csci 3411: Operating Systems

Real-Time CPU Scheduling

Gabriel Parmer

Slides evolved from Silberschatz and West

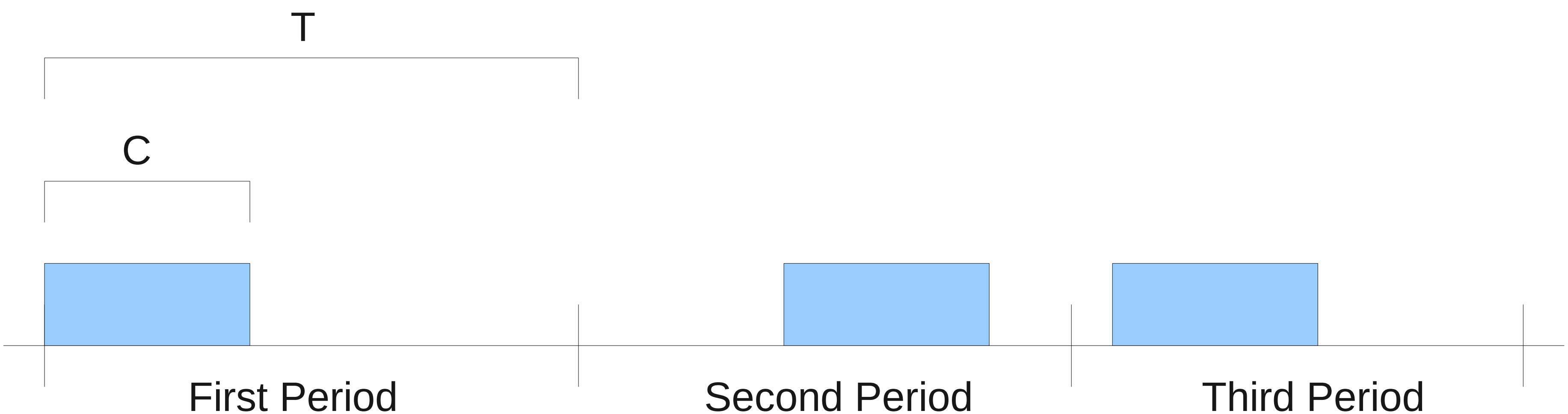
Real-Time Scheduling

- System needs to meet timeliness constraints
  - System interacts with the “real” world and “real” time
    - Anti-lock brakes, flight control, etc...
  - Tasks can have *deadlines*
  - *Predictable* task execution

Real-Time Scheduling II

- Earliest Deadline First (EDF)
  - Dynamic priority algorithm
- Rate Monotonic Scheduling (RM)
  - Static priority algorithm

A Real Time Task Model

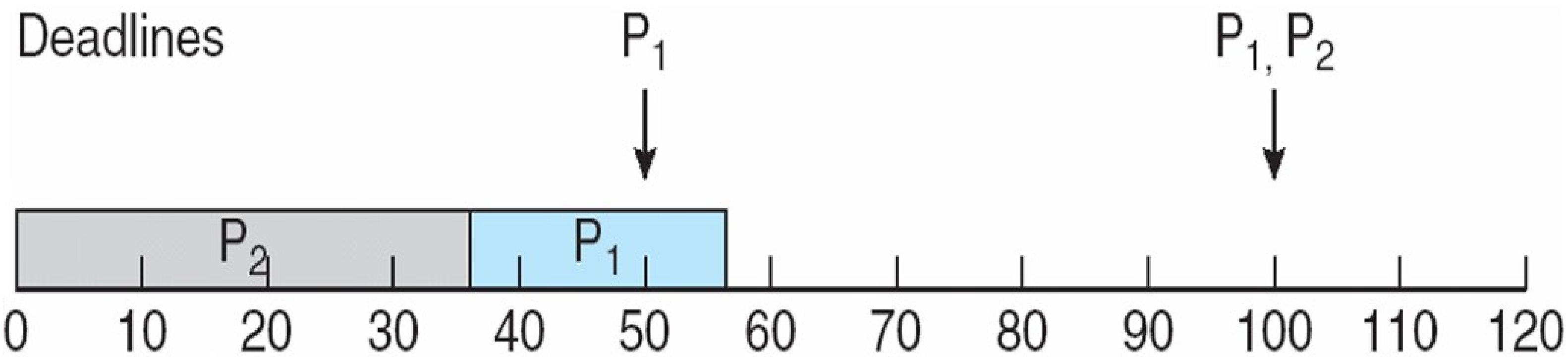


- Each task has a
  - Maximum (worst-case) execution time: C
  - Period: T
  - Deadline: D (we'll assume D == T)
- What is a task's CPU utilization?

RT System Scheduling Criteria

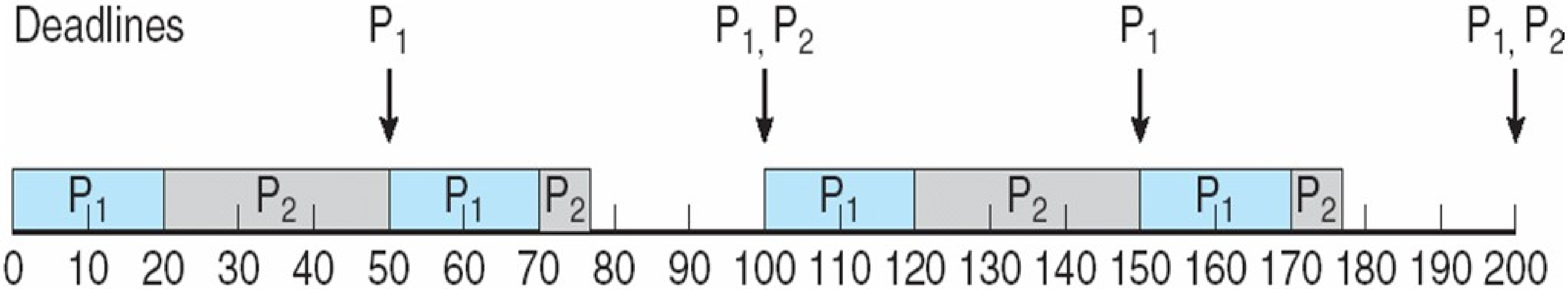
- Meet task deadlines!
  - The *schedulability* of a task set
- *Lateness* – difference between completion time and deadline of a task
  - *Late* if lateness is positive, *early* otherwise
- *Tardiness* – max(0, *lateness*)
  - How *late* is a task
- Why are these useful measures?

Missing Deadlines



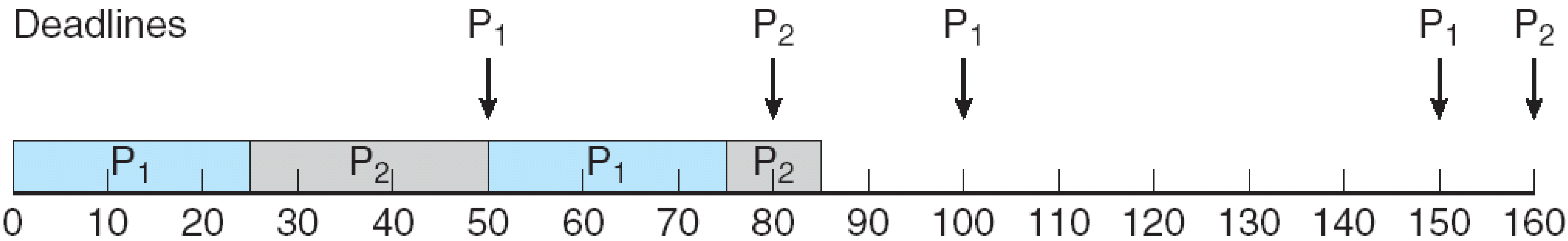
- P_1's T = 50, C = 20, low priority
- P_2's T = 100, C = 36, high priority
- Should be able to meet both deadlines
  - Why aren't we, and what can we do?

Rate Monotonic Scheduling



- Static (Fixed) Priority Preemptive Scheduling
  - Main question: how do we assign priorities to tasks?
- Task's priority inversely related to period length
  - Smaller T = higher priority and vice-versa

RM Schedulability



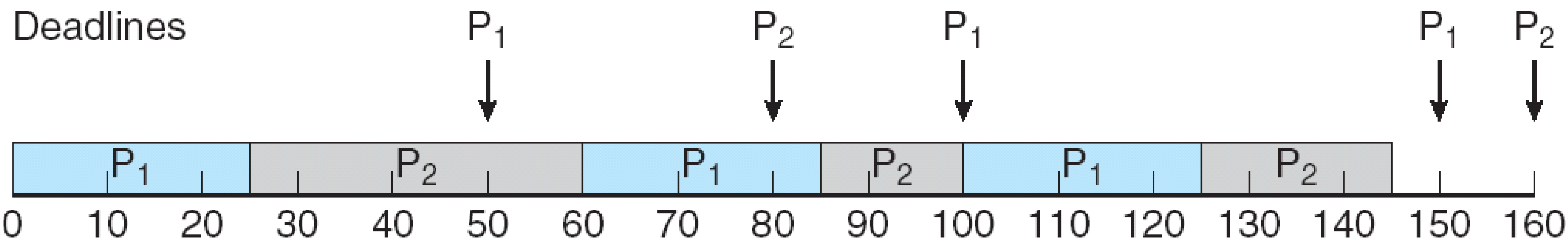
- Does not always work: Can still miss deadlines
  - When does it fail?
- Schedulability test (tasks can be scheduled if):
  - $\Sigma_{i=1...n} C_i/T_i \leq n(2^{1/n} - 1)$
  - $\text{Limit}_{n\rightarrow\text{infinity}} \Sigma_{i=1...n} C_i/T_i = \log_e 2 = 69\%$

RM Schedulability II

- Scheduling test
  - *Sufficient*, but not *necessary*
    - Passing test → will work, not passing → *might* work
  - Task sets with a higher utilization *might* still work!
  - Is there a *necessary*, but not *sufficient* test?
    - Passing test → *might* work, not passing → will *not* work

- When execute schedulability test?
  - Admission control

Earliest Deadline First (EDF)



- Priority of a task at time *t* inversely related to distance to deadline
  - Dynamic priorities
- Minimize maximum lateness (thus tardiness)

Earliest Deadline First (EDF)

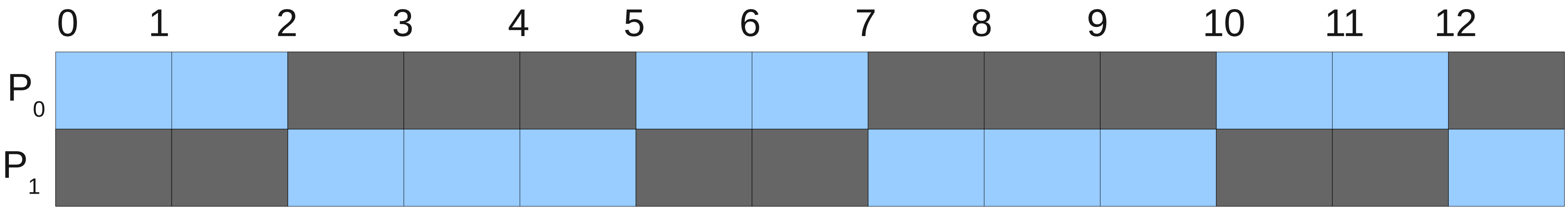
- If all deadlines can be met using *some ordering of tasks*, EDF will guarantee to meet all deadlines
  - $\Sigma_{i=1\ldots n} C_i/T_i \leq 1$
  - Necessary *and* sufficient: exact

- Fantastic, we're done! Lets all go home!
  - Not quite: what happens with EDF in overload?
  - Implementation costs?

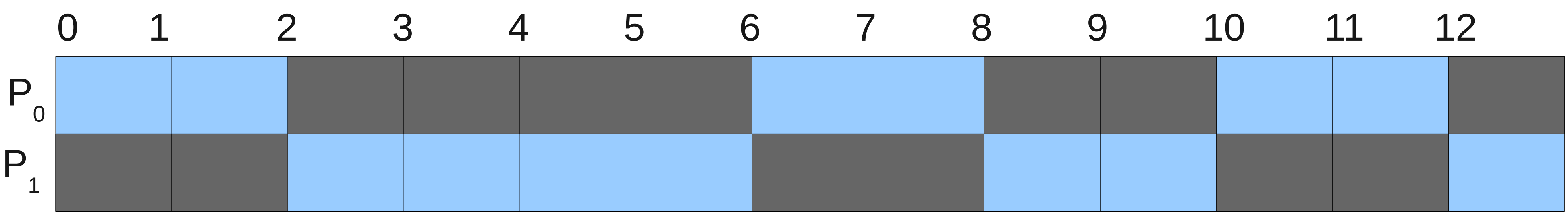
P_0 <C = 4, T = 8>

P_1 <C = 2, T = 5>

RM:



EDF:



RT Scheduling Recap

- RM
  - Simple policy
  - Schedulability test
    - response time analysis → exact
  - Behavior in overload?
- EDF
  - More complex policy
  - Exact schedulability test
  - Overload situation