Graceful Degradation in Semi-Clairvoyant Scheduling

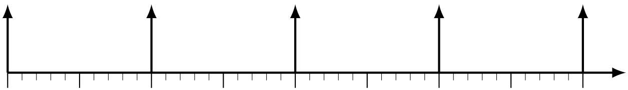
Sanjoy Baruah
Washington University in Saint Louis

Pontus Ekberg
Uppsala University

ECRTS 2021

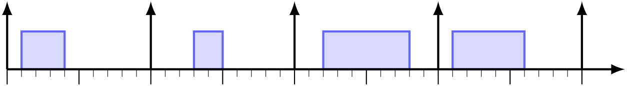
Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)



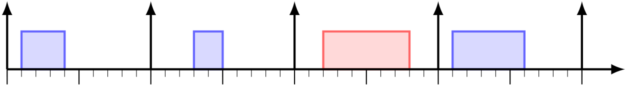
Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)



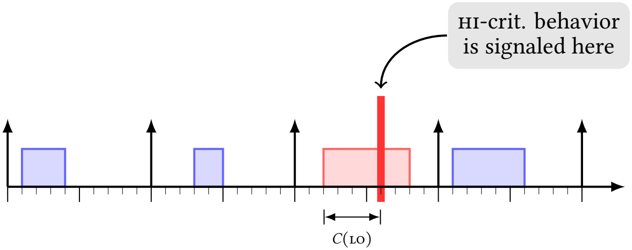
Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)



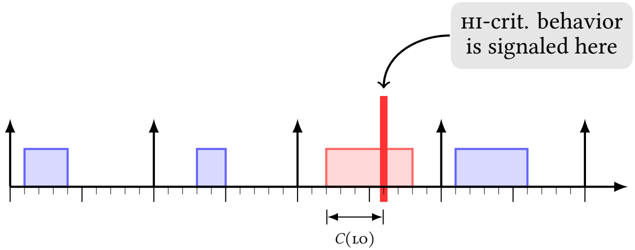
Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)

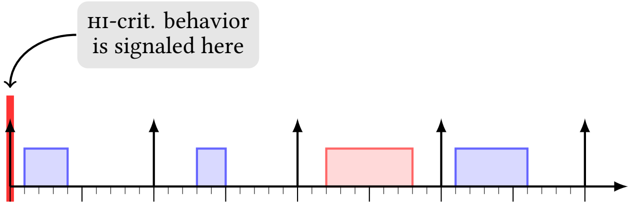


Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)

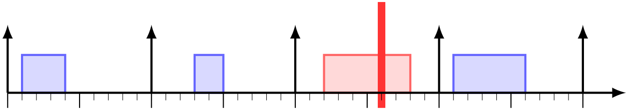


Clairvoyant:

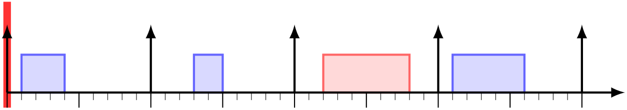


Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)

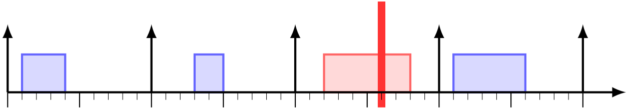


Clairvoyant:

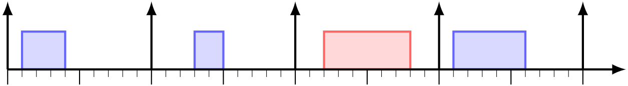


Semi-clairvoyant mixed-criticality scheduling

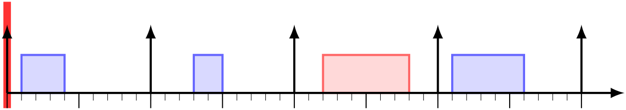
Non-clairvoyant:
(ordinary MC)



Semi-clairvoyant:
(Agrawal et al., RTSS'19)



Clairvoyant:

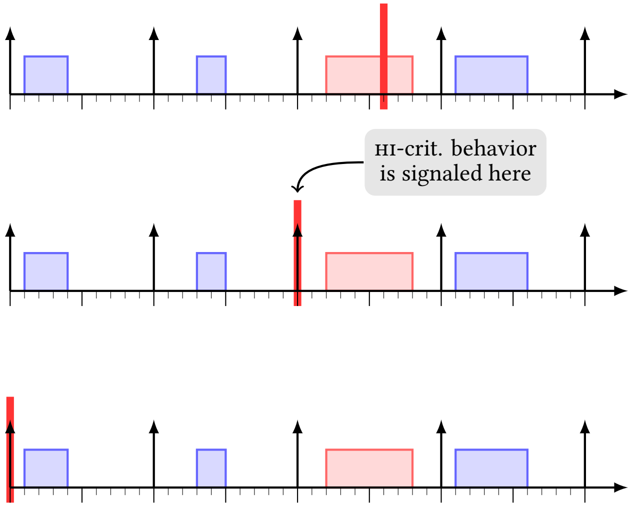


Semi-clairvoyant mixed-criticality scheduling

Non-clairvoyant:
(ordinary MC)

Semi-clairvoyant:
(Agrawal et al., RTSS'19)

Clairvoyant:



Graceful degradation – what we mean

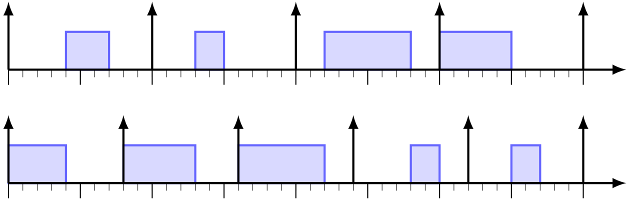
lo-criticality jobs are allowed to execute also after hi-criticality behavior is signaled, but with *reduced* WCET.

Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.

HI-criticality:
$C(\text{LO}) \leq C(\text{HI})$

LO-criticality:
$C(\text{HI}) \leq C(\text{LO})$

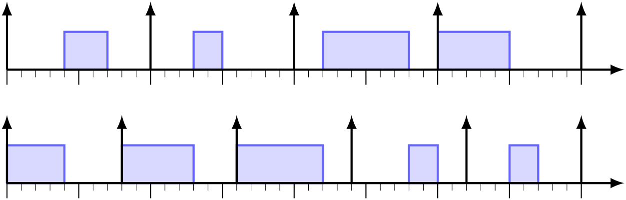


Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.

HI-criticality:
$C(\textsc{lo}) \le C(\textsc{hi})$

LO-criticality:
$C(\textsc{hi}) \le C(\textsc{lo})$

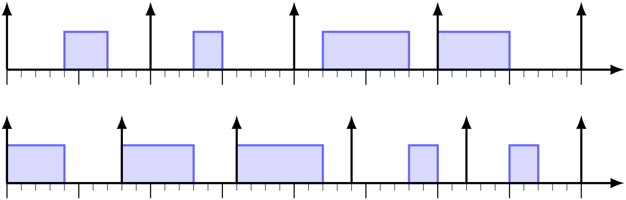


Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.

HI-criticality:
$C(\text{LO}) \leq C(\text{HI})$

LO-criticality:
$C(\text{HI}) \leq C(\text{LO})$

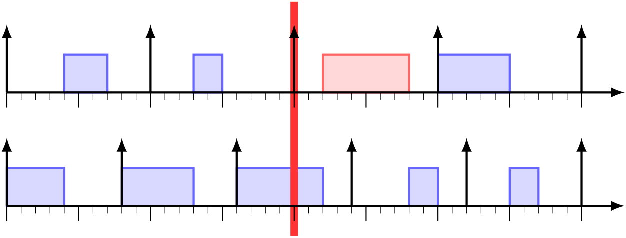


Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.

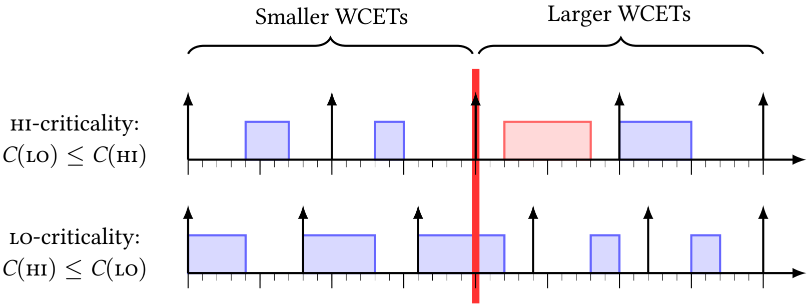
HI-criticality: $C(\text{LO}) \leq C(\text{HI})$

LO-criticality: $C(\text{HI}) \leq C(\text{LO})$



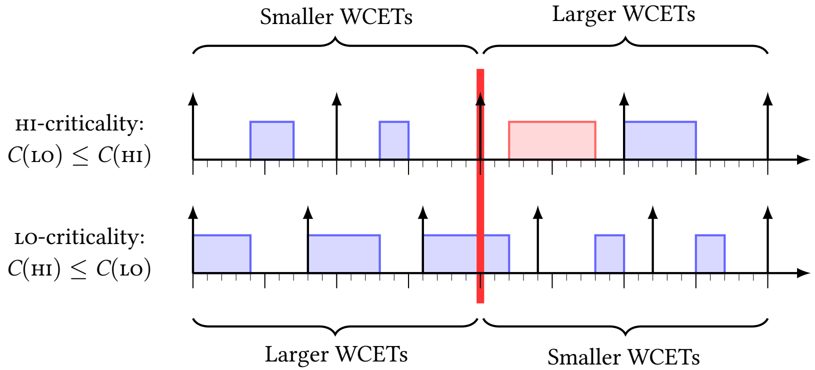
Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.



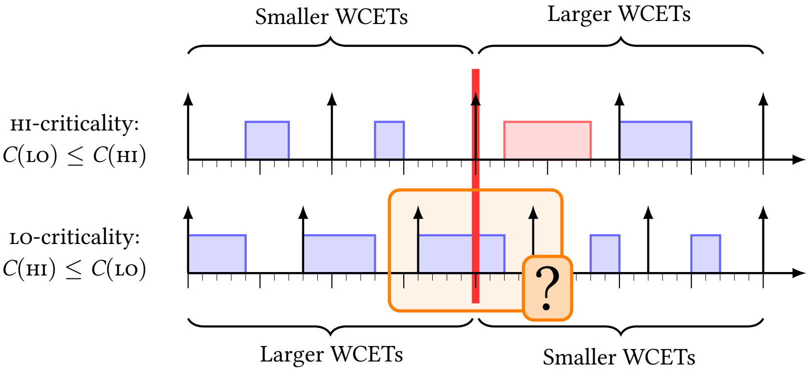
Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.



Graceful degradation – what we mean

LO-criticality jobs are allowed to execute also after HI-criticality behavior is signaled, but with *reduced* WCET.



Three different correctness criteria

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

Three different correctness criteria

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

CC-1 reduces to the *standard mixed-criticality semantics* in the absence of graceful degradation (i.e., if $C(\text{HI}) = 0$).

Three different correctness criteria

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

CC-2

Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.

Three different correctness criteria

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

CC-2

Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.

CC-3

All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.

Three different correctness criteria

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

CC-2

Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.

CC-3

All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.

$$\text{CC-3} \implies \text{CC-2} \implies \text{CC-1}$$

All the problems

We consider *optimal scheduling* and *exact analysis* in different settings:

- With one of three correctness criteria:
  1. CC-1
  2. CC-2
  3. CC-3

- With one of two workload models:
  1. Independent jobs
  2. Sporadic tasks

All the problems

We consider *optimal scheduling* and *exact analysis* in different settings:

- With one of three correctness criteria:
  1. CC-1
  2. CC-2
  3. CC-3
- With one of two workload models:
  1. Independent jobs
  2. Sporadic tasks
- Semi-clairvoyant scheduling
- Graceful degradation
- A preemptive uniprocessor

Correctness criterion CC-1 — insights

CC-1

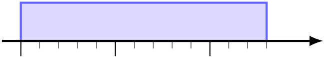
No active lo-crit. job gets to keep their $C(\text{LO})$ budgets.

Correctness criterion CC-1 — insights

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

LO-crit. job:

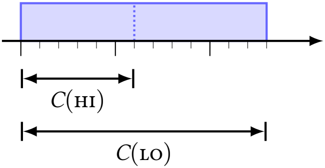


Correctness criterion CC-1 — insights

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.

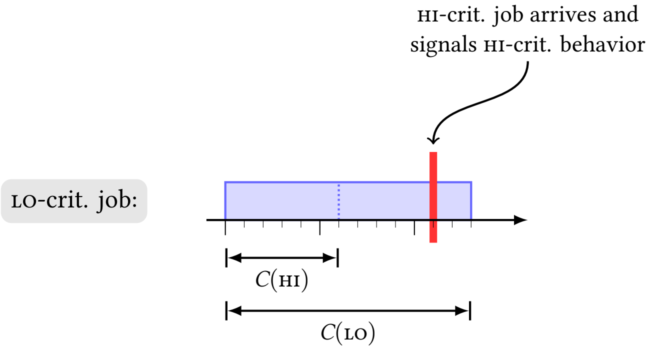
LO-crit. job:



Correctness criterion CC-1 — insights

CC-1

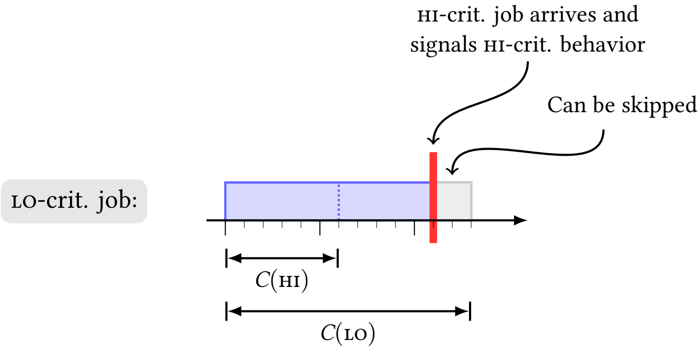
No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-1 — insights

CC-1

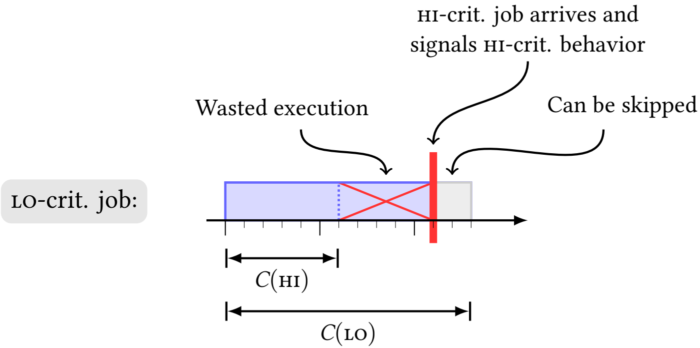
No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-1 — insights

CC-1

No active LO-crit. job gets to keep their $C(\text{LO})$ budgets.



CC-1 — RESULTS

Prior work

CC-1 without graceful degradation (i.e., $C(\textsc{hi}) = 0$ for LO-crit. jobs) is the setting in Agrawal et al., RTSS'19.

CC-1 — RESULTS

Prior work

CC-1 without graceful degradation (i.e., $C(\textsc{hi}) = 0$ for LO-crit. jobs) is the setting in Agrawal et al., RTSS'19.

We extend the same results to work *with graceful degradation*.

CC-1 — RESULTS

Prior work

CC-1 without graceful degradation (i.e., $C(\textsc{hi}) = 0$ for LO-crit. jobs) is the setting in Agrawal et al., RTSS'19.

We extend the same results to work *with graceful degradation*.

1. A polynomial-time solution for jobs:
   - A Linear Program (LP) for exactly solving the feasibility problem.
   - A table-based optimal scheduler extracted from the LP solution.

CC-1 — RESULTS

Prior work

CC-1 without graceful degradation (i.e., $C(\textsc{hi}) = 0$ for LO-crit. jobs) is the setting in Agrawal et al., RTSS'19.

We extend the same results to work *with graceful degradation*.

1. A polynomial-time solution for jobs:
   - A Linear Program (LP) for exactly solving the feasibility problem.
   - A table-based optimal scheduler extracted from the LP solution.
2. A polynomial-time solution for implicit-deadline sporadic tasks.
   - An exact utilization-based feasibility test.
   - A fluid-based optimal scheduler.
   - (This followed directly from Agrawal et al.)

Correctness criterion CC-2 — insights

CC-2

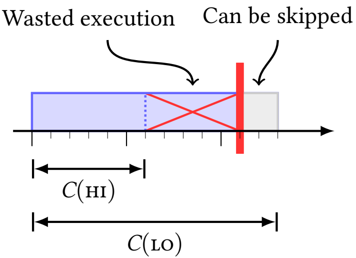
Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.

Correctness criterion CC-2 — insights

CC-2

Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.

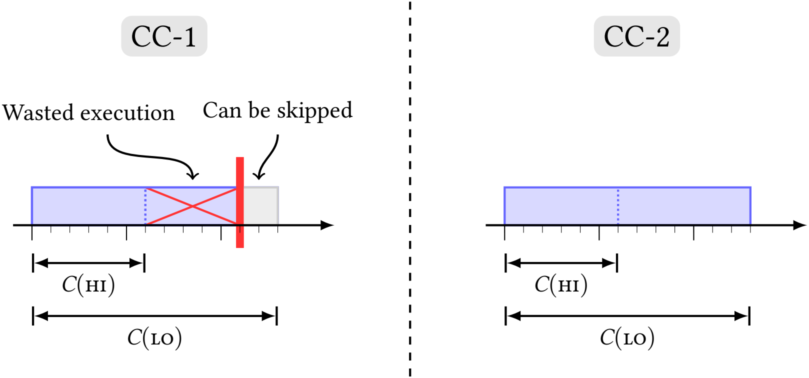
CC-1



Correctness criterion CC-2 — insights

CC-2

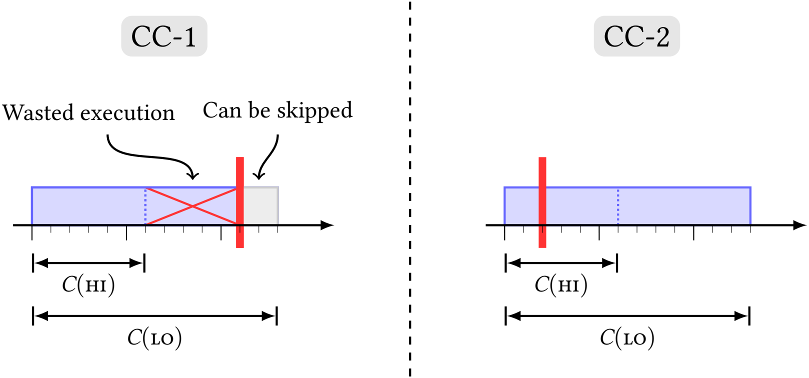
Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-2 — insights

CC-2

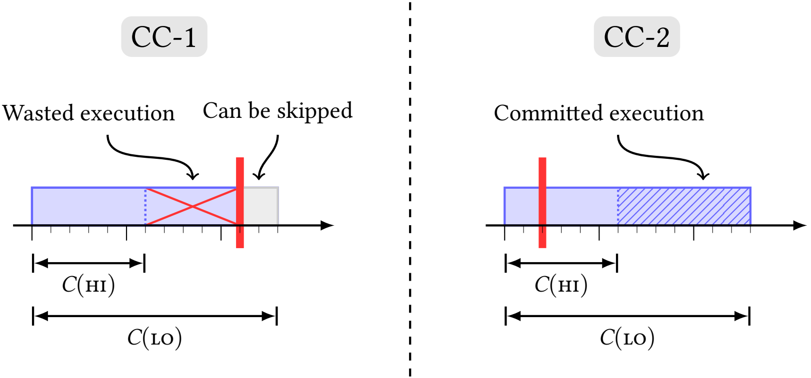
Active LO-crit. jobs that have *started execution* get to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-2 — insights

CC-2

Active LO-crit. jobs that have *started execution* get to keep their $C(\mathrm{LO})$ budgets.



CC-2 — RESULTS

1. Feasibility for jobs under CC-2 can be solved exactly with an *Mixed Integer Linear Program* (MILP).

CC-2 — RESULTS

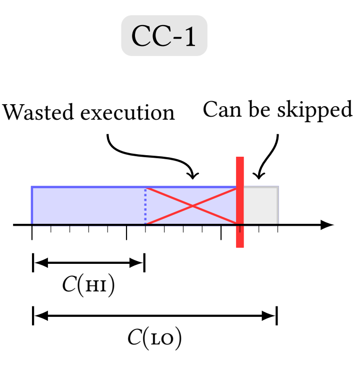
1. Feasibility for jobs under CC-2 can be solved exactly with an *Mixed Integer Linear Program* (MILP).
2. A *table-based optimal scheduler* can again be extracted.

CC-2 — RESULTS

1. Feasibility for jobs under CC-2 can be solved exactly with an *Mixed Integer Linear Program* (MILP).
2. A *table-based optimal scheduler* can again be extracted.
3. Feasibility for jobs under CC-2 is *strongly NP-complete*.

CC-2 — RESULTS

1. Feasibility for jobs under CC-2 can be solved exactly with an *Mixed Integer Linear Program* (MILP).
2. A *table-based optimal scheduler* can again be extracted.
3. Feasibility for jobs under CC-2 is *strongly NP-complete*.



Correctness criterion CC-3 — insights

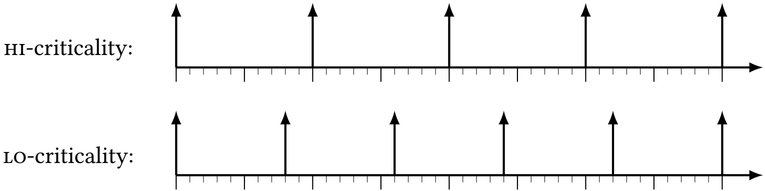
CC-3

All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.

Correctness criterion CC-3 — insights

CC-3

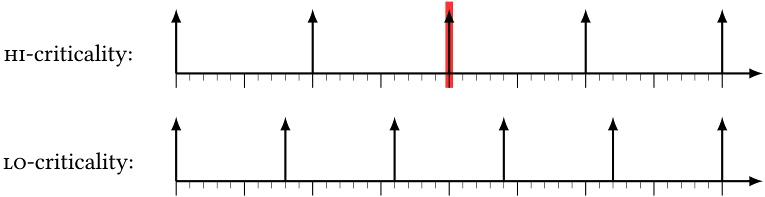
All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-3 — insights

CC-3

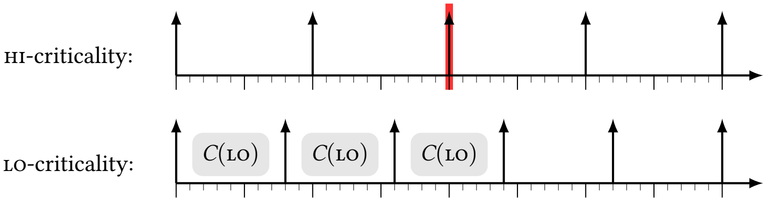
All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-3 — insights

CC-3

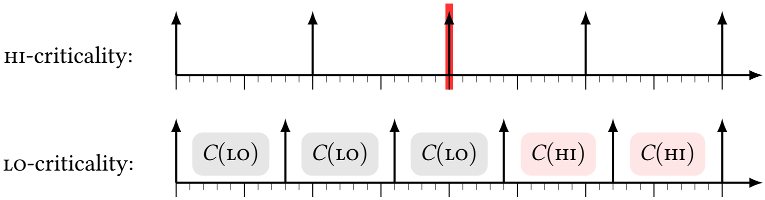
All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.



Correctness criterion CC-3 — insights

CC-3

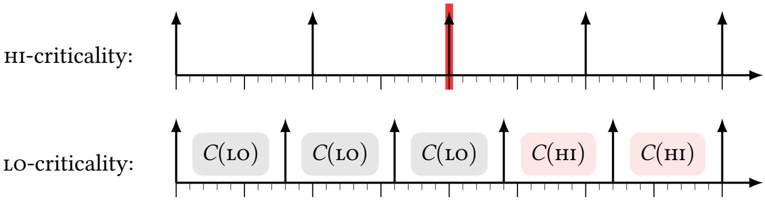
All active lo-crit. jobs get to keep their $C(\text{lo})$ budgets.



Correctness criterion CC-3 — insights

CC-3

All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.



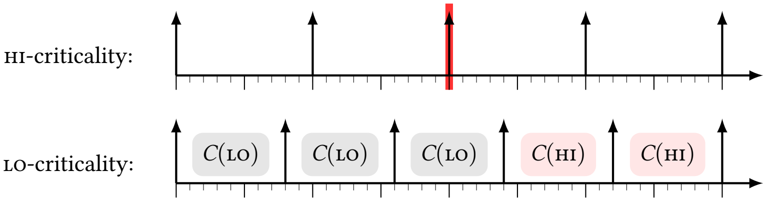
Observation

Under CC-3, the amount of execution time spent on each job is *independent* on the scheduling decisions.

Correctness criterion CC-3 — insights

CC-3

All active LO-crit. jobs get to keep their $C(\text{LO})$ budgets.



Observation

Under CC-3, the amount of execution time spent on each job is *independent* on the scheduling decisions.

$\Longrightarrow$ *EDF is an optimal scheduler* for both jobs and tasks!

CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

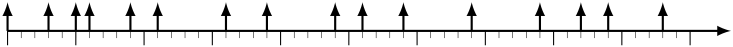
(Simply simulate EDF $O(n)$ times.)

CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

(Simply simulate EDF $O(n)$ times.)

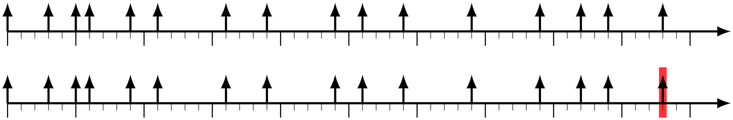


CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

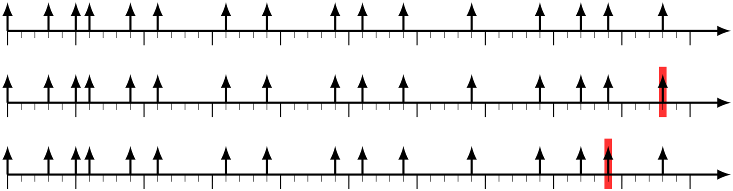
(Simply simulate EDF $O(n)$ times.)



Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

(Simply simulate EDF $O(n)$ times.)

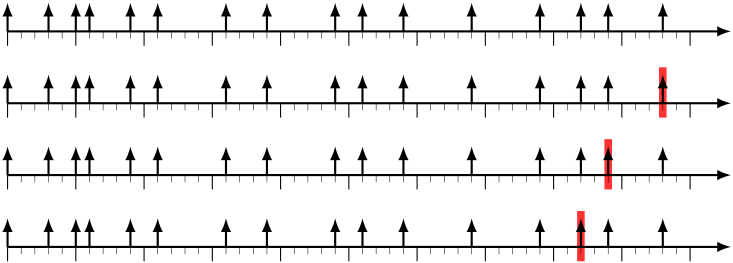


CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

(Simply simulate EDF $O(n)$ times.)

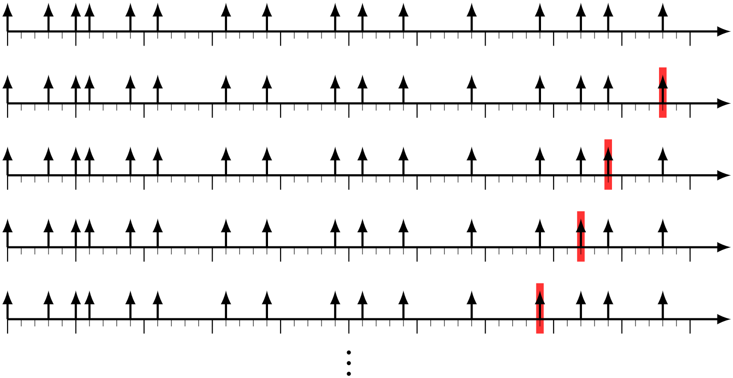


CC-3 — RESULTS FOR JOBS

Jobs

Feasibility for *jobs* can be done in $O(n^2 \log n)$ time.

(Simply simulate EDF $O(n)$ times.)



CC-3 — results for tasks

Tasks

Feasibility for *arbitrary-deadline sporadic tasks* can be done in pseudo-polynomial time if U is bounded by $c < 1$.

CC-3 — RESULTS FOR TASKS

Tasks

Feasibility for *arbitrary-deadline sporadic tasks* can be done in pseudo-polynomial time if U is bounded by $c < 1$.

(Based on *dbf* analysis.)

Which correctness criteria is the correct one?

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs			
Sporadic tasks			

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs	Polynomial-time solvable (LP formulation) Table-based optimal scheduler		
Sporadic tasks	Simple utilization test Optimal fluid scheduler (Only implicit deadlines)		

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs	Polynomial-time solvable (LP formulation) Table-based optimal scheduler	Strongly NP-complete (MILP formulation) Table-based optimal scheduler	
Sporadic tasks	Simple utilization test Optimal fluid scheduler (Only implicit deadlines)		

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs	Polynomial-time solvable (LP formulation) Table-based optimal scheduler	Strongly NP-complete (MILP formulation) Table-based optimal scheduler	Solvable in $O(n^2 \log n)$ EDF optimal
Sporadic tasks	Simple utilization test Optimal fluid scheduler (Only implicit deadlines)		Pseudo-poly. time solvable with bounded utilization EDF optimal (Even arbitrary deadlines)

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs	Polynomial-time solvable (LP formulation) Table-based optimal scheduler	Strongly NP-complete (MILP formulation) Table-based optimal scheduler	Solvable in $O(n^2 \log n)$ EDF optimal
Sporadic tasks	Simple utilization test Optimal fluid scheduler (Only implicit deadlines)		Pseudo-poly. time solvable with bounded utilization EDF optimal (Even arbitrary deadlines)

Which correctness criteria is the correct one?

	CC-1	CC-2	CC-3
Jobs	Polynomial-time solvable (LP formulation) Table-based optimal scheduler	Strongly NP-complete (MILP formulation) Table-based optimal scheduler	Solvable in $O(n^2 \log n)$ EDF optimal
Sporadic tasks	Simple utilization test Optimal fluid scheduler (Only implicit deadlines)		Pseudo-poly. time solvable with bounded utilization EDF optimal (Even arbitrary deadlines)

$$\text{CC-3} \Longrightarrow \text{CC-2} \Longrightarrow \text{CC-1}$$

What is the modeling overhead?

The modeling overhead of CC-3

Over-approximating either CC-1 or CC-2 by CC-3 has a worst-case speedup cost of 2 (which is tight).

What is the modeling overhead?

The modeling overhead of CC-3

Over-approximating either CC-1 or CC-2 by CC-3 has a worst-case speedup cost of 2 (which is tight).

The modeling overhead of CC-2

Over-approximating CC-1 by CC-2 has a worst-case speedup cost in $[\varphi, 2]$. ($\varphi \approx 1.618$ is the golden ratio.)

$\forall$Thank you!

$\Diamond$

$\exists$Questions?