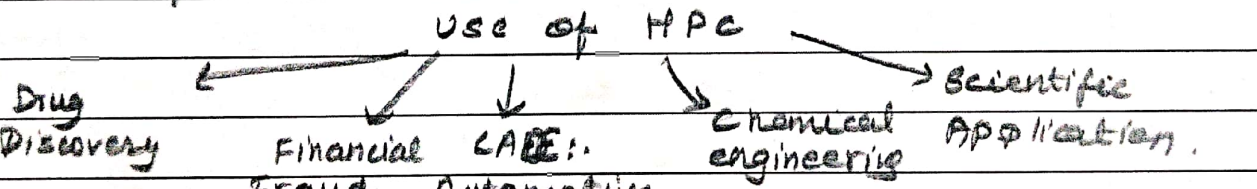
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

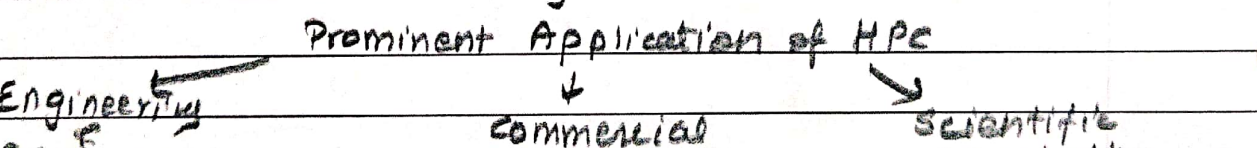


O : - is a computation Node.

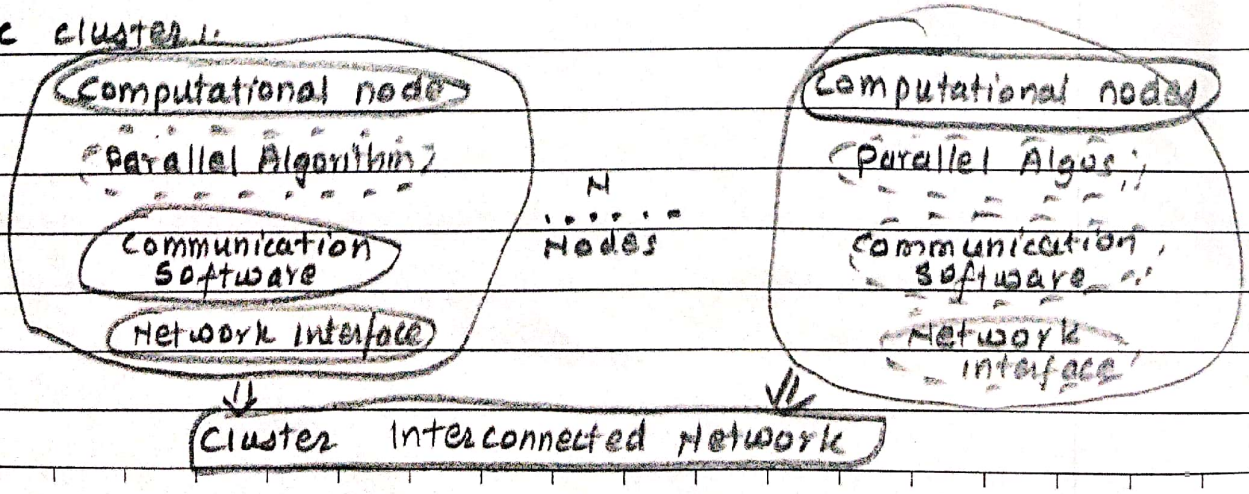
- HPC uses large no. of high ~~capacity~~ efficiency computational node, ~~various~~ whereas rest use less nodes but high end.
- HPC uses aggregate computing power for handling compute and data intensive tasks.
-

• Uses of HPC:





• HPC cluster:



Types of parallelism → Bit-Level, Instruction-Level, Task-Level, Data-Level

Motivating parallelism :-

Factors that motivated Parallelism

- The Computational/Power Argument → Making increase in translators & FLOPS
- Memory/Disk Speed Argument → Reducing bottleneck between CPU & Memory
- Data communication/Network argument. → Parallel Algorithms for data mining, web scraping.

Implicit Parallelism :-

- Helps programmer write programs without any concern for exploitation of parallelism.

Implicit Parallelism

- What? → Compiler, Runtime System help achieve parallelism also the hardware sys.
- Helps Programmer because → Programmer writes normal sequential code that turns into parallel code.
- Implicit Parallel mechanism are used in → Pipelining & Superscalar execution; VLIW Processors

Pipelining & Superscalar execution :-

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

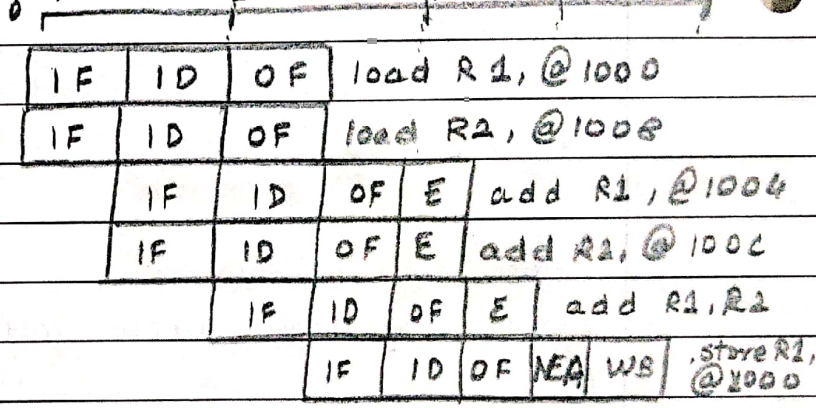
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Enhance Processor Performance by
- Implementing instruction-level parallelism
- Subdivide pipeline into multiple stages.

- In superscalar processor multiple instructions are issued per cycle and multiple results are generated by multiple pipelines per cycle.
- Superscalars designed to exploit instruction-level parallelism.
- Superscaling execution of code:

1) load R1, @1000
2) Load R2, @1008
3) add R1, @1004
4) add R2, @100C
5) add R1, R2
6) store R1, @2000

(i) Code



(ii) Execution schedule

IF: Instruction Fetch. ID: Instruction Decode. OF: Operand Fetch
E: Execution. WB: Write Back. NA: No Action

- [Describe Execution schedule: (i) Instructions issued at same clock cycle (t = i) are mutually exclusive. (ii) Instructions issued after a cycle are not mutually exclusive.]

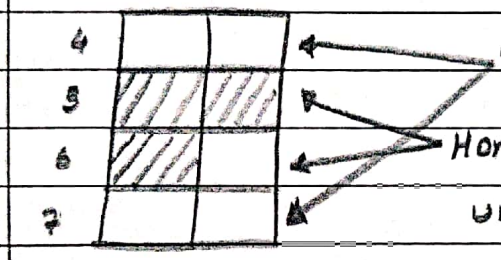
Superscalar execution depends on 3 things

1. True data dependency
   - Execution of one instr. depends on result of prev.
2. Resource Dependency
   - Two instr. compete for single resource
3. Branch / Procedural Dependency
   - In exec. of conditional statement we don't know the flow of execun. As a result scheduling instr. a priori may lead to errors.

LUSDBP SDLUBP BPLUSD

- To tackle branch dependency speculative scheduling is done.
- control independent instructions are moved up.
- Main superscaling job is to detect & schedule concurrent instructions.

Superscaler schedules instruction
- in order → Instructions executed as written
- out of order → instructions are rescheduled by processor by execun.



Very Long Instruction Word Processors

- Helps exploit parallelism on instruction-level like superscales.
- Compiler is the heart of ~~VLIW~~ VLIW processors.
- Resource availability and data dependency is checked during compile time.
- Techniques used :- BP SD LU [Acronym :- BPLUSD]
  - BP: Branch Predication
  - SD: Speculative decomposition
  - LU: Loop Unrolling
- Instructions that can run concurrently are combined into group or bundles and paraceled off to processor as a single long instruction word.

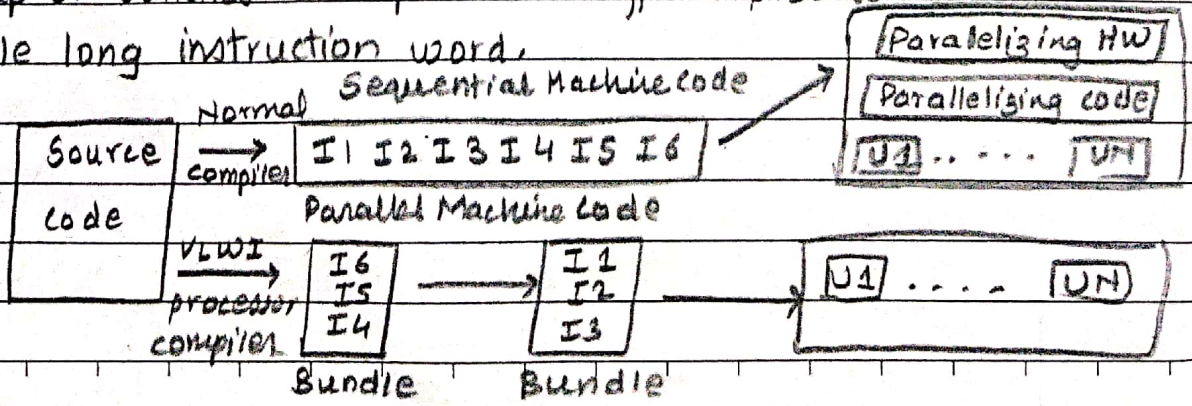
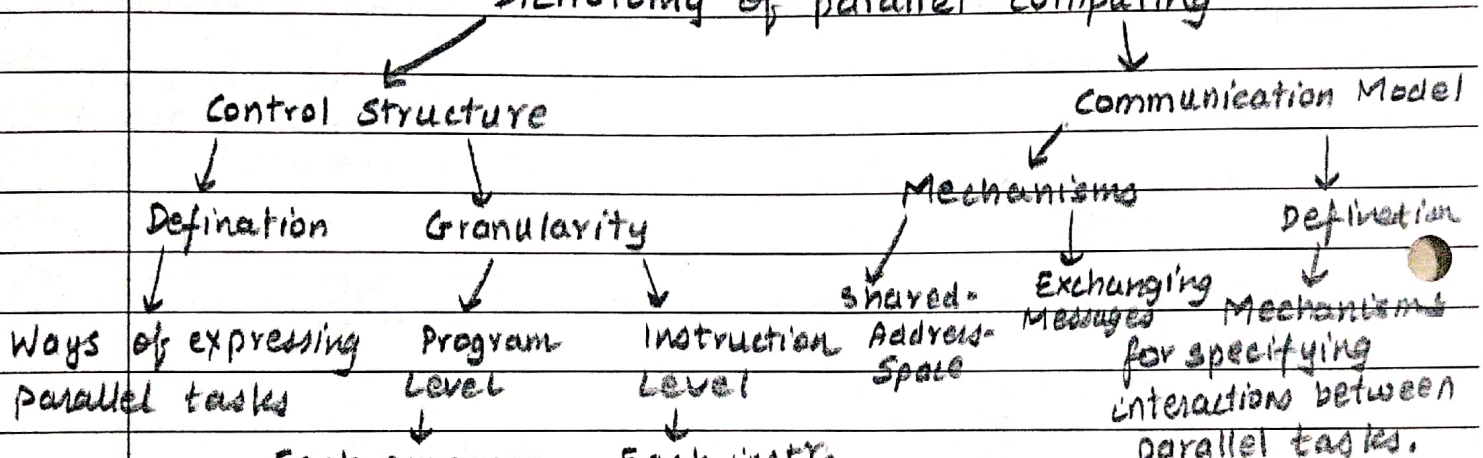


Fig: Working of VLWI processors

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

- Drawbacks :- (1) Dynamic program unavailable to make schedules.
(2) Stalls on fetching data due to cache miss is unpredictable.

@ Dichotomy of parallel computing :-

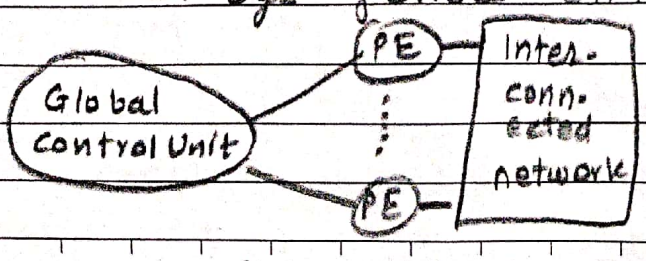


• Control Structure :-

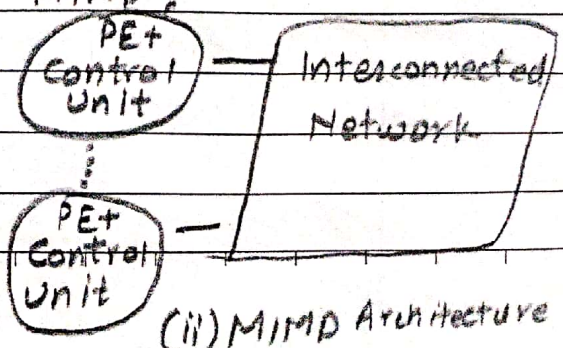
- Based on granularity various architectures are proposed

Sequential	Parallel	
SISD Single Instruction, Single Data	SIMD Single Instruction, Multiple Data	This is called Flynn's taxonomy.
MISD Multiple Instru?, Single Data	MIMD Multiple Instruction, Multiple Data	

Parallel sys. follow SIMD & MIMD.



(i) SIMD Architecture



(ii) MIMD Architecture

• Communication Model :-

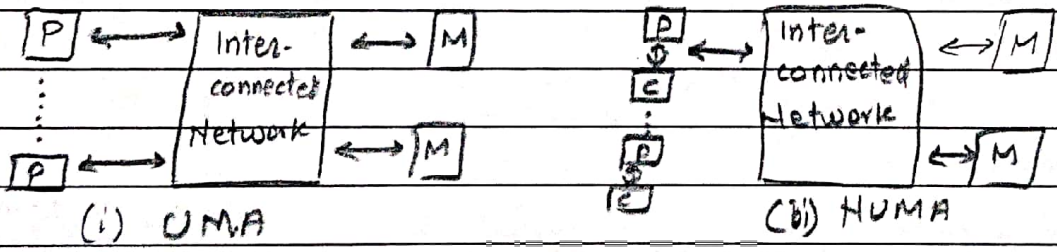
1) Shared-address-space :-

Classified as

Uniform Memory Access [UMA]
↓
Time taken to access word is identical

Non-Uniform Memory Access [NUMA]
↓
Some words require different time to access.

- Set of all possible physical address is called address space of that processor.
- Shared-address-space is a platform where all processors access common data space.



2) Messaging ~~Platforms~~ Passing Platforms :-

- P processors with seperate address space communicate.
- Processors interact with each other through messages.
- Single node is complete in a sense. It is a single processor or shared-address-space multiprocessor.
- Operations consits of
  (i) send (ii) Recieve (iii) ID (iv) Number of processes.

• Physical organization of parallel platforms :-

ϕ Architecture of an ideal parallel computer :-

- Random Access Machine
  - what? → A simple model of computation
  - Memory → A seq. of unbound regs.
  - Control unit → Holds program [Numbered list of statements]
  - Program counter → determines code to be exe. nxt.

- RAM can be made parallel by adding p processors and global memory of unbound size which is UMA type.
- This is Parallel Random Access Machine [PRAM]
- Processors :- (i) Work in same address space. (ii) Have different cycle. (iii) Same clock.

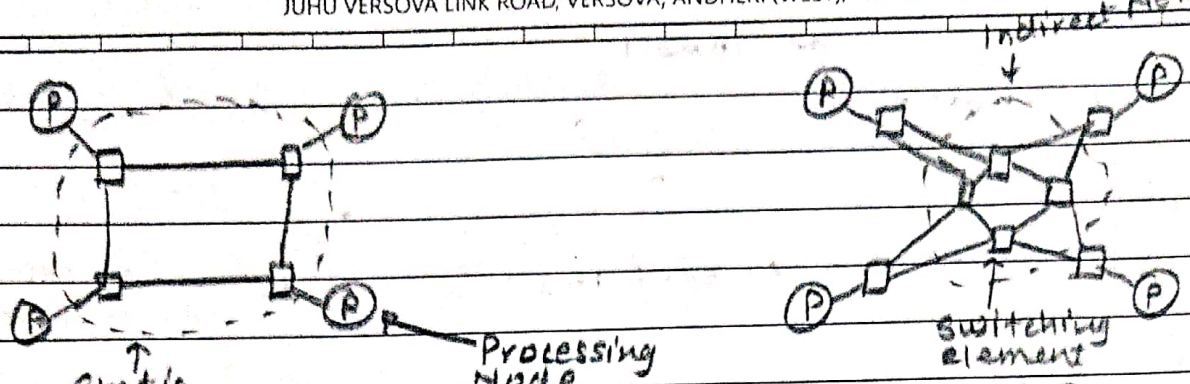
PRAM Class
- EREW [Exclusive Read, Exclusive write] (weakest)
- ERCW [Exclusive Read, Concurrent Write]
- CREW [Concurrent Read, Exclusive Write]
- CRCW [Concurrent Read, Concurrent write] (strongest)

Exclusive :- Access is exclusive only to one resource
Concurrent :- Multiple Access.

- Consider a PRAM with p processors & m global memory.
- Set of switches connect processors to memory.
- A word cannot be accessed by more than 1 processor hence the total no. of switches required = $\theta(mp)$
- This is not possible, hence PRAM cannot be implemented practically.

Q) Interconnection network for parallel computers :-
- ~~Switch~~ Switches
  - Provide support for
    - Routing
    - Multicast
    - Internal buffering
  - Help in interconnection
  - Link → physical medium like wires, optical fibre.

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.



(a) Static network (b) Dynamic network

Fig:- Types of interconnected

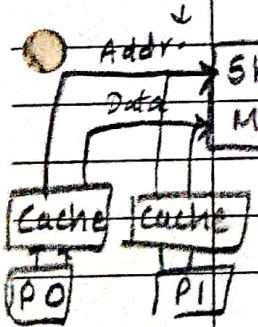
- Mapping of input to output is the basic functionality provided by switch.

~~Cost of mapping hardware~~
Costs of hardware

- Mapping hardware [Grows Square of degree of switch]
- Peripheral [Grows linearly as the degree]
- Packaging [Grows linearly with pins]

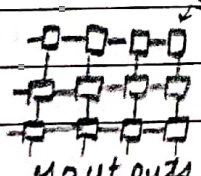
Network Topologies

Bus-Based:



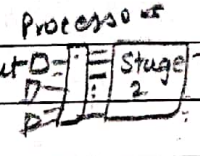
- Cost ∝ Nodes, Bus interface
- Non-blocking network [No one block each other]
- Scalable in performance but not in cost

Cross-Bar switch



- Simple way to connect p processors to m memory banks.

Multistage



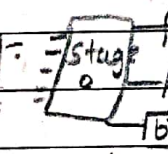
- Scalable in perfor. like cross-bar and less cost like bus-based
- Instance is Ω network

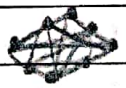
Let p : no. of I/Os then

$$j = \begin{cases} 2i & 0 \le i \le p/2 - 1 \\ 2i + 1 - p & p/2 \le i \le p - 1 \end{cases}$$

If true then link bet^n i & j exists.

Completely Connected Networks





- Non blocking

Star Network

- central processor may bottleneck

Linear Arrays, Meshes, k-d meshes:

1) 
Linear array with wraparound link,

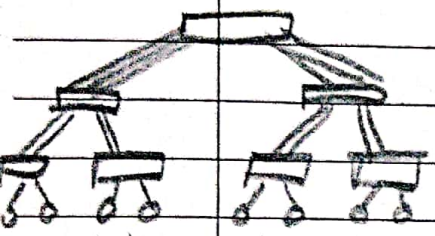
2) 

Meshes

3) In k-d meshes dis dimen^n k = no. of nodes along a d dimension

Another one .... tried hard to fit in [I tried so hard ....]

7) Tree-based Networks :-



Fat tree network

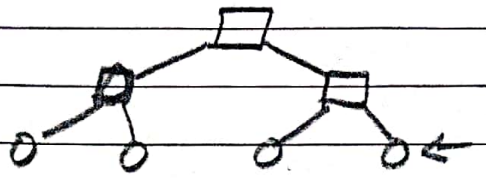


Fig: Dynamic.

□ :- Switching nodes
○ :- Processing nodes.

← only leaf nodes are processing nodes.

- Static tree based shall have no switches.

8) Static interconnected Network costs :-

Costs Associated with →
- Diameter
- Connectivity
- Bisection-width and Bisection-bandwidth
- Costs :- No. of wires required

- Max dist betⁿ two processors.
- Different Networks have different diameter.
- Some network diameter :-
  1) Fully-connected :- 1
  2) Star-connected :- 2
  3) Ring-network :- $\lfloor p/2 \rfloor$

- Measure of multiplicity of paths between any two processors.
- Reliability ∝ Connectivity
- If network breaks in 2 disconnected network by removing min. no. of arcs it is known as arc connectivity.

- Min. no. of links to remove to divide network in equal parts is called bisection width of a network.
- Vol. of communication allowed betwⁿ any 2 halves of network is bisection bandwidth.

9) Dynamic Interconnected networks :-

- Cost : Link cost + switch cost.
- Bisection determined by drawing various equal-partition and selecting min. no. of edges crossing the partition.

10) Cache coherence :-

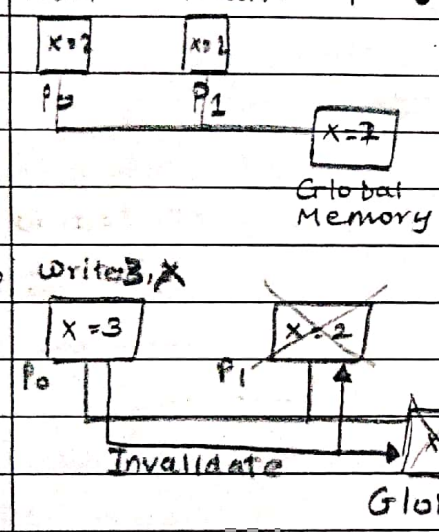
- In case of shared address space processors additional hardware is required to keep multiple copies of data consistent with each other.

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

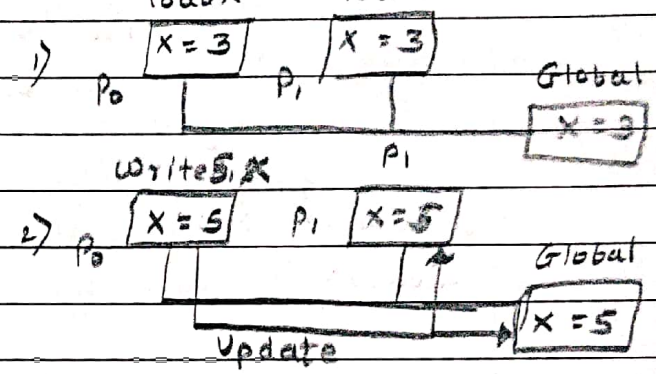
- This problem is known as cache coherence.
- To solve this additional protocols & hardware required.
- In multiprocessor sys. multiple processors modify values making cache coherence more complex.

PROTOCOLS USED FOR COHERENCE

Invalidate Protocol [invalidates all other cache copies]



Update protocol [Updates all other cache copies with the same address line]



- Whenever processors tries to modify var either invalidate or update proto. shall be fired to maintain cache coherence.

Q. Snoopy Cache System :-

- Coherence protocols are implemented using Snoopy buses ~~cache system~~.
- Coherency controller monitor / snoops bus transactions to maintain cache coherency protocols.
- Bus designed to constantly monitor caching events betn processor and memory m, they are called as snoopy coherence protocols.

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

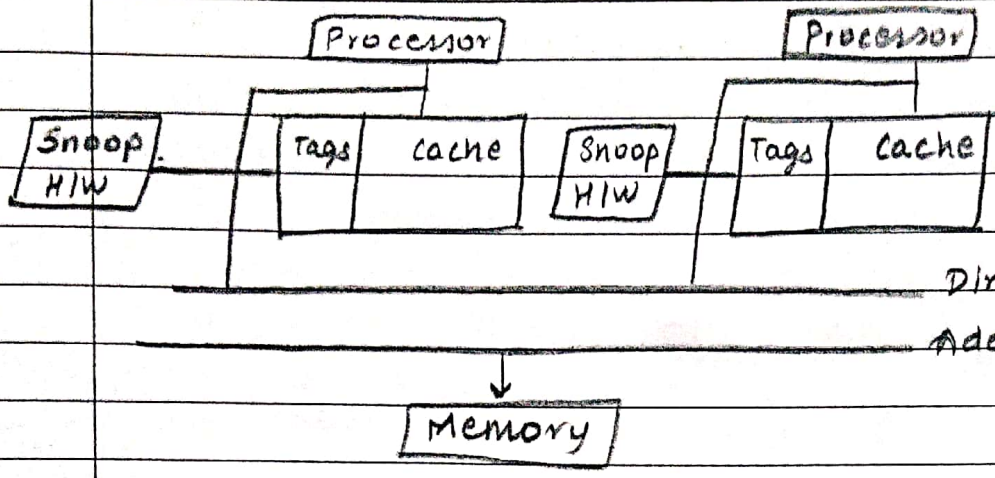
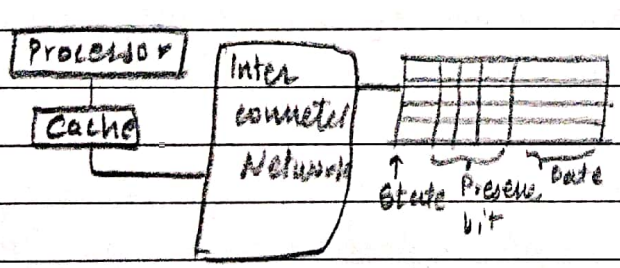


Fig: Snoopy Bus based coherence system.

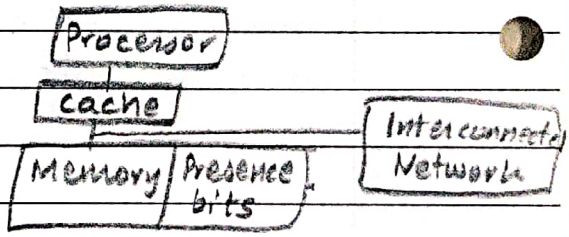
- Tags :- Determine the state of cache as per coherence protocol state diagram.
- Two cases :- (i) Snoop HW detects read ~~access~~ req. to dirty data it then puts data outta bus.
(ii) If write req. on cache block then it invalidates it.

Q. Directory Based System :-

- Global memory connected to a directory that has bitmap representation of cache blocks and corresponding processor.



(i) Centralized directory



(ii) Distributed director

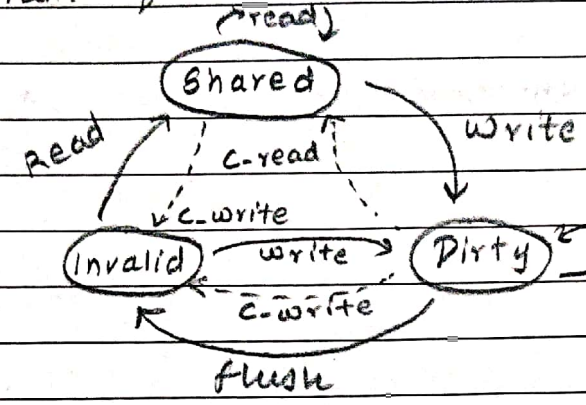
- If any processor attempts to read a dirty value directory notices the dirty tag.
- Using the presence bit the request for the value is sent to the processor with the dirty value.

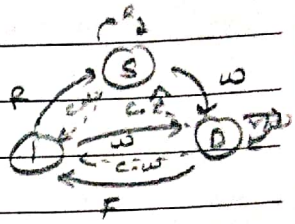
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Overhead in cache coherence System

- Movement of data through bus
- Sending status updates: (System)
  - Invalid
  - Update
- Generation of state info in directory based system.

Q State-diagram of 3 state coherence protocol.





Q Communication cost in parallel machine:-

- Cost of communication depends on :- (i) Programming models semantic (ii) Network topology (iii) Data handling and routing. (iv) Software protocols associated with a program.

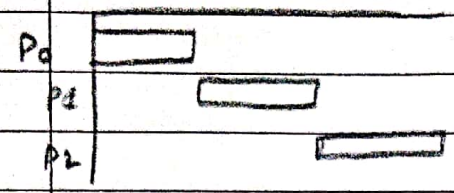
Q Message Passing Costs:-

- Following params decide delay/latency in communication:-

i) Startup time(ts):- Time taken to handle message.
ii) Per-hop time (th):- Time taken by header to travel betⁿ nodes.
iii) Per-word transfer time (tw):- Time taken by each word to traverse the link.

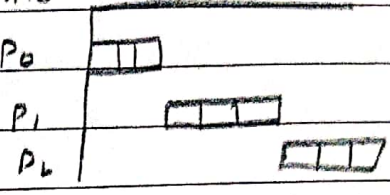
Parallel machine routing → Store & forward / Cut-through Routing / Packet Routing

Store & forward
time →



- $t_{com} = t_s + l(t_h + mt_w)$
  l :- No. of links
- Info stored and forwarded node to node

Cut-through Routing
- Isolate msgs into flits.
- $t_c = t_s + t_h l + t_w m$



Packet Routing
time

- Same msg broken in parts
- Utilizes resources more
- $t_c = t_s + t_h l + t_w m$

Q) Communication costs in shared - Address - space machines

Q) Levels of Parallelism :-

Levels of Parallelism
- Instruction Level [ILP]
  - H/W :- Superscaler
  - S/W :- VLWI
  - Limited
- Thread / Task Level [TLP]
  - Process divides in threads
  - Overhead of thread creation / synchronization
- Data level [DLP]
  - simultaneous working on data

Q) Types of dataflow execution model :-

Types of HPC Model
1) Parallel computing Across Multiple architectures
   - HPC clusters
   - Scalable
2) Cluster Computing
   - Multiple comps Connected through LAN,
   - Act as single computer
3) Grid and Distributed Computing.
   - Multiple comps connected to solve large complex task.
4) Cloud infrastructure

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Point	SIMD	MIMD
Full Form	Single Instr. Multiple data	Multiple Instr. Multiple Data
Memory Req.	Small / Less	Large / More
Cost	less	More
No. of Decoders	Single	Multiple
Synchronization	Lacit	Accurate
Type of programming	Synchronous	Asyna.
Complexity	Low	High
Efficency	Less	More

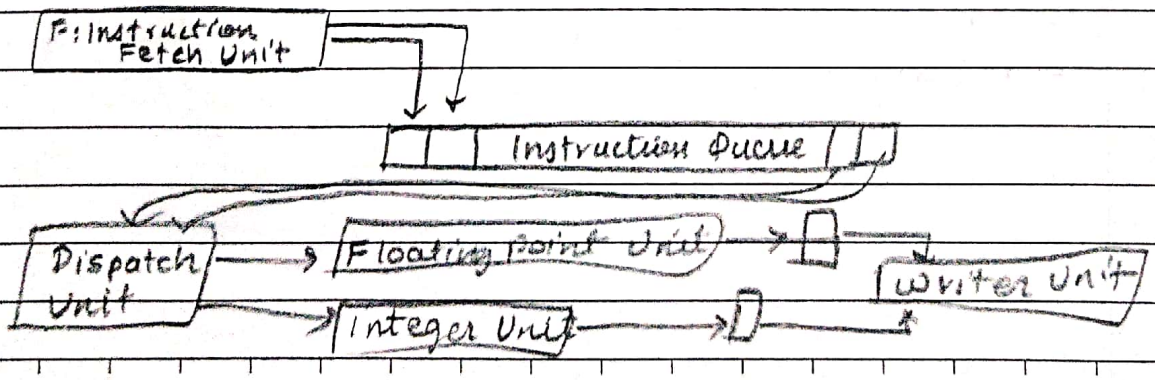
Q SIMT: [Single Instruction, Multiple Threads]

- SIMT = SIMD + Multi-threading
- Mostly implemented on GPUs.
- All threads are executed in lock-step.
- SIMT is like a multi-core system where instructions are synchronized among the cores and not ran independently.

Q SPMD :- [Single Program, Multiple data]

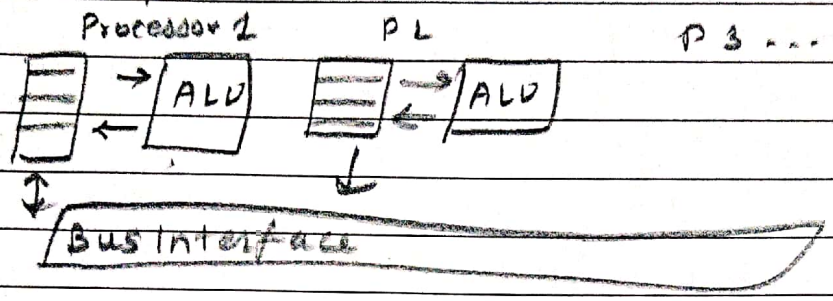
- ~~Single~~ Technique to achieve parallelism.
- Sub-category of MIMD.
- Tasks split and ran on multiple processors.

Q N-Wide Superscalar Architecture :-

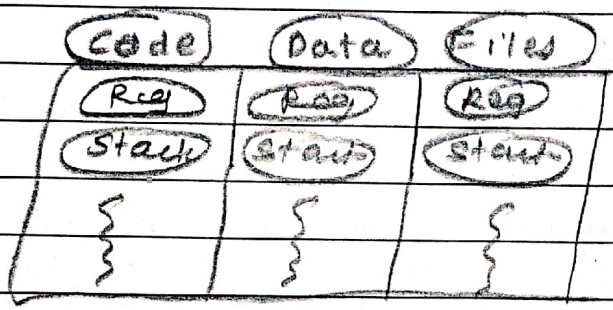


- Superscalar Archi is N-wide if it supports fetch and dispatch of N instructions in every cycle.
- [Describe the diagram.]
- [Diagram for superscalar execution can be drawn]

Q Multi-core processors :-



Q Multi-thread :-



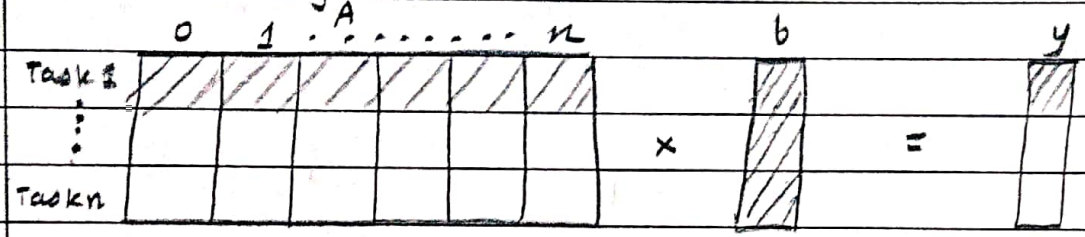
code, data & files are shared among threads.

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

HPC UNIT 2 :- Parallel Algorithm Design

Q. Decomposition, Tasks and Dependency Graph :-

- Decomposition :- Divide computations into sub-computation to execute them parallely.
- Tasks :- Task is a programmer defined unit of computation.
- Tasks are generated by subdividing computation by decomposition.
- Tasks are indivisible parts of computation.
- Tasks don't need to be of same size they can be arbitary size.
- Understanding tasks :-



A : n x n matrix y : resultant vector n x 1
b : n x 1 vector

- Each y[i] of the resultant vector is the dot product of entire i th row of A and the entire vector b.
- We can consider computation of y[i] as a tasks.
- Following points can be made about the tasks :-

1) All n tasks are mutually independent.
2) No task has to wait for another task to finish.
3) No data dependency exists bet^n tasks so they can be executed in any order.

Q Task dependency graphs :-

- Acyclic graph where nodes are tasks and directed edges are dependencies amongst them.
- Possible dependencies among task and their order of execution

can be represented pictorally by task dependency graphs.

- Consider a query to fetch the following data with the following conditions :-

Civic AND 2001 AND (White OR Green)

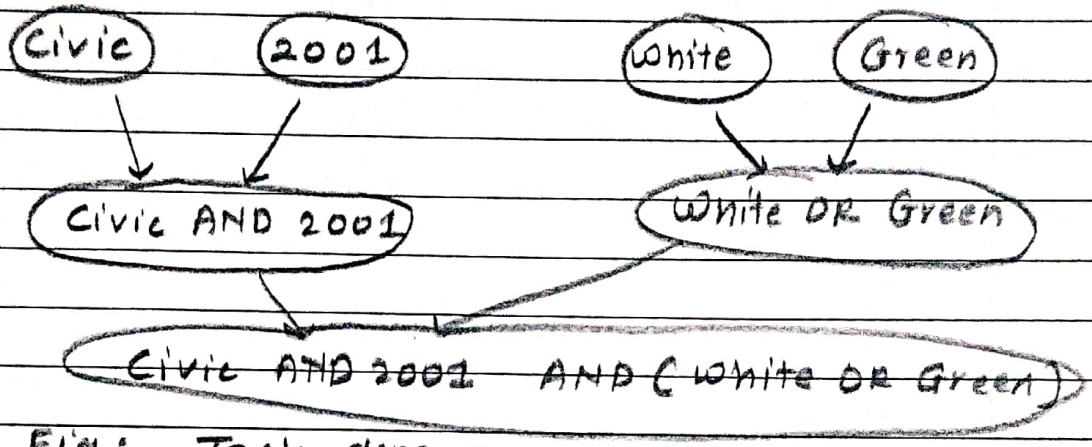


Fig: Task dependency graph.

- Granularity, concurrency and Task interaction :-
- Size of tasks = Granularity.
- Granularity in a parallel algo can be :- i) Fine or ii) Coarse

i) Decomposing a computation into large number of small tasks is called fine-grained granularity.

ii) If large subroutines of an algo are independent of each other, then they can be executed in parallel. This is coarse-grained parallelism. ∴ Coarse-grained decomposition of a computation into a ~~large~~ small no. of ~~small~~ large task.

- Degree of concurrency :-
- Number of tasks that can be executed in parallel.
- Max tasks that can be done is maximum degree of concurrency.

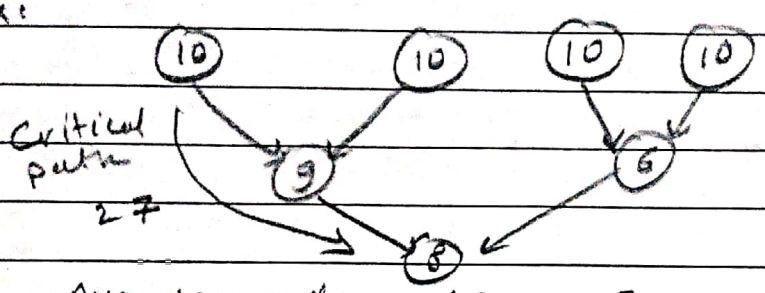
$$\text{Degree of concurrency} \propto \frac{1}{\text{Granularity}}$$

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

- If dependency graph is a tree, maximum degree of concurrency = no. of leaves in tree.
- longest directed path betⁿ start & finish node = critical path
- The sum of weights of nodes along critical path is called critical path length.
- Ratio of total amount :-

$$\text{Avg. degree of concurrency} = \frac{\text{Total amount of work}}{\text{Critical path length}}$$

Ex:



$$\text{Avg. degree of concurrency} = \frac{63}{27} = \frac{7}{3}$$

- Dense matrix can't do more than n^2 concurrent task hence there's always a limit to granularity.

Task Interaction graphs:-

- Graphs of tasks and their data exchange is known as task interaction graph.
- Task Interaction : Data Dependency whereas task dependency = control graph,
- This is required in case of sparse matrix computation.

Processing & Mapping:-

- Process is the computation agent that performs tasks.
- Mapping of tasks to processes is called mapping. Assignment

- The mapping scheme should exploit maximum concurrency.

Q Decomposition Techniques :-

- Job of dividing problems into sub-problems is called decompositions. Techniques to do this :-

Decomposition Techniques

- Recursive Decomposition
  - Suited to Divide & conquer Problems
  - Example is quicksort

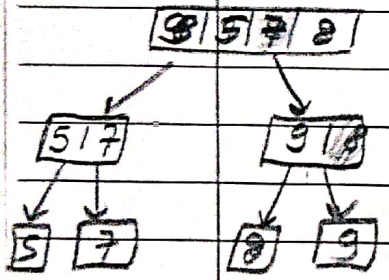
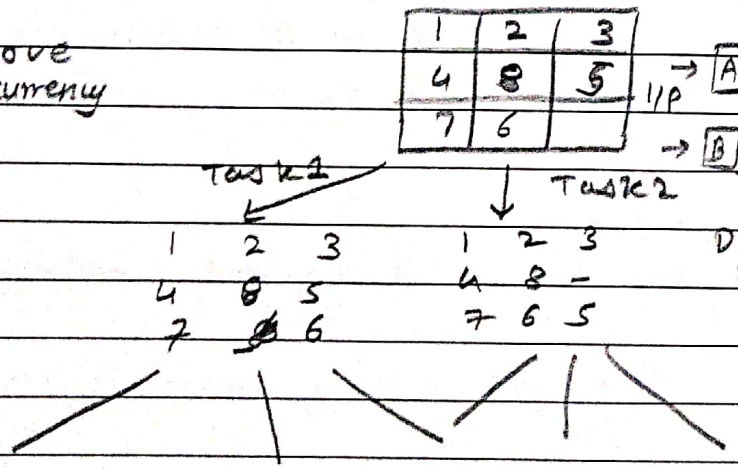


Fig Task dependency Graph

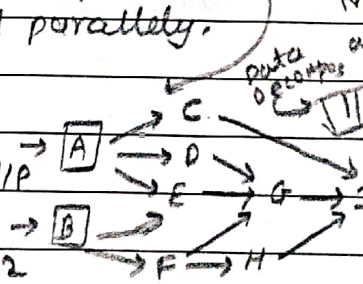
- As we move down concurrency increases.

- Exploratory Decomposition
  - Suited to search space problems.
  - State space search used in AI to reach goal state from given start state.
  - Example of 8 puzzle problem
  - Intially states are generated sequentially & each of the state is explored parallely.



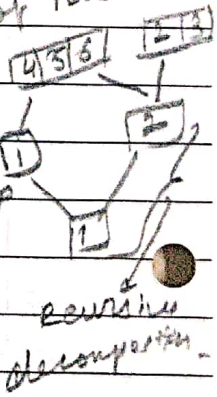
- Our problem is decomposited into 2 tasks
- In exploratory only one task can find a soln and as it finds it all tasks are terminated.

- Speculative
  - Dependencies betn tasks are not known a priori.
  - Ex :- Switch statements in C.



Discrete Event Simulation simple network.

- Hybrid decomposition
  - Combining various decomposition techniques
  - Ex :- Finding minimum no. from a set of nos.



Recursive decomposition

Characteristics of Tasks & Interactions :-

- Good mapping depends on tasks & interactions.
- The effective parameters are :-

Characteristics of tasks

- **Task Generation**
  - **Static**
    - In decomposing data centric apps decomposition of size of data and operation is known apriori.
    - Tasks can be generated by execu. of algo.
    - This is static task gen..
  - **Dynamic**
    - Tasks not known apriori
    - explaratory decomposition Examples.
    - This is dynamic Task gen.
- **Task Size**
  - Time needed to complete task is the size of task.
  - matrix multiplication partioned is done for each row. This is uniform task as size is uniform.
  - In case of quicksort partioned is done by pivot which is random. This is non-uniform tasks.
- **Knowledge of Task size**
  - If knowledge of size of data is known or not,
  - Its known for matrix x but not for 8-puzzle problem.
- Size of data associated with task.
  - If size of data is known overhead of data mvt. is avoided.
  - size of input.

* Characteristics of Inter-Task Interactions :-

i) **Static Interaction** :- Interaction happen at predetermined times. Ex: Mx

ii) **Dynamic** :- Interaction timings not known apriori. Ex :- 8 puzzle.

iii) **Regular** :- Interaction has a structure that can be exploited for efficient implementation. Ex :- Image Compression / Dithering

iv) **Irregular** :- No regular pattern exists. Ex :- Sparse matrix x.

v) **Read-only interaction**. vi) **Read write Interaction**.

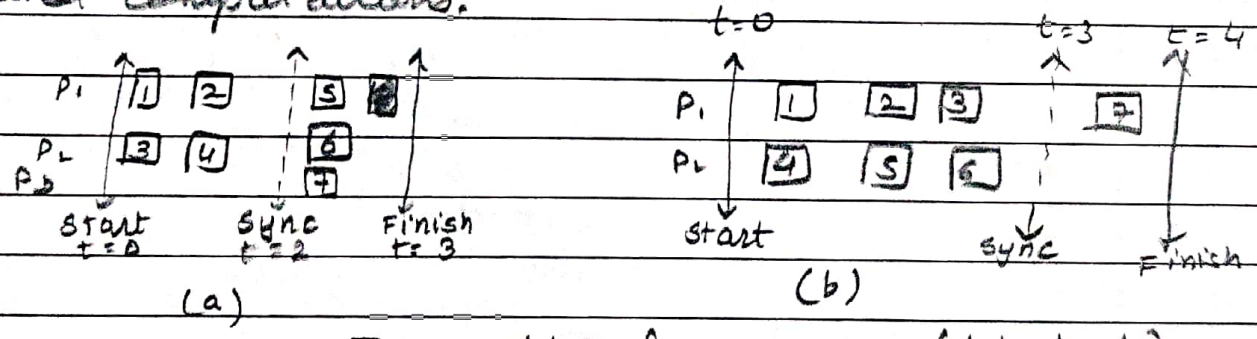
vi) **One-way interaction** :- One pair of communicating task initiates and completes the task. vii) **Two way interaction** :- output of one is input to other.

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Mapping technique for load balancing :-

- Mapping of tasks to process should be done in less time.
- Overheads that occur in decompositions are :-
  (i) Inter-process interaction time (ii) Time for which processes are idle.
- Good mapping shall aim to reduce this overheads.
- Due to load imbalancing some work is finished early.
- To balance load one may put tasks having interactions in the same process, however as some tasks may have different execution time optimum mapping is not obtained
- Good mapping schemes achieve balance betn interaction and computations.



Two configs for mapping (b's bad)

Types of Mapping

Static

- Tasks distributed to program exe.
- Task size known apriori
- Mapping schemes
  - Data partitioning
    - Array distribution schemes :-
  - Task partitioning

Dynamic

- Task distributed during exe.
- Task gen. & map. done dynamically.
- Gets LS in shared-addr. space

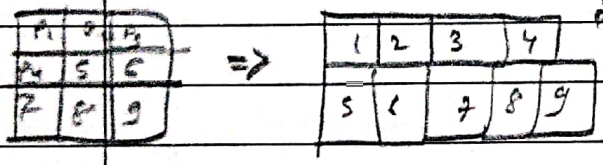
1%3=1 0%3=

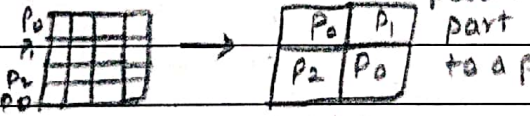
MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Schemes for static mapping

- Data Partitioning
  - Array distribution schemes
    - Block distributions :-
      - Uniform contiguous array portions distributed to different processes.
    - Cyclic & Block
      - Load balancing & idling can be eliminated.
    - Randomized block distribution :-
      - similar to cyclic but blocks randomly distributed.
    - Graph Partitioning
      - Sparse data with highly irregular interactions
  - Block Distribution
- Task Partitioning

- Tasks are responsible for exe. of data holded by them.
- This is owner computes rule
- Techniques used





- Mesh is divided in p parts, each part is assigned to a process p.

Assign blocks with repeating cycles as :-
(i % p)

- Such distributions are called cyclic distribution

* Minimizing Interaction Overhead :-

1) Maximize data locality. 2) Minimize data Volume Exchange. 3) Minimize frequency of interactions. 4) Overlapping computation with interactions. 5) Replicating data/ communication computations. 6) Using group comm^n instead of point-to-point primitives. 7) Overlap interactions with other interactions.

MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Schemes of Dynamic Mapping

Centralized
- Master/slave model for processors.
- When slave ps run outta jobs they request master.
- With more ps master may bottleneck.
- To solve this processors pick chunk of tasks. This is chunk scheduling.

Distributed
- Any process can send/receive work from other processes.

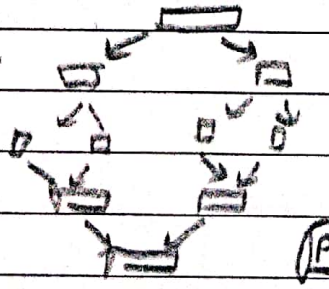
Parallel Algorithm Models :-

- Different ways to structure ~~parallel~~ algorithms to run on parallel system.

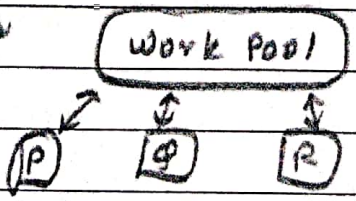
Models

Data-Parallel
- Identical ops on different data concurrently.
- Supports shared-address-space & message passing paradigm
- Uses locality preserving decomposition and overlapping computation.

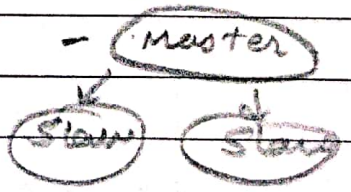
Task-Graph
- Task dependency graph is used to promote locality / reduce interaction costs.
- DFC algos.



Work-Pool
- Dynamic mapping
- Use message passing paradigm.
- Process generate work and add it to global pool.



Master-Slave
- 1 or > generate work and assign it to worker process.
- Allocation can be static/dynamic.



MANJARA CHARITABLE TRUST
RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI
JUHU VERSOVA LINK ROAD, VERSOVA, ANDHERI (WEST), MUMBAI - 53.

Sequential Computational Complexity :-

- $T(n)$:- maximum no. of ops. performed by algo for a given input n.

Parallel Computational Complexity :-

- If p is the no. of processors then,
  - if $p \geq 1$ it is bounded parallelism.
  - if $p = \infty$ or $p \to \infty$ it is unbounded parallelism.
- Time complexity of a para algo to a solve a problem of size is given by a fun^n $T(p, n)$ which is the max^t time that elapses bet^n start of algos execution by one processor and is terminated by 1 or ≥ 1 proces
- Elementary ops are arithmetic or logical ops performed locally by a processor. They take $O(1)$ time.
- A problem is said to belong in NC (Nick's class) if it can be solved in polylogarithmic time using at most polynomial no. of processors.
- Problems in NC are thought to be parallel.

Anomalies :-

- If we write an unbounded parallel algorithm, the performance of the algo cannot be made better beyond a certain limit.
- This is so as some information needs to be known before further computation proceeds.
- The unbounded parallelism reflects this characteristic of a problem and is known as anomalies.
- In a balanced distribution each processors has $\lceil n/p \rceil$ or $\lceil p/n \rceil$ items

Types of anomalies :-

If increase in P increases performance it is acceleration anomaly if it performs worse it is deceleration anomaly.