Scheduling

What is process Scheduling? & describe different types of Scheduler?

=> It is the task of the OS to Schedule the processes of different states like Ready, Running, waiting ... According to Scheduling algorithm with the help of different Schedulers.

=> Process Scheduling allocates the CPU time to each process for execution.

Why do we need?

- For CPU utilization in multi-programming & multi-tasking os we need Scheduling.

Def: The act of determining which process is in the ready state and should be moved to the running state is known as process Scheduling.

=> The number of processes is more than resources we have only one processor, but process are more than one.

=> These process are not sensible enough the Schedule themselves one after another so the OS take this responsibility.

=> OS Schedule these process using the Scheduling algorithms.

Prime purpose of process scheduling is to keep the CPU busy all the time and to deliver minimum response time for all programs.

Scheduling Queues:

=> A place where the processes waits to get the resources are known as job queue.

=> There are three kinds of queues:
- Job Queue
- Ready Queue
- Waiting Queue

Job queue:
- when a process enters in the system, it is placed in a queue. That queue is called job queue.
- These queue of the process resides in the secondary memory

Ready Queue:
- The processes that are placed in the main memory and waiting for the CPU time are called ready queue.

Waiting queue:
- if the process is waiting for some kind of I/O operations it will be kept in waiting queue & wait here.

Process migration:
- A process begins in the job queue & moves b/w the ready queue and waiting queue and eventually finishes the task. This process of moving b/w the queue is called process migration.

What is Schedules?

=> The part of OS that "allots the Computer resources" to the processes is termed as Schedules.

=> It uses "Scheduling algorithms" to decide which process it must allot to the CPU.

Types of Schedules

=> There are three types of Schedules.
- Long-term Schedules
- Short-term Schedules
- medium term Schedules

1) Long-term Schedules:-

- Responsible for managing the process in a job queue which is in the secondary memory.
- it check which process want to go first priorities that process in the list
- Also control the no of processes in the memory is called the degree of multi-programming.

Steps it perform:

- Get a new job, check it priority, put it in the queue; give it to the ready queue.
- Also known as job scheduler.
- it has less scope in time-sharing os.
- It executes relatively infrequently.

Use:

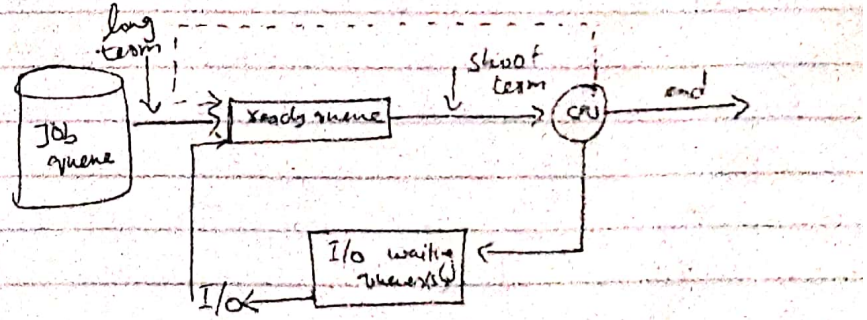
- it is mainly present in Batch os like IMB 1360 mainframe Computers.
- it is not present in 'windows' os.

⇒ it bring "new process to ready state".

Short-term Schedulers:-

- Responsible for managing the process in the ready queue which is in the main memory.
- Process also gets the processes from this queue because it is near the processor. It is also called CPU Scheduler.
- In simple, select a process from ready queue to running state for CPU.
- In some OS it is called dispatcher which is responsible for
  - Switching to users mode
  - Context switching.
- It is also invoked when an event occurs which may lead to interruption of the current process.
- Highly frequently execute.

Diagram:

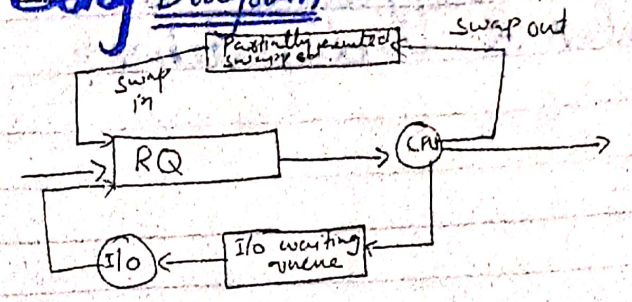


iii) Medium term Shedules:-

=> Responsible to keep the things in the smooth flow.
=> it smooth by keeping the ways of the process clear.
- o who blocks the ways other process?
- o A process may come into the main memory and may do nothing other than keeping the memory reserve.

=> Medium term shedules take this process out of the main memory and place it in the secondary memory.
=> Sometime a process of high priority want to come in the main memory, but the memory is not enough because it is reserved by some low priority process medium term play its role swape that low priority process out and make space for high priority.
- Perform swapping in & swapping out.
- Responsible for suspending & resuming the process.

Long Diagram



PP-SQ (2014-2020)

- o Explain long term Shedules.
- o Difference between short-term & medium term Shedules.

(Long)

Describe the differences b/w long term, short-term and medium term Shedules

What is difference blw Shedules & dispatches?

Dispatches

Def: ○ A dispatches refers to a module that provides the control of the CPU to that process that gets selected by the short-term - Schedules.

Dependency:-

○ The overall working of a dispatches depends entirely on the schedules. it means that the dispatches has to wait unless & until a schedules performs the selection of processes.

Types:

○ has no major type. it is a type of code segment.

Schedules

○ it is basically a system of software that helps in selecting a process out of various available processes.

○ The overall working of a schedules occurs independently. it happens pretty immediately as & when required.

○ Basically, there are three types of schedules.

Time taken:

○ Dispatches latency refers to the time taken by a dispatches to stop one process and to start another process.

Algorithm

○ No specific algorithm for the implementation of a dispatches

Functions:

○ Switching of Context
○ Switching to user mode
○ Jumping to the user's program location for restarting that given program

○ The overall time taken by a schedules is very negligible as we neglect it.

○ A typical schedules can work on various type of algorithm like RR, SJF, FCFS etc.

○ Perform the selection of various processes.

Modes of Scheduling / Types

Non-Preemptive	Preemptive
Def:	
○ In non-preemptive mode, once if a process enters into a running state, it continues to execute until it terminates or blocks itself to wait for I/O or by requesting some OS services	○ In preemptive mode currently running process may be interrupted and moved to the Ready State by the OS.
Basics:	
○ Allocation of resources to process holds till terminated or switch to waiting state	○ Allocation of resources to process for limited time.
Interrupt:	
○ Process cannot be interrupted.	○ Process can be interrupted.
Starvation:	
○ If process of long burst time is running then low burst process has to starve	○ Due to high priority process low priority process has to starve
Flexibility:	
○ Not flexible	○ flexible
Cost:	
○ Not cost associated	○ cost associated
Switching overhead:	
○ It has no overhead of switching process	○ It has overhead of switching process
System use:	
○ Mainly batch System use	○ Time sharing System use
CPU utilization:	
○ Low CPU utilization	○ High
Example:	
○ FCFS ○ SJF	○ Round Robin ○ SRTF ○ Priority

Terminology used in Sheduling Algorithm:-

1) Arrival time:

- Time at which process arrives in ready queue.
- Arrival time can vary from process to process.

main memory میں جب CPU Process آئے

2) Completion time:

- Time at which process completes its execution.

process جب اپنا کام مکمل کر لیتا ہے CPU

3) Burst time: /Execution time/ Running time:

- Time required by the process for CPU execution.

process کو اپنا کام مکمل کرنے میں لگنے والا time

4) Response time:

- Time it takes to start responding from arrival time.

CPU کو respond کرنے میں - when process come for it is the duration b/w the arrival of a CPU process and the first response time

Time taken by a process since it enters a ready queue for the process execution till the completion

5) Turn around time:

- How long a process takes to execute it, limited by the speed of the o/p device
- Calculated by the following formula

Turn around time: Completion time - Arrival time

TAT = CT - AT

TAT = WT + BT - Use time کو بھی CPU

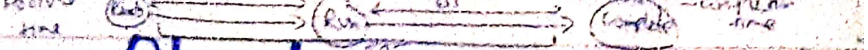
6) Waiting time:

Amount of time that a process spends while waiting in a ready queue until it gets the CPU (for the execution)

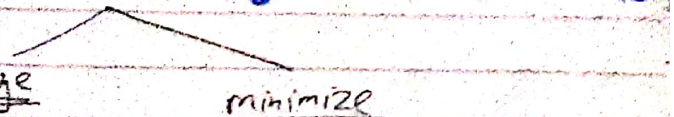
- Sum of all times spent in waiting queue.
- Arrival time is the time a process enters the ready queue
- it is calculated by the following formula. It is not limited by the speed of the o/p device

W.t = TAT - BT

W.t = finish time - Arrival time - burst time



Objective of CPU Shedulig



CPU Scheduling Criteria

- Every CPU algorithm has different properties, so criteria used for comparing these algorithms.

1) CPU utilization:
   - Keep CPU as busy as possible
   Range: 0 — 100%.
   in practice: 40% — 90%.

2) Throughput:
   - Rate at which process are completed per unit time.
   - Increases for short process
   - Decreases for process of huge size.

3) Average turn around time:
   - $\frac{\text{Sum of all TAT of Process}}{\text{No of process}}$

4) Average waiting time:
   - $\frac{\text{Sum of all waiting time of Process}}{\text{No of process.}}$

5) Average response time:
   - $\frac{\text{Sum of all response time of Process}}{\text{No of Process.}}$

6) Fairness of CPU
   Share in fairly manner with every process (i.e. that process does not go into starvation)

Scheduling algorithm

- A scheduling algorithm is the algorithm which tells us how much CPU time we can allocate to the processes and when.

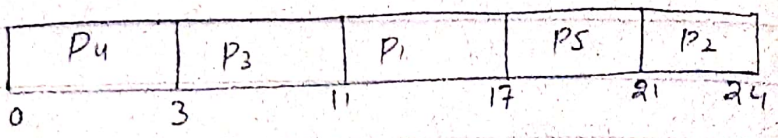
1) First Come, First served Scheduling

- CPU Scheduling algorithm is the FCFS algorithm.
- The process that requests the CPU first is allocated the CPU first.
- It can be managed with FIFO queue.
- + working from back.

Pros:	Cons:
- Simple method - No Starvation - Less overhead - Better for long Processes - Non-preemptive - Convey effect Criteria: "Arrival time"	- Avg, WT, Response time is poor - Convey effect

Example :-

Process	AT	BT	FT	TAT (FT-AT)	WT (TAT-Burst)
P_1	2	6	17	17-2=15	15-6=9
P_2	5	3	24	24-5=19	19-3=16
P_3	1	8	11	11-1=10	10-8=2
P_4	0	3	3	3-0=3	3-3=0
P_5	4	4	21	21-4=17	17-4=13

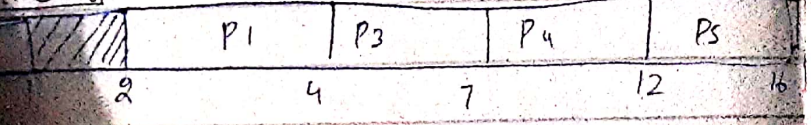


AWT: $\frac{9+16+2+0+13}{5} = \frac{40}{5} = 8$

Ex 2:

Process	AT	BT	FT	TAT (FT-AT)	WT (TAT-BT)	RT (response time)
P_1	2	2	4	4-2=2	2-2=0	0
P_2	0	1	1	1-0=1	0	0
P_3	2	3	7	7-2=5	2	2
P_4	3	5	12	12-3=9	4	4
P_5	4	9	16	16-4=12	3	3

Gantt chart:



AWT = $\frac{0+0+8+4+3}{5} = \frac{9}{5}$

Note: In non-preemptive Scheduling algorithm WT and response time are same.

(ii) Shortest - Job - First: (SJF)

⇒ It overcomes the problem of FCFS algorithm.

⇒ SJF is a type of disk scheduling algorithm in the OS in which the processor executes the job first that has the smallest execution time.

⇒ In SJF processes are Scheduled according to the burst time (reduction time) of these processes.

⇒ This is the best approach to minimize waiting time.

⇒ if two processes have the same burst time the FCFS should be apply.

⇒ This Scheduling algorithm is optimal if all processes are available at the same time (either arrival time is 0 or Arrival time is same for all)

Types:-
- Preemptive (SRTF/SRN)
- Non-preemptive (by default)

Non-preemptive:

=> Once the CPU is given a process it cannot be preempted until the current CPU burst finishes.

Algorithm:
- Sort all the processes according to the arrival time.
- Then select that process that has minimum arrival time and minimum burst time.
- After completion of the process make a pool of processes that comes after wards till the completion of the previous process and select that process among the pool which is having minimum burst time.

Problem with non-preemptive:
- if the arrival time for processes are different, which means all the processes are not available in the ready queue at time 0, and some jobs arrive after some time, in such situation, sometimes process with short burst time have to wait for the current process execution to finish because in non-preemptive SJF, on arrival of a process with short duration, the existing processes execution is not halted/stopped to execute the short job first.
- This leads to the problem of "starvation", where a shorter process has to wait for a long time until the current longer process gets executed. This happens if shorter jobs keep coming, but this can be solved using the concept of aging.

Preemptive Shortest Job first:-

- In preemptive Shortest job first Scheduling, jobs are put into ready queue as they arrive, but as a process with short burst time arrives, the existing process is preempted or removed from execution, and the shortest job is ...

Advantages of SRTF:

- Less waiting time than 'FCFS' and 'SJF'.
- Good response time for short processes
- Remains Convoy effect
- max throughput
- Gives guarantee of less waiting time.

Disadvantages of SRTF:

- It is difficult to estimate remaining time necessary to complete execution.
- Poor response time for long burst time process.
- Starvation is possible for long process. Context switching overhead is here.

SJF With Predict BT:

By:

- Process Size
- Process Type
  - Paid = 100KB ⇒ 5 units
  - Free = 10KB ⇒ 5 units

Simple averaging method:

$$\frac{t_1}{1} + \frac{t_2}{2} + \frac{t_3}{3} + \frac{t_4}{4} + \frac{t_5}{5}$$

$$\frac{1}{n} \sum_{i=1}^{n} t_i$$

Exponential averaging:

$$E_{i+1} = \alpha A_i + (1-\alpha)\varepsilon_i$$

ε_{i+1} = expected time for process $i+1$

A_i = Actual burst time of process i

$0 \le \alpha \le 1$ → better to take in point.

$$\varepsilon_i = \alpha A_{i-1} + (1-\alpha)\varepsilon_{i-1}$$

$$\varepsilon_{i-1} = \alpha A_{i-2} + (1-\alpha)\varepsilon_{i-2}$$

⋮

example:

$\alpha = 0.5$, $\varepsilon_1 = 5$, $\varepsilon_5 = ?$

Process	BT
P_1	4
P_2	8
P_3	5
P_4	6

$\varepsilon_5 = (0.5)6 + (1-0.5) * \varepsilon_4$

$\varepsilon_4 = (0.5)5 + (1-0.5) * \varepsilon_3$

$\varepsilon_3 = (0.5)8 + (1-0.5) * \varepsilon_2$

$\varepsilon_2 = (0.5)4 + (1-0.5)\varepsilon_1$

$\varepsilon_2 = 2.0 + (0.5)(5)$

$\varepsilon_2 = 2.0 + 2.5 = 4.5$

$\varepsilon_3 = 4.0 + (0.5)(4.5)$

$\varepsilon_3 = 6.25$

$\varepsilon_4 = 2.5 + 0.5 * 6.25$

$\varepsilon_4 = 5.625$

$\varepsilon_5 = 5.81$

Note:

When arrival time is same then SRT and is same SJF with non-premption.

Priority Scheduling

- Each process has its own priority.
- out of all available processes highest priority gets the CPU.
- if tie then use FCFS.

Mode: Preemptive & Non-preemptive.
Priority: External & internal.

- Lower the number higher the priority.

Pros:

- Easy
- Good response for the highest priority process

Cons

- indefinite blocking or Starvation

↳ solution

→ Aging: increasing the priority of low priority process based on the its waiting.

Round Robin algorithm:

- Each selected process is assigned a time interval called quantum or time slice.
- Here 2 things are possible:-

1) if a process is blocked or terminated before the time slice has elapsed, then CPU switching is done and other process is Scheduled to run.

2) if process needs the CPU burst longer than 'time slice' then the process is running at the end of time quantum.

⇒ Now that process will preempted move to end of ready queue and CPU will allocated to process which in front of ready queue.

mode: preemptive.

Pros:

- fair amount of CPU is allocated for each job
- it is not affected by convoy effect.

Cons:

- leads to Starvation for processes with larger burst time as they have to repeat the cycle many times.

- Round Robin Scheduling is FCFS scheduling with preemptive mode.

Note:

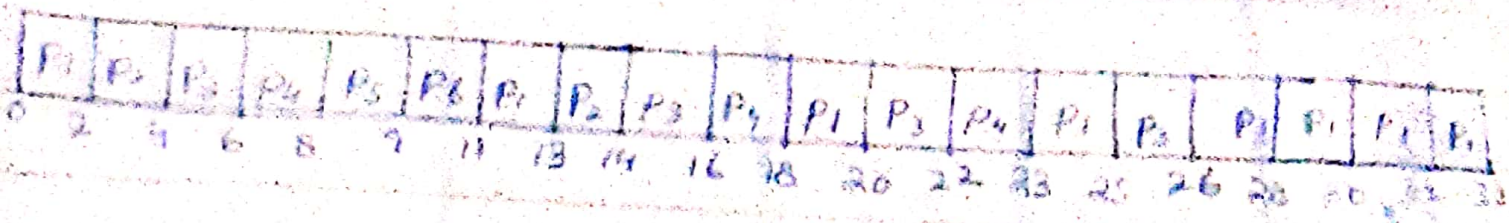
⇒ With decreasing value of time quantum
- No of content switch increase
- Response time decrease
- chances of starvation decreases

- Thus, smaller value of time quantum is better in terms of response time.
- Time quantum should be such that it neither too big nor too small.

PP-question

(iii) Ans:

efficient resource allocation
optimized throughput
fairness
effective resource management



Q. What processor affinity?

⇒ Processor affinity refers to the assignment of specific processes or threads to run on a particular CPU cores of processors with a multicore system.

⇒ it allows for control over CPU resource allocation.

⇒ Processor affinity can be set at the process or thread level, enabling processes or threads to consistently execute on any cores.

⇒ The benefits of processor affinity:

- Performance optimization by reducing cache thrashing
- Control over resource allocation

(iv) (PP-2017)

What is Starvation?

=> Starvation refers to a situation where a process or task is unable to make progress or obtain necessary resources, even if is ready to execute.

=> it occurs when certain processes or tasks are consistently favored over others, leading to a deprivation of resources for the starved process.

=> This can be caused by

- Priority inversion
- unfair resource allocation
- flaws in scheduling algorithm
- Deadlocks.

=> Starvation can negatively impact system performance and fairness, resulting in lower throughput and longer response times.

=> To mitigate starvation, techniques like

- Priority aging
- RR time slicing
- fair resource allocation strategies are employ

(How to create thread in C++)

Past Papers Short Questions

(Ch#5)

(i) What does it mean to preempt a process? (Repeat)

(ii) Difference b/w preemptive and non-preemptive Scheduling? (Repeat)

(iii) What advantages is there in having different quantum sizes at different level of a multi-level queueing system?

(iv) Which one of the following scheduling algorithm could result in Starvation? FCFS, SJF, Round Robin.? (Repeat)

(v) Different Types of real time Scheduling? (Rep

(vi) What is the processor Affinity? (Repeat

(vii) What are Turnaround & response time?

(viii) What is meant by the term dispatch latency? + waiting time? (Repeat)

(ix) Define Starvation?

(x) List the various Criteria must be considered in making a good Scheduling algorithm?

(xi) What is RR Scheduling?