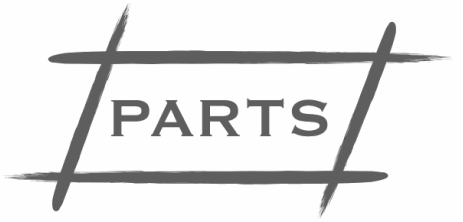
U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

Geoffrey Nelissen, Vandy Berten, Vincent Nélis,
Joël Goossens, Dragomir Milojevic



Research Centre in
Real-Time Computing Systems

Create a New Scheduling Algorithm What for?

Theoretical considerations

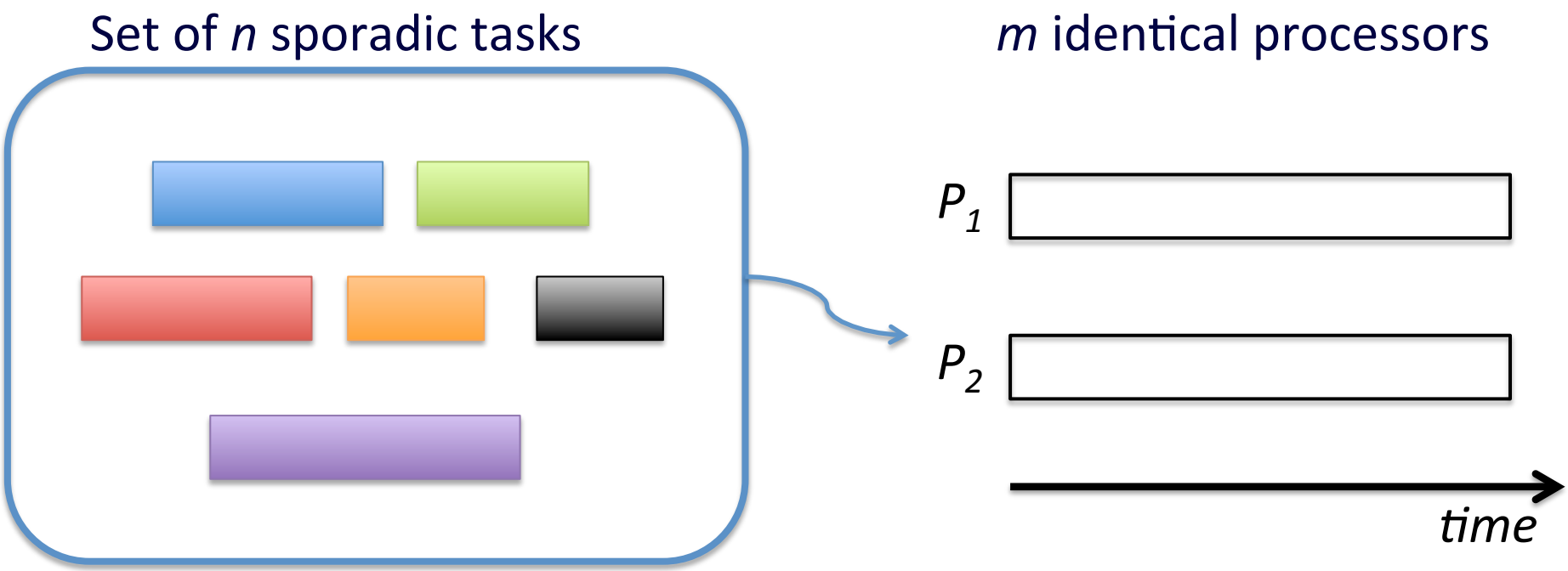
- Schedule as many task sets as possible

➔ Optimality

Practical considerations

- Run-Time Complexity
- Implementation
- # preemptions
- # migrations

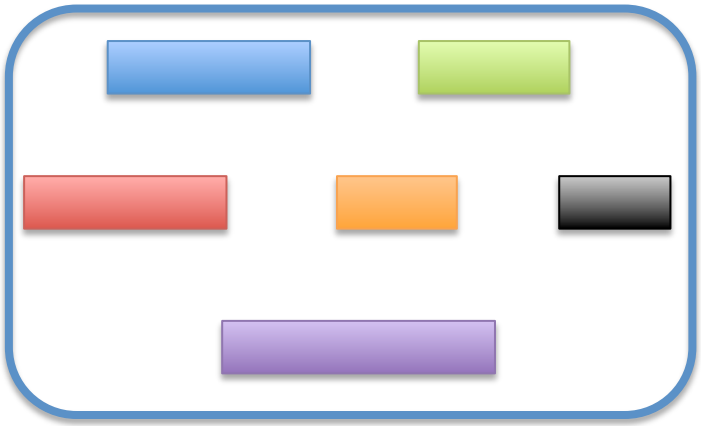
The Scheduling Problem



- Worst case execution time: C_i
- Minimum inter-arrival time: T_i
- Utilization: $U_i = C_i/T_i$

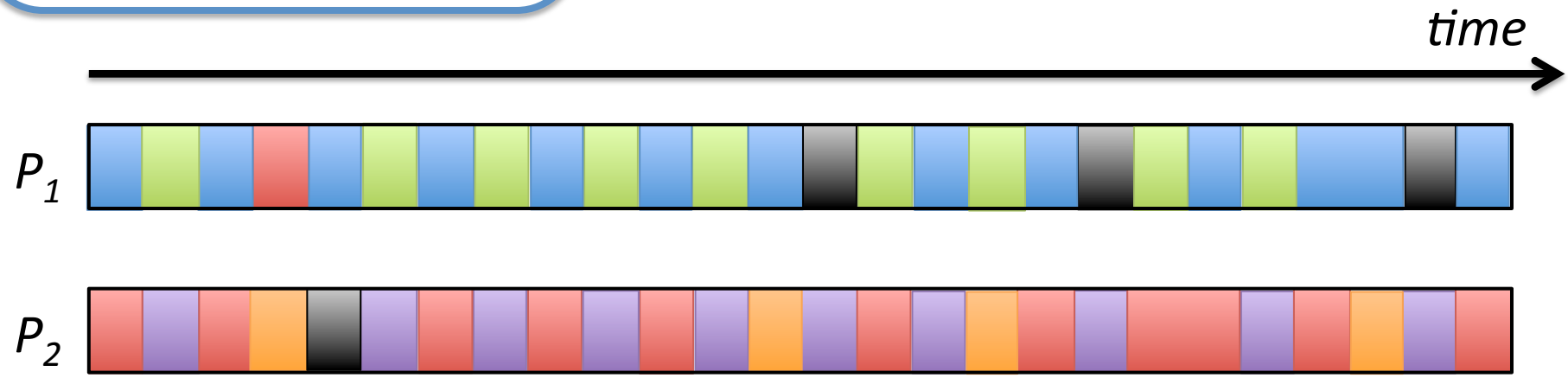
- Scheduling any task set such that $\sum_i U_i \leq m$

Review of Existing Solutions: Pfair Algorithms

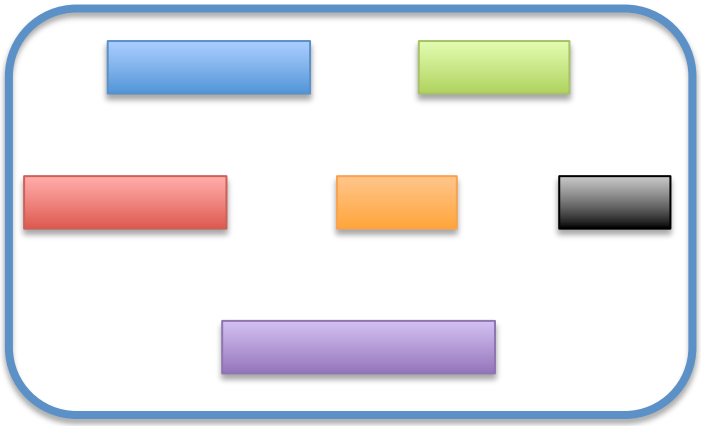


Fairness:

- for **all** tasks
- at **any** instant *t*

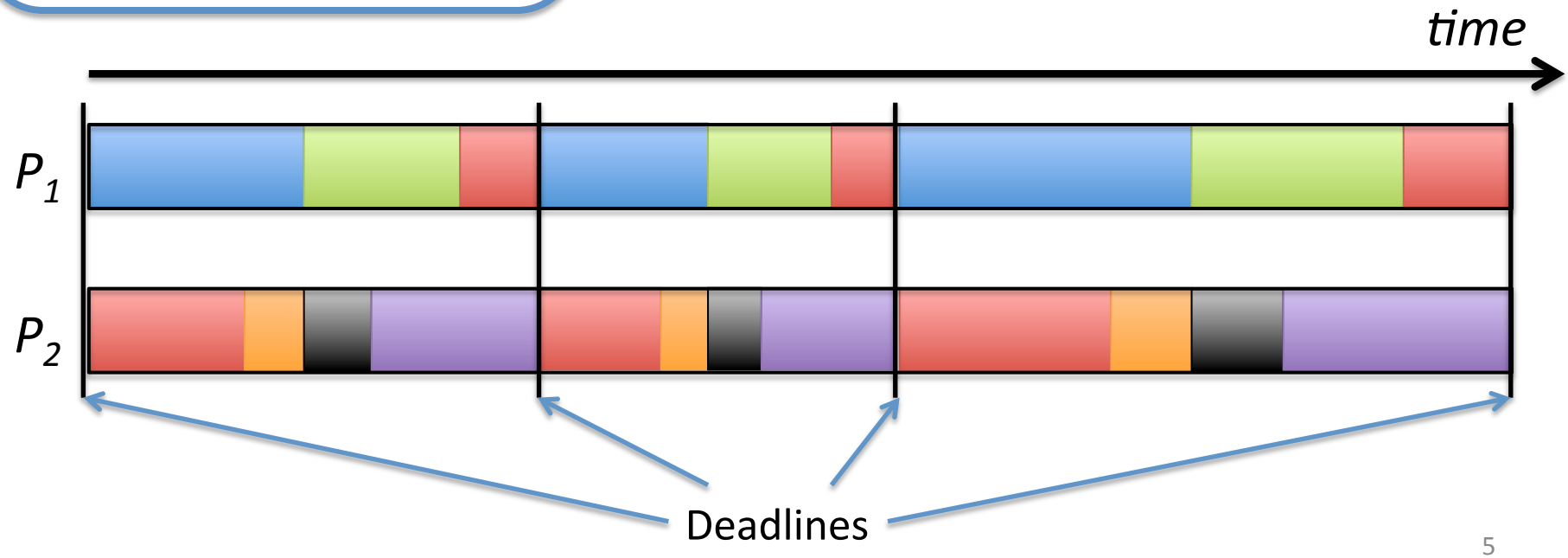


Review of Existing Solutions: Boundary-Fair Algorithms

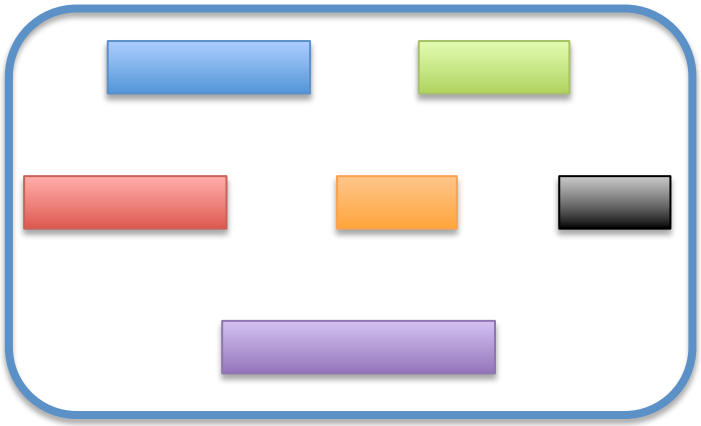


Fairness:

- for **all** tasks
- only at job **deadlines**

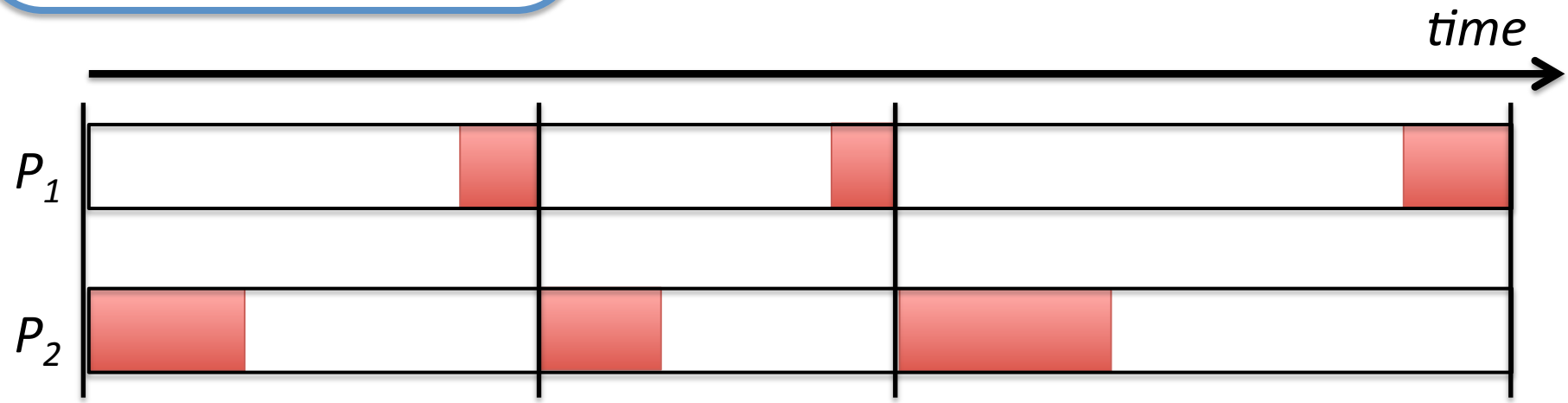


Review of Existing Solutions: EKG Algorithm

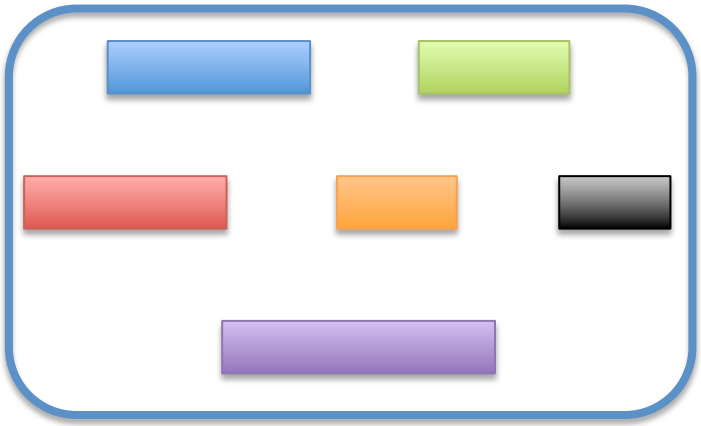


Fairness:

- only for **migrating** tasks
- only at job **deadlines**

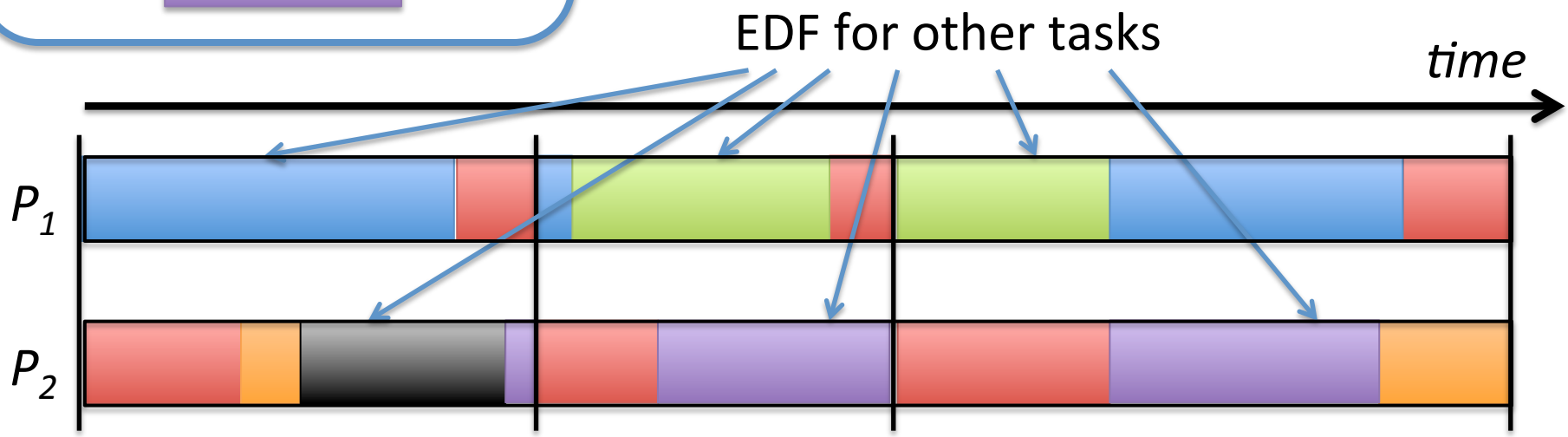


Review of Existing Solutions: EKG Algorithm

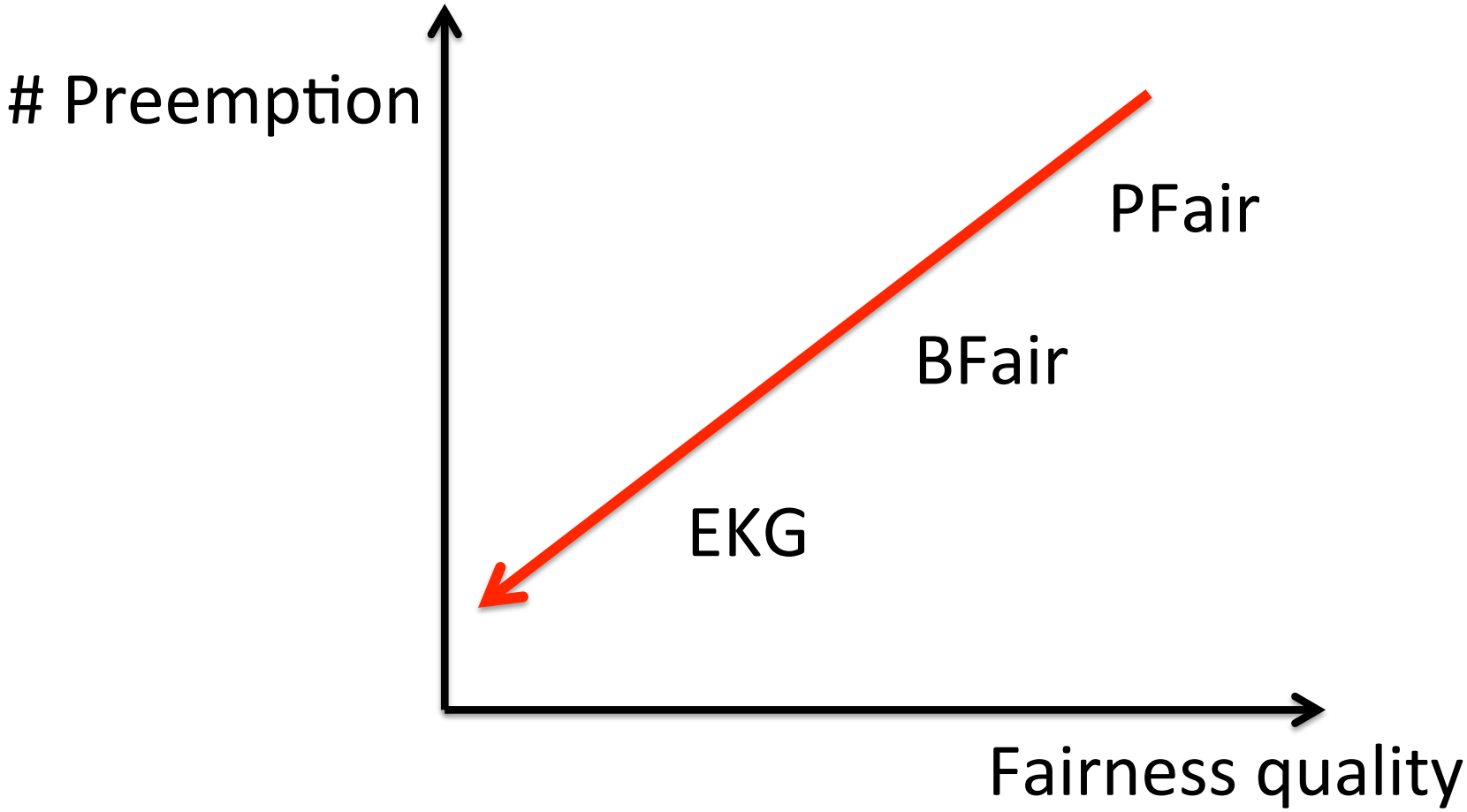


Fairness:

- only for **migrating** tasks
- only at job **deadlines**



First Conclusion: less fairness = less preemptions



The next step

More than one year ago, we claimed

No fairness is needed to reach the optimality

An unfair algorithm will induce really few preemptions

We had evidences but no formal proof!

8 months ago (RTSS 2011), RUN validated our claims for ***periodic*** tasks

The next step

More than one year ago, we claimed

No fairness is needed to reach the optimality

An unfair algorithm will induce really few preemptions

Our goal:

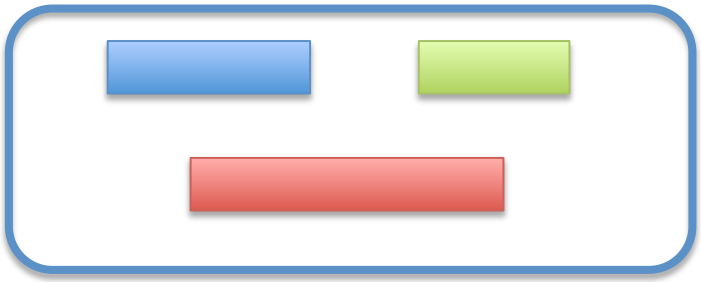
Propose an **unfair optimal** algorithm for *****sporadic***** tasks

Starting point: EDF

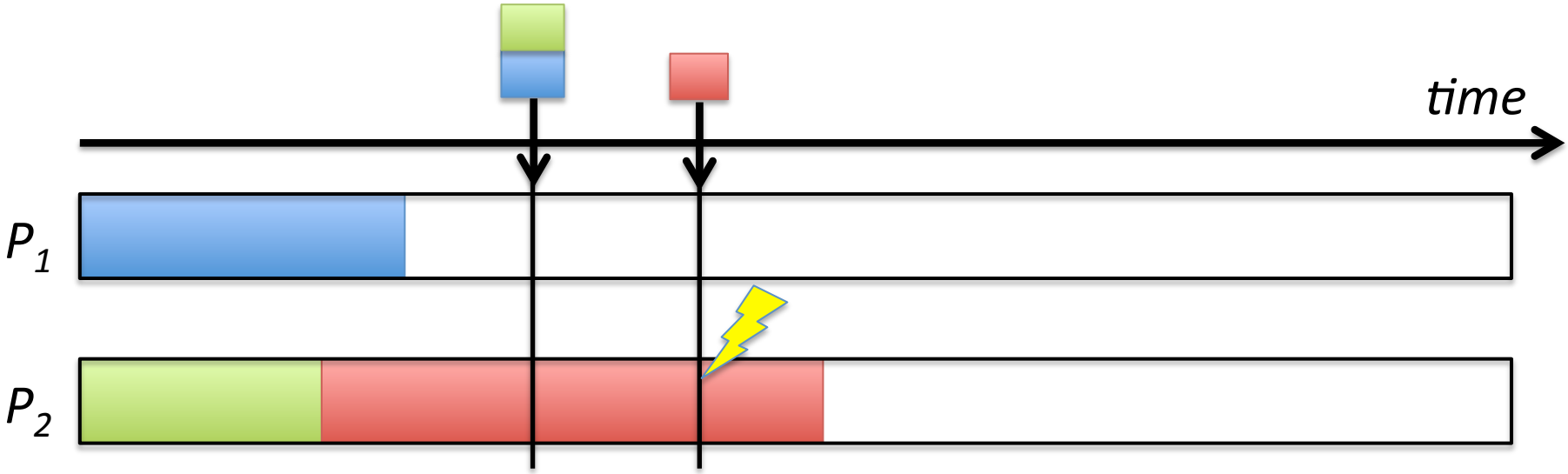
- Optimal on uniprocessor
- Few preemptions
- Simple

- Could it be exended to multiprocessor while keeping its advantages?

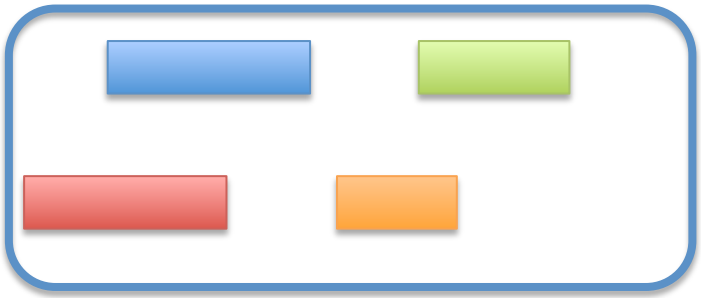
A vertical generalization of EDF: Global EDF



On multiprocessor,
G-EDF is **not** optimal

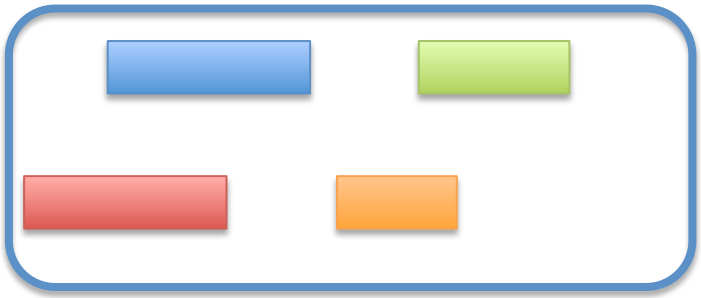


Studying EDF

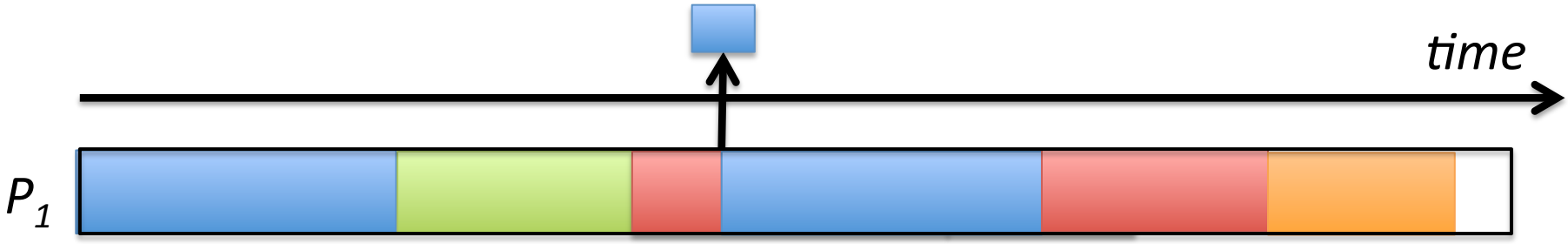


On uniprocessor, EDF **horizontally** assign jobs

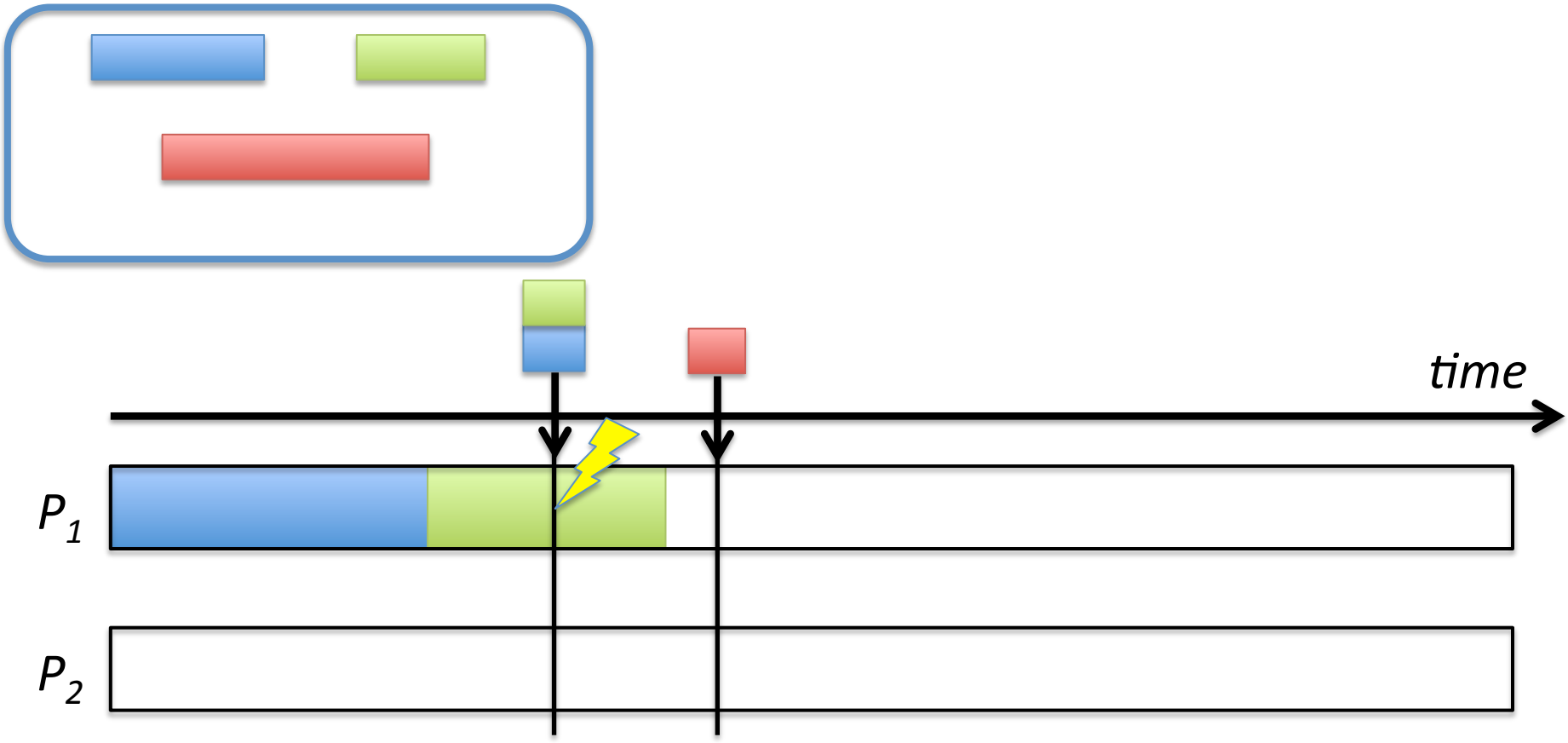
Studying EDF



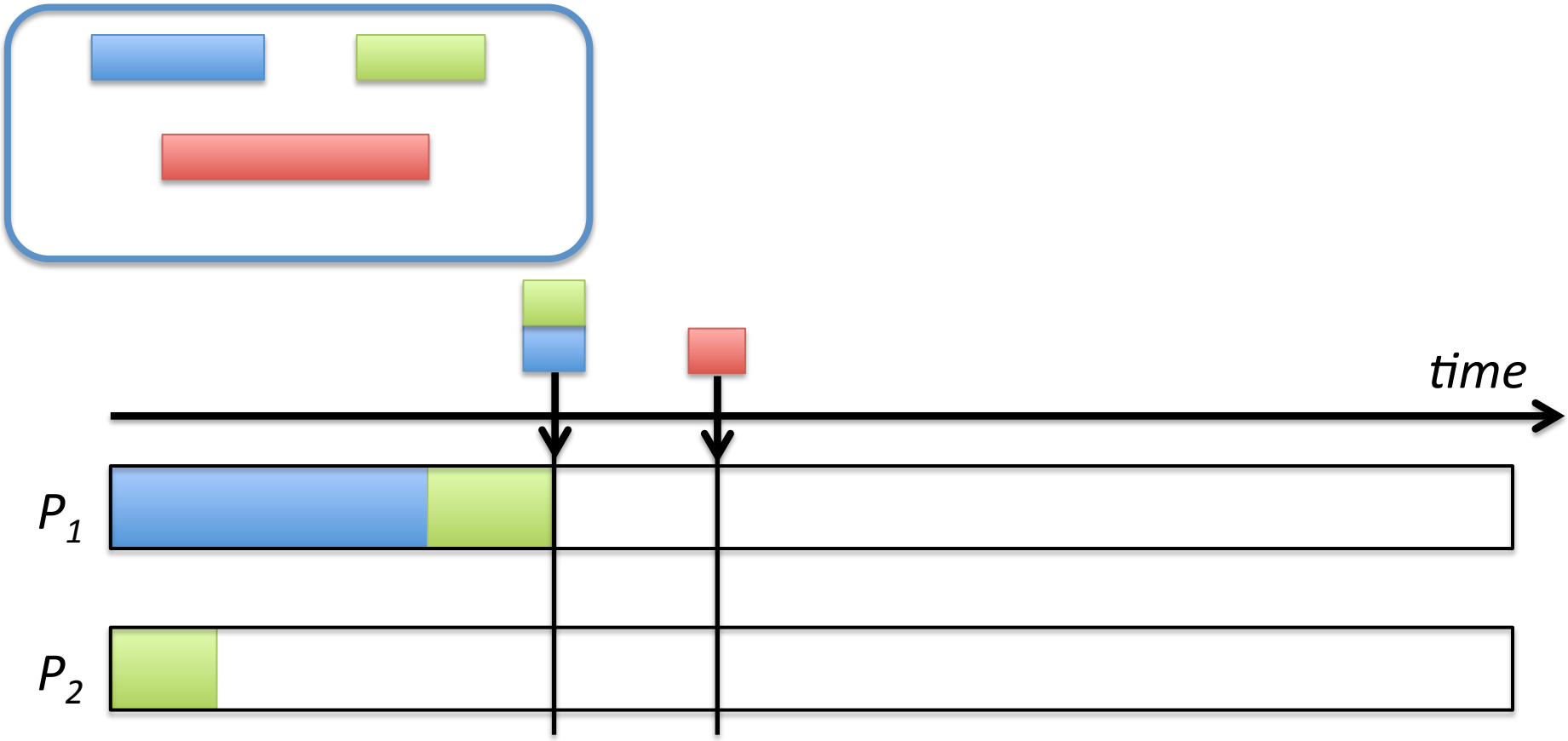
On uniprocessor, EDF **horizontally** assign jobs



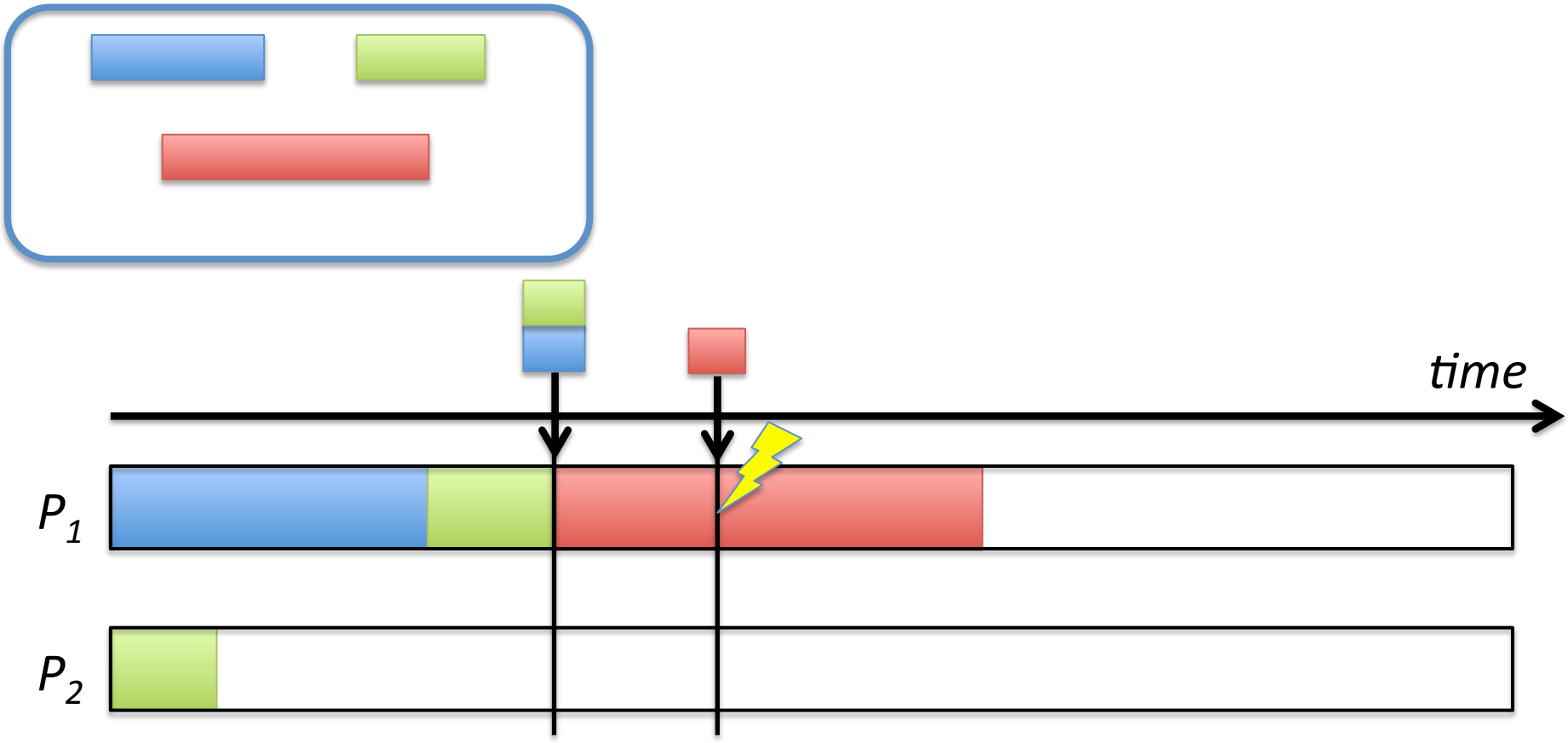
A new solution: Horizontally extend EDF



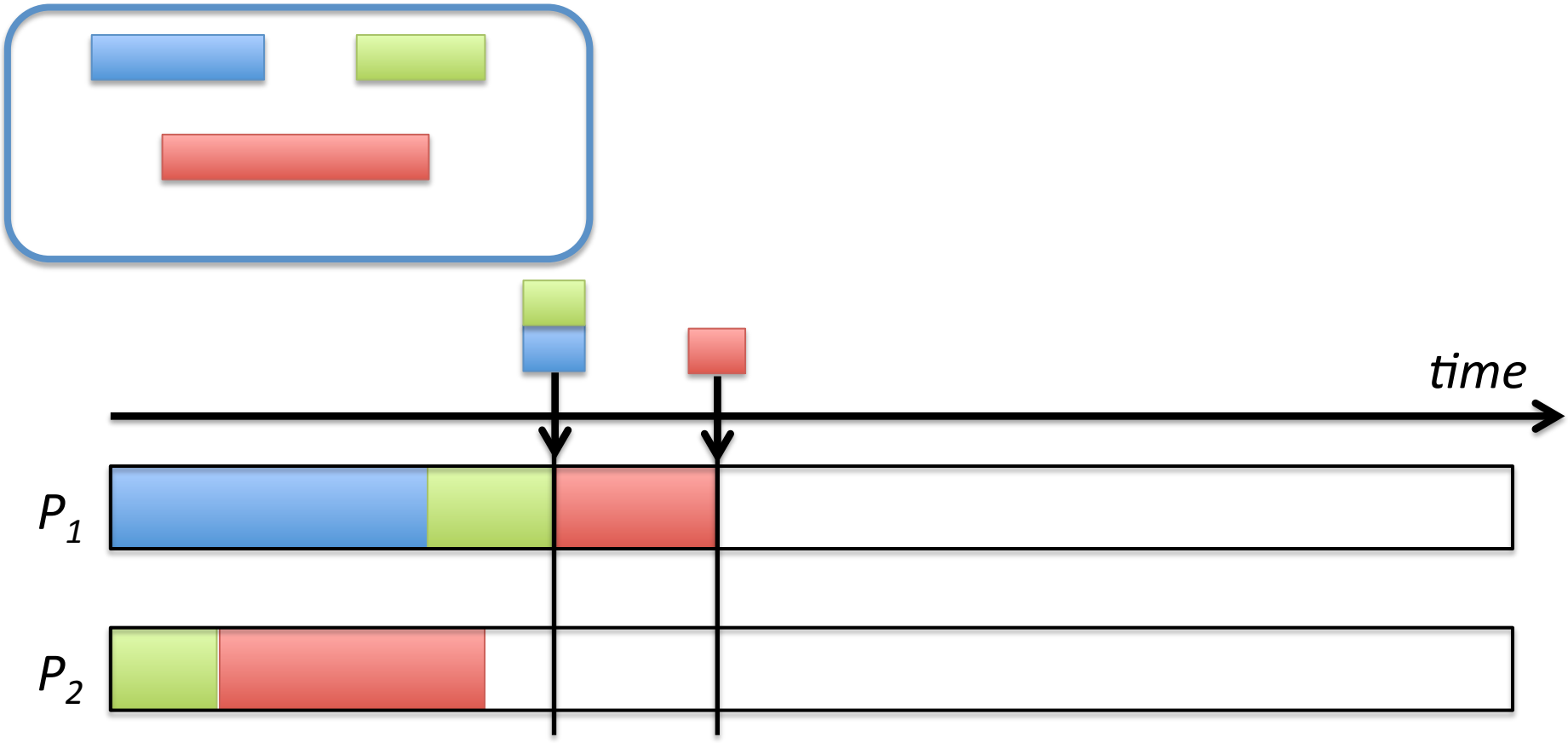
A new solution: Horizontally extend EDF



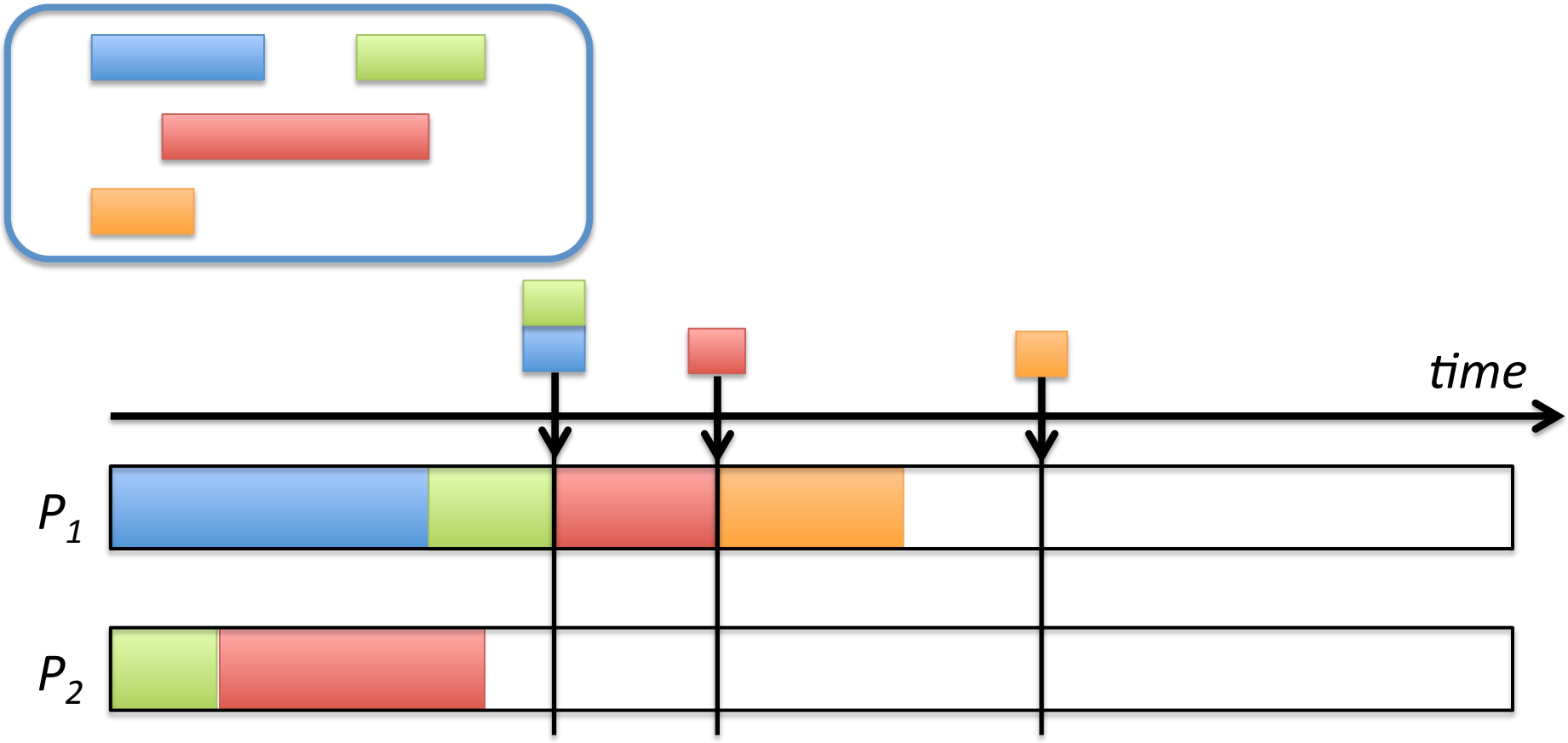
A new solution: Horizontally extend EDF



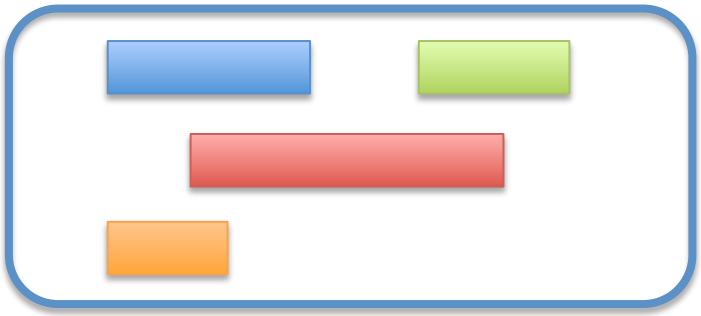
A new solution: Horizontally extend EDF



A new solution: Horizontally extend EDF

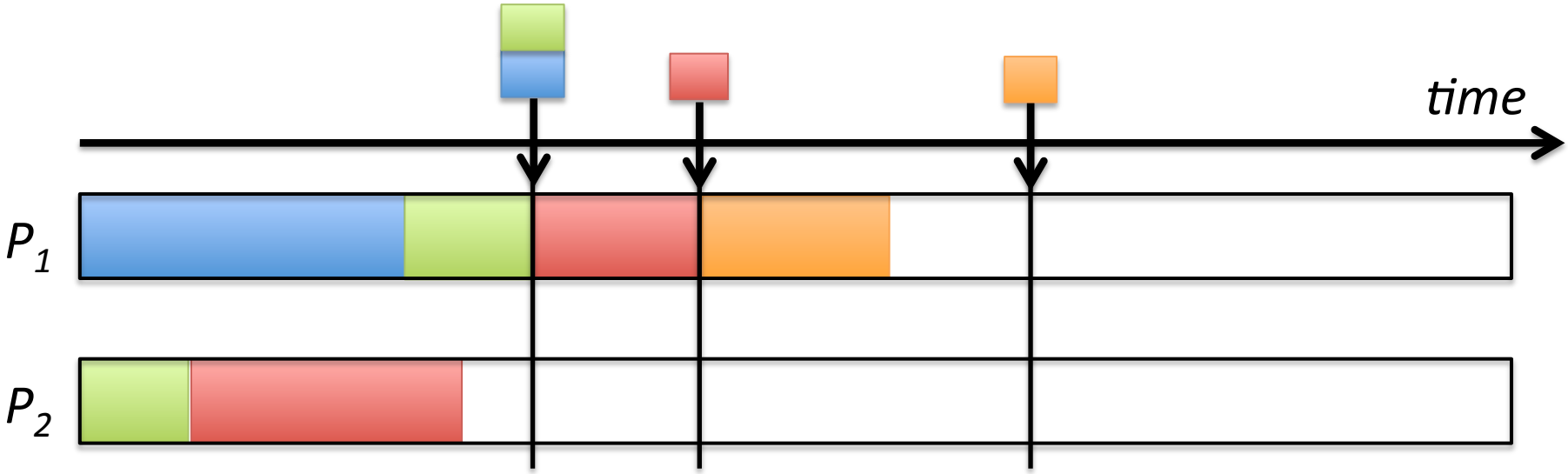


A new solution: Horizontally extend EDF

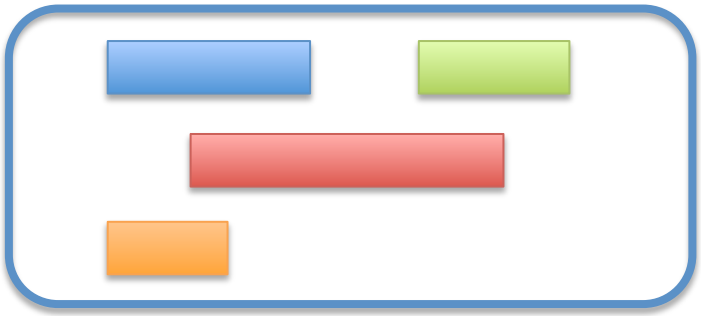


Problem:

We do not know when next jobs will arrive!

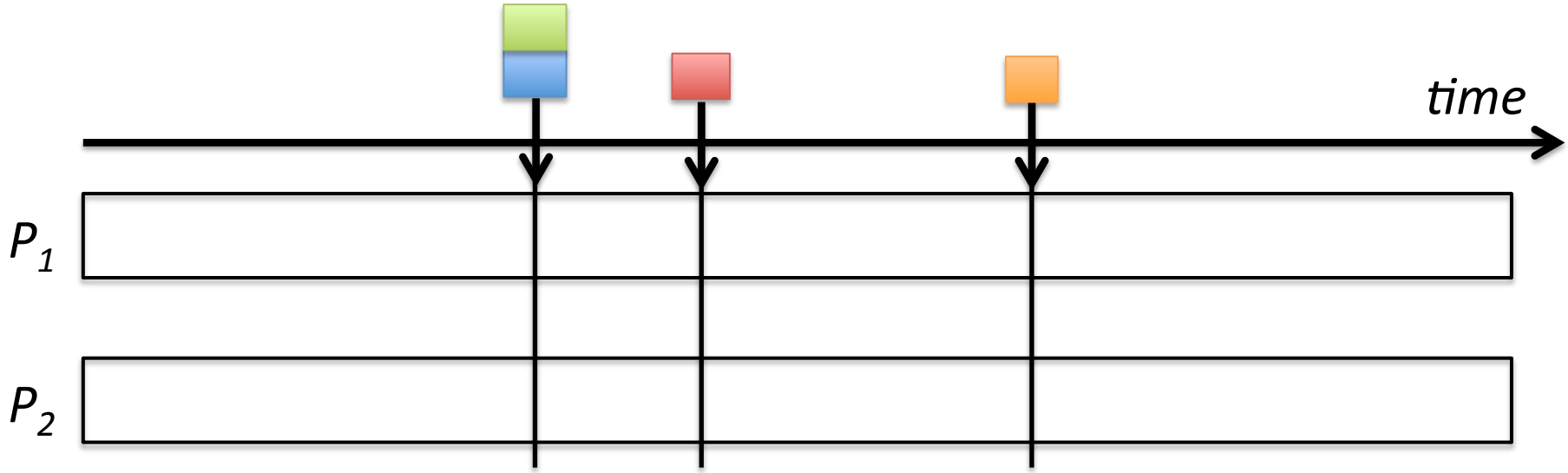


A new solution: Horizontally extend EDF

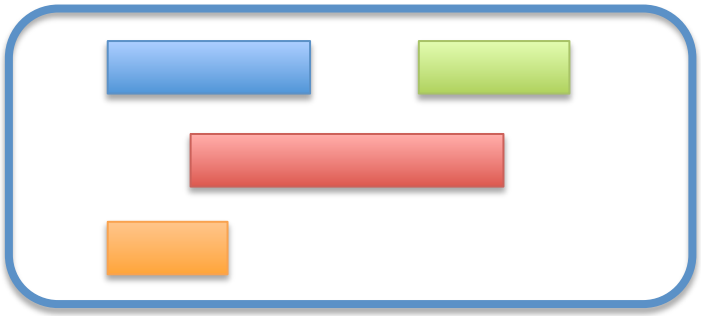


Solution:

- Reserve time for potential future job arrivals

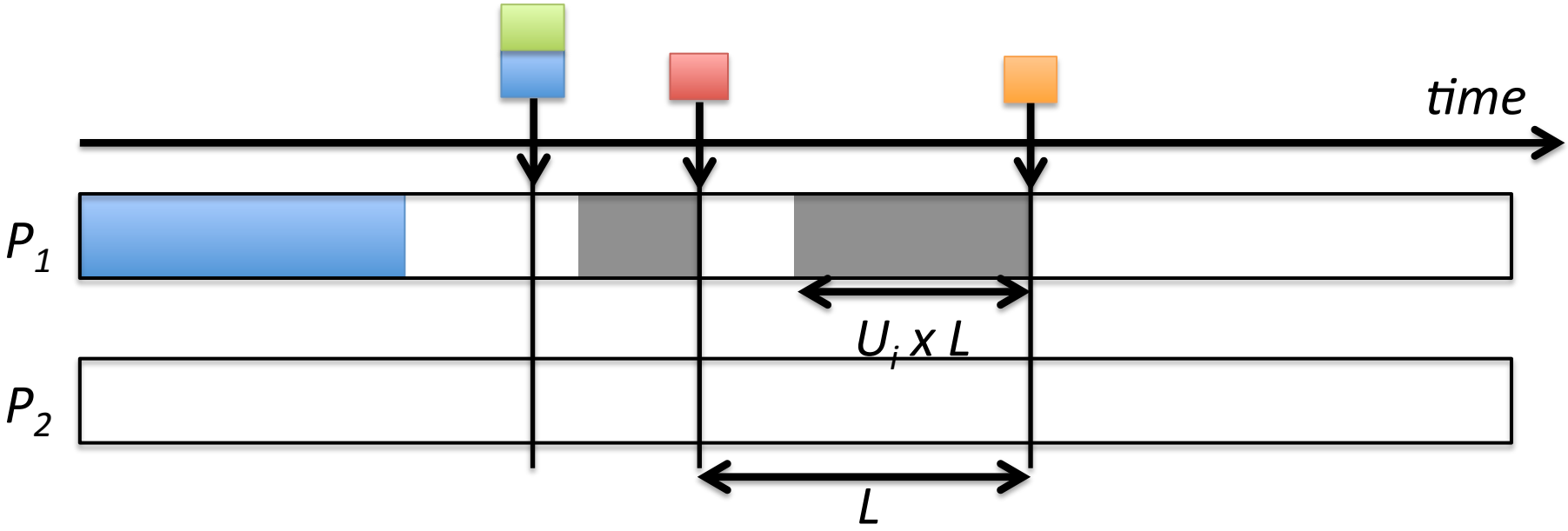


A new solution: Horizontally extend EDF

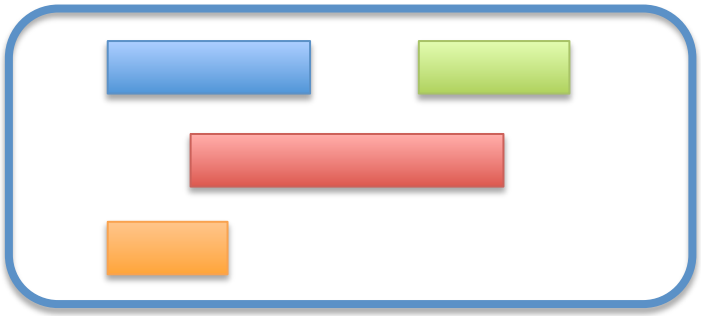


Solution:

- Reserve time for potential future job arrivals

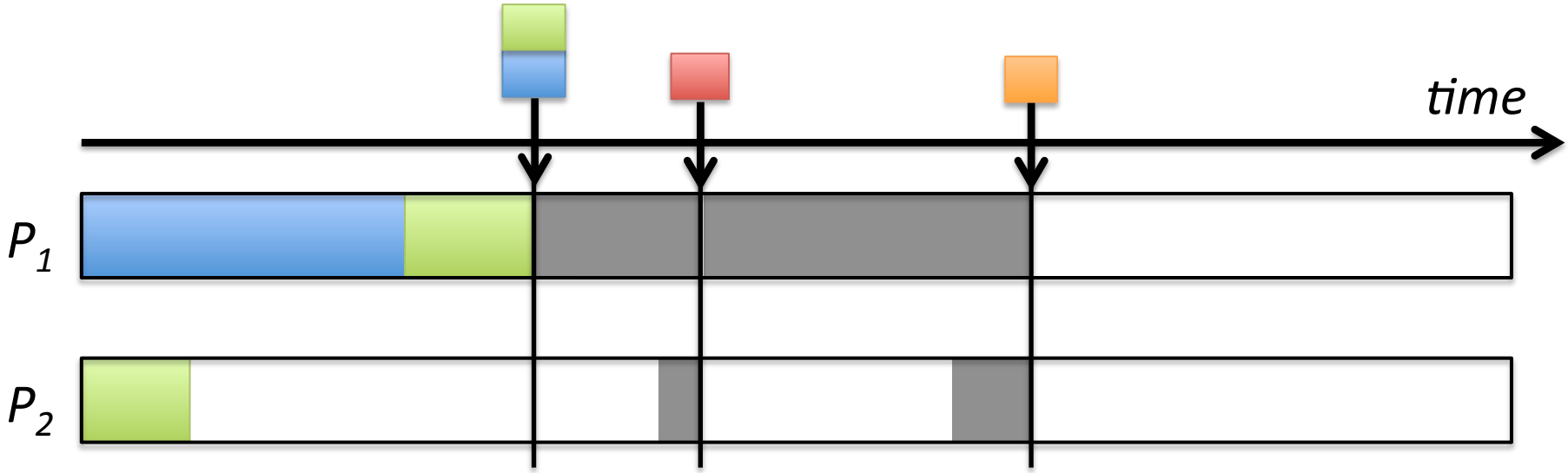


A new solution: Horizontally extend EDF

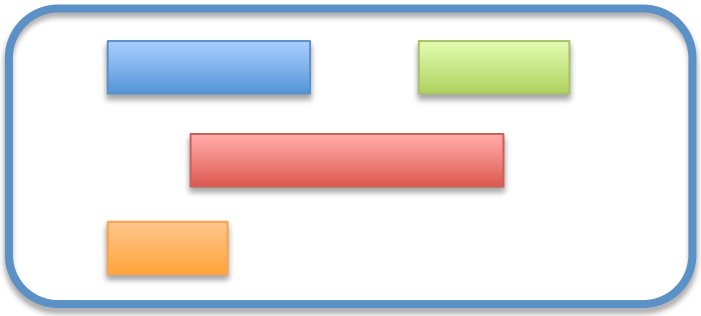


Solution:

- Reserve time for potential future job arrivals

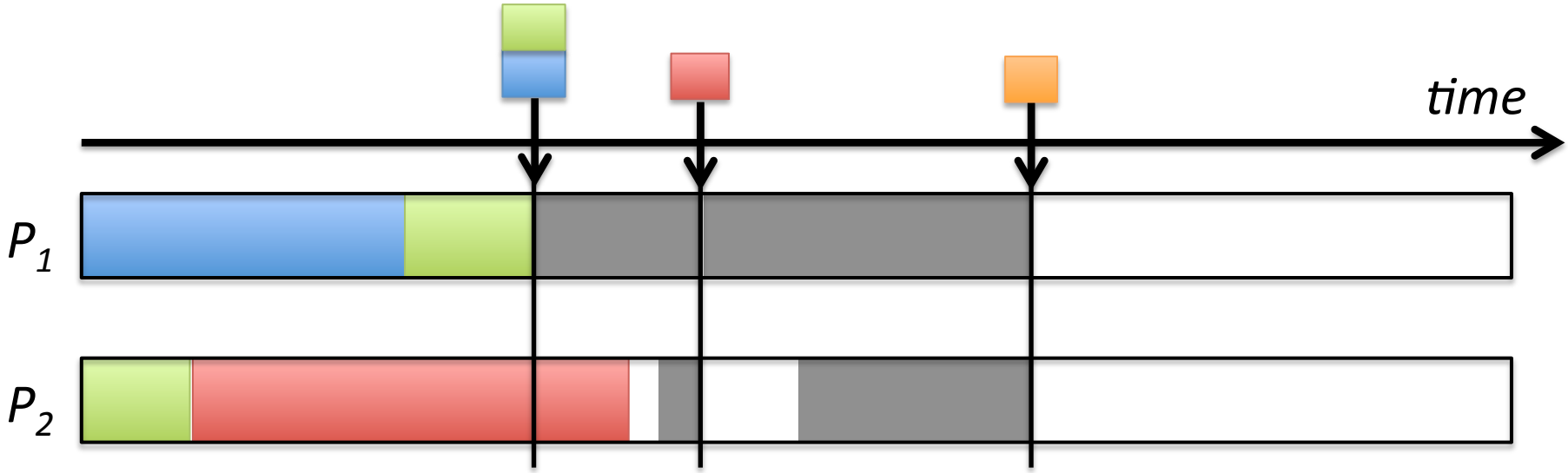


A new solution: Horizontally extend EDF

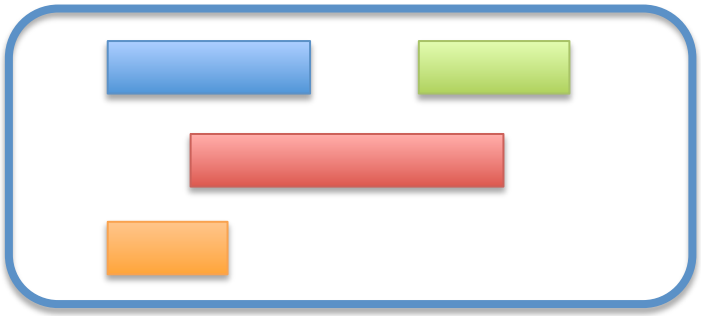


Solution:

- Reserve time for potential future job arrivals

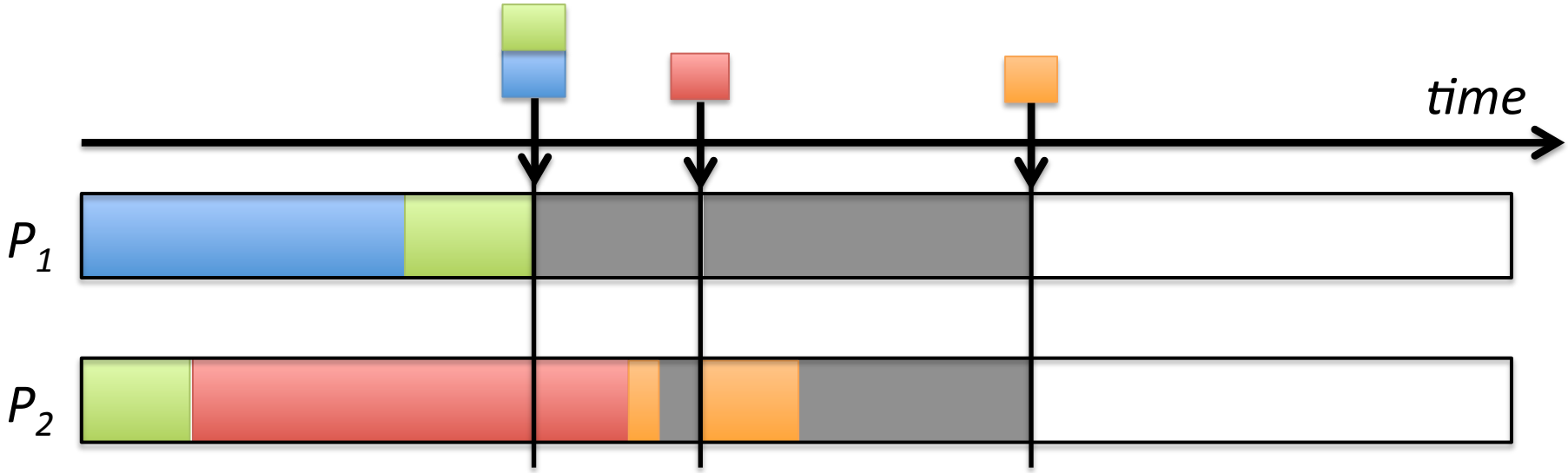


A new solution: Horizontally extend EDF

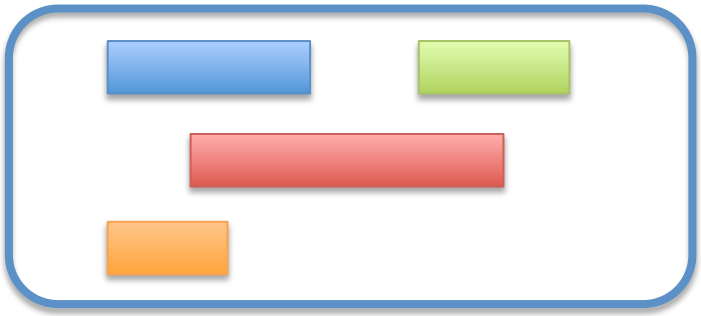


Solution:

- Reserve time for potential future job arrivals

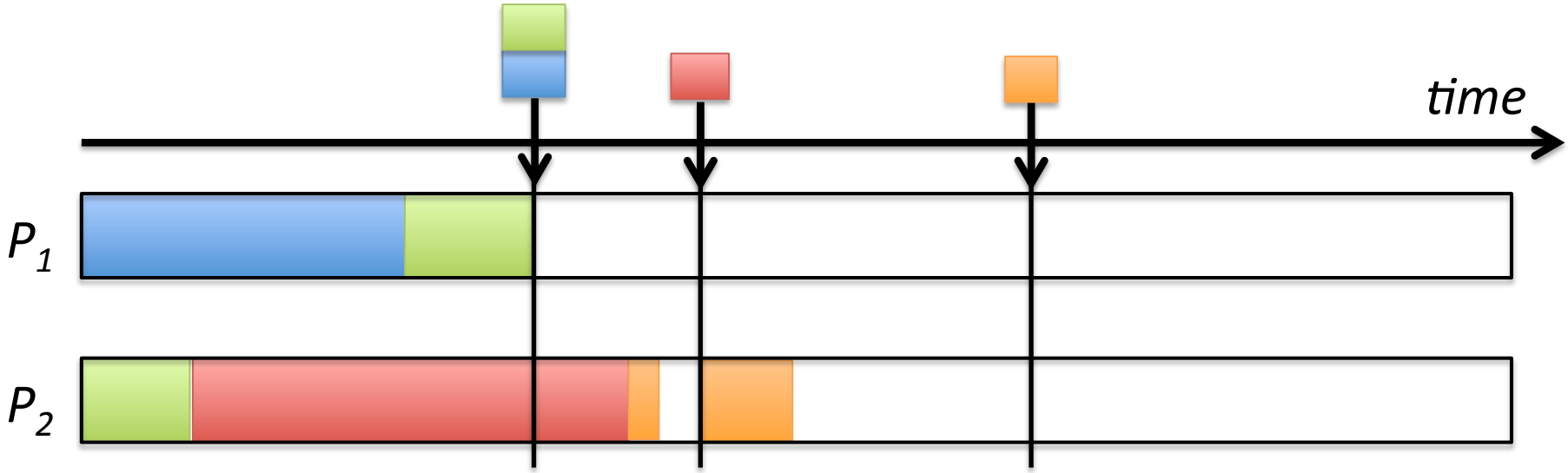


A new solution: Horizontally extend EDF

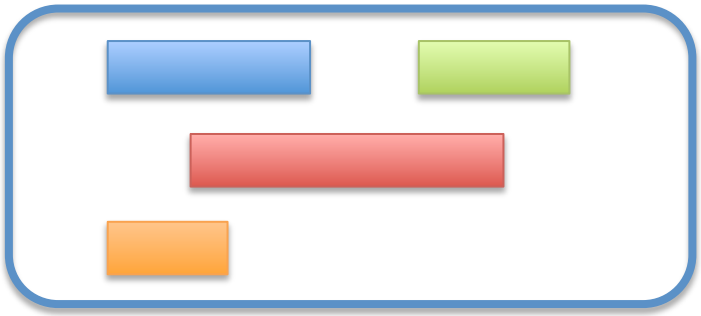


Solution:

- Reserve time for potential future job arrivals

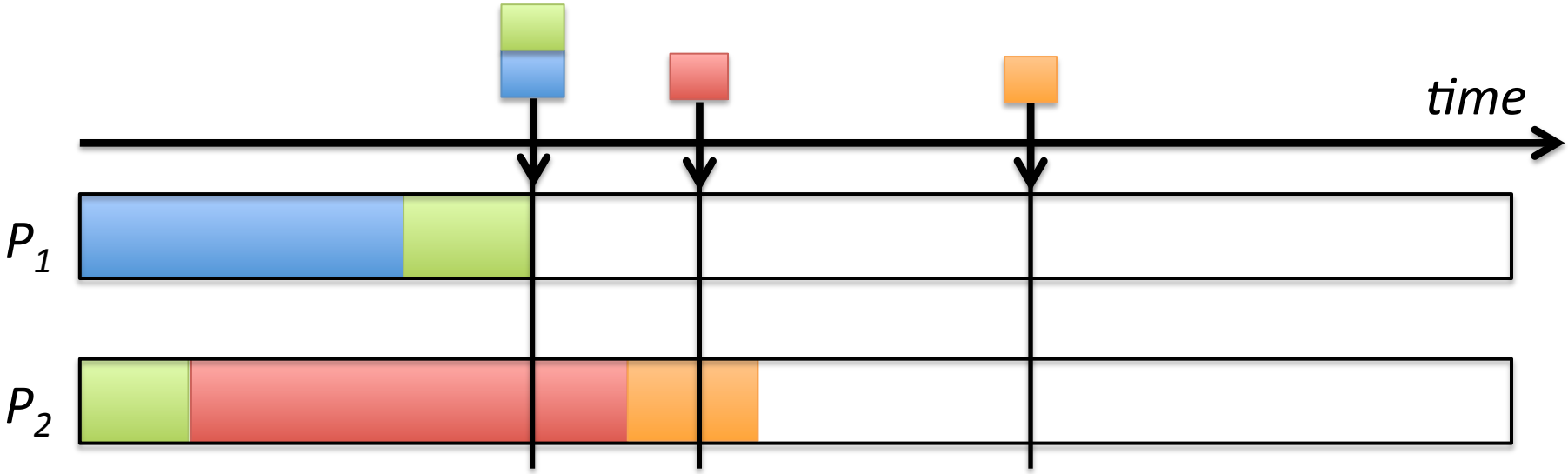


A new solution: Horizontally extend EDF

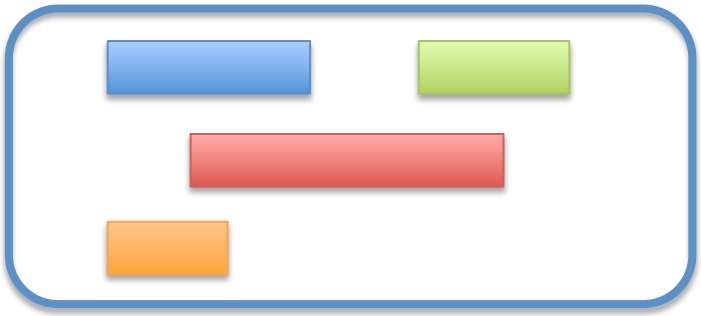


Solution:

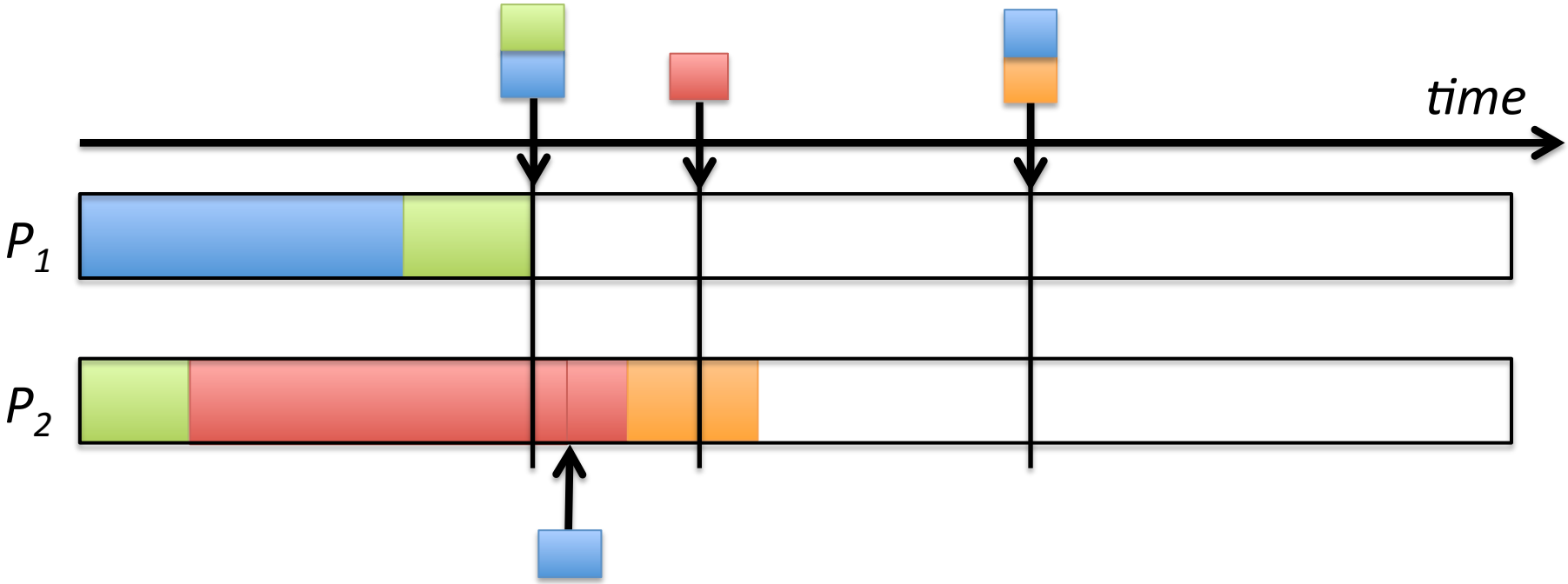
- Reserve time for potential future job arrivals
- Execute EDF on each processor



A new solution: Horizontally extend EDF

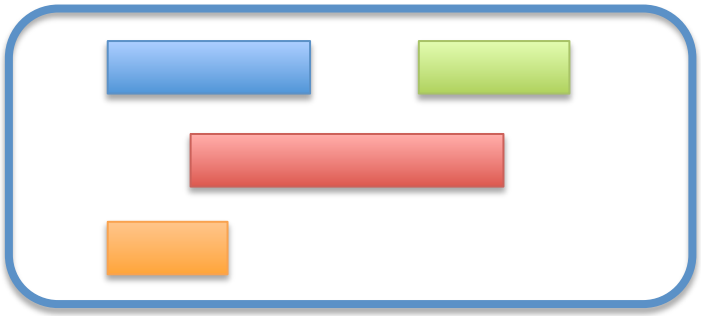


If a new job arrives, reallocates tasks

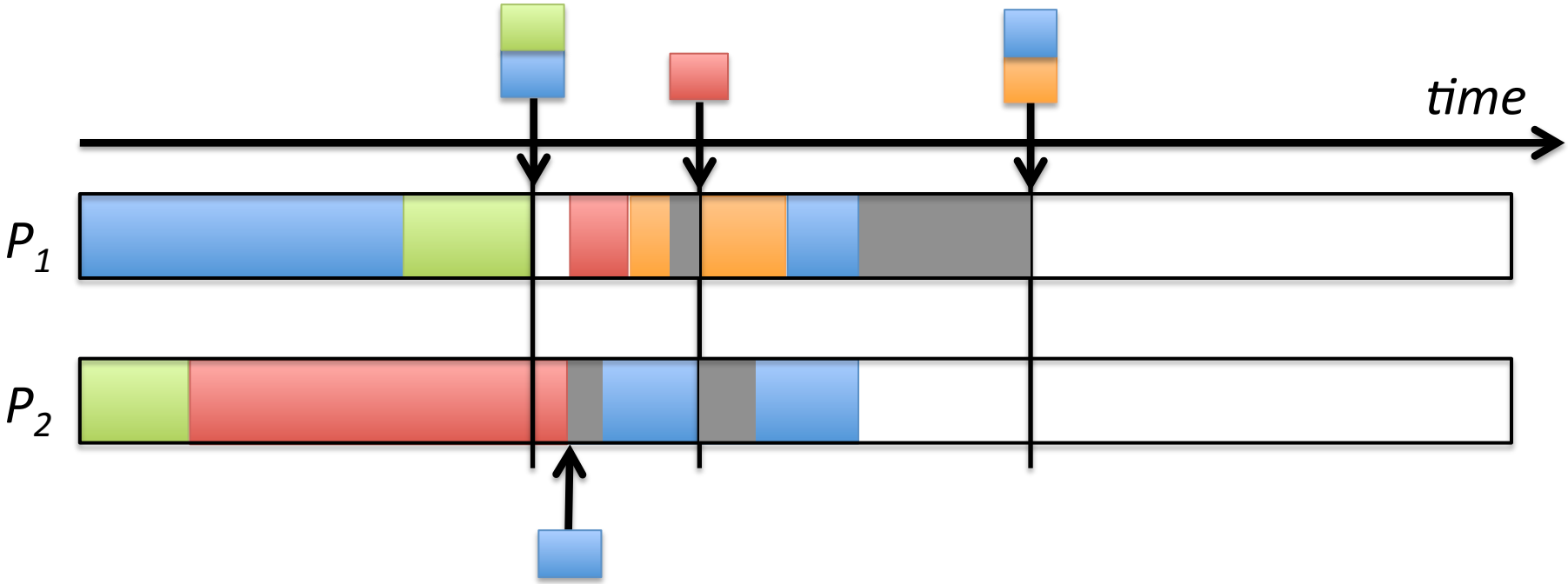


U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

A new solution: Horizontally extend EDF

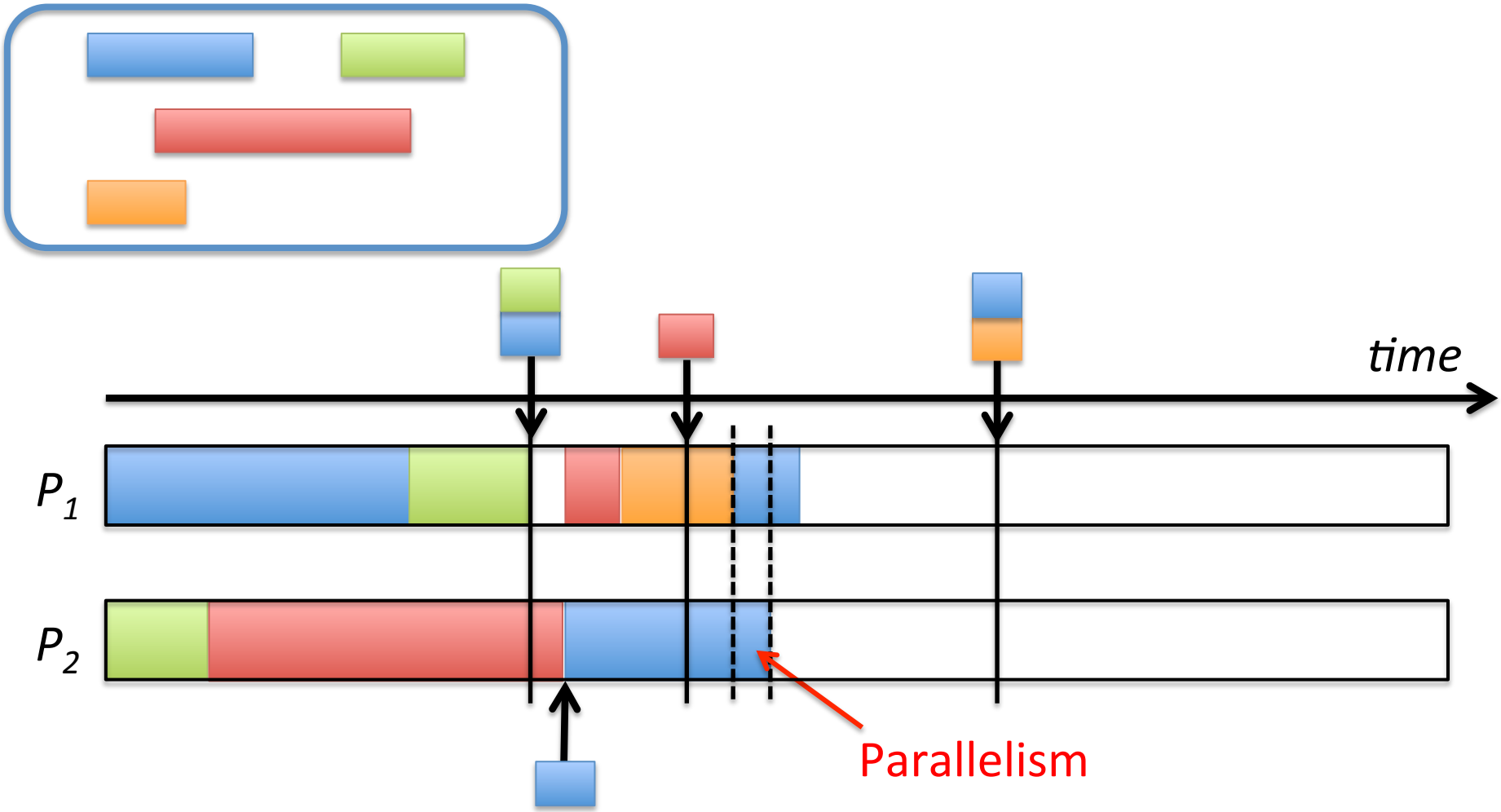


If a new job arrives, reallocates tasks



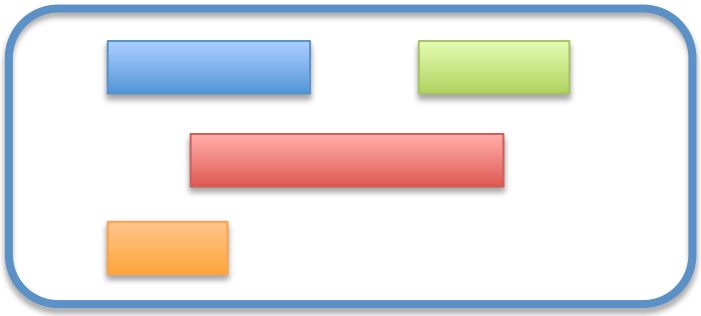
U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

A new solution: Horizontally extend EDF

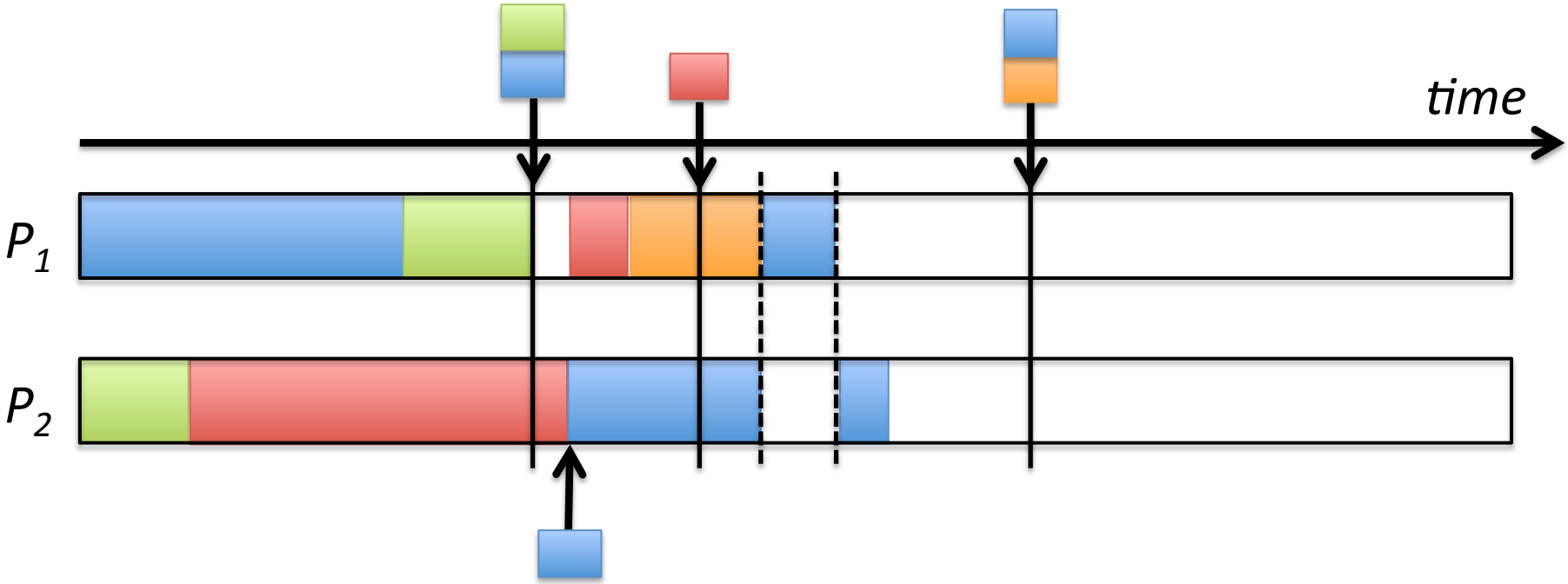


U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

A new solution: Horizontally extend EDF



EDF-D: slight variation of EDF avoiding parallelism



U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

To summarize: Two phases

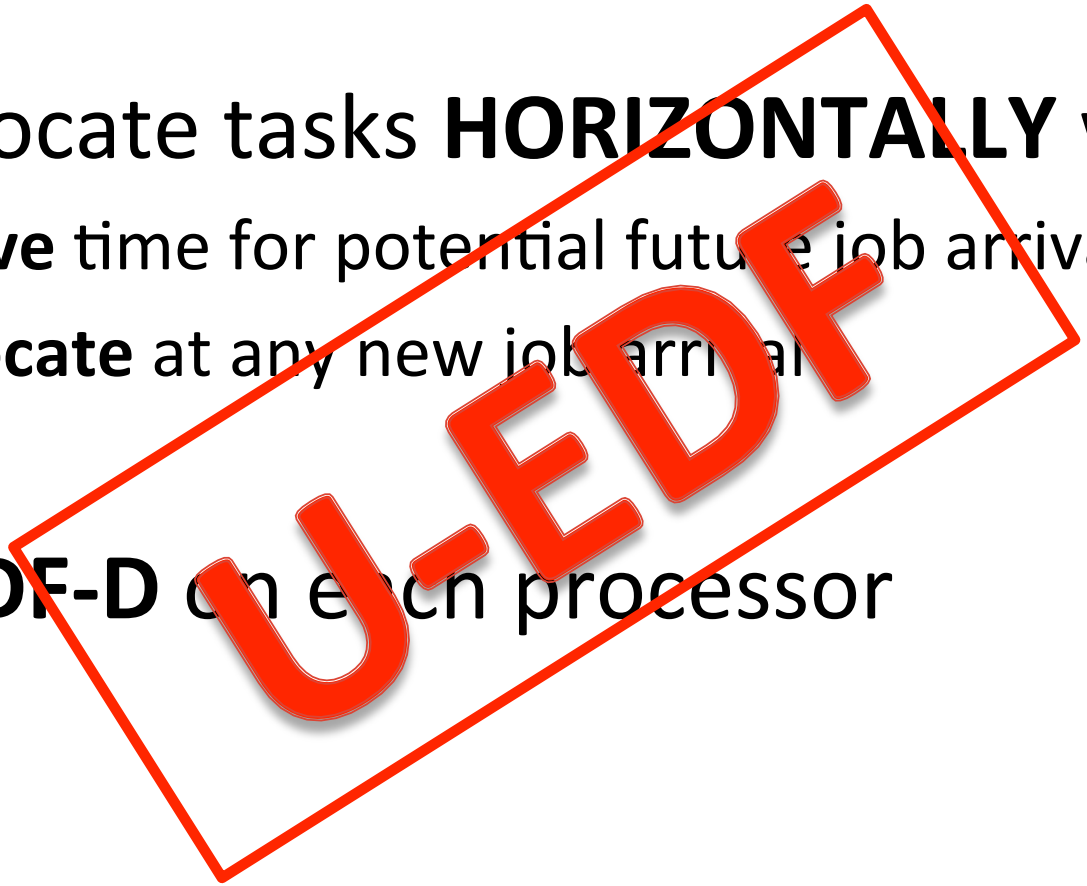
1) Preallocate tasks **HORIZONTALLY** with **EDF**
   - **Reserve** time for potential future job arrivals
   - **Reallocate** at any new job arrival

2) Use **EDF-D** on each processor

To summarize: Two phases

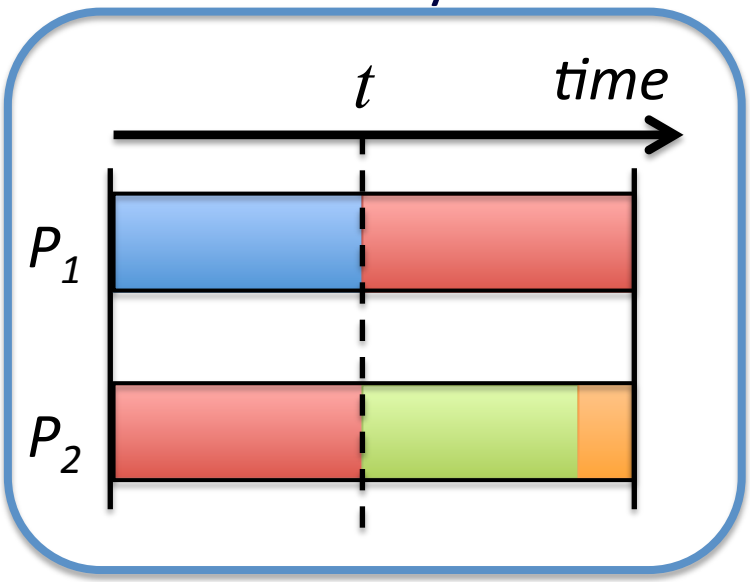
1) Preallocate tasks **HORIZONTALLY** with **EDF**
   - **Reserve** time for potential future job arrivals
   - **Reallocate** at any new job arrival

2) Use **EDF-D** on each processor

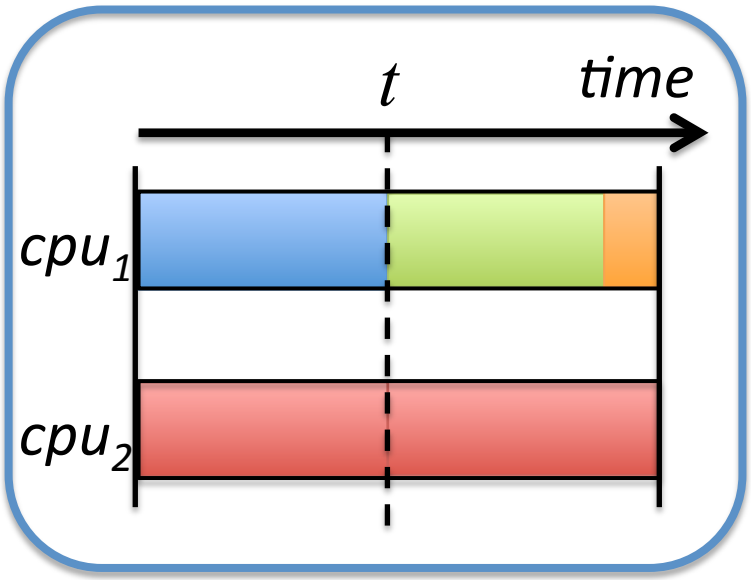


Implementation Considerations: Virtual Processing

Virtual Schedule
Produced by U-EDF

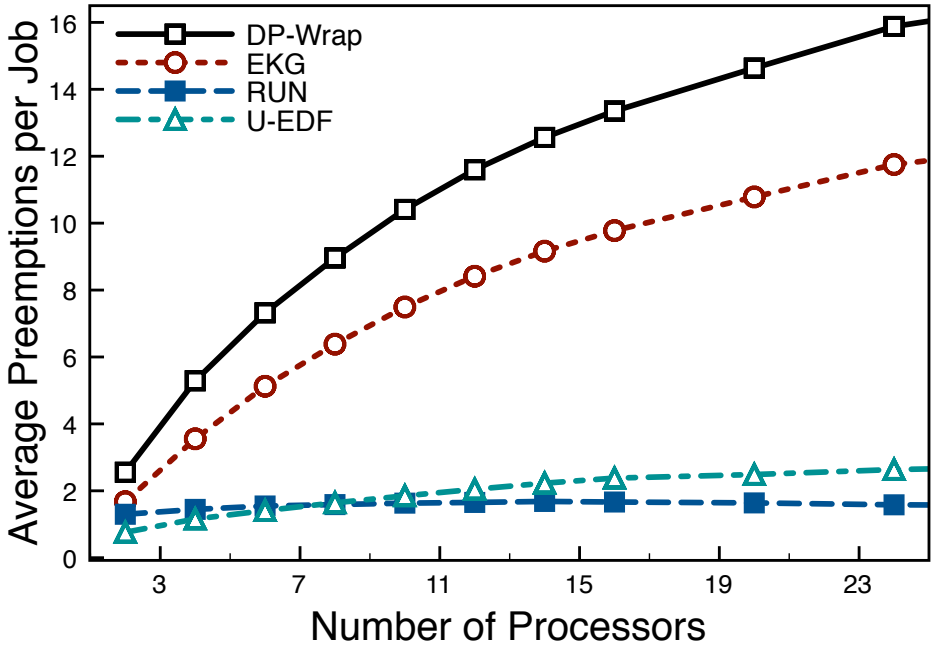


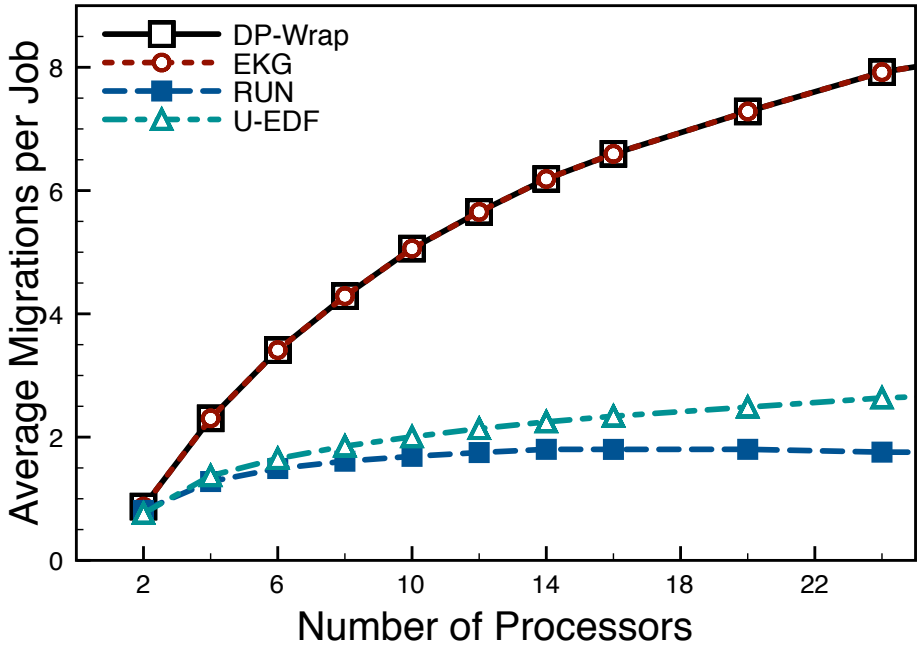
Physical Schedule



U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

Simulation Results: Few preemptions and migrations





U-EDF: An Unfair but Optimal Multiprocessor Scheduling Algorithm for Sporadic Tasks

Conclusion

- **U-EDF**:
  - Optimal for **sporadic** tasks with implicit deadlines
  - Unfair
  - Extends EDF to multiprocessor platforms
  - Causes few preemptions and migrations

- A first step to reconcile theoretical and practical considerations