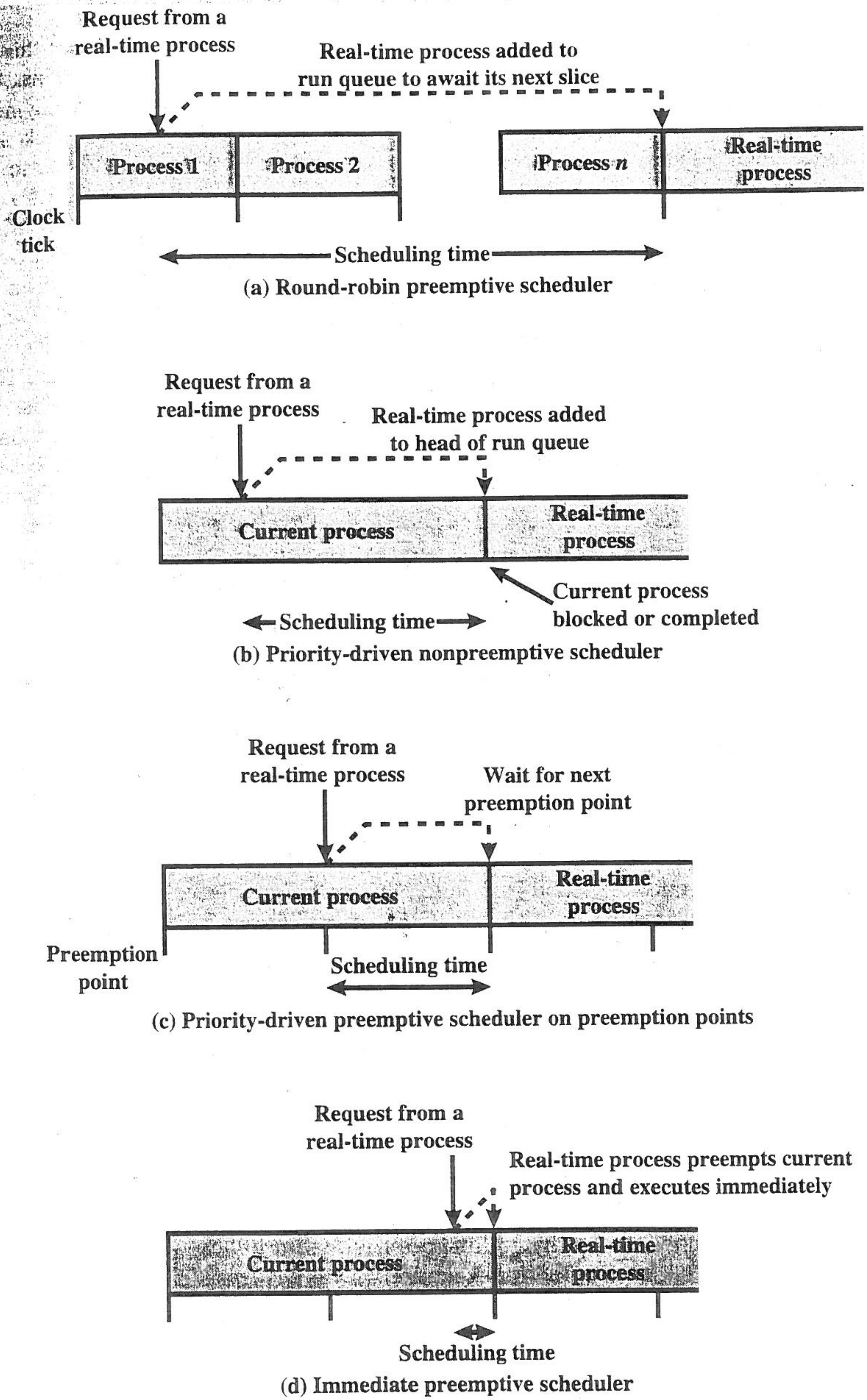

Request from a real-time process
Real-time process added to run queue to await its next slice
Process 1
Process 2
Process n
Real-time process
Clock tick
Scheduling time
Request from a real-time process
Real-time process added to head of run queue
Current process
Real-time process
Current process blocked or completed
Scheduling time
Request from a real-time process
Wait for next preemption point
Current process
Real-time process
Preemption point
Scheduling time
Request from a real-time process
Real-time process preempts current process and executes immediately
Current process
Real-time process
Scheduling time

(a) Round-robin preemptive scheduler

(b) Priority-driven nonpreemptive scheduler

(c) Priority-driven preemptive scheduler on preemption points

(d) Immediate preemptive scheduler

Figure 10.4 Scheduling of Real-Time Process

Table 10.2 Execution Profile of Two Periodic Tasks

Process	Arrival Time	Execution Time	Ending Deadline
A(1)	0	10	20
A(2)	20	10	40
A(3)	40	10	60
A(4)	60	10	80
A(5)	80	10	100
•	•	•	•
•	•	•	•
•	•	•	•
B(1)	0	25	50
B(2)	50	25	100
•	•	•	•
•	•	•	•
•	•	•	•

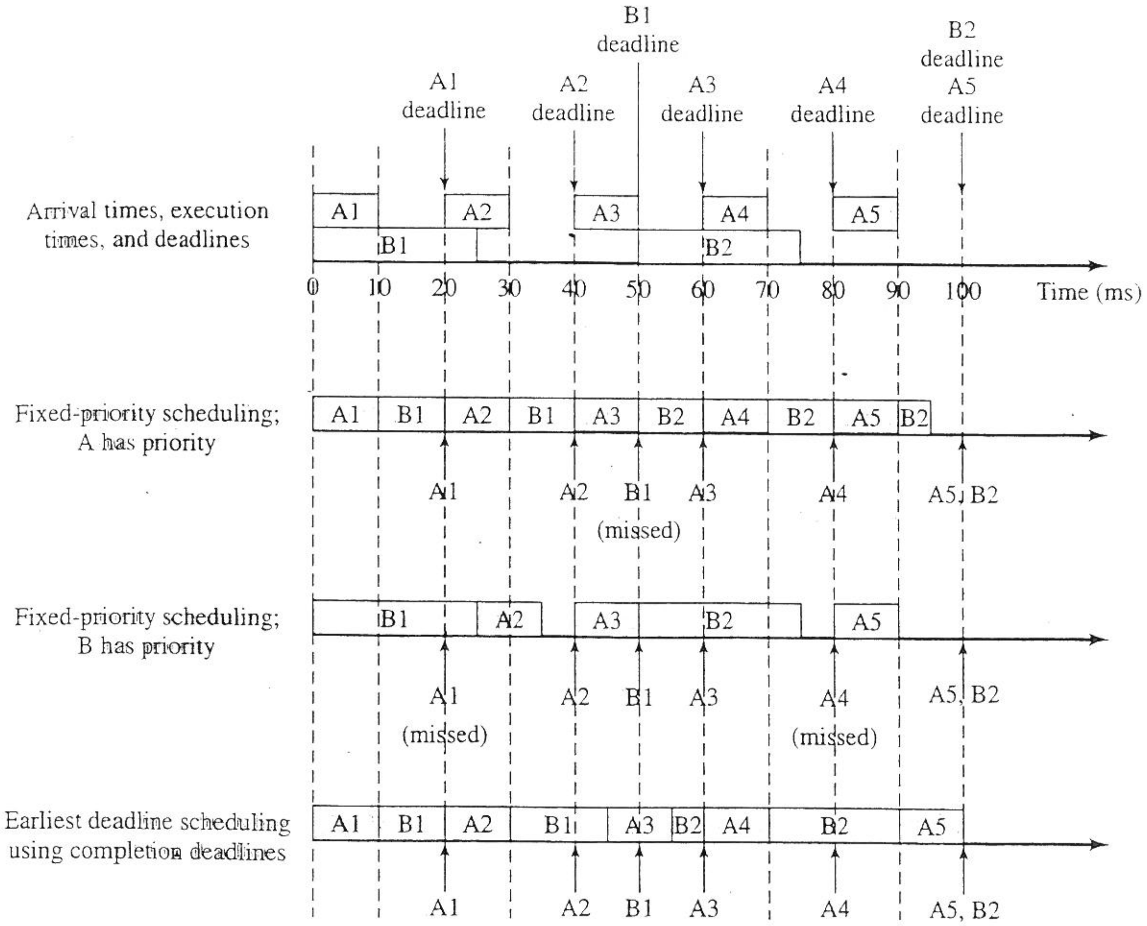

B1 deadline
B2 deadline
A5 deadline
A1 deadline
A2 deadline
A3 deadline
A4 deadline
Arrival times, execution times, and deadlines
A1
A2
A3
A4
A5
B1
B2
0
10
20
30
40
50
60
70
80
90
100
Time (ms)
Fixed-priority scheduling; A has priority
A1 B1 A2 B1 A3 B2 A4 B2 A5 B2
A1
A2
B1 (missed)
A3
A4
A5, B2
Fixed-priority scheduling; B has priority
B1 A2 A3 B2 A5
A1 (missed)
A2
B1
A3
A4 (missed)
A5, B2
Earliest deadline scheduling using completion deadlines
A1 B1 A2 B1 A3 B2 A4 B2 A5
A1
A2
B1
A3
A4
A5, B2

Figure 10.5 Scheduling of Periodic Real-time Tasks with Completion Deadlines (based on Table 10.2)

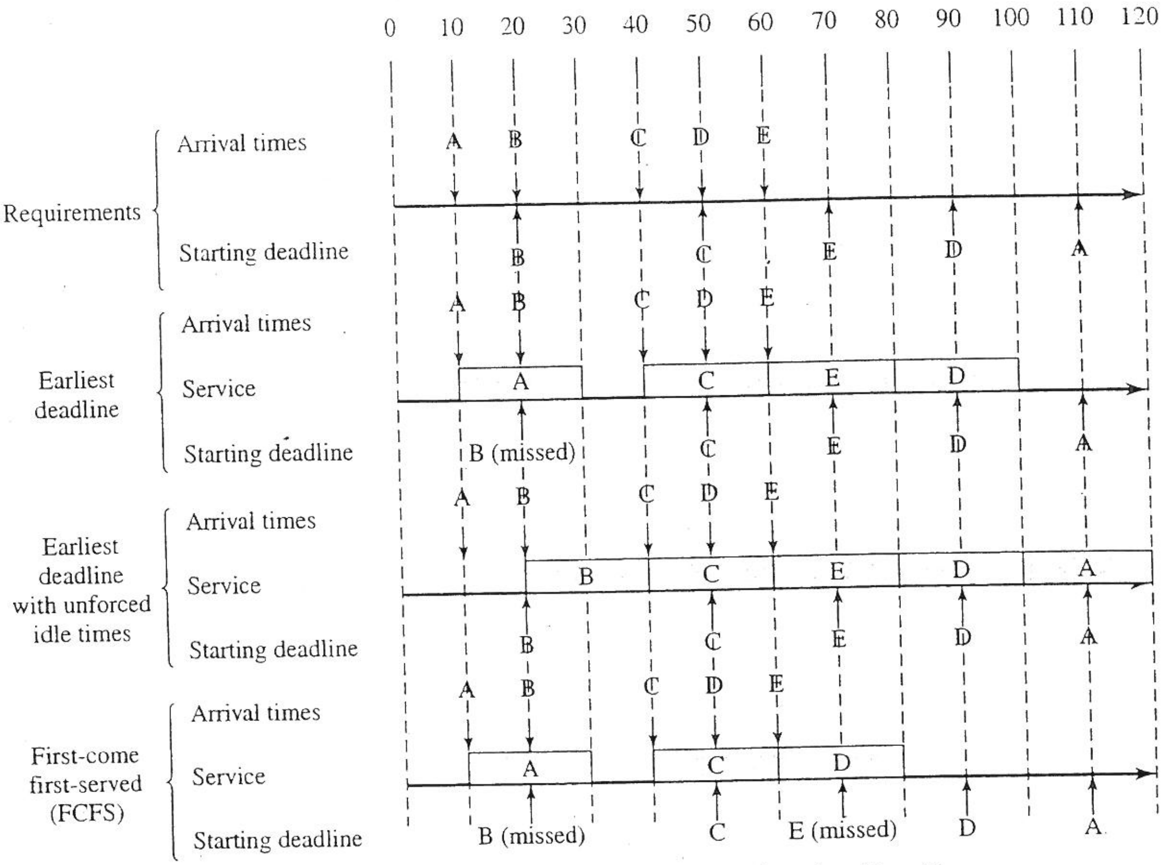


Figure 10.6 Scheduling of Aperiodic Real-Time Tasks with Starting Deadlines

Table 10.3 Execution Profile of Five Aperiodic Tasks

Process	Arrival Time	Execution Time	Starting Deadline
A	10	20	110
B	20	20	20
C	40	20	50
D	50	20	90
E	60	20	70

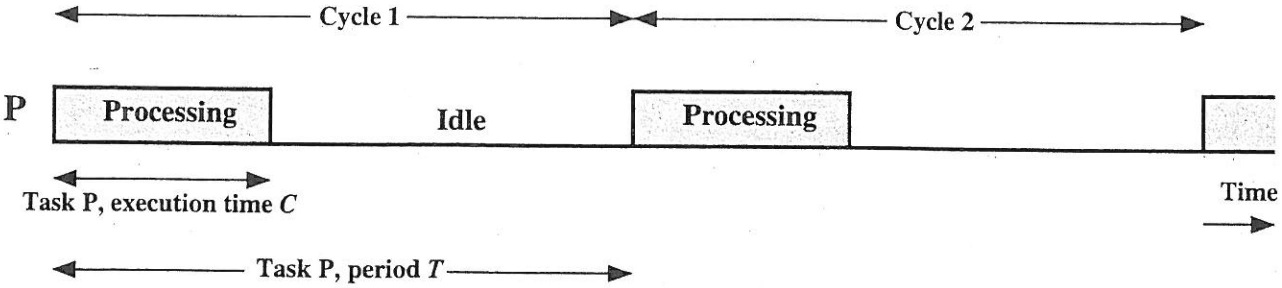


Figure 10.7 Periodic Task Timing Diagram

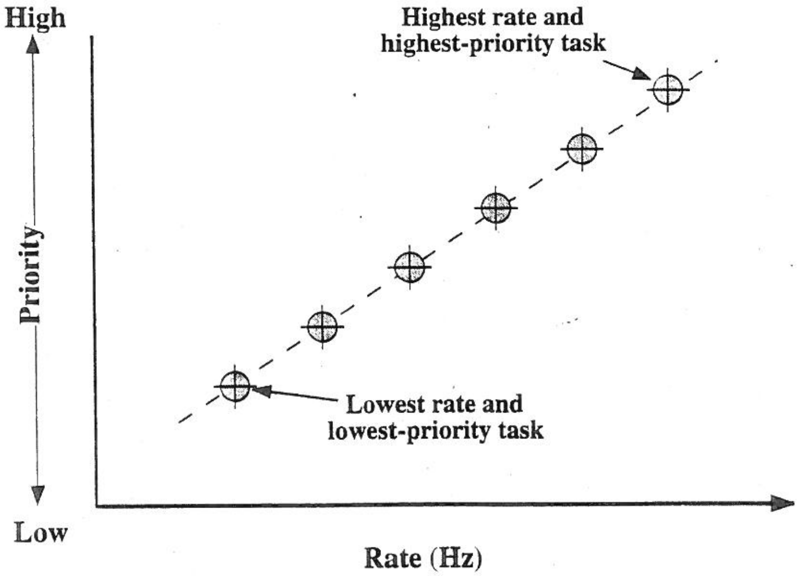


Figure 10.8 A Task Set with RMS [WARR91]

Table 10.4 Value of the RMS Upper Bound

n	$n(2^{1/n}-1)$
1	1.0
2	0.828
3	0.779
4	0.756
5	0.743
6	0.734
⋮	⋮
∞	$\ln 2 \approx 0.693$

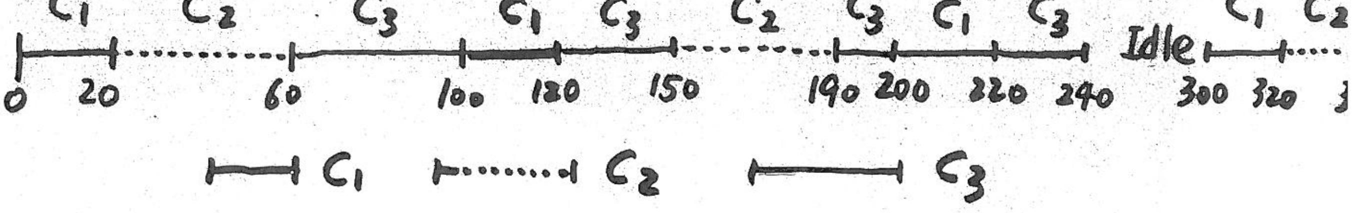
Example 1. $P_1 : C_1 = 20; \ T_1 = 100; \ U_1 = 0.2$

$P_2 : C_2 = 40; \ T_2 = 150; \ U_2 = 0.267$

$P_3 : C_3 = 100; \ T_3 = 350; \ U_3 = 0.286$

$$U_1 + U_2 + U_3 = 0.753 \leq 3(2^{1/3} - 1) = 0.779$$

* This implies that using RMS, these 3 tasks can be scheduled.



Mars Pathfinder — The rover robot landed on Mars on July 4. 1997.

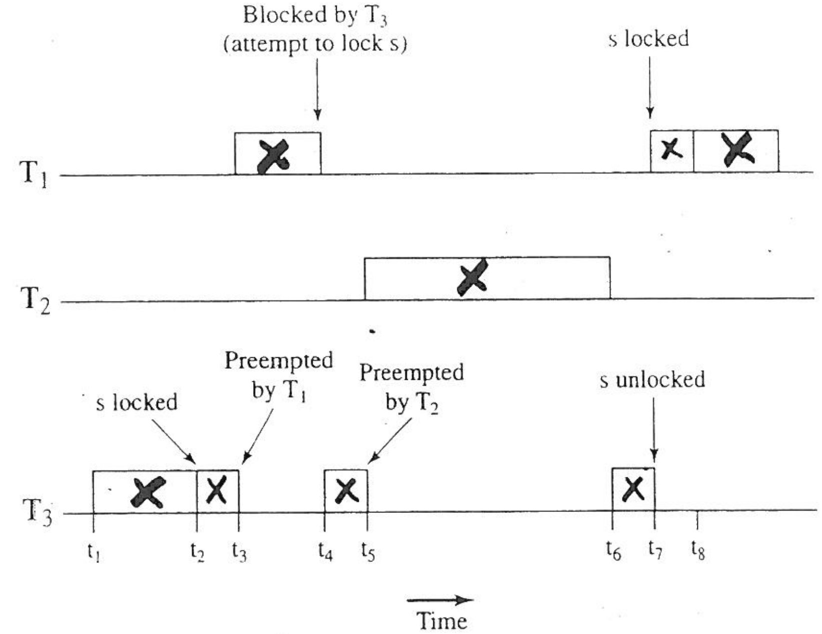
The pathfinder software has the following 3 tasks, in decreasing order of priori

T_1: Periodically checks the health of the spacecraft systems and software

T_2: Processes image data

T_3: Performs an occasional test on equipment status

T1. T3 share a data structure protected by semaphore s



(a) Unbounded priority inversion

t_1: T_3 begins executing.

t_2: T_3 locks semaphore **s** and enters its critical section.

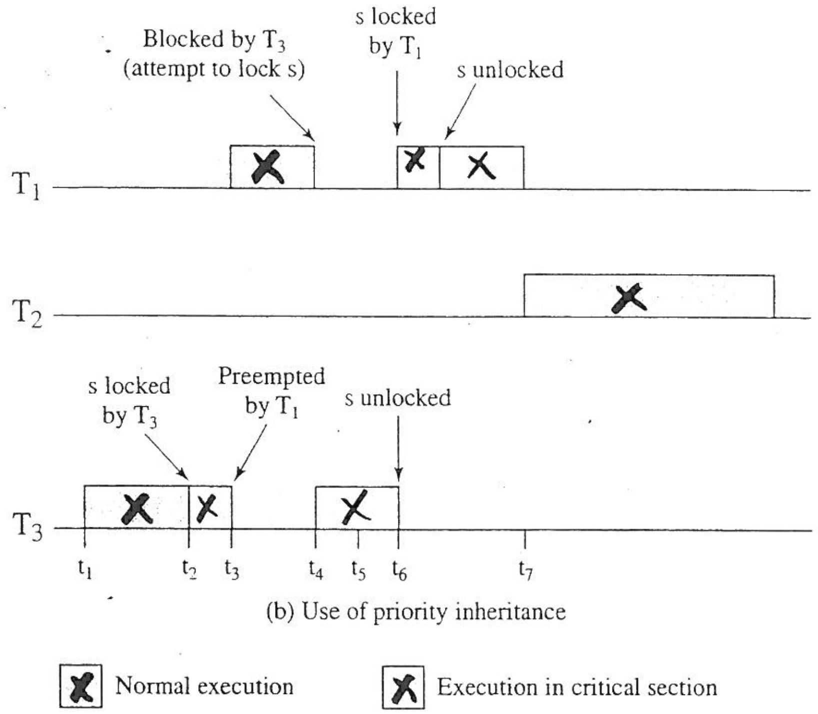
t_3: T_1, which has a higher priority than T_3, preempts T_3 and begins executing.

t_4: T_1 attempts to enter its critical section but is blocked because the semaphore is locked by T_3; T_3 resumes execution in its critical section.

t_5: T_2, which has a higher priority than T_3, preempts T_3 and begins executing.

t_6: T_2 is suspended for some reason unrelated to T_1 and T_2, and T_3 resumes.

t_7: T_3 leaves its critical section and unlocks the semaphore. T_1 preempts T_3, locks the semaphore, and enters its critical section.



(b) Use of priority inheritance

Figure 10.9 Priority Inversion

t_1: T_3 begins executing.

t_2: T_3 locks semaphore **s** and enters its critical section.

t_3: T_1, which has a higher priority than T_3, preempts T_3 and begins executing.

t_4: T_1 attempts to enter its critical section but is blocked because the semaphore is locked by T_3. T_3 is immediately and temporarily assigned the same priority as T_1. T_3 resumes execution in its critical section.

t_5: T_2 is ready to execute but, because T_3 now has a higher priority, T_2 is unable to preempt T_3.

t_6: T_3 leaves its critical section and unlocks the semaphore: Its priority level is downgraded to its previous default level. T_1 preempts T_3, locks the semaphore, and enters its critical section.

t_7: T_1 is suspended for some reason unrelated to T_2, and T_2 begins executing.