

Assignment - 5

- 1 necessity of operating system and essential abstractions
- modern computer system still heavily on OS despite advantages in hardware because OS serves two crucial, non-redundant roles:-
 - Resources management - It manages and detects & allocates system resources (CPU, memory, I/O device) efficiently.
 - Abstraction layers :- It provides a simple clean and consistent interface to underlying complex, raw hardware making it easier for programmer to write application.
- 2 Comparison of OS Structure :-

→ monolithic :-

All OS services are combined into single binary running in kernel space and has fast execution due to minimal overhead.

layered :-

Services are organized hierarchically in layers with each layer relying only on the function of the layers below it.

Microkernel :-

The kernel is minimal only handling core services other service run as user level servers slowest be cause most service call require context switching and inter process

3 Analysis of Thread & Process Efficiency

→ Process :-

Slow requires allocating new virtual address space a new process central block, and copying data structure

Threads :-

fast creation only requires creating a new thread central block

Efficiency :-

Hence, Threads are most efficient.

4 memory Allocation Simulation (first-fit, Best-fit, Worst-fit)

Process	Requirement
P ₁	12 MB
P ₂	18 MB
P ₃	8 MB

Algorithm

Allocation steps

first-fit

$$\begin{aligned} \text{Total unused space} &: 8\text{MB} + 4\text{MB} \\ &+ 5\text{MB} = 17\text{MB} \end{aligned}$$

 P_2 is waiting.

best-fit

$$\begin{aligned} \text{Total unused space} &: 8\text{MB} + 4\text{MB} \\ &+ 5\text{MB} = 17\text{MB} \end{aligned}$$

 P_2 is waiting

worst-fit

$$\begin{aligned} \text{Total unused space} &: 8\text{MB} + \\ &4\text{MB} + 5\text{MB} = 17\text{MB} \end{aligned}$$

 P_2 is waiting

S FCFS (first-come-first-served)

processes are executed in the order of their arrival time.

- P_1 arrives at 0, executes for 5ms (0-5)
- P_2 arrives at 1, but wait until P_1 finished (5-8)
- P_3 arrives at 2, wait until P_2 finished (8-16)
- P_4 arrives at 3, wait P_3 finished for execution 3 ms (16-19).

P_1	P_2	P_3	P_4
0	5	8	16-19

2 Shortest - Job first (SJF)

→ Process are selected based on the shortest burst time among those that have arrived

- $t = 0$, only P_1 has arrived,
 P_1 starts ($BR = 5$)

$t = 5$; P_1 finishes; P_2 ($AT = 3$, $BR = 3$) now arrived

Selection: P_2 and P_4 both have ($BR = 3$)

- choose P_2 arrives first

P_1 executes:-

$t = 8$, P_2 finishes P_3 ($BR = 8$)
and P_4 ($BR = 3$) remain

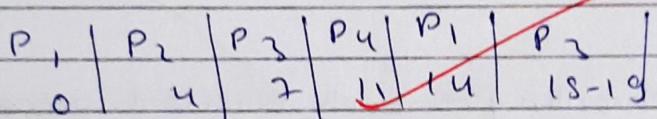
- Selection: P_4 has the shortest
($BR = 3$). P_4 executes ($B = 11$)
 - $t = 11$, P_4 executes only P_3
($BR = 8$), P_3 executed
 $11 - 19$

P_1	P_2	P_4	P_3
0	5	8	11-19

3 Round-Round Quantum = 4 ms

Process are given a maximum of 4 ms CPU time, if not finished they are preempted and moved to the back to the back of the ready queue.

Time (ms)	Execution Process	Remaining BT	Ready Queue
0-4	P ₁ (BT=5)	1	P ₂ (at+1), P ₃ (at+2), P ₄ (at+3)
4-7	P ₂ (BT=3)	0	P ₃ , P ₄ , P ₁ (remain BT=1)
7-11	P ₃ (BT=8)	4	P ₄ , P ₁ , P ₃ (remain BT=5)
11-14	P ₄ (BT=3)	0	P ₁ , P ₃
14-15	P ₁ (BT=1)	0	P ₃
15-19	P ₂ (BT=4)	0	∅



① distributed time management

- (a) Issue in distributed OS Design for file sharing and resource management
 - Global clock inconsistency -

Distributed system lack a single, synchronized clock. This makes it challenging to establish a consistent order of event which is vital for file consistency and lock ordering.

- 2 network latency : The network introduced delay for every file access, severely degrading performance. A network partition can split the system to contradictory update.

- 3 authentication securing resource is complex issue control list must be consistently managed and synchronized across multiple independent servers

- (o) Synchronous checkpointing and recovery
- 1 Initiation :- A check point initiates P, it begins the process by recording its local state and sending a checkpoint request message to all other processes.
 - 2 Co-ordination :- upon receiving the request all processes (Pi) immediately block their normal execution and stop sending application message
 - 3 Local state Recording :- Each blocked process records its correct local state and the state of its communication channels.
- * (i) IoT smart Home system

Process scheduling strategy

- Algorithm solution : preemptive
 - Priority scheduling with interrupt driven activation
 - Synchronization  strategy

- 1 Priority Assignment :- Assign priority to the security device interruption. All other receives low priority.

- 2 Preemption :- We ~~exception~~ preemption a high-priority security interrupt access while a low priority task will immediately respond to power-priority tasks.

~~dispatch the critical interrupt handles~~

3. Justification:- This ensures minimal latency for life safety and security critical tasks, fulfilling the primary requirement of priority interrupts. The low-priority can tolerate delay but securing alerts can not.

Satish
21/11/95