# On Spin Locks in AUTOSAR: Blocking Analysis of FIFO, Unordered, and Priority-Ordered Spin Locks


Alexander Wieder and Björn Brandenburg
MPI-SWS



RTSS 2013
12/04/2013
Vancouver, Canada



Max Planck Institute for Software Systems

MAX-PLANCK-GESELLSCHAFT

# Motivation: AUTOSAR

AUTOSAR: OS-specification widely used for embedded applications

Resources accessed from multiple cores:

AUTOSAR mandates spin locks.

# Motivation: AUTOSAR

AUTOSAR: OS-specification widely used for embedded applications

Resources accessed from multiple cores:

AUTOSAR mandates spin locks.

But AUTOSAR **does not specify spin lock type**!

# Motivation: AUTOSAR

AUTOSAR: OS-specification widely used for embedded applications

Resources accessed from multiple cores:

AUTOSAR mandates spin locks.

But AUTOSAR **does not specify spin lock type**!

**Which type should be used?**

# Spin Lock Types

Variety of reasonable choices:

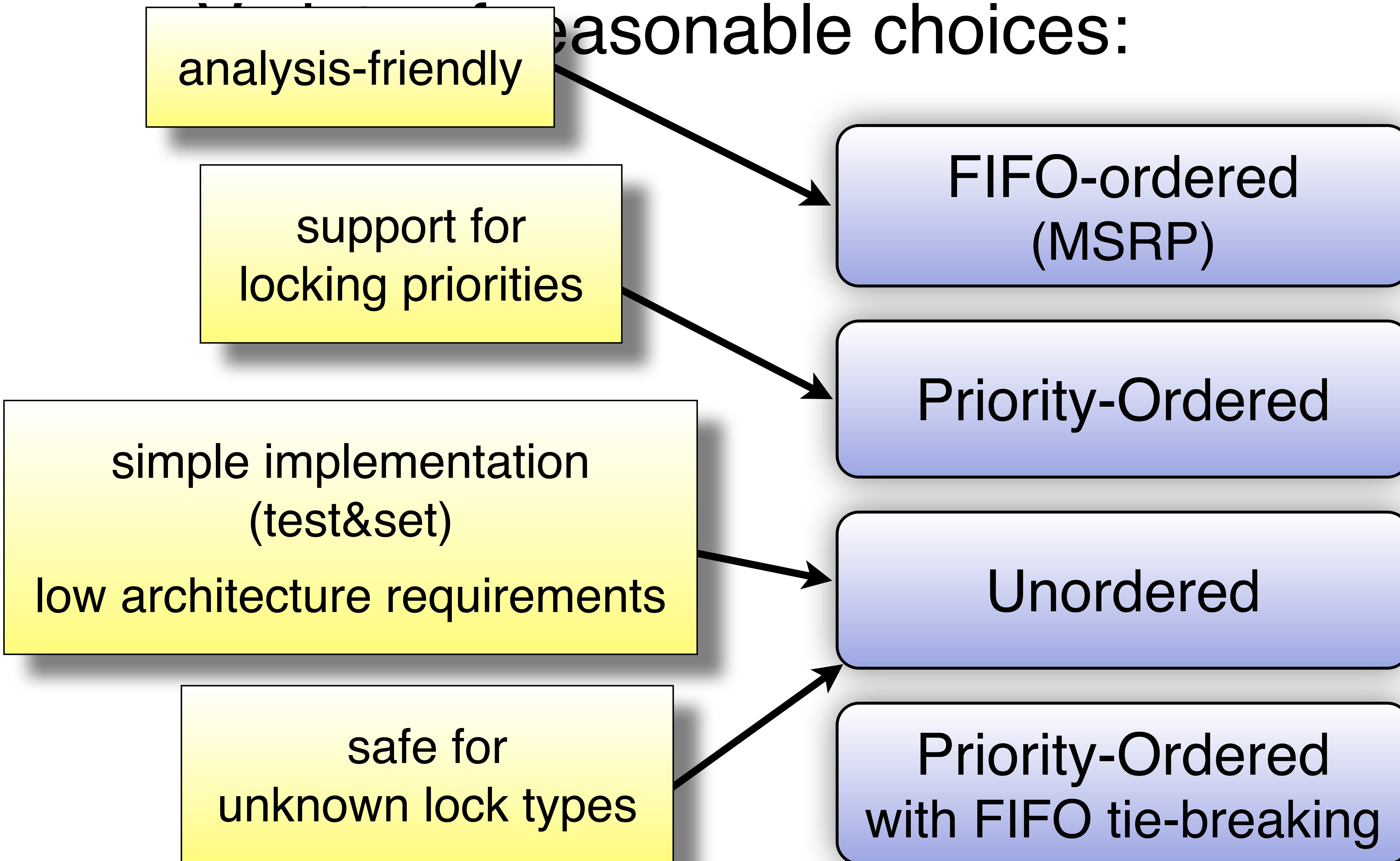
- Non-Preemptable Spinning
- Preemptable Spinning

×

- FIFO-ordered (MSRP)
- Priority-Ordered
- Unordered
- Priority-Ordered with FIFO tie-breaking

# Spin Lock Types

easonable choices:



# Spin Lock Types

Variety of reasonable choices:

FIFO-ordered
SRP)

Non-Preemp
Spinning

**No blocking analysis**
available
for most types!

Ordered

dered

Preemptable
Spinning

Priority-Ordered
with FIFO tie-breaking

# Contributions

Blocking analysis
for **8 types** of
spin locks

# Contributions

Blocking analysis
for **8 types** of
spin locks

7/8 spin lock types:
**first blocking analysis**

# Contributions

Blocking analysis for **8 types** of spin locks

7/8 spin lock types: **first blocking analysis**

Asymptotically **less pessimistic** than prior approaches

# Contributions

Blocking analysis
for **8 types** of
spin locks

7/8 spin lock types:
**first blocking analysis**

Asymptotically
**less pessimistic**
than prior approaches

**Suggest AUTOSAR API changes**
based on evaluation results

# Novel Spin Lock Analysis

Key Technique:

Blocking Analysis
modeled as
**Integer Linear Program** (ILP)

# Novel Spin Lock Analysis

Key Technique:

Blocking Analysis
modeled as
**Integer Linear Program** (ILP)

Worst-case blocking bounds
determined by ILP solver

# Challenges

Prior analysis is **pessimistic due to inflation**.

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

# Challenges

Prior analysis is **pessimistic due to inflation**.

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

# Task Model

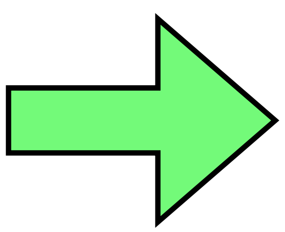
- sporadic tasks: $T_i : \ (e_i, d_i, p_i)$
- constrained deadlines: $d_i \leq p_i$
- partitioned fixed-priority scheduling

# Basic Spin Lock Analysis

Spin locks **busy-wait** while waiting for contended resource.

# Basic Spin Lock Analysis

Spin locks **busy-wait** while waiting for contended resource.



Straight-forward analysis approach:

- treat spin-time as execution time
- apply response-time analysis

# FIFO-Ordered Spin Locks

## Multiprocessor Stack Resource Policy (MSRP) [1]

[1] P. Gai, G. Lipari, and M. Di Natale, “Minimizing memory utilization of real-time task sets in single and multi-processor systems-on-a-chip,” in RTSS’01. IEEE, 2001.

# FIFO-Ordered Spin Locks

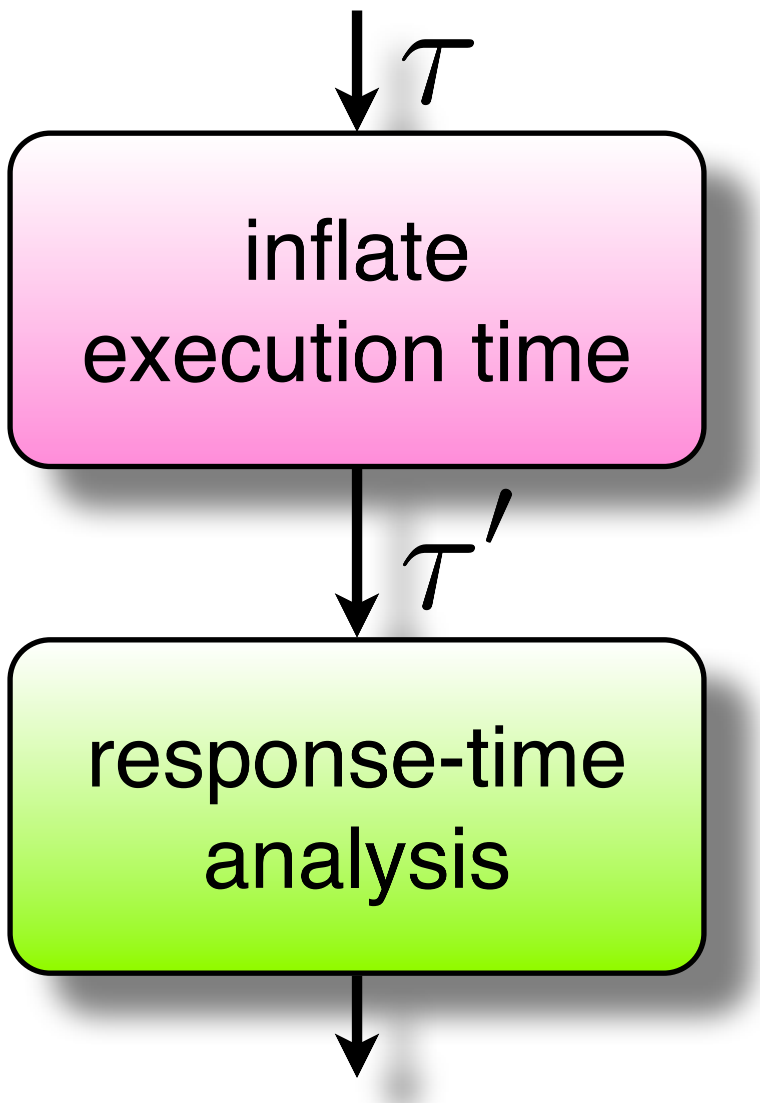
## Multiprocessor Stack Resource Policy (MSRP) [1]

The MSRP uses
**non-preemptable**
**FIFO-ordered spin locks**
for resources shared across processors.

[1] P. Gai, G. Lipari, and M. Di Natale, “Minimizing memory utilization of real-time task sets in single and multi-processor systems-on-a-chip,” in RTSS’01. IEEE, 2001.

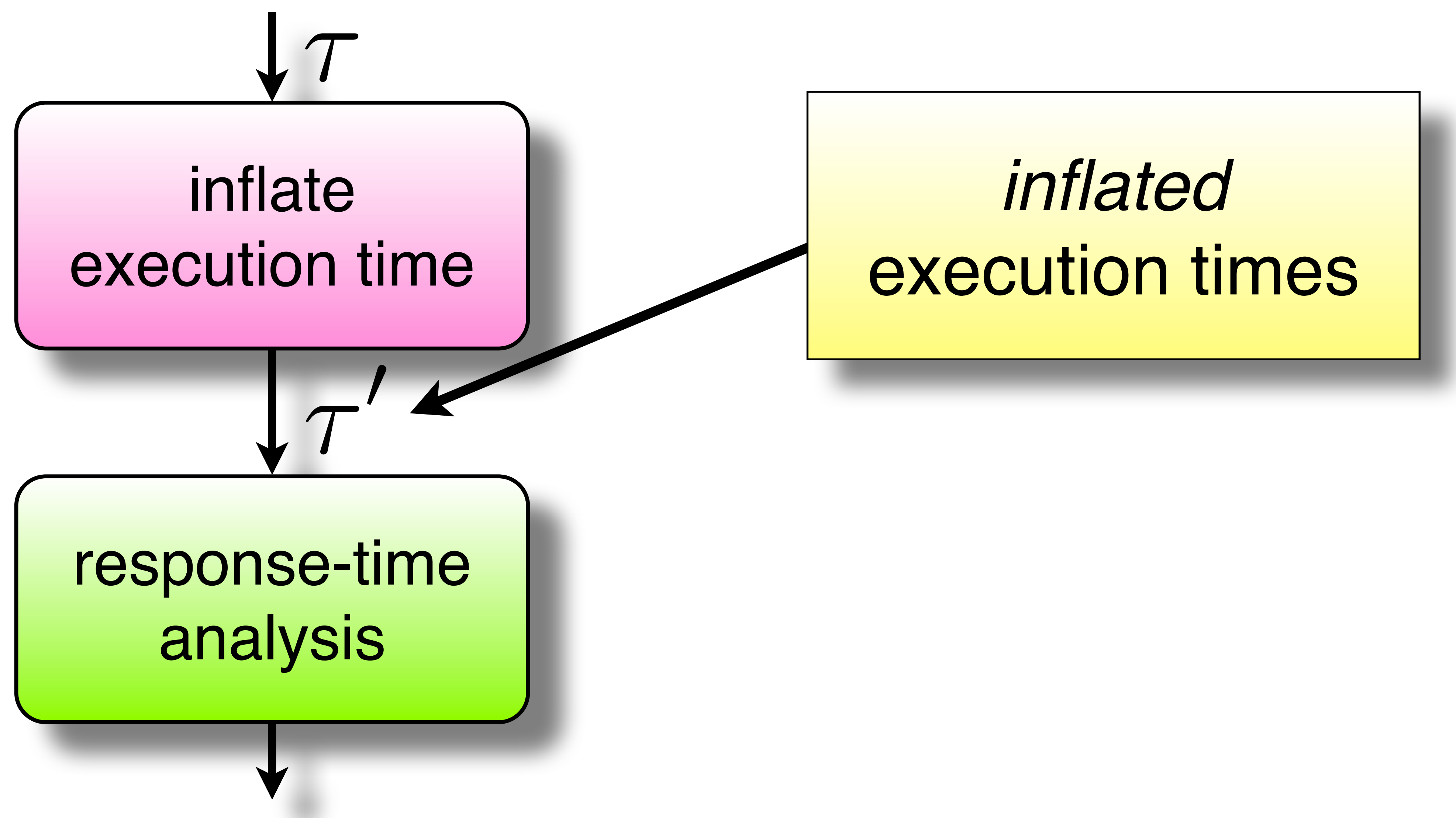
# FIFO-Ordered Spin Locks

classic MSRP analysis



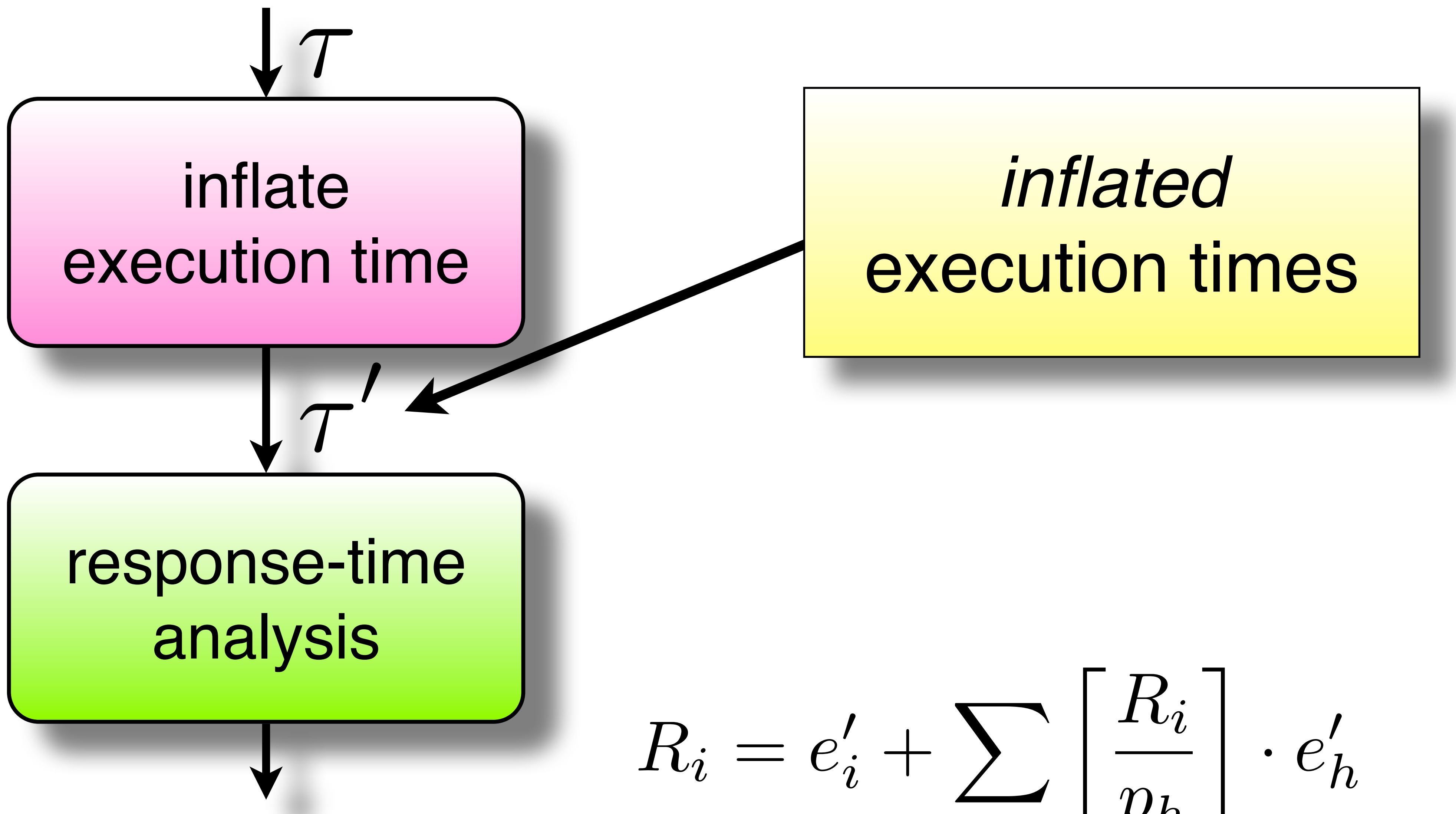
# FIFO-Ordered Spin Locks

classic MSRP analysis



# FIFO-Ordered Spin Locks

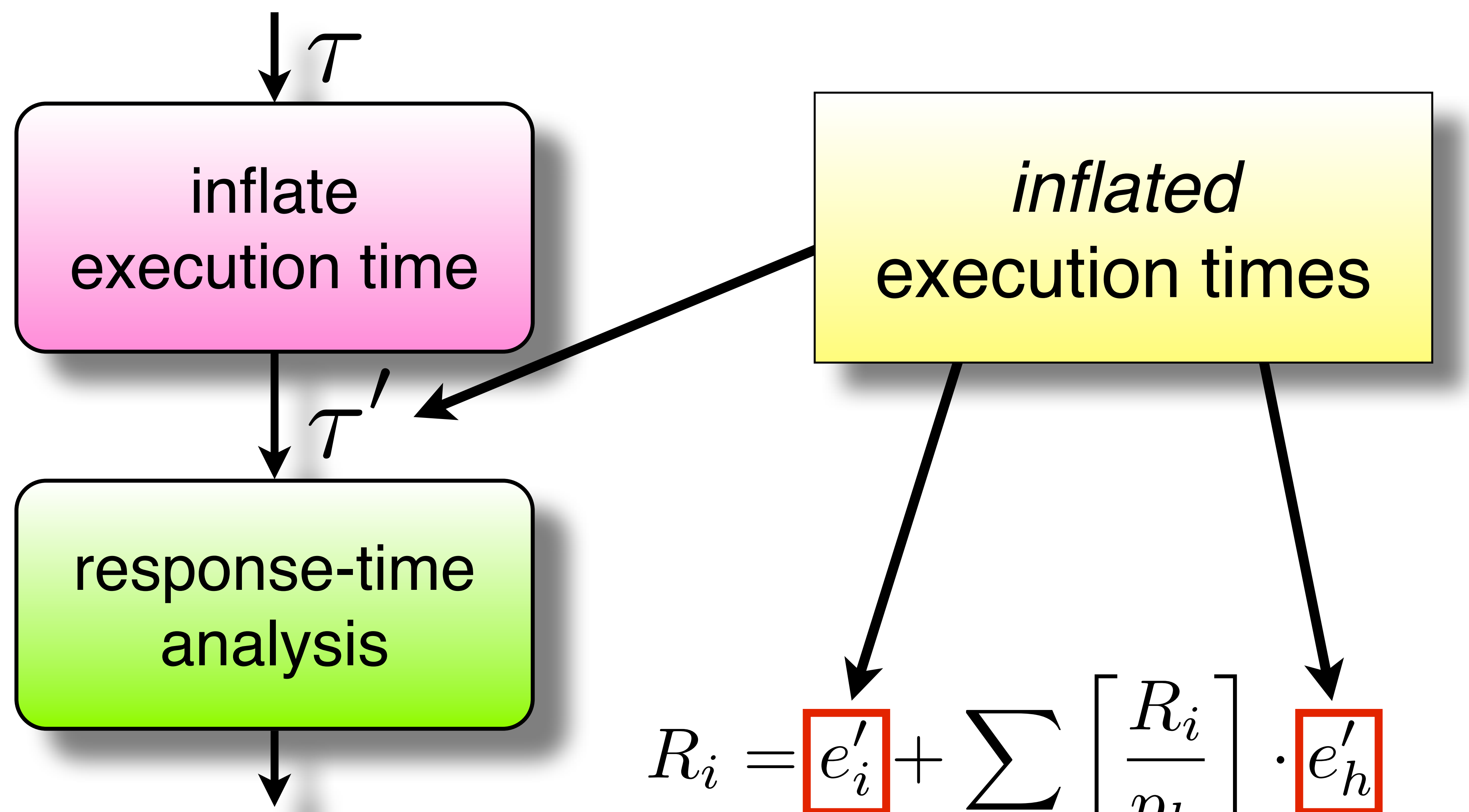
classic MSRP analysis




$$R_i = e'_i + \sum_{h<i} \left\lceil \frac{R_i}{p_h} \right\rceil \cdot e'_h$$

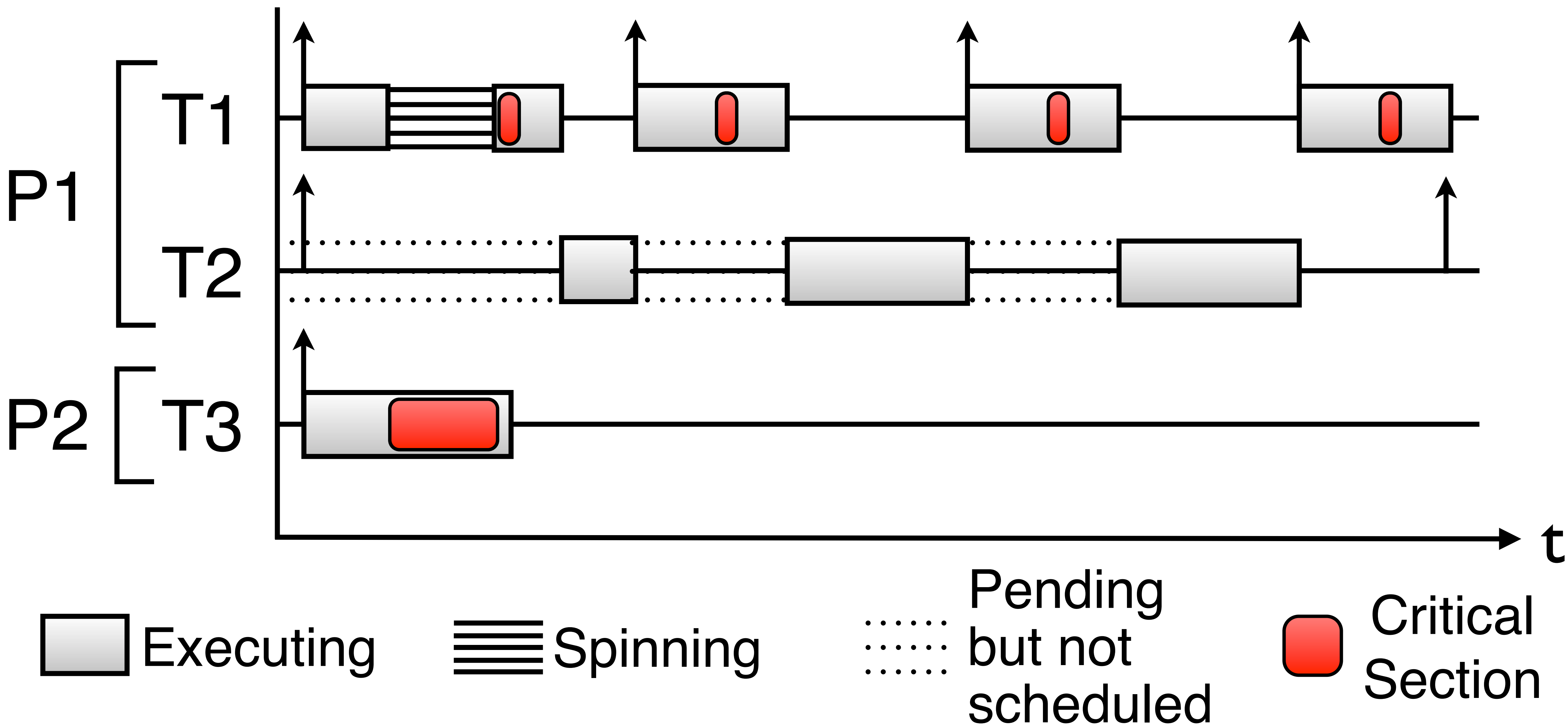
# FIFO-Ordered Spin Locks

classic MSRP analysis

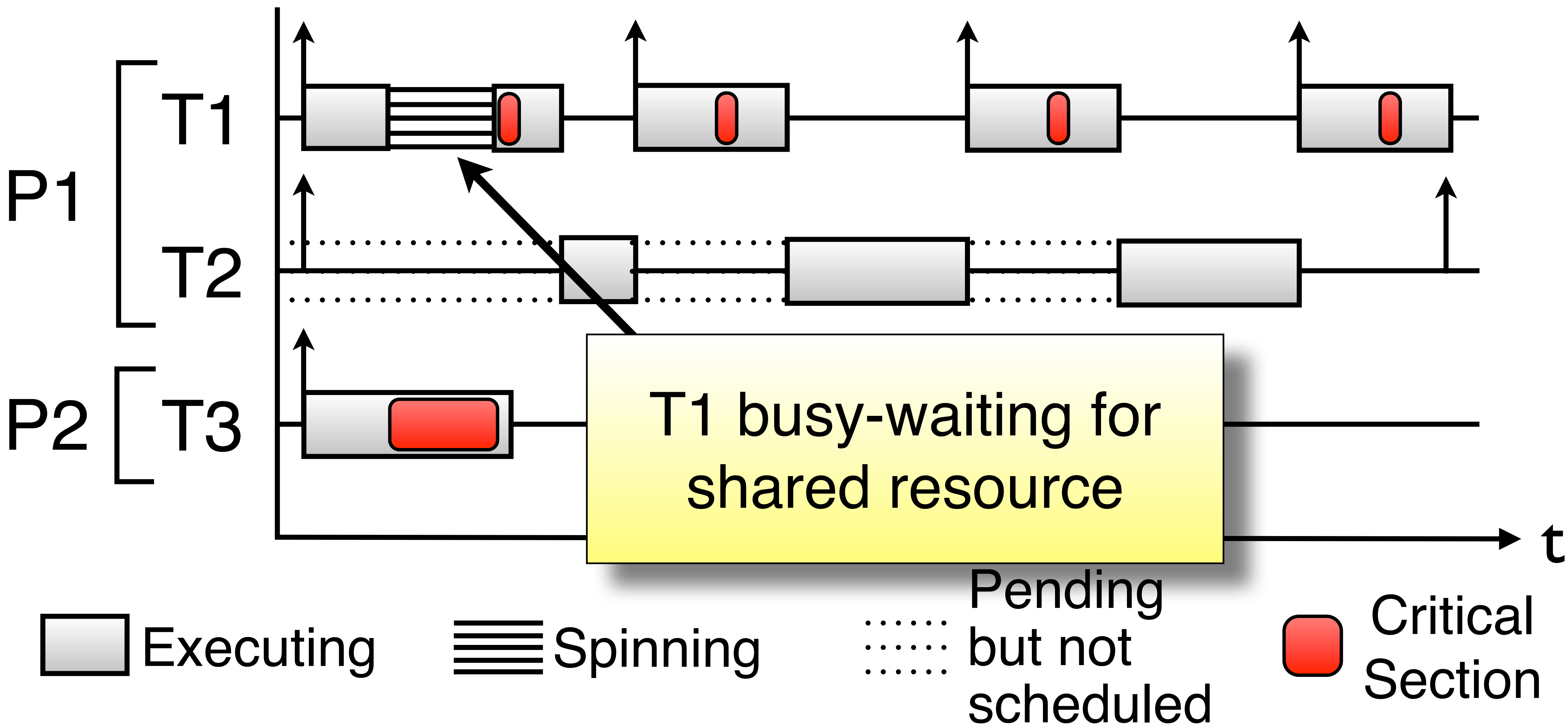



$$R_i = \boxed{e_i'} + \sum_{h<i} \left\lceil \frac{R_i}{p_h} \right\rceil \cdot \boxed{e_h'}$$

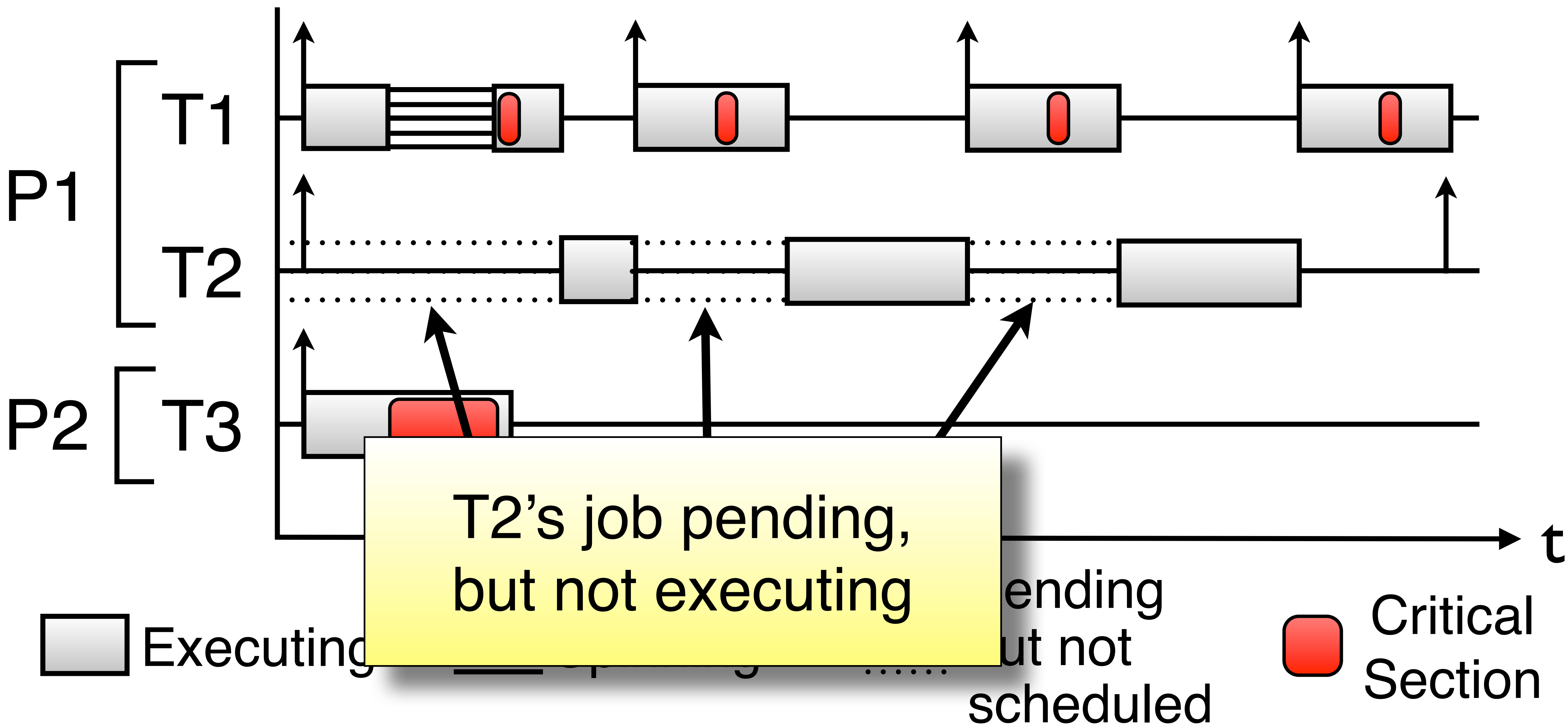
# Execution-Time Inflation in Classic MSRP Analysis



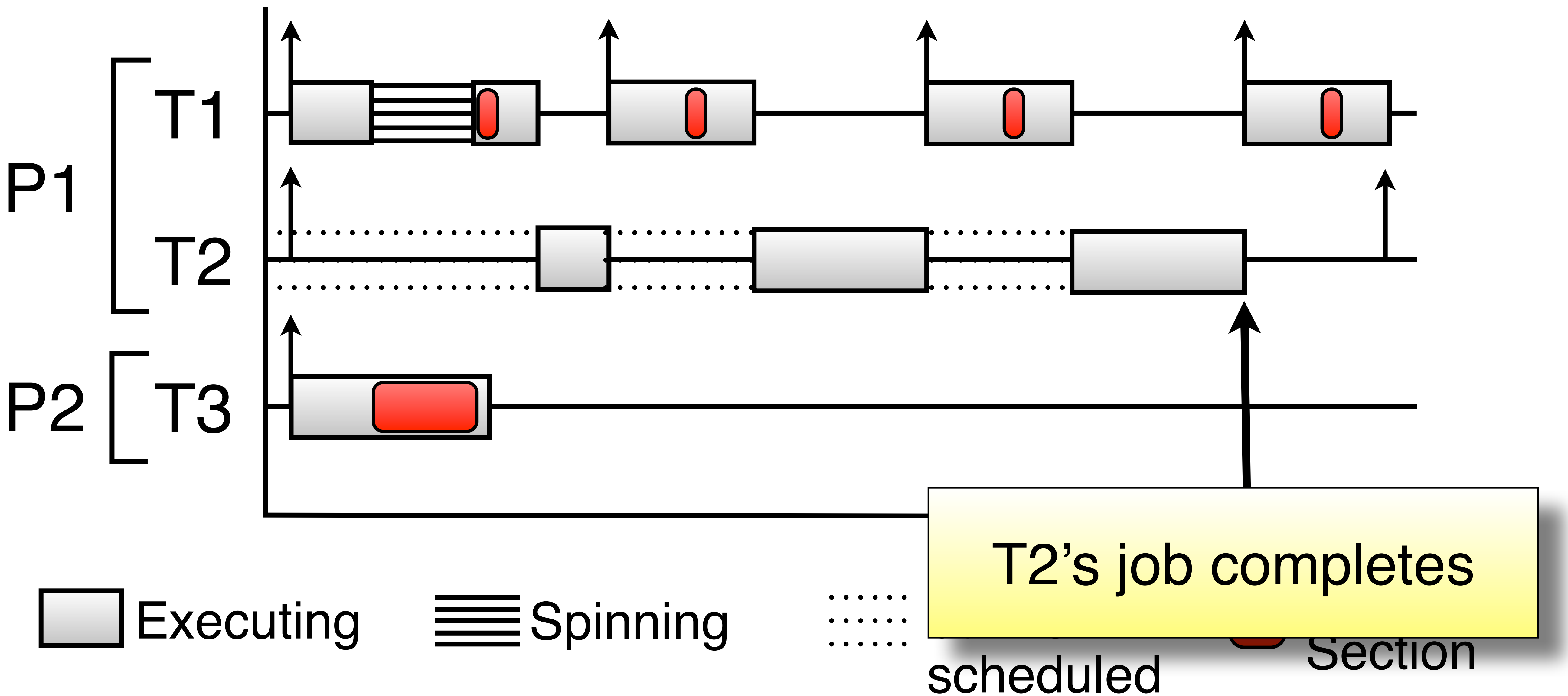
# Execution-Time Inflation in Classic MSRP Analysis



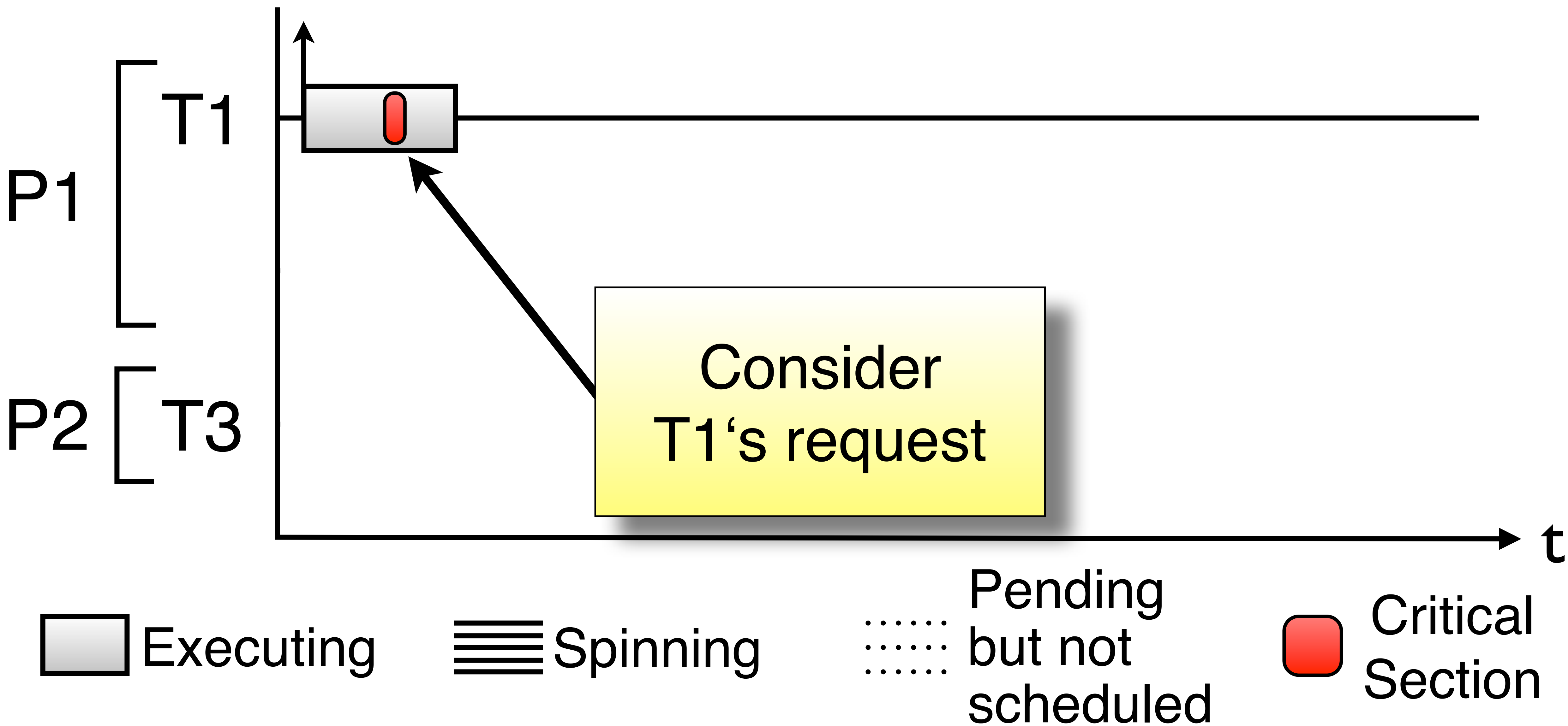
# Execution-Time Inflation in Classic MSRP Analysis



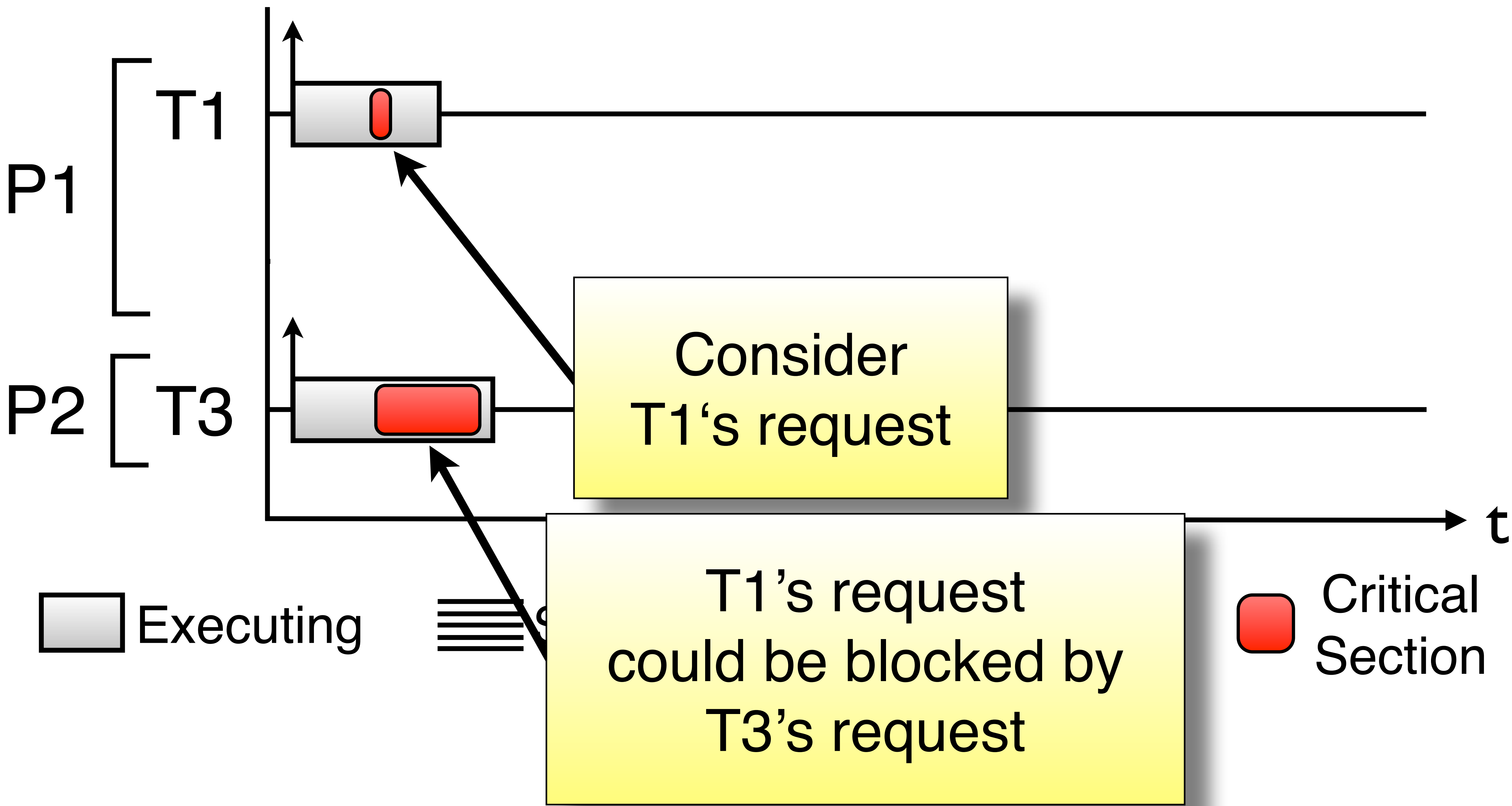
# Execution-Time Inflation in Classic MSRP Analysis



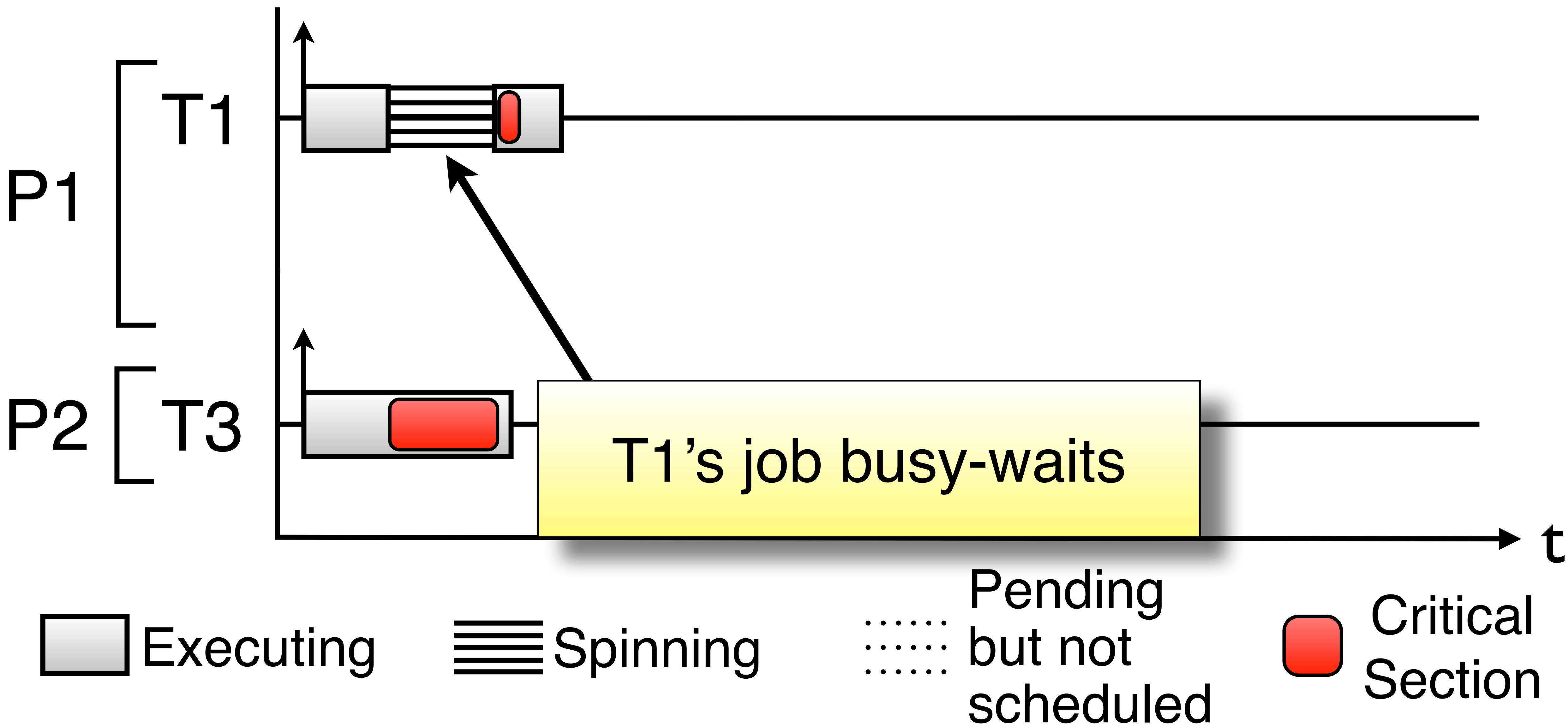
# Execution-Time Inflation in Classic MSRP Analysis



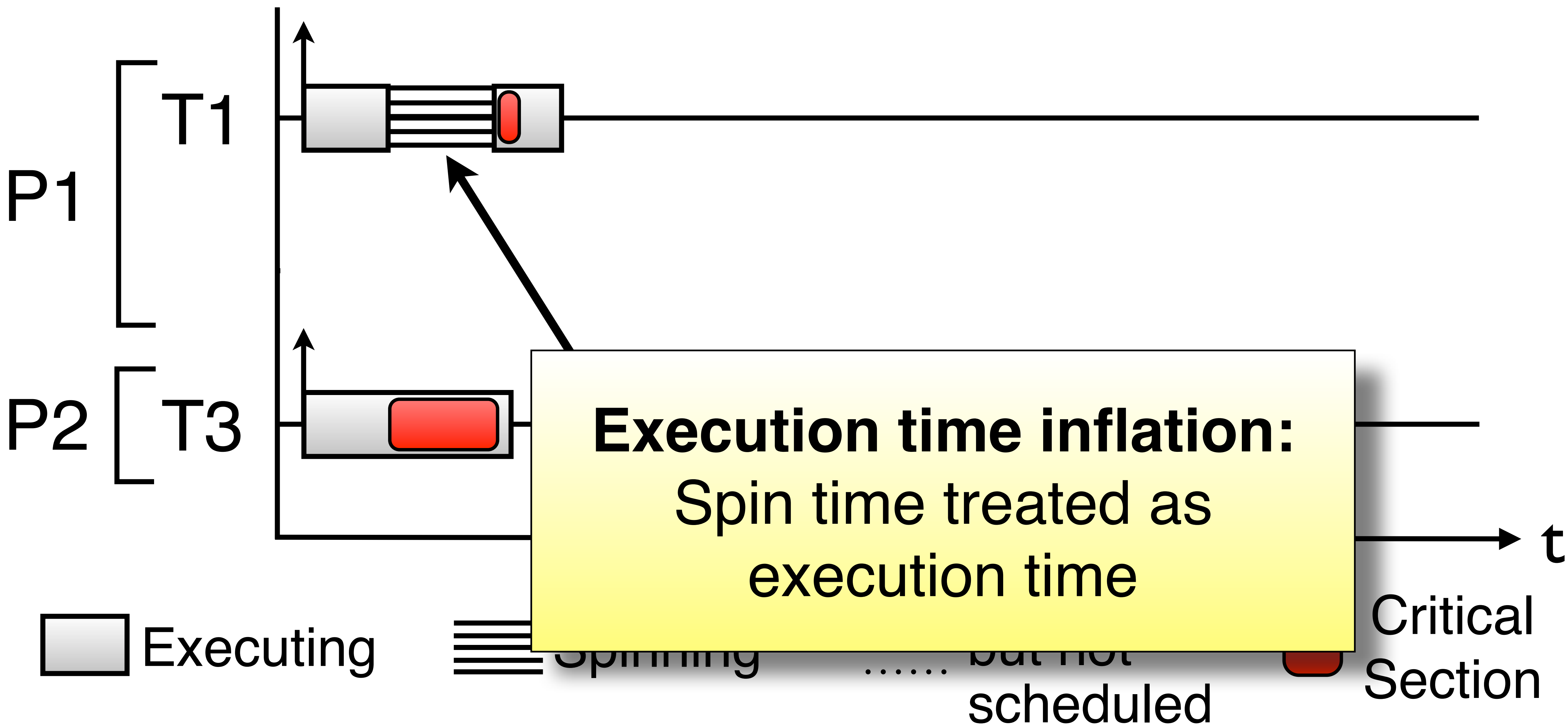
# Execution-Time Inflation in Classic MSRP Analysis



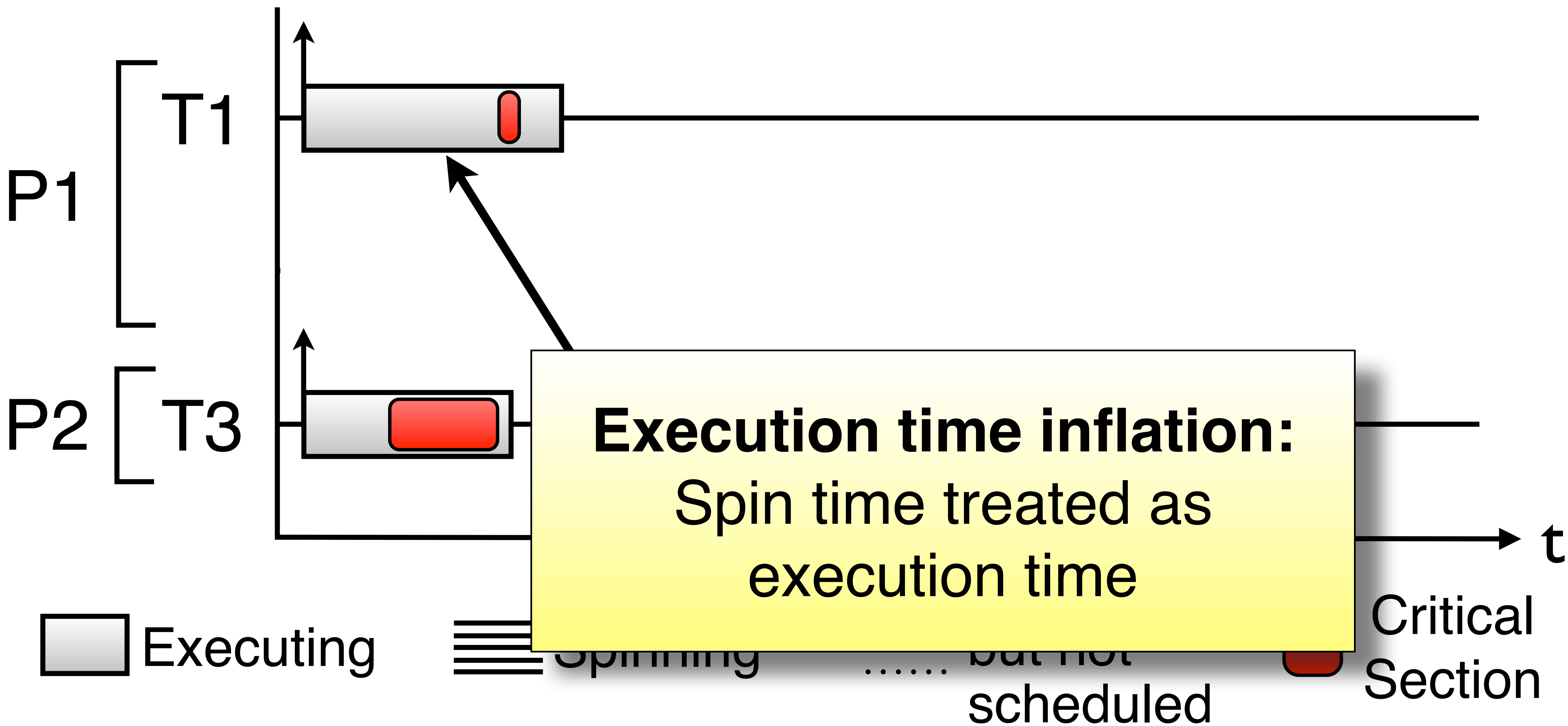
# Execution-Time Inflation in Classic MSRP Analysis



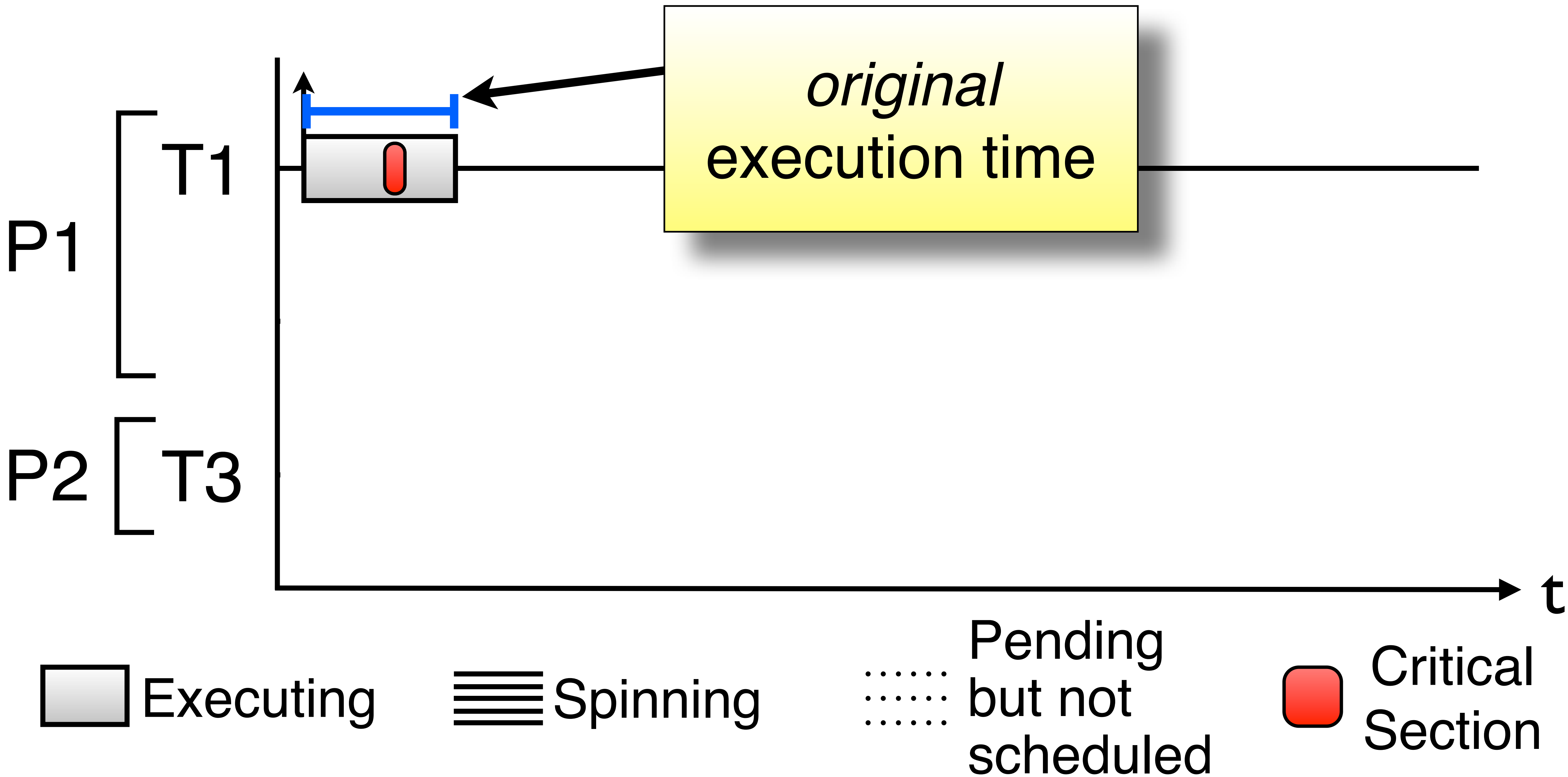
# Execution-Time Inflation in Classic MSRP Analysis



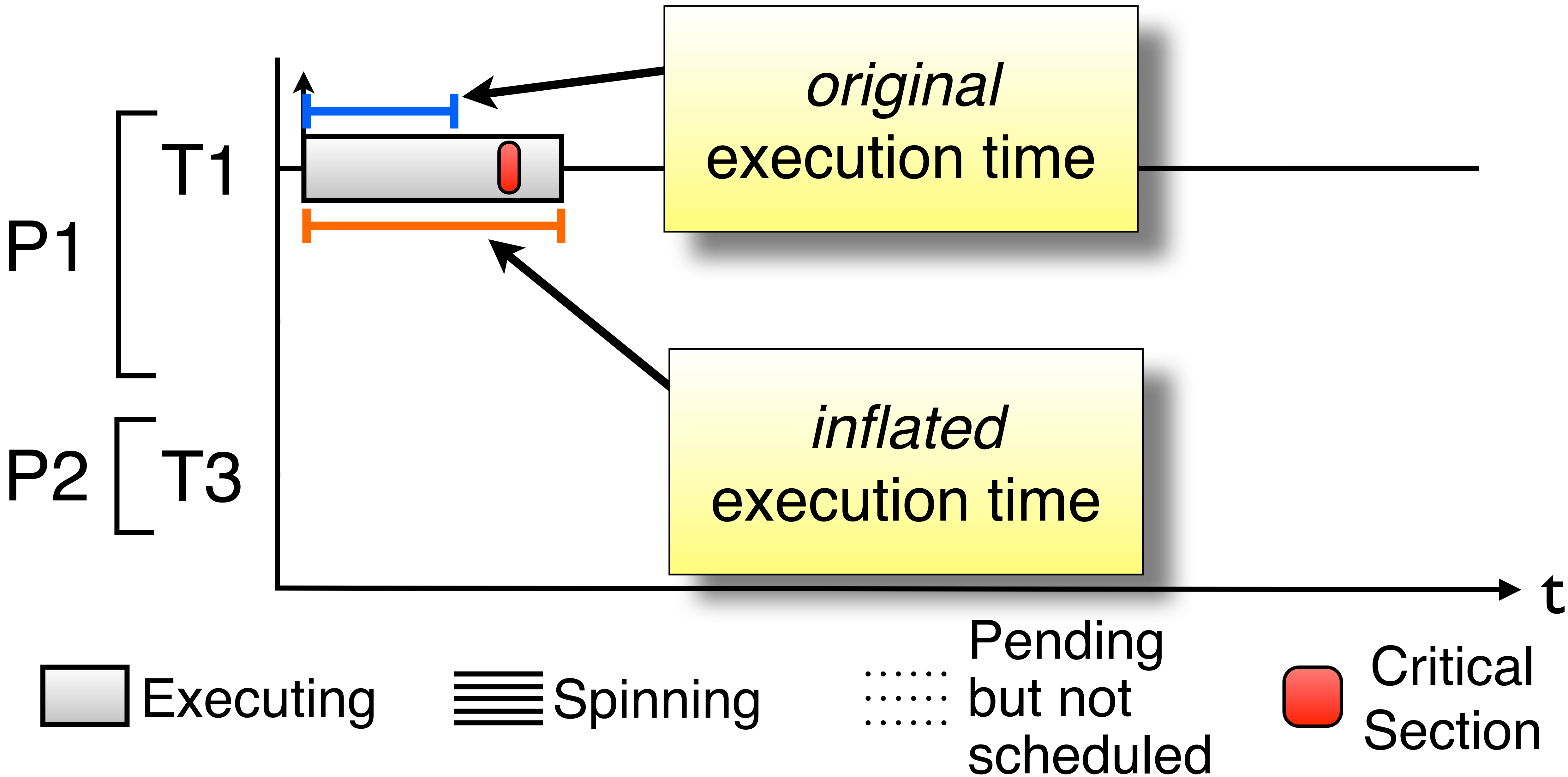
# Execution-Time Inflation in Classic MSRP Analysis



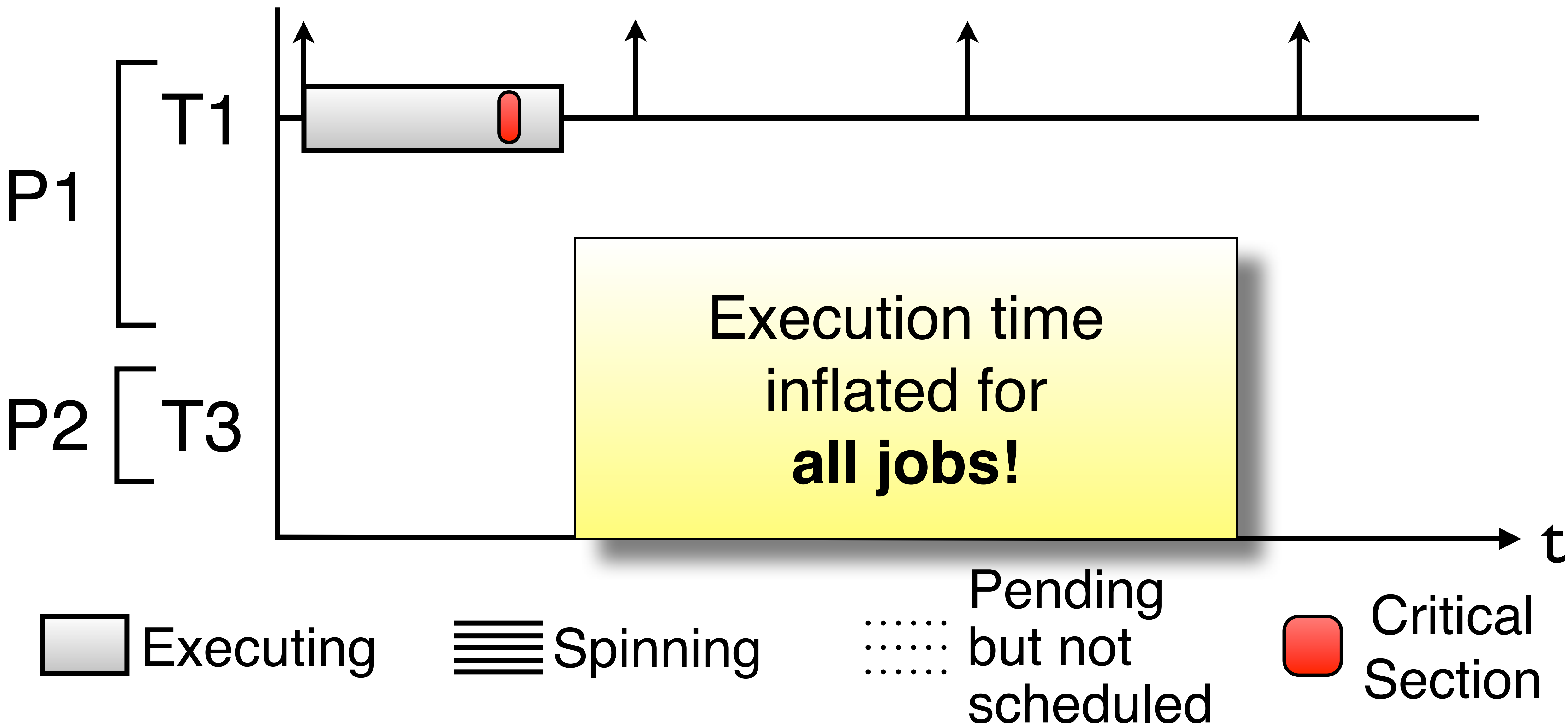
# Execution-Time Inflation in Classic MSRP Analysis



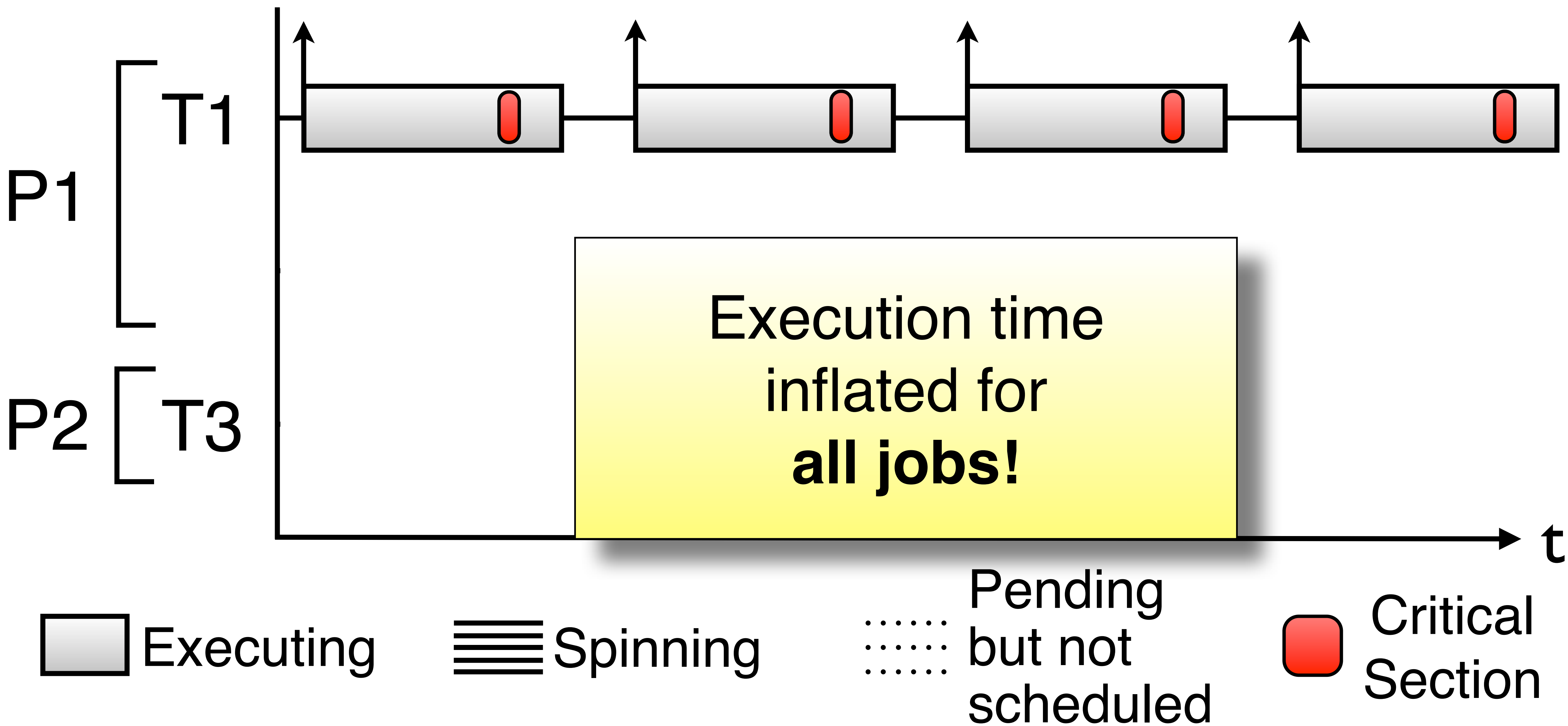
# Execution-Time Inflation in Classic MSRP Analysis



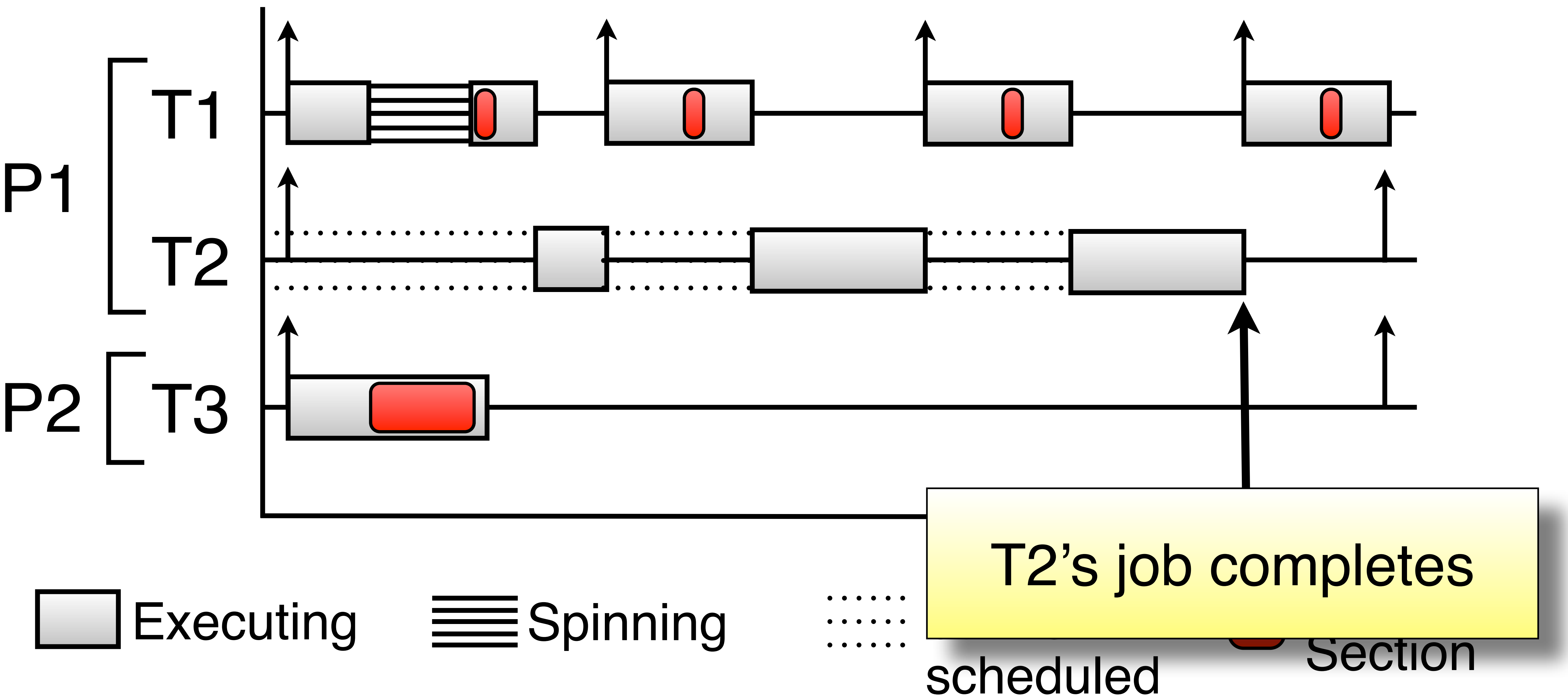
# Execution-Time Inflation in Classic MSRP Analysis



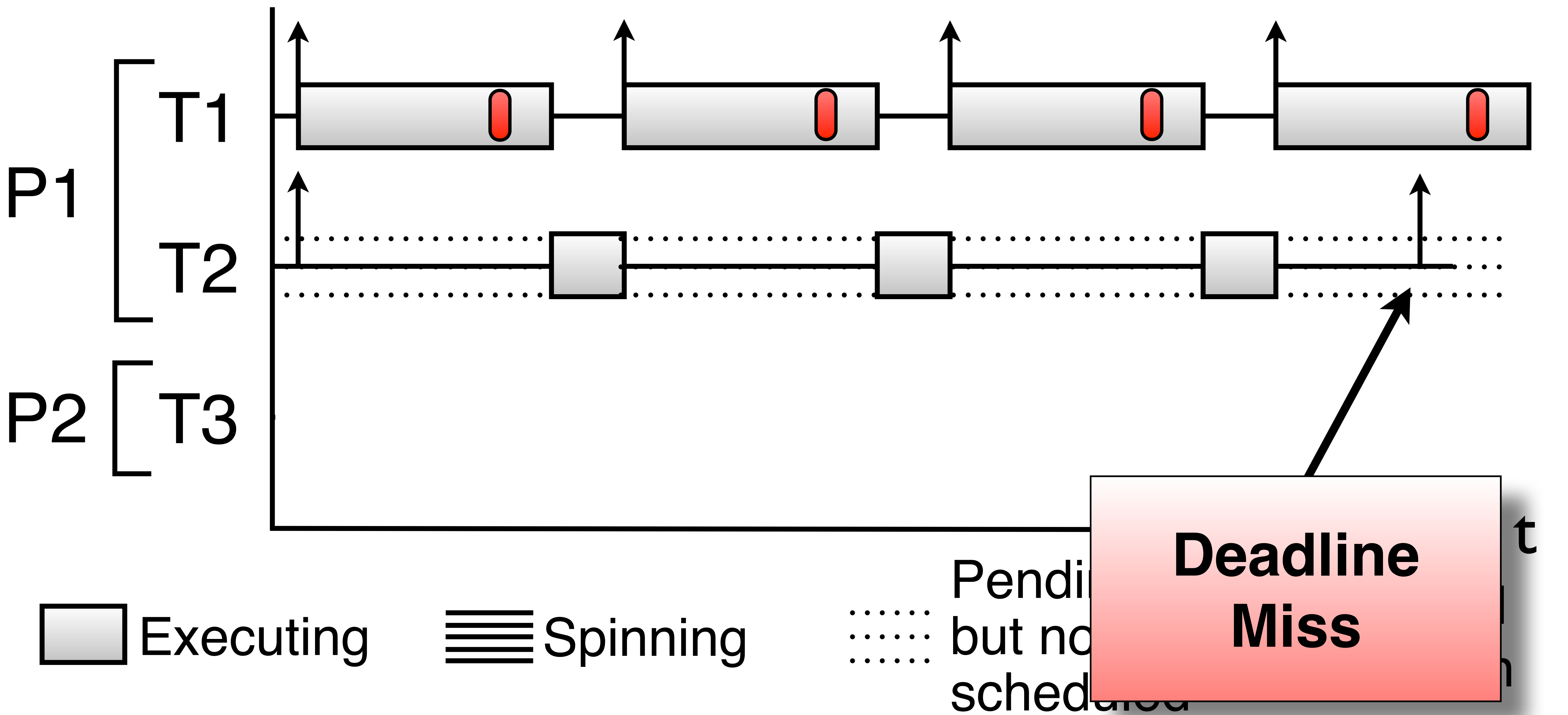
# Execution-Time Inflation in Classic MSRP Analysis



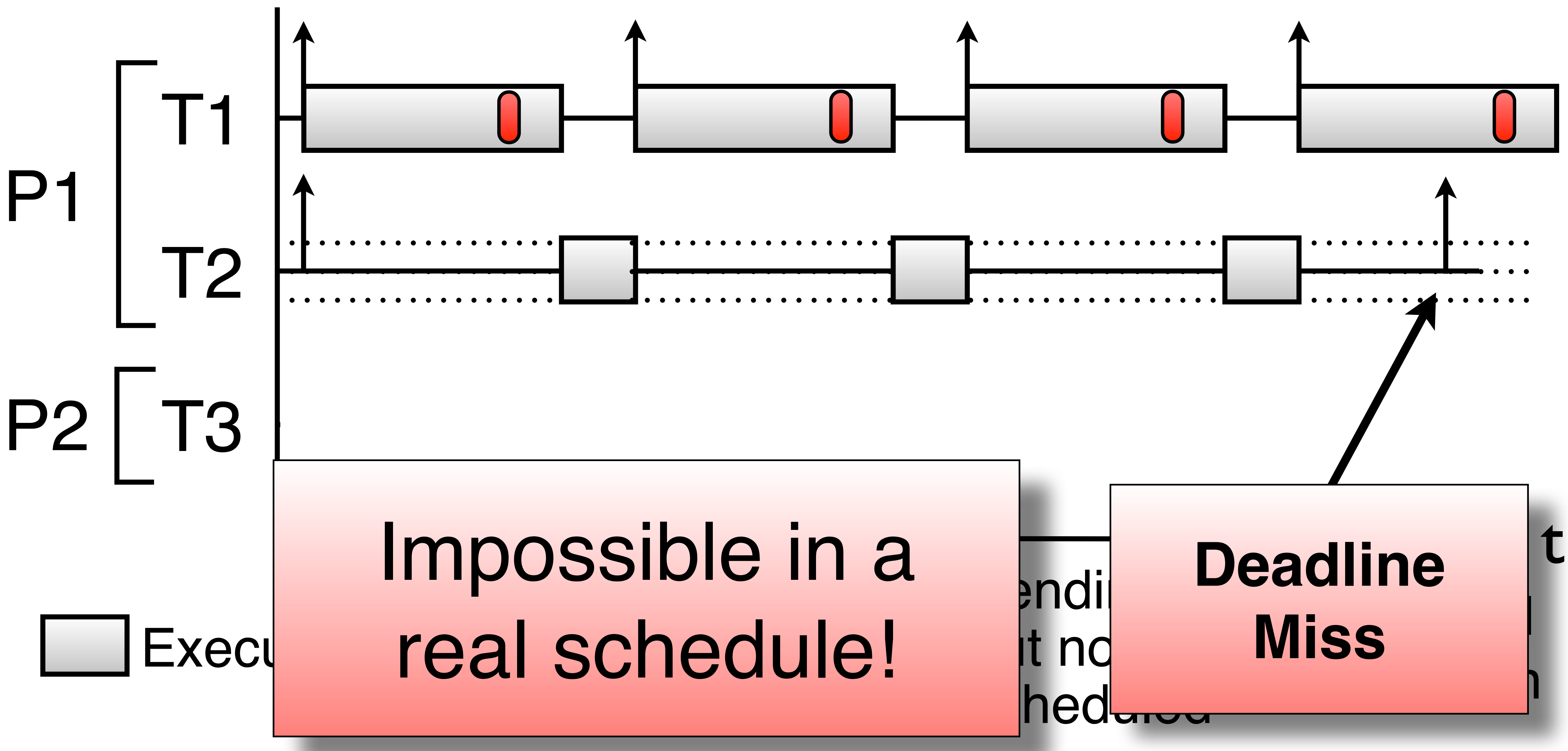
# Original Schedule



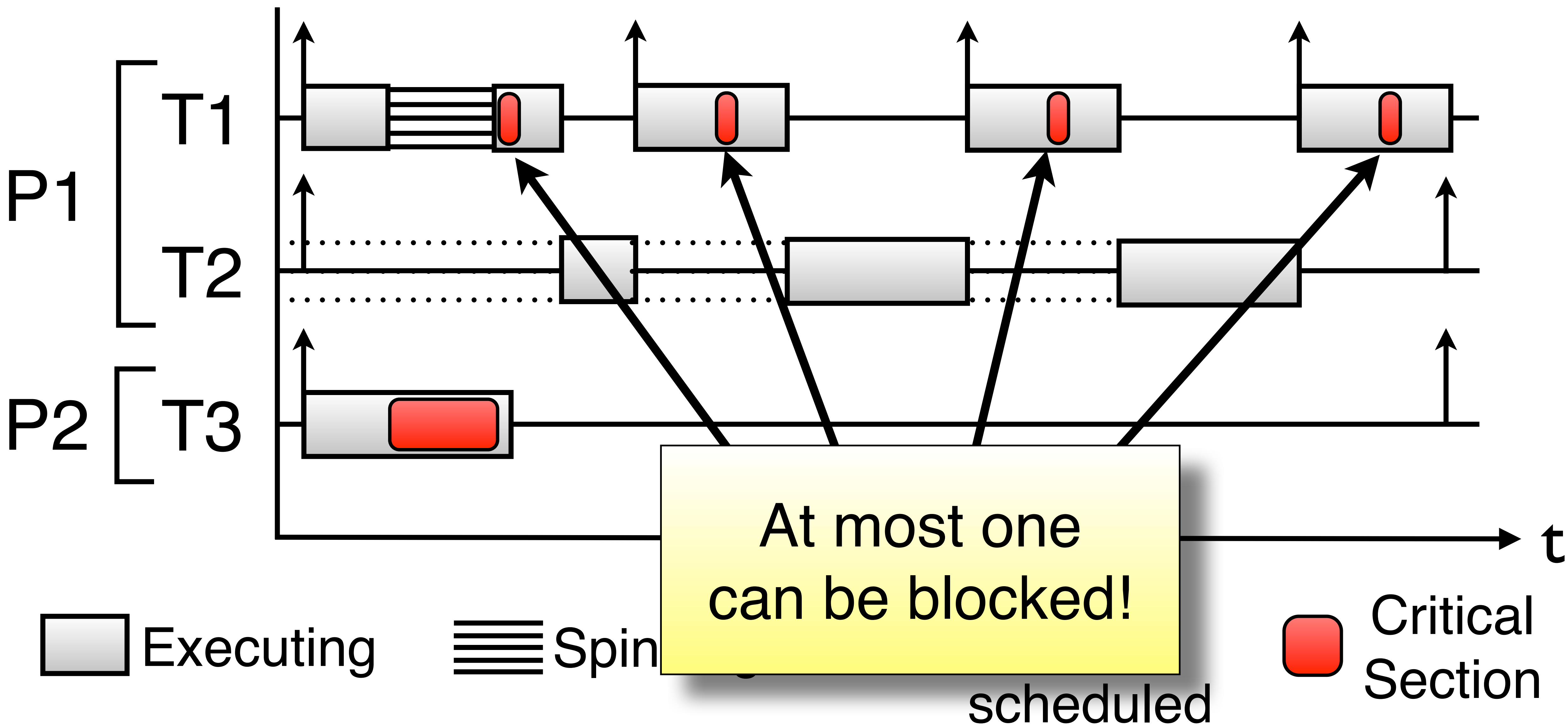
# Schedule with Inflated Execution Times



# Schedule with Inflated Execution Times



# Original Schedule



# Inflation is Inherently Pessimistic

All prior analyses rely on execution time inflation!

# Inflation is Inherently Pessimistic

All prior analyses rely on execution time inflation!

We show that
**execution time inflation**
is an
**inherent source of pessimism**
in blocking analysis.

# Inflation is Inherently Pessimistic

**Theorem**

Any blocking analysis relying on the inflation of job execution costs can be pessimistic by a factor of $\Omega(\phi \cdot n)$.

# Inflation is Inherently Pessimistic

**Theorem**

Any blocking analysis relying on the inflation of job execution costs can be pessimistic by a factor of $\Omega(\phi \cdot n)$.

maximal ratio of shortest and longest task period

number of tasks

# Inflation is Inherently Pessimistic

**Theorem**

Any blocking analysis relying on the inflation of job execution costs can be pessimistic by a factor of $\Omega(\phi \cdot n)$.

Details and proof in paper

# Challenges

Prior analysis is
**pessimistic**
**due to inflation**.

Prior analysis is **specific**
**to non-preemptable**
**FIFO-ordered** spin locks.

# Challenges

Prior analysis is
**pessimistic**
**due to inflation**.

ILP formulation
Explicit blocking terms

Prior analysis is **specific**
**to non-preemptable**
**FIFO-ordered** spin locks.

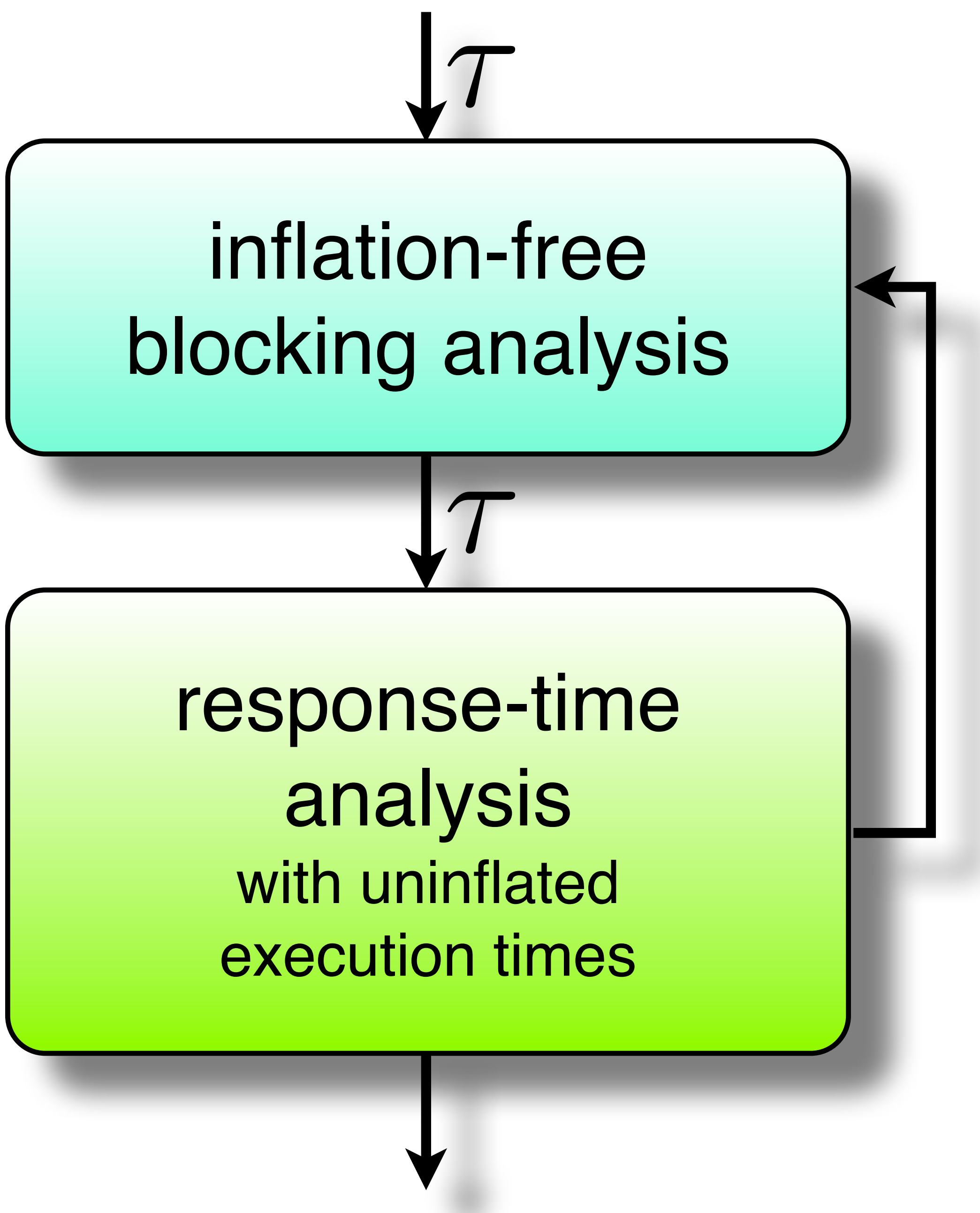
# Challenges

Prior analysis is **pessimistic due to inflation**.

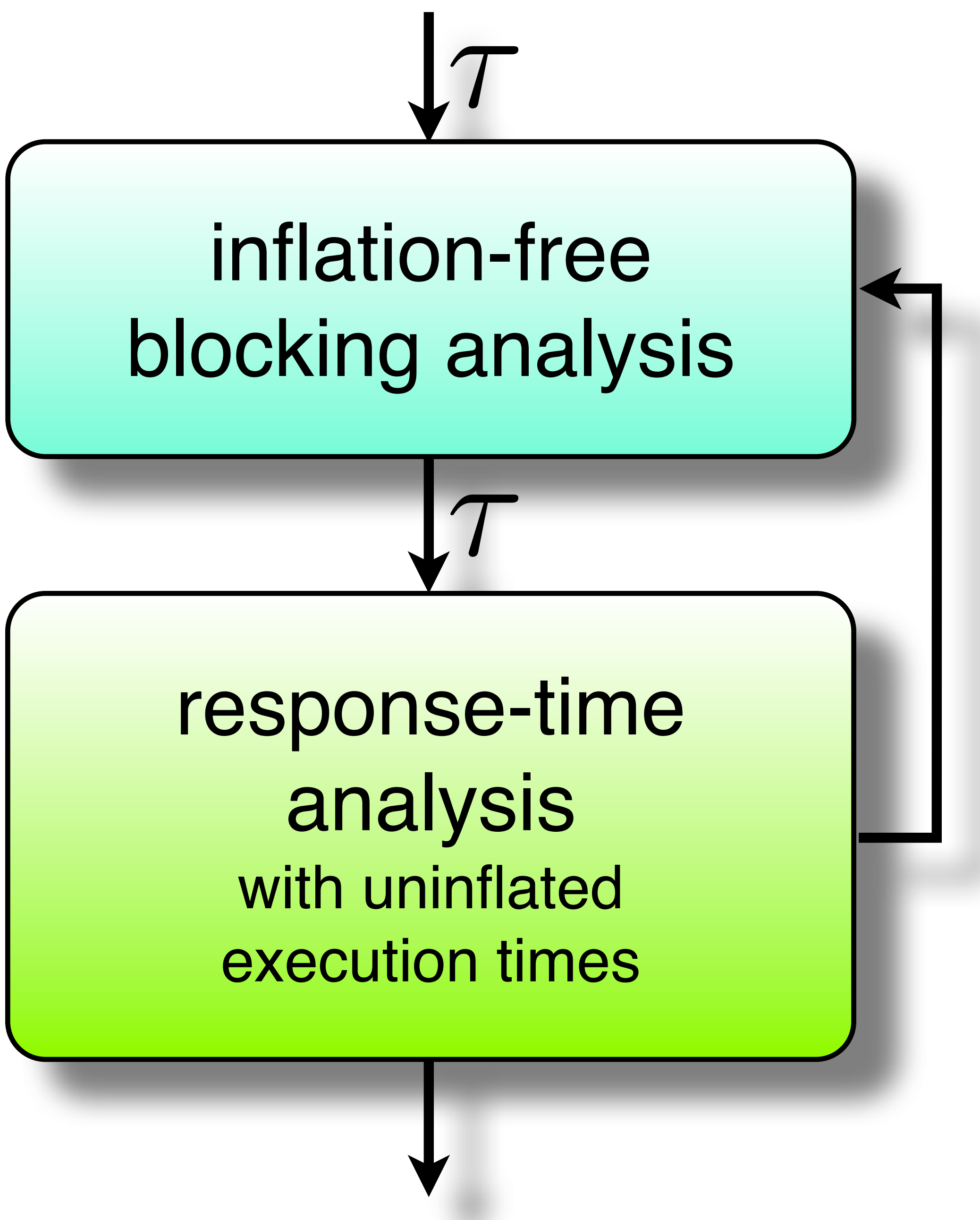
ILP formulation
Explicit blocking terms

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

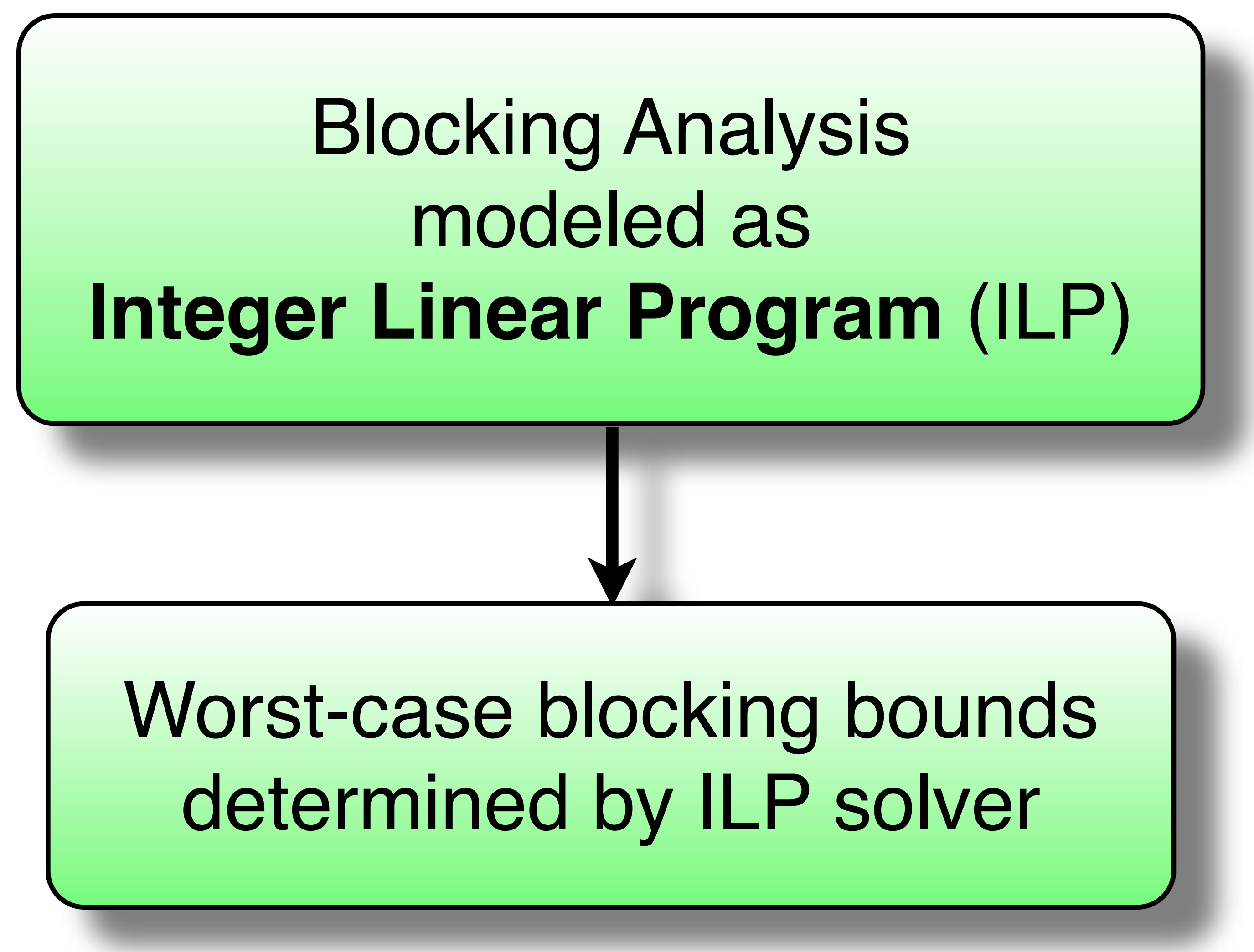
# ILP-Based Blocking Analysis of Spin Locks



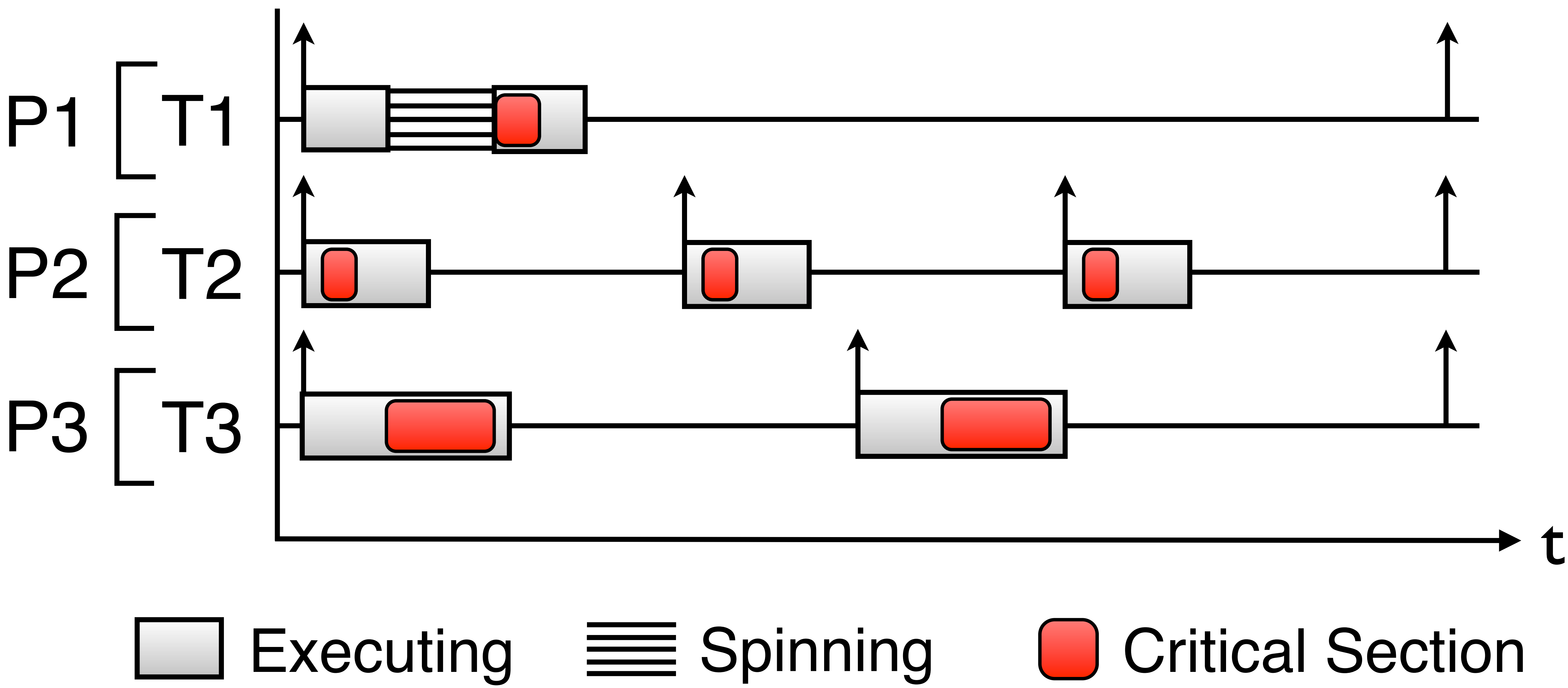
# ILP-Based Blocking Analysis of Spin Locks



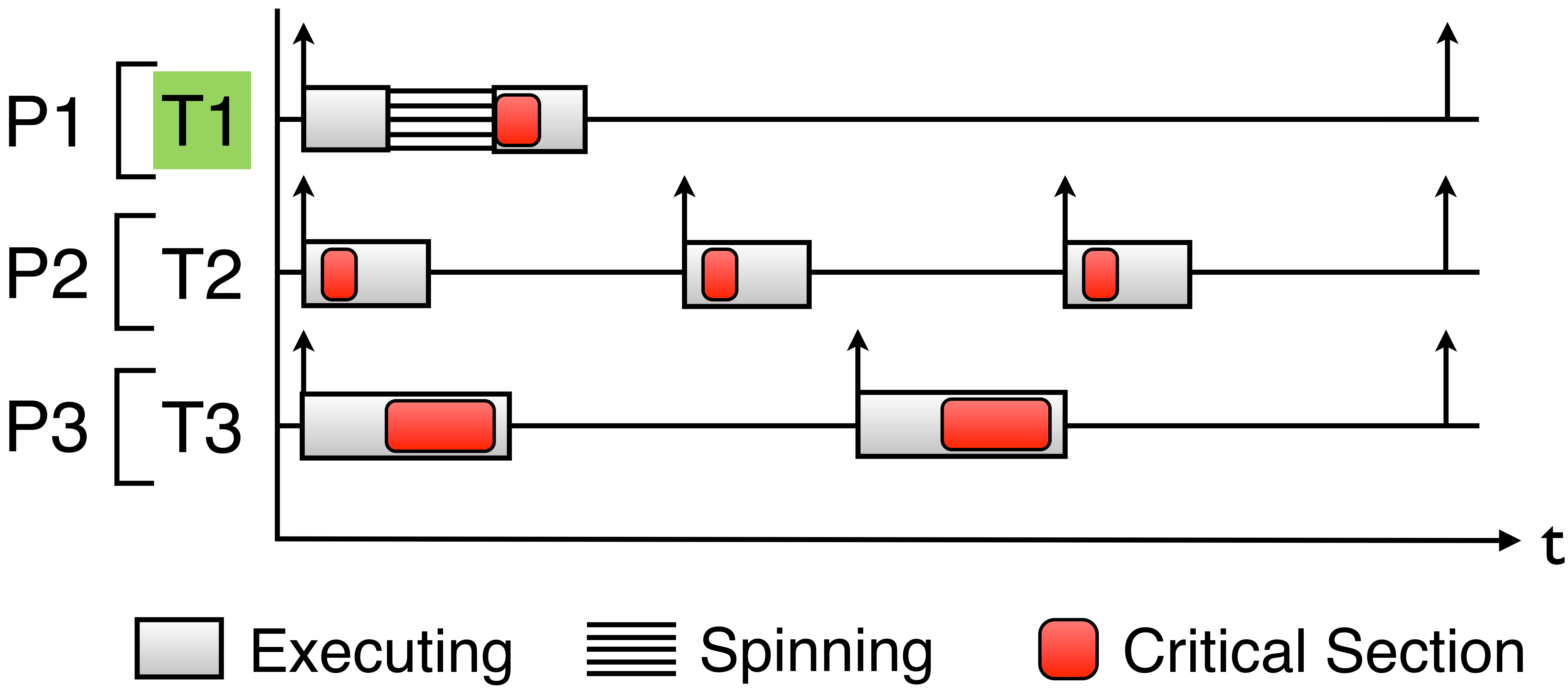

Key technique for inflation-free blocking analysis:



# ILP Generation for FIFO-Ordered Spin Locks

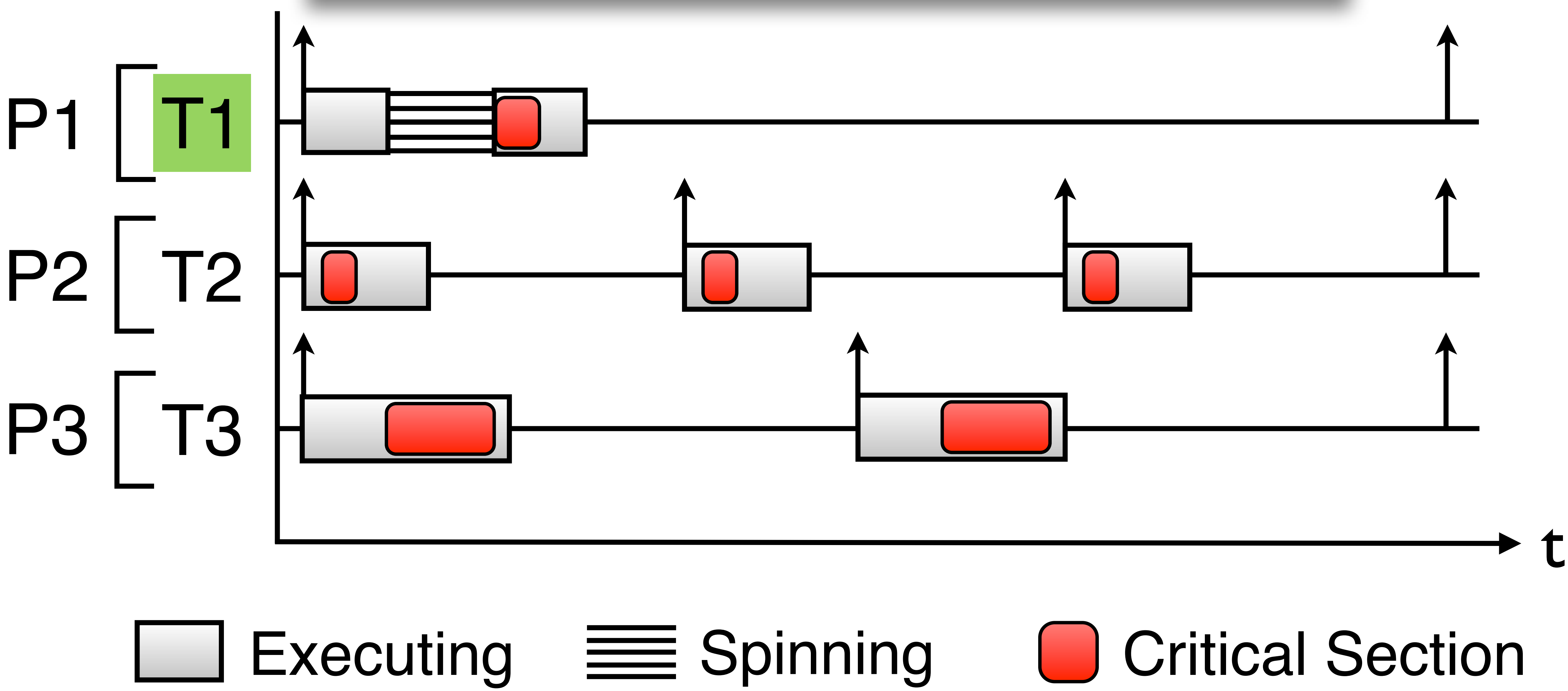


# ILP Generation for FIFO-Ordered Spin Locks



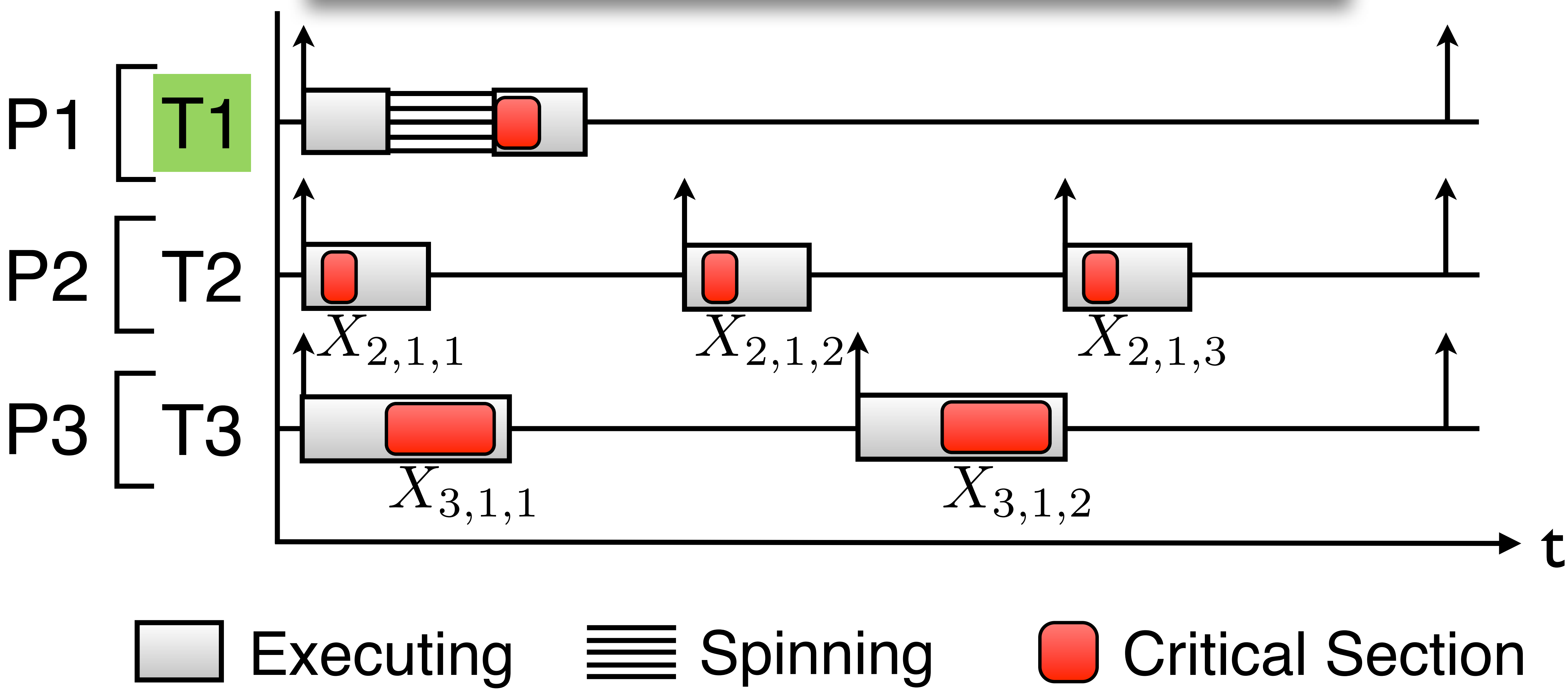
# ILP Generation for FIFO-Ordered Spin Locks

Assign *blocking variables*:



# ILP Generation for FIFO-Ordered Spin Locks

Assign *blocking variables*:



# Blocking Variables

$$0 \leq X \leq 1$$

$X:$ Fraction of critical section length contributing to T1's blocking

# Blocking Variables

$$0 \leq X \leq 1$$

$X :$ Fraction of critical section length contributing to T1's blocking

$X = 0$ Request **does not contribute** to T1's blocking.

$X = 1$ Request **contributes** to T1's blocking.

# Blocking Variables

$$0 \leq X \leq 1$$

Analysis accounts
**at most once**
for each request

$X$ … length
… cking

$X = 0$ … to T1's blocking.

$X = 1$

Request
**contributes**
to T1's blocking.

# Blocking Variables

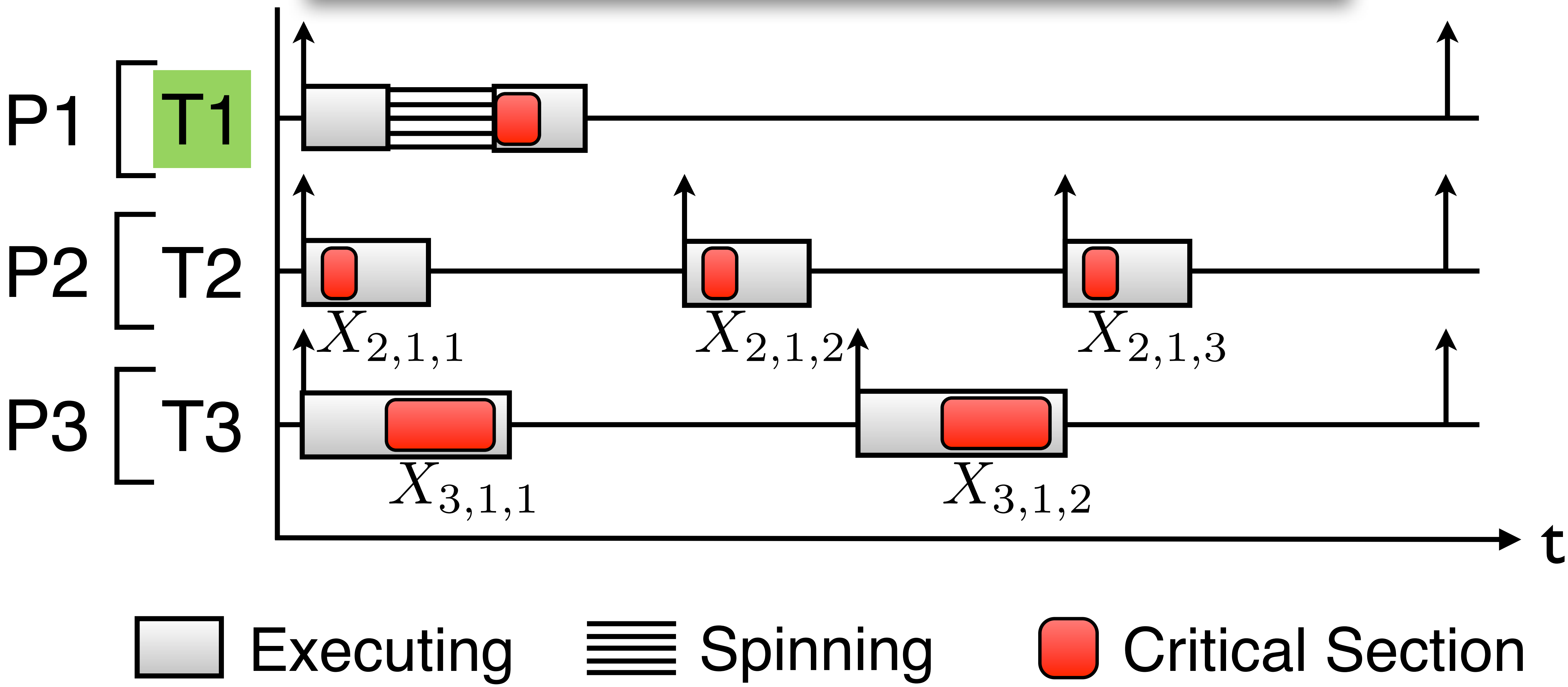
$$0 \leq X \leq 1$$

Analysis accounts
**at most once**
for each request

No double counting!

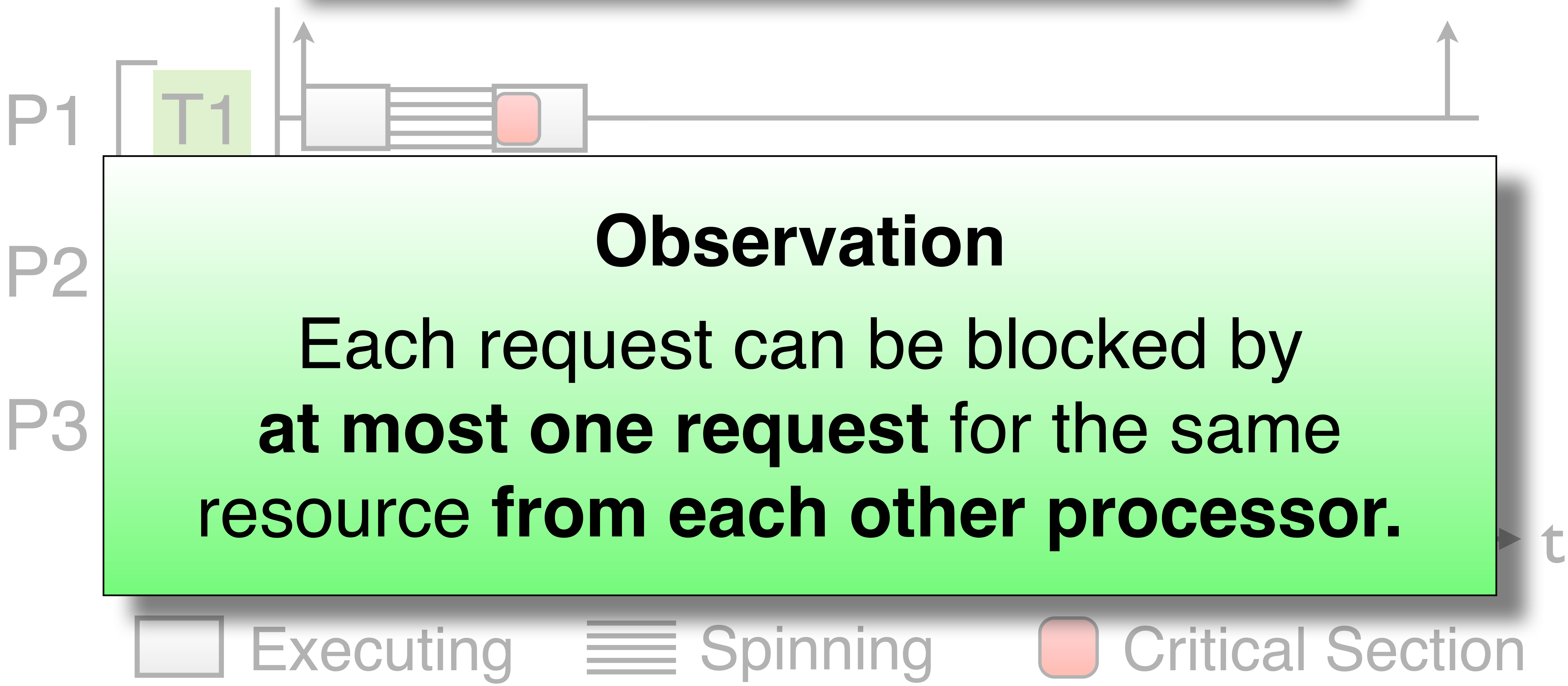
# ILP Generation for FIFO-Ordered Spin Locks

Impose constraints:



# ILP Generation for FIFO-Ordered Spin Locks

Impose constraints:

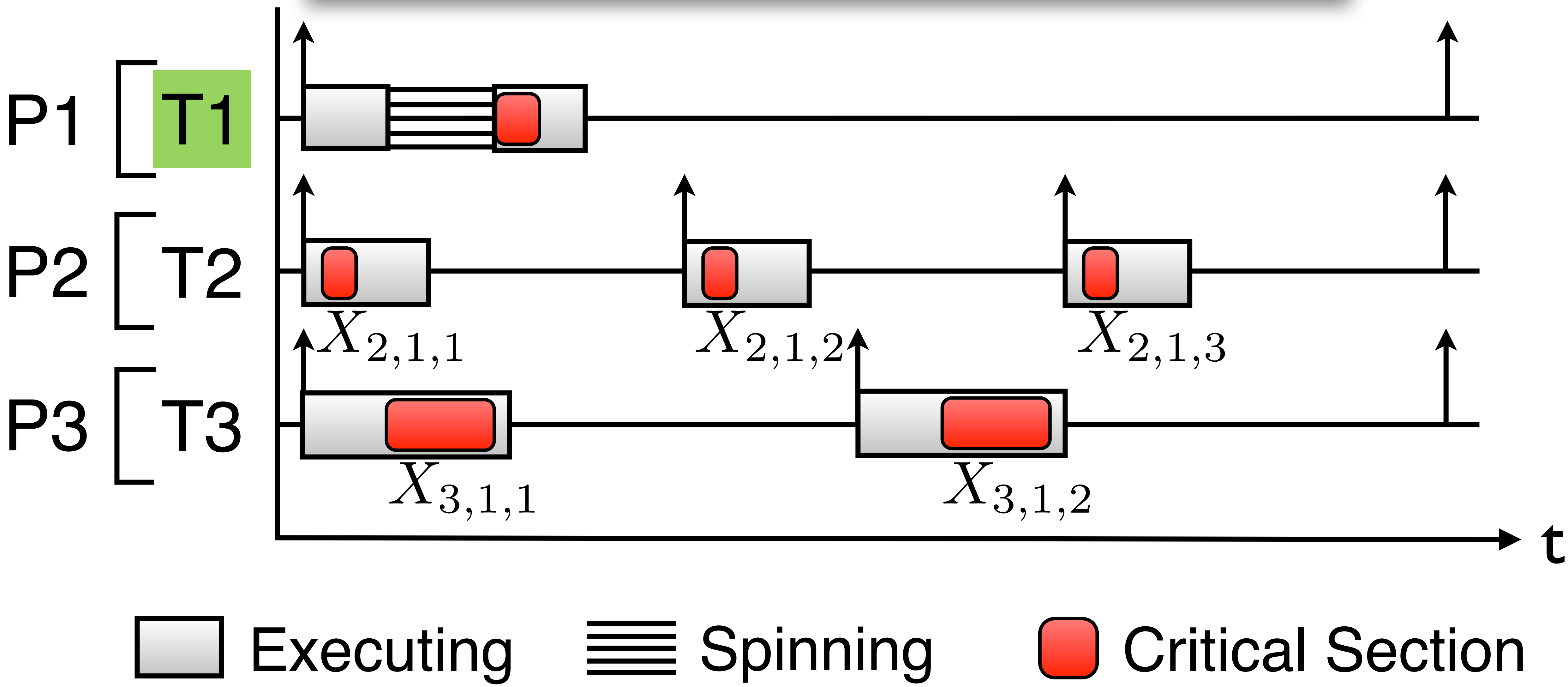



**Observation**

Each request can be blocked by **at most one request** for the same resource **from each other processor.**

# ILP Generation for FIFO-Ordered Spin Locks

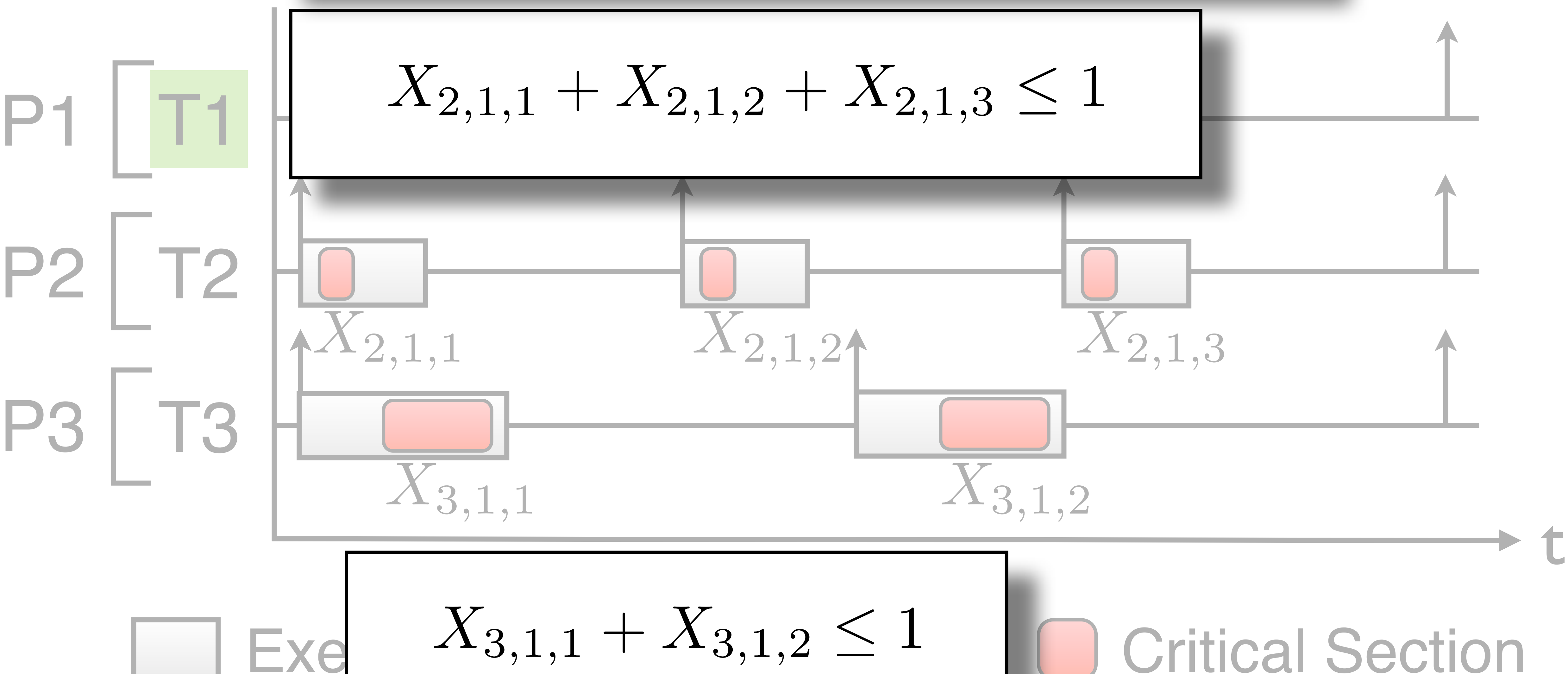
Impose constraints:



# ILP Generation for FIFO-Ordered Spin Locks

**Impose constraints:**

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$




$$X_{3,1,1} + X_{3,1,2} \leq 1$$

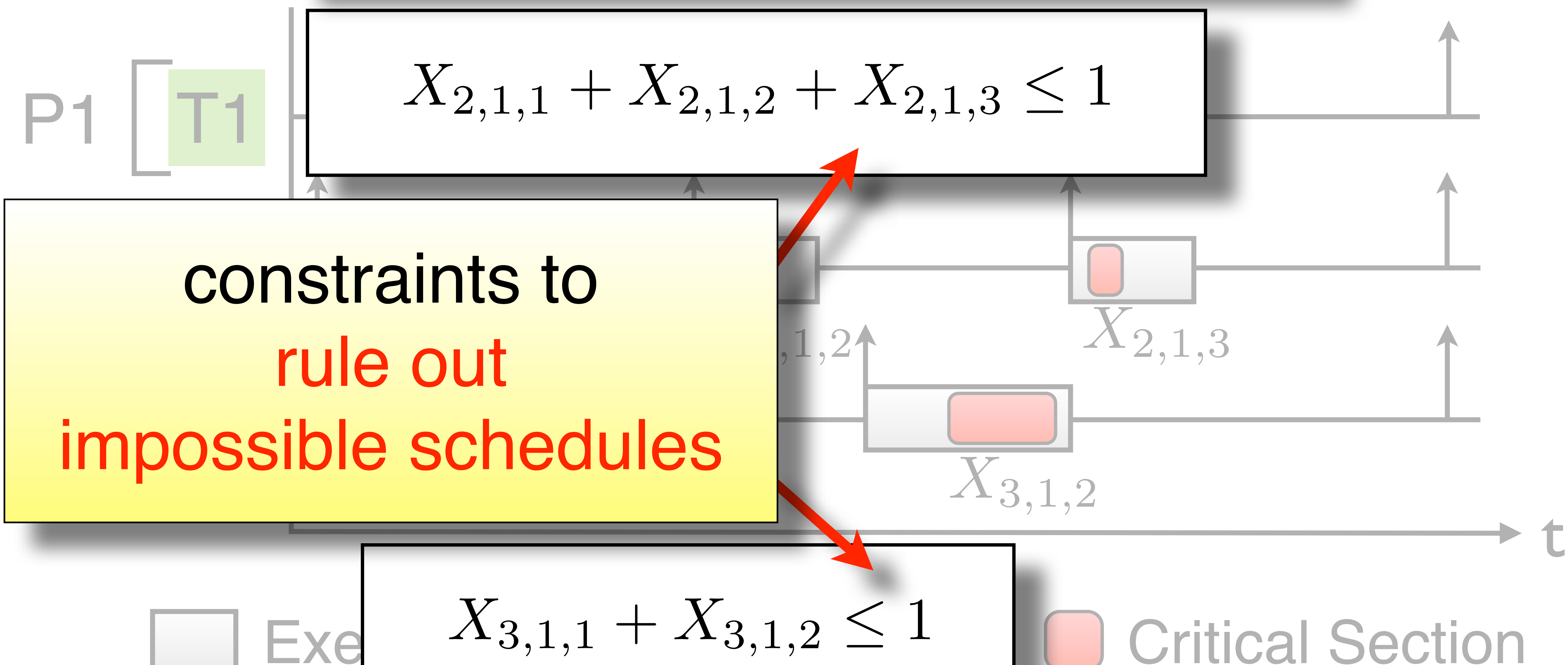
# ILP Generation for FIFO-Ordered Spin Locks

Impose constraints:

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

constraints to
rule out
impossible schedules

$$X_{3,1,1} + X_{3,1,2} \leq 1$$



# ILP Generation for FIFO-Ordered Spin Locks

Impose constraints:

P1 T1

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

constraints to
rule out
impossible schedules

**ILP Constraints**

- simple
- composable

$$X_{3,1,1} + X_{3,1,2} \leq 1$$

Exe Critical Section

# ILP for Worst-Case Blocking

Generate Integer Linear Program:

$$\boxed{X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1}$$

$$\boxed{X_{3,1,1} + X_{3,1,2} \leq 1}$$

# ILP for Worst-Case Blocking

Generate Integer Linear Program:

worst-case blocking

$\simeq$

maximal blocking

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

$$X_{3,1,1} + X_{3,1,2} \leq 1$$

# ILP for Worst-Case Blocking

Generate Integer Linear Program:

**maximize**

$$X_{2,1,1} \cdot L_{2,1,1} + X_{2,1,2} \cdot L_{2,1,2} + X_{2,1,3} \cdot L_{2,1,3} + X_{3,1,1} \cdot L_{3,1,1} + X_{3,1,2} \cdot L_{3,1,2}$$

**subject to**

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

$$X_{3,1,1} + X_{3,1,2} \leq 1$$

# ILP for Worst-Case Blocking

Generate Integer Linear Program:

**maximize**

$$X_{2,1,1} \cdot L_{2,1,1} + X_{2,1,2} \cdot L_{2,1,2} + X_{2,1,3} \cdot L_{2,1,3} + X_{3,1,1} \cdot L_{3,1,1} + X_{3,1,2} \cdot L_{3,1,2}$$

maximal critical section length

**subject to**

$$X_{2,1,1} + \ldots$$

$$X_{3,1,1} + X_{3,1,2} \leq 1$$

# ILP for Worst-Case Blocking

Generate Integer Linear Program:

**maximize**

$$X_{2,\ldots} \quad \ldots L_{2,1,3}$$

$$+ \ldots$$

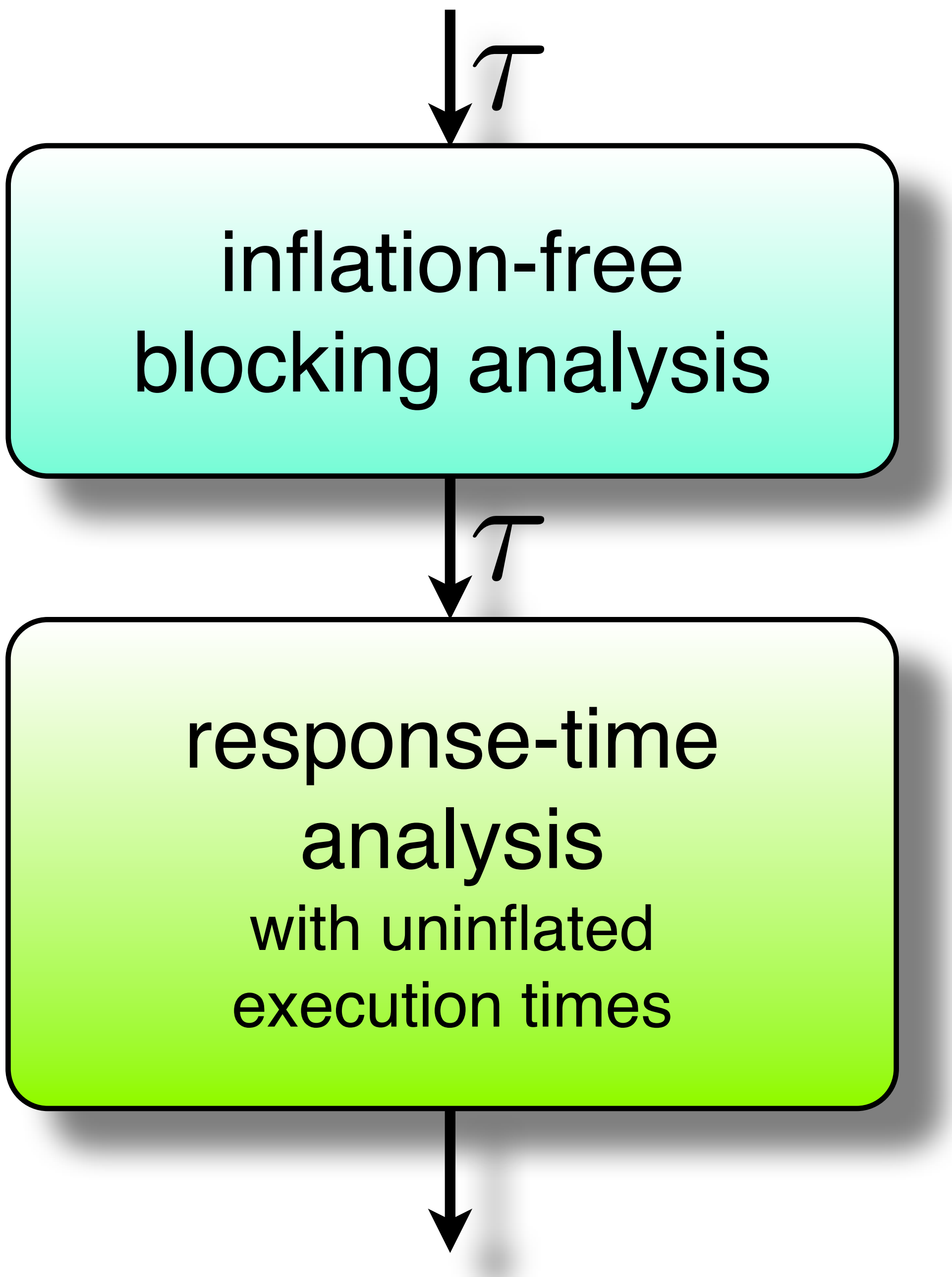
Worst-case blocking bound determined by ILP solver.

**subject to**

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

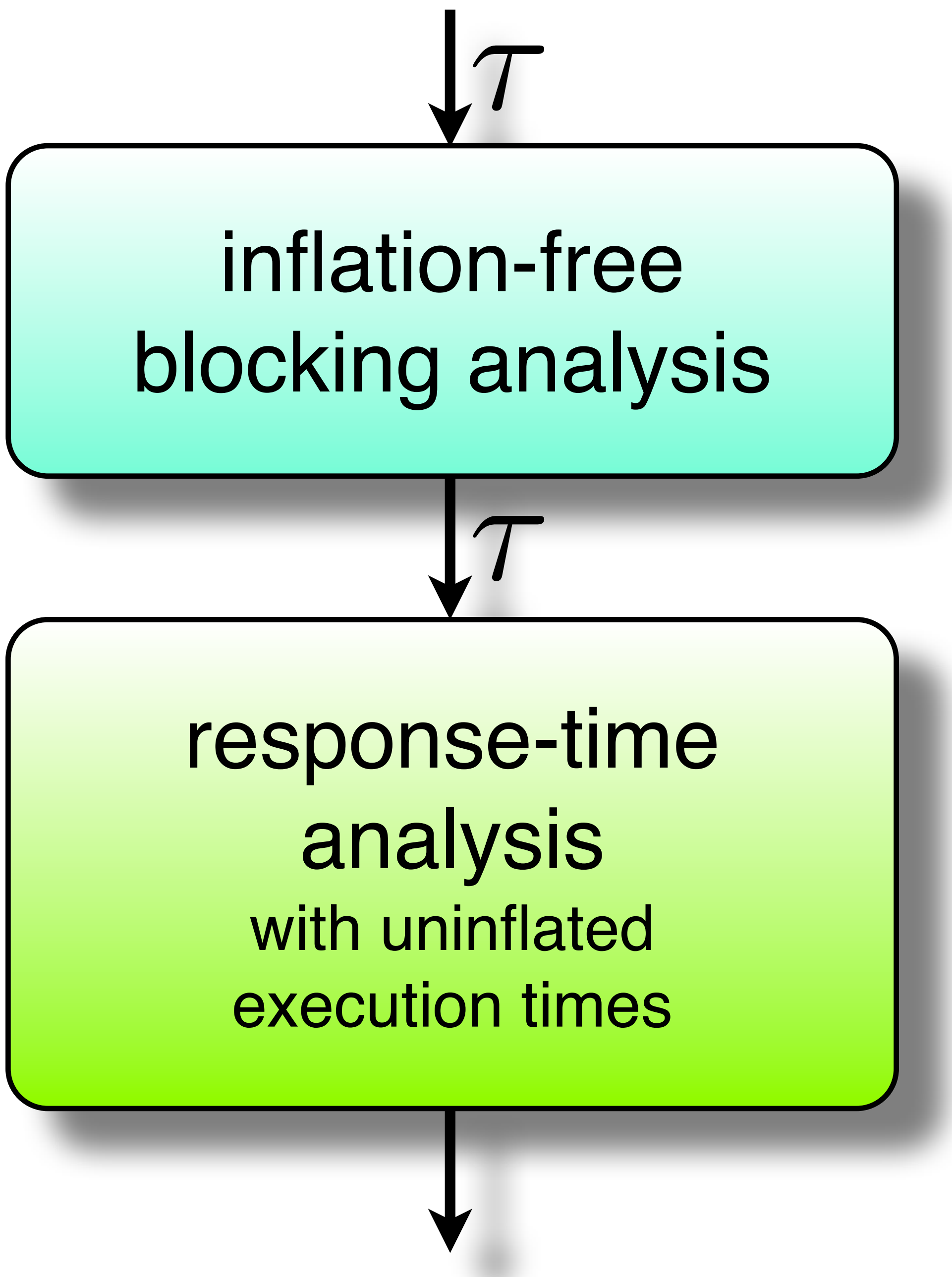
$$X_{3,1,1} + X_{3,1,2} \leq 1$$

# Explicit Accounting for Blocking




$$R_i = e_i + b_i + \sum_{h<i} \left\lceil \frac{R_i}{p_h} \right\rceil \cdot e_h$$

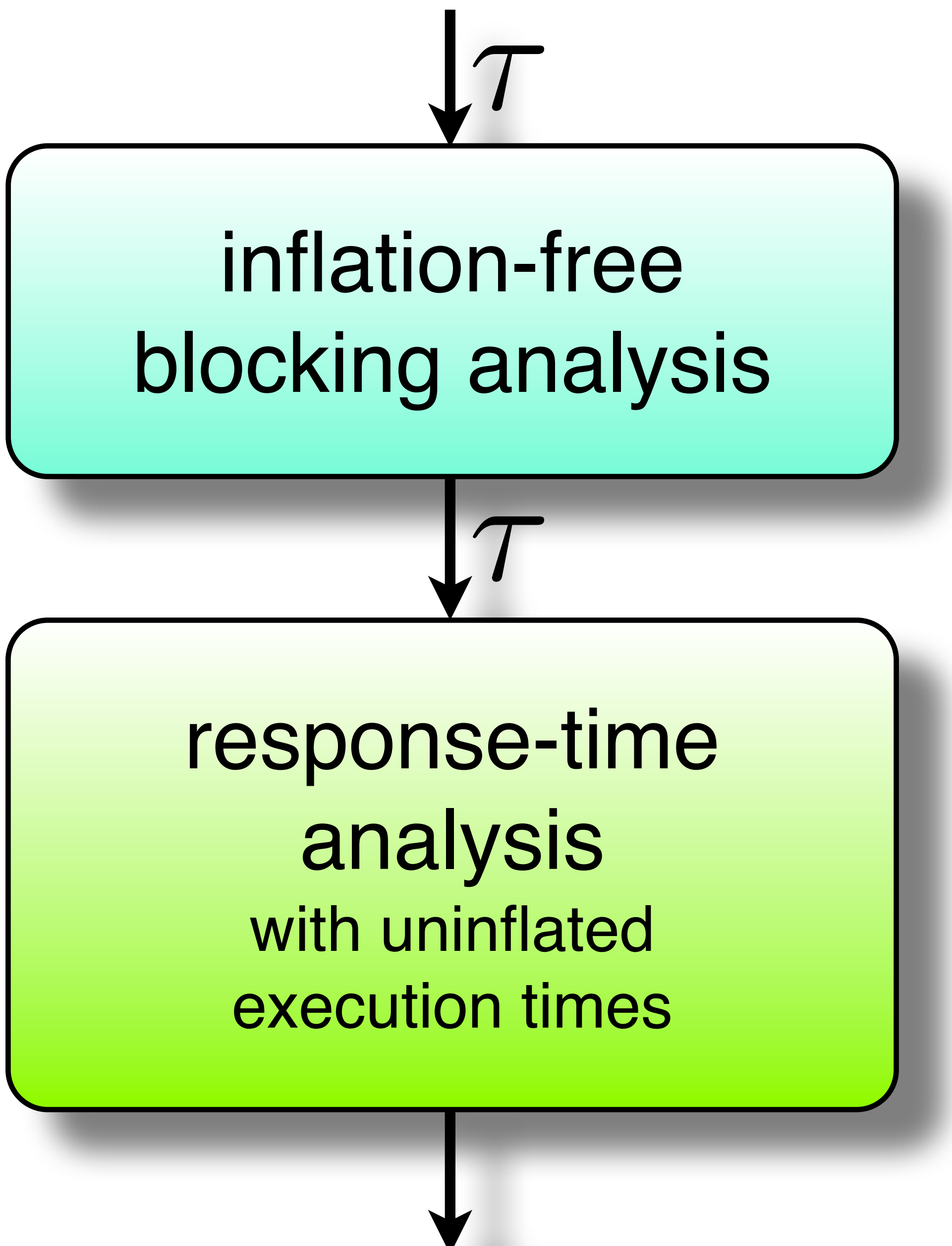
# Explicit Accounting for Blocking




**worst-case blocking bound**
determined by
ILP-solver

$$R_i = e_i + \boxed{b_i} + \sum_{h<i} \left\lceil \frac{R_i}{p_h} \right\rceil \cdot e_h$$

# Explicit Accounting for Blocking




**worst-case blocking bound** determined by ILP-solver

$$R_i = \boxed{e_i} + \boxed{b_i} + \sum_{h<i} \left\lceil \frac{R_i}{p_h} \right\rceil \cdot \boxed{e_h}$$

**uninflated** execution costs

# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

So far we only talked about the MSRP...

So far we only talked about the MSRP...

...but there's more.

# Spin Lock Types

Non-Preemptable Spinning

Preemptable Spinning

×

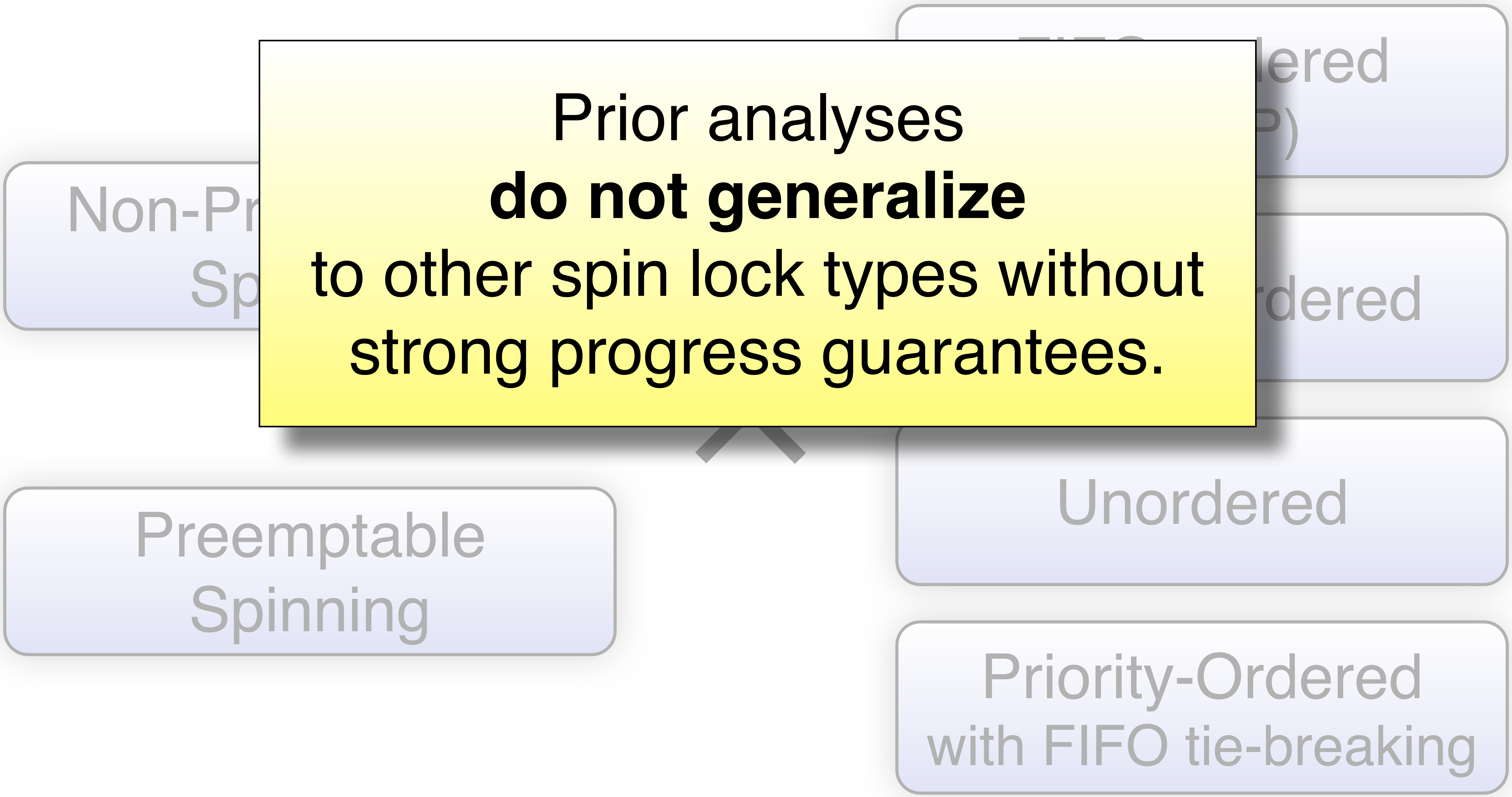
FIFO-ordered (MSRP)

Priority-Ordered

Unordered

Priority-Ordered with FIFO tie-breaking

# Spin Lock Types




Prior analyses
**do not generalize**
to other spin lock types without
strong progress guarantees.

# Prior Analyses Rely on Strong Progress Guarantees

FIFO-Ordering is analysis-friendly:

Each request can be blocked by
**at most one request** for the same
resource **from each other processor.**

# Prior Analyses Rely on Strong Progress Guarantees

FIFO-Ordering is analysis-friendly:

Each request can be blocked by **at most one request** for the same resource **from each other processor.**

**Prior analyses rely on strong progress guarantees** provided by FIFO-ordering.

# Prior Analyses Rely on Strong Progress Guarantees

Unordered spin locks:

Each request can be blocked by
**all other requests**
for the same resource.

# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms
ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms
ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

Wait-time bounds
Composable constraints

# Challenges

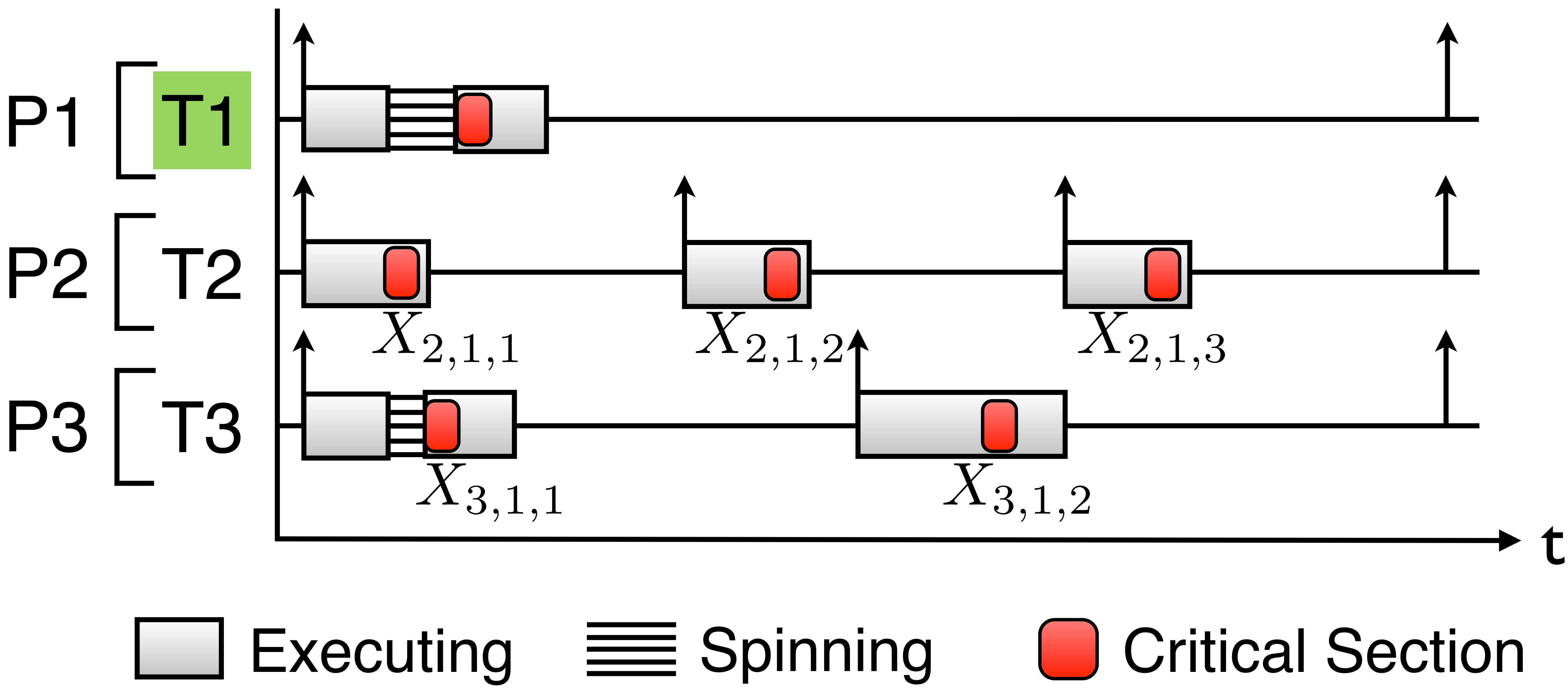
Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms
ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

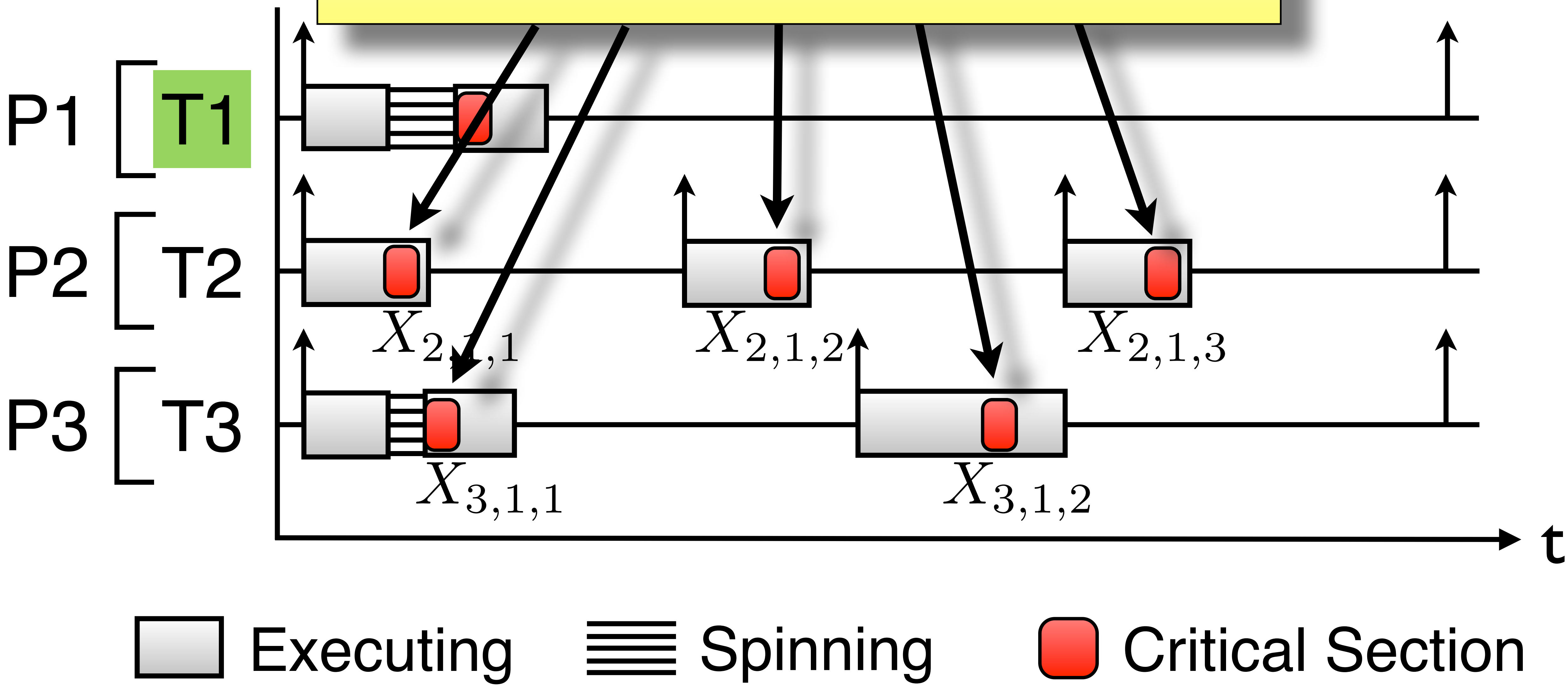
Wait-time bounds
Composable constraints

# Analysis of Unordered Spin Locks



# Analysis of Unordered Spin Locks

Can all of these really block T1's request?

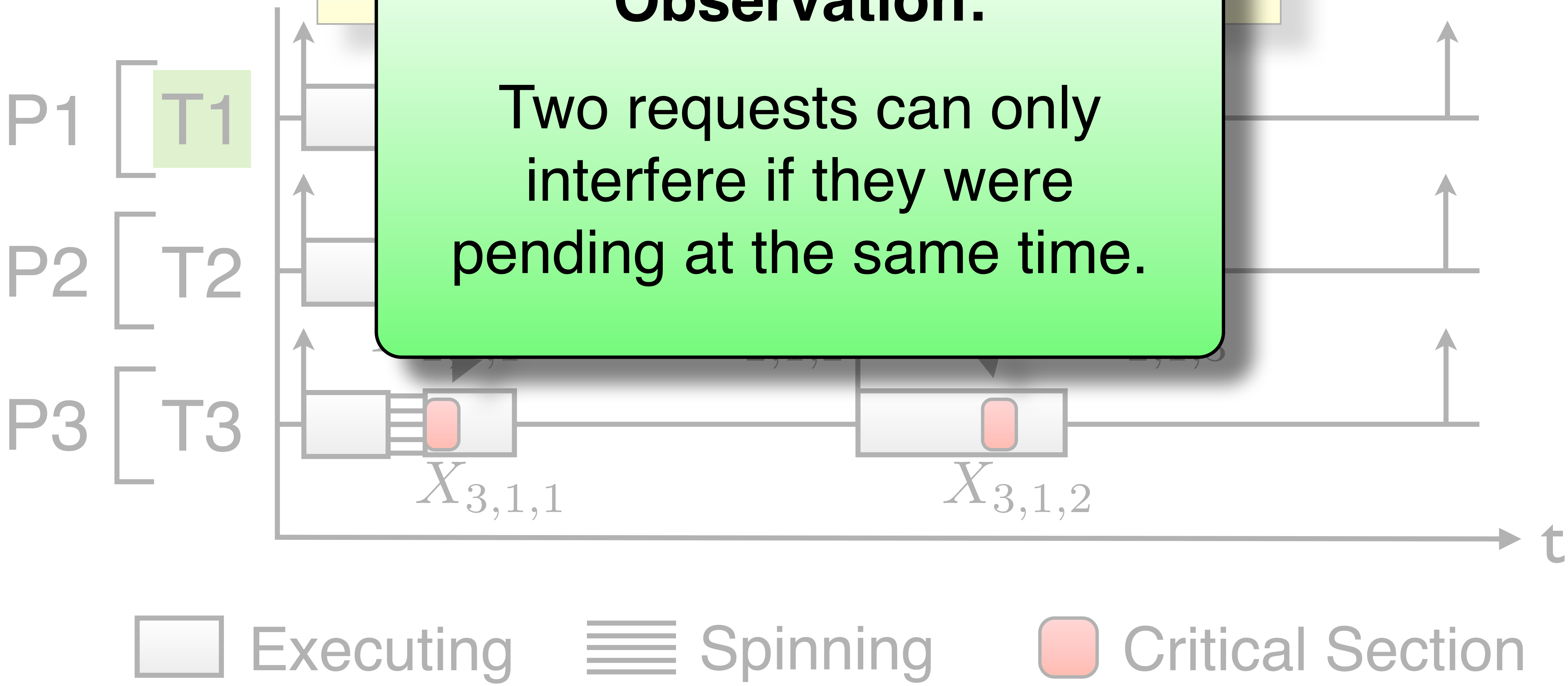


# Analysis of Unordered Spin Locks

Can all of these really

**Observation:**

Two requests can only interfere if they were pending at the same time.



# Analysis of Unordered Spin Locks

**Observation:**

Two requests can only interfere if they were pending at the same time.

# Per-Request Wait-Time Bounds

How many remote request can be pending while T1’s request is pending?

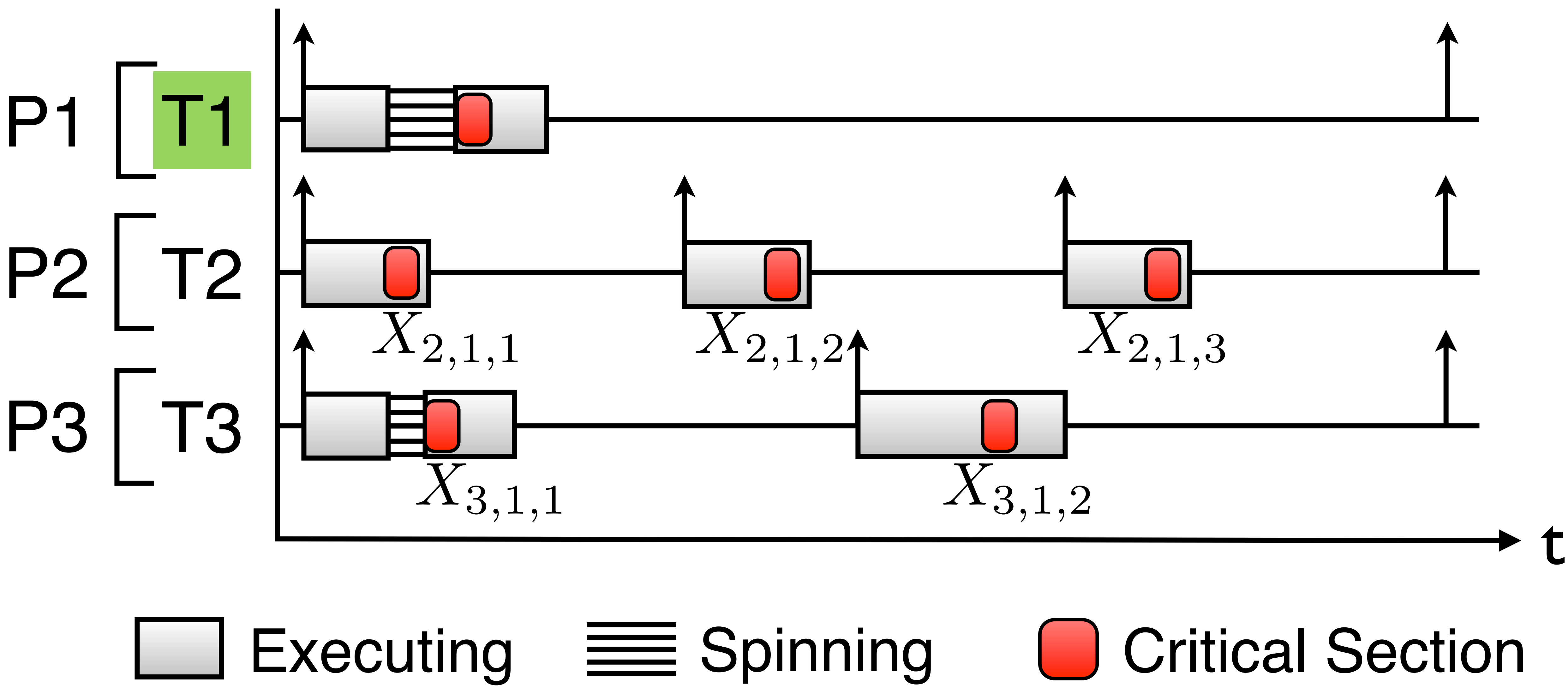
# Per-Request Wait-Time Bounds

How many remote request can be pending while T1's request is pending?

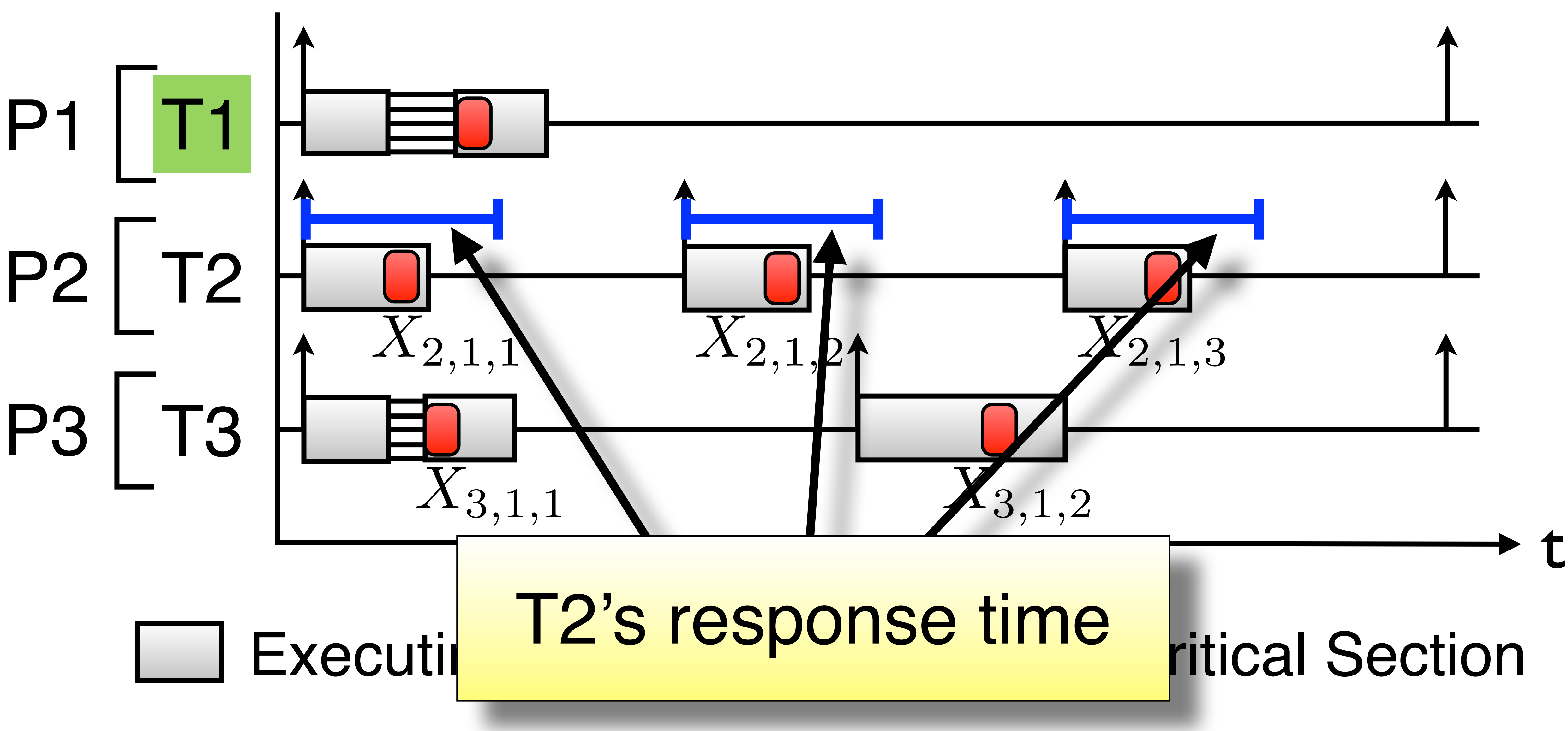
Bound **wait time** of T1's **request**. [1]

[1] K. Lakshmanan, D. Niz, and R. Rajkumar, "Coordinated task scheduling, allocation and synchronization on multiprocessors," in RTSS'09, 2009.

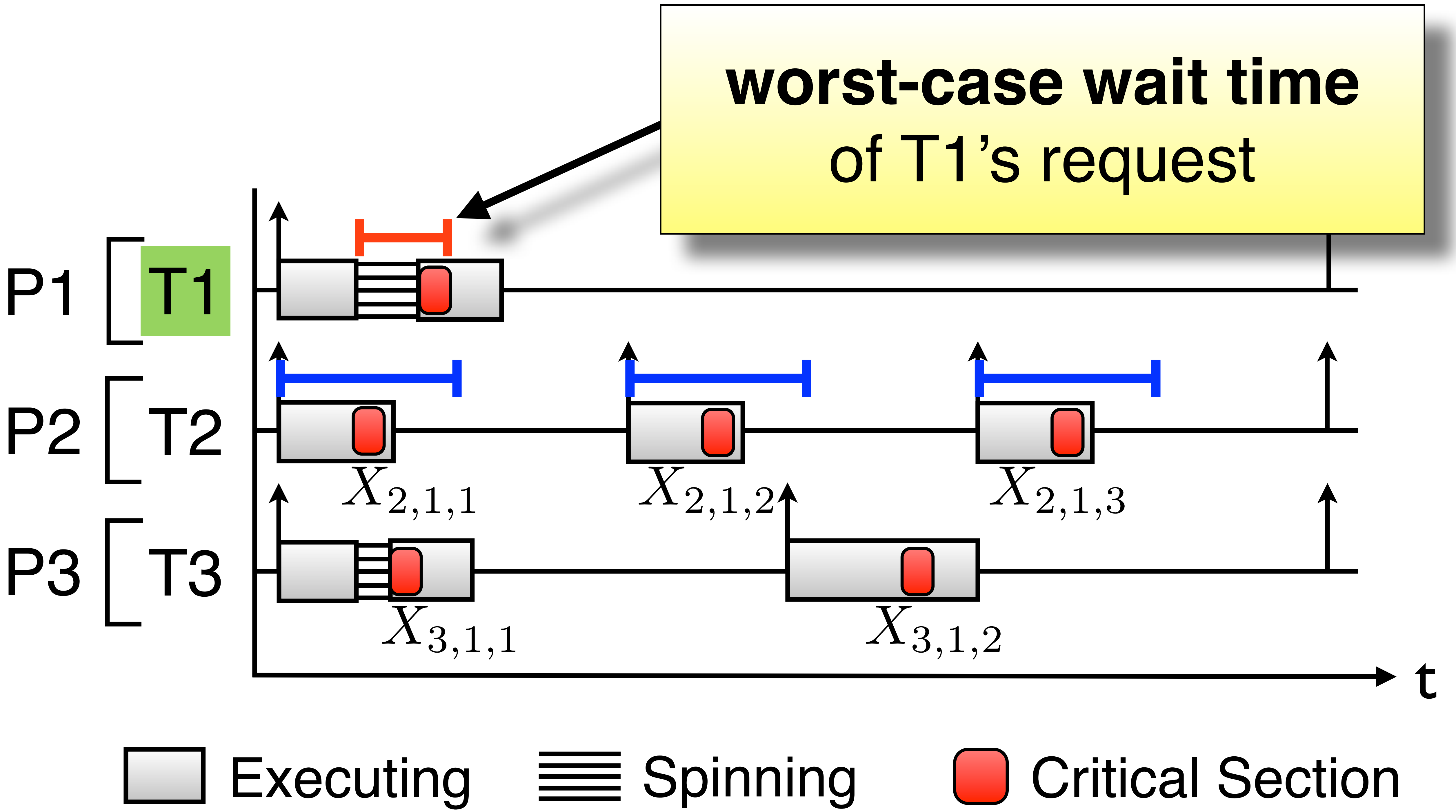
# Bounding the Wait Time of Requests



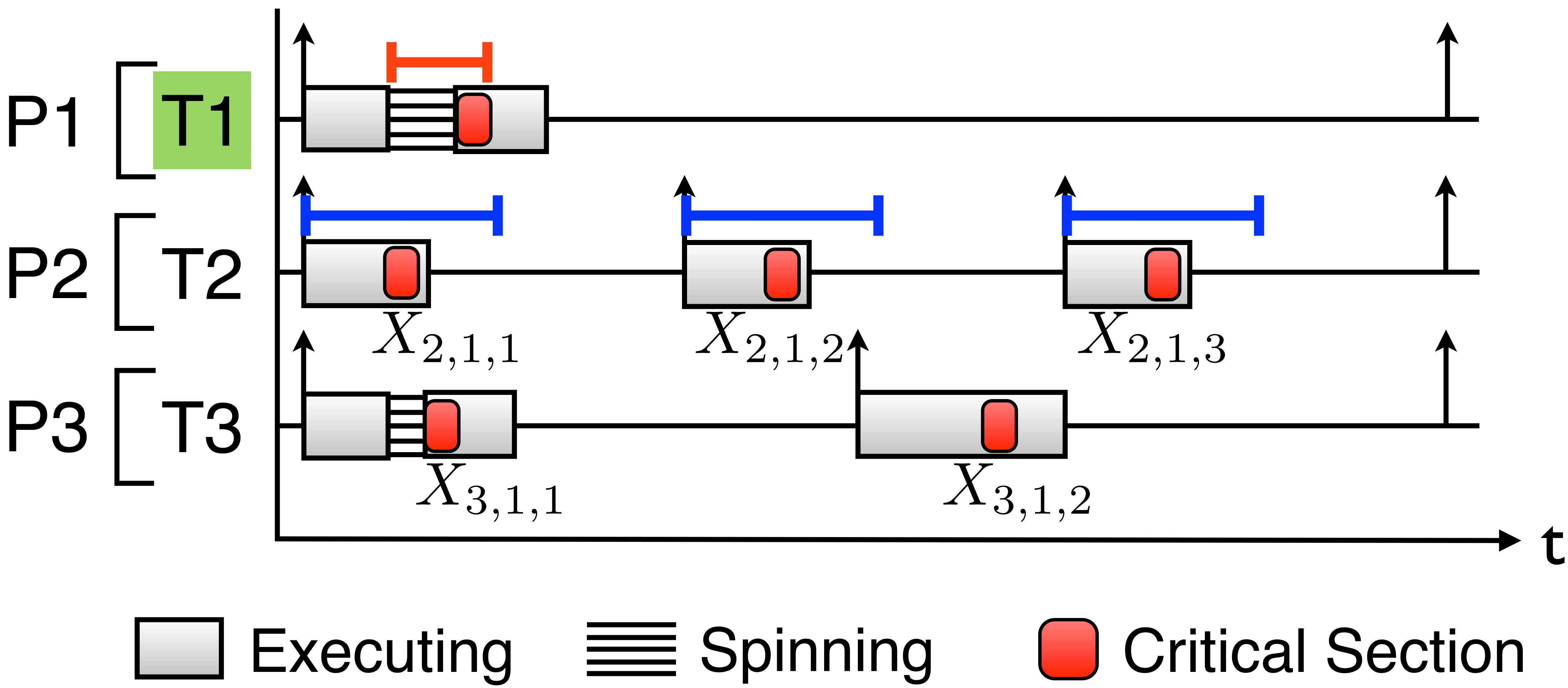
# Bounding the Wait Time of Requests



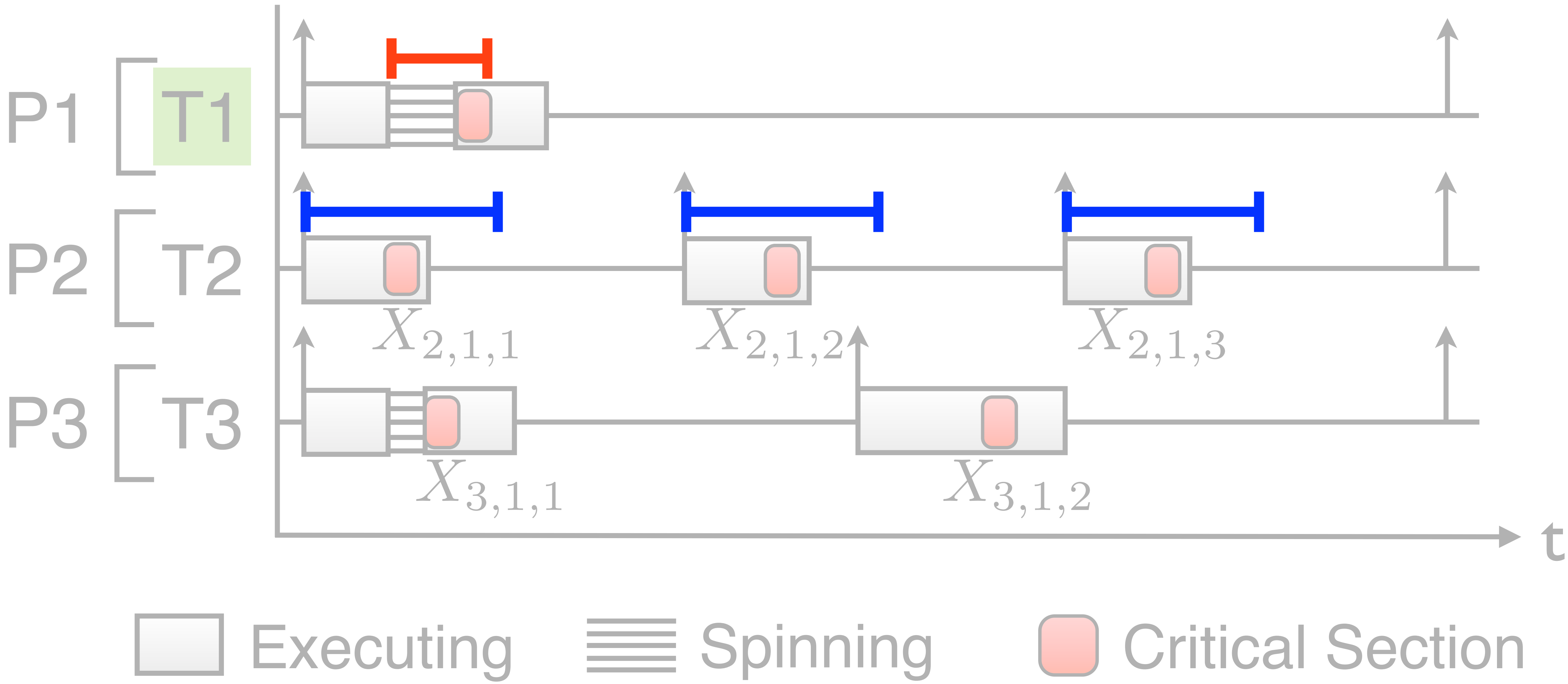
# Bounding the Wait Time of Requests



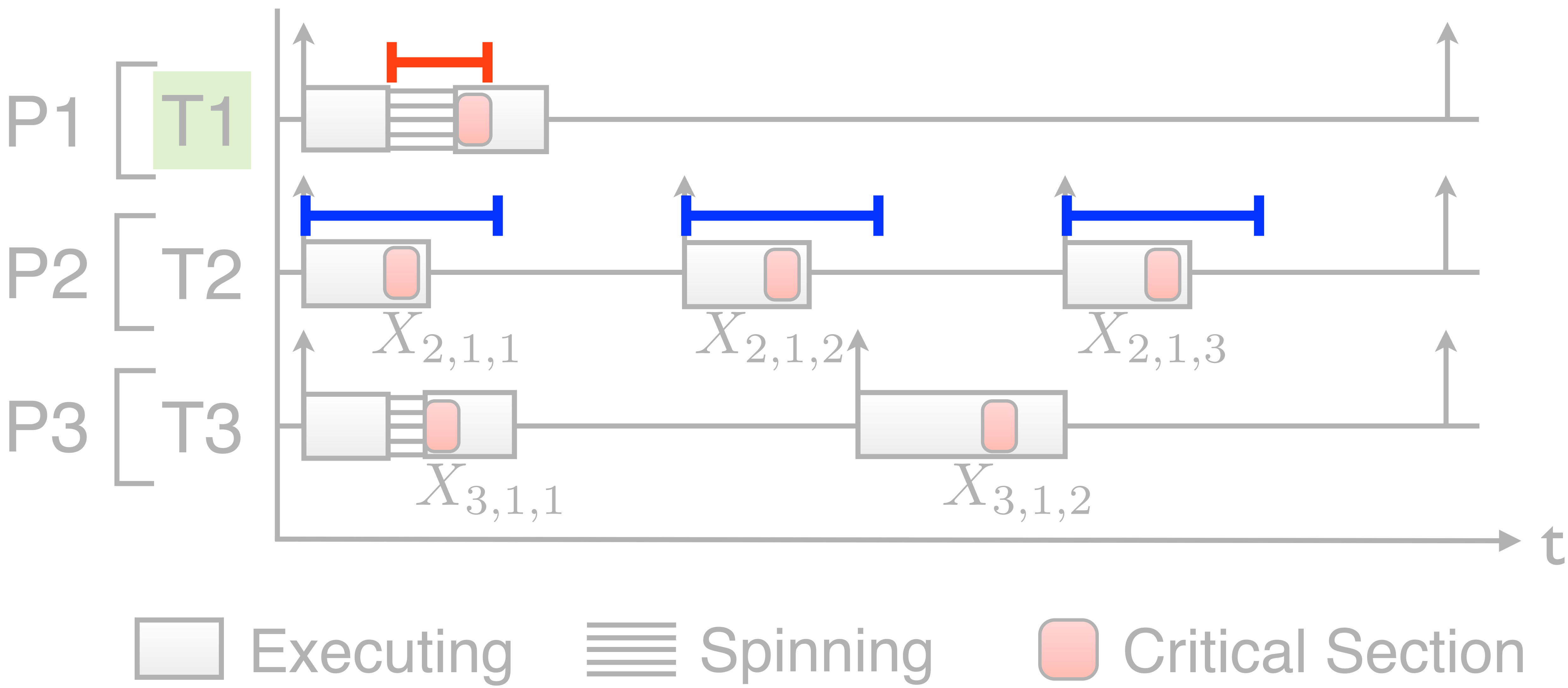
# Bounding the Wait Time of Requests



# Bounding the Wait Time of Requests

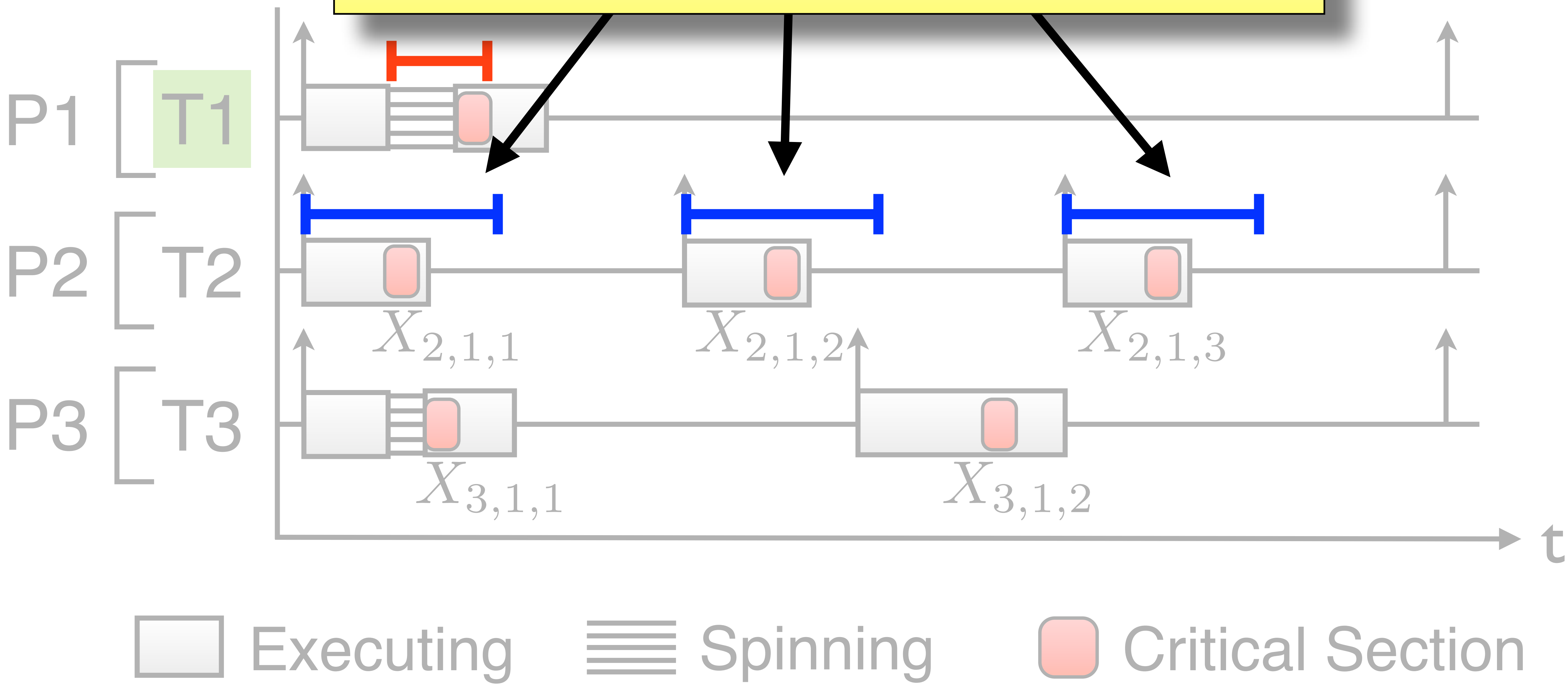


# Bounding the Wait Time of Requests



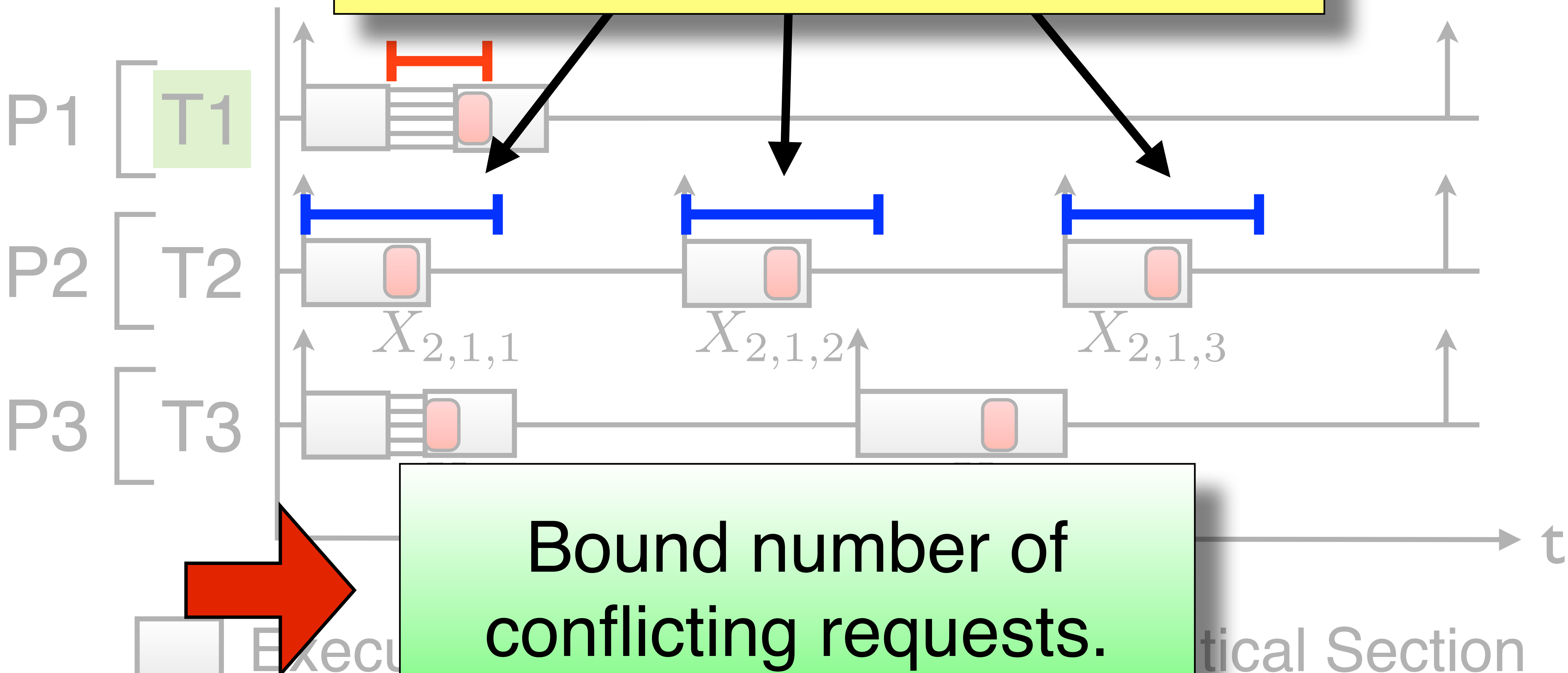
# Bounding the Wait Time of Requests

At most one of T2's jobs can overlap with T1's request.



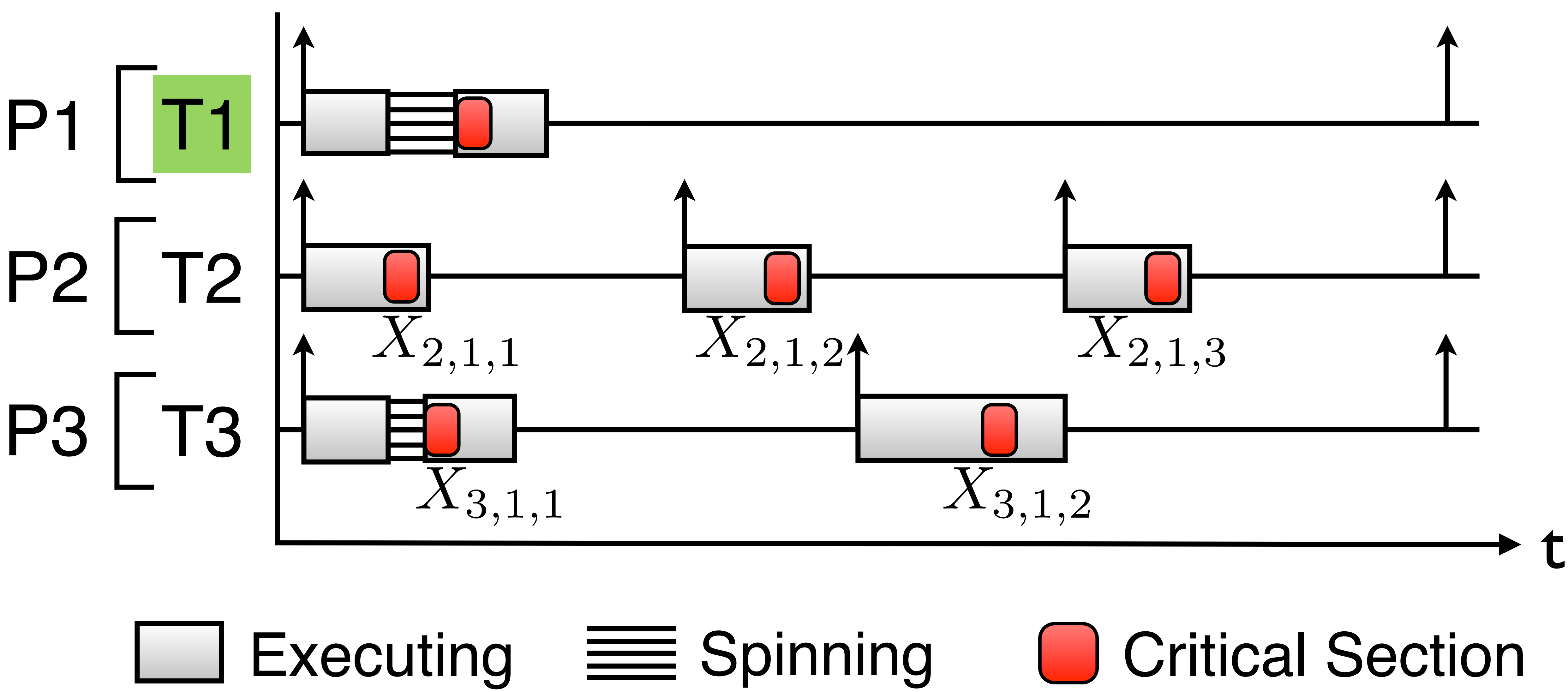
# Bounding the Wait Time of Requests

At most one of T2's jobs can overlap with T1's request.



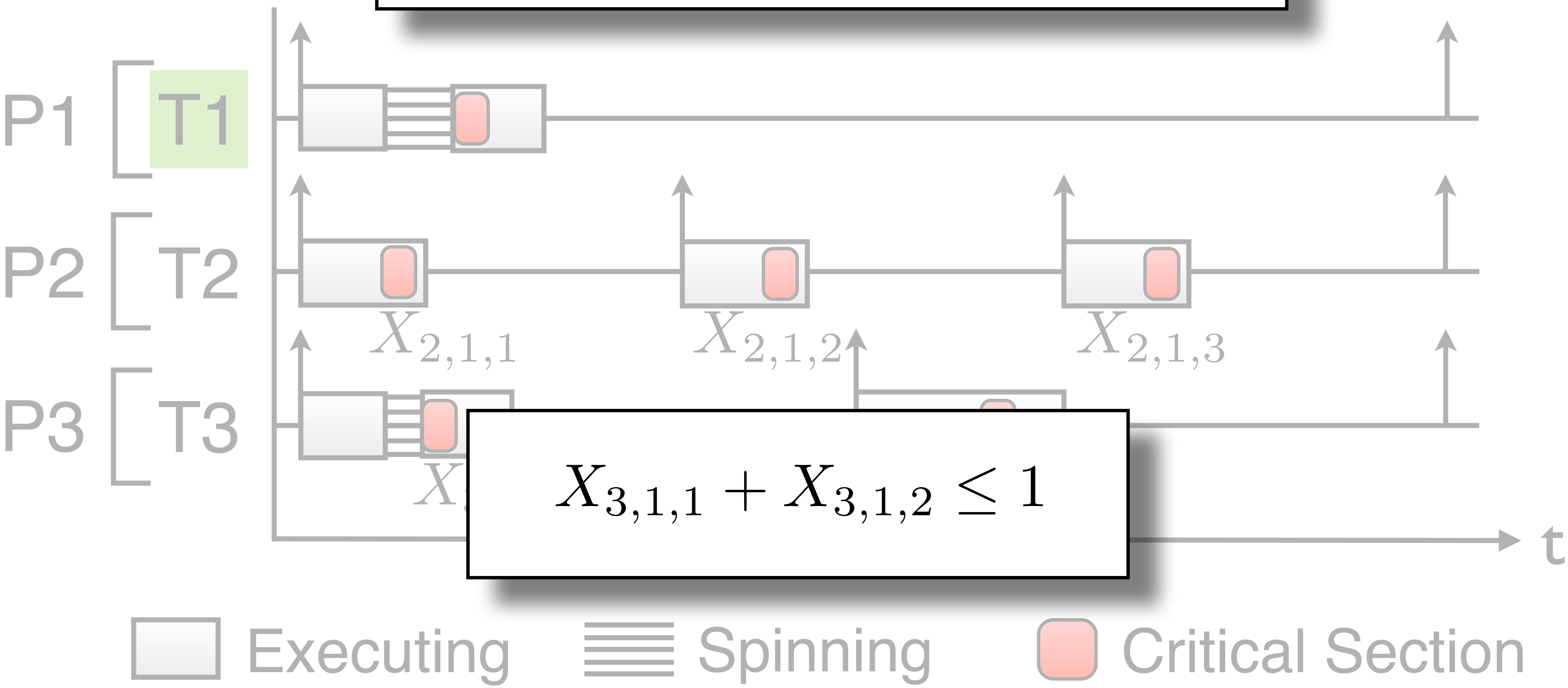

Bound number of conflicting requests.

# ILP Constraints
# Unordered Spin Locks



# ILP Constraints
# Unordered Spin Locks

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$



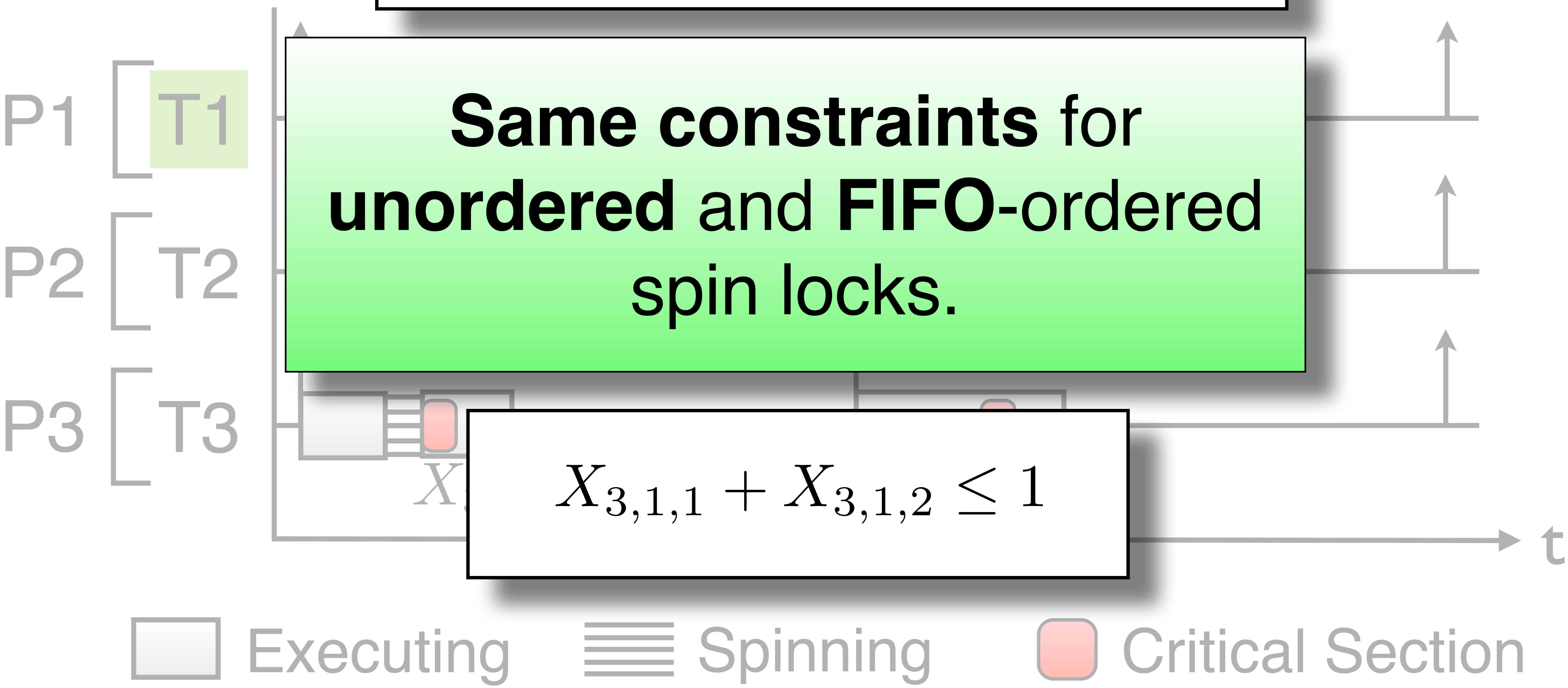

$$X_{3,1,1} + X_{3,1,2} \leq 1$$

# ILP Constraints
# Unordered Spin Locks

$$X_{2,1,1} + X_{2,1,2} + X_{2,1,3} \leq 1$$

**Same constraints** for **unordered** and **FIFO**-ordered spin locks.

$$X_{3,1,1} + X_{3,1,2} \leq 1$$



# Challenges

Prior analysis is **pessimistic due to inflation**.

Explicit blocking terms
ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

Wait-time bounds
Composable constraints

# Challenges

Prior analysis is **pessimistic due to inflation**.

→ Explicit blocking terms
ILP formulation

Prior analysis is **specific to non-preemptable FIFO-ordered** spin locks.

→ Wait-time bounds
Composable constraints

# Evaluation

# Evaluation

Non-Preemptable Spinning

Preemptable Spinning

×

FIFO-ordered (MSRP)

Priority-Ordered

Unordered

Priority-Ordered with FIFO tie-breaking

# Evaluation

Could we **reduce pessimism**?

FIFO-ordered (MSRP)

Non-Preemptable Spinning

×

Priority-Ordered

Unordered

Preemptable Spinning

Priority-Ordered with FIFO tie-breaking

# Evaluation

Could we **reduce pessimism**?

Does the spin lock type matter at all?

FIFO-ordered (MSRP)

Priority-Ordered

Unordered

Priority-Ordered with FIFO tie-breaking

Preemptable Spinning

# Evaluation

Could we **reduce pessimism**?

FIFO-ordered
(MSRP)

Does the spin lock type matter at all?

×

**Which type**
should we use?

Preemptable
Spinning

Unordered

Priority-Ordered
with FIFO tie-breaking

# Evaluation

Could we **reduce pessimism**?

FIFO-ordered (MSRP)

Does the spin lock type matter at all?

**Which type** should we use?

Unordered

**When** can **unordered** locks be used?

Priority-Ordered with FIFO tie-breaking

# Evaluation

Could we **reduce pessimism**?

FIFO-ordered

Does the spin lock type matter at all?

**Which type** should we use?

**When** can **unordered** locks be used?

Should spinning be **preemptable or non-preemptable?**

# Evaluation

**Large-scale schedulability experiments:**

- number of processors $m$: 4, 8, 16
- average task utilization: 0.1, 0.2, 0.3
- critical section lengths: $[1\mu s, 15\mu s]$, $[1\mu s, 100\mu s]$
- number of resources: $m/2,\ m,\ 2m$
- number of requests: 1, 2, 5, 10, 15
- resource sharing factors: 0.1, 0.25, 0.4, 0.75

# Evaluation

**Large-scale schedulability experiments:**

1296 different configurations

- number of processors $m$: 4, 8, 16
- aver
- critical section lengths: $[1\mu s, 15\mu s]$, $[1\mu s, 100\mu s]$
- number of resources: $m/2,\ m,\ 2m$
- number of requests: 1, 2, 5, 10, 15
- resource sharing factors: 0.1, 0.25, 0.4, 0.75

# Evaluation

**Large-scale schedulability experiments:**

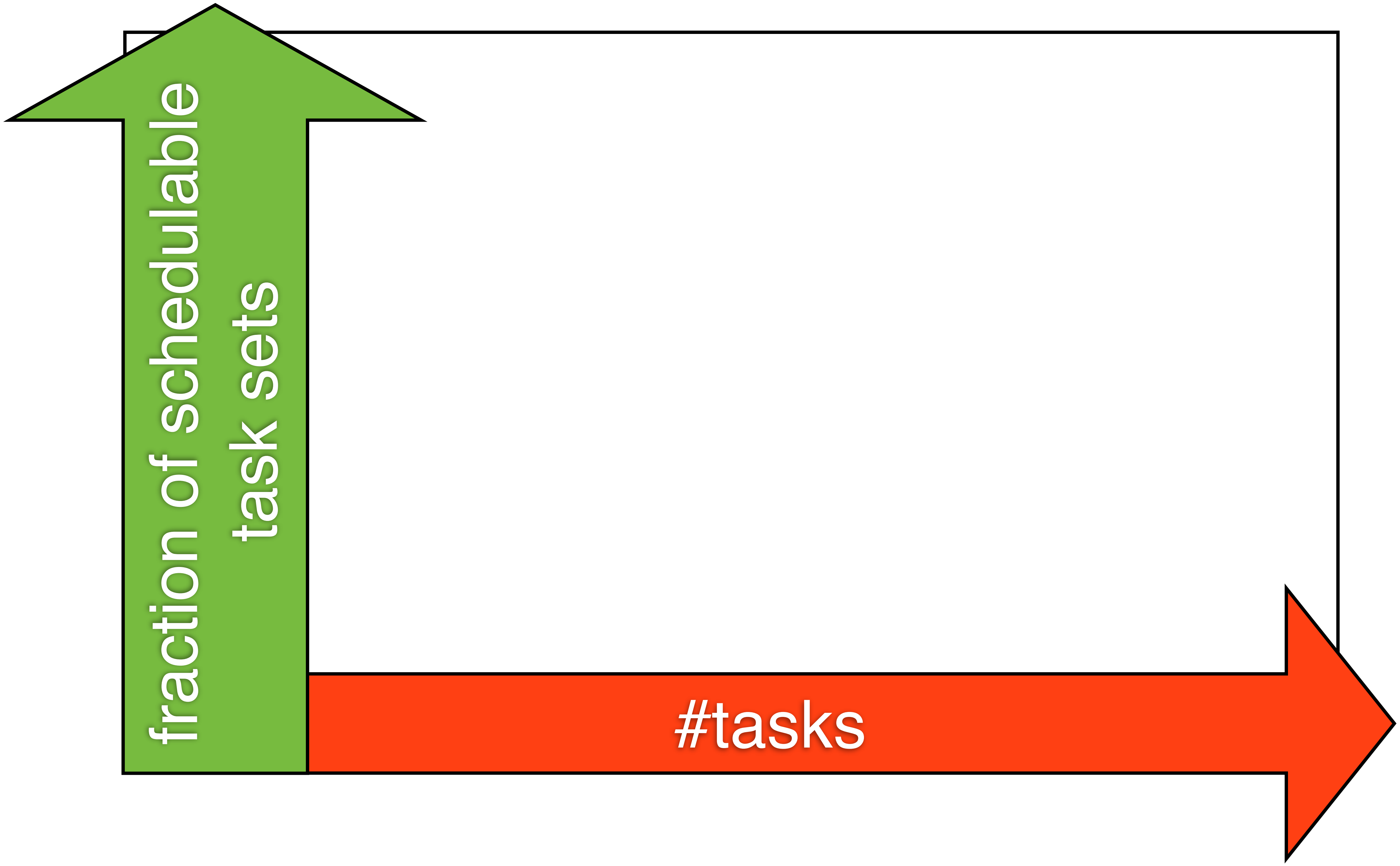
- number of processors $m$: 4, 8, 16

1296 different configurations

- aver
- critical section lengths: $[1\mu s, 15\mu s]$, $[1\mu s, 100\mu s]$
- number
- number
- resource 0.4, 0.75

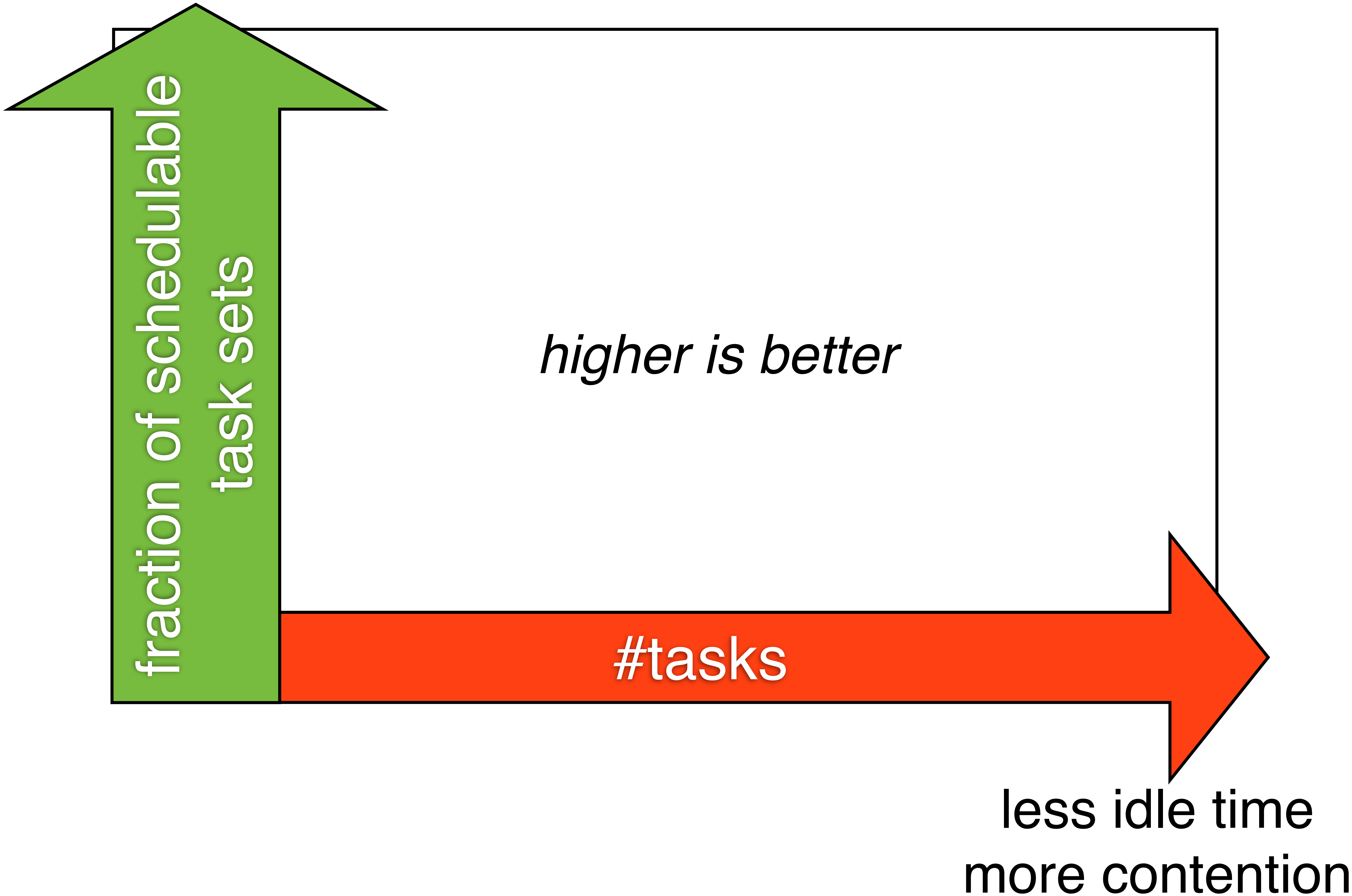
≥1000 samples
per data point
in each configuration

# Schedulability Experiments




less idle time
more contention

# Schedulability Experiments



# Notation

$$F \mid N$$

# Notation

**F**/N

FIFO-ordered

# Notation

F|**N**

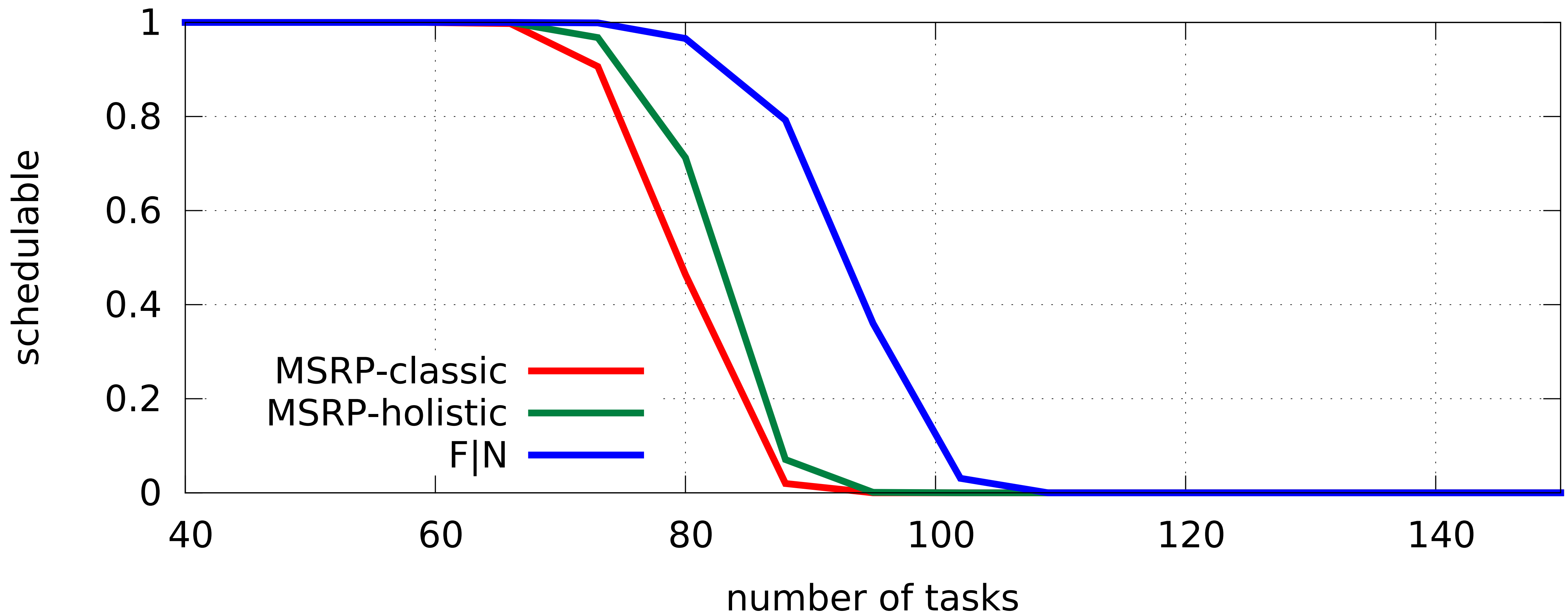
Non-preemptable

# Notation

| **Type** | **Ordering** |
|---|---|
| F\|N | FIFO-ordered non-preemptable |
| P\|N | Priority-ordered non-preemptable |
| U\|N | Unordered non-preemptable |
| PF\|N | Priority-ordered non-preemptable with FIFO tie-breaking |

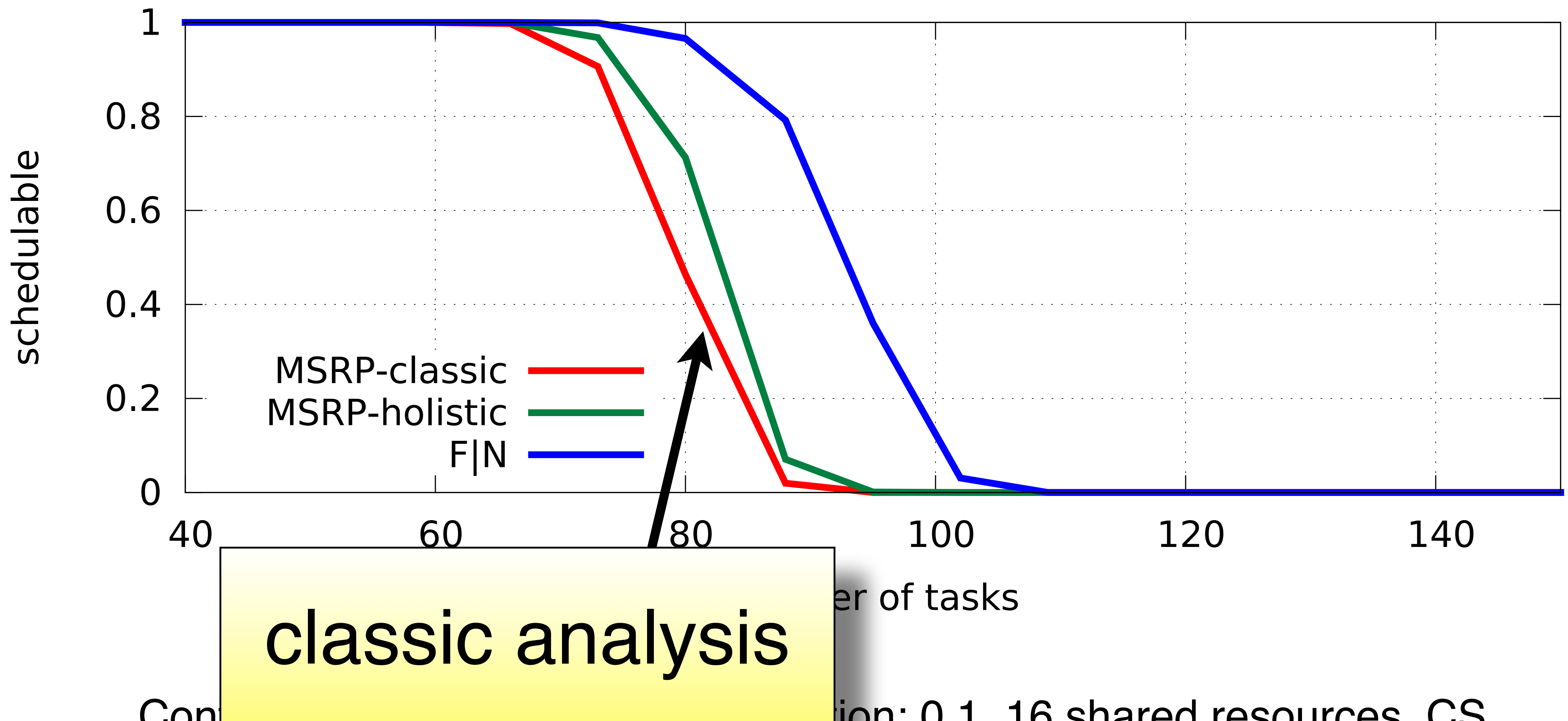
Does the new analysis reduce pessimism?

# Does the new analysis reduce pessimism?



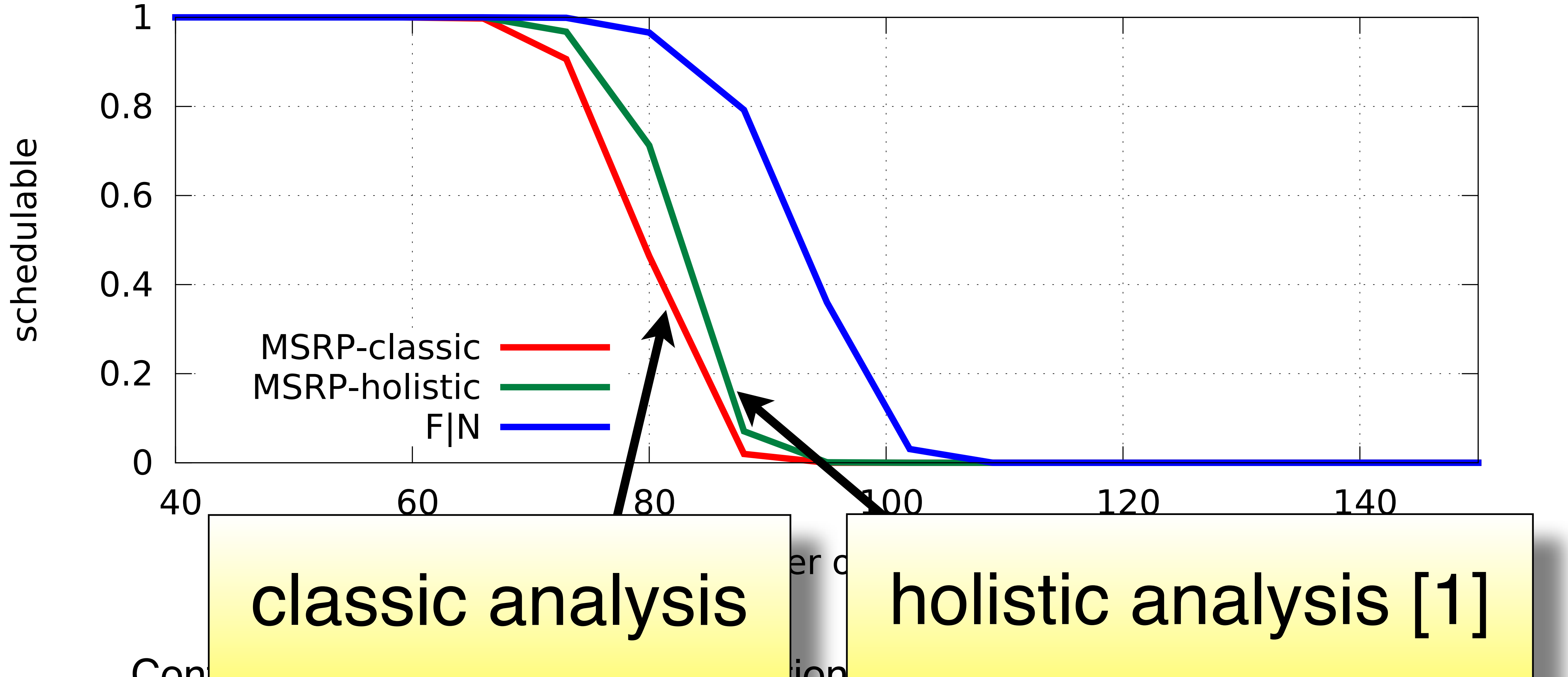

Configuration: 16 CPUs, avg. utilization: 0.1, 16 shared resources, CS lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

# Does the new analysis reduce pessimism?




Con [illegible] ion: 0.1, 16 shared resources, CS lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

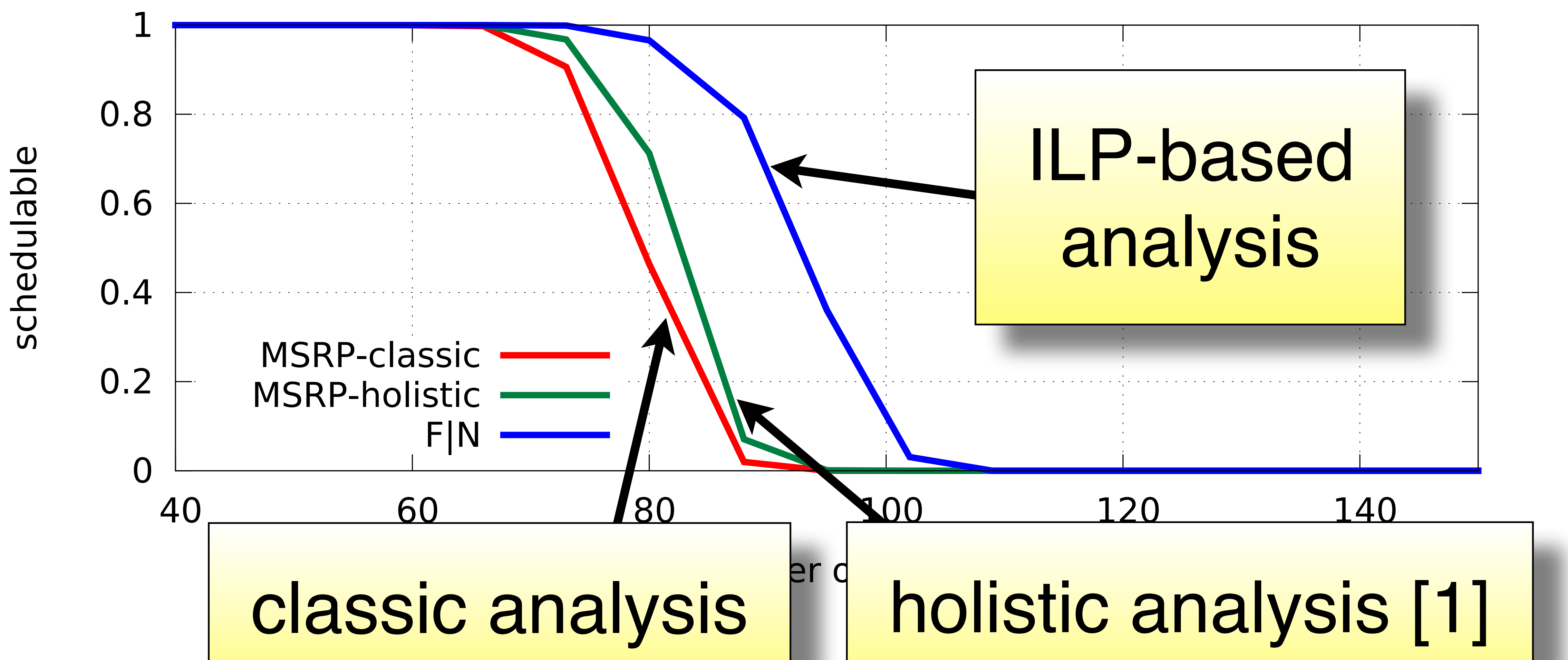
# Does the new analysis reduce pessimism?




lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

[1] B. Brandenburg, “Scheduling and locking in multiprocessor real-time operating systems,” Ph.D. dissertation, The University of North Carolina at Chapel Hill, 2011.

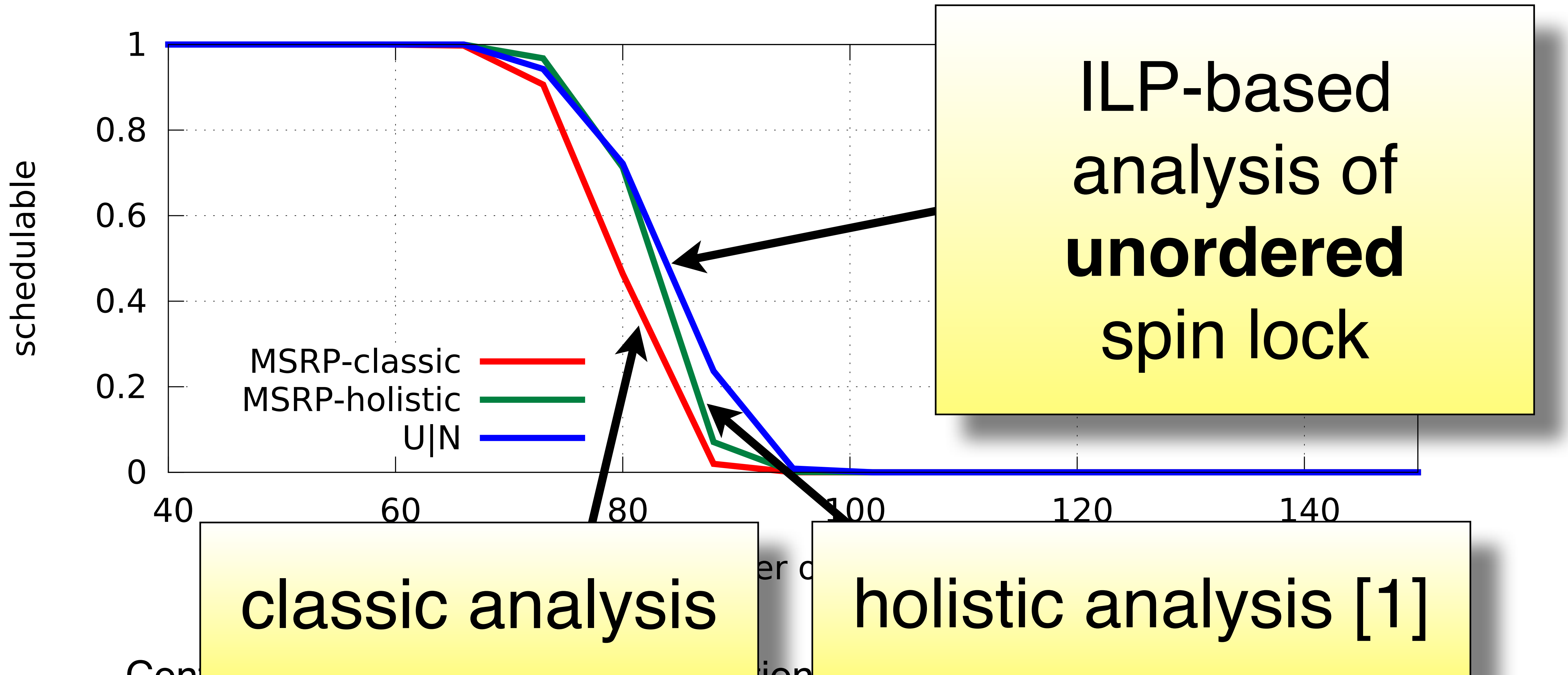
# Does the new analysis reduce pessimism?




Con[illegible] lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

[1] B. Brandenburg, “Scheduling and locking in multiprocessor real-time operating systems,” Ph.D. dissertation, The University of North Carolina at Chapel Hill, 2011.

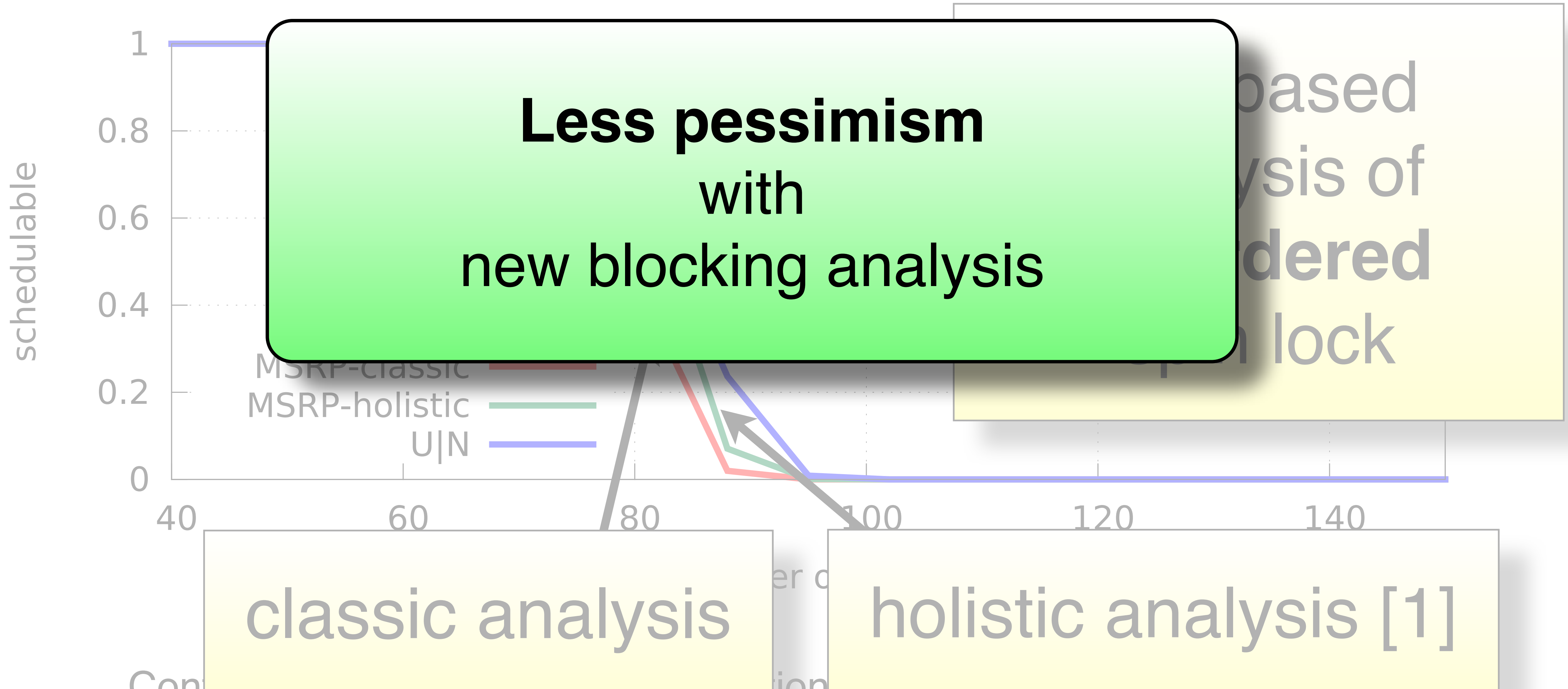
# Does the new analysis reduce pessimism?




Con[...]ion[...]
lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

[1] B. Brandenburg, "Scheduling and locking in multiprocessor real-time operating systems," Ph.D. dissertation, The University of North Carolina at Chapel Hill, 2011.

# Does the new analysis reduce pessimism?

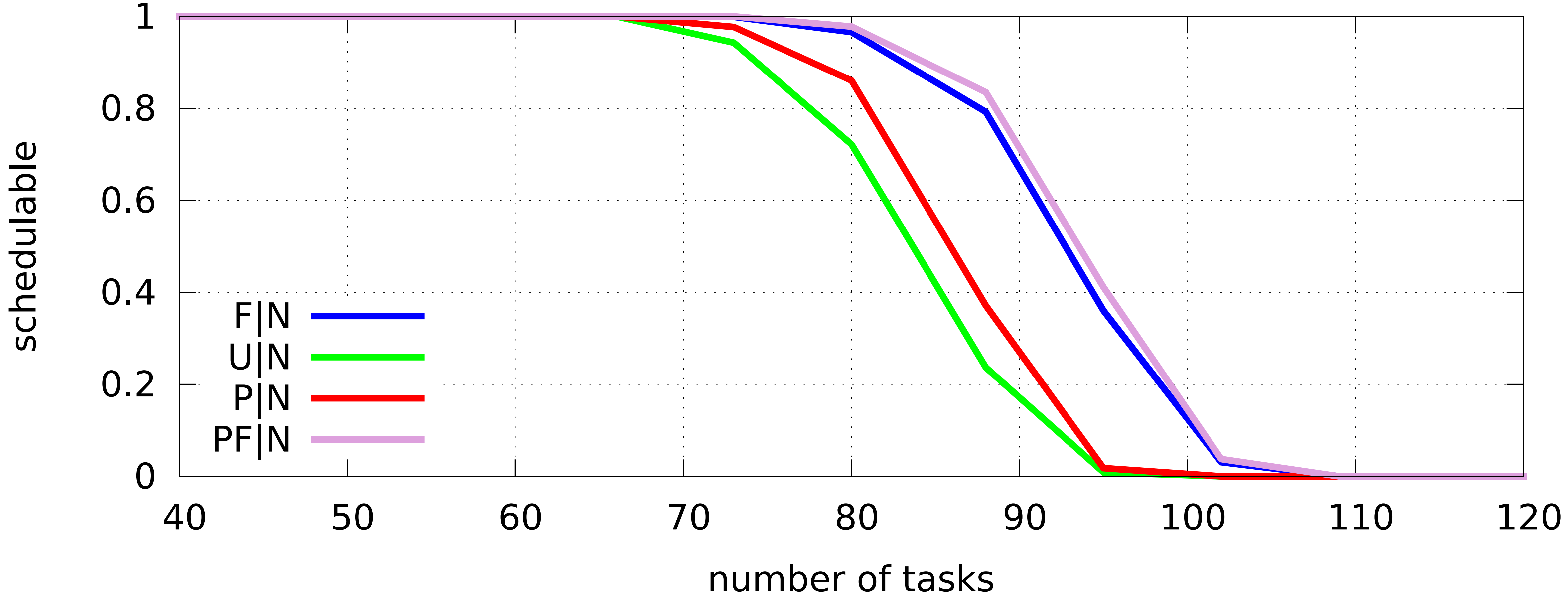



lengths $[1\mu s, 15\mu s]$, at most 2 requests per resource, contention = 0.4

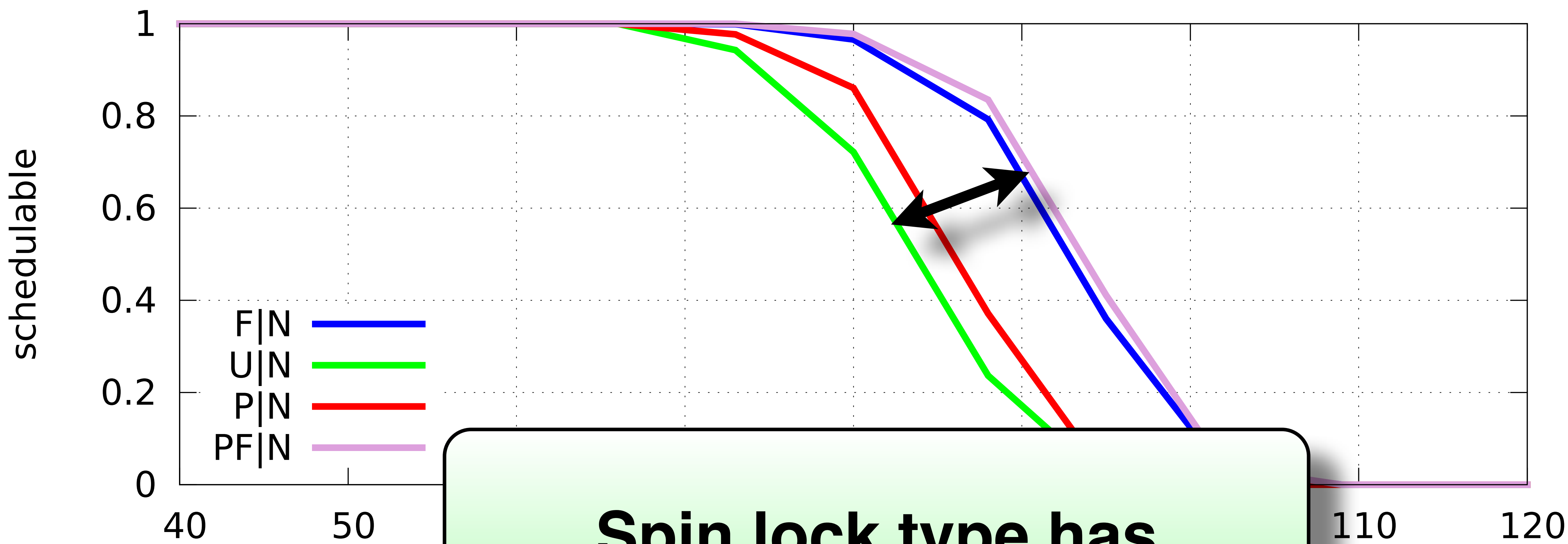
[1] B. Brandenburg, “Scheduling and locking in multiprocessor real-time operating systems,” Ph.D. dissertation, The University of North Carolina at Chapel Hill, 2011.

# Does the
# Spin Lock Type matter?

# Does the Spin Lock Type matter?



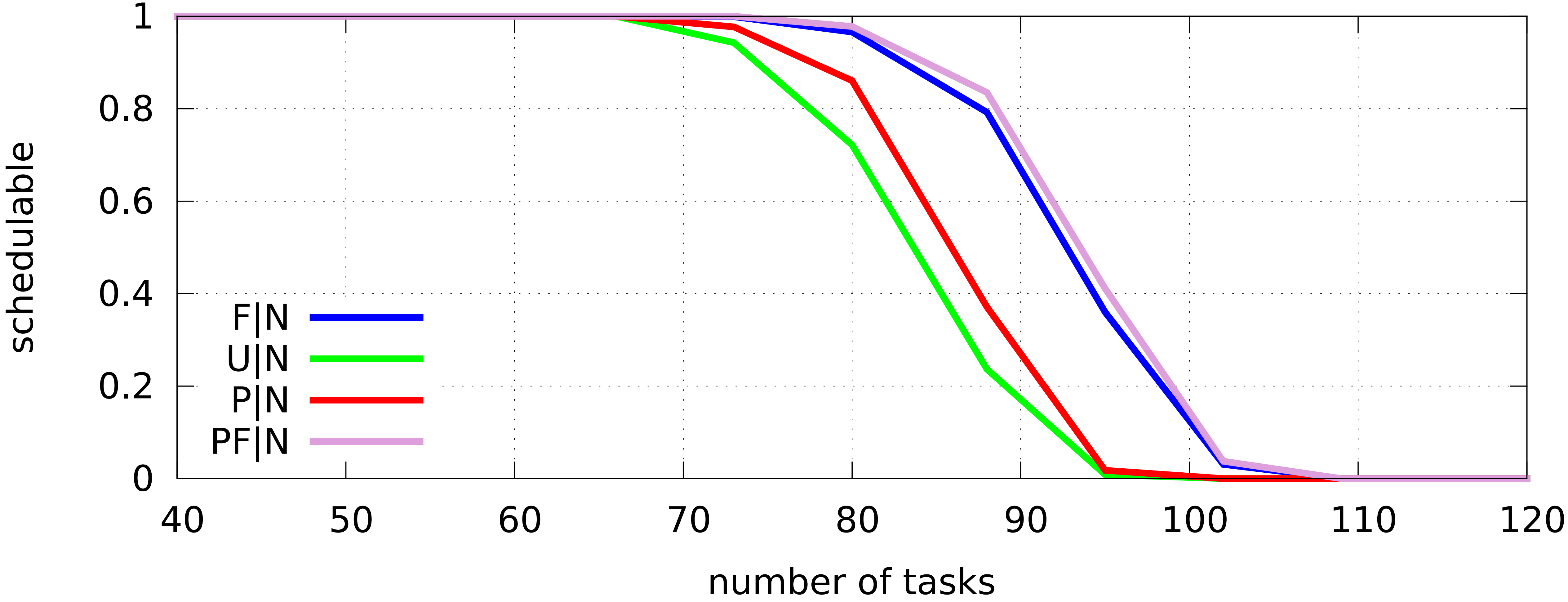
# Does the
# Spin Lock Type matter?



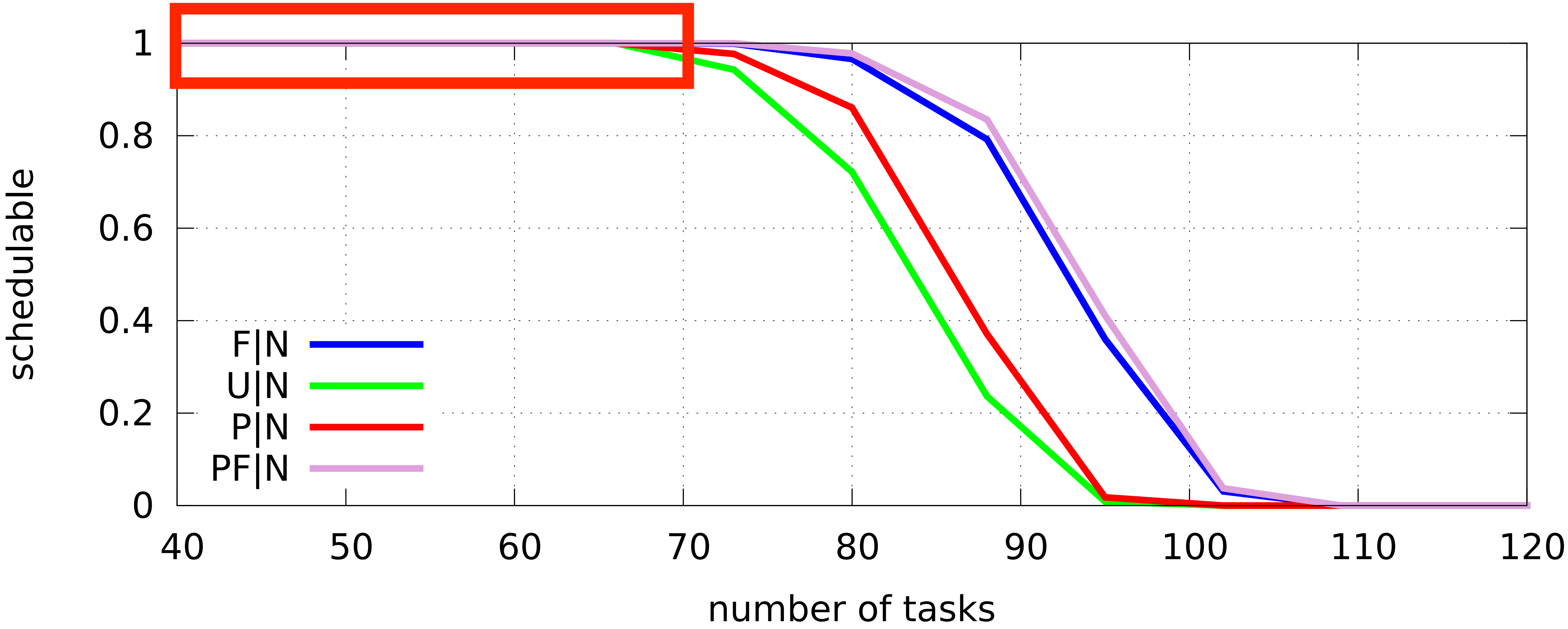

**Spin lock type has significant impact** on schedulability.

# When can unordered locks be used?

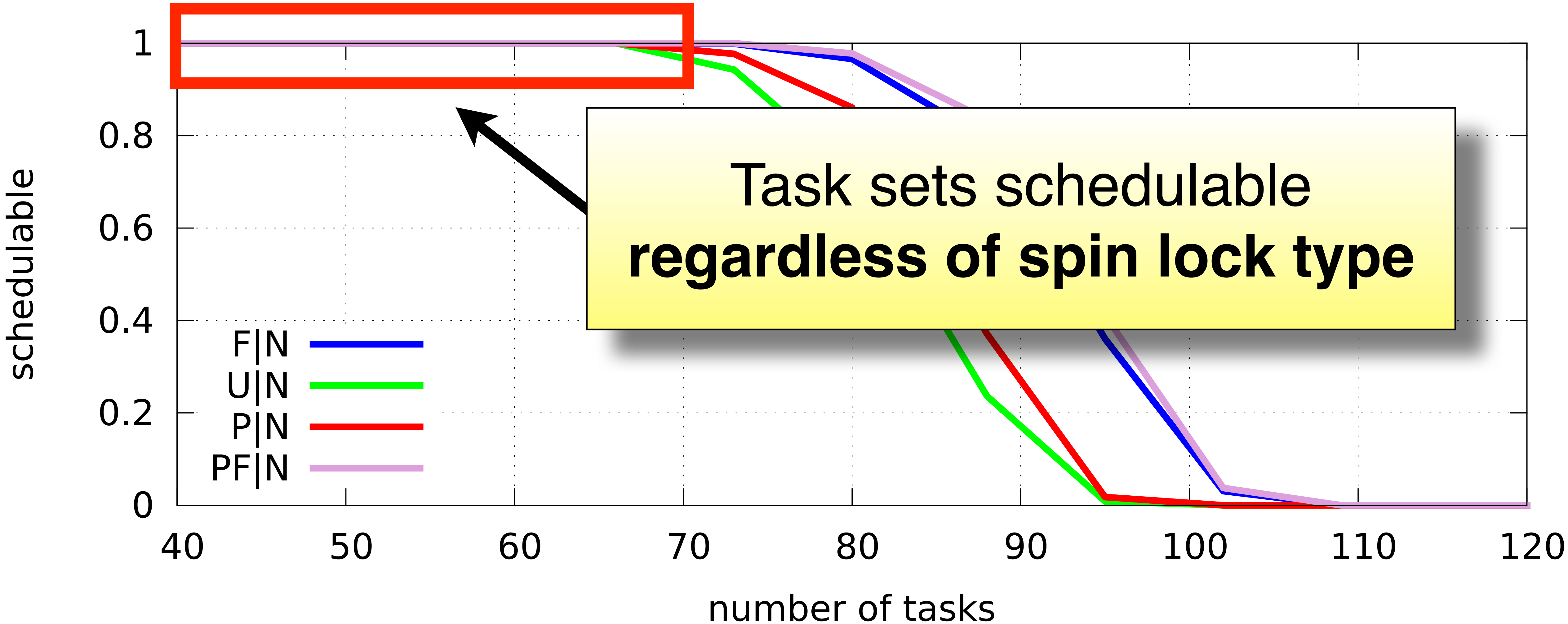
# When can unordered locks be used?



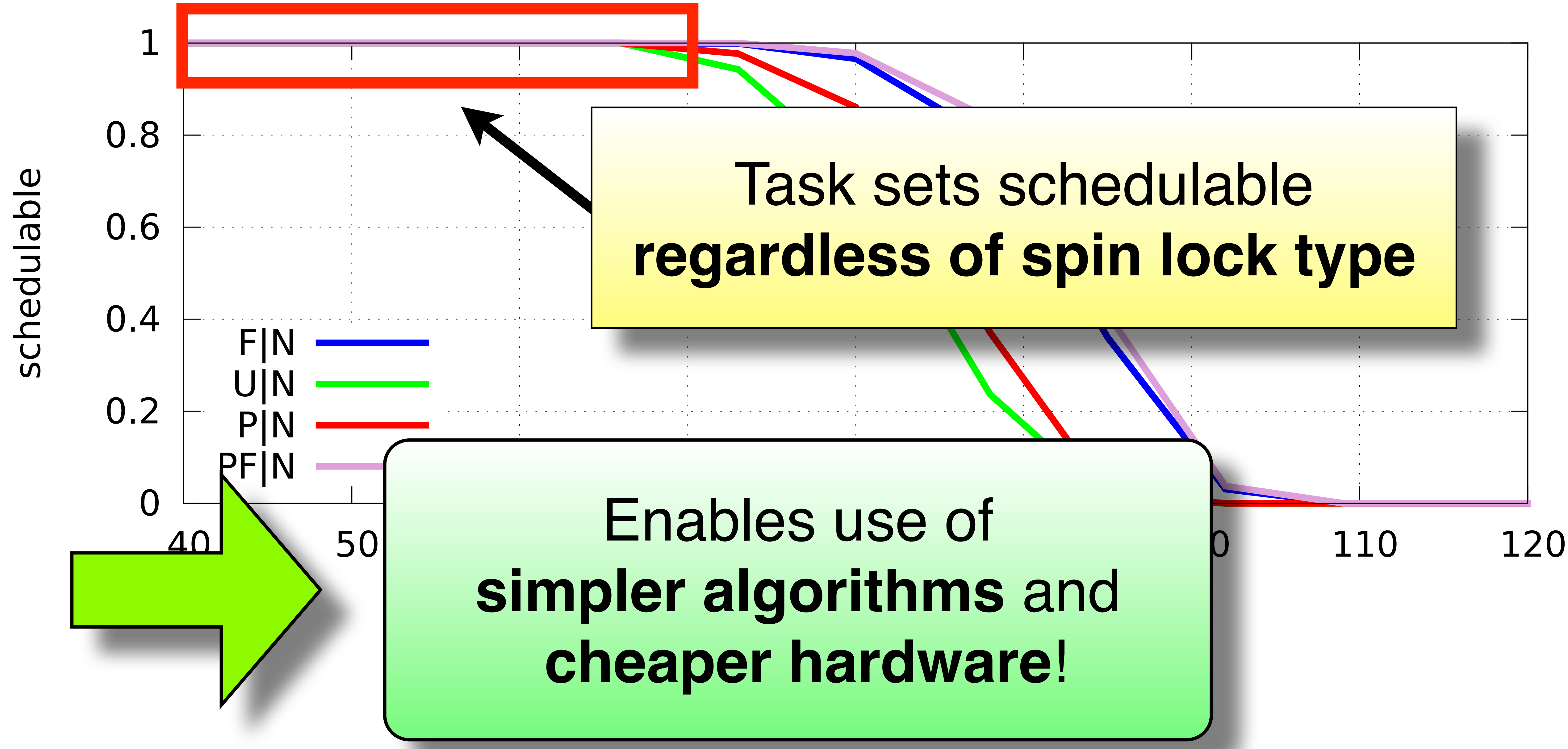
# When can unordered locks be used?



# When can unordered locks be used?



# When can unordered locks be used?




Task sets schedulable **regardless of spin lock type**

Enables use of **simpler algorithms** and **cheaper hardware**!

# Handling
# Unknown Spin Lock Types

Analysis for unordered spin locks
makes
**no ordering assumptions**

# Handling Unknown Spin Lock Types

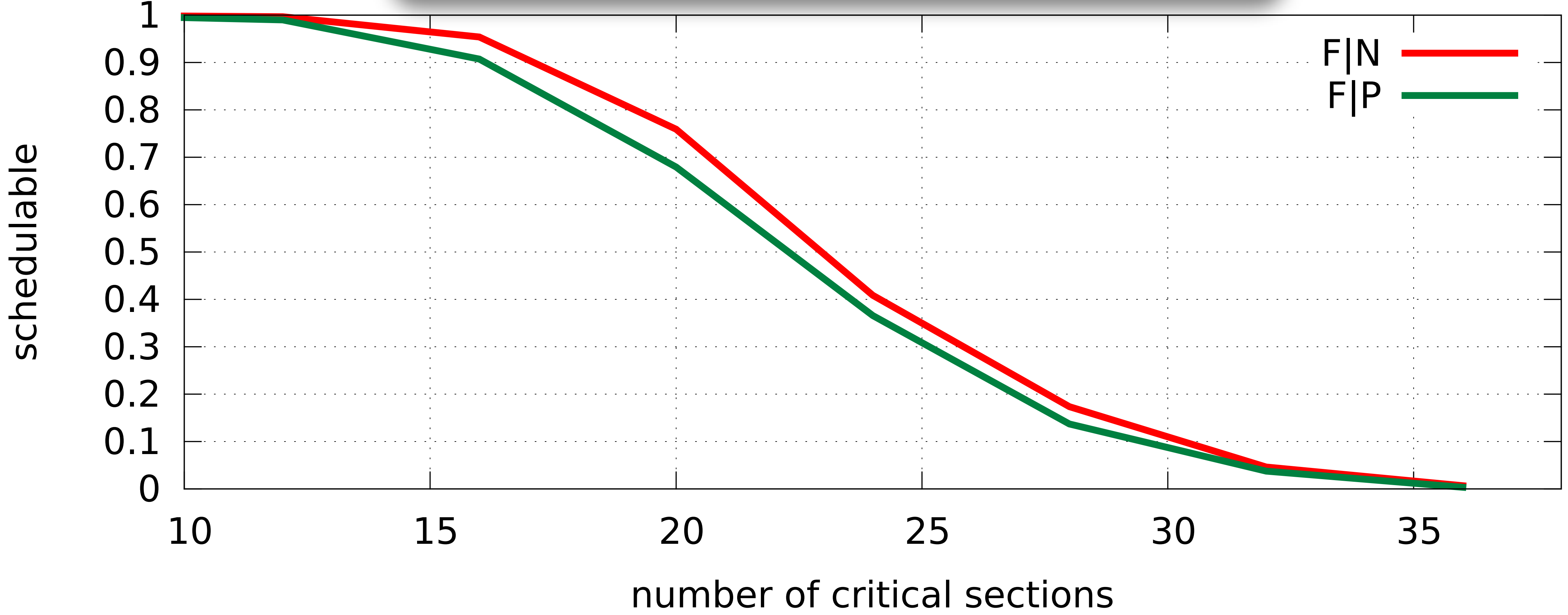
Analysis for unordered spin locks makes **no ordering assumptions**

Analysis for unordered spin locks **applicable for unknown types!**

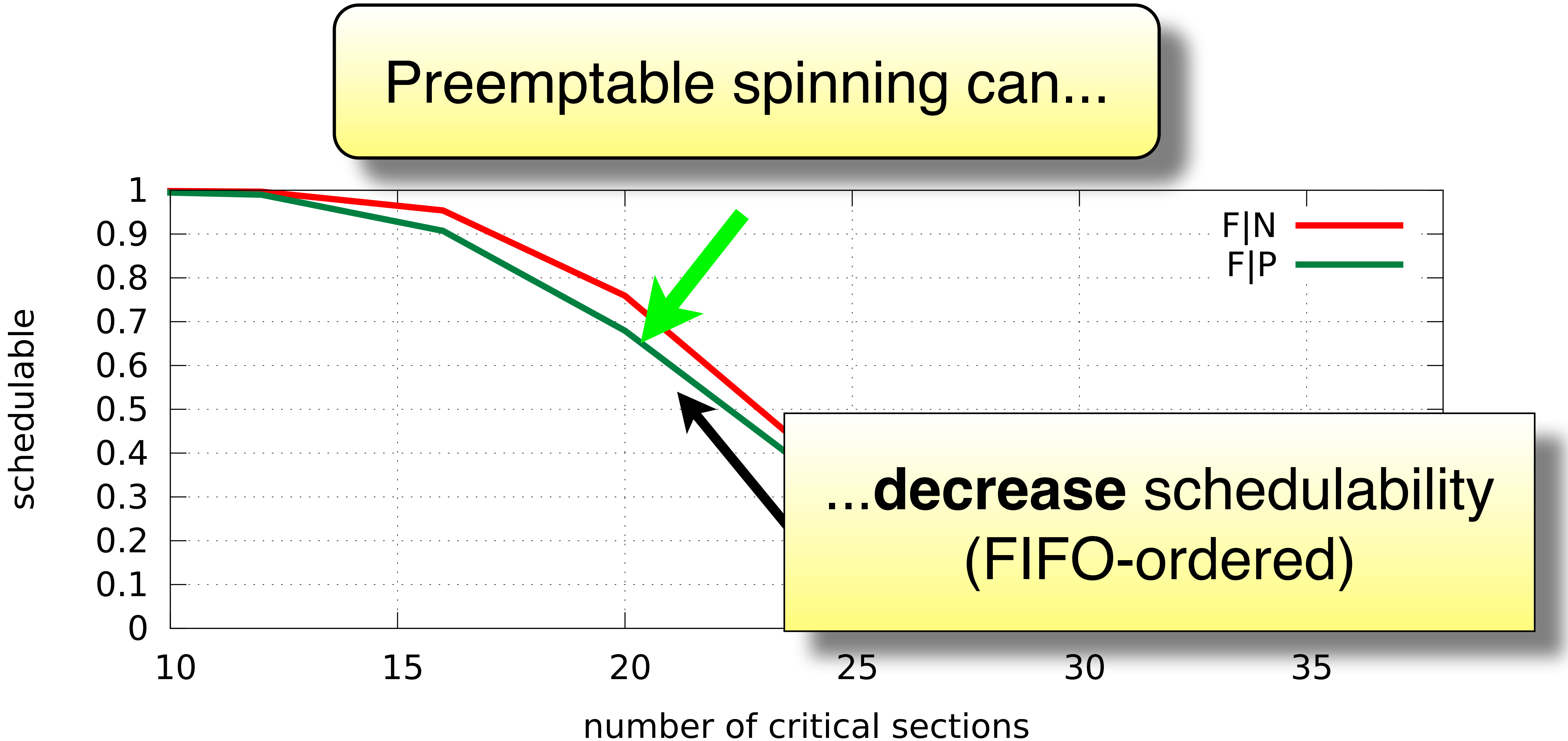
What is the impact of allowing preemptable spinning?

# What is the impact of allowing preemptable spinning?

Preemptable spinning can...

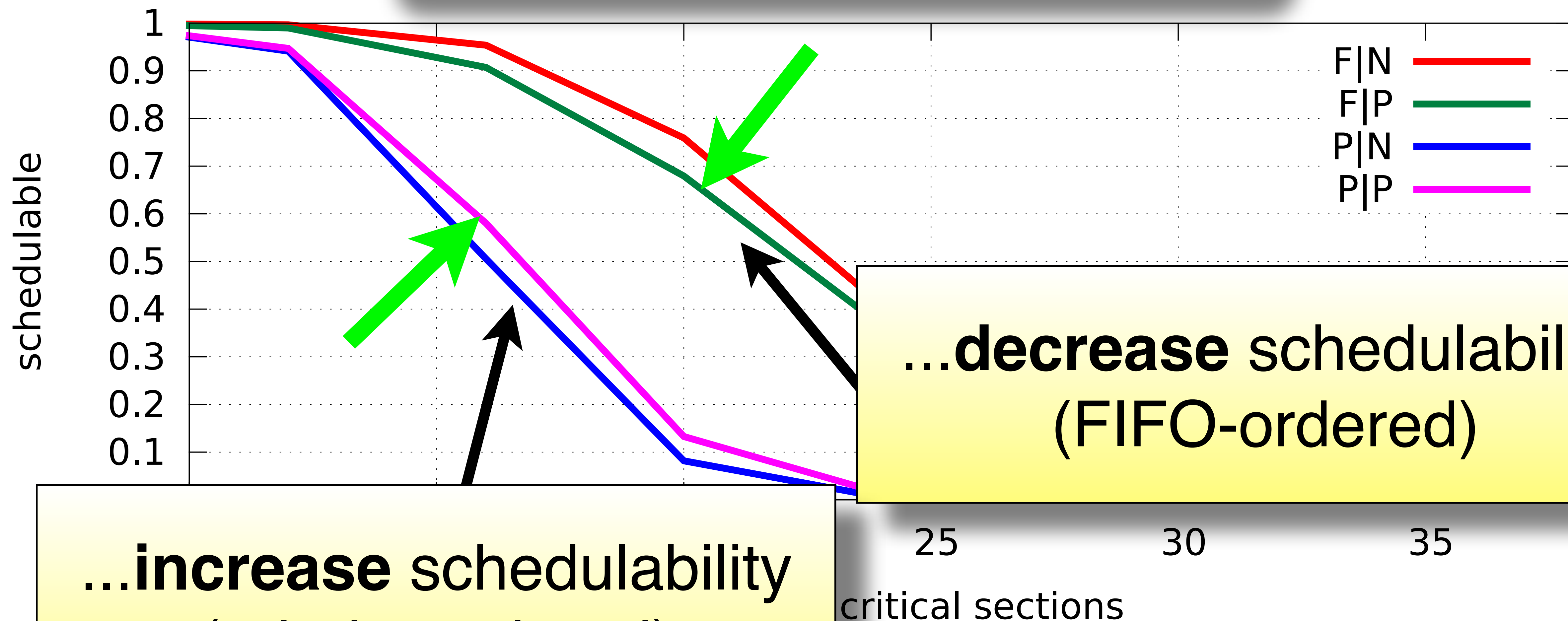


# What is the impact of allowing preemptable spinning?



# What is the impact of allowing preemptable spinning?

Preemptable spinning can...




...**decrease** schedulability (FIFO-ordered)

...**increase** schedulability (priority-ordered)

# Summary and Conclusions

# Suggested AUTOSAR API changes

Spin lock type has significant impact on schedulability.

→

**Specify spin lock type**

# Suggested AUTOSAR API changes

Spin lock type has significant impact on schedulability.

**Specify spin lock type**

FIFO- and priority ordering required to support many workloads.

**Support FIFO and priority ordering** in AUTOSAR

# Suggested AUTOSAR API changes

Spin lock type has significant impact on schedulability.

**Specify spin lock type**

FIFO- and priority ordering required to support many workloads.

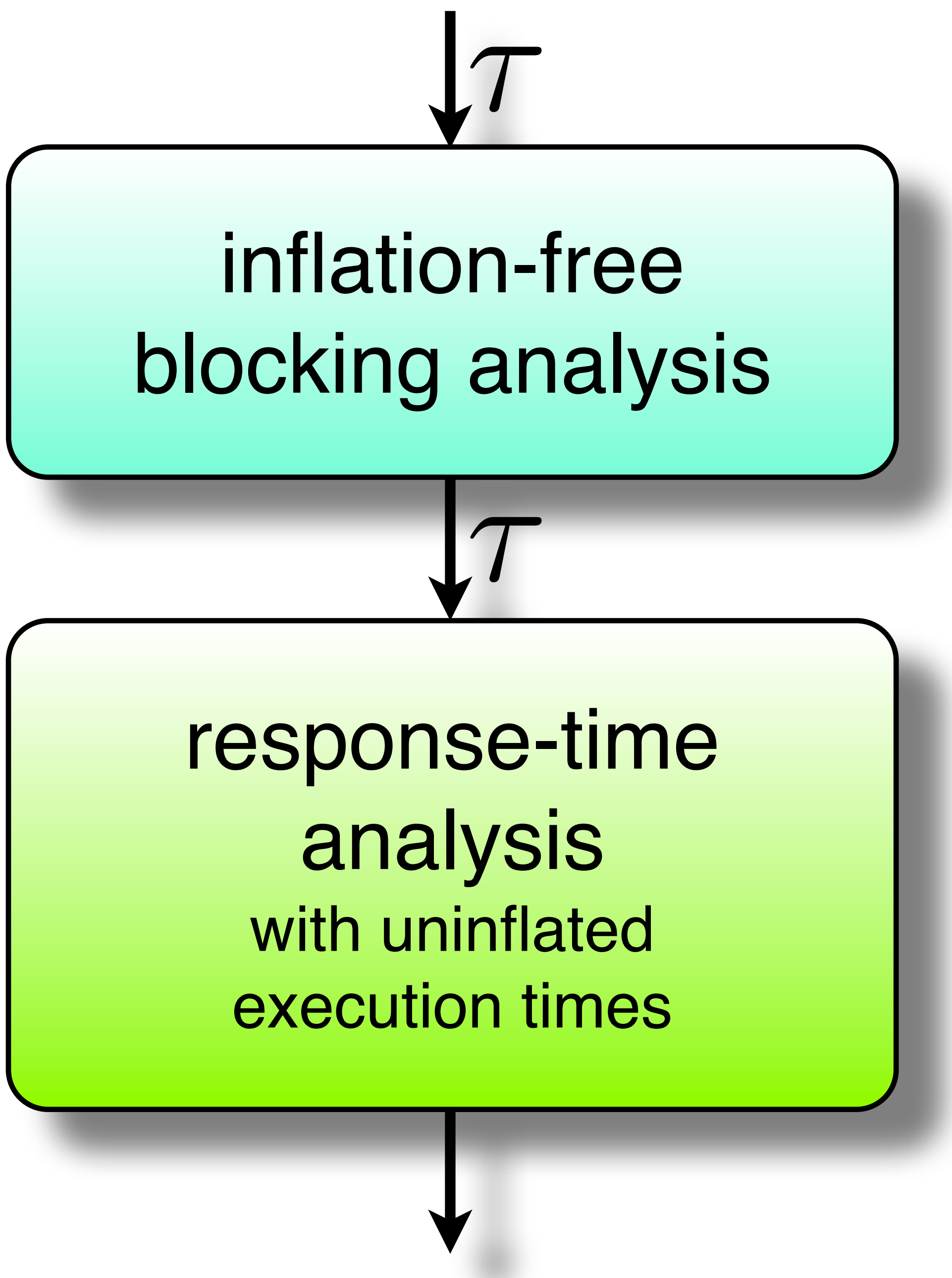
**Support FIFO and priority ordering** in AUTOSAR

Preemptable spinning can improve schedulabiliy.

Support **preemptable spinning with ordering guarantees**

# Summary

Novel blocking analysis
for spin locks:



# Summary

Novel blocking analysis
for spin locks:

$\downarrow \tau$

inflation-free
blocking analysis

$\downarrow \tau$

response-time
analysis
with uninflated
execution times

support for
**8 spin lock types**

# Summary

Novel blocking analysis for spin locks:

$\downarrow \tau$

inflation-free blocking analysis

$\downarrow \tau$

response-time analysis
with uninflated execution times

$\downarrow$

support for **8 spin lock types**

...including **unordered locks**: safe for unknown types

# Summary

Novel blocking analysis
for spin locks:

$\downarrow \tau$

inflation-free
blocking analysis

$\downarrow \tau$

response-time
analysis
with uninflated
execution times

$\downarrow$

support for
**8 spin lock types**

...including
**unordered locks**:
safe for unknown types

asymptotically
**less pessimistic**

# Summary

Novel blocking analysis for spin locks:

$\downarrow \tau$

inflation-free blocking analysis

$\downarrow \tau$

response-time analysis
with uninflated execution times

$\downarrow$

support for **8 spin lock types**

...including **unordered locks**: safe for unknown types

asymptotically **less pessimistic**

suggestions for AUTOSAR

# Future Work

Current analysis assumes non-nested critical sections.

# Future Work

Current analysis assumes non-nested critical sections.

Nested critical sections:
**work in progress**

*Fin*