H.S.

Page
Date

☆ Distributed system :-

A distributed system is one that runs on a collection of inter-connected machine that do not have shared memory, yet it looks to it users like a virtual uni-processor system.

☆ Advantages of distributed system over centralized system :-

1. Economics :-

microprocessors offer a better price / performance ratio than mainframe machine

2 speed :

A distributed system may have more total computing power better than mainframe

3. Inherent distribution :-

some applications involve specially separated machines eg. Banking applications for multiple branches located in scattered geographical areas

4. Reliability :-

If one machine goes down (or crashes) the system as whole can till survive.

5. Incremental growth :

Computing power can be added in small increments

☆ Advantage of distributed systems over independant PCs

1. Data sharing :-

Allow many User to access common database

2 Device sharing :-

Allow many users to share expensive peripheral devices.
A color laser printer

3. communication :

makes a human to human communication easier.
For Ex :- by electronic mail.

4. Flexibility:

spread the workload over the available machines in the most cost effective way.

☆ Disadvantages of distributed system

1 Software: little software exists at present for distributed system.

- How much should the user know about distribution?
- How much should the system do?

2 Networking:
- saturation of network
loss of messages, congesion
Requirement of rewiring
requirement of replacement of NIC.

3. security:-

Authentication
data sharing: also requires to confidential data.

⇒ multi CPU implementations:

Flynn's taxonomy.

1 SISD topology:

b. single Instolation stream
· single Data stream

e.g. traditional von-Neumann machines

2 SIMD computers

single Instruction stream, multiple Data stream.
e.g Array Processors like ILLAC IV

3. MISD computers
multiple Instruction stream
single Data stream.

4. MIMD

multiple Instruction stream
multiple Data stream

=) Types of MIMD :

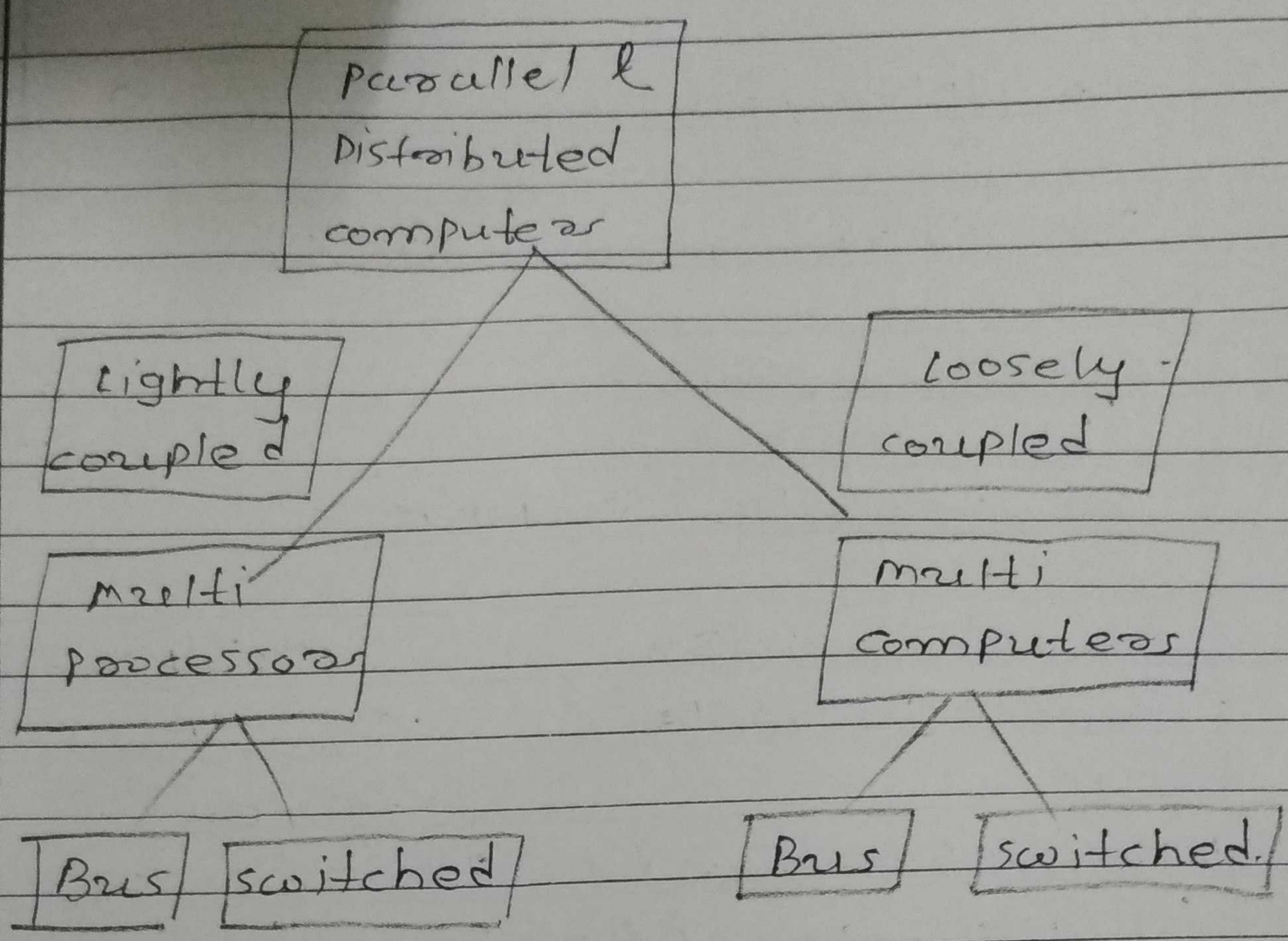
1 multicomputers

eg. Distributed systems

2 Multiprocessors.

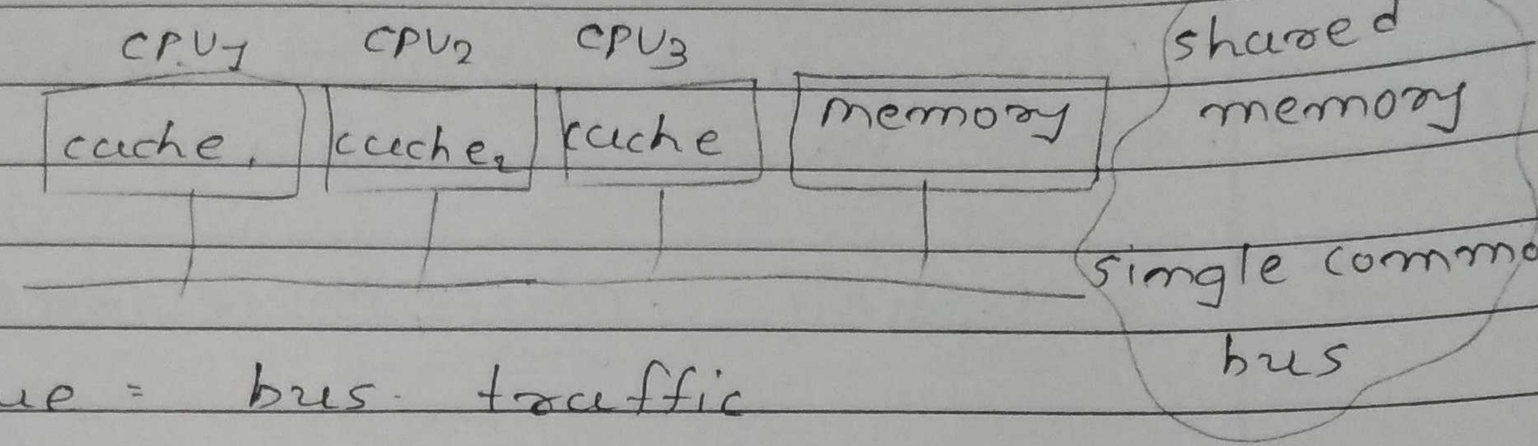
eg. Parallel computers

1.	multicomputer · loosely-coupled sys	Multiprocessor · Tightly coupled system
1	Delay experienced by messages - high	1 low delay
2	Data transmittion speed - Low	2 High data Transmittion speed



classification of parallel and distributed computer system.

☆ Bus-based multiprocessors



Issue = bus traffic

solution = use cache per CPU

New Problem = cache consistency problem

solution = usage of write-through snoopy cache

- constantly monitor the bus
- whereever a word is written to

write-through snoopy cache

It is also written through the memory as well.

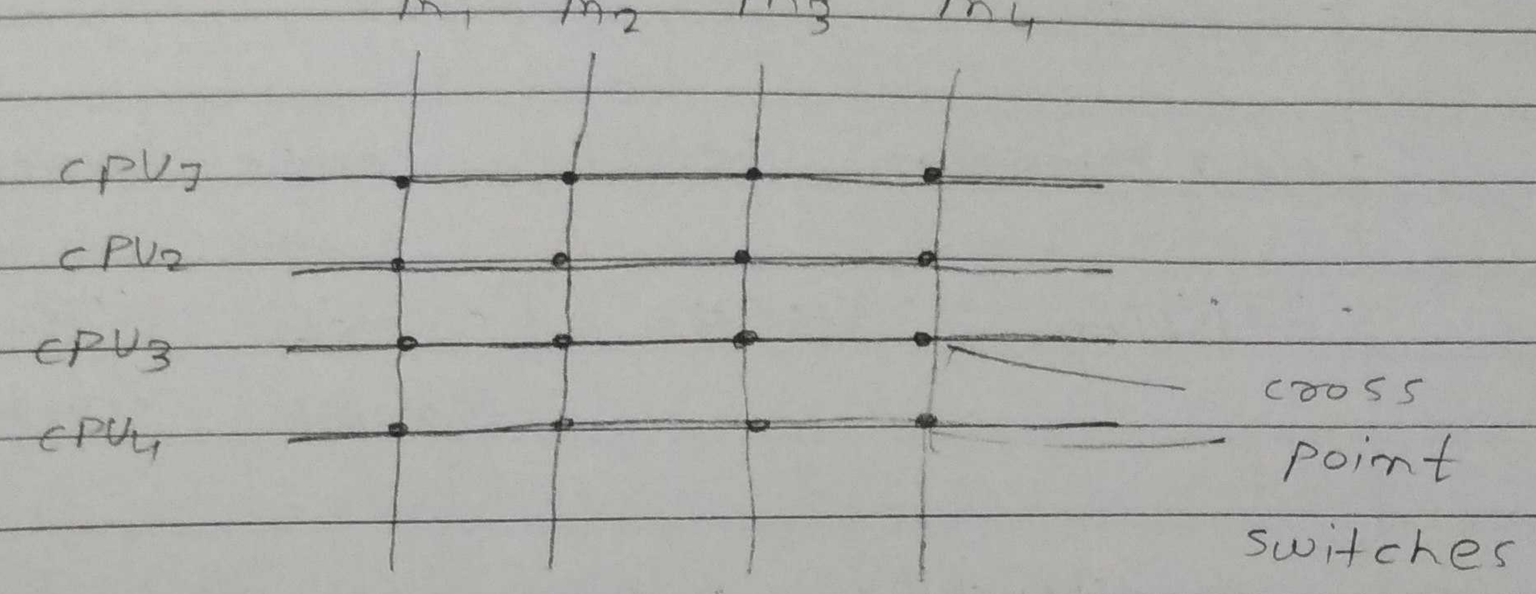
- cache hits for reads do not cause bus traffic

- but cache misses for reads and all writes (hits & misses) cause bus traffic

whenever a snoopy cache sees a occurring to a memory cache address present in cache

it either removes entry from its cache or updates the cache entry. write the new value.

4 switched multiprocessors



(a) crossbar switch

- Devide the memory into modules connect tham with CPUs with a cross bar switch.

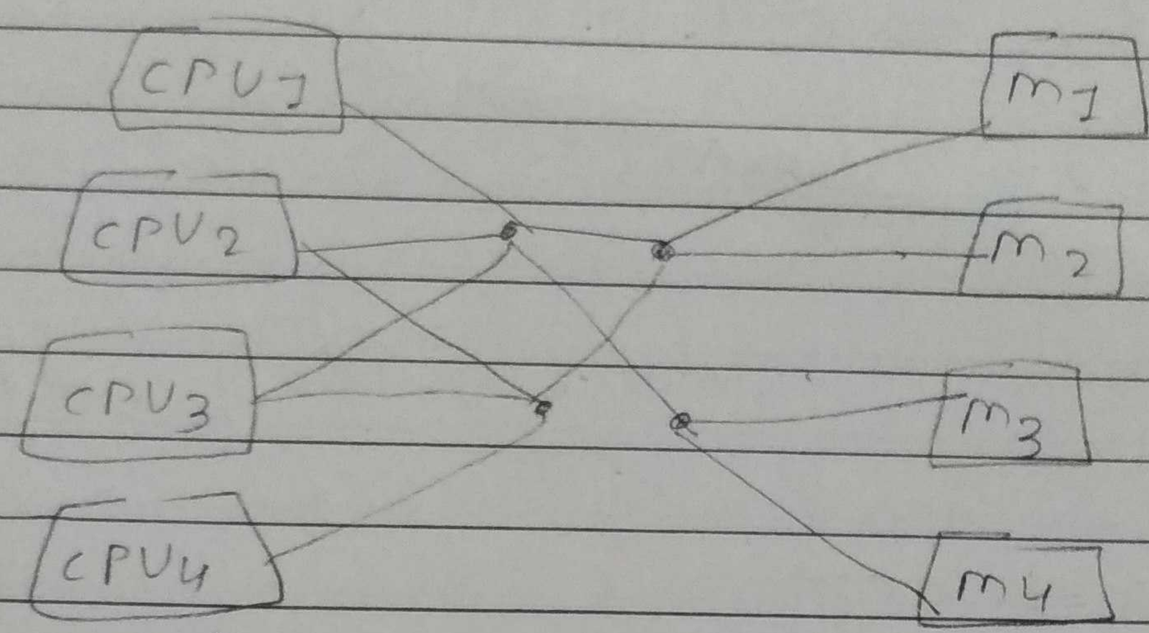
- each CPU and each memory has a connection.

- At every interaction there is tiny cross point switch.

- Virtue :- many CPUs can be accessing memory at same time

limitation :- with a CPUs and memories n^2 crosspoint switches are needed

If n is large, n^2 will be very large.



4 switches used

(b) omega network:

- requires fewer switches.
- Every CPU can access every memory (with proper settings
- In the general case, with n CPU and n memories.
- the omega network requires $\log_2 n$ switching stages

$n/2$ switches per stage.

total number of switches required

$= \frac{n(\log_2 n)}{2} =$

~~greater~~ than n^2
better

If $n = 1024$

total # switches required by omega network

$= \frac{n(\log_2 n)}{2}$

$\frac{1024 (\log_2 1024)}{2} = 512 (10) = 5120$ switches

- consider
  a modern microprocessor

  RISC chip supporting
  50 MHz clock speed

∴ Instruction execution time
20 n sec.

memory request must be traverse.

10 stages
+ 10 stages

total # switches required

$= \frac{n (\log_2 n)}{2}$

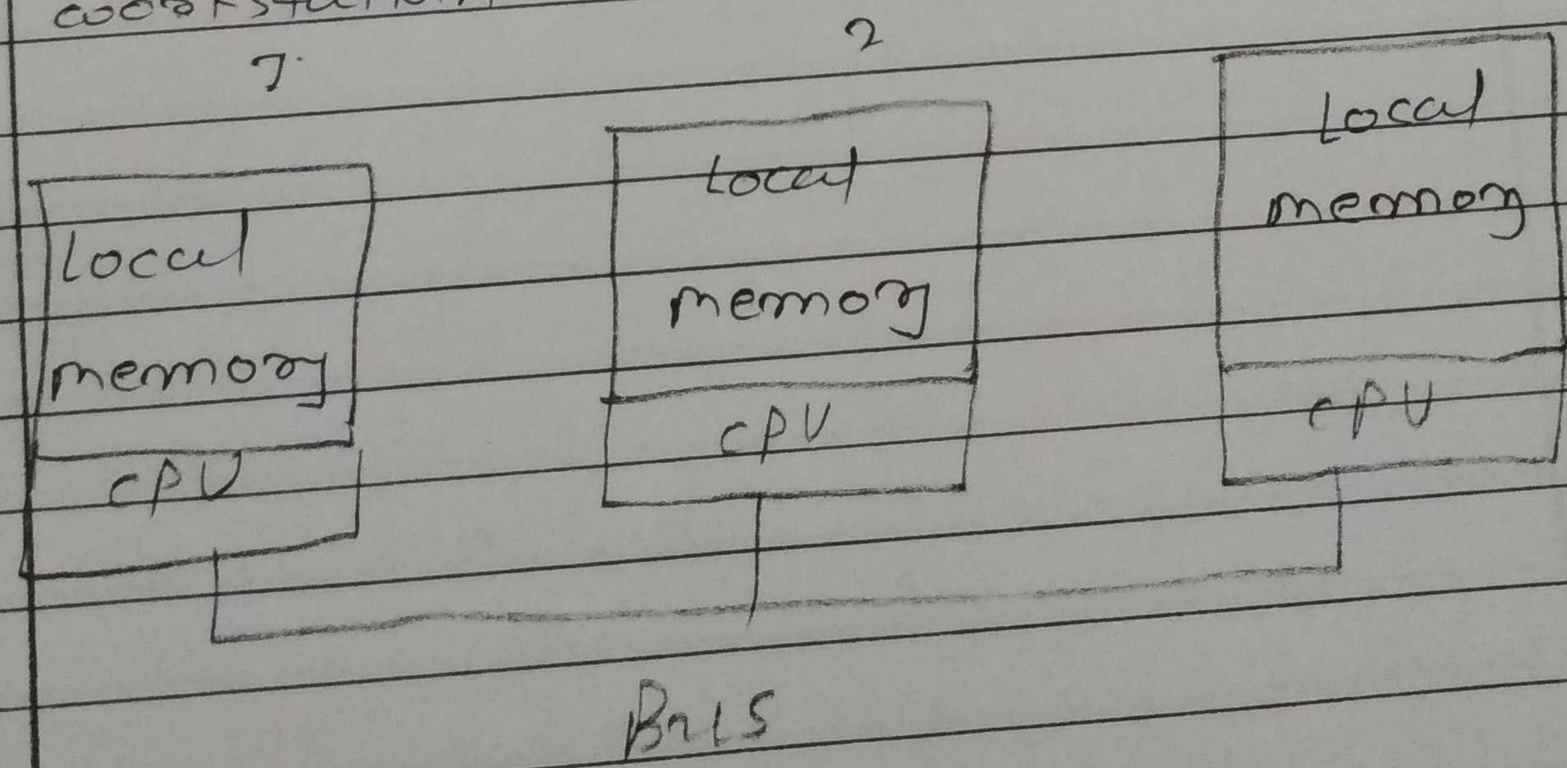
= better than n^2

Page
Date

☆ Non-Uniform memory Access (NUMA) machines

Each CPU can access its own local memory quickly but accessing anybody elses memory is slower

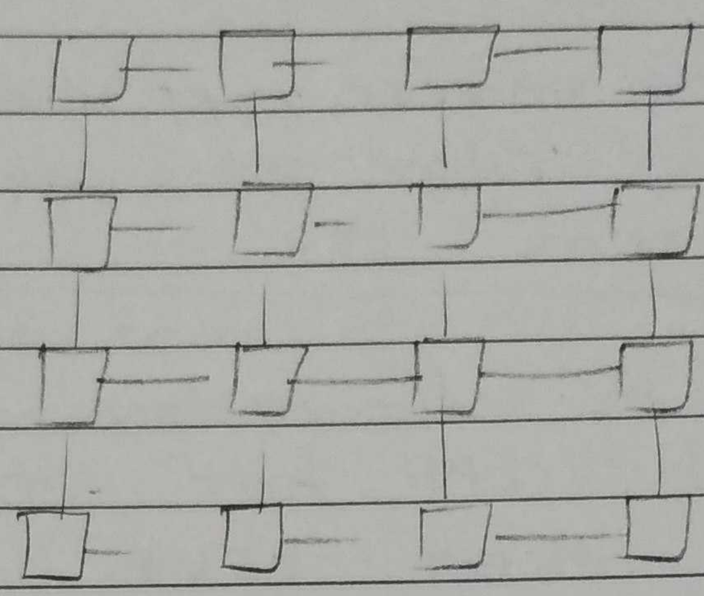
- have better average access time than omega machine's
- placements of programs & data becomes critical
- in order to make most access goto local memory.
- Bus based multicomputers



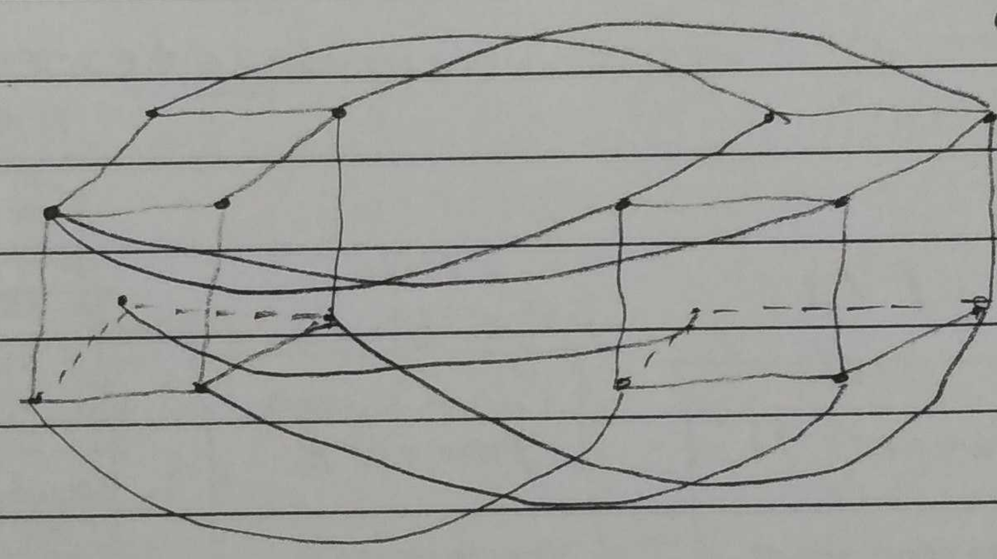
- Building a multicomputer is easy.
- each CPU has a direct connection to its own local memory
- CPU-to-CPU communication
- CPU to memory.
- looks topologiically similar to a bus-based multiprocessor
- Bus will have much less traffic
- A collection of workstations on a LAN.

=>

☆ Switched Multicomputers:



(a) grid



Two ordinary cubes whose corresponding vertices are connected.

4 dimensional

(b) Hypercube

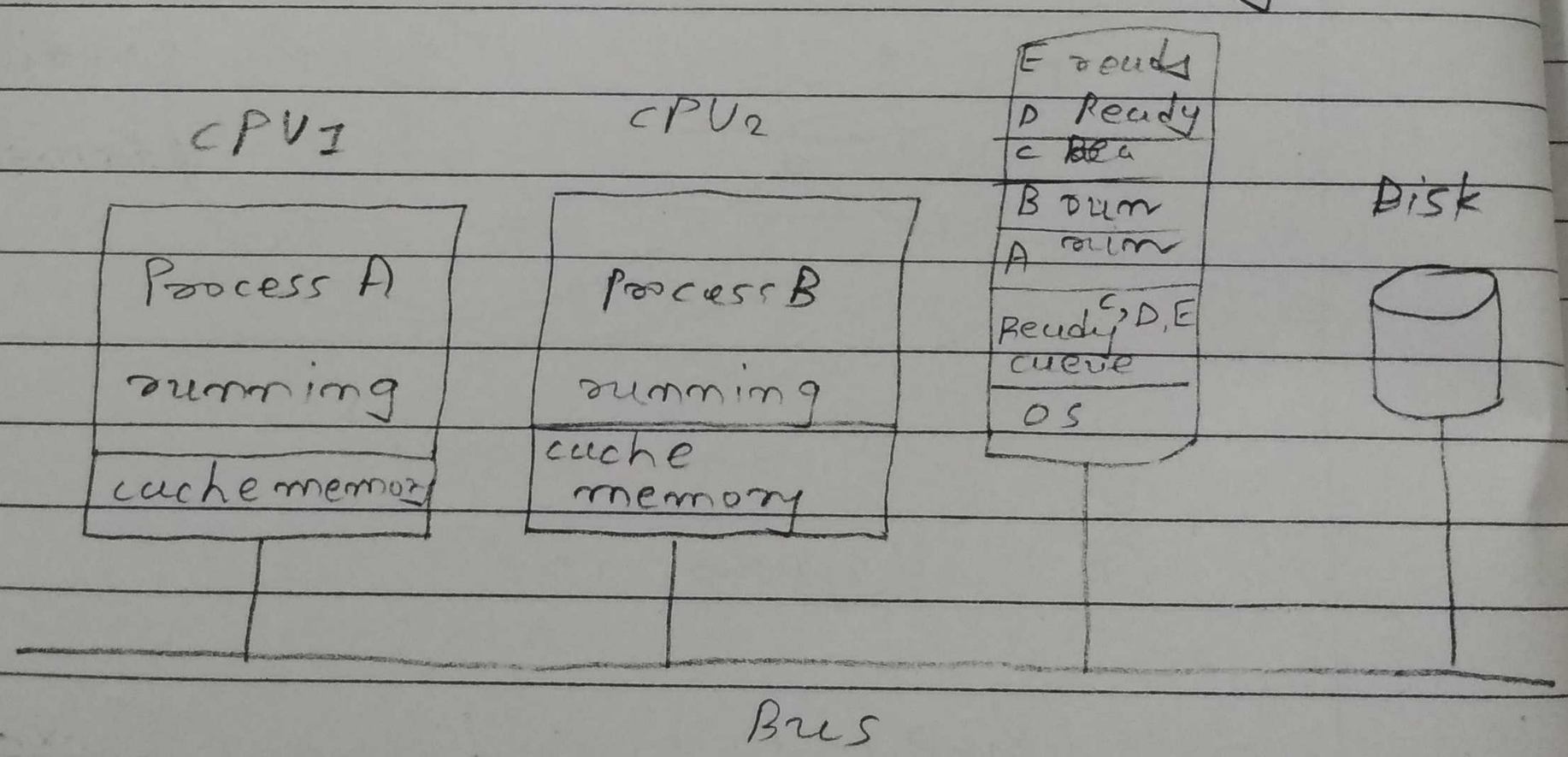
- Each CPU has a direct & Exclusive access to its own private memory.
- Two popular topologies are grid and a hypercube.
- Grids are easy to understand.

- easy to layout on PCB
- best suited to problems
- that have inherently 2 dimensional nature

(graph theory, robot eyes, analyzing photographs) computer vision etc.

- Each vertex represents a CPU
- Each Edge is connection between CPU.
- For n-dimension hypercube each CPU has n connections to other CPUs
-

Multiprocessor Time-sharing system:



- single run queue.

* Comparison of three Different ways of organizing N CPUs

	Network OS	Distributed OS	Multiprocessor OS
Does it look like a virtual Uniprocessor?	No	yes	yes
Do all have to run the same OS?	No	yes	yes
How many copies of the OS are there?	N	N	1
How is communication achieved	shared files	mess ages	shared memory
Are agreed upon network protocol required?	yes	yes	No
Is there single ready queue?	No	No	yes
Does file sharing have well-defined sementics?	Usually No	yes	yes

Design Issues :-

1. Transparency :-

   1.1. location transparency
   1.2 migration transparency
   1.3 Replication transparency
   1.4 parallelism transparency
   1.5 concurrency transparency.

2. Flexibility :

   2.1 monolithic kernel design
   2.2 microkernal design

   more flexible :-

3. Reliability :-

   various aspects of Reliability.
   - availability / replication / consistancy, security / fault tolerance.

4. Performance :

   various performance parameters (metrics)
   response time throughput
   system utilization
   amount of network capacity consumed

- fine grained parallelism
- course-grained parallelism

5 Scalability :-

Potential bottlenecks to be avoided very large scale distribuited system.

- centralized algorithms
- centralized tables
- centralized components

☆ Design Issues

[7.] Transparency :-

1.1 Location transparency :-

The users cannot tell where the hardware / software resources are located.
e.g :- CPUs, printers, file, databases.

The name of the resources must not secretly encode. the location of the resource.

So, names like machine1 : prog.c / or /machine / prog.c } are not acceptable

1.2 :- migration transparency

Resources can move at will without changing their names

✓ Resources must be free to move from one location to another location without having their names change.

/work / news / xyz.txt
/ games / xyz.txt

1.3 Replication transparency :-

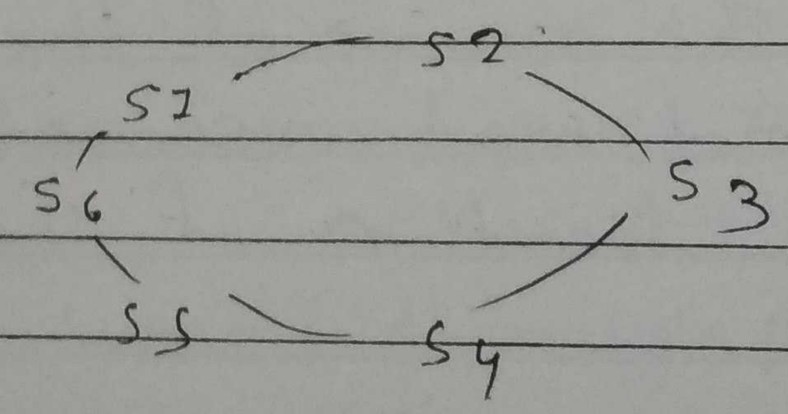
✓ The User cannot tell how many copies exist.

✓ OS is free to make additional copies of certain files & resources.

Ex :-

Consider a collection of server logically connected to form a ring

server



Each server contains a the entire directory tree structure.

OS can replicat the heavily used files and keep additional copies of that file on more than one server.

This way requested file can be made available quickly.
& performance Improved.

1.4:-

Concurrency transparency.

- Multiple Users can share resources automatically.

- what should the system do when two or more Users try to access the same resource at the same time ?

- Two or more Users may try to update the same file at same time so

If the system is concurrency transparency system
then User does not notice Existance of other Users

solution :-

Automatically lock a resource once someone started using it

Page
Date

- Unlock the resource when the access is finished

Usage of locking / Unlocking mechanism

1.5 :- parallelism transparency :-

- Activities can happen in parallel without users knowing

Ex :- chess board Evolution :-

[2] Flexibility :-

There are two schools of thought concerning the structure of a distributed system.

1 monotomic kernel design.

2 ~~microkernel design~~

that provides most services itself

e.g :- UNIX

centralized OS + networking Facility.

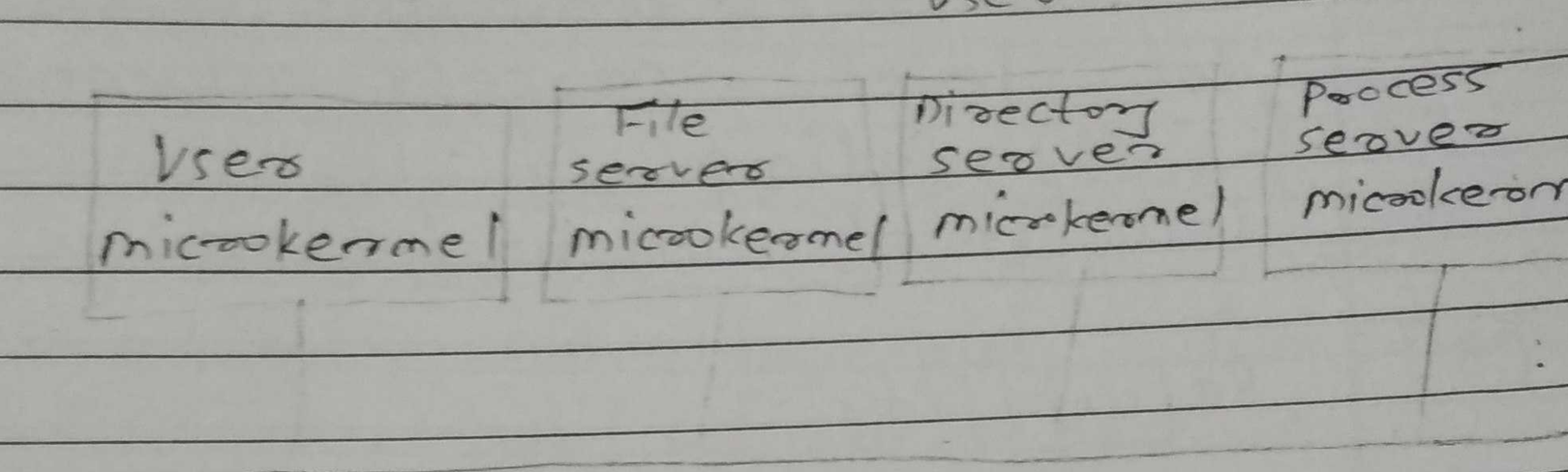
+ remote services

most system calls are made by trapping the kernel

21

1. kernel
2. getting the work done there
3. returning the results to the calls

2. microkernel design



highly modular design
- well defined interface to each service.
- every service is equally accessible to every client irrespective of location
- ✓ it is easy to implement, install & debug.
- Adding or changing the service does not require stopping the system & booting a new kernel
- Ability to add, delete, modify, services, easily.
∴ flexible design approach

- The microkernel basically provides just 4 minimal services:

1. An interprocess communication mechanism
2. Some memory management.
3. A limited amount of low level process management & scheduling
4. Low level input/output.

[3] Reliability :-

associated aspects :-

Availability
Replication
consistency
security
Fault Tolerance.

Availability can be enhanced by replicating key pieces of hardware / software.

Ex = certain heavily used files can be replicated (additional copies can be created)

larger the number of copies, greater the problem of maintain consistency.

Availability refers to fraction of time for which system is usable
highly reliable system must be highly available

Availability can be enhanced by a design that does not require simultanious functioning of a substantial number of critical components to provide some service.

Fault tolerance.

(will bring down users with it?

suppose a server crashes and them quickly reboots
server - tables / information about ongoing activities

servers

S_1	S_2	S_3	S_4
0.95	0.96	0.95	0.95

chance probablity of being that up it will be functioning

Probablity of all four down

$0.05 \times 0.05 \times 0.05 \times 0.05$
$= 0.054$
$= 0.000006$

$= 1 - 0.000006$ probability that the system functions

at least one server working $= 0.999994 > 0.95$

[4] Performance :-

a. various performance metrics

- Response time
- Throughput
- system Utilization
- Amount of network capacity consumed. etc

⇒ fine-grained v/s coars. grained parallelism

- Jobs that involve large num of small computations & high interaction rate are said fine-grained parallelism. course trouble	- Jobs that involve small # of large computational & low integration rate. course-grained parallelism.

(5) Scalability :-

potential bottlenecks to be avoided by the designers for a very large scale distributed system

1. ✓ centralized components
2. ✓ centralized table → not good idea
3. ✓ certain algorithm.

1. ✓ e.g. A single mail server for all users.
2. e.g A single online telephone book
3. Doing routing based on complete information

✓ The n/w capacity into and out of the single mail server may be an issue.

✓ The system would not tolerate falls well

✓ A single power outage → could bring the entire system down.

=) not good idea.

saturation of communication lines
loss of data
vulnerable to failure. eg.
single speck of dust may being. downth

entire directory service)

3 collecting & transporting all information at one place is not a good idea

Heavy communication - performance degraded.

Page
Date

The Distributed systems

A distributed system is one that runs on a collection of inter-connected machines that do not have shared memory, yet looks to its users like a single computer.

- single system image
- a virtual uniprocessor system.

salient characteristics of a distributed system

1. There must be a single, global interprocess communication mechanism, so that any process can talk to any other process.

2. There must also be a global protection scheme.

mixing access control lists, Unix protection bits, and capabilities will not give a single system image.

Page
Date

3. Process mumagement must also be same anywhere.

How process are created, destroy started stop must not vary from machine to machine.

4. There must be a single set of system calls available on all machines.

5. There must be some system calls interface everywhere.

6. Identical kernels run on all CPUs.

system

Page
Date

Disk Usage :- on workstations :

disk Usage	Advantages	Disadvantages
diskless	- low cost - Easy hardware - software maintain - Symmetry & flexible - No noise.	- heavy network usage - file servers - may become bottlenecks
Paging & scratching files	Reduces network load over diskless case.	Higher cost due to large no. of disk needed.
paging, scratch files, system binaries	Reduces network load even more.	Higher cost additional complexity of updating, binary.
paging scratch files, binaries	still lower network load, reduces load on file servers file caching, also.	- Higher cost - cache consistency problems.
Fully local of system. file.	Hardly any new load Eliminates need for server	Loss of transparency

Page
Date

* Idle workstations :-

- no user processes running.
- no-user initiated process running

⇒ Algorithms used to locate idele workstation:

→ server-driven

when a workstation goes idle, it announces its availability

methods:

1. By entering its name network address & properties in a registry file.

2. Put a broadcast message onto the network. Each machine maintain its private copy of registery file.

client-driven

The client machine broadcast a request message to say that

- what program it wants to run, it needs?
- processing requirement etc.

Replies are received