PES UNIVERSITY ONLINE

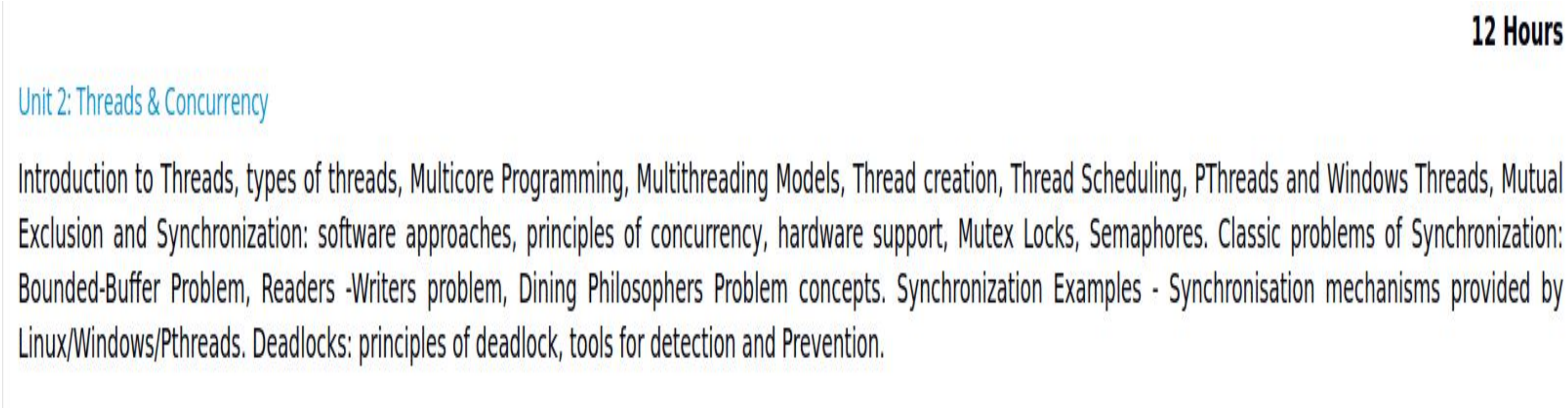
OPERATING SYSTEMS

Deadlocks - 3

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

OPERATING SYSTEMS

Course Syllabus - Unit 2

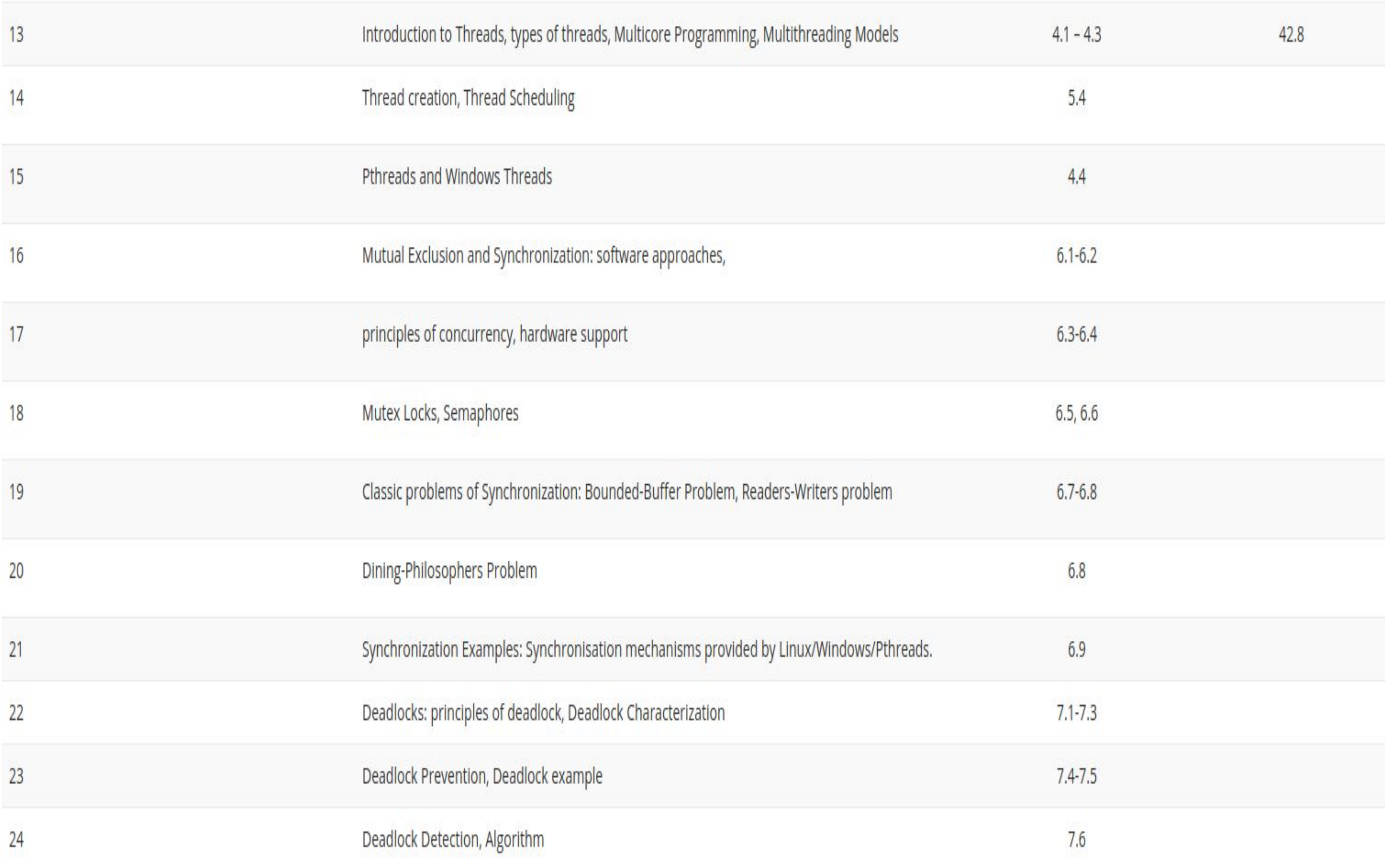


12 Hours

Unit 2: Threads & Concurrency

Introduction to Threads, types of threads, Multicore Programming, Multithreading Models, Thread creation, Thread Scheduling, PThreads and Windows Threads, Mutual Exclusion and Synchronization: software approaches, principles of concurrency, hardware support, Mutex Locks, Semaphores. Classic problems of Synchronization: Bounded-Buffer Problem, Readers -Writers problem, Dining Philosophers Problem concepts. Synchronization Examples - Synchronisation mechanisms provided by Linux/Windows/Pthreads. Deadlocks: principles of deadlock, tools for detection and Prevention.

Course Outline



13	Introduction to Threads, types of threads, Multicore Programming, Multithreading Models	4.1 - 4.3	42.8
14	Thread creation, Thread Scheduling	5.4	
15	Pthreads and Windows Threads	4.4	
16	Mutual Exclusion and Synchronization: software approaches,	6.1-6.2	
17	principles of concurrency, hardware support	6.3-6.4	
18	Mutex Locks, Semaphores	6.5, 6.6	
19	Classic problems of Synchronization: Bounded-Buffer Problem, Readers-Writers problem	6.7-6.8	
20	Dining-Philosophers Problem	6.8	
21	Synchronization Examples: Synchronisation mechanisms provided by Linux/Windows/Pthreads.	6.9	
22	Deadlocks: principles of deadlock, Deadlock Characterization	7.1-7.3	
23	Deadlock Prevention, Deadlock example	7.4-7.5	
24	Deadlock Detection, Algorithm	7.6	

Topic Outline

- Examples : Deadlock Avoidance & Deadlock Detection

Example 1 - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax		
A	B	C
10	5	7

Available=>Rmax-Allocated		
A	B	C
3	3	2

Process	Allocation			Max		
	A	B	C	A	B	C
P0	0	1	0	7	5	3
P1	2	0	0	3	2	2
P2	3	0	2	9	0	2
P3	2	1	1	2	2	2
P4	0	0	2	4	3	3
Total	7	2	5	25	12	12

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			work<= Available => (Rmax-Allocated)		
A	B	C	A	B	C
10	5	7	3	3	2

i	Not Initialised

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	3	3	2

i	Not Initialised

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	False
P1	False
P2	False
P3	False
P4	False

Safe Sequence				

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	3	3	2

i	1

Work		
A	B	C
3+2	3+0	2+0

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	False
P1	True
P2	False
P3	False
P4	False

Safe Sequence				

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	5	3	2

i	3

Work		
A	B	C
5+2	3+1	2+1

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	False
P1	True
P2	False
P3	True
P4	False

Safe Sequence				
P1	P3			

Deadlocks: Deadlock Avoidance & Detection - Examples

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax		
A	B	C
10	5	7

Work		
A	B	C
7	4	3

i	4

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	False
P1	True
P2	False
P3	True
P4	True

Work		
A	B	C
7+0	4+0	3+2

Safe Sequence				
P1	P3	P4		

Deadlocks: Deadlock Avoidance & Detection - Examples

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	7	4	5

i	2

Work		
A	B	C
7+3	4+0	3+2+2

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	False
P1	True
P2	True
P3	True
P4	True

Safe Sequence				
P1	P3	P4	P2	

Deadlocks: Deadlock Avoidance & Detection - Examples

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	10	4	7

i	0

Work		
A	B	C
10+0	4+1	7+0

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	True
P1	True
P2	True
P3	True
P4	True

Safe Sequence				
P1	P3	P4	P2	P0

Example - Safety Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken is as follows then find the system is in a safe state or not, after finding the need

RMax			Work		
A	B	C	A	B	C
10	5	7	10	5	7

i	0

Process	Allocation			Max			Need=>Max-Allocated		
	A	B	C	A	B	C	A	B	C
P0	0	1	0	7	5	3	7	4	3
P1	2	0	0	3	2	2	1	2	2
P2	3	0	2	9	0	2	6	0	0
P3	2	1	1	2	2	2	0	1	1
P4	0	0	2	4	3	3	4	3	1
Total	7	2	5	25	12	12	18	10	7

Process	Flag
P0	True
P1	True
P2	True
P3	True
P4	True

Work		
A	B	C
10	5	7

Safe Sequence				
P1	P3	P4	P2	P0

Example 1 - Resource Request Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken if the system is safe; a new request comes from P1 =>(1,0,2). Can the resource request be granted immediately and safely

RMax		
A	B	C
10	5	7

Available=>Rmax-Allocated		
A	B	C
3	3	2

Process	Allocation			Max		
	A	B	C	A	B	C
P0	0	1	0	7	5	3
P1	2	0	0	3	2	2
P2	3	0	2	9	0	2
P3	2	1	1	2	2	2
P4	0	0	2	4	3	3
Total	7	2	5	25	12	12

Request by P1		
A	B	C
1	0	2

Example 1 - Resource Request Algorithm

Suppose we have 5 processes(P0, P1, P2, P3, P4) and 3 resource types(A, B, C) each having (10,5,7) instances. Suppose at time t1 if the snapshot of the system taken if the system is safe; a new request comes from P1 =>(1,0,2). Can the resource request be granted immediately and safely

RMax		
A	B	C
10	5	7

Work		
A	B	C
5	3	2

Request by P1		
A	B	C
1	0	2

i	1

Process	Allocation			Max		
	A	B	C	A	B	C
P0	0	1	0	7	5	3
P1	0	0	0	3	2	2
P2	3	0	2	9	0	2
P3	2	1	1	2	2	2
P4	0	0	2	4	3	3
Total	7	2	5	25	12	12

Process	Flag
P0	False
P1	True
P2	False
P3	False
P4	False

Work		
A	B	C
5	3	2

Safe Sequence				
P1				

THANK YOU

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

nitin.pujari@pes.edu

For Course Deliverables by the Anchor Faculty click on www.pesuacademy.com