Real Time Systems and Control Applications

Contents

PIP

PCP

NPCS → PIP

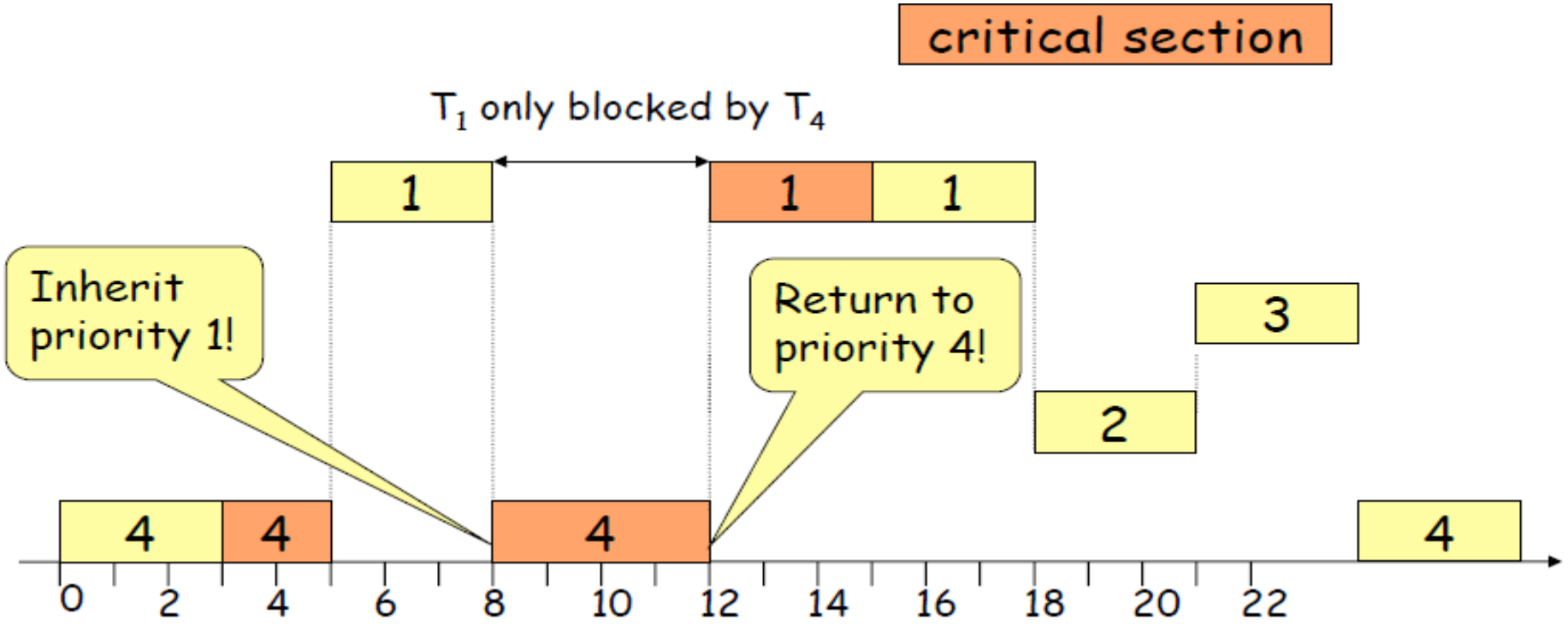
- NPCS: While a task holds or locks a resource it executes at a priority higher than the priorities of all tasks, and a task recovers its priority when resources are unlocked.

- Disadvantage of NPCS: A task can be blocked by a lower priority task, even without a resource conflict

- Priority Inheritance Protocol (PIP): Increase the priorities only upon resource contention

Solution --- Priority Inheritance

- Under priority inheritance the priority of the task that is in the critical section inherits the priority of a higher priority task when the higher priority task requests the resource.



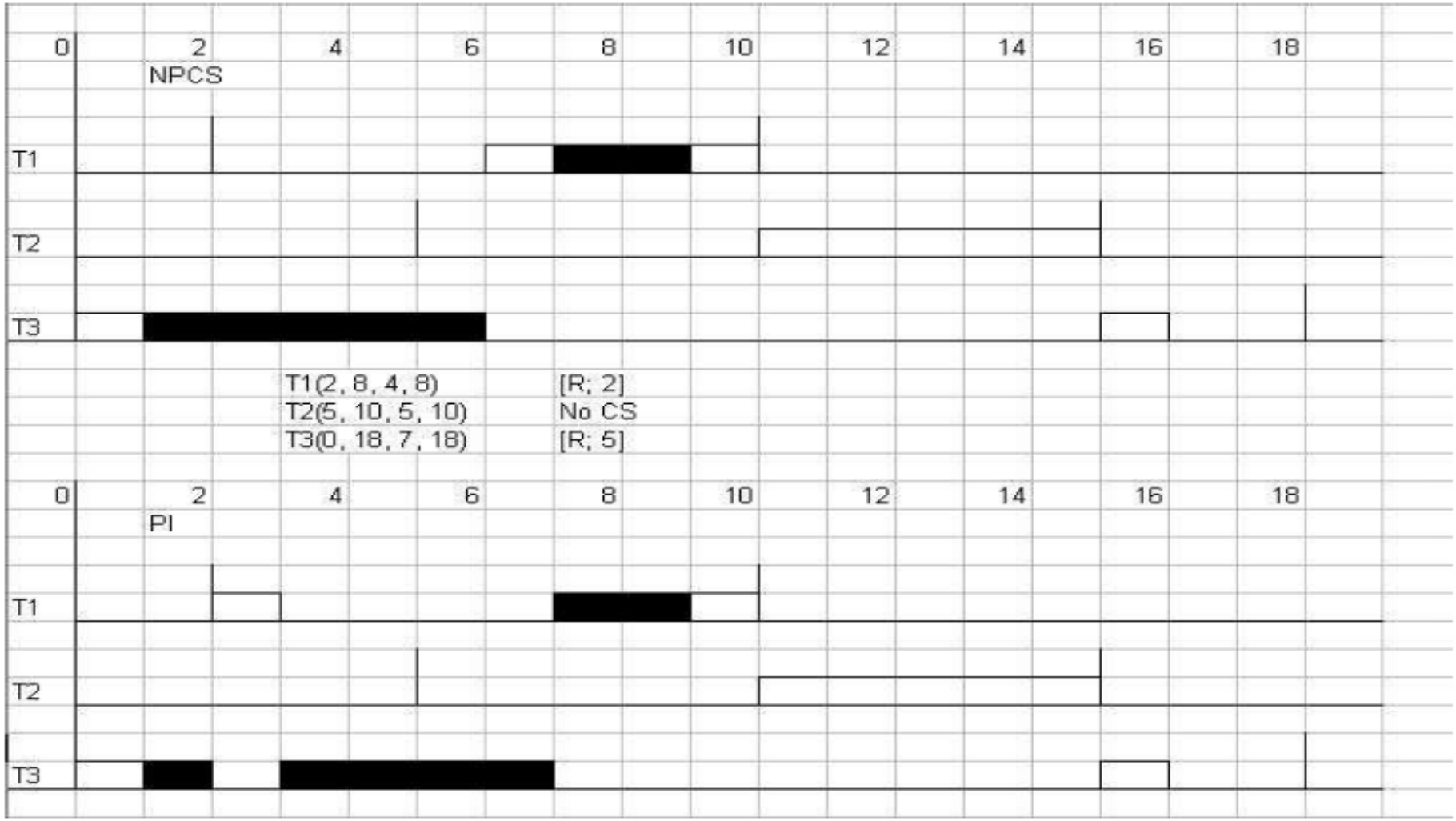
Priority Inheritance Rule

- When a task T1 is blocked due to non availability of a resource that it needs, the task T2 that holds the resource and consequently blocks T1, and T2 inherits the current priority of task T1.

- T2 executes at the inherited priority until it releases R.

- Upon the release of R, the priority of T2 returns to the priority that it held when it acquired the resource R.

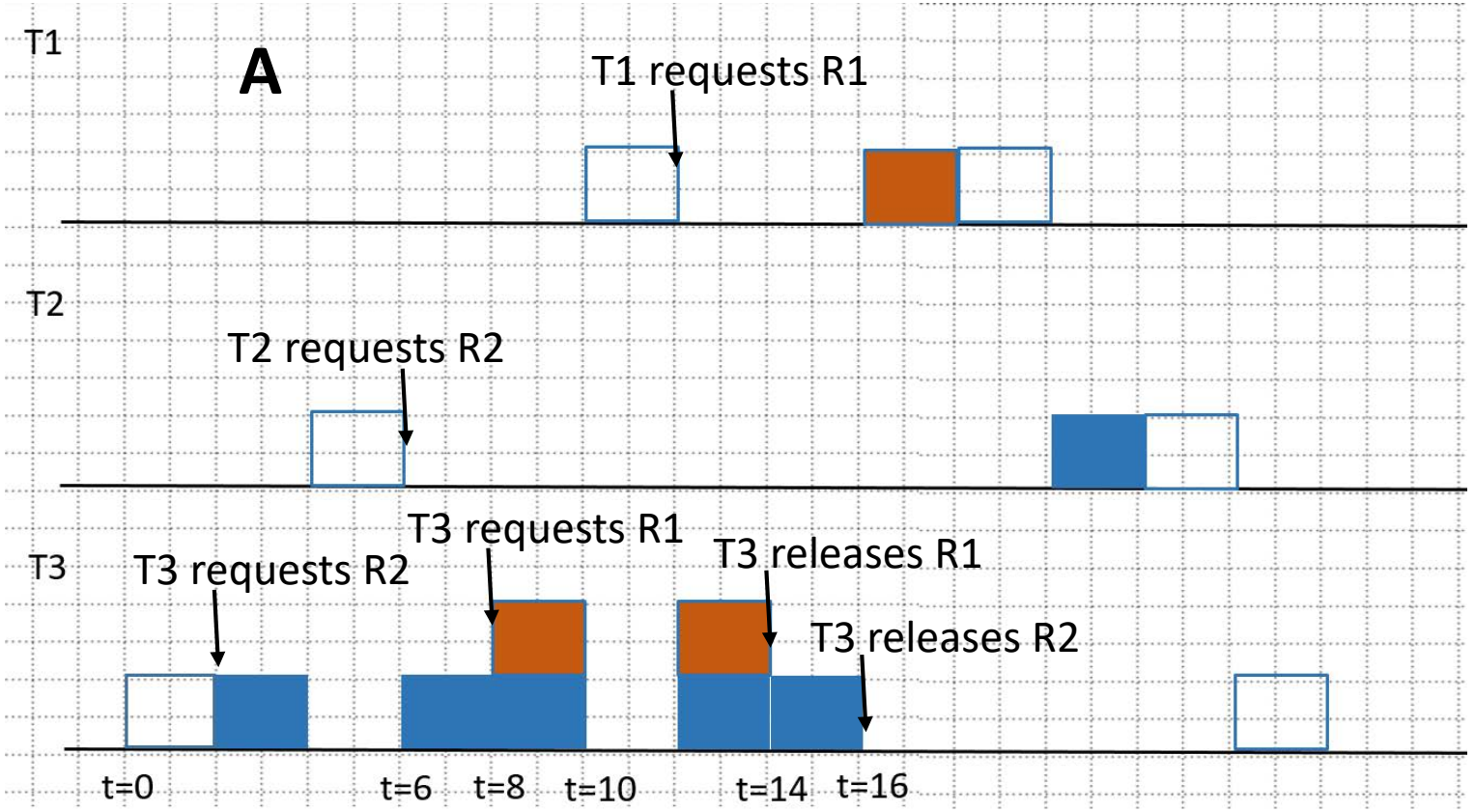
NPCS v.s. PIP


0
2
4
6
8
10
12
14
16
18
NPCS
T1
T2
T3
T1(2, 8, 4, 8)
T2(5, 10, 5, 10)
T3(0, 18, 7, 18)
[R; 2]
No CS
[R; 5]
0
2
4
6
8
10
12
14
16
18
PI
T1
T2
T3

Priority Inheritance Protocol (PIP)

- If lower priority task TL blocks a higher priority task TH, priority(TL) ← priority(TH)
- When TL releases a resource, it returns to its normal priority if it doesn't block any task. Or it returns to the highest priority of the tasks waiting for a resource held by TL (example on next page)
- Transitive
  - T1 blocked by T2: priority(T2) ← priority(T1)
  - T2 blocked by T3: priority(T3) ← priority(T1)

Which one of the scheduling output is correct? Assume T1 has priority high, T3 has priority low, and T2 has priority medium.

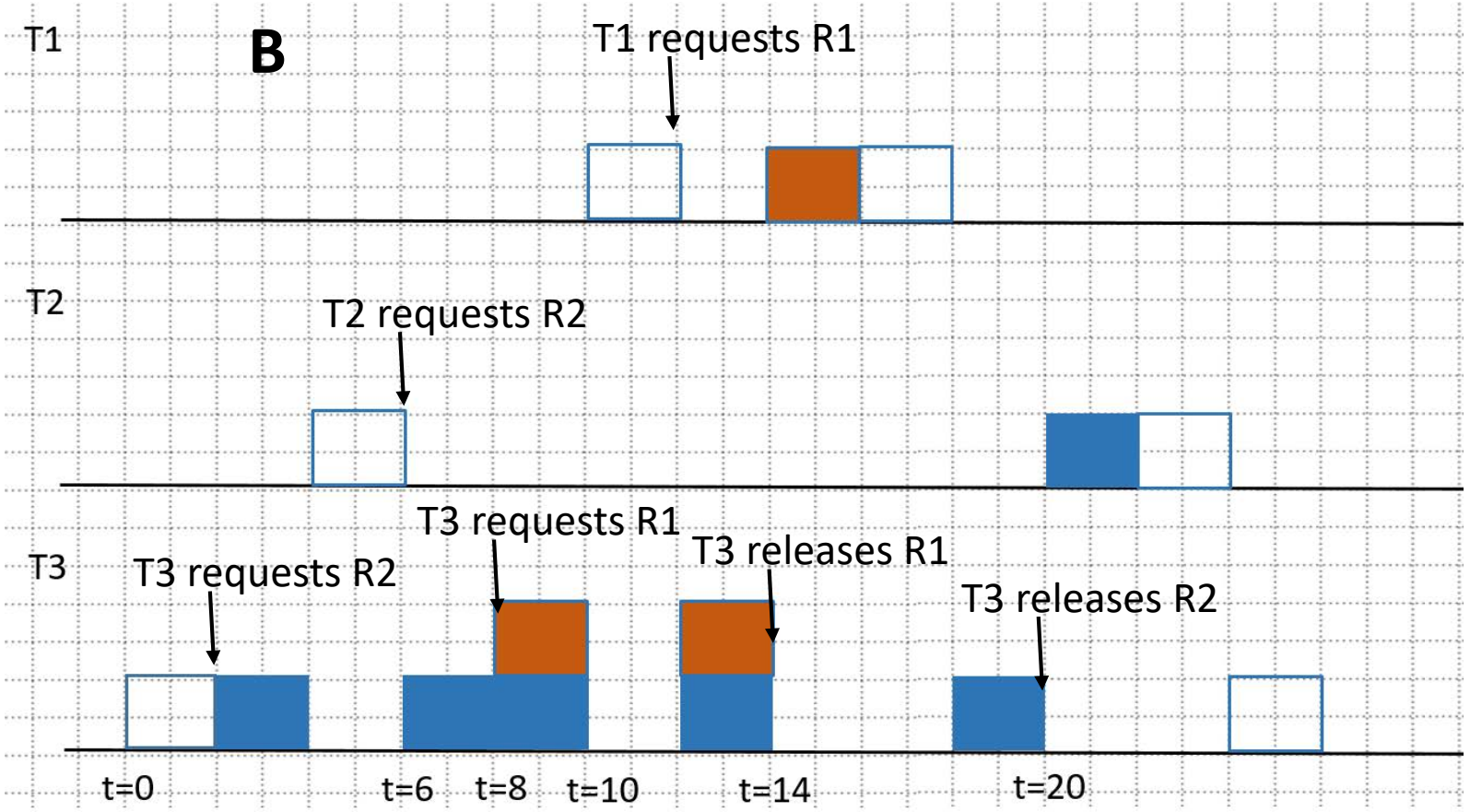


Resources R1 and R2.

R1

R2

Which one of the scheduling output is correct? Assume T1 has priority high, T3 has priority low, and T2 has priority medium.

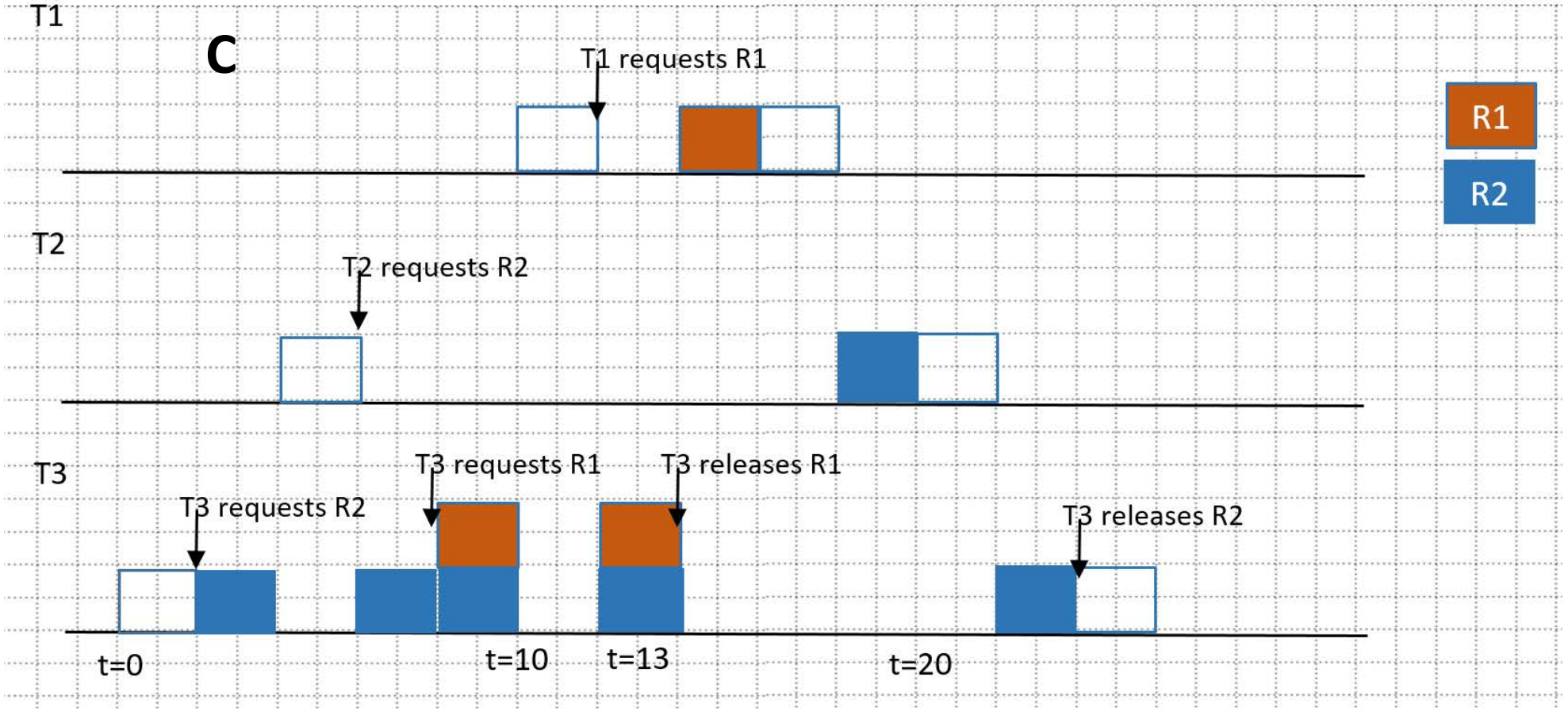


Resources R1 and R2.

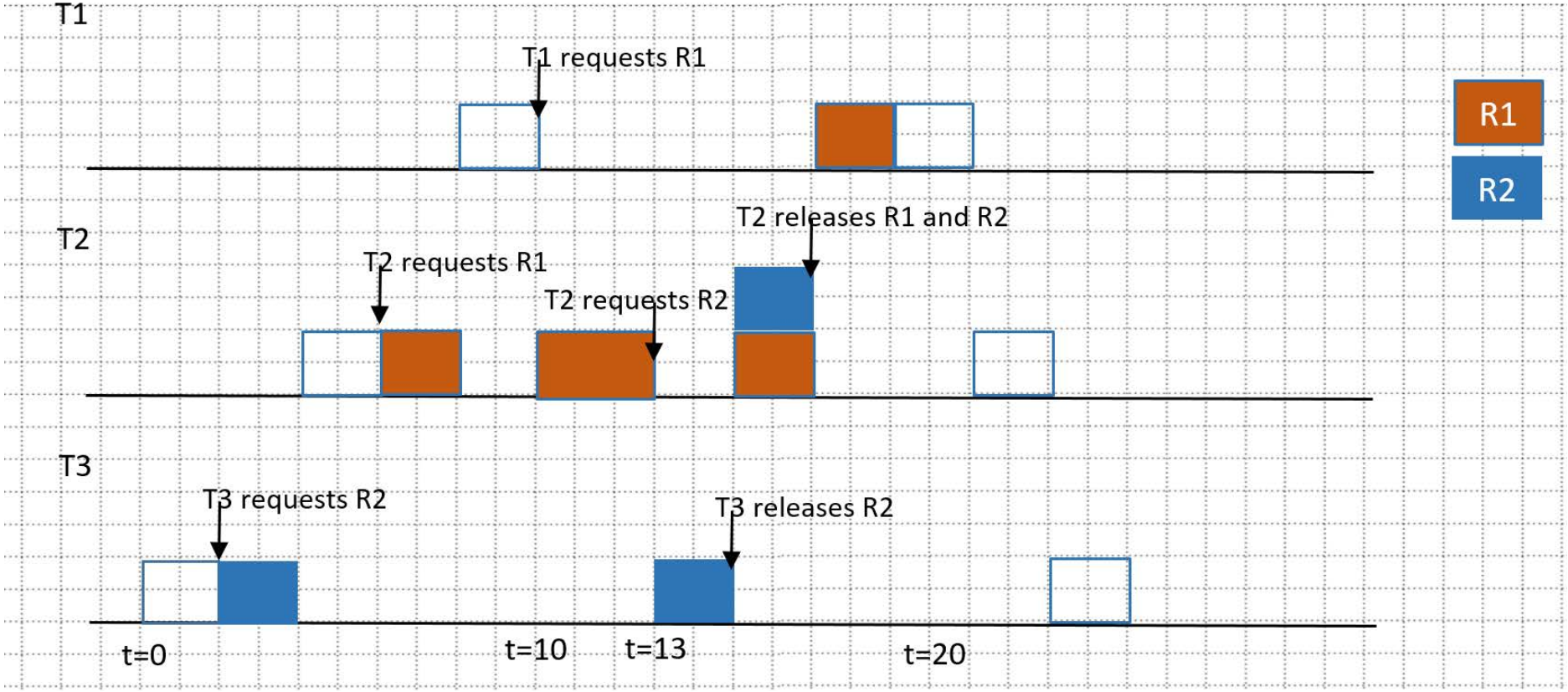
R1

R2

Which one of the scheduling output is correct? Assume T1 has priority high, T3 has priority low, and T2 has priority medium.



Example of Transitive Inheritance



At t=10, T2 inherits T1's priority, so Pr(T2)=H

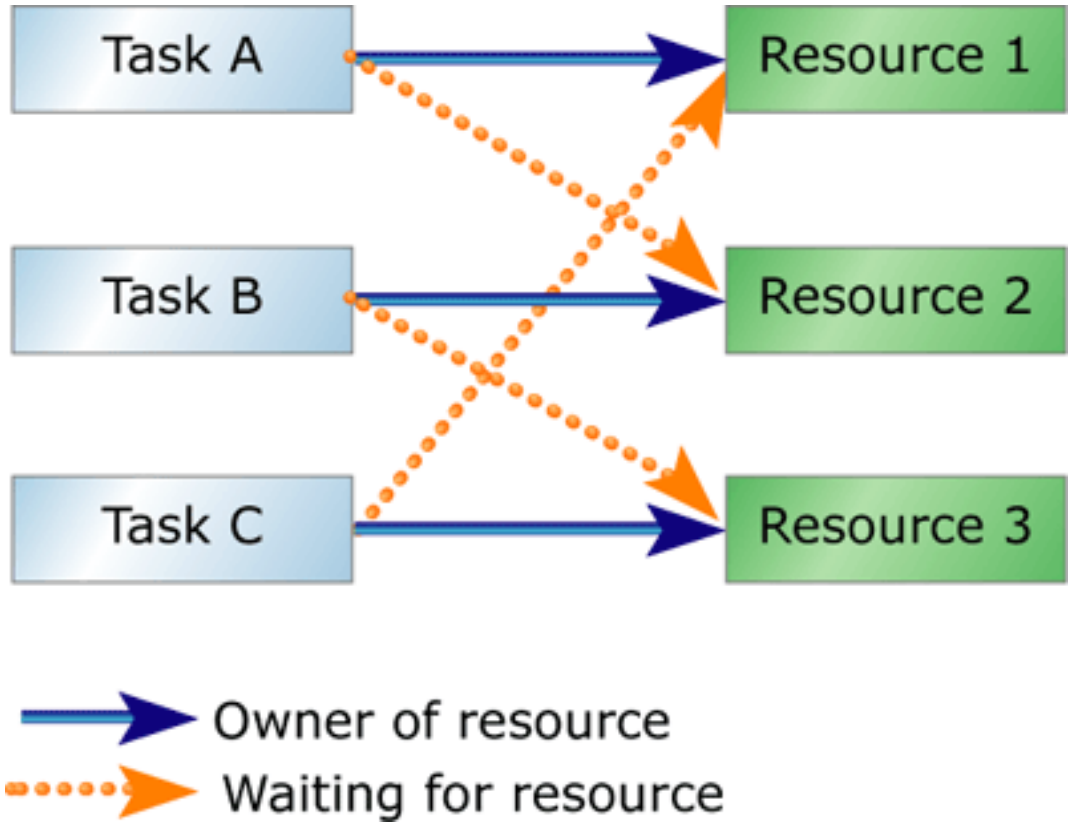
At t=13, T3 inherits T2's priority, so Pr(T3)=H

Comments on PIP

- PIP avoids the disadvantage of NPCS
- PIP has most of the advantages of NPCS.
- However it does not avoid deadlocks.

- Question: Why PIP has the problem of deadlock?

Deadlock Scenario



What will happen in the scenario of deadlock? (Assuming Task A has the highest priority and Task C has the lowest priority)

A requests R2 which is held by B, so B blocks A and inherits A's priority. Similarly, C blocks B and inherits B's priority. A blocks C and inherits C's priority. They will all set to be the highest priority, but still waiting for resources.

Priority Ceiling Protocol (PCP)

- Extend priority inheritance protocol to prevent deadlocks and to further reduce the blocking time.
- Assumptions:
  (1) Assigned priorities of all jobs are fixed
  (2) Resource requirements of all the tasks that will request a resource R is known

- **Priority ceiling of a resource R**:
  - ceiling(R) = highest priority among all the tasks that request R
  - Each resource has the fixed priority ceiling

- **Priority ceiling of a system**:
  - At any given time a set of resources are being used, the highest priority ceiling of this resource set is called the priority ceiling of the system.

Resource Allocation Rule in PCP

- When a task request resource R,
  - If R is held by another task, the request fails and the requesting task is blocked
  - If R is free then:
    1. If the requesting task's priority is higher than the current priority ceiling of the system, R is allocated to it.
    2. If the priority of the requesting task is not higher than the priority ceiling of the system, the request is denied and the task is blocked.

Exception: If the requesting task is holding the resource(s) whose priority ceiling is equal to the priority ceiling of the system, in which case the resource is allocated to the requesting task.

Priority Inheritance Rule in PCP

- When a task T1 gets blocked by T2, T2 inherits the priority of T1 or the ceiling of the allocated resources (both are OK but may result in different schedules). For consistency, in this course, we will make T2 inherits the priority of T1.

- T2 executes at the inherited priority until it releases every resource whose priority ceiling is equal to or higher than the inherited priority of T2. At this time the priority of T2 returns to the priority that it held when it acquired the resource R.

More Comments About PCP

- Priority ceiling of a resource is fixed since we assume that we know all the resource requests. However, priority ceiling of the system is dynamically changed.

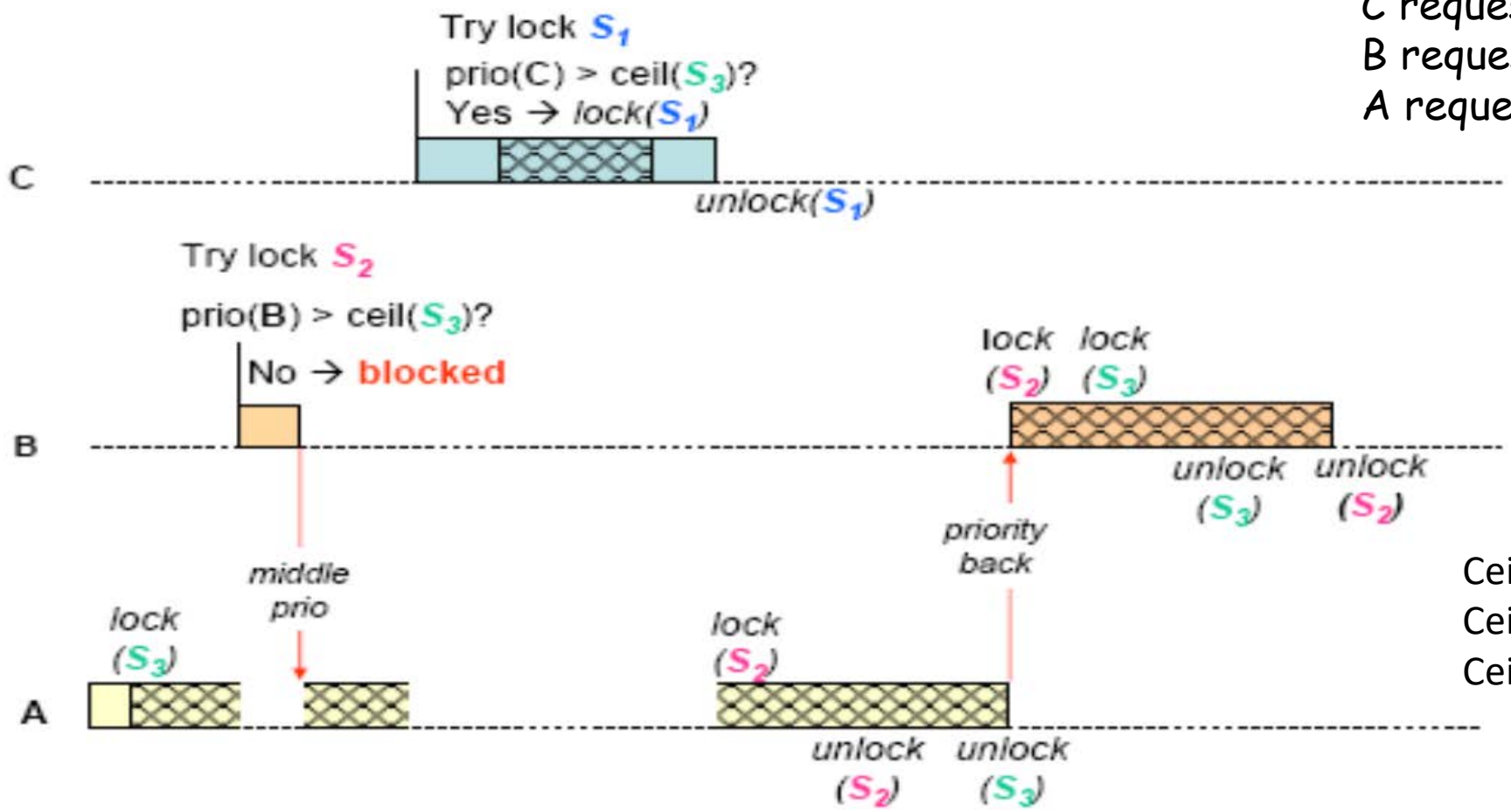
- When the priority of a task is updated?
  - When a task (T1) is blocked by another (T2) , we know that T2 is in its critical section which holds or will request a certain resource required by T1 (note that the contention occurs now or will occur in the future), so T2's priority is updated to the priority of T1.

- Only when a task has higher priority than the system's priority ceiling, the task will acquire a certain resource if it is available.

PCP

Assume C has a high priority (H), A has a low priority (L), and B has a medium priority (M).

C requests S1
B requests S2 and S3
A requests S3 and S2



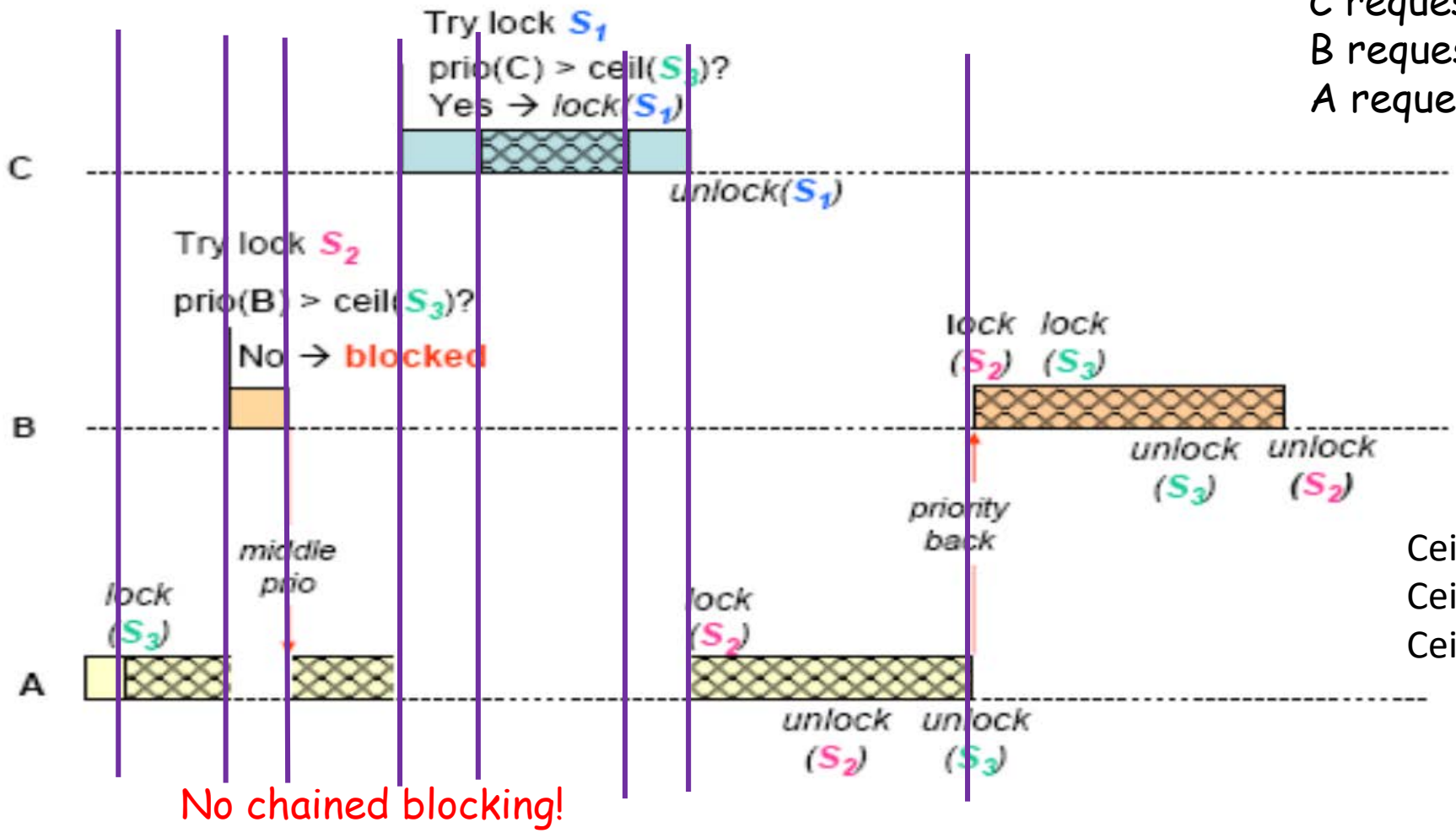
Ceiling(S1) = H (high)
Ceiling(S2) = M (medium)
Ceiling(S3) = M (medium)

No chained blocking!

PCP

Assume C has a high priority (H), A has a low priority (L), and B has a medium priority (M).

C requests S1
B requests S2 and S3
A requests S3 and S2



No chained blocking!

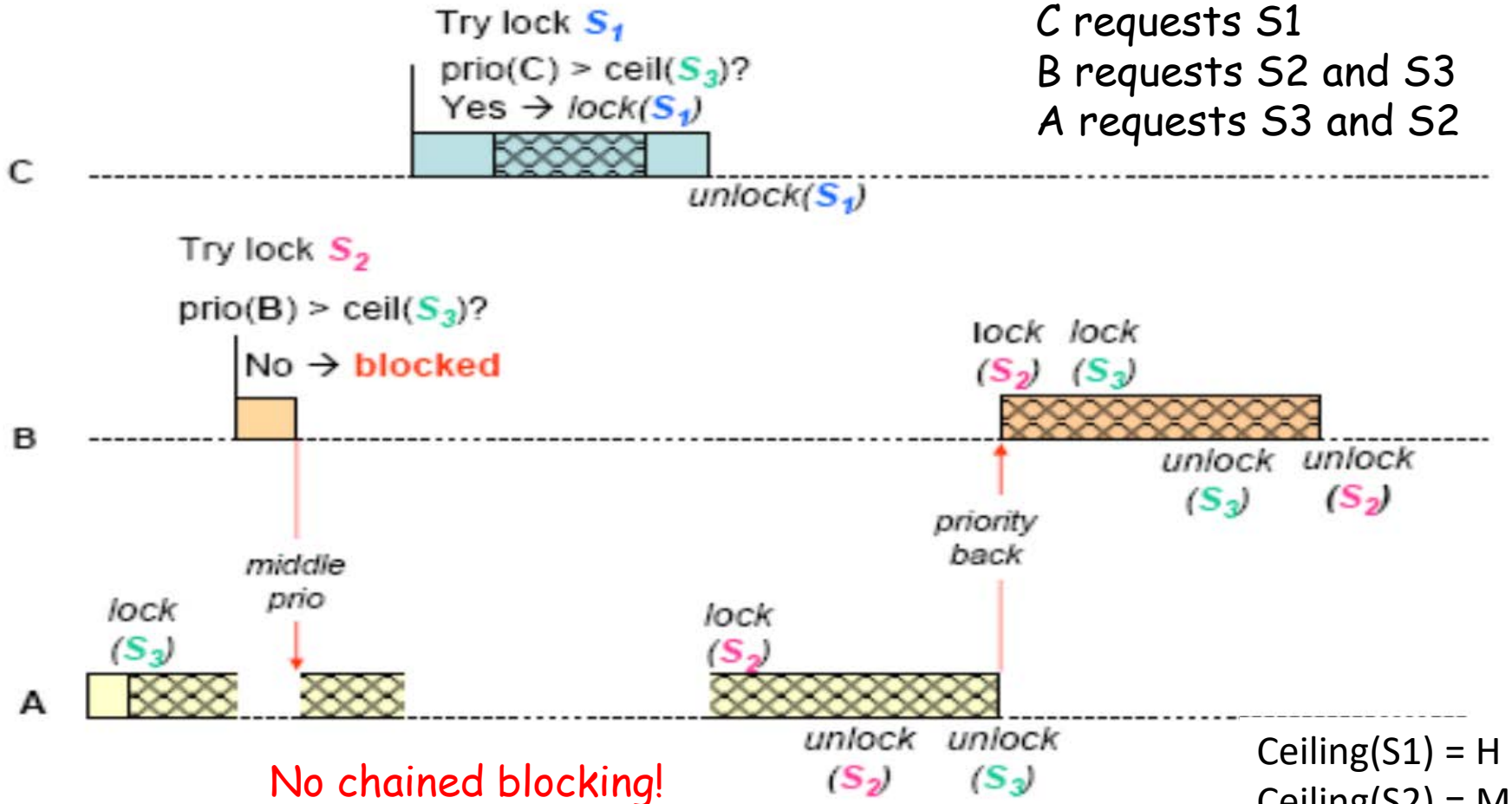
Ceiling(S1) = H (high)
Ceiling(S2) = M (medium)
Ceiling(S3) = M (medium)

Facts about Priority Ceiling

- A task can acquire a resource only if
  - the resource is free, AND
  - it has a higher priority than the priority ceiling of the system, or when the requesting task is holding the resource(s) whose priority ceiling is equal to the priority ceiling of the system.
- A task can be blocked by at most one critical section.
- Higher run-time overhead than Priority Inheritance Protocol

Revisit PCP Example

Assume C has a high priority (H), A has a low priority (L), and B has a medium priority (M).



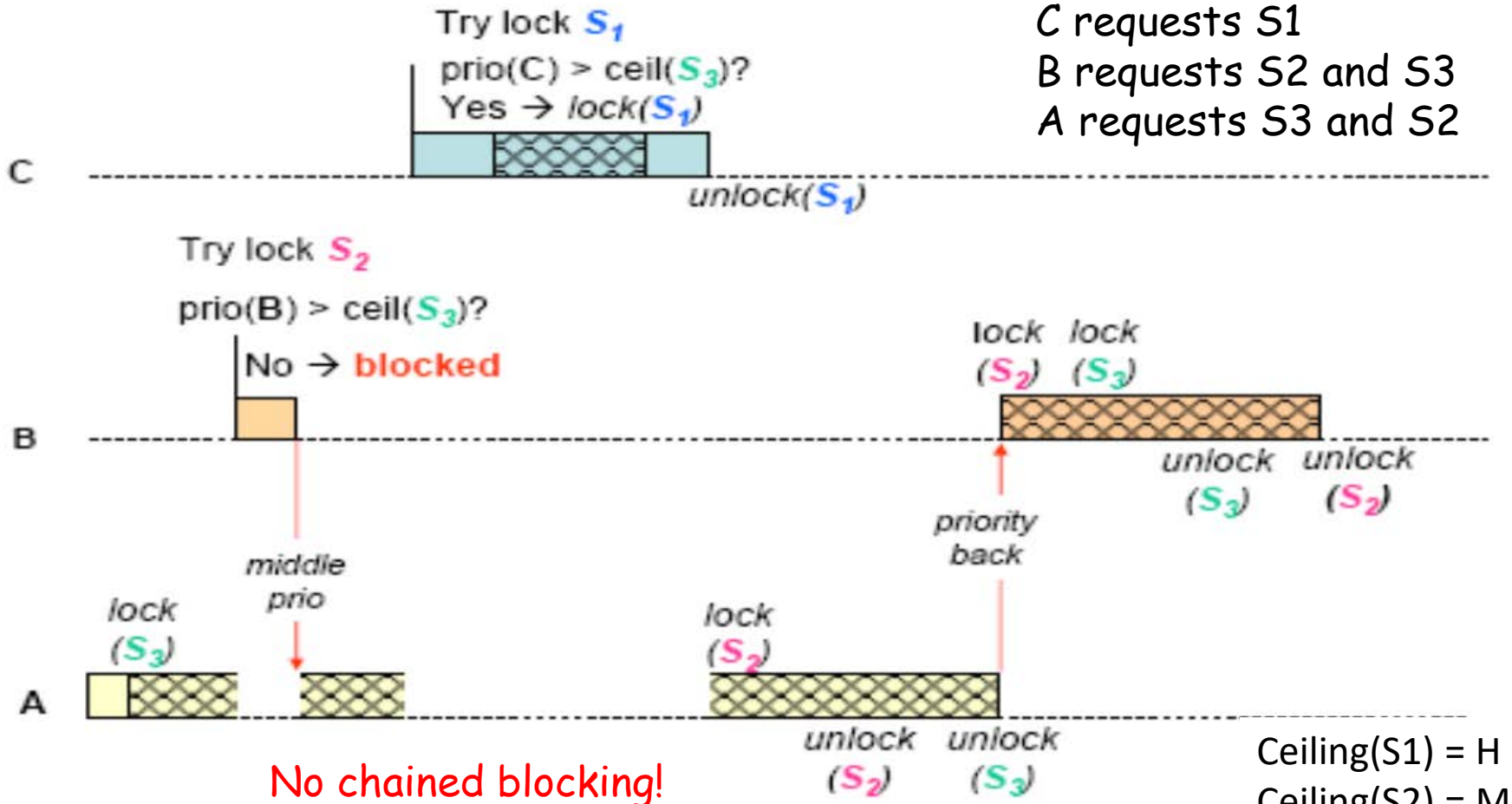
C requests S1
B requests S2 and S3
A requests S3 and S2

Q1:
At which time point, priority inheritance or priority update occurs?
How the priority of a task is computed?

Ceiling(S1) = H (high)
Ceiling(S2) = M (medium)
Ceiling(S3) = M (medium)

Revisit PCP Example

Assume C has a high priority (H), A has a low priority (L), and B has a medium priority (M).



C requests S1
B requests S2 and S3
A requests S3 and S2

Q2:
At which time point, the priority ceiling of the system changes?

Ceiling(S1) = H (high)
Ceiling(S2) = M (medium)
Ceiling(S3) = M (medium)