Date
MON TUE WED THU FRI SAT SUN

Chap No:-6.

Resource & Process Mgmt.

Resource it is an entity that is required for execution of any process.

Resource Mgmt:-

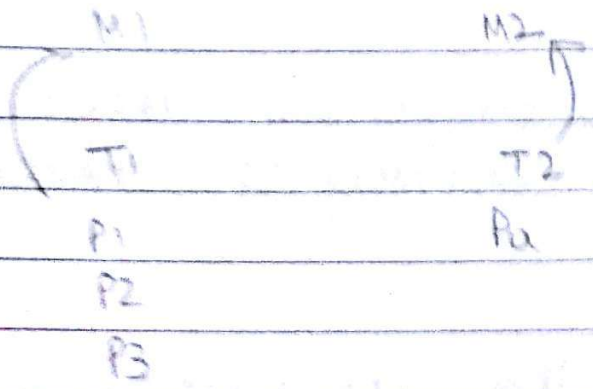
Resource Scheduler is responsible to manage all the resources and considers three methodology to schedule resource

1) Task mgmt:
2) Load Balancing
3) Load Sharing

Features of good scheduling algorithm.

1) NO apriori knowledge of processes:-
NO scheduler should depend upon taking a decision for assigining a m/c to a process based on ~~of~~ what all things the process may require for execution.

2) Dynamic scheduling:-
In static scheduling their are fixed set of rules for Eg:- All task of T1 is give to M1 and all task of T2 give to M2.



M2 will sit free after executing Pa.
In Dynamic scheduling their will not be fixed set of rules. depending on m/c & process availability, scheduler will take decision.

3) Balanced S/4 peryormance & scheduling overhead:-
Scheduling & Processing should be balanced in such a way that both should improve S/4 peryormance

3) Quick decision making:-
Scheduler should never analyse the process in detail & quickly assign a m/c for execution.

Date :

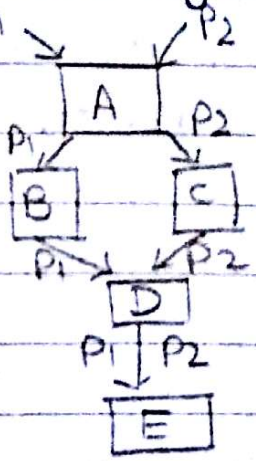
MON TUE WED THU FRI SAT SUN

4) Scalability :-
Increase in number of processes & machines should not hamper working of scheduler.

5) Fault Tolerance :-
If any machine fails the, scheduler should reschedule the process of those m/c to other m/c

6) Fairness of Service :-
No scheduler should ~~process~~ favour any m/c

7) Stability :-
Instability system keeps on migrating the proces without migrating it, this is known as Thrashing.
So, In a stable s/y the migration will be Controled
for eg :- migration of two processes P1 & P2.



Date :

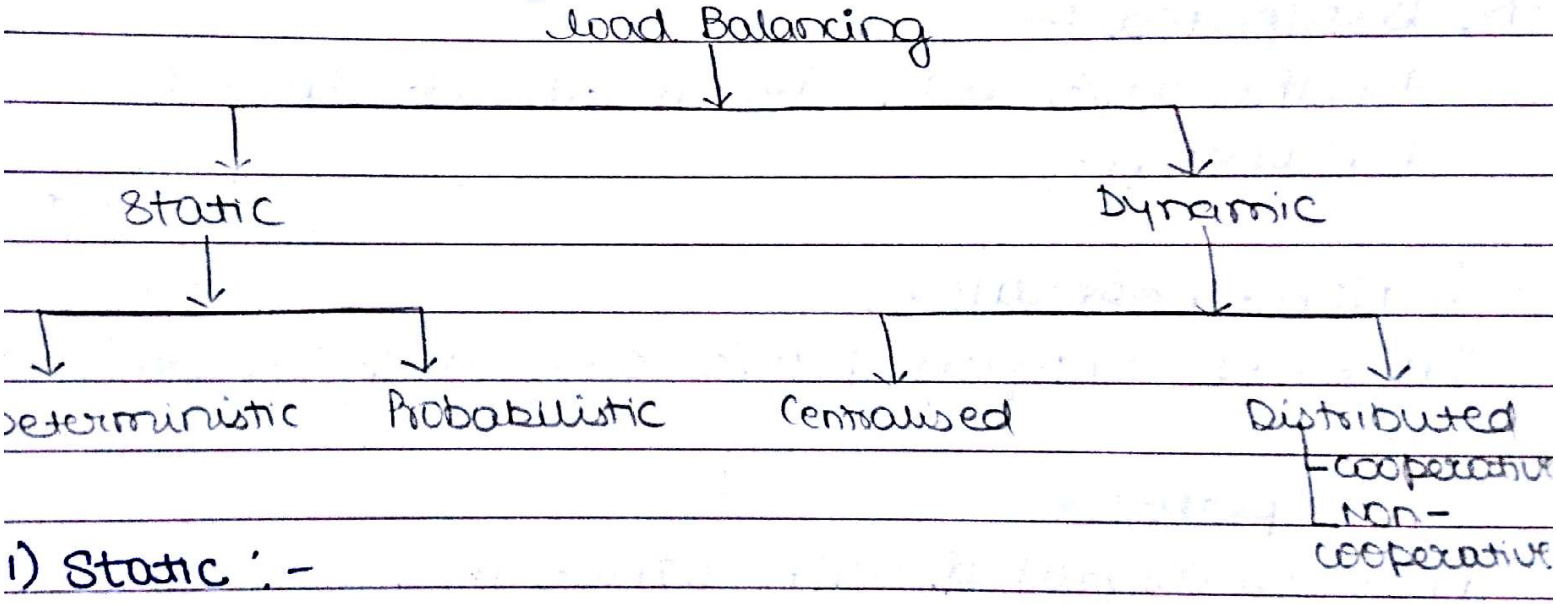
MON TUE WED THU FRI SAT SUN

**

Load Balancing :-

Load Balancing focuses on a concept where neither the s/m should be underloaded nor overloaded.

load balancing approach.



1) Static :-

Decision taken ~~as~~ based on fixed set of rules

a) Deterministic :-

Decision about scheduling will be taken by identifying some features of m/c.

b) Probabilistic :-

Some mathematical calculation will be calculated which tells, which m/c is more probable to execute the task successfully.

Date :
MON TUE WED THU FRI SAT SUN

2) Dynamic :-
Decision based upon the current Situation of n/w.

a) Centralised :-
There will be centralised server that peform load balancing.

b) Distributed :-
All the m/c will decide about load balancing.

i) Non-Cooperative :-
Processes migrated w/o any m/c concern.

ii) Cooperative :-
Process migrated with other m/c concern.

** Issues in Designing Load Balancing :

1) load Estimation policy :-
- In this load of a m/c will be calculated.
- load of any given m/c depends upon the following properties of m/c.
  a) Processor architecture
  b) Speed of processor.
  c) Number of processes present in the s/y.

Date :

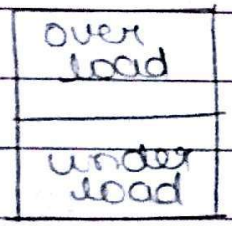
MON TUE WED THU FRI SAT SUN

d) Number of Instructions executed per sec.

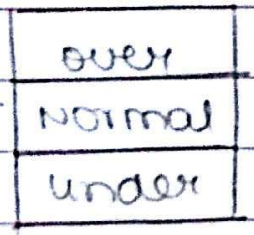
2) Process Transfer Policy:-

In this a concept of threshold value is used which decides whether the m/c is underloaded or overloaded.

a) Single threshold.



b) Double Threshold

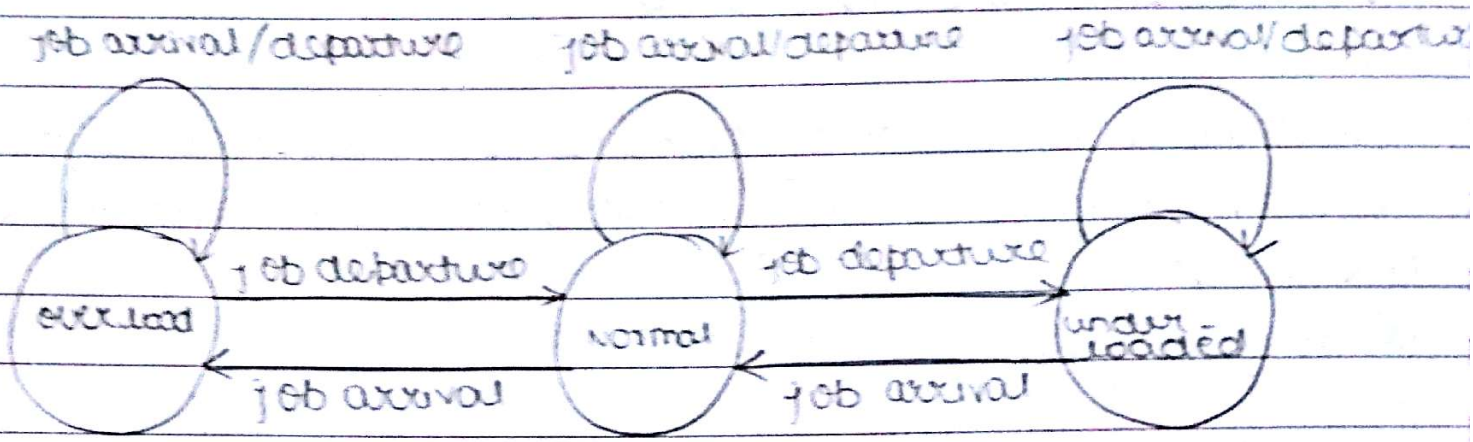


This policy will tell the actual load of the m/c after load estimation.

Date :

MON TUE WED THU FRI SAT SUN

State transition Diagram of a load of a node in case of Double threshold policy



3) Priority Assignment policy:-
This will decide which processes have high priority : local or remote.

i) Selfish :-
Higher priority to local.

ii) Altruistic :-
Higher priority to remote.

iii) Intermediate :-
Higher priority to majority of process.

4) Migration limiting
Before migrating any process m/c should decide on how many migration a process can perform.

i) Controlled :- limitation on migration.

ii) uncontrolled :- No limitation on migration

Date :
MON TUE WED THU FRI SAT SUN

5) Location policy
This policy will decide where the process will finally be migrated.

a) Threshold :-
on a node which is ~~overloaded~~ generally underoverloaded.

b) Shortest :-
The node having the lowest load.

c) Bidding :-
Either the bidding is sender initiated. where overloaded m/c will receive offer from underloaded or normal in the form of Bid
In receiver Initiated, all the m/c which are underloaded will keep on advertising their load. & when any sender wants to migrate will select one of the receiver.

d) Pairing :-
Always pair an underloaded with overloaded and normal node with normal.

Date :

MON TUE WED THU FRI SAT SUN

6) State information exchange policy :-
This will decide when to exchange state information with others.
a) Periodic Broadcast
b) Broadcast with state change.
c) On-demand exchange.
d) Exchange by polling.

Load Sharing :-
It focuses on the principle that no CPU should sit ideal.

Issues :-
1) load estimation policy.
2) Process Transfer Policy.
3) location Policy.
4) State information exchange Policy.

explanation same as load balancing

Task assignment :-
Refer notes

Date :

MON TUE WED THU FRI SAT SUN

Process Management :-

Process Migration :-

The concept of migrating, the process from one node to another node is known as process migration.

following are two types of migration technique.

1) Preemptive :- "Process migration"
The migration is done in b/w the process of execution.

2) Non-preemptive :- "Code migration"
The migration is done before the process of execution

Why to migrate processes or code :-
either the Source m/c is not having resource or the Source m/c is overloaded.

Desirable features of good process migrating mechanism :-

1) Transparency :- user should be unaware of the fact that the process is being migrated & is executed on some other machine.

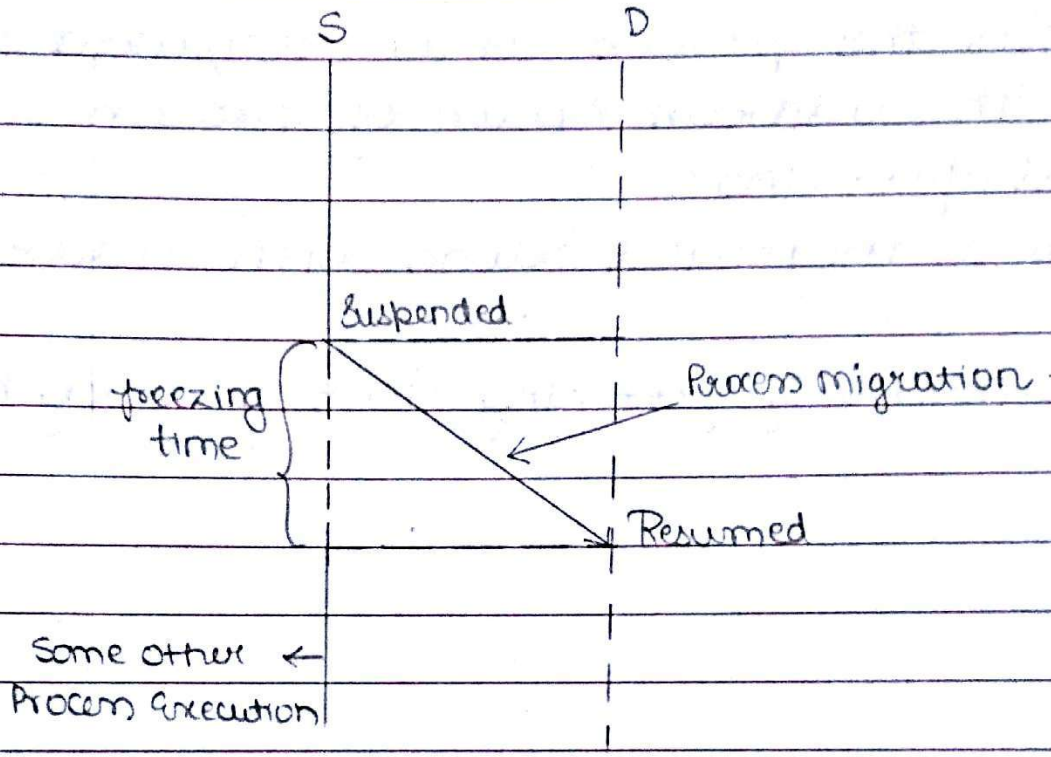
2) Minimal Interference:-
The source node should never keep on interfering in execution of a process which is on another node.

3) Minimal residual dependency:-
When the processes is migrated also migrate the required resource. so, that their is no residual dependency.

4) Efficiency:-
Efficiency is defined by
a) less amount of freezing time
b) No Thrashing (migration w/o execution).

5) Robustness:-
Failure of any m/c in the n/w should not affect execution of any process

6) Communication b/w Co-processes:-
Both these processes are interdependent on Each other.



Process Migration Mechanism :-

1) Freezing at source & starting at Destination.
whenever a decision is taken about migrating the process, check whether the process to be migrated is executing a system call.
If yes, delay the migration process &
If it is not executing any imp task suspend the execution and migrate the process.

Date :
MON TUE WED THU FRI SAT SUN

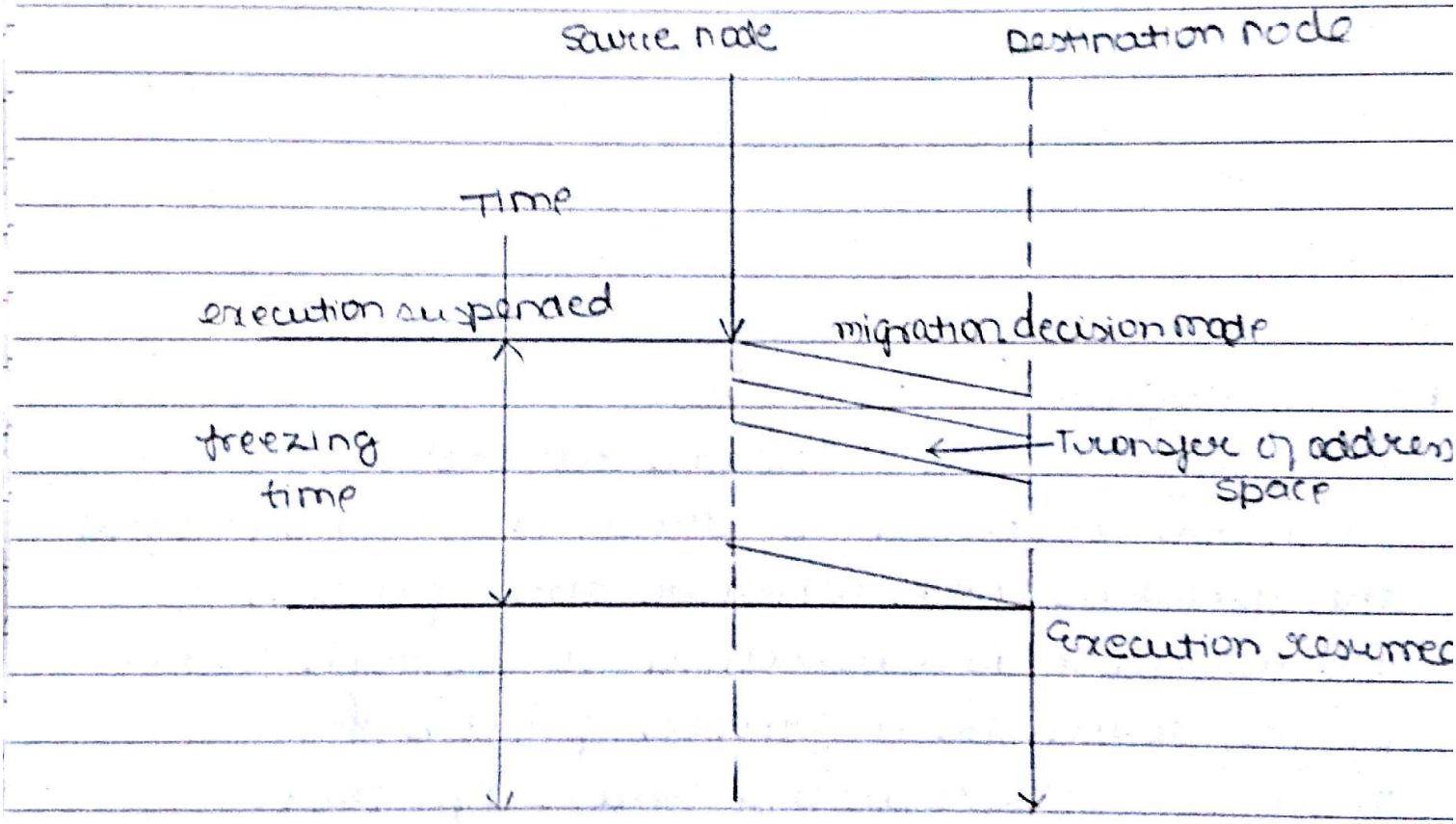
2) Transferring address space :-

Along with the process it is required to migrate the address space of process.

i) Total freezing :-

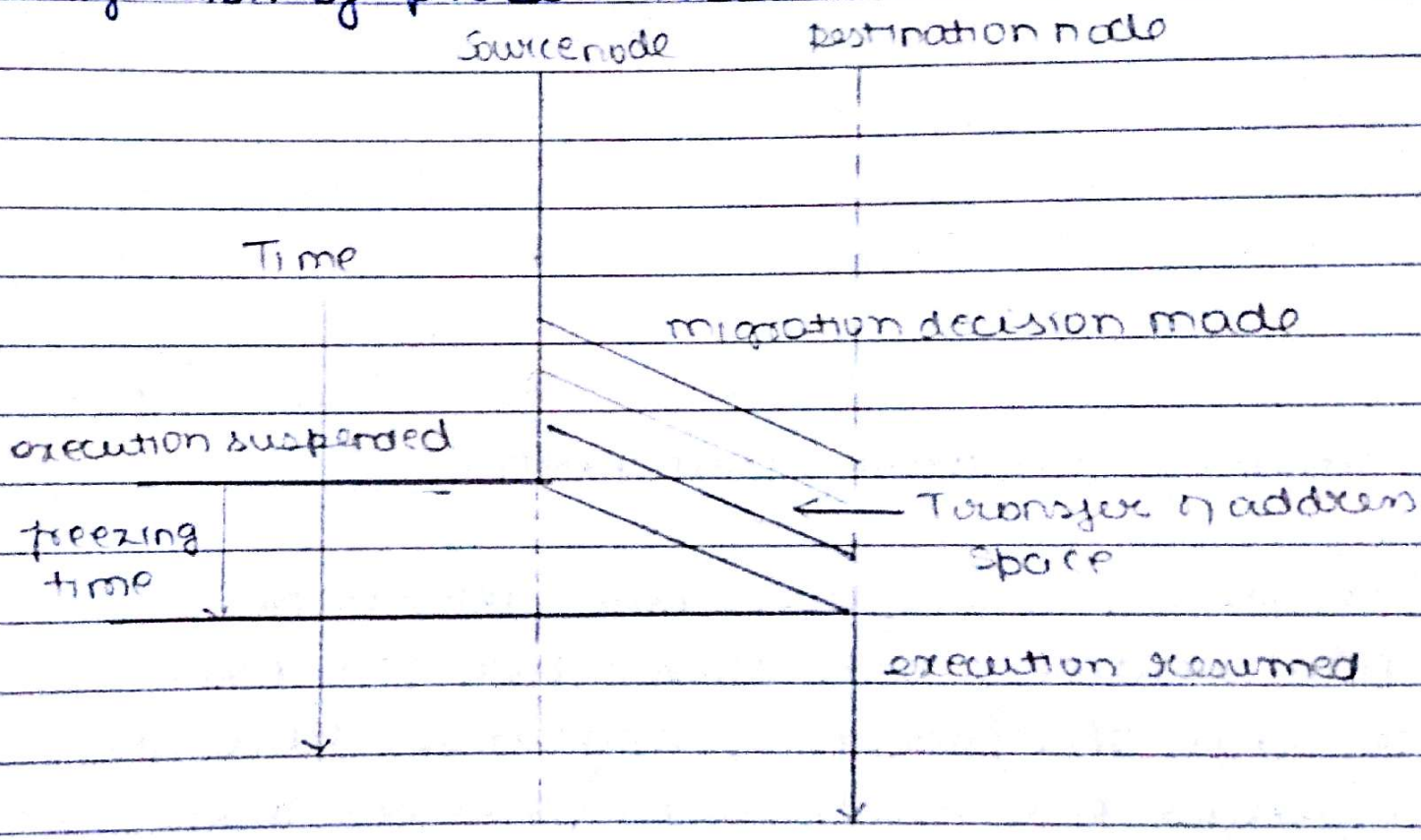
Process is migrated along with address space.

Here, amount of freezing time is very high.

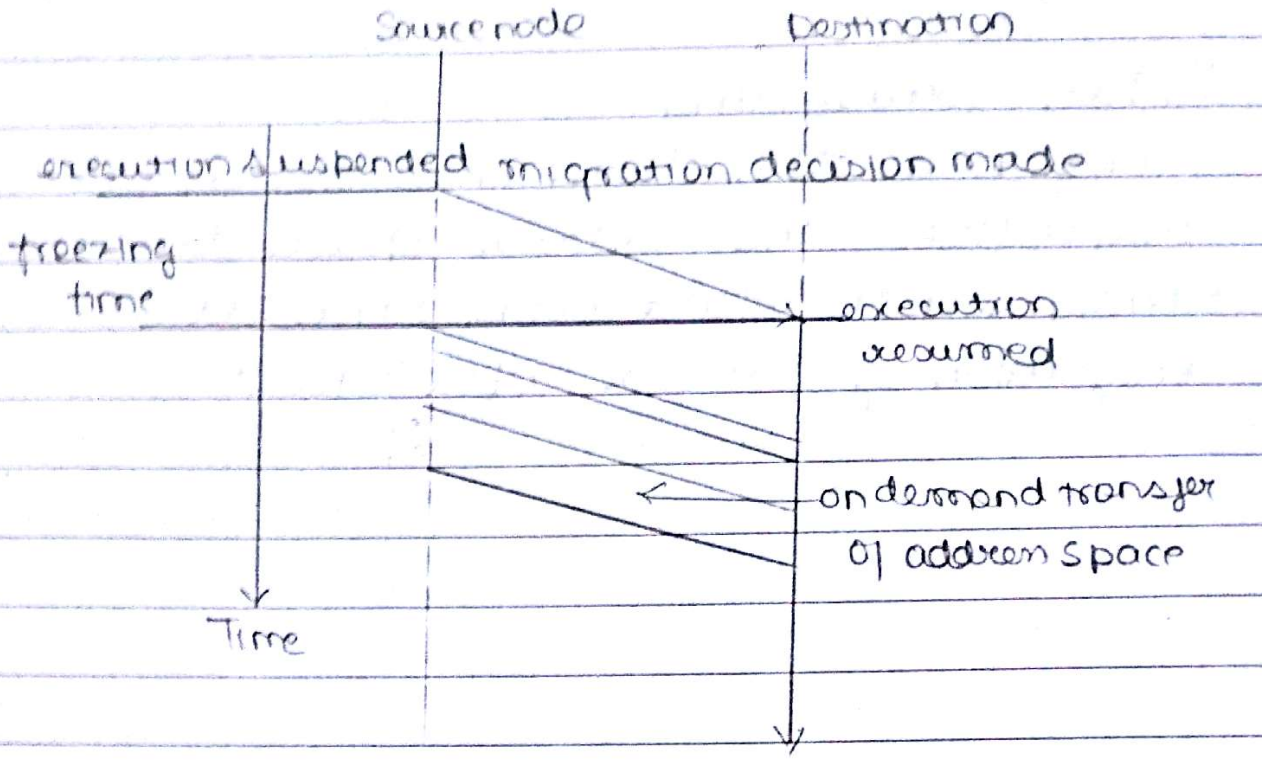


Date :
MON TUE WED THU FRI SAT SUN

ii) Pre-Transferring :-
Here, once the decision is taken about migration. The address space transfer will immediately start and once it is done migration of process will start.

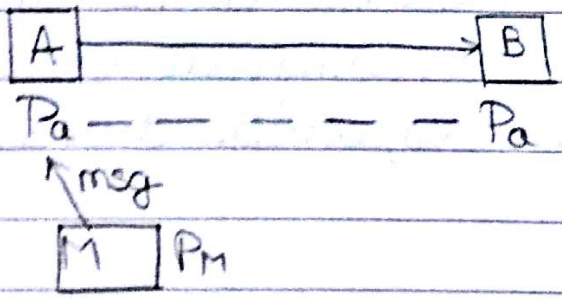


iii) on Demand Transfer
It is not always necessary to migrate the address space. so, migrate only the process & whenever required transfer the address space.



3) Message forwarding mechanism:-

Process on source machine might be communicating with some other process. So, when the process is migrated, how the messages b/w such such processes are handled is discussed here.



Date :

MON TUE WED THU FRI SAT SUN

Solution:-

i) If Pm sends message to Pa on A that msg is dropped as Pa has migrated.

ii) If Pm sends message to Pa after migration A will forward to B where A has migrated. This is called origin site mechanism.

iii) When Pm sends msg to Pa. A gives the path to reach to B where Pa has migrated.
This is called as Linked Traversal.

iv) When Pa migrate. A will update M with address of Pa. (link update

4) Handling the communication b/w co-process:-

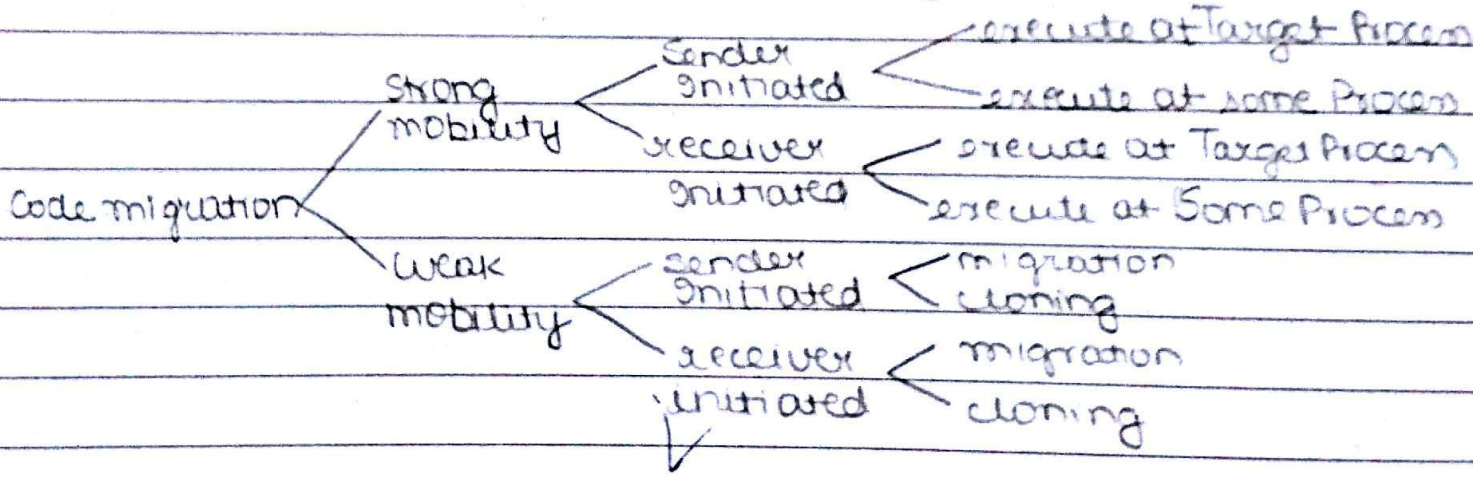
Never Separate Co-processes as they interact with each other more often.

In this, every processor should remember only two format.

1) common format & its own format.

Code Migration in Distributed System.

The process of migrating any program before execution is known as code migration.

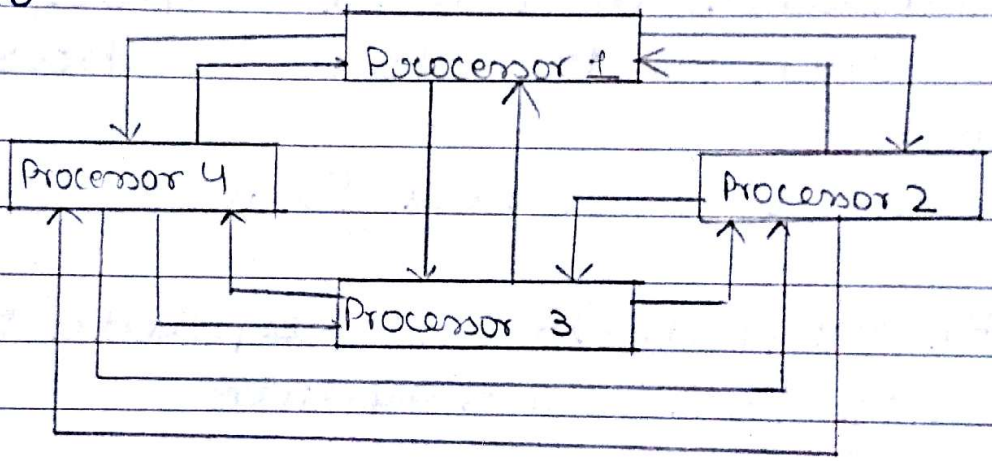


Date :

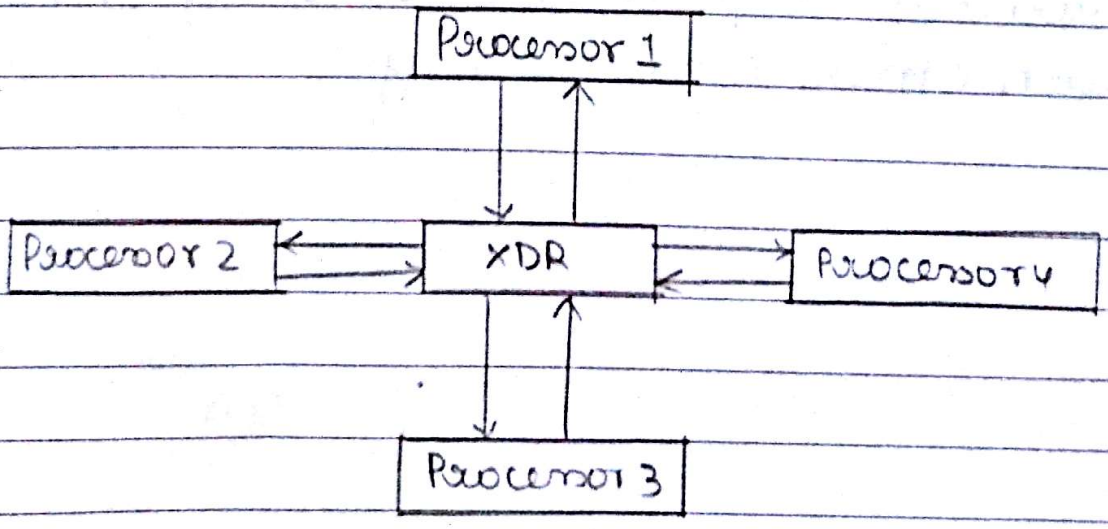
MON TUE WED THU FRI SAT SUN

Process migration in Heterogenous Environment

In a distributed environment there are diff m/c with diff configuration
Therefore, while migrating the process every processor should remember multiple format as every processor is using diff format.



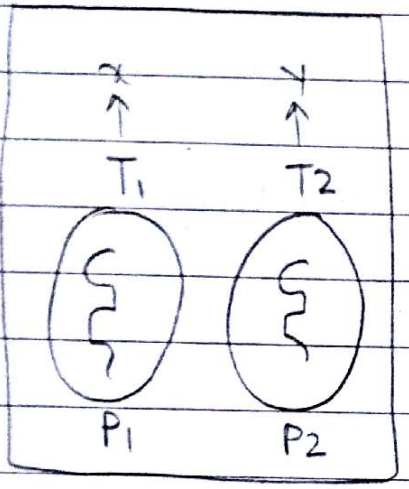
To avoid such complex architecture a common format is developed is known as XDR (external data representation).



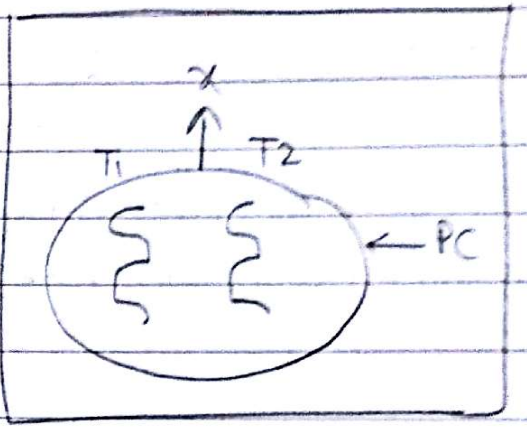
Date :
MON TUE WED THU FRI SAT SUN

Threads.

It is a smallest part of a process.
Process is Heavy weight but Thread is light weight.



Processes P1 & P2 having different add spaces



Processes P1 with two threads T1 & T2 sharing common adder spaces

Thread Models :-

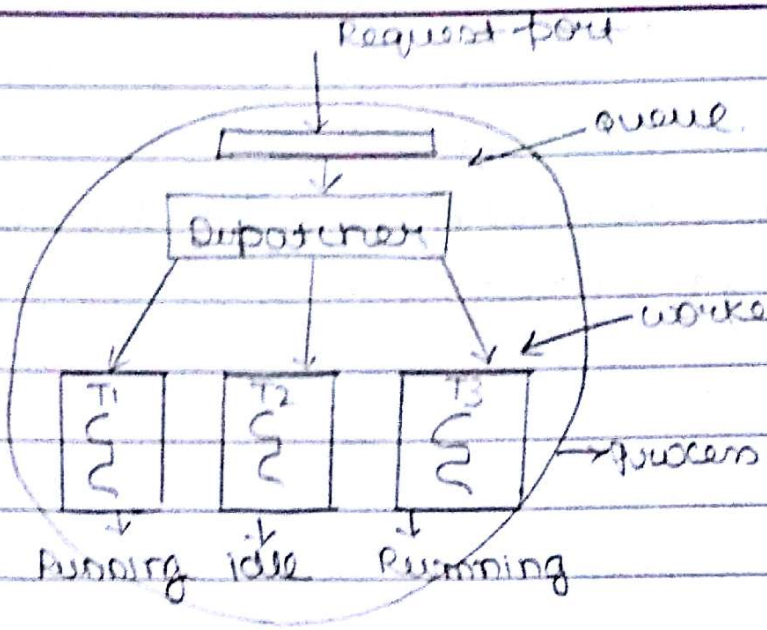
1) Dispatcher worker Model :-

In this their is a dispatcher which accept the request from client given to the available thread for execution and returns the result to the calling process.

Each thread works independently and multiple request can be executed parallely.

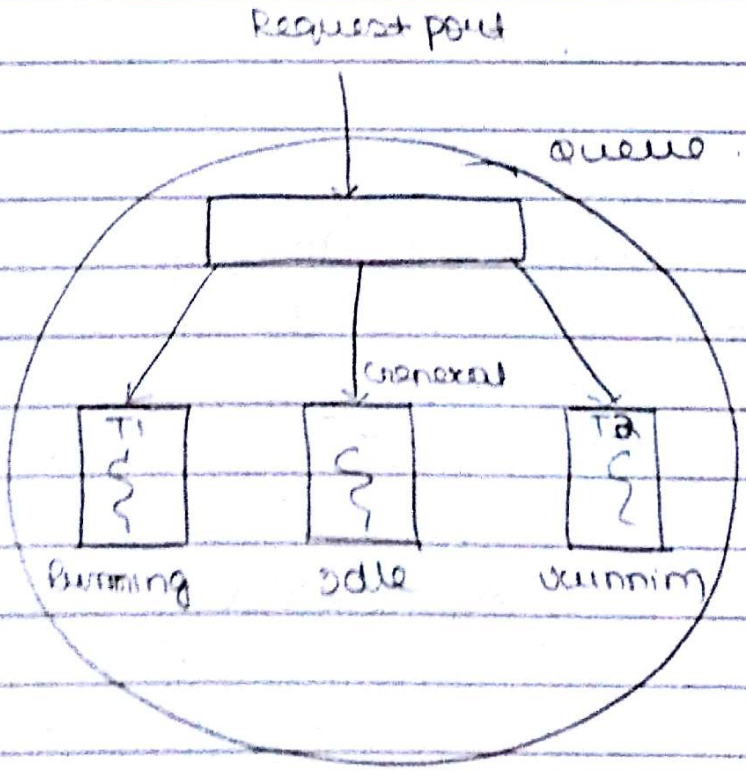
Date :

MON TUE WED THU FRI SAT SUN



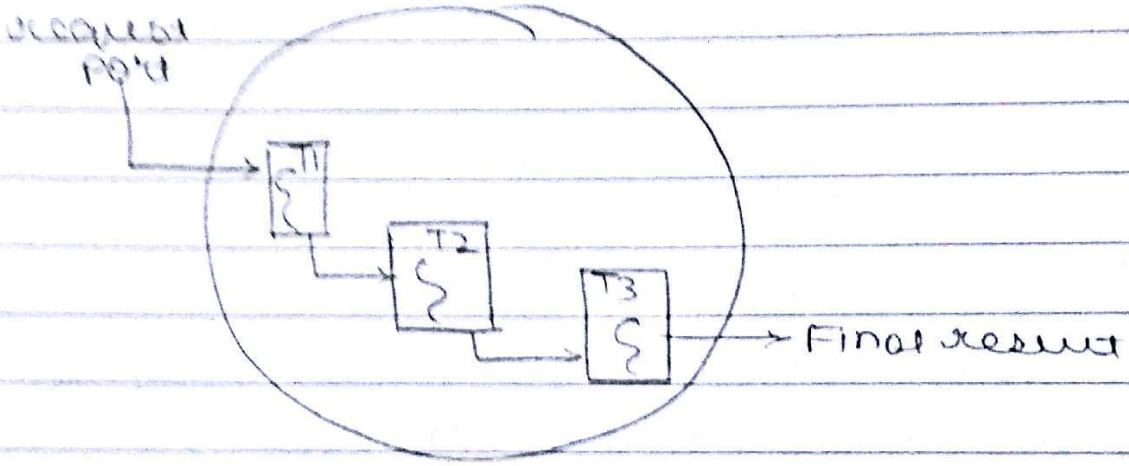
2) Team model :-

Their is no dispatcher here, request comes in queue. & depending on the type of request required thread executes it & gives the result.



Date :
MON TUE WED THU FRI SAT SUN

3) Pipeline model :-
It is suitable when O/P of last thread acts as I/P to next thread.



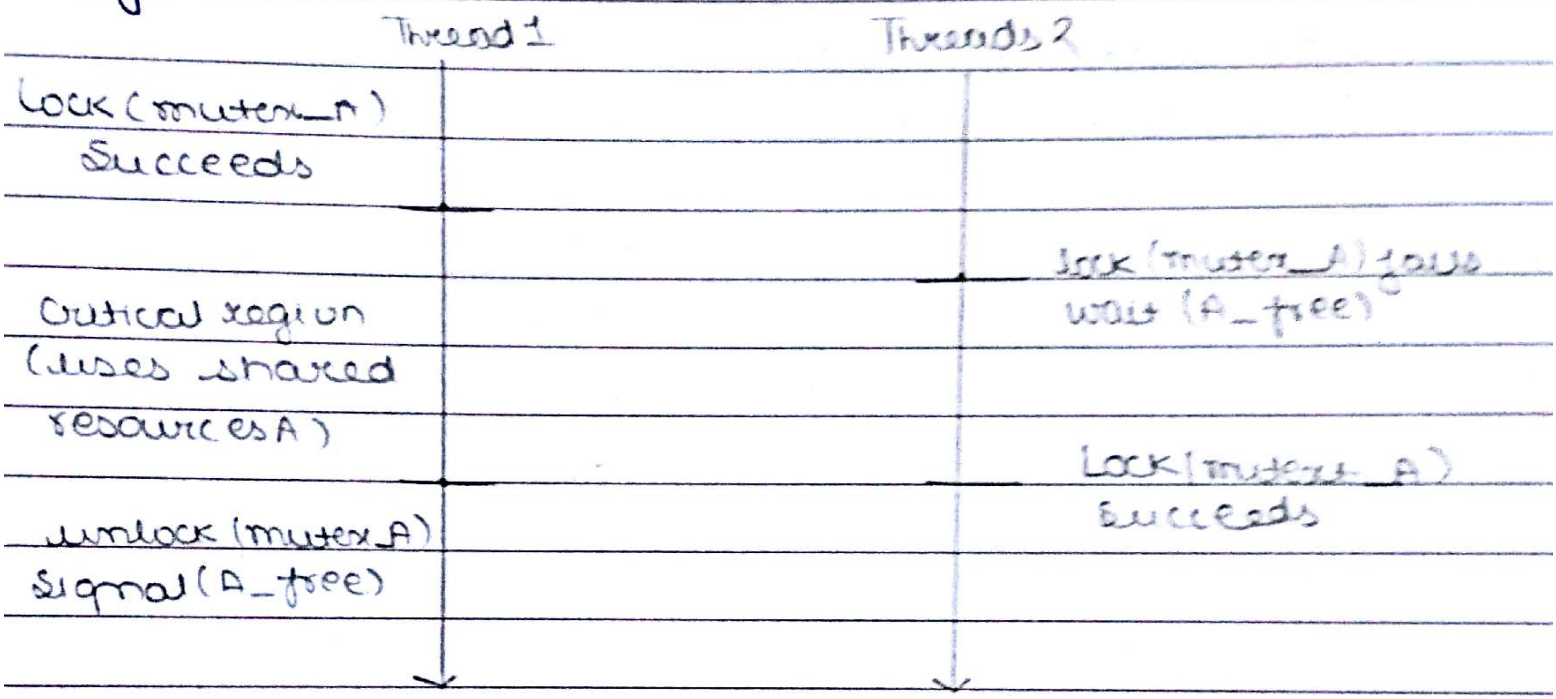
Issues in designing a threads package :-

1) Thread Creation :-
Static :- fixed number of thread.
Dynamic :- Initially single thread depending upon request increase number of thread.

2) Thread Termination :-
either the thread is terminated automatically or manually.

3) Thread Synchronisation :-

two thread trying to access a critical section eg :- A.



mutex_A is mutex variable for exclusive use of shared resource A . A_free = condition variable for resource A to become free

4) Thread Scheduling :-

1) FCFS (First come First serve)

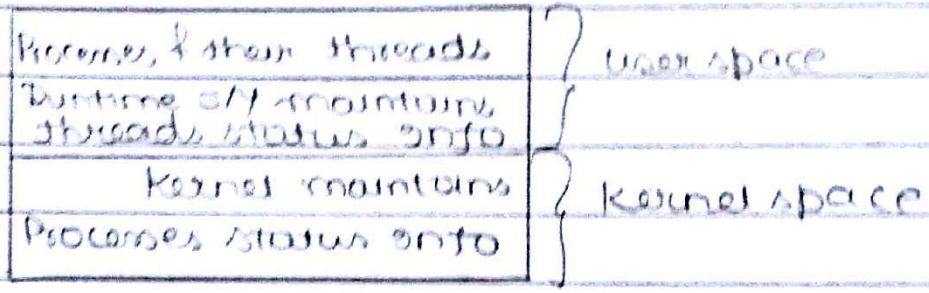
2) R R (Round Robin) :- fixed time quantum.
Dynamic time quantum.

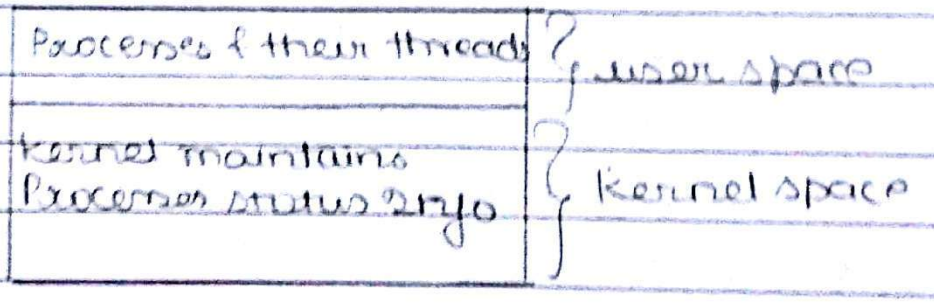
3) Handoff Scheduling :-
one in the execution will tell, which is the next thread to be executed.

4) Affinity Scheduling :-
A thread is scheduled on CPU last it ran on under the assumption that its, address space is in processor cache.

Implementation of Thread package.

There are two types of packages
1) user level packages.
2) kernel level packages.





As shown in the diagram Run time s/y (RTS) is present in user area which manages all the threads.

Kernel level packages:- No RTS is maintained here & kernel is responsible to perform Thread management.