Fair Lateness Scheduling:

Reducing Maximum Lateness in G-EDF-like Scheduling

Jeremy P. Erickson
James H. Anderson

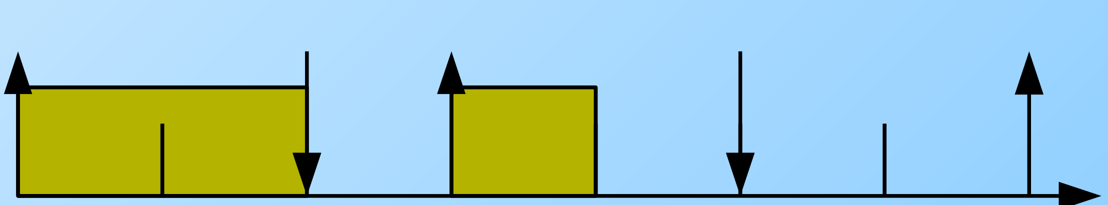
THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Basic Idea

- We will be examining a scheduler that is similar to global earliest-deadline-first (G-EDF).
- Upshot: for soft real-time, we can do better than G-EDF by making some small changes.
- Instead of going into proof details, will provide some intuition.

Background

- System with *m* identical cores/processors.
- Arbitrary-deadline sporadic task model:
  - Worst-Case Execution Time C_i
  - Minimum Separation Time T_i
  - Relative Deadline D_i
  - Utilization $U_i = C_i/T_i$



$C_i = 2$

$T_i = 3$

$D_i = 2$

$U_i = 2/3$

▨ Execution

↑ Release

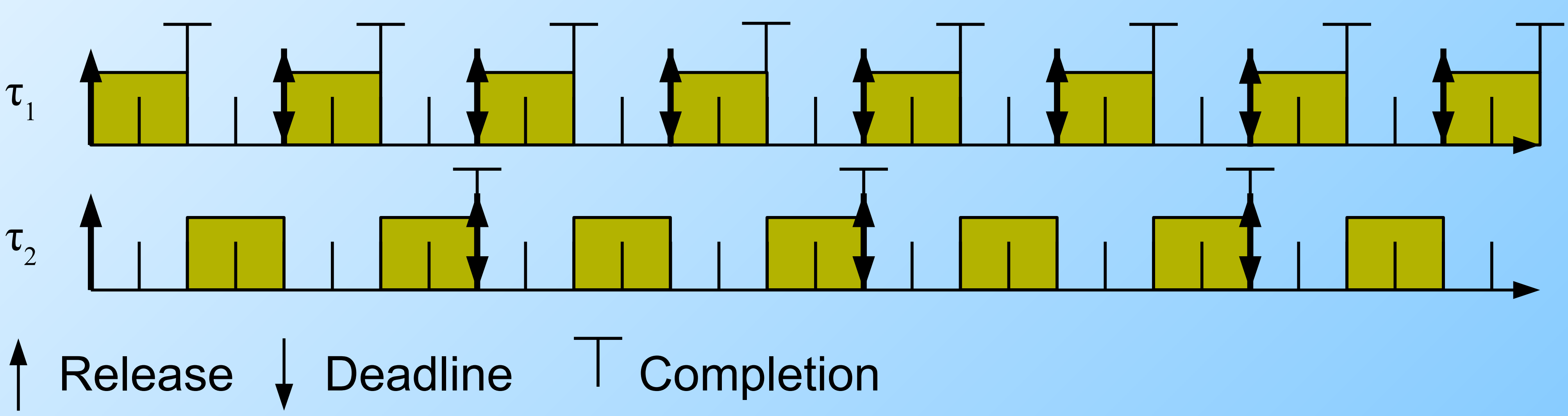
↓ Deadline

Intuition - Uniprocessor Scheduling

- To gain intuition, we'll think about the *implicit deadline* case, where $D_i = T_i$.
- On a uniprocessor, can schedule using earliest-deadline-first as long as $\Sigma U_i \leq 1$.

Intuition – Uniprocessor Scheduling

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 4$, $T_2 = D_2 = 8$, $U_2 = 0.5$
- Observe how schedule works:

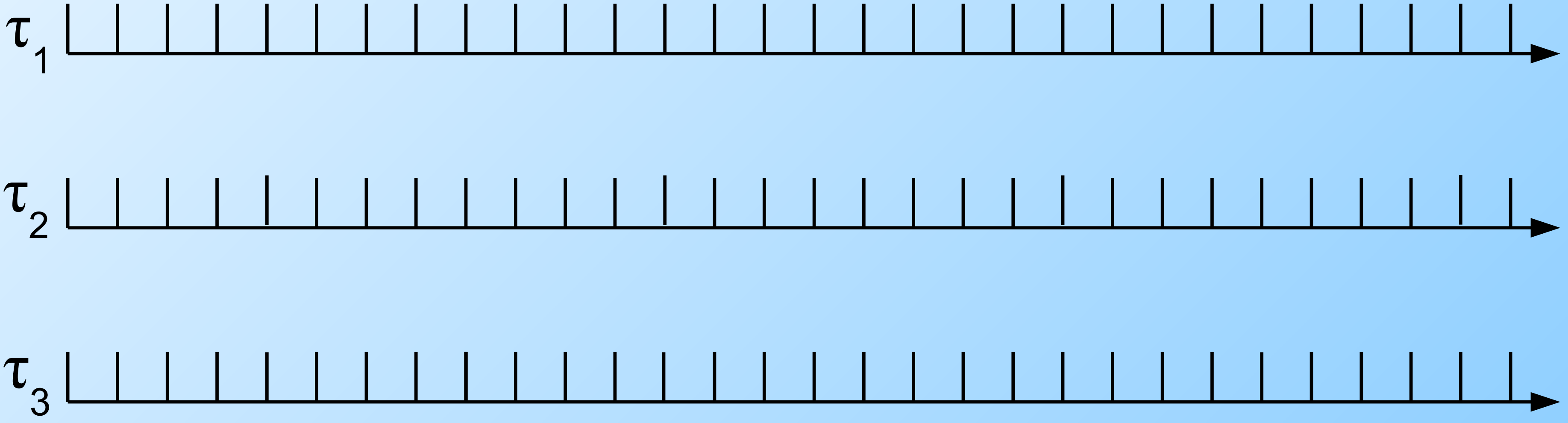


Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



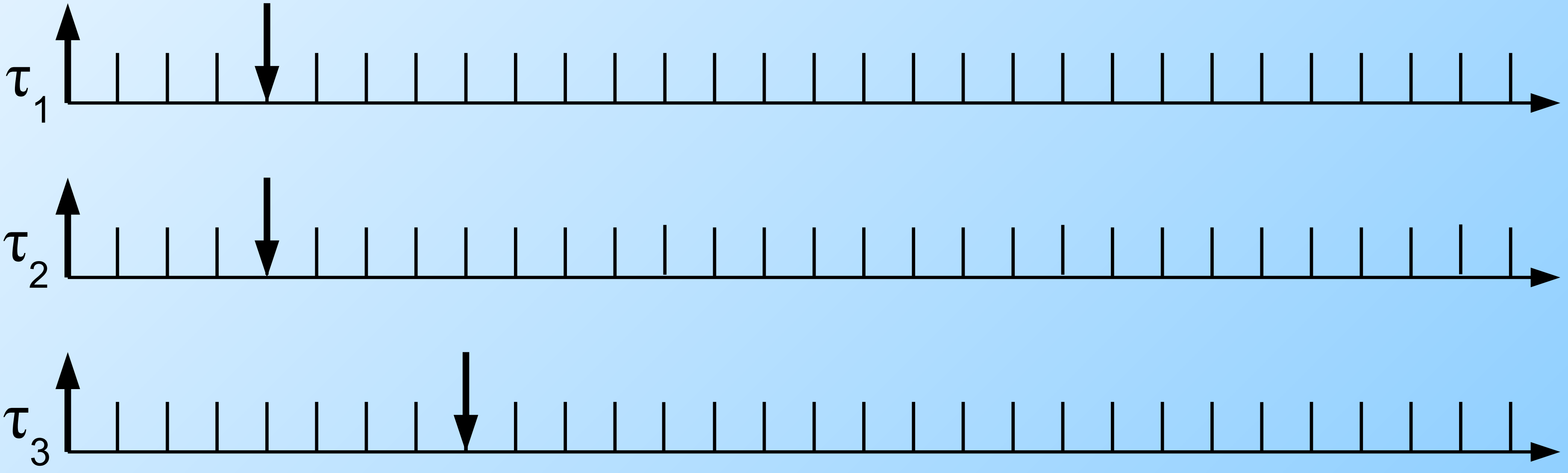
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



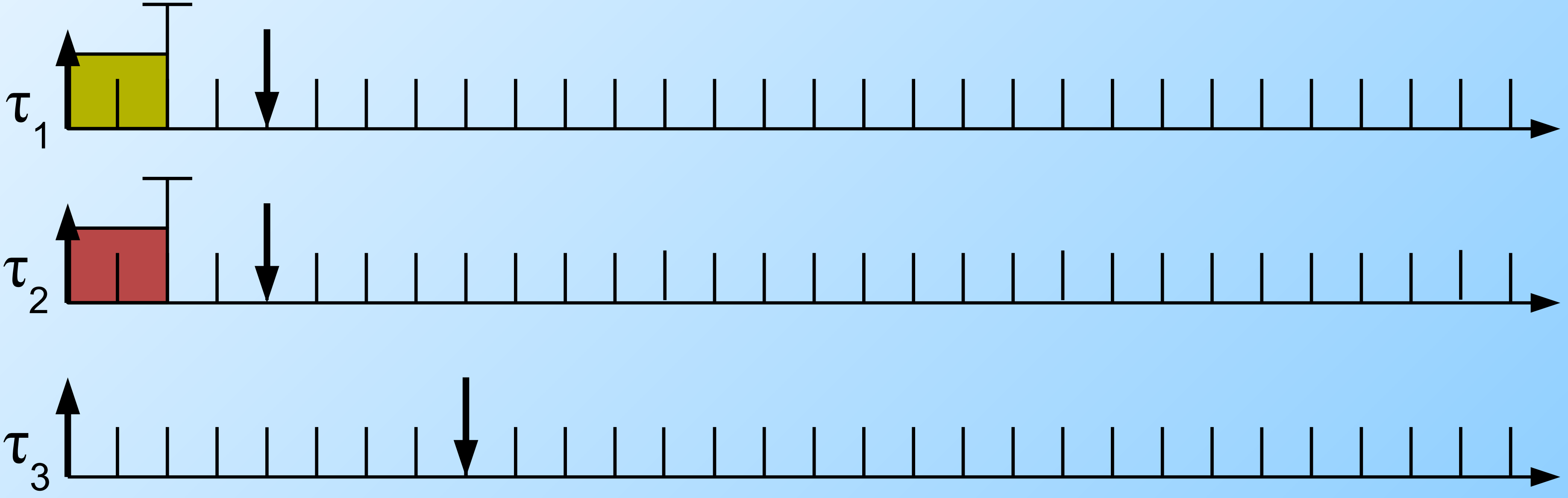
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



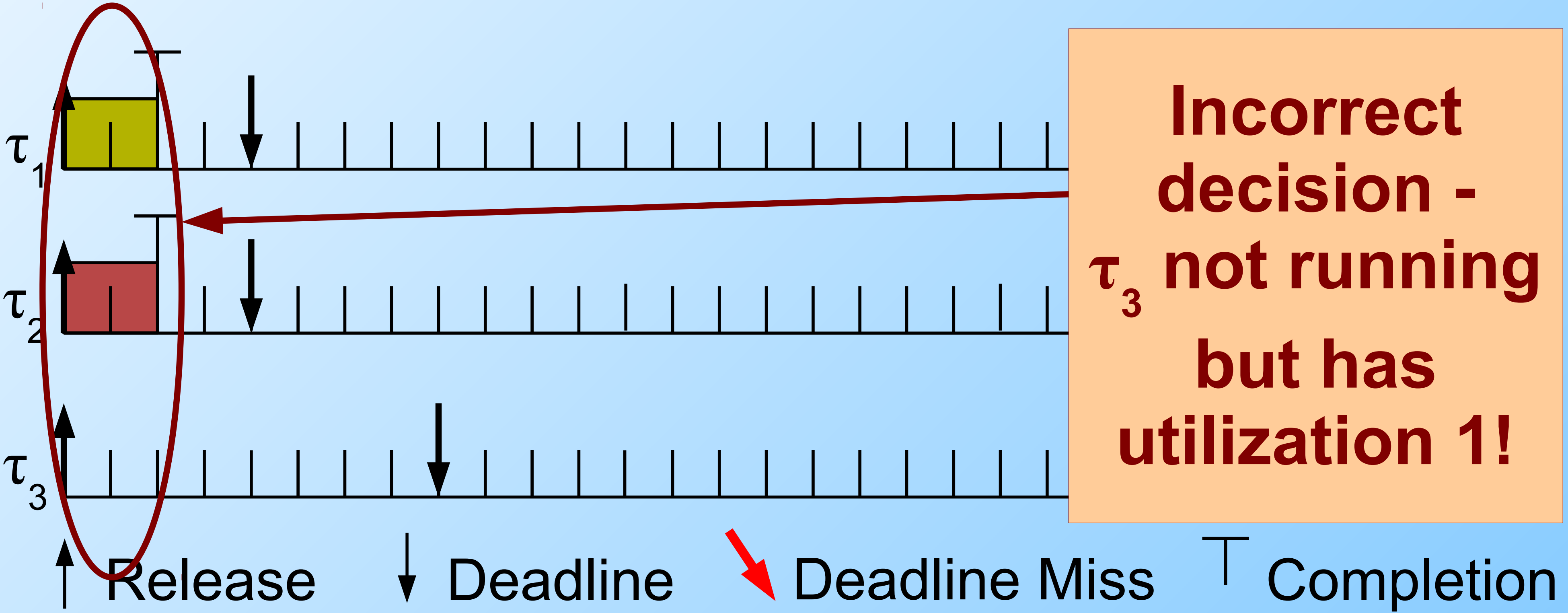
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



Incorrect decision - τ_3 not running but has utilization 1!

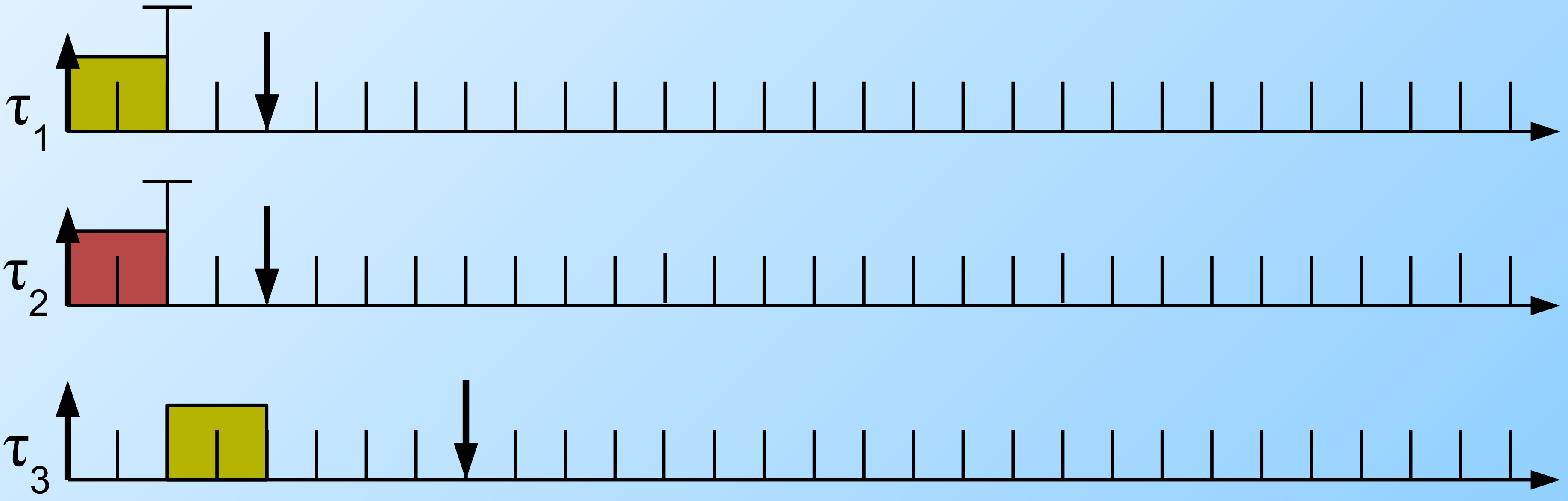
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

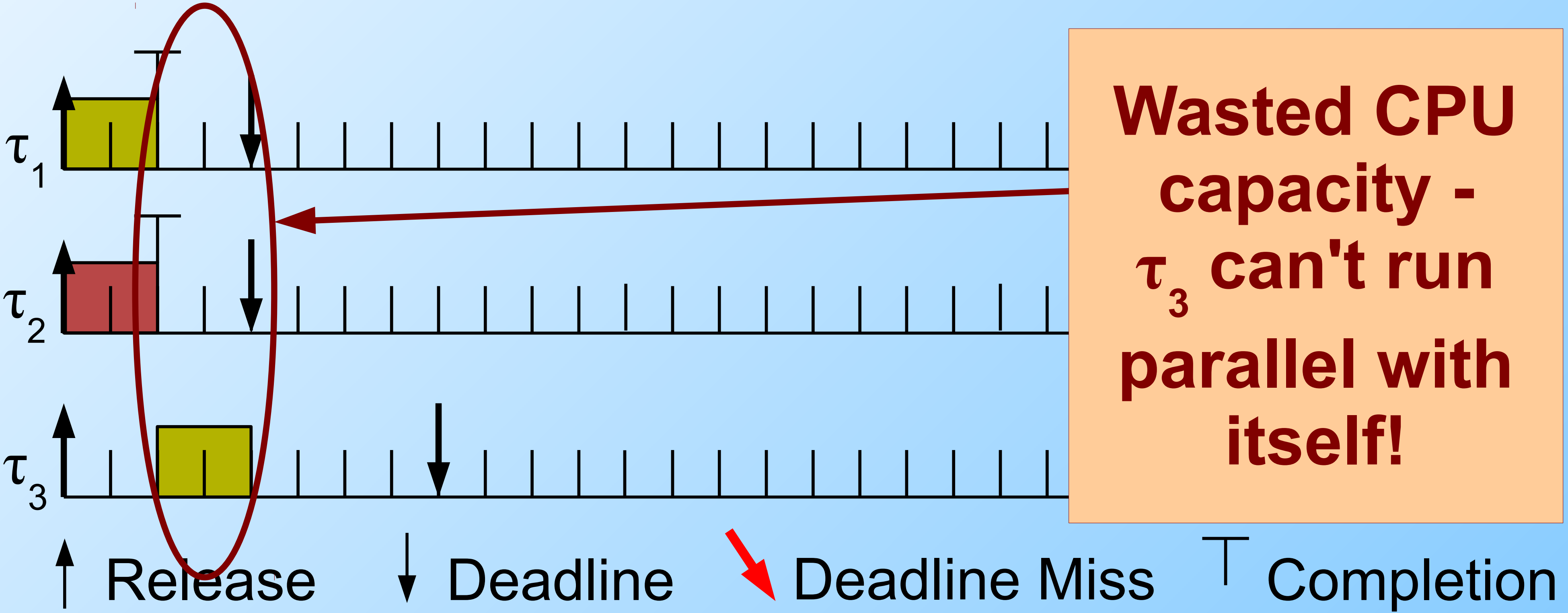
Intuition - EDF on Multiprocessors

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



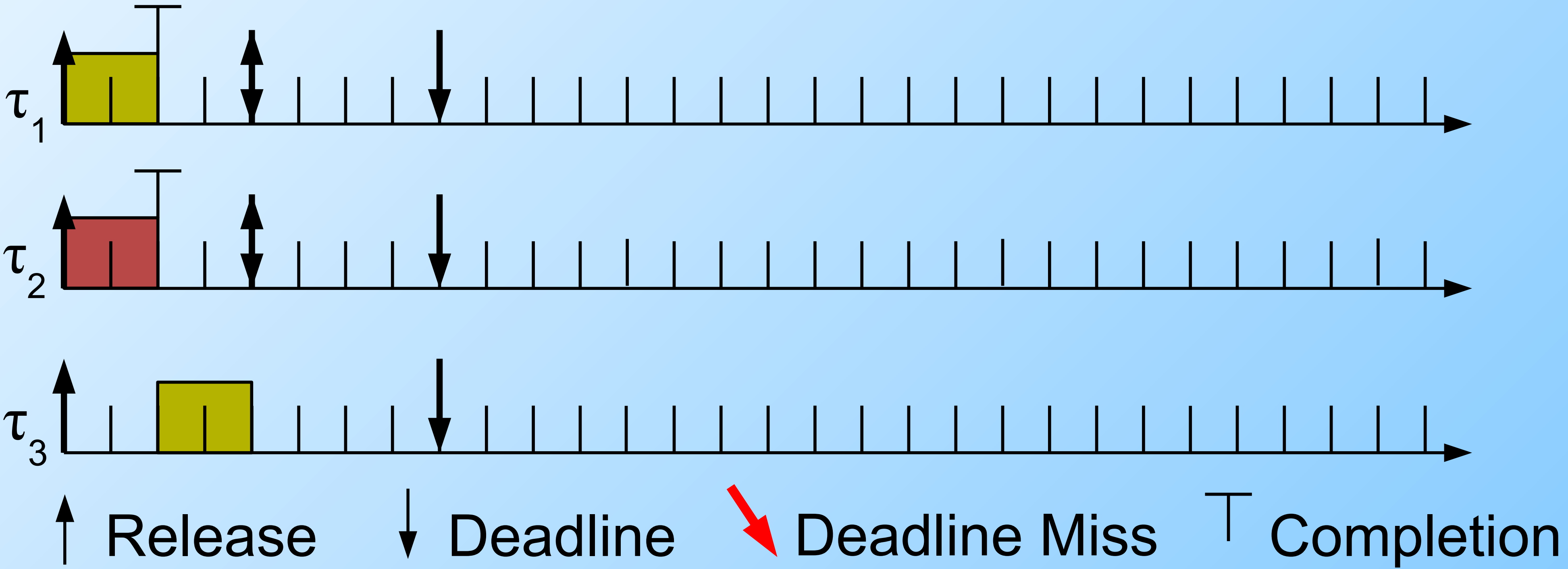
Wasted CPU capacity - τ_3 can't run parallel with itself!

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

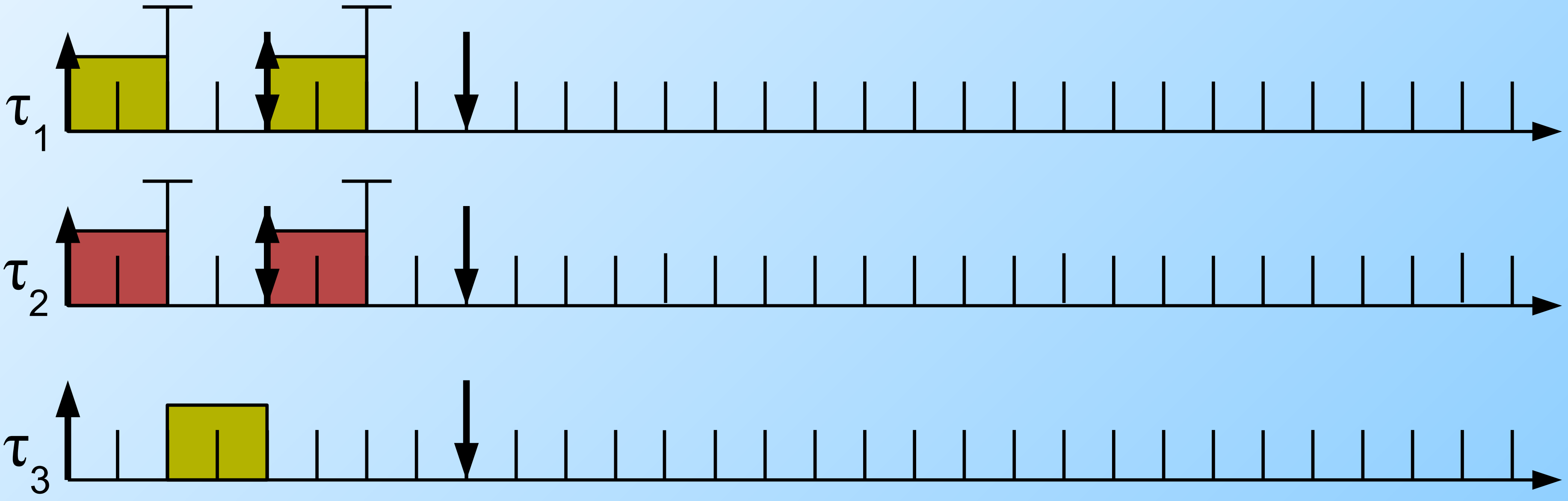
🟥 CPU 2



Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1
🟥 CPU 2

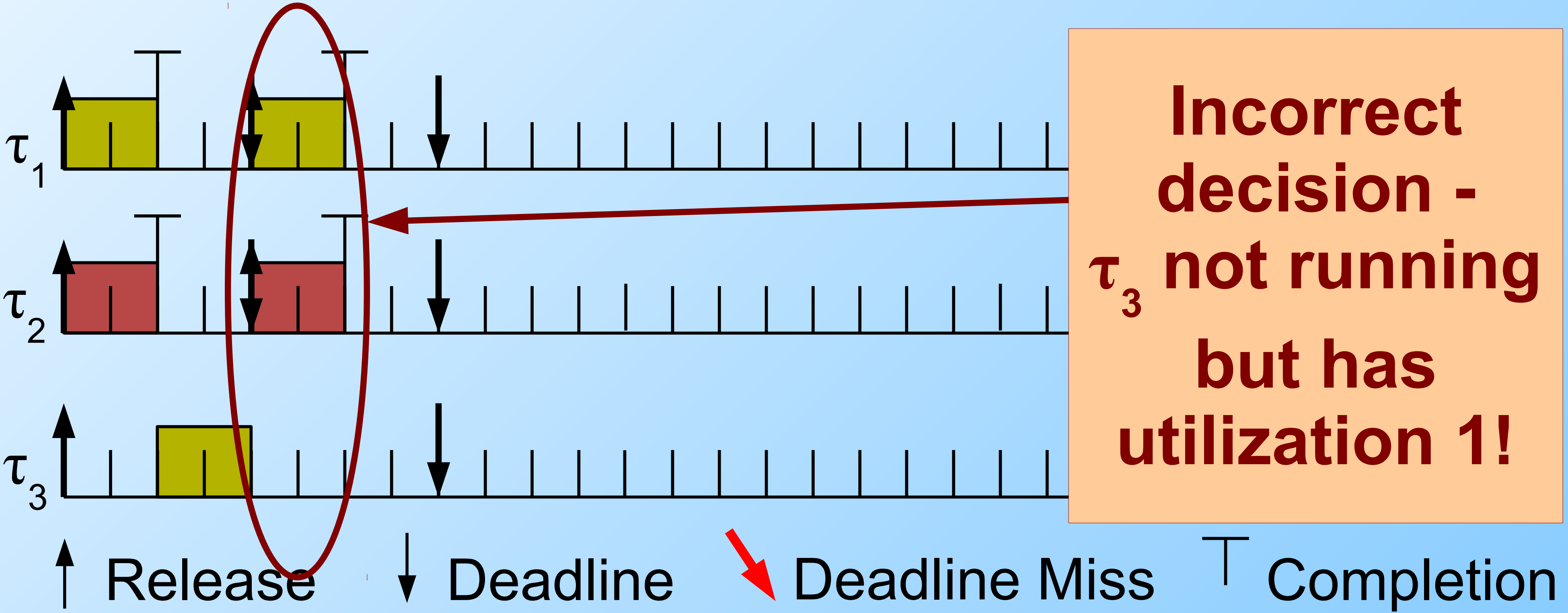


↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$



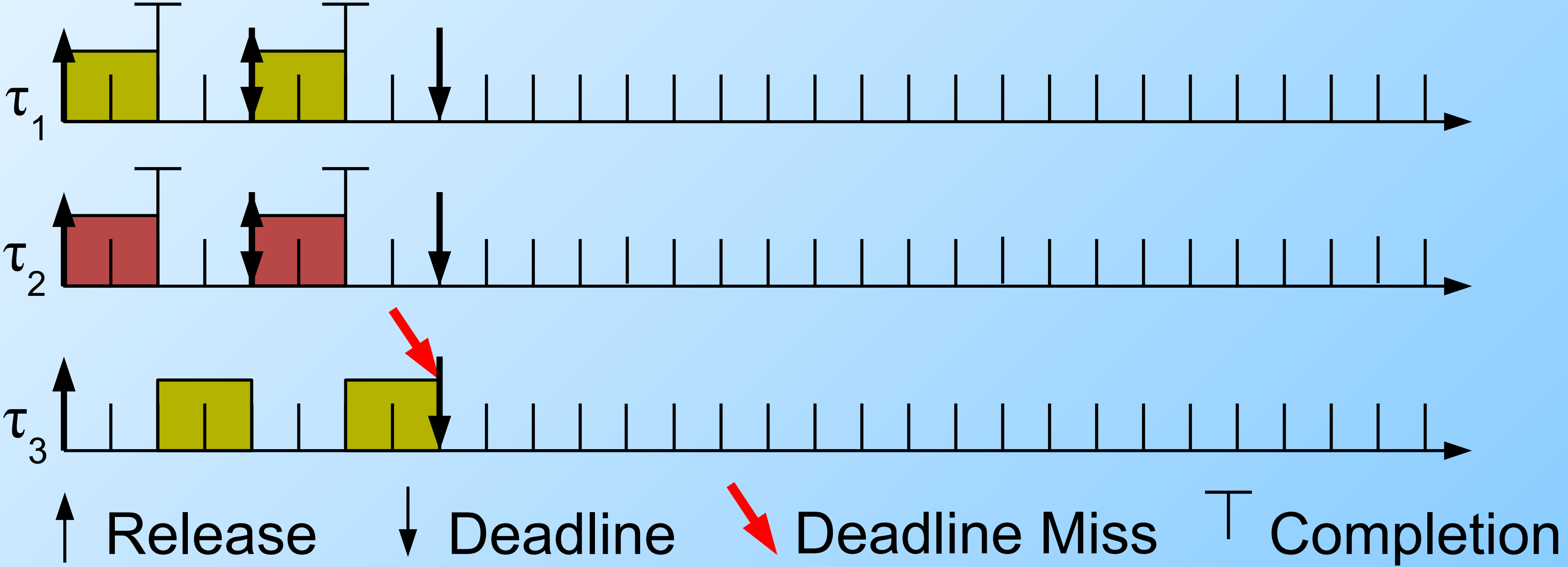
Incorrect decision - τ_3 not running but has utilization 1!

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2

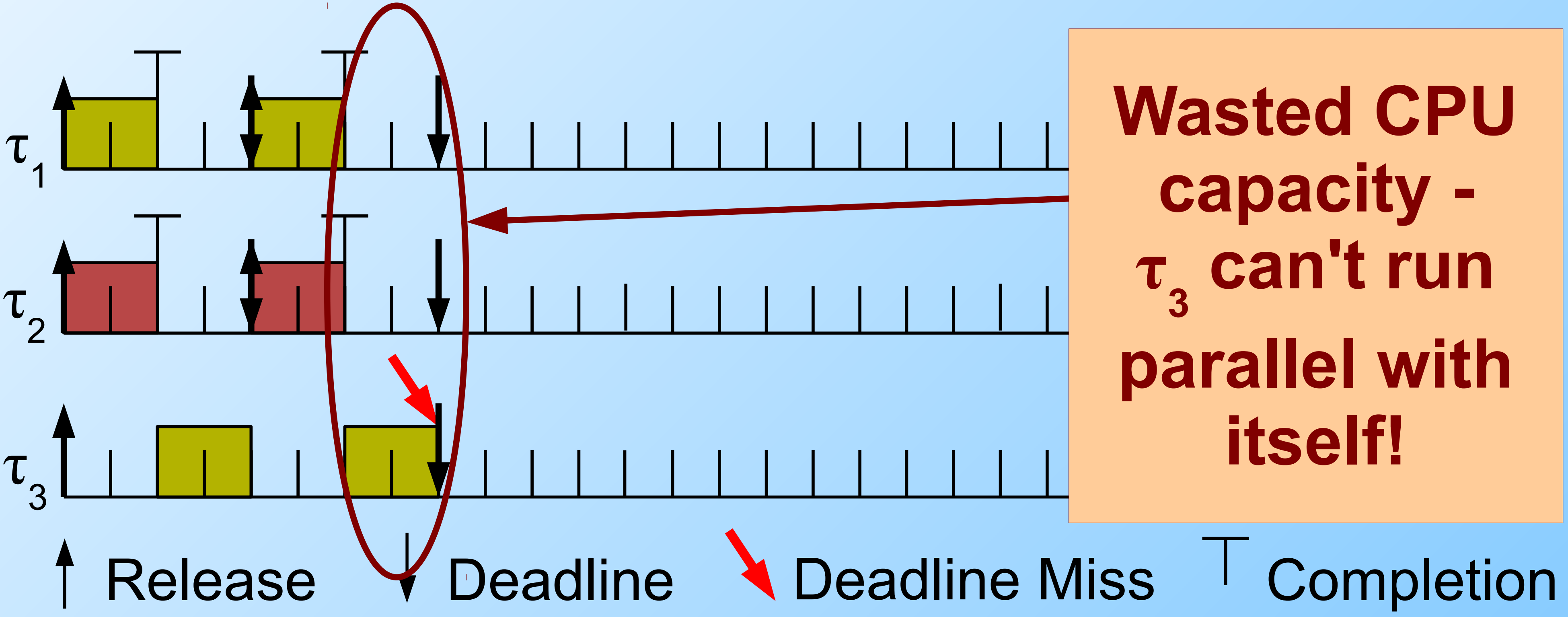


Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



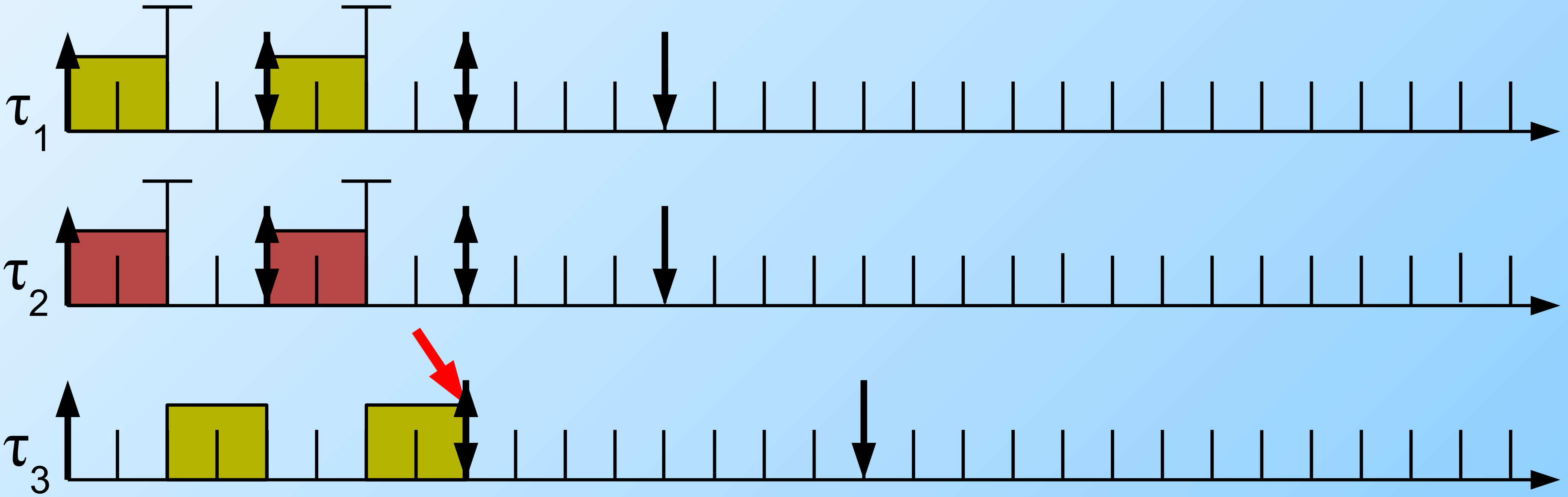
Wasted CPU capacity - τ_3 can't run parallel with itself!

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



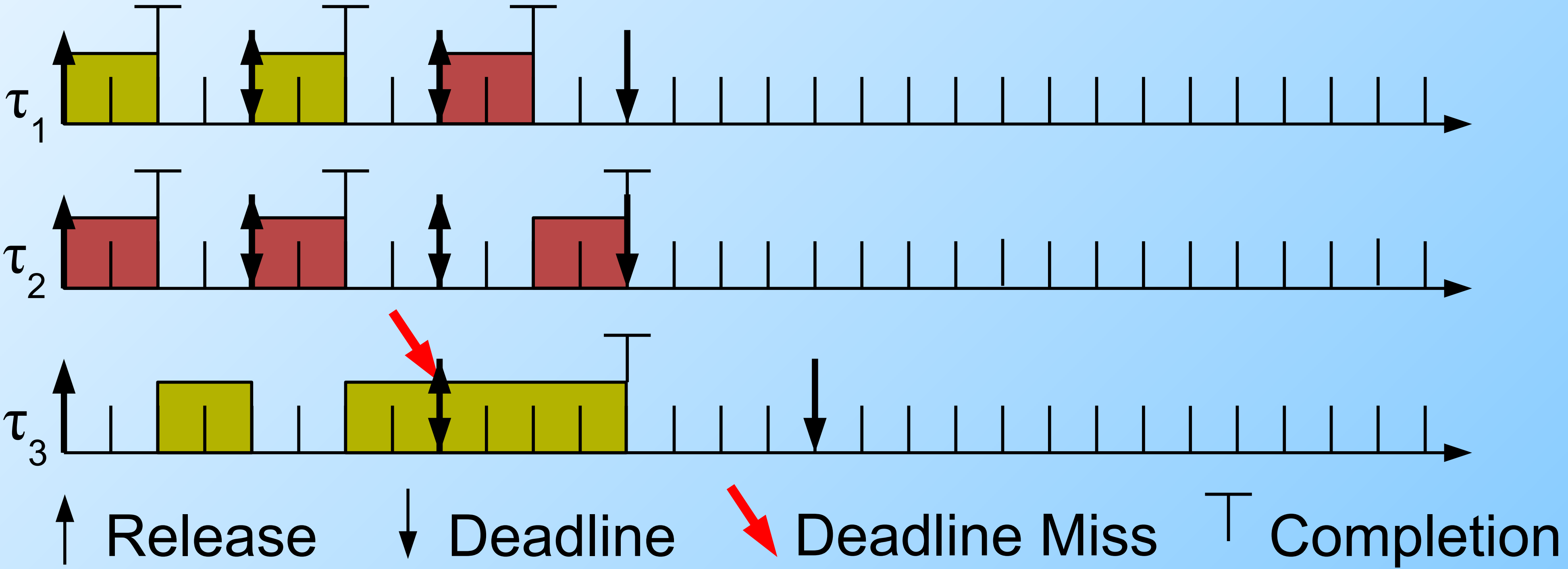
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2

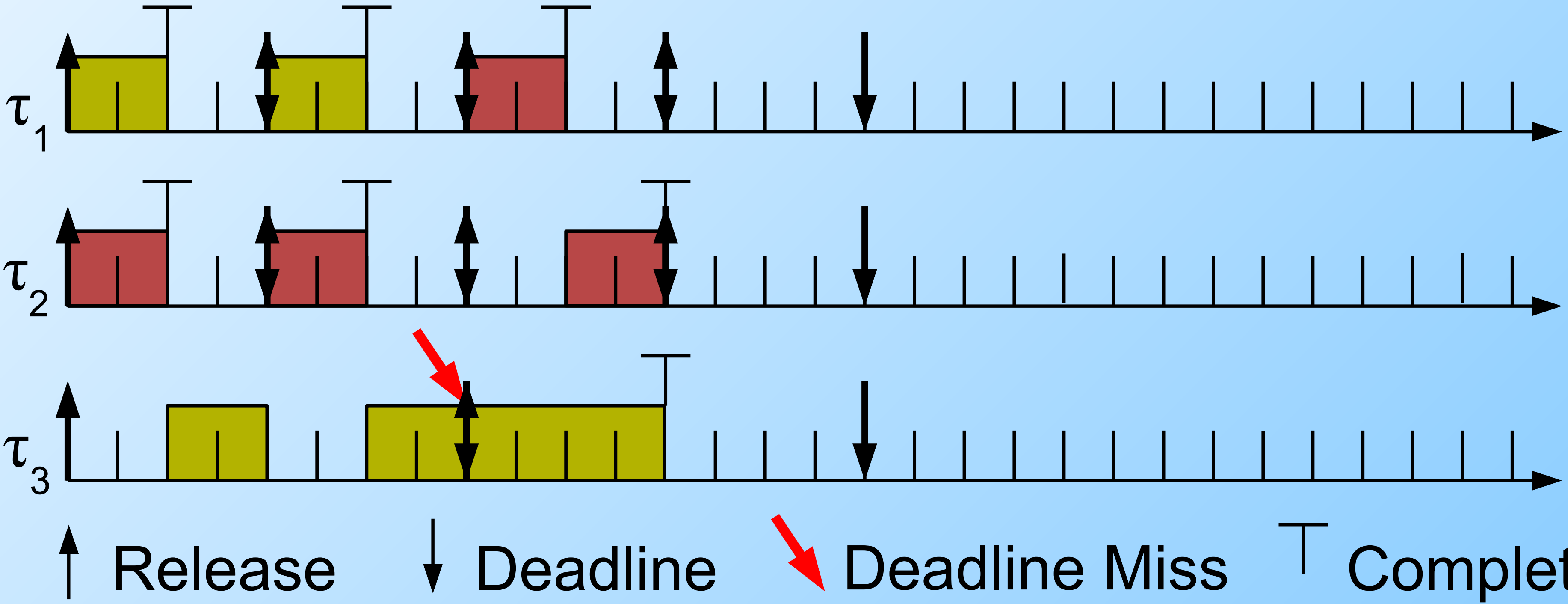


Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

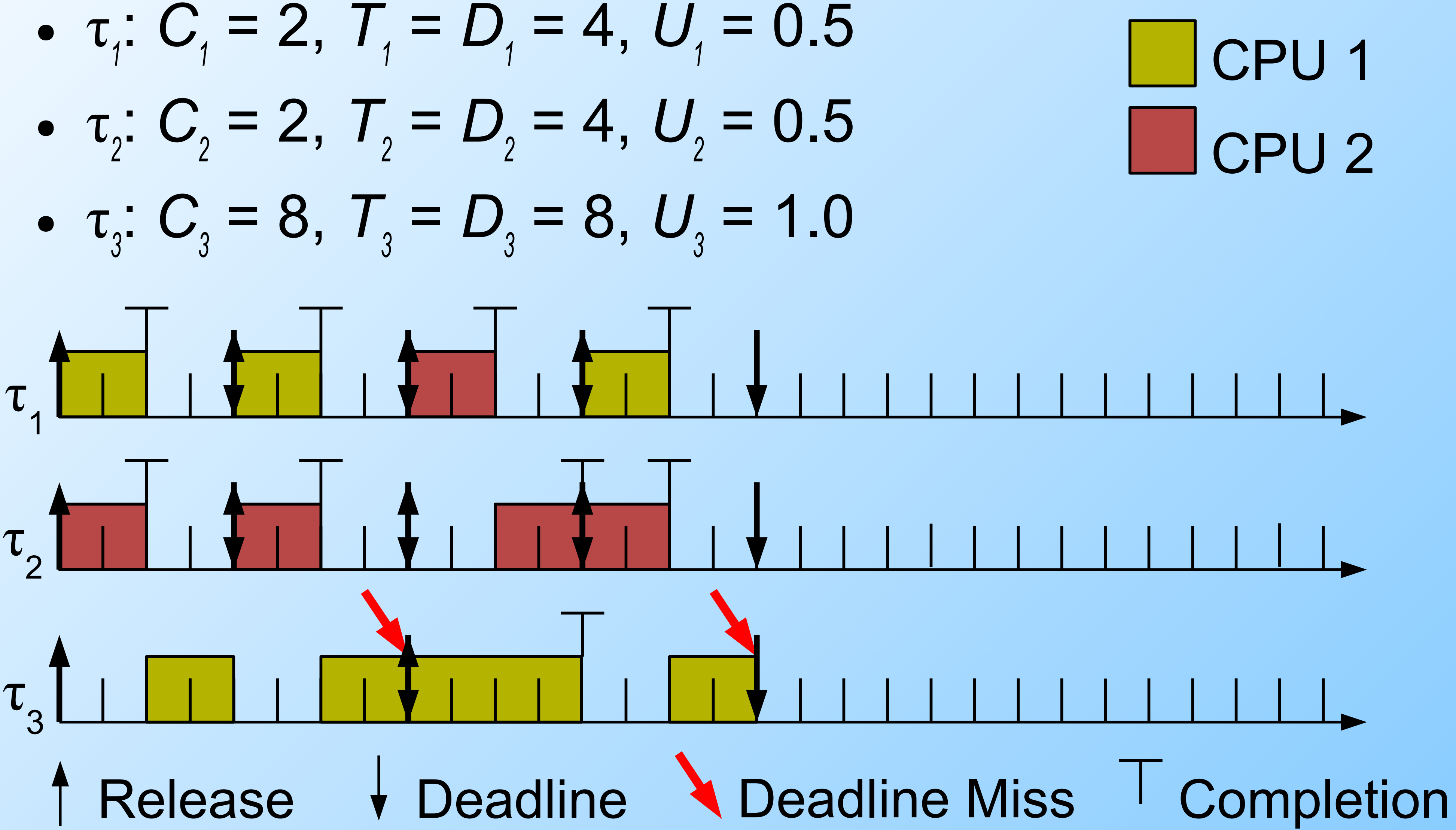
🟥 CPU 2



Intuition - EDF on Multiprocessors

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

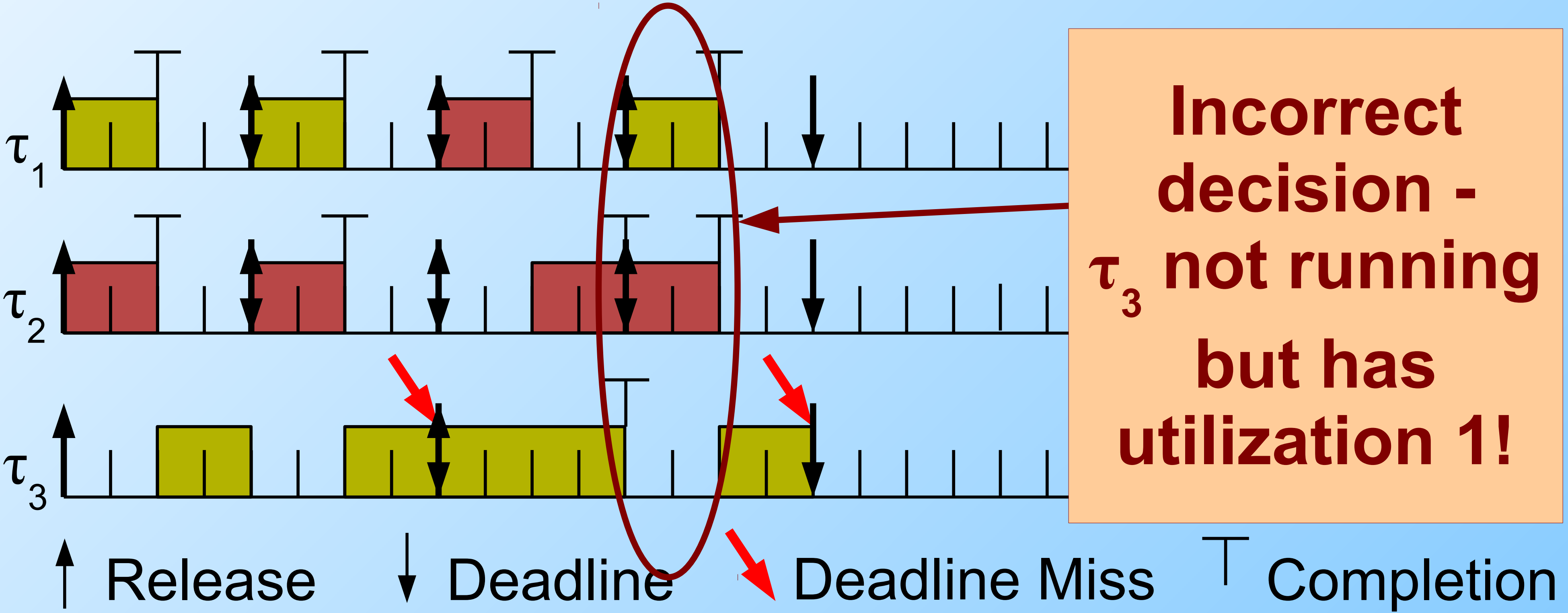


Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟩 CPU 1

🟥 CPU 2

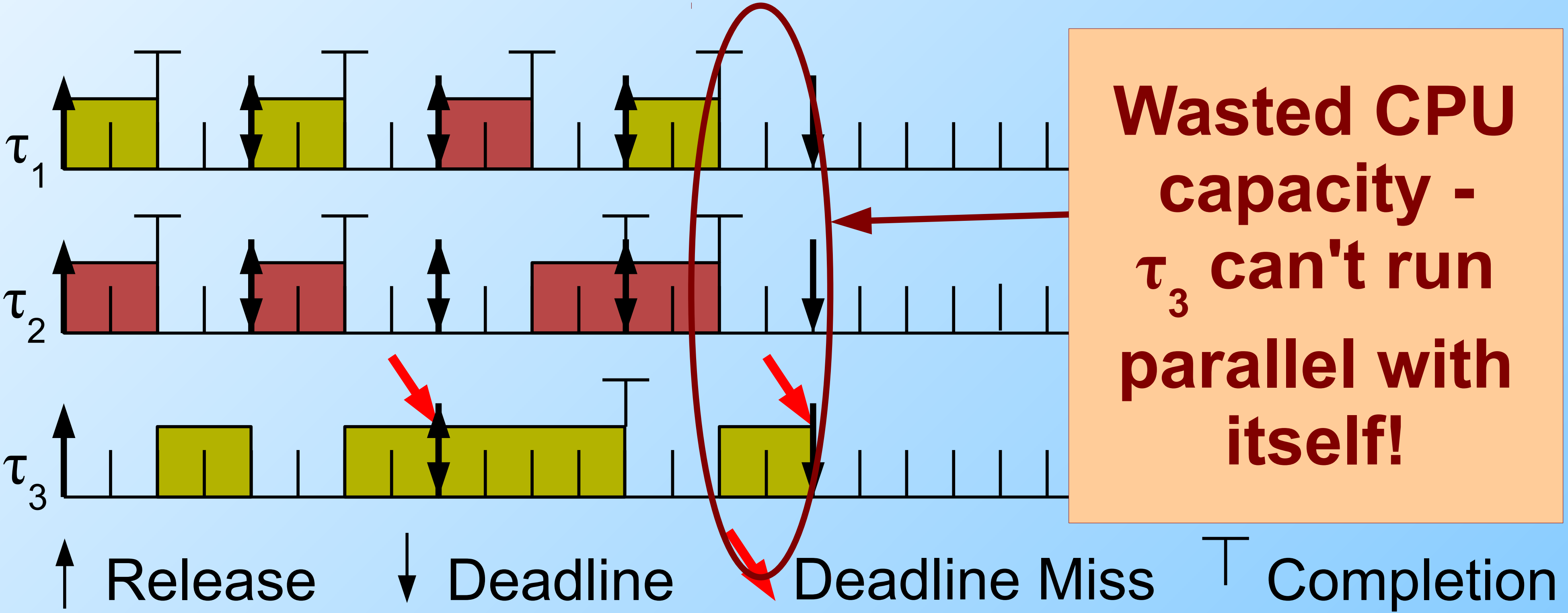


Incorrect decision - τ_3 not running but has utilization 1!

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1
🟥 CPU 2



Wasted CPU capacity - τ_3 can't run parallel with itself!

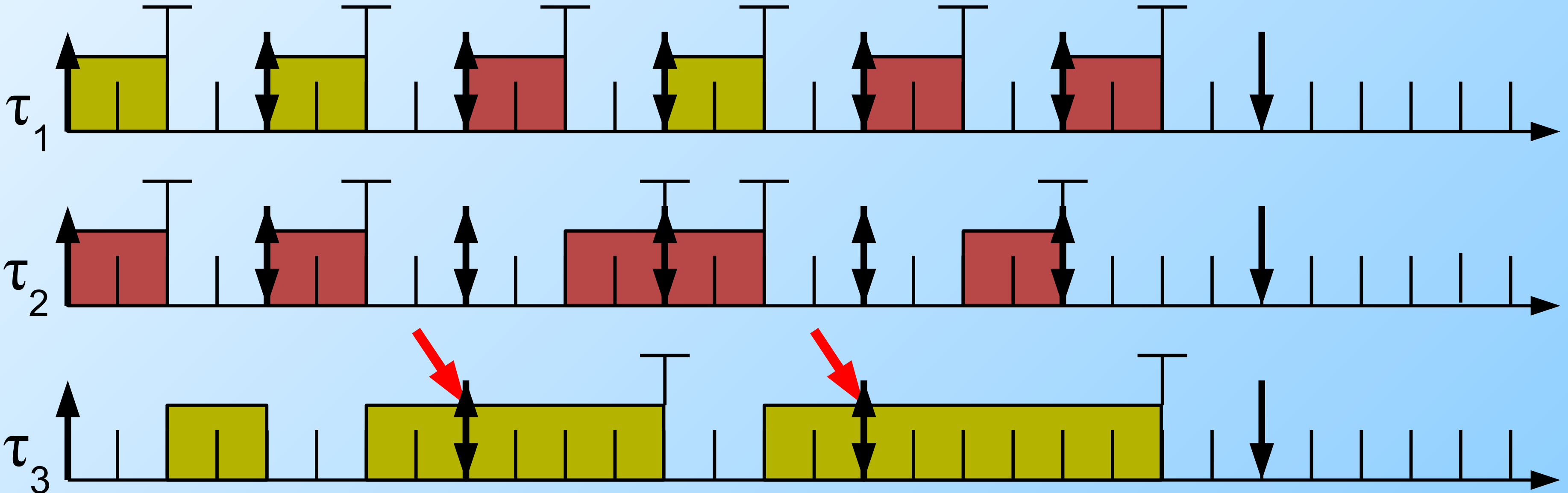
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



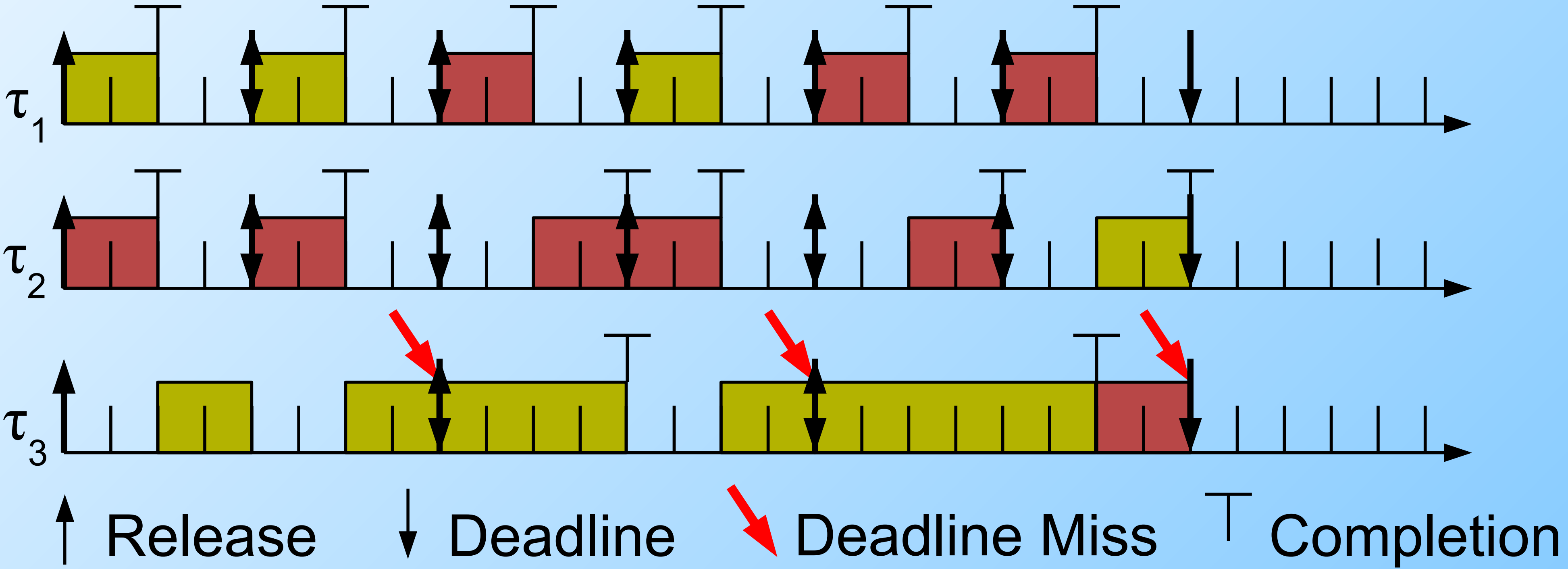
↑ Release ↓ Deadline ↘ Deadline Miss ⊤ Completion

Intuition - EDF on Multiprocessors

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

🟥 CPU 2



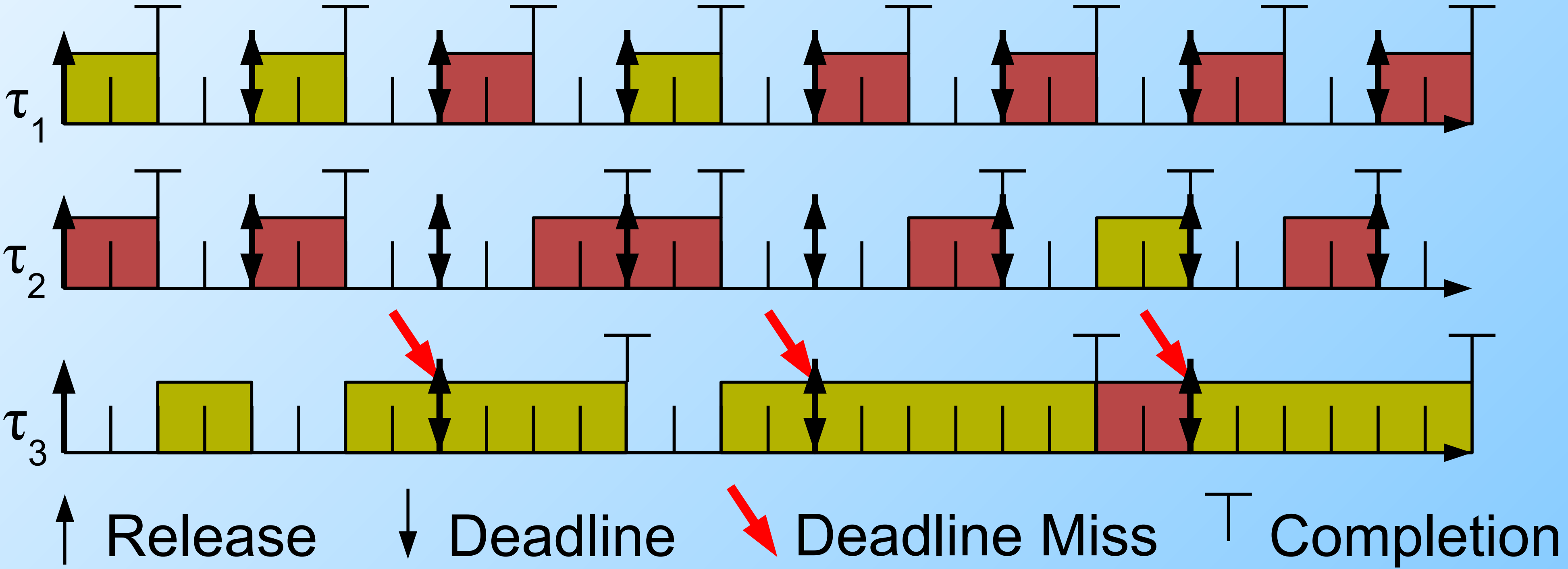
Intuition - EDF on Multiprocessors

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- τ_1: $C_1 = 2$, $T_1 = D_1 = 4$, $U_1 = 0.5$
- τ_2: $C_2 = 2$, $T_2 = D_2 = 4$, $U_2 = 0.5$
- τ_3: $C_3 = 8$, $T_3 = D_3 = 8$, $U_3 = 1.0$

🟨 CPU 1

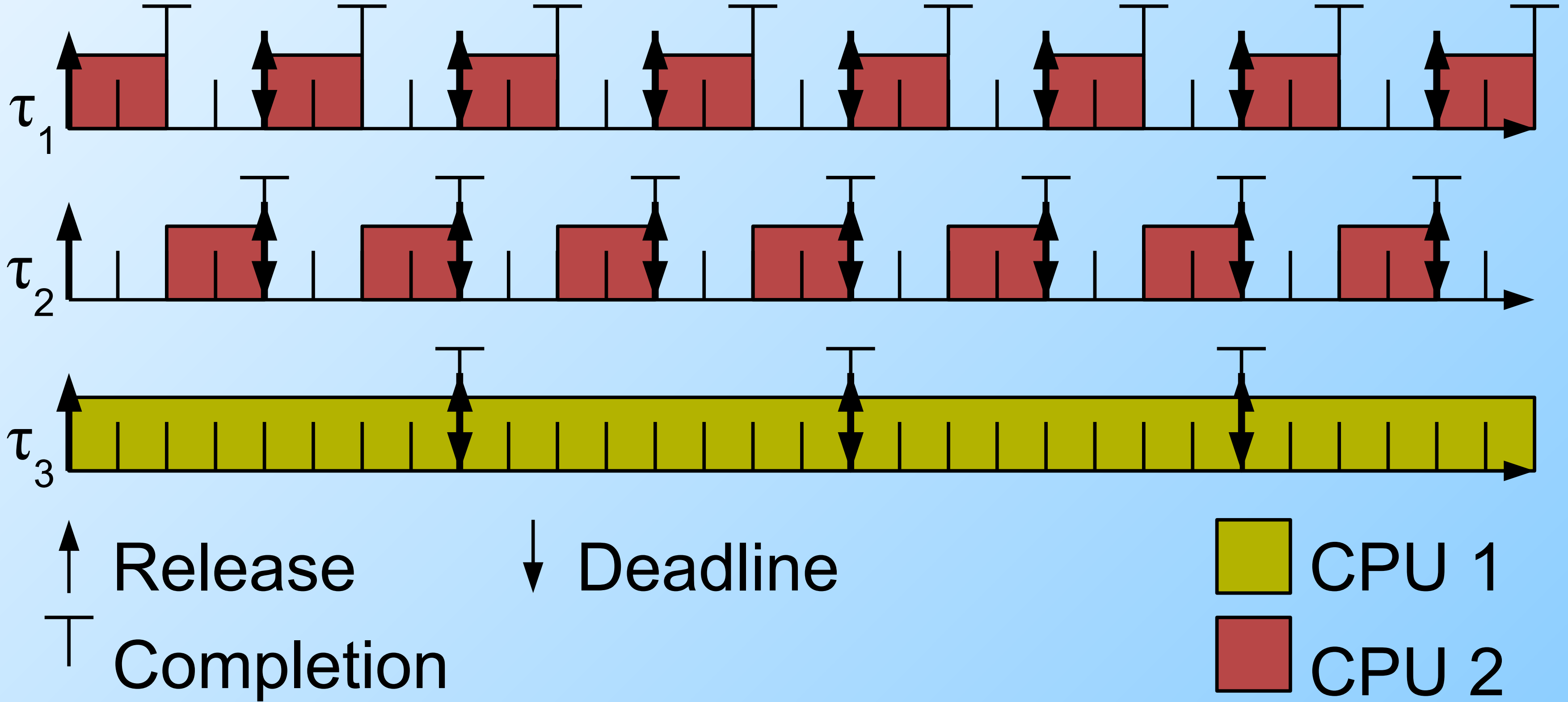
🟥 CPU 2



Other Multiprocessor Schedulers

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

- EDZL
- Optimal Schedulers

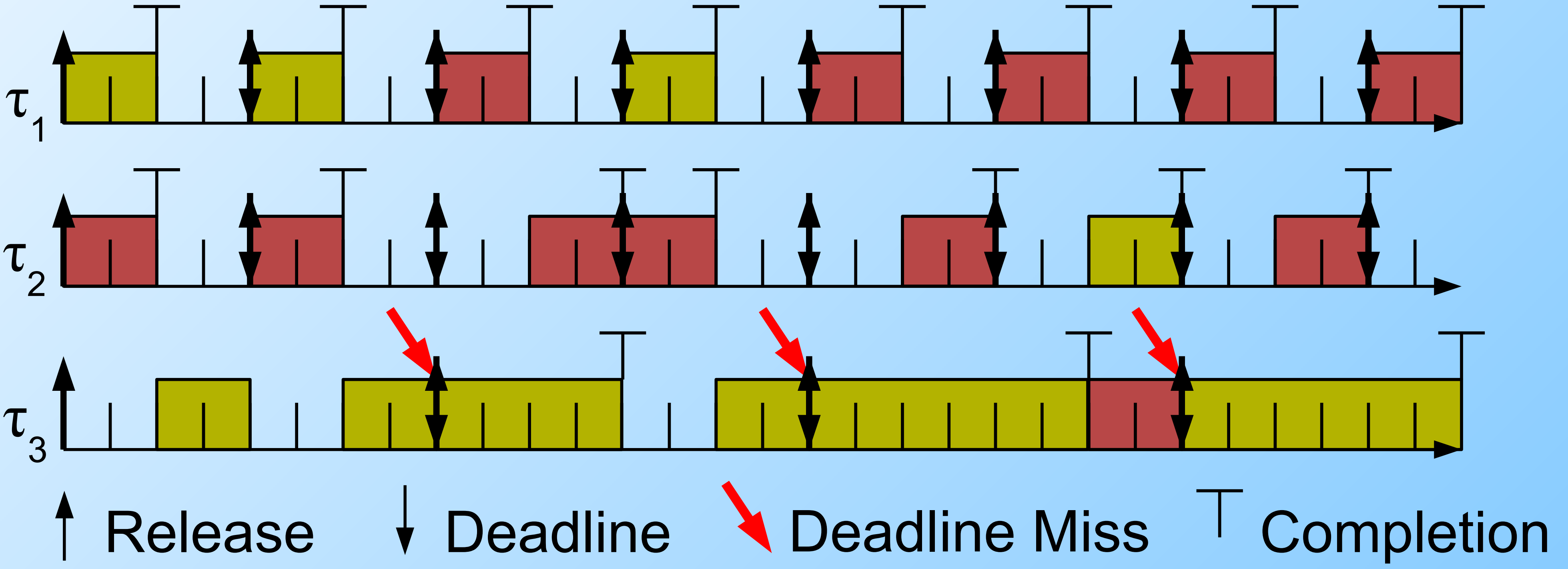


Problem with Alternative Schedulers

- Can have high overheads
- May be difficult to implement
- Jobs can change priorities while running – causes problems with synchronization

Bounded Lateness

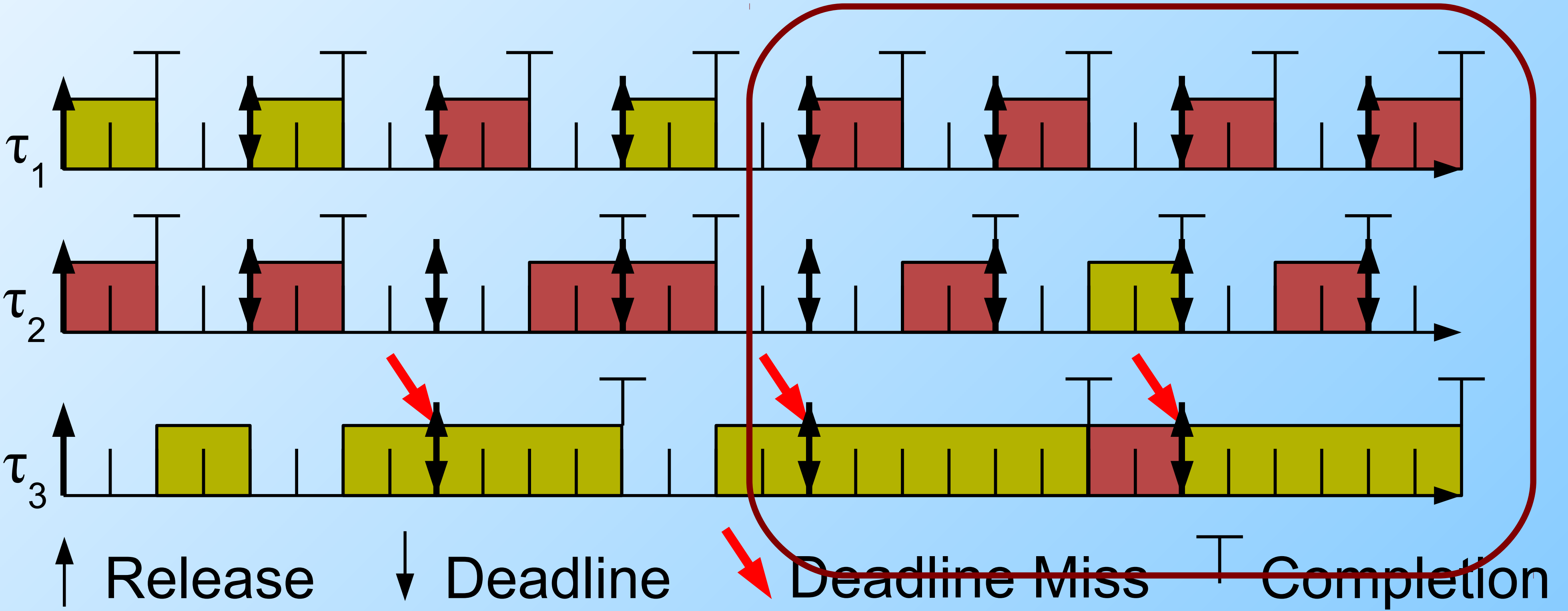
- G-EDF does provide *bounded lateness*.



Bounded Lateness

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- G-EDF does provide *bounded lateness*.

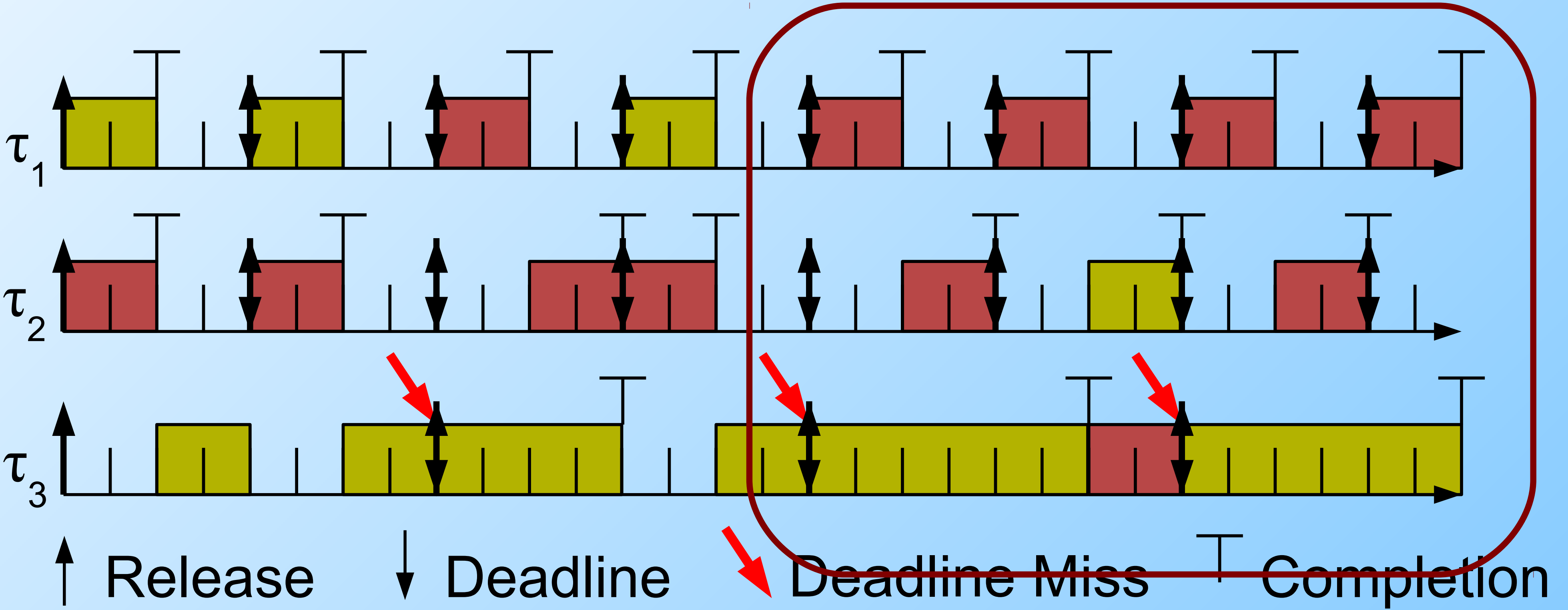


Bounded Lateness

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

- G-EDF does provide *bounded lateness*.

Schedule repeats itself.



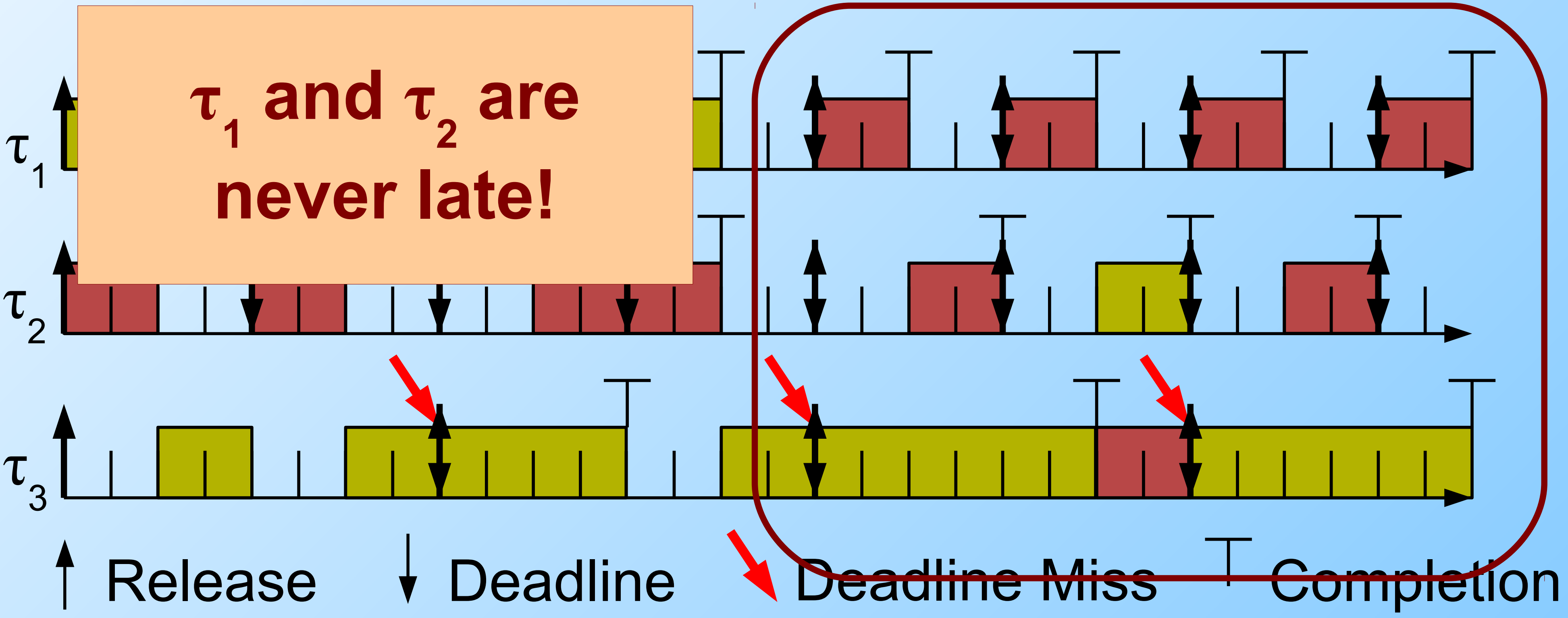
Bounded Lateness

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

- G-EDF does provide *bounded lateness*.

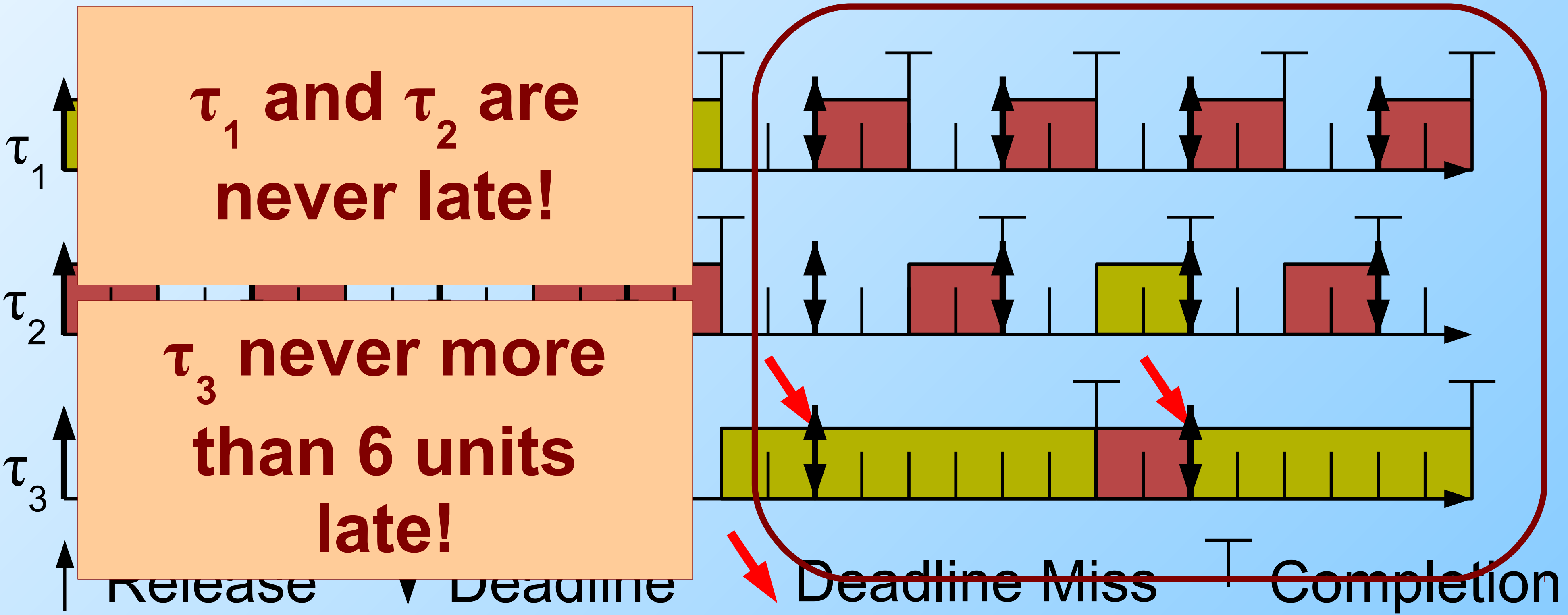
Schedule repeats itself.

τ_1 and τ_2 are never late!



Bounded Lateness

- G-EDF does provide *bounded lateness*.



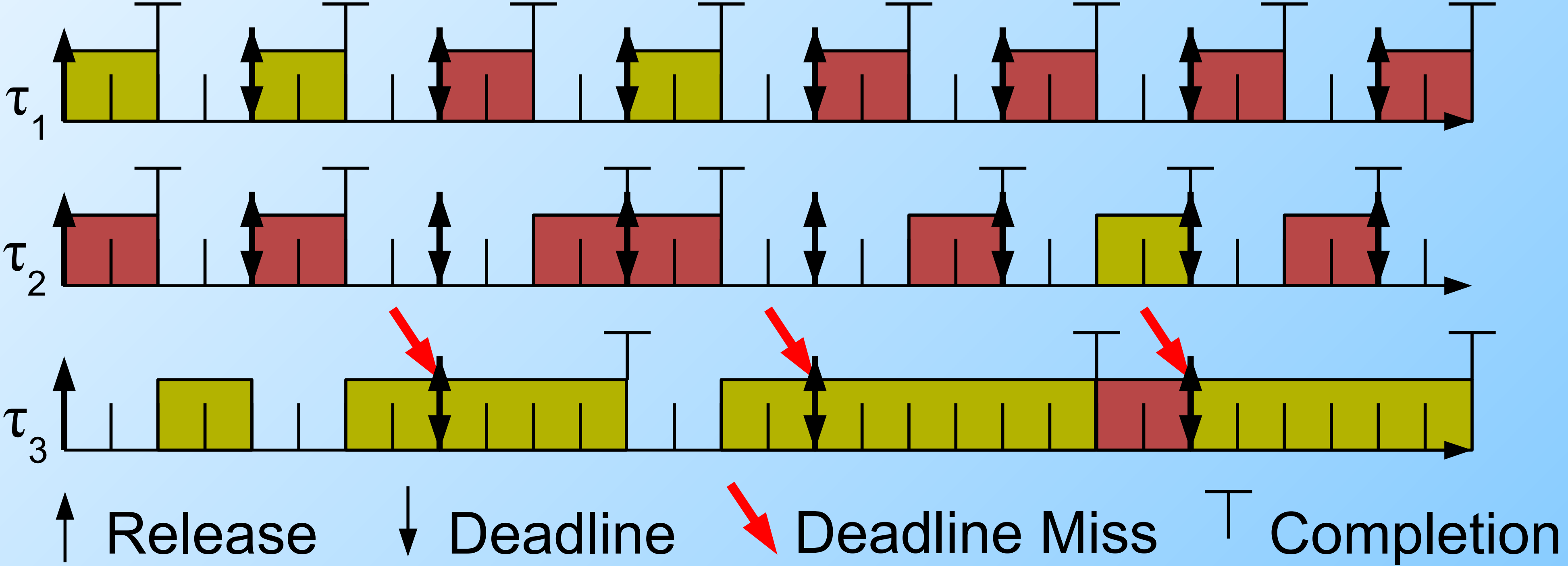
Schedule repeats itself.

τ_1 and τ_2 are never late!

τ_3 never more than 6 units late!

Prior Work

- Can already determine *tardiness* bounds given system parameters
- Larger WCETs = larger bounds



Can We Do Better?

- Obviously possible with optimal schedulers
- But can we do so without the disadvantages of those schedulers?
- **Yes.**

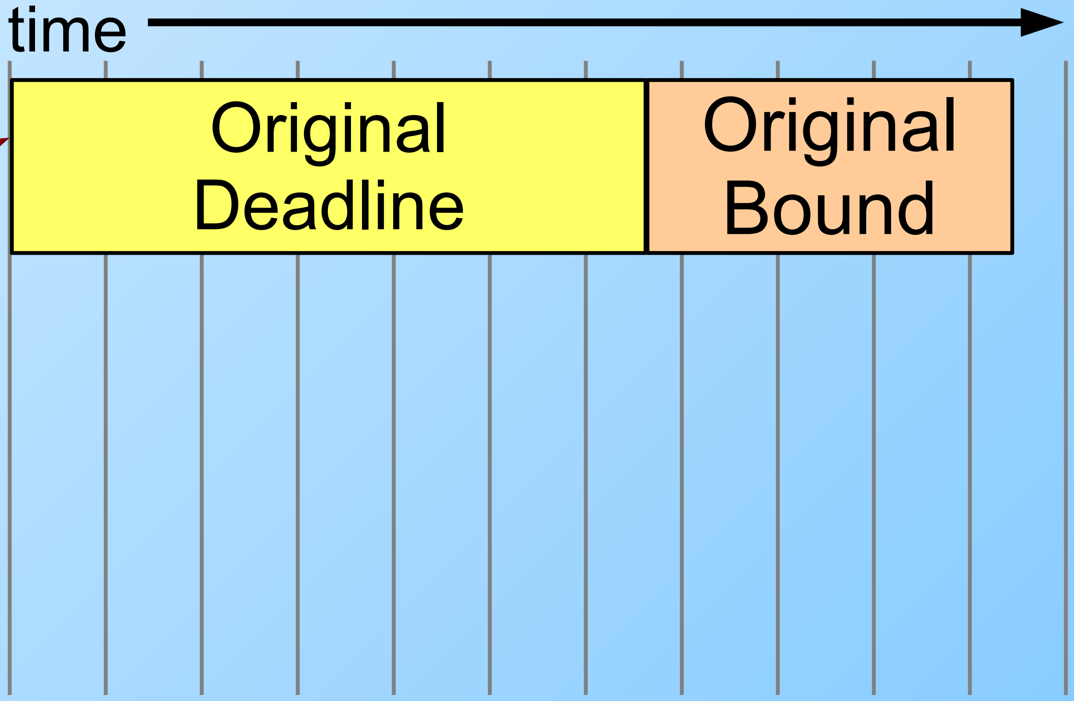
Priority Points

- Deadlines serve **both** to determine scheduler priorities **and** to specify when a job should be complete.
- We separate out these ideas (concept of **priority point** from Leontyev and Anderson 2007).

Priority Points

- Deadlines serve **both** to determine scheduler priorities **and** to specify when a job should be complete.
- We separate out these ideas (concept of **priority point** from Leontyev and Anderson 2007).

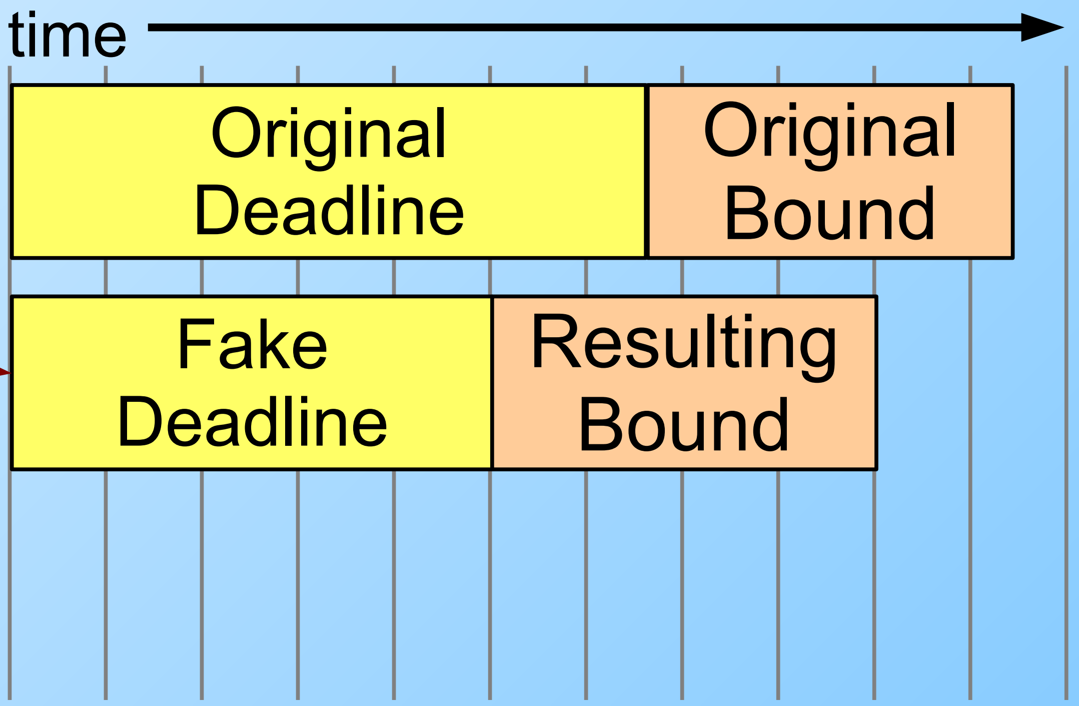
Actual deadline determines priority (old analysis)



Priority Points

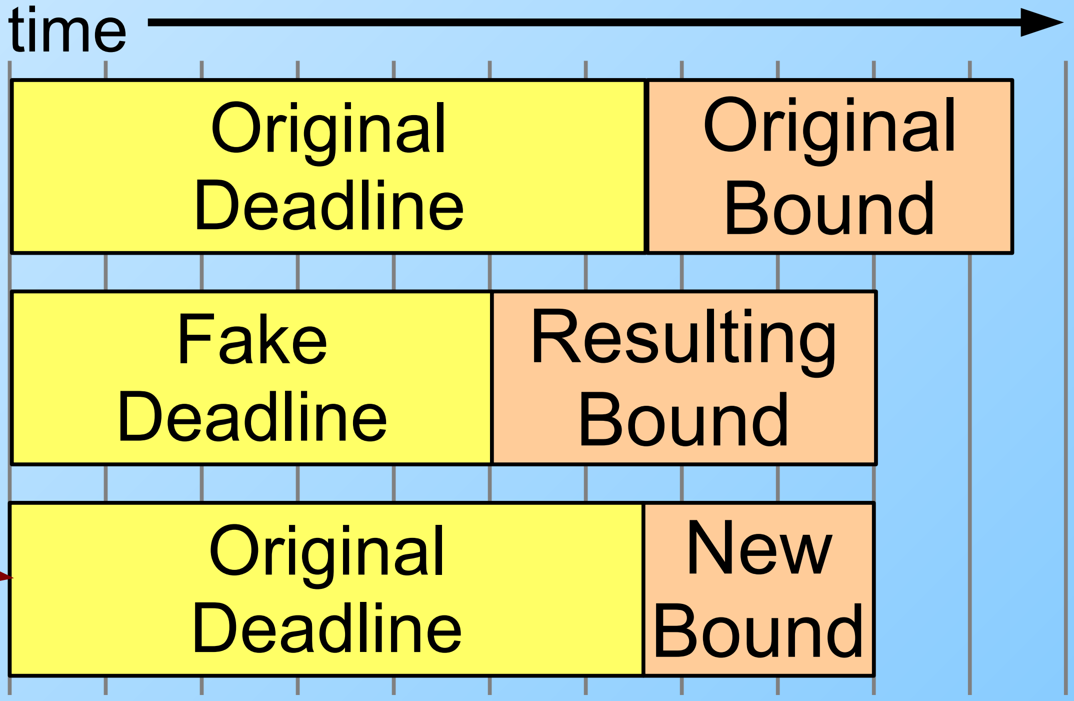
- Deadlines serve **both** to determine scheduler priorities **and** to specify when a job should be complete.
- We separate out these ideas (concept of **priority point** from Leontyev and Anderson 2007).

Use a shorter "deadline" as priority point. Plug into old analysis.



Priority Points

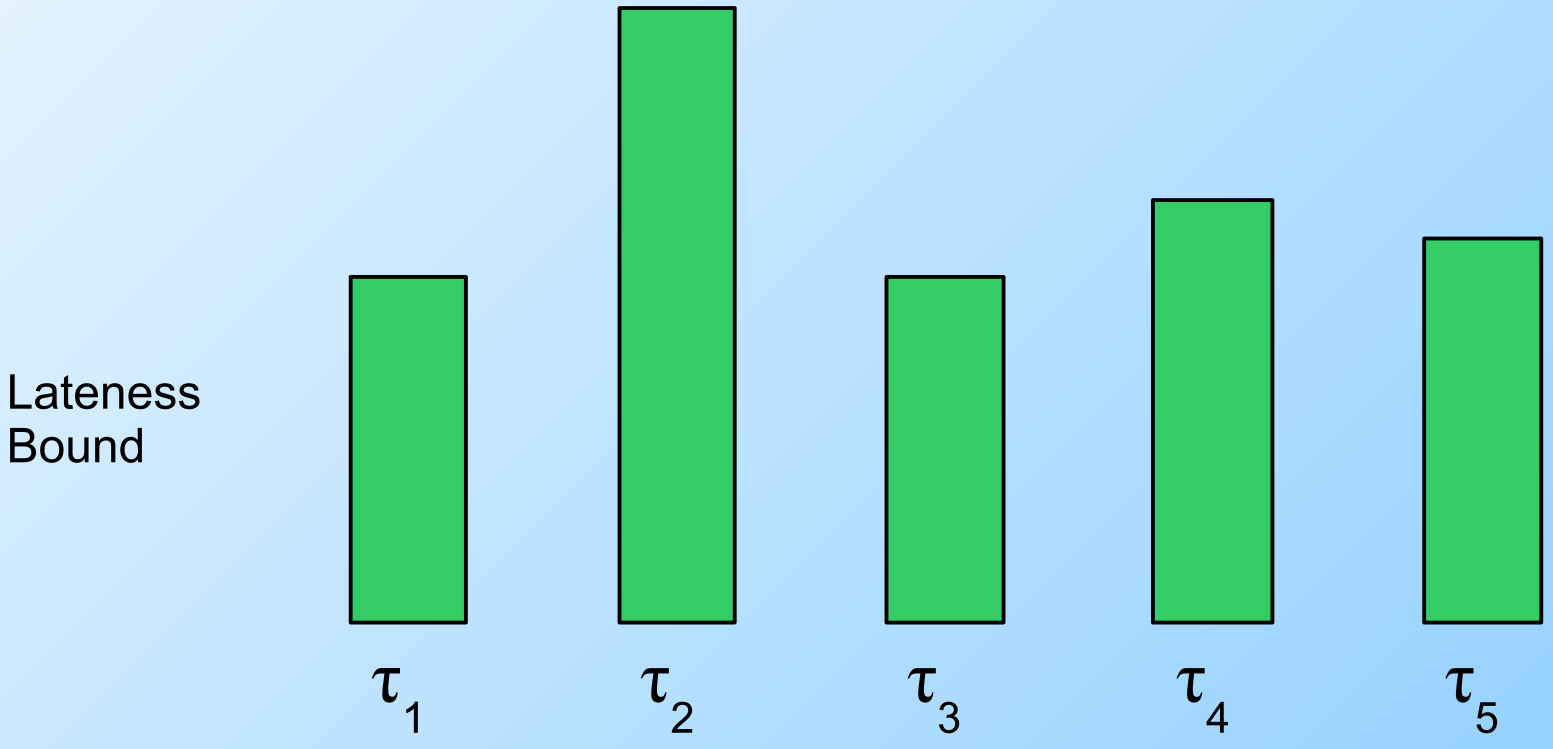
- Deadlines serve **both** to determine scheduler priorities **and** to specify when a job should be complete.
- We separate out these ideas (concept of **priority point** from Leontyev and Anderson 2007).



Use that response time to compute new lateness bound.

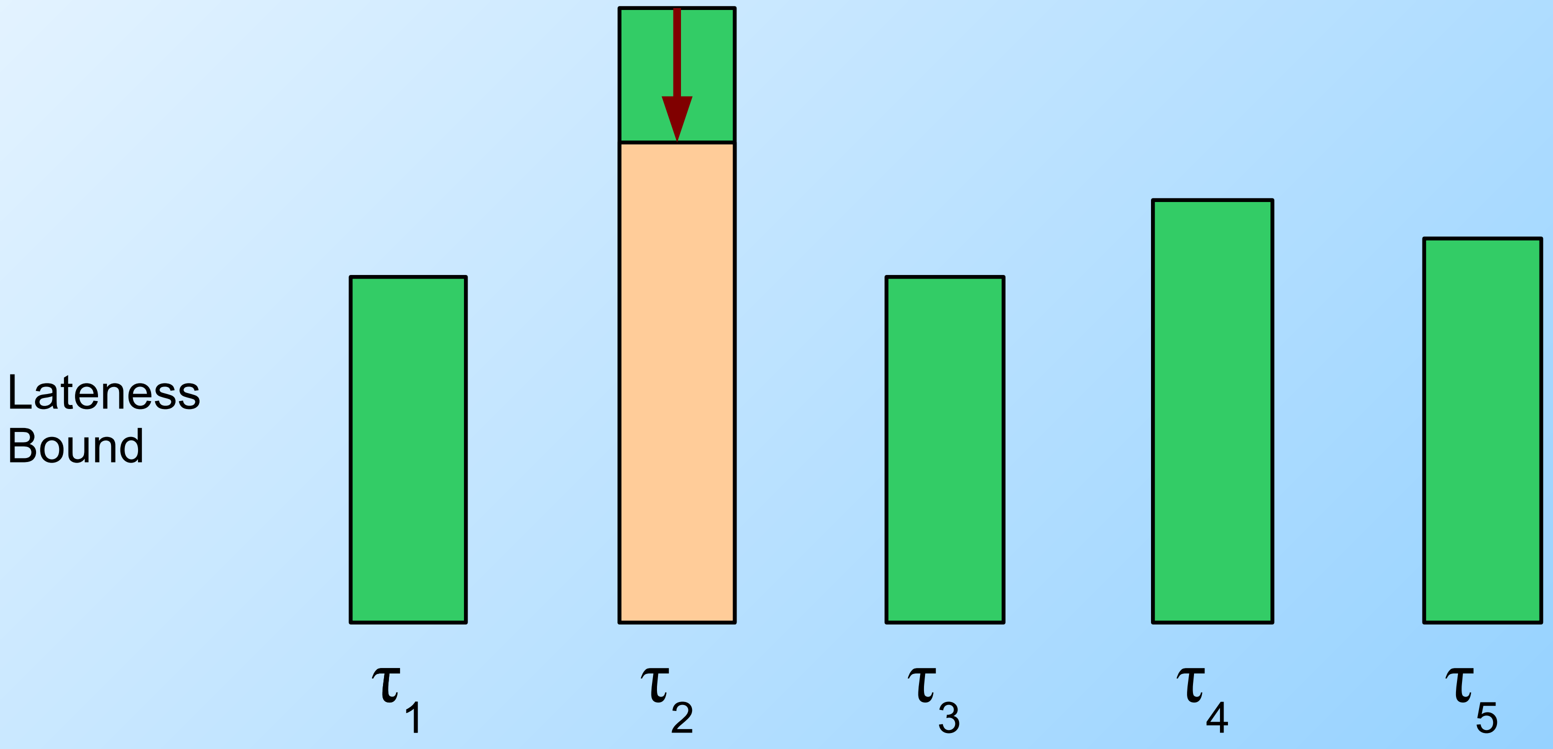
Reducing Priority Points

- Using an earlier priority point improves the bound for that task at the expense of other tasks.



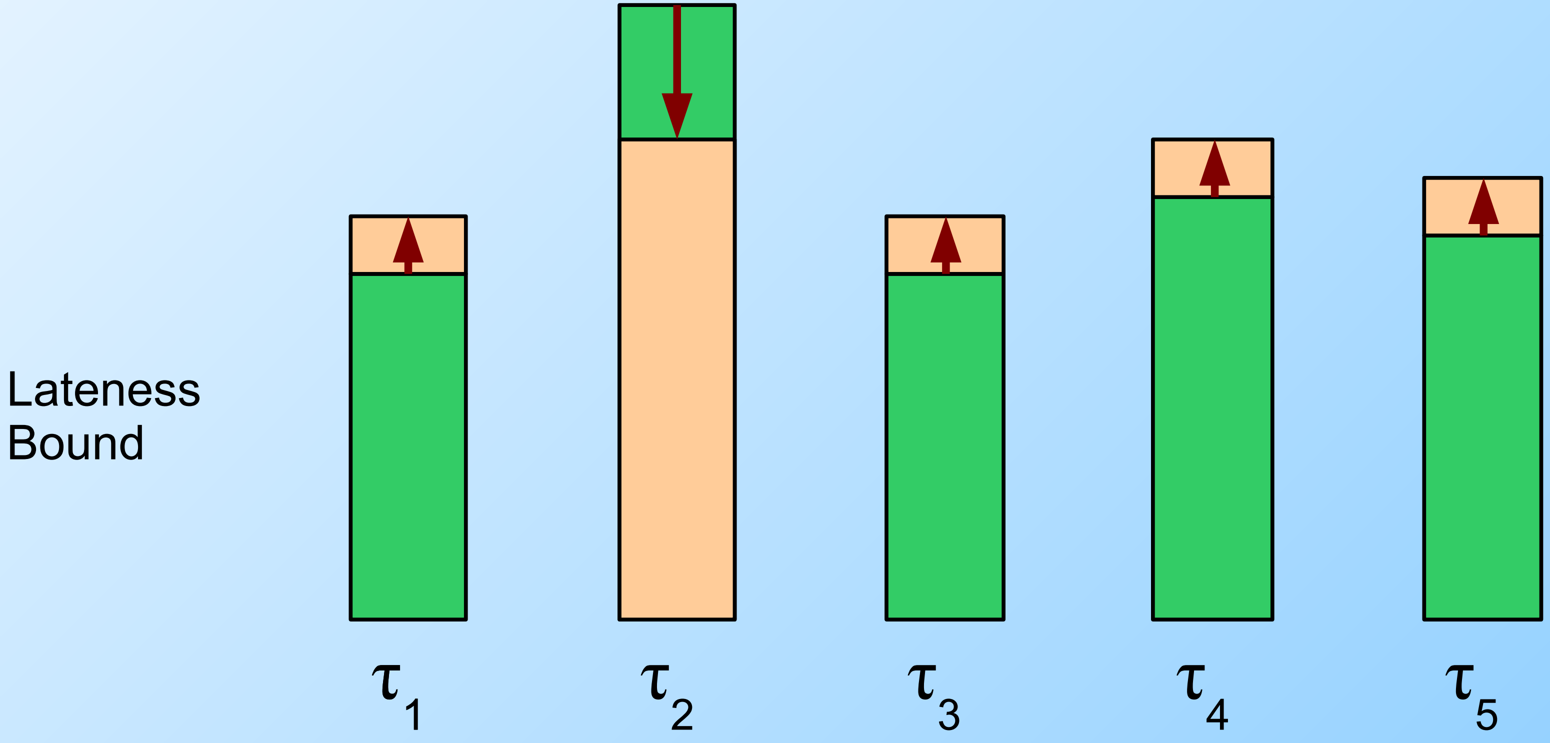
Reducing Priority Points

- Using an earlier priority point improves the bound for that task at the expense of other tasks.



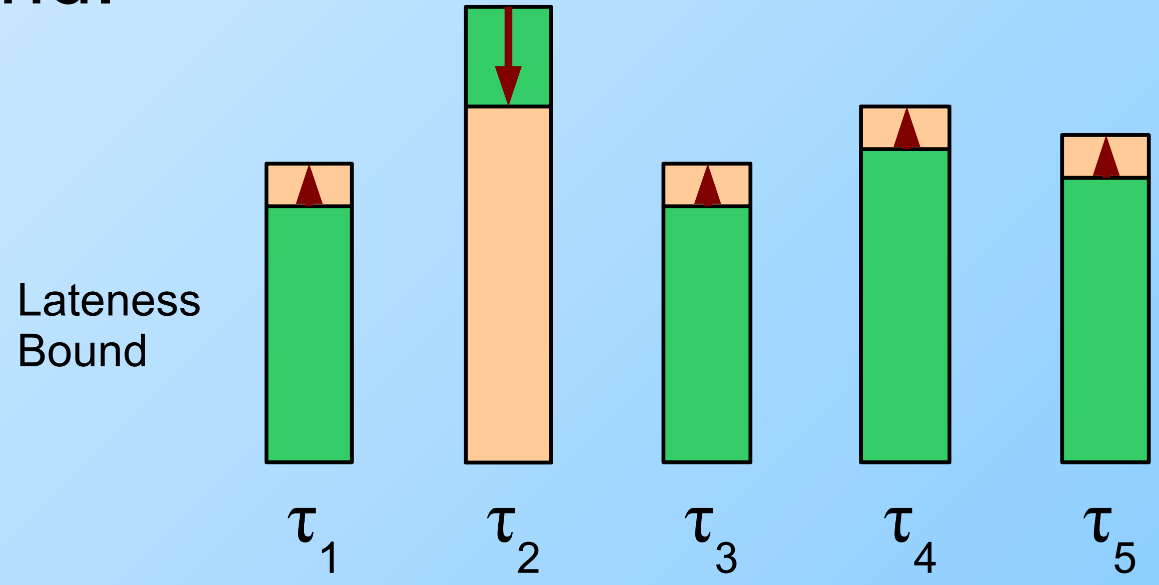
Reducing Priority Points

- Using an earlier priority point improves the bound for that task at the expense of other tasks.



Best Assignment

- What is the “best” assignment?
- Our metric: minimize the **maximum** lateness bound.
- Optimal solution happens when **all** tasks have the **same** bound.



Fair Lateness

- Optimal solution = **fair lateness**.
- Scheduler = **Global Fair Lateness (G-FL)**

G-FL Implementation

- G-FL is G-EDF-like.
- Can use existing arbitrary deadline G-EDF scheduler with “fake deadlines.”

Relative Priority Point	
G-EDF	G-FL
D_i	$D_i - \frac{m-1}{m} C_i$

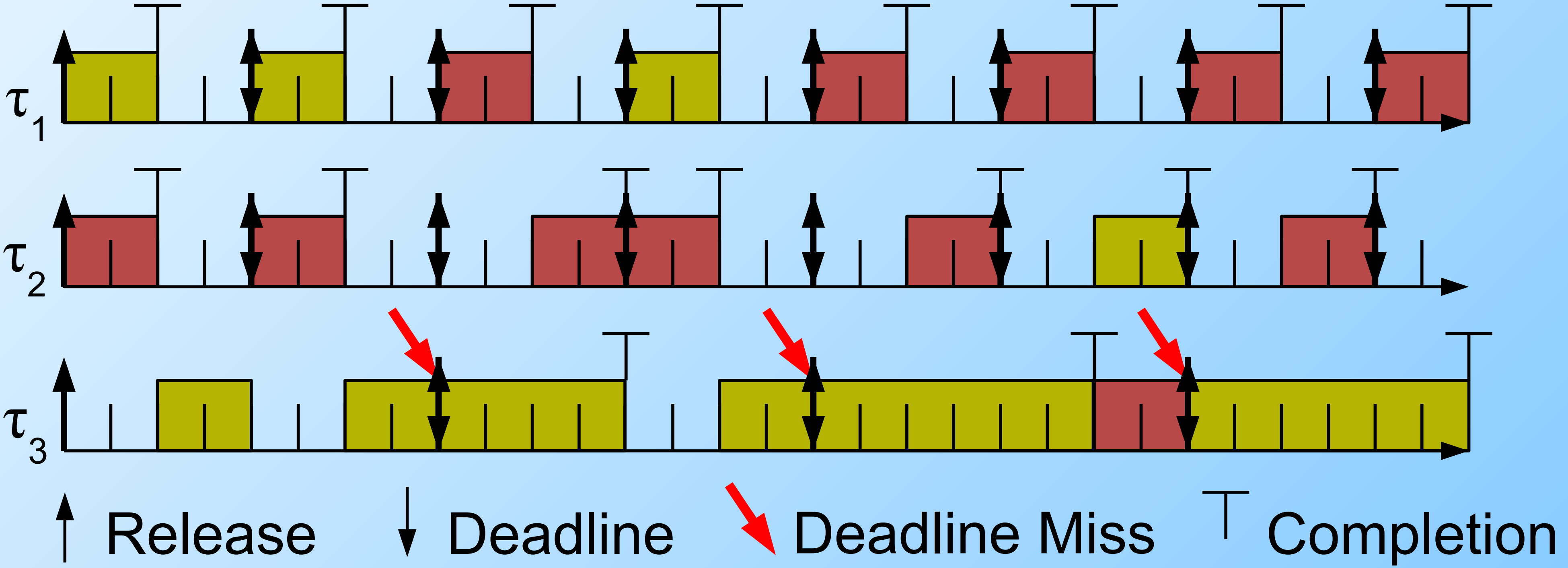
G-FL Implementation

- G-FL is G-EDF-like.
- Can use existing arbitrary deadline G-EDF scheduler with “fake deadlines.”

Relative Priority Point	
G-EDF	G-FL
D_i	$D_i - \frac{m-1}{m} C_i$

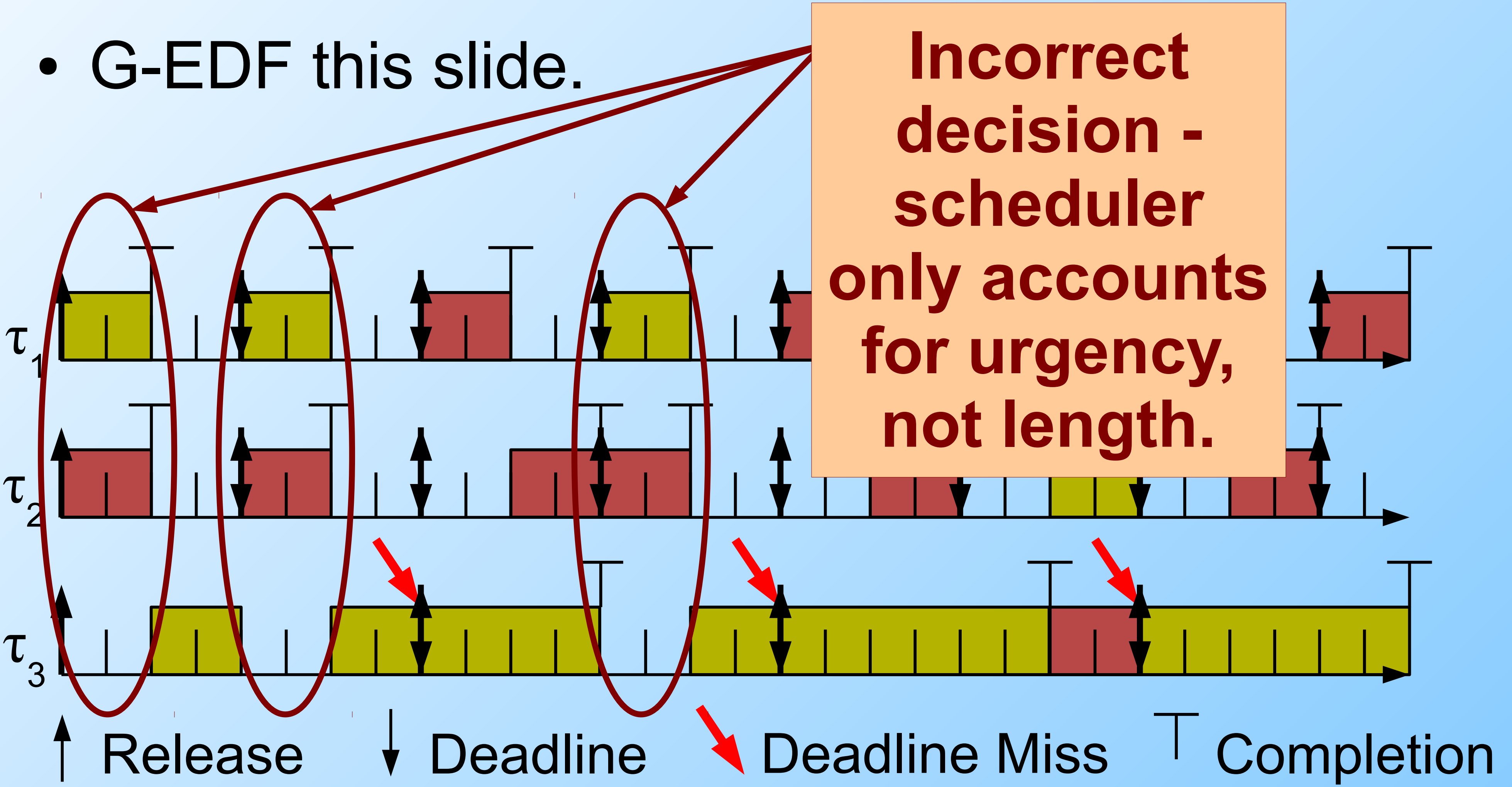
Why Does it Work?

- Due to limited time, only giving intuition here.
- G-EDF this slide.



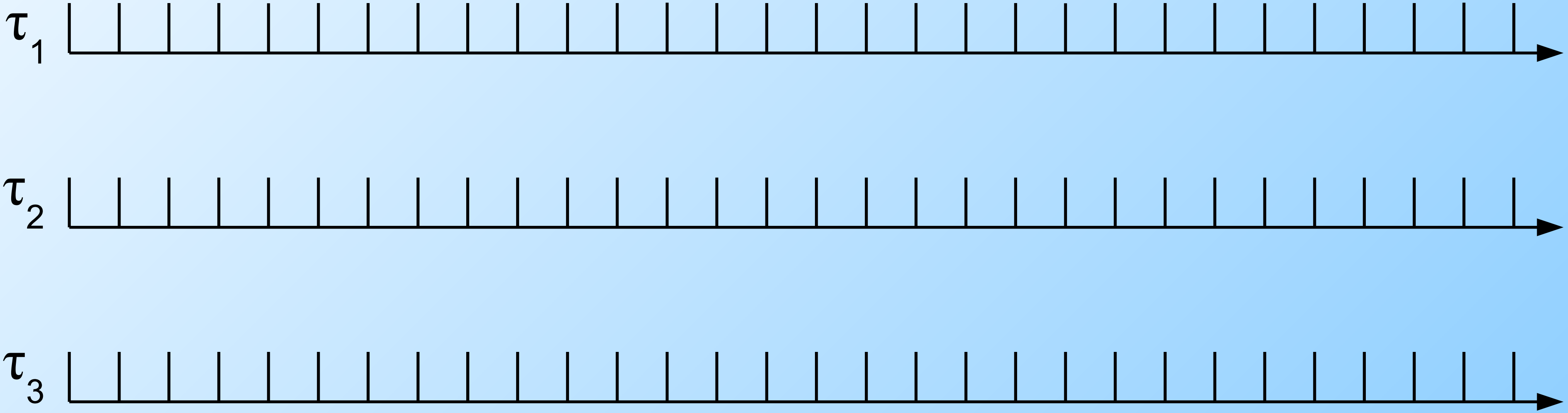
Why Does it Work?

- Due to limited time, only giving intuition here.
- G-EDF this slide.



Incorrect decision - scheduler only accounts for urgency, not length.

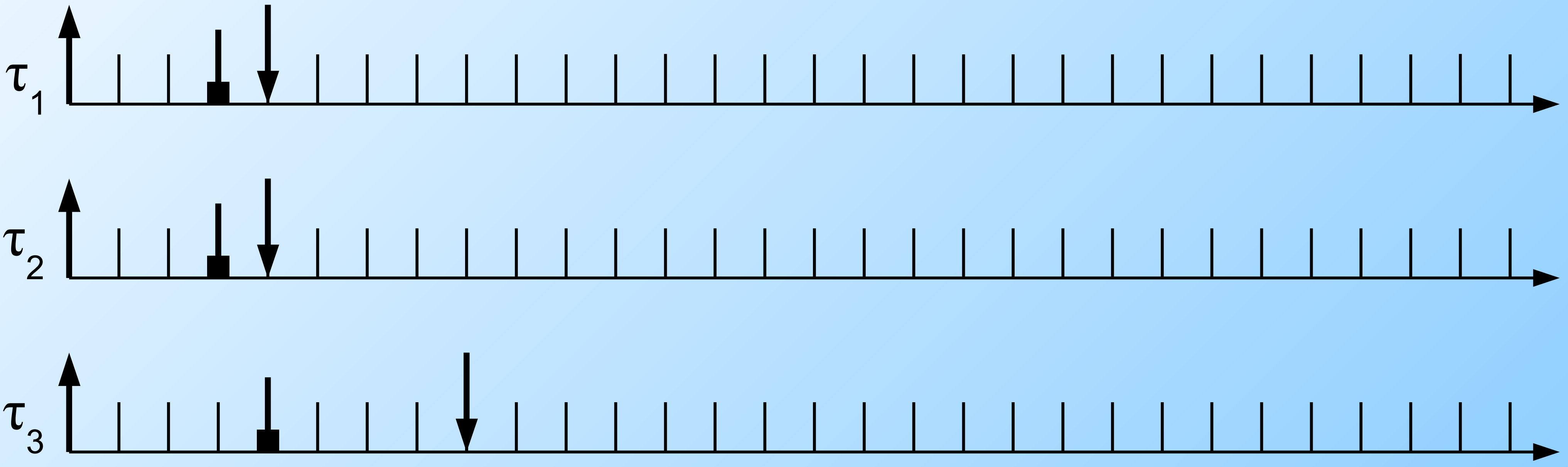
G-FL Schedule



↑ Release ↓ Deadline ▙ Priority Point 🟨 CPU 1

⊤ Completion ↘ Deadline Miss 🟥 CPU 2

G-FL Schedule



↑ Release

↓ Deadline

Priority Point

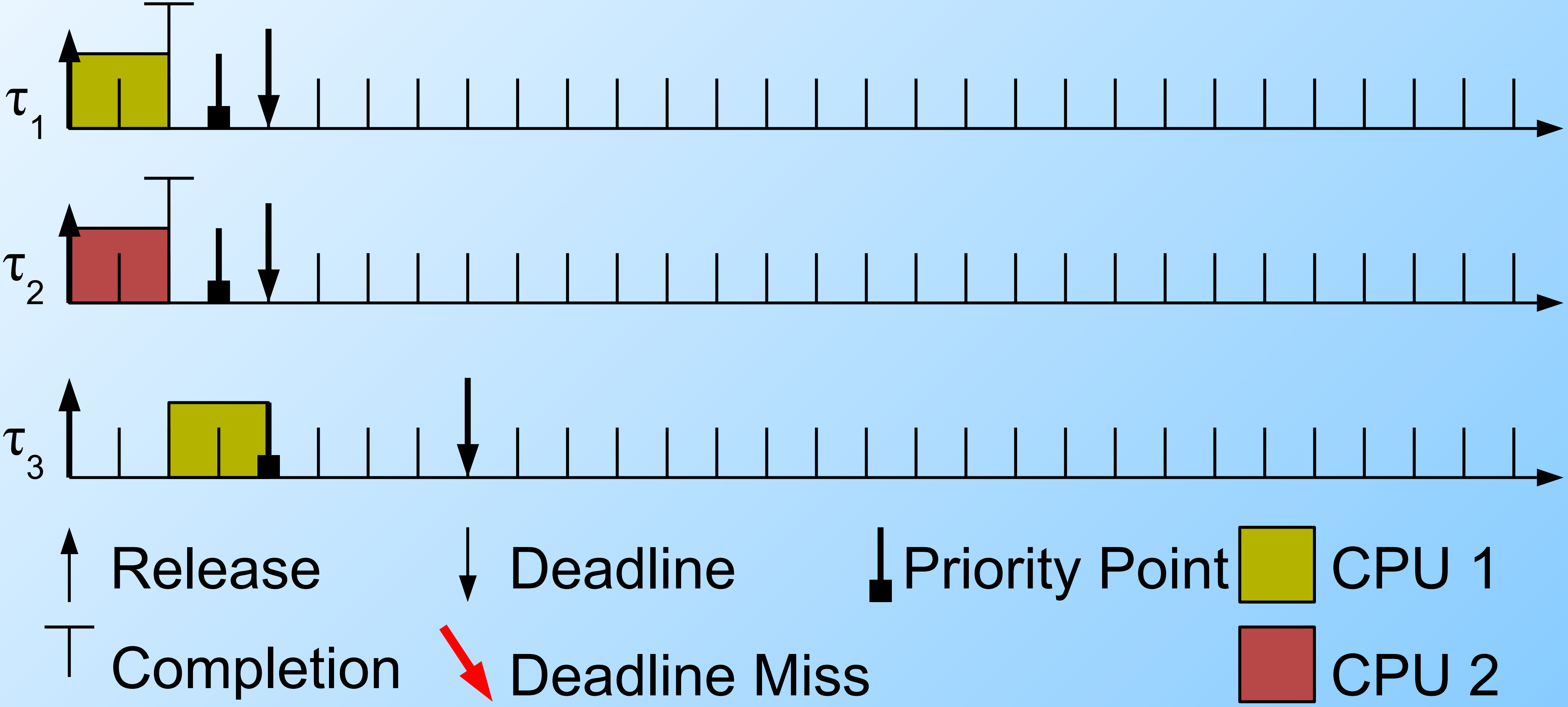
🟨 CPU 1

⊤ Completion

↘ Deadline Miss

🟥 CPU 2

G-FL Schedule



Release

Deadline

Priority Point

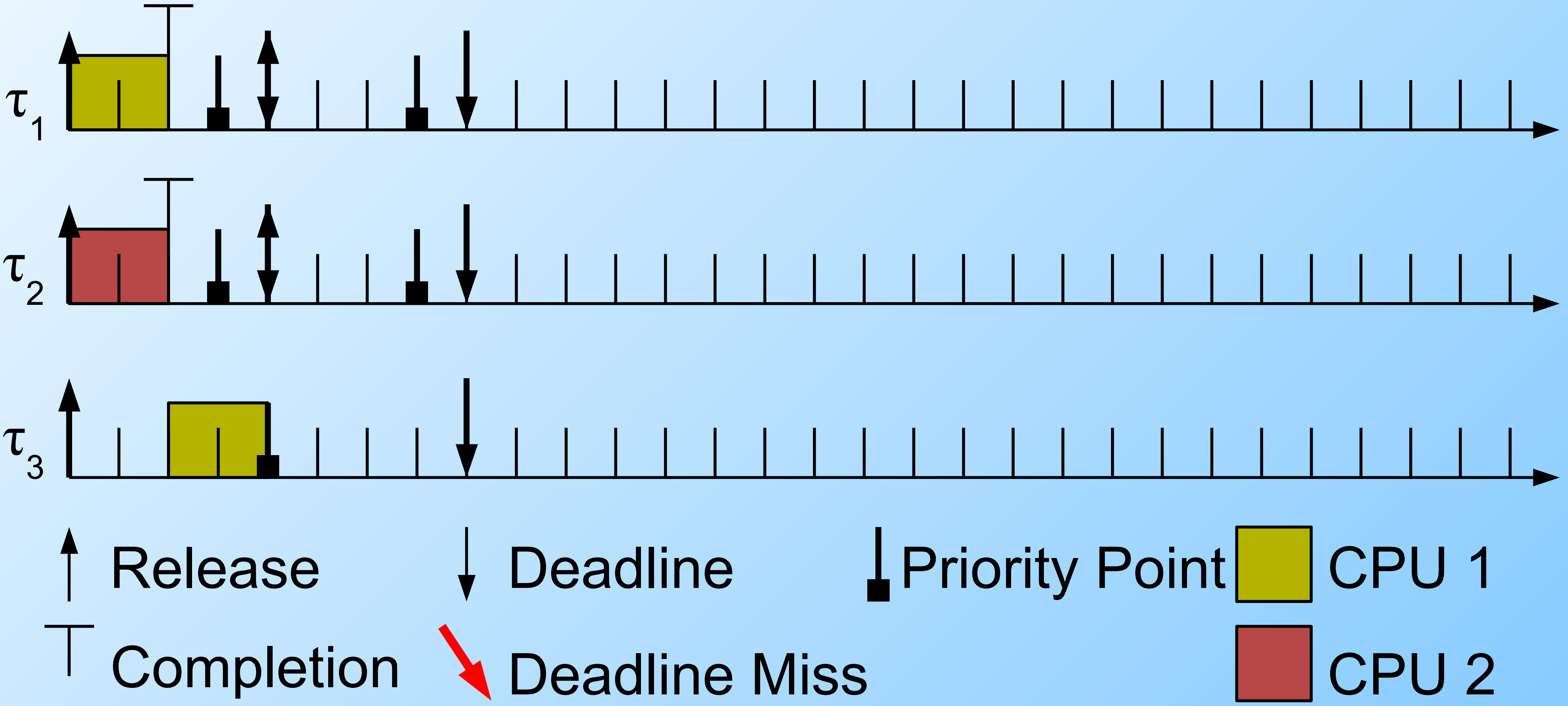
🟨 CPU 1

Completion

Deadline Miss

🟥 CPU 2

G-FL Schedule



↑ Release

↓ Deadline

Priority Point

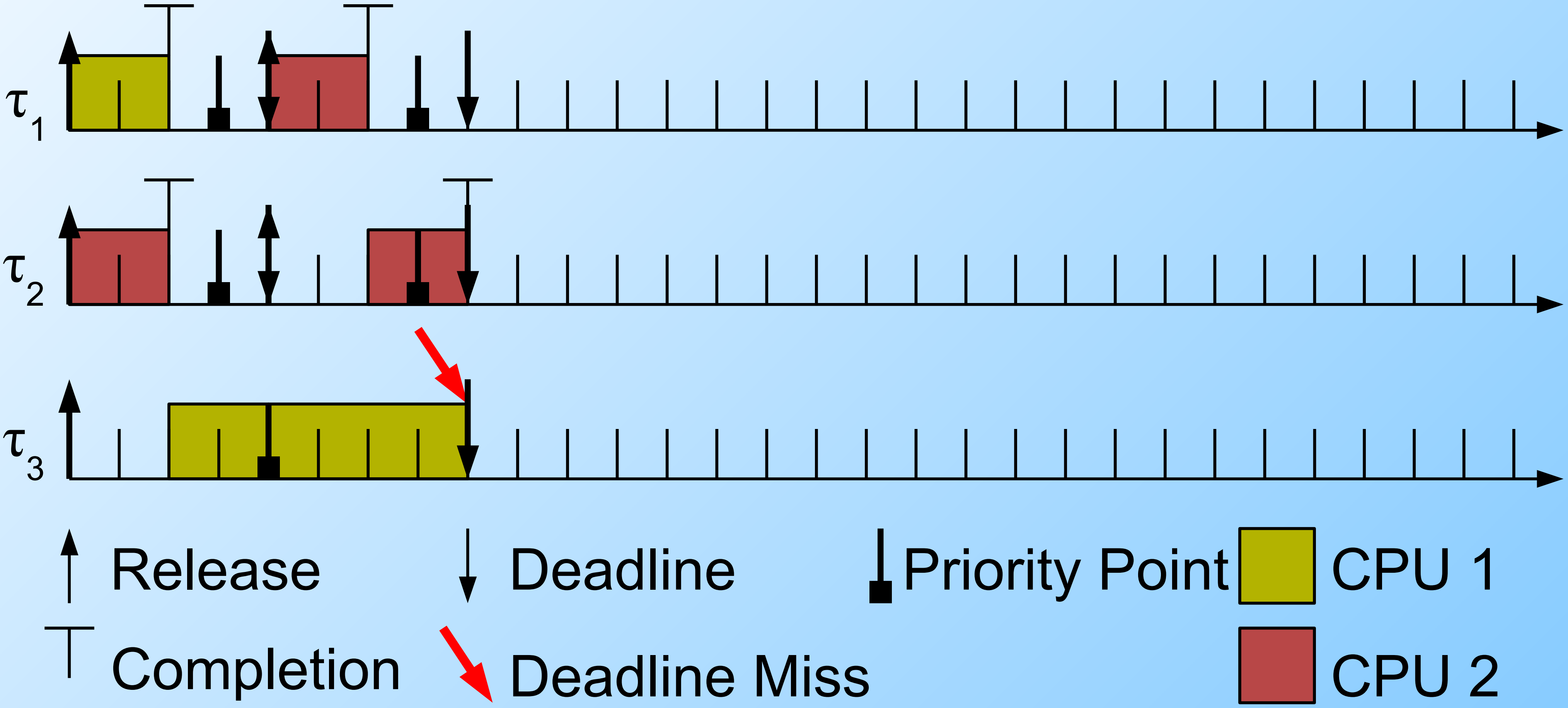
🟨 CPU 1

Completion

Deadline Miss

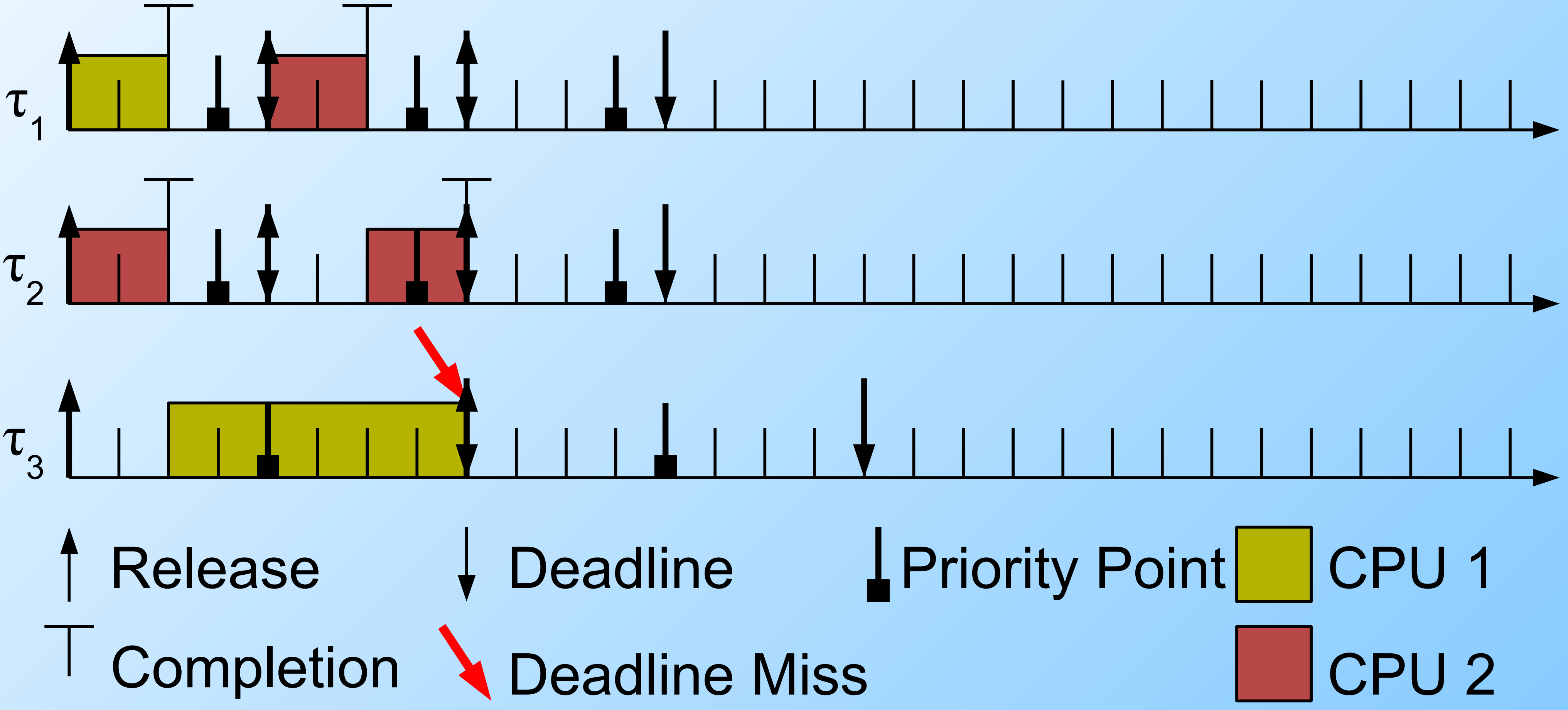
🟥 CPU 2

G-FL Schedule

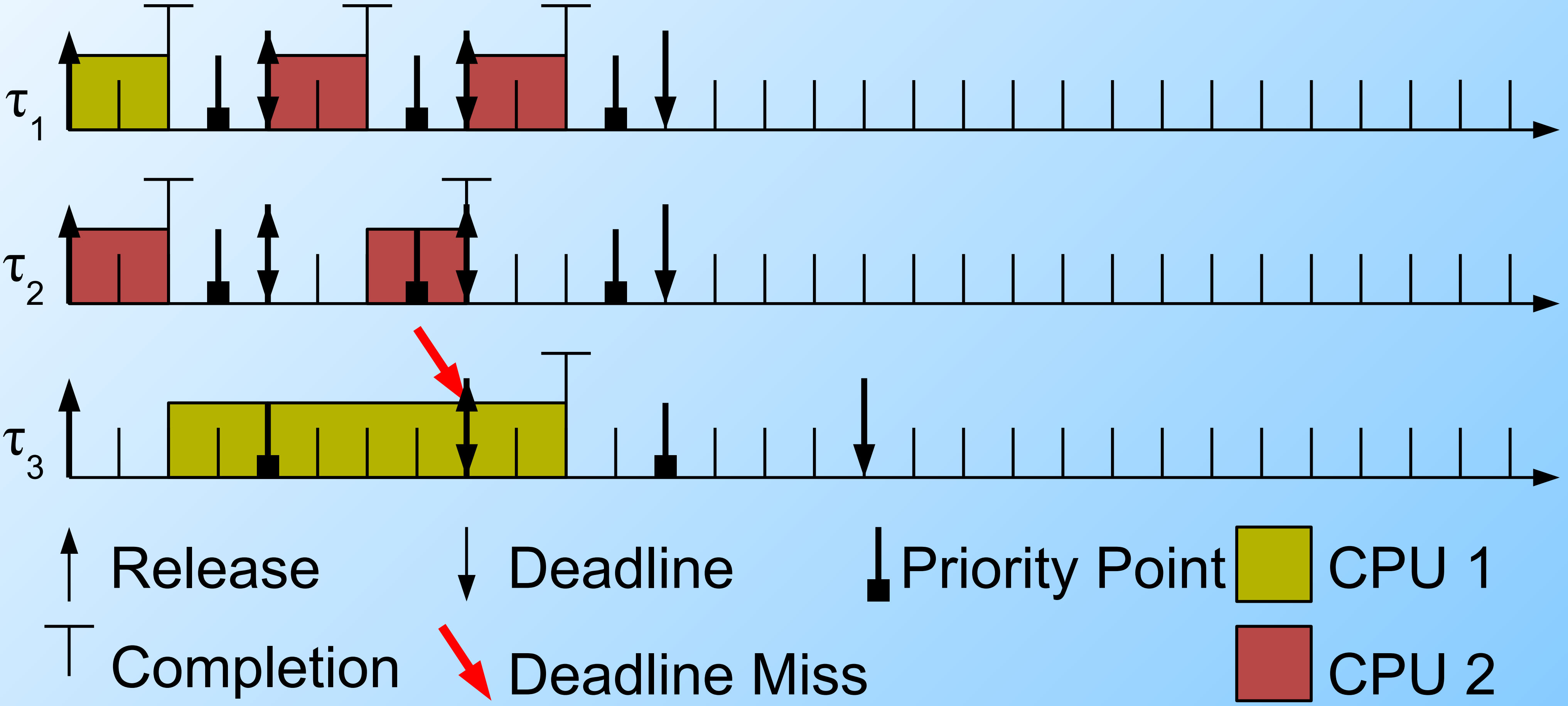


G-FL Schedule

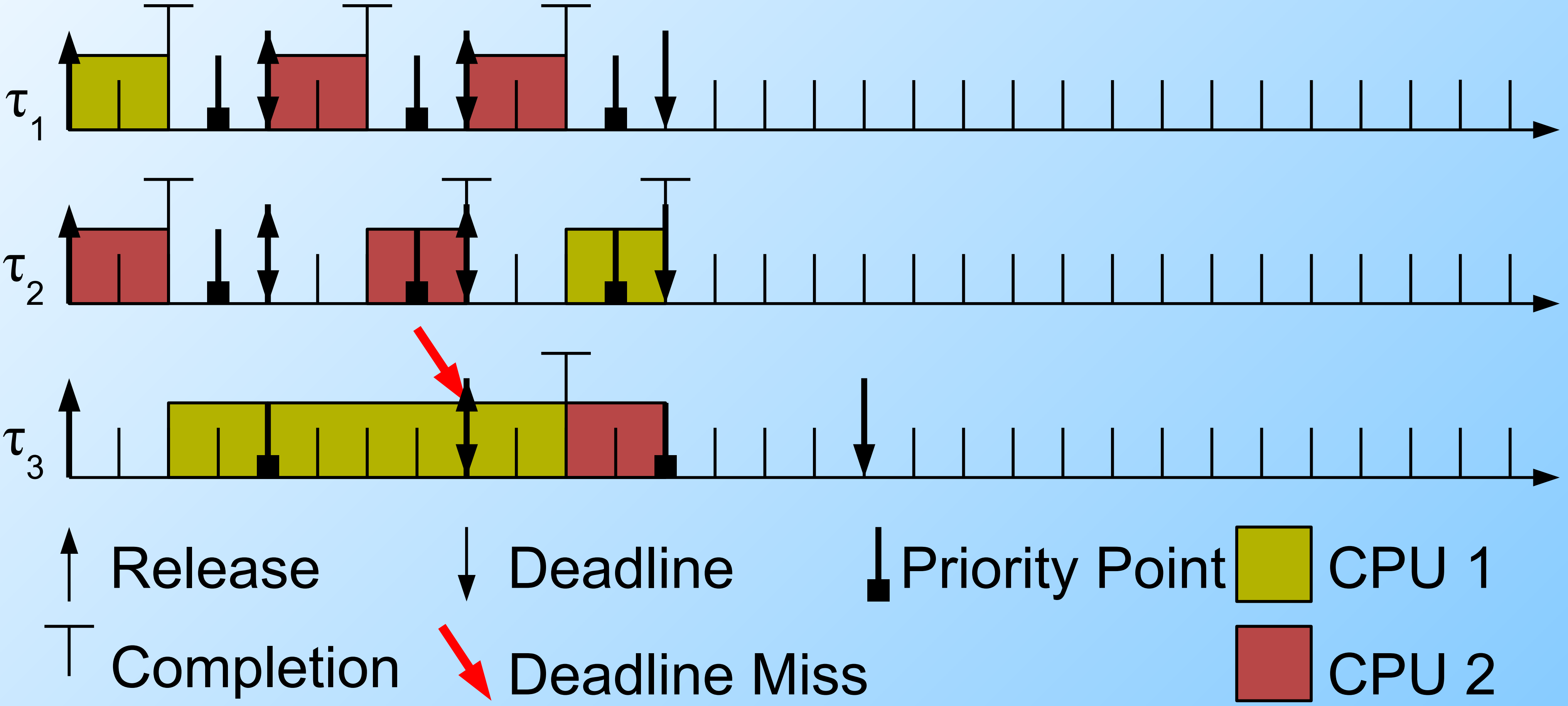
THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL



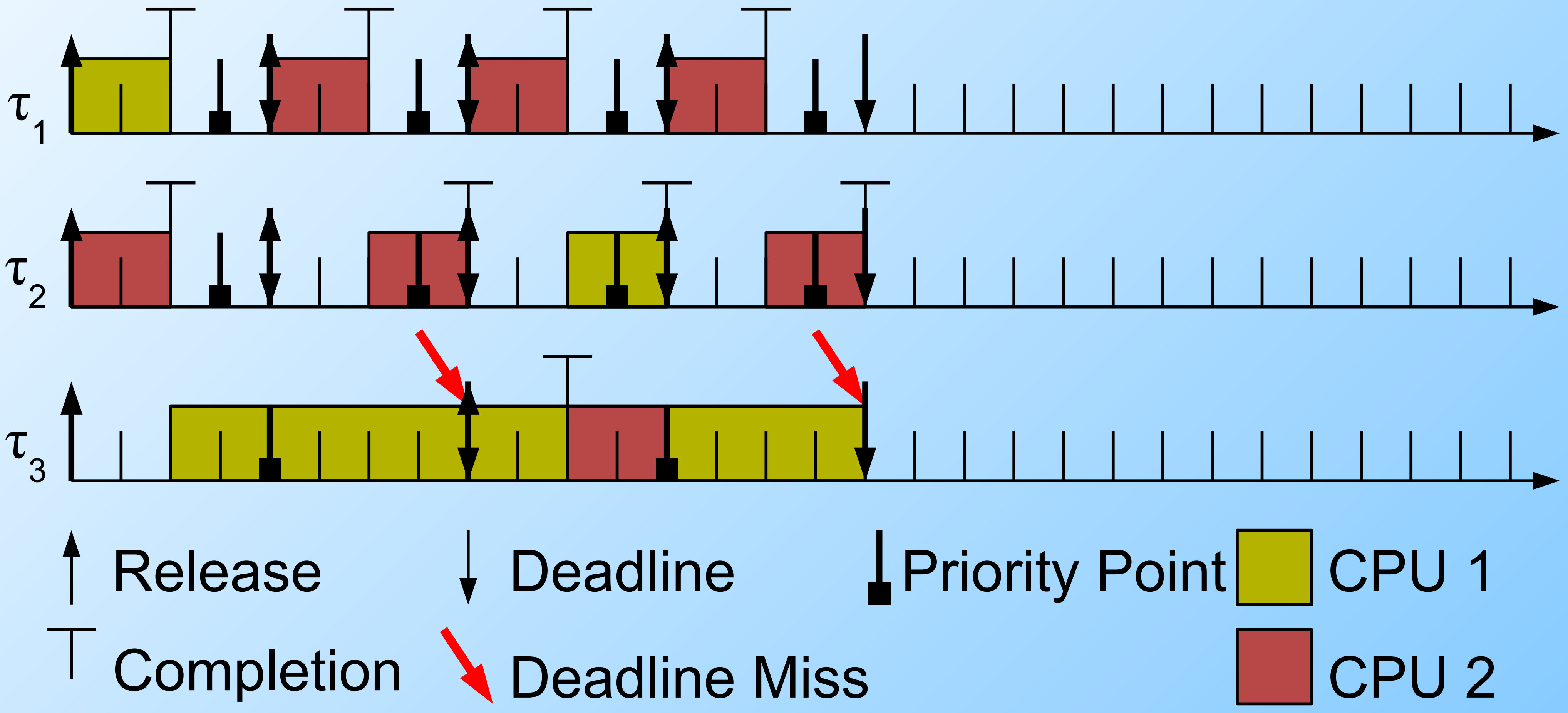
G-FL Schedule



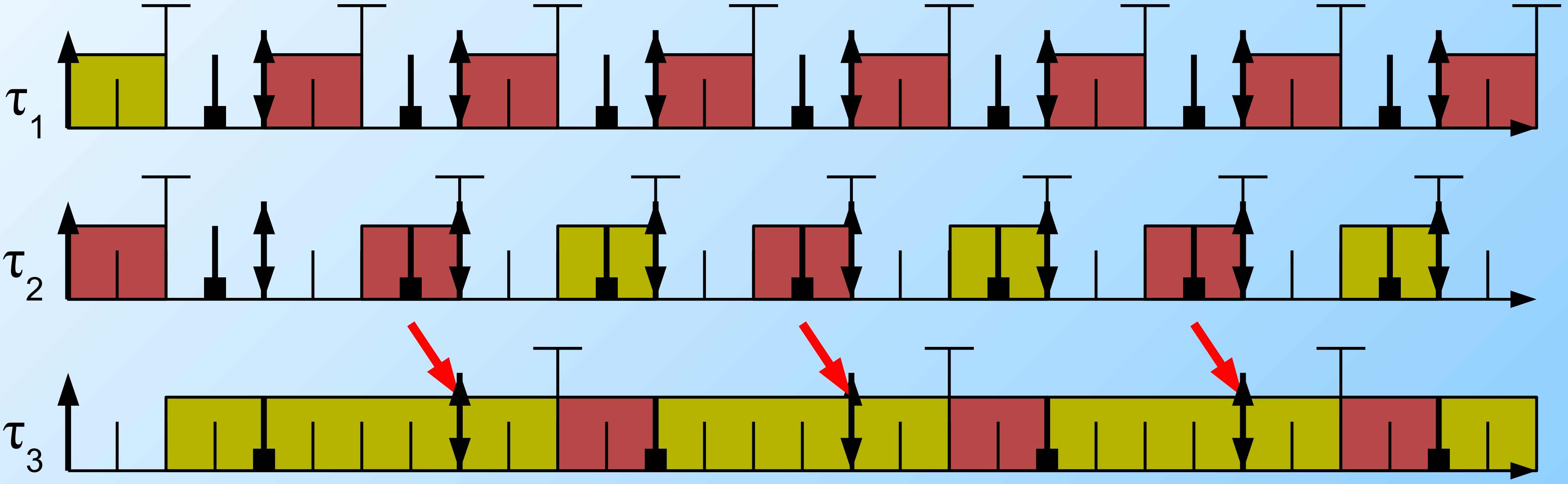
G-FL Schedule



G-FL Schedule



G-FL Schedule



↑ Release

↓ Deadline

Priority Point

🟨 CPU 1

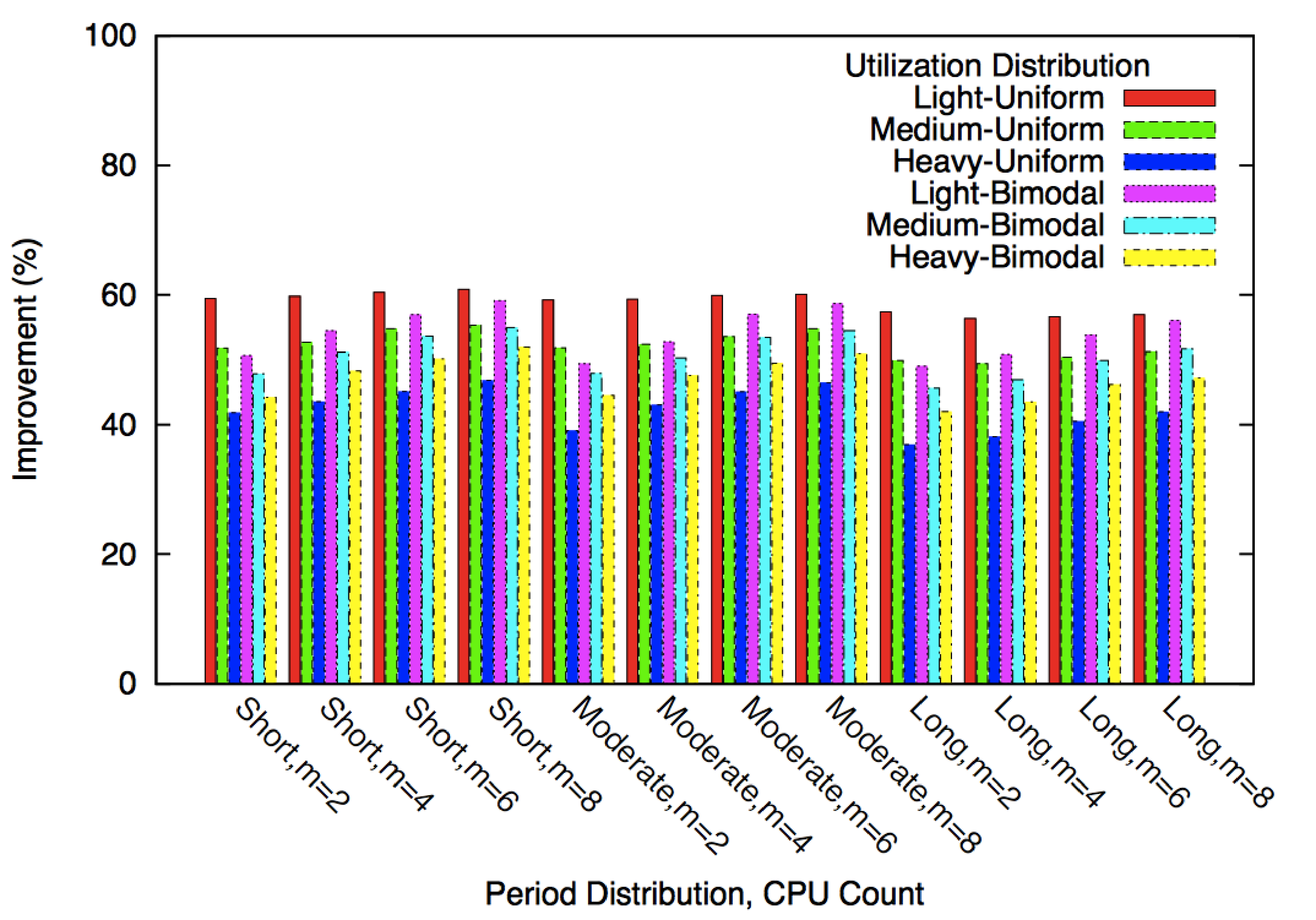
⊤ Completion

↘ Deadline Miss

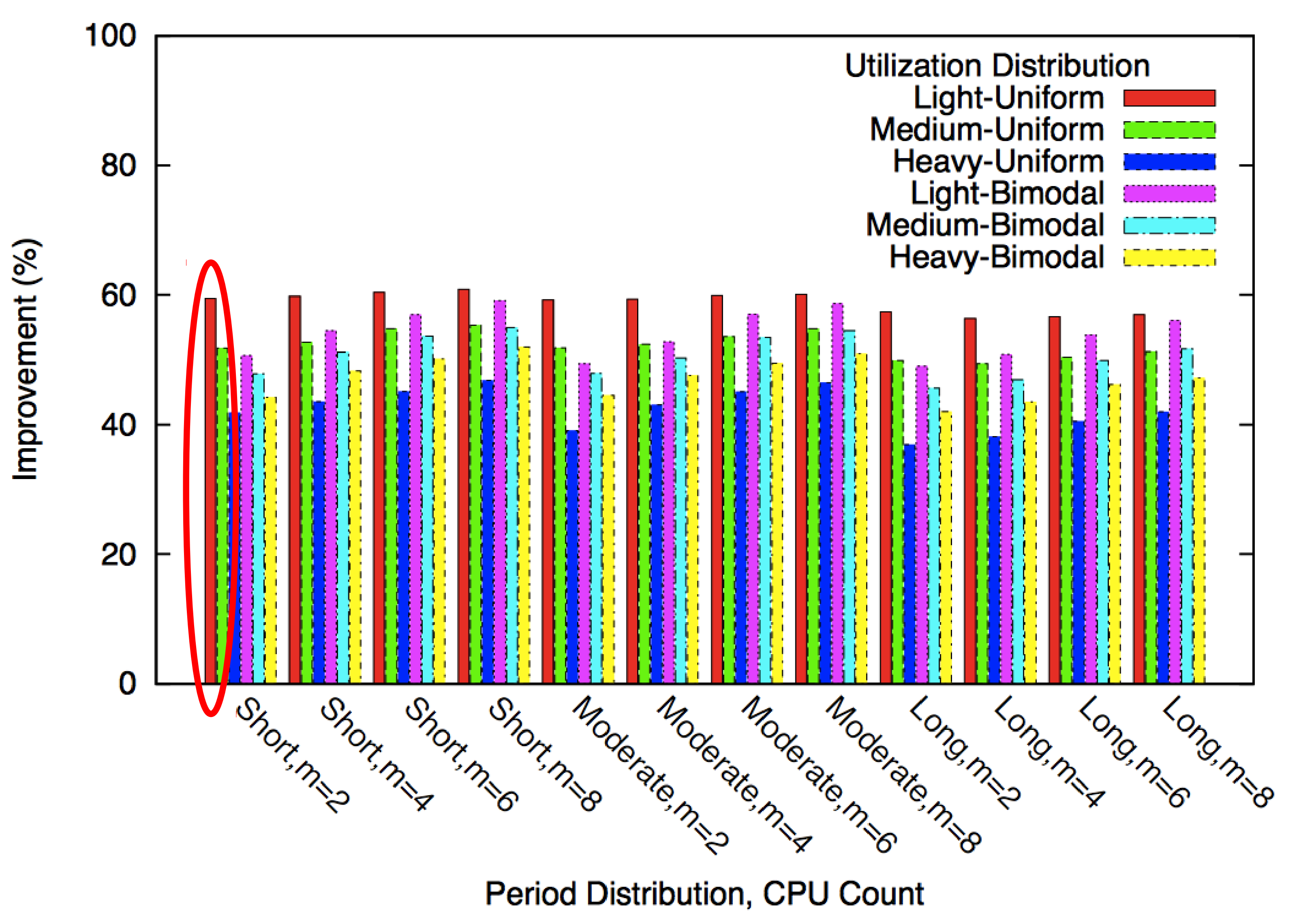
🟥 CPU 2

Experiments - Bounds

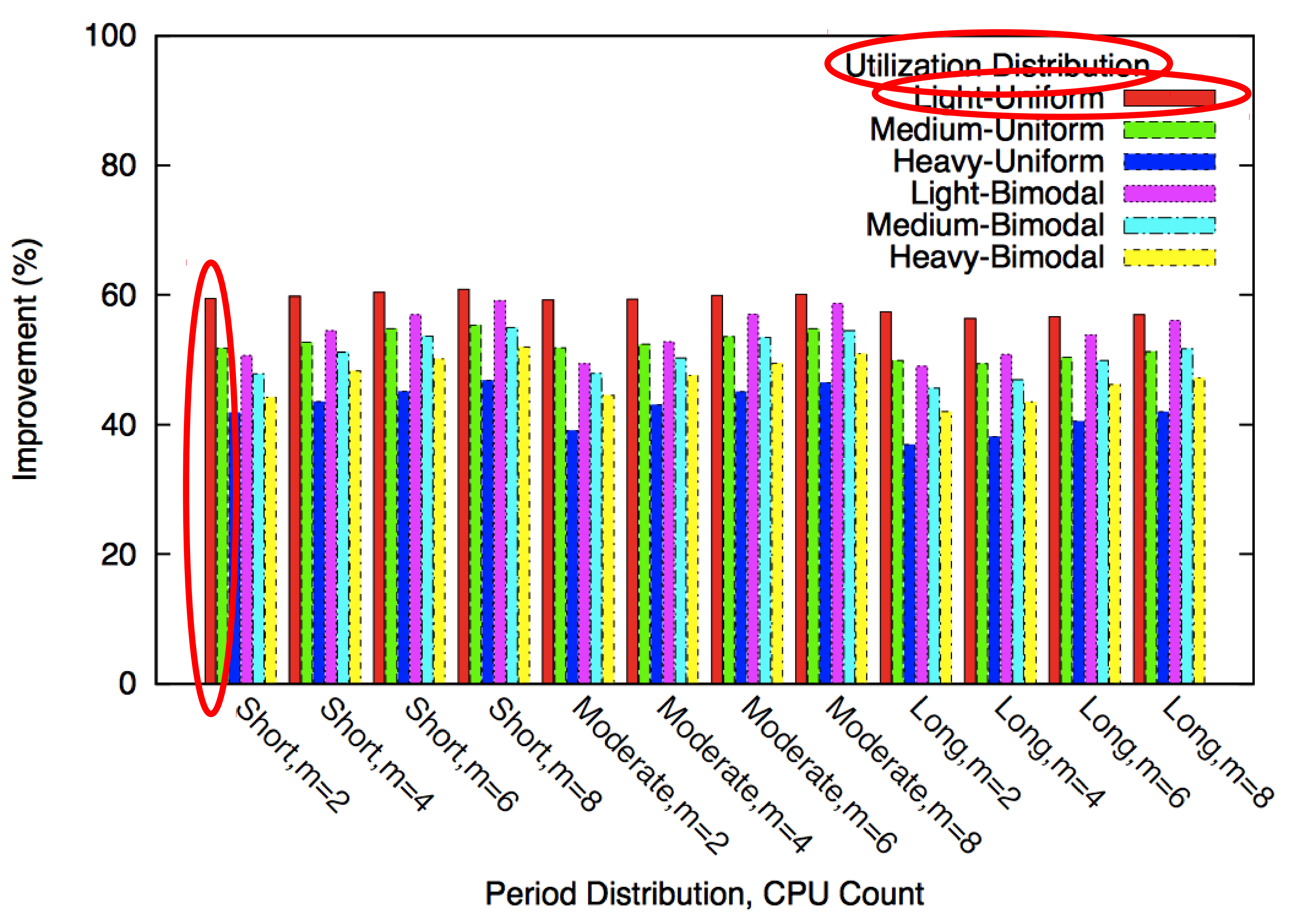
THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL



Experiments - Bounds

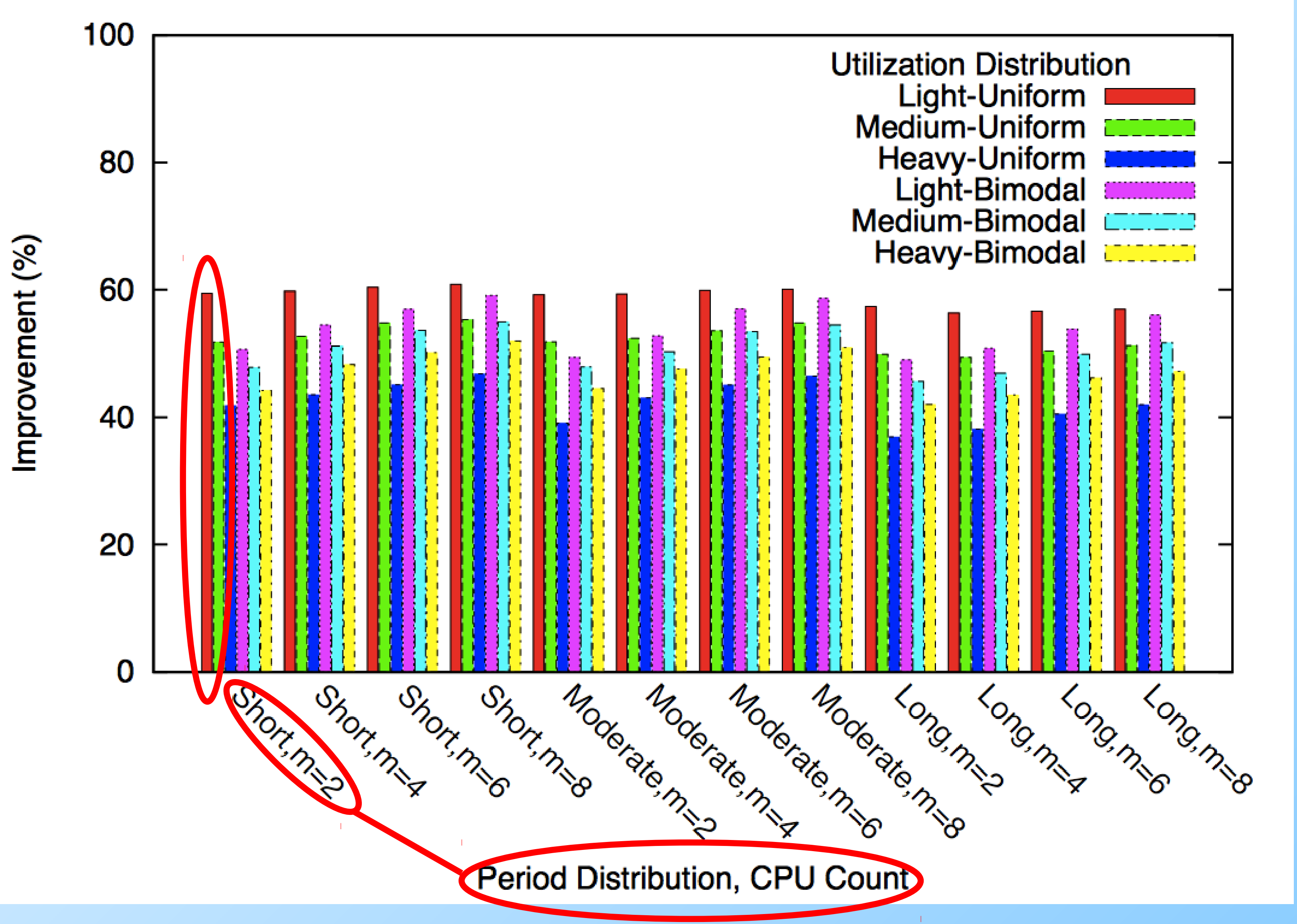


Experiments - Bounds



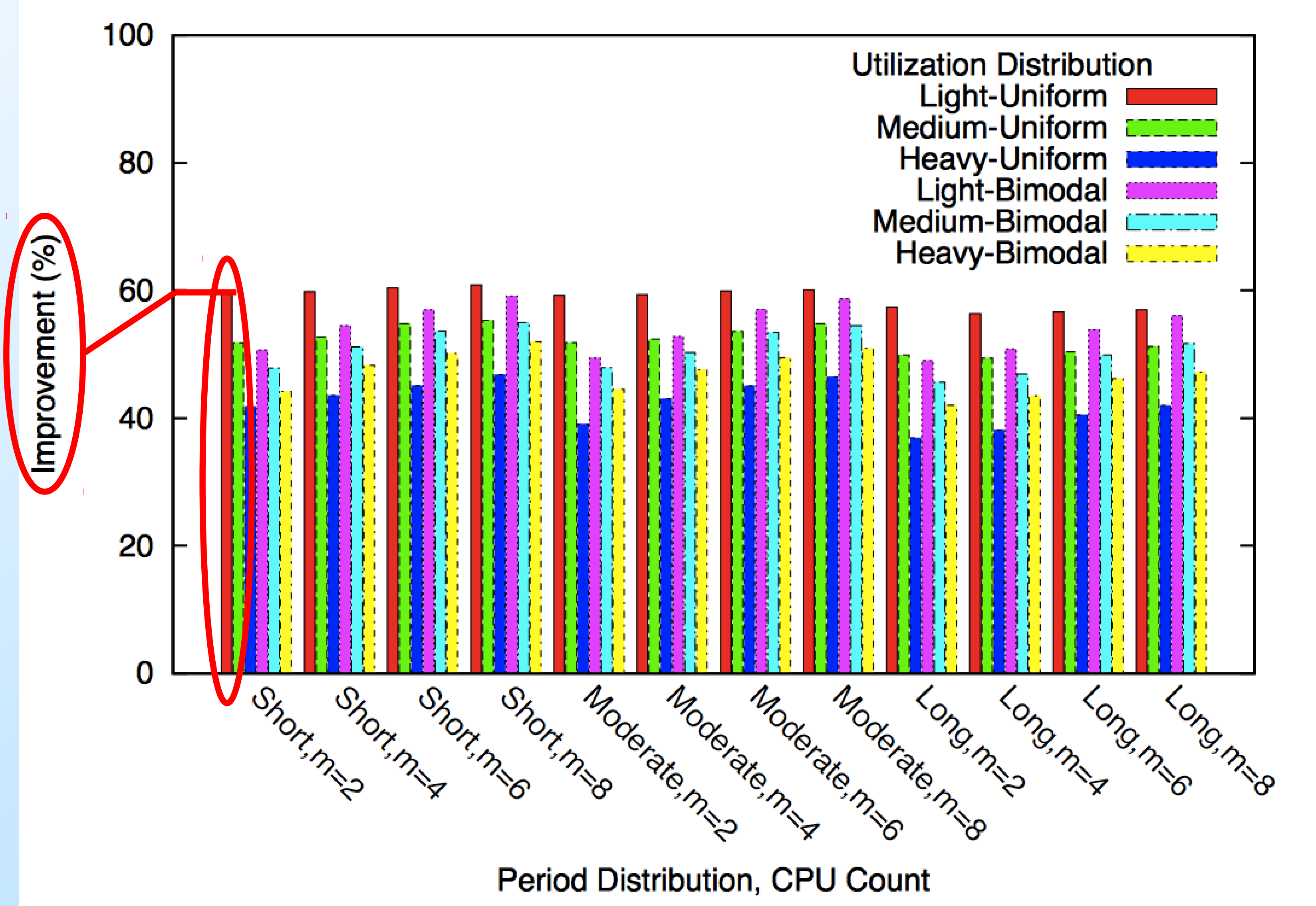
Experiments - Bounds

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



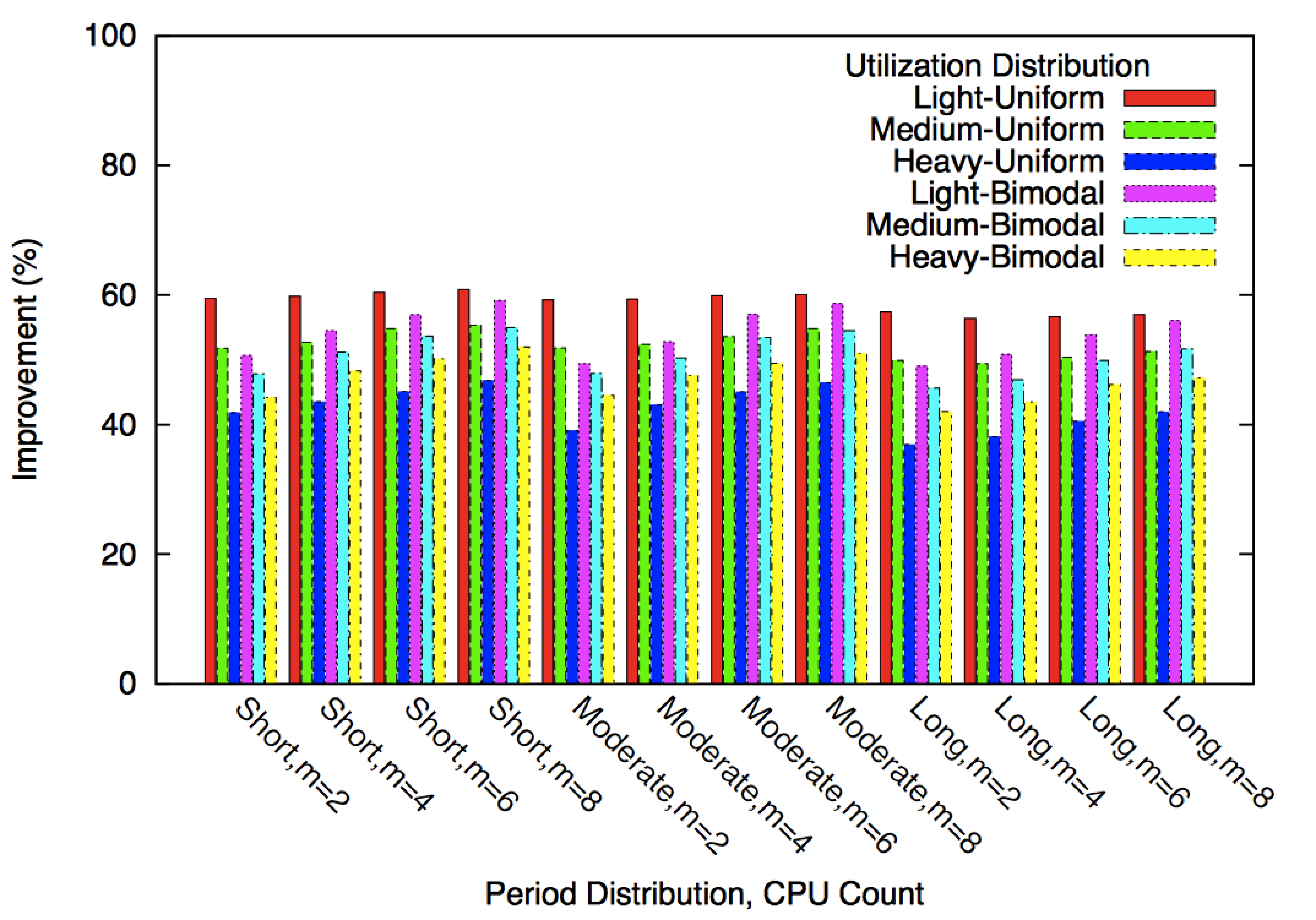
Experiments - Bounds

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

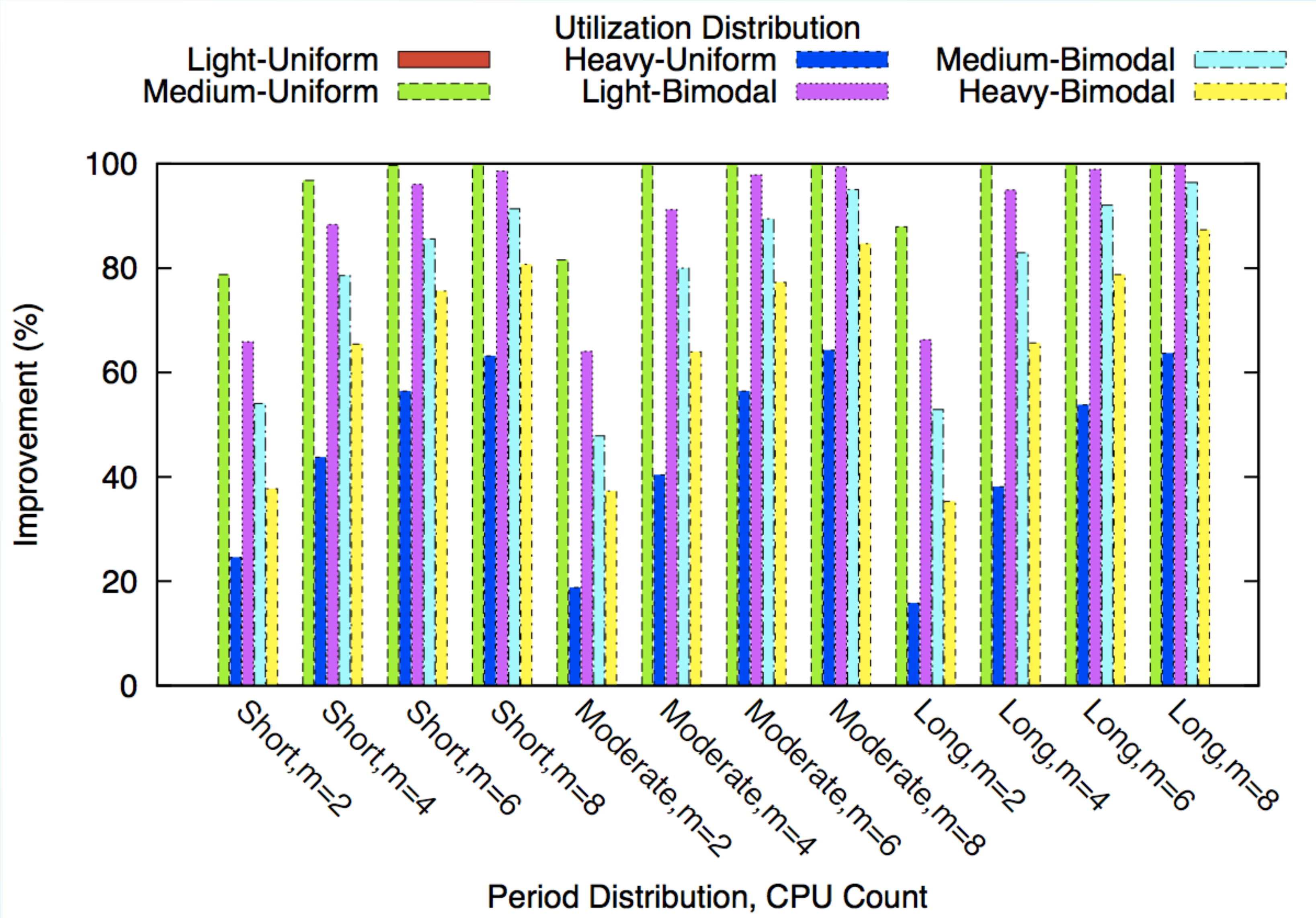


Experiments - Bounds

THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL



Experiments – Computed Schedules



Conclusion

- Tardiness bounds can be reduced by about 50% by switching from G-EDF to G-FL.
- Actual tardiness also likely to be lower.
- Remember: implementation still like G-EDF!

Future Work

- HRT scheduling efficiency. (Related: see Back, Chwa, and Shin, RTAS 2012)
- Other notions of “fair lateness” - e.g. same percentage of period length instead of absolute lateness.

Questions?

Thank You!