Non-Preemptive Scheduling with History-Dependent Execution Time

Björn Andersson, Sagar Chaki, Dionisio de Niz
Software Engineering Institute

Brian Dougherty, Jules White
Virginia Tech

Russell Kegley
Lockheed Martin Aeronautics

Example of Task Set in the Old Model

n=3

T_1=50, D_1=10 T_2=150, D_2=15 T_3=500, D_3=500

C_1=4 C_2=7 C_3=5

Example of Task Set in the New Model

Example of Task Set in the New Model

n=3

Example of Task Set in the New Model

n=3

T_1=50, D_1=10 T_2=150, D_2=15 T_3=500, D_3=500

Example of Task Set in the New Model

n=3

T_1=50, D_1=10 T_2=150, D_2=15 T_3=500, D_3=500

Assume non-preemptive scheduling

Example of Task Set in the New Model

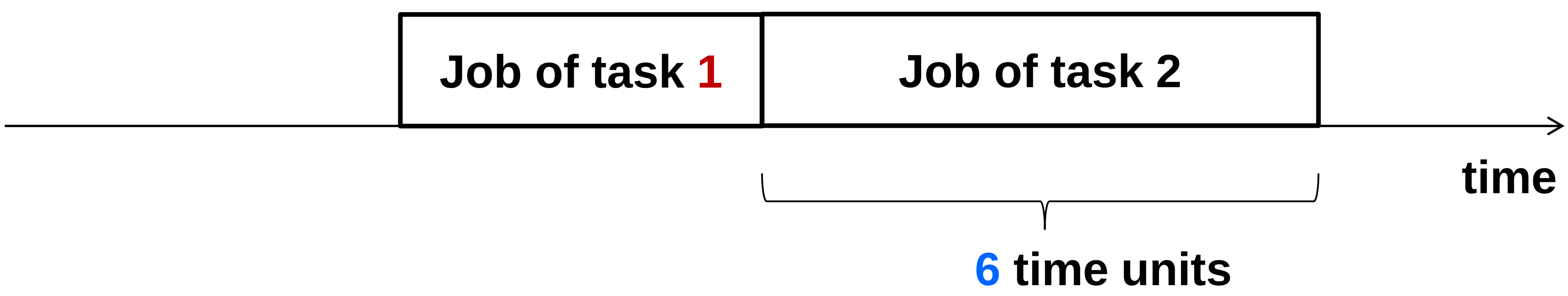
n=3

$T_1=50, D_1=10$

$T_2=150, D_2=15$

$T_3=500, D_3=500$

$ubc_2^2=6$, $lhubc_2^2=1$, $ihubc_2^{2,1}=1$



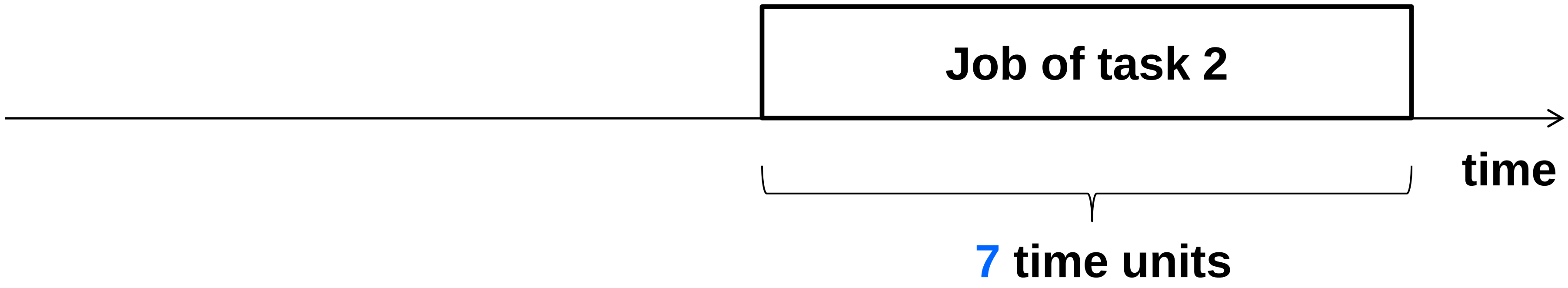
Execution time of a job when a pre-specified history is matched at run-time

Example of Task Set in the New Model

n=3

T_1=50, D_1=10 T_2=150, D_2=15 T_3=500, D_3=500

ubc_2^1=7, $lhubc_2^1$=0



Execution time of a job when the zero-length pre-specified history is matched at run-time

Example of Task Set in the New Model

n=3

T_1=50, D_1=10

T_2=150, D_2=15

T_3=500, D_3=500

$nhubc_2$=2

ubc_2^1=7, $lhubc_2^1$=0

ubc_2^2=6, $lhubc_2^2$=1, $ihubc_2^{2,1}$=1

Pre-specified histories of task τ_2 and their associated upper bounds on execution times

Example of Task Set in the New Model

$n=3$

$T_1=50, D_1=10$

$nhubc_1=2$

$ubc_1^1=4, lhubc_1^1=0$

$ubc_1^2=3, lhubc_1^2=1, ihubc_1^{2,1}=2$

$T_2=150, D_2=15$

$nhubc_2=2$

$ubc_2^1=7, lhubc_2^1=0$

$ubc_2^2=6, lhubc_2^2=1, ihubc_2^{2,1}=1$

$T_3=500, D_3=500$

$nhubc_3=1$

$ubc_3^1=5, lhubc_3^1=0$

Upper bounds on execution times of a job as a function of jobs that executed before it

Example of Task Set in the New Model

n=3

T_1=50, D_1=10
$nhlbc_1$=1
lbc_1^1=0, $lhlbc_1^1$=0

T_2=150, D_2=15
$nhlbc_2$=1
lbc_2^1=0, $lhlbc_2^1$=0

T_3=500, D_3=500
$nhlbc_3$=1
lbc_3^1=0, $lhlbc_3^1$=0

Lower bound on execution times of a job

Example of Task Set in the New Model

n=3

T_1=50, D_1=10
$nhlbc_1$=1
lbc_1^1=0, $lhlbc_1^1$=0
$nhubc_1$=2
ubc_1^1=4, $lhubc_1^1$=0
ubc_1^2=3, $lhubc_1^2$=1, $ihubc_1^{2,1}$=2

T_2=150, D_2=15
$nhlbc_2$=1
lbc_2^1=0, $lhlbc_2^1$=0
$nhubc_2$=2
ubc_2^1=7, $lhubc_2^1$=0
ubc_2^2=6, $lhubc_2^2$=1, $ihubc_2^{2,1}$=1

T_3=500, D_3=500
$nhlbc_3$=1
lbc_3^1=0, $lhlbc_3^1$=0
$nhubc_3$=1
ubc_3^1=5, $lhubc_3^1$=0

Upper and lower bounds on execution times of a job as a function of jobs that executed before it

Example of Task Set in the New Model

n=3

T_1=50, D_1=10

$nhlbc_1$=1

lbc_1^1=0, $lhlbc_1^1$=0

$nhubc_1$=2

ubc_1^1=4, $lhubc_1^1$=0

ubc_1^2=3, $lhubc_1^2$=1, $ihubc_1^{2,1}$=2

T_2=150, D_2=15

$nhlbc_2$=1

lbc_2^1=0, $lhlbc_2^1$=0

$nhubc_2$=2

ubc_2^1=7, $lhubc_2^1$=0

ubc_2^2=6, $lhubc_2^2$=1, $ihubc_2^{2,1}$=1

T_3=500, D_3=500

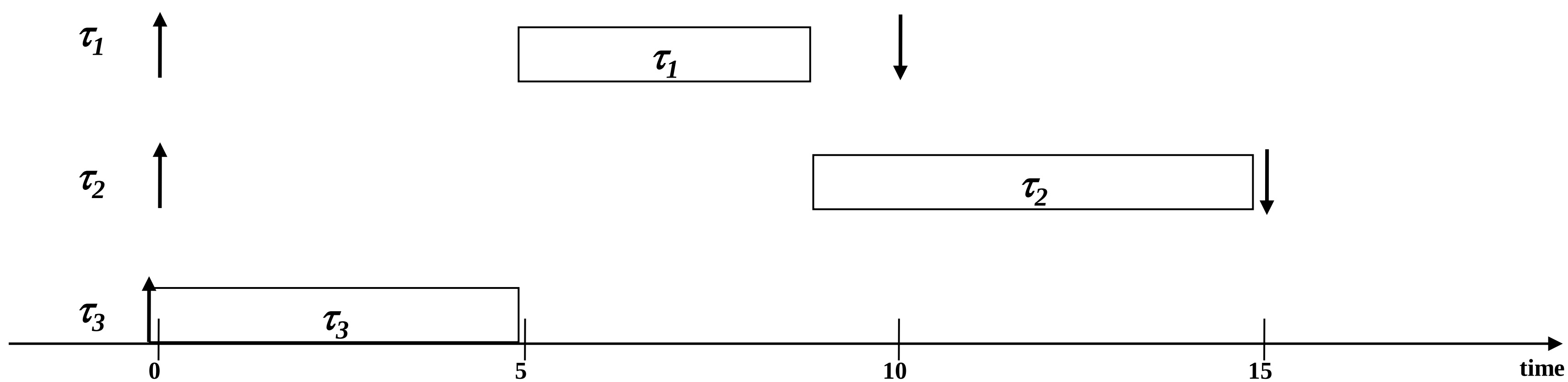
$nhlbc_3$=1

lbc_3^1=0, $lhlbc_3^1$=0

$nhubc_3$=1

ubc_3^1=5, $lhubc_3^1$=0

Schedule these tasks with non-preemptive fixed priority scheduling



This is the worst-case arrival for τ_2. With new model: deadline of τ_2 is met.

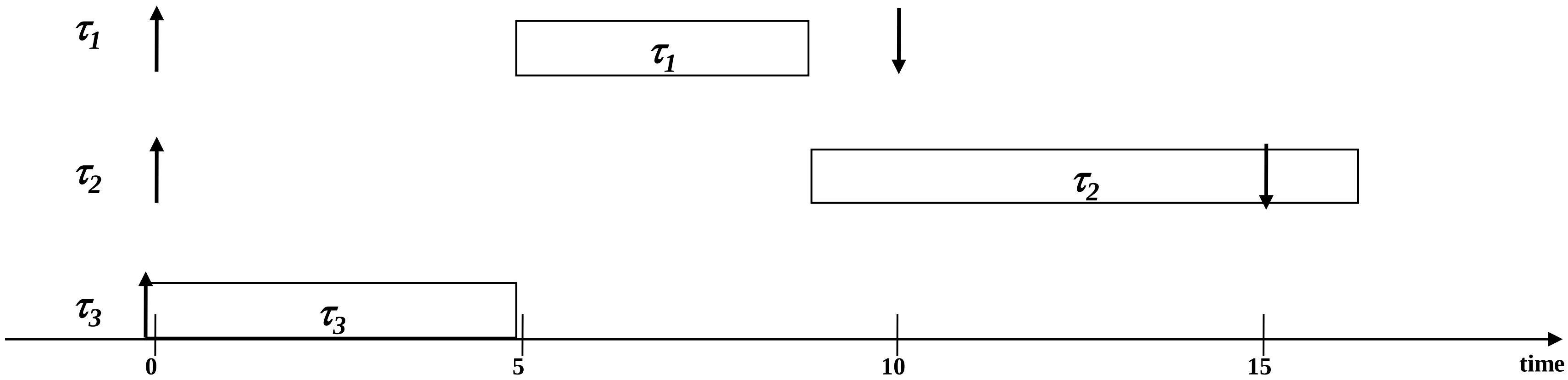
Example of Task Set in the Old Model

n=3

T_1=50, D_1=10 T_2=150, D_2=15 T_3=500, D_3=500

C_1=4 C_2=7 C_3=5

Schedule these tasks with non-preemptive fixed priority scheduling



This is the worst-case arrival for τ_2. With old model: deadline of τ_2 is missed.

Example of Task Set in the New Model

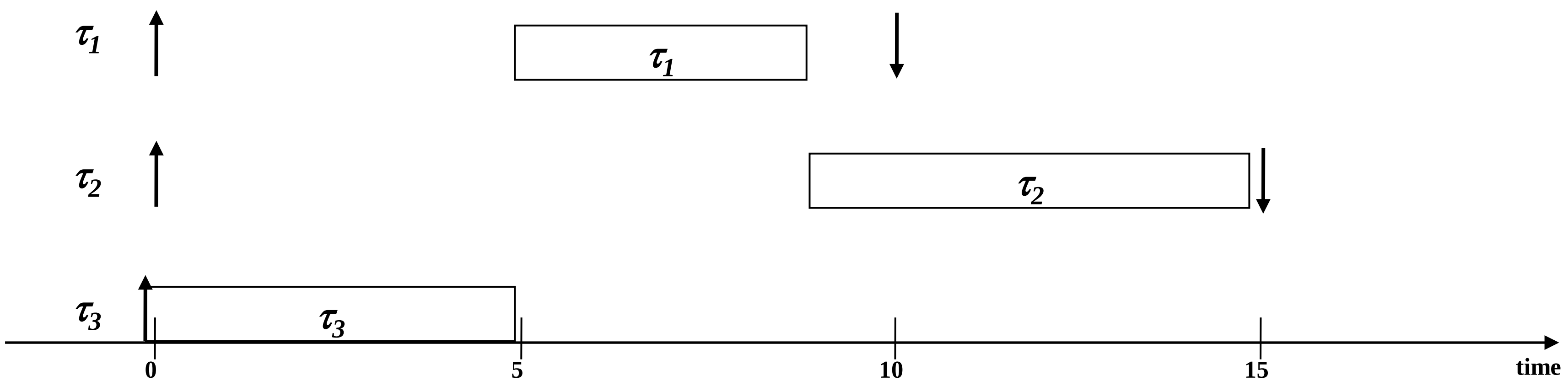
n=3

T_1=50, D_1=10
$nhlbc_1$=1
lbc_1^1=0, $lhlbc_1^1$=0
$nhubc_1$=2
ubc_1^1=4, $lhubc_1^1$=0
ubc_1^2=3, $lhubc_1^2$=1, $ihubc_1^{2,1}$=2

T_2=150, D_2=15
$nhlbc_2$=1
lbc_2^1=0, $lhlbc_2^1$=0
$nhubc_2$=2
ubc_2^1=7, $lhubc_2^1$=0
ubc_2^2=6, $lhubc_2^2$=1, $ihubc_2^{2,1}$=1

T_3=500, D_3=500
$nhlbc_3$=1
lbc_3^1=0, $lhlbc_3^1$=0
$nhubc_3$=1
ubc_3^1=5, $lhubc_3^1$=0

Schedule these tasks with non-preemptive fixed priority scheduling



This is the worst-case arrival for τ_2. With new model: deadline of τ_2 is met.

Example of Task Set in the New Model

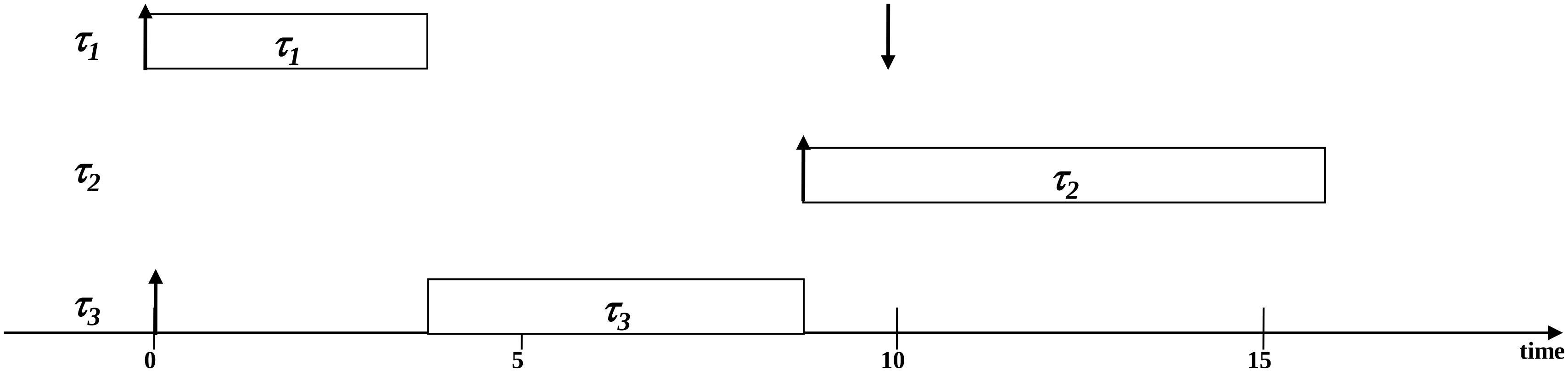
n=3

T_1=50, D_1=10
$nhlbc_1$=1
lbc_1^1=0, $lhlbc_1^1$=0
$nhubc_1$=2
ubc_1^1=4, $lhubc_1^1$=0
ubc_1^2=3, $lhubc_1^2$=1, $ihubc_1^{2,1}$=2

T_2=150, D_2=15
$nhlbc_2$=1
lbc_2^1=0, $lhlbc_2^1$=0
$nhubc_2$=2
ubc_2^1=7, $lhubc_2^1$=0
ubc_2^2=6, $lhubc_2^2$=1, $ihubc_2^{2,1}$=1

T_3=500, D_3=500
$nhlbc_3$=1
lbc_3^1=0, $lhlbc_3^1$=0
$nhubc_3$=1
ubc_3^1=5, $lhubc_3^1$=0

Schedule these tasks with non-preemptive fixed priority scheduling



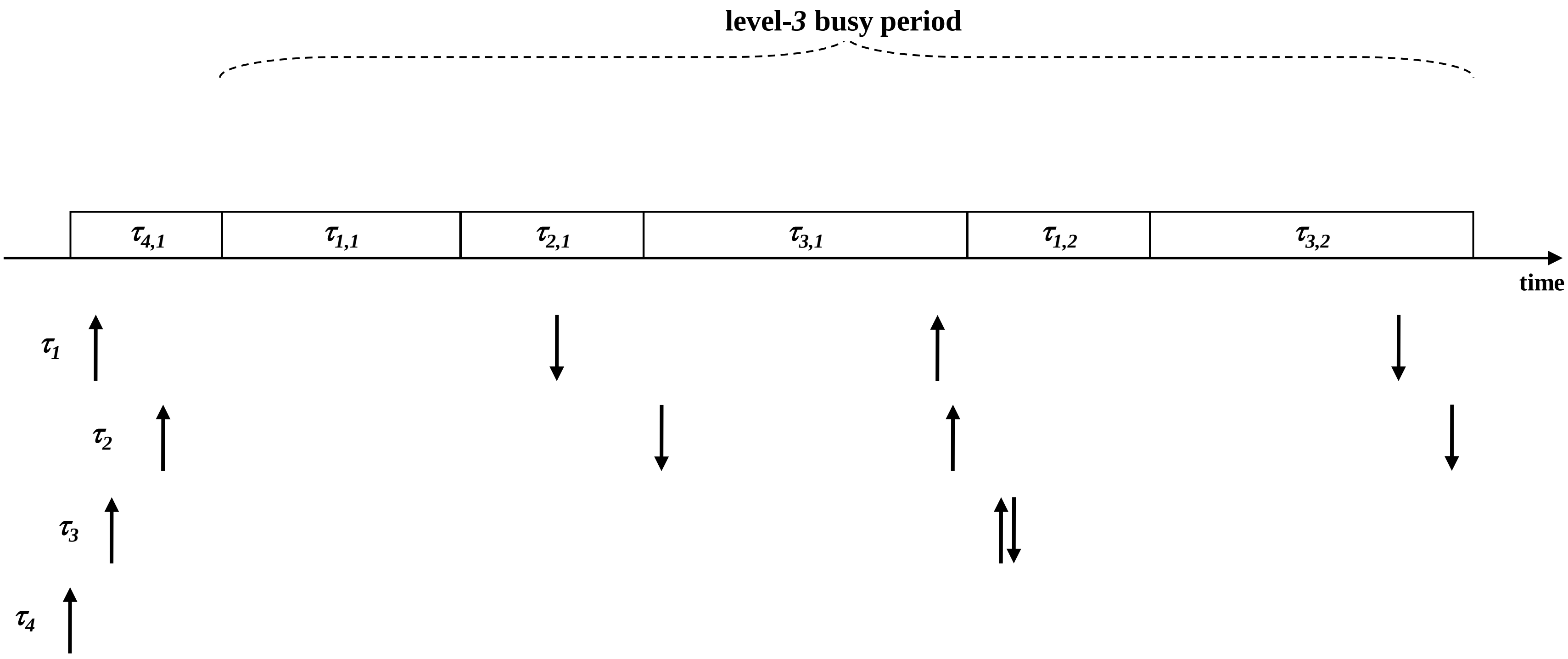
For another arrival for τ_2. Job of τ_2 has longer execution time but deadline of τ_2 is met.

Problem discussed in this presentation

How to perform schedulability analysis of tasks described with the new model under fixed-priority non-preemptive scheduling?

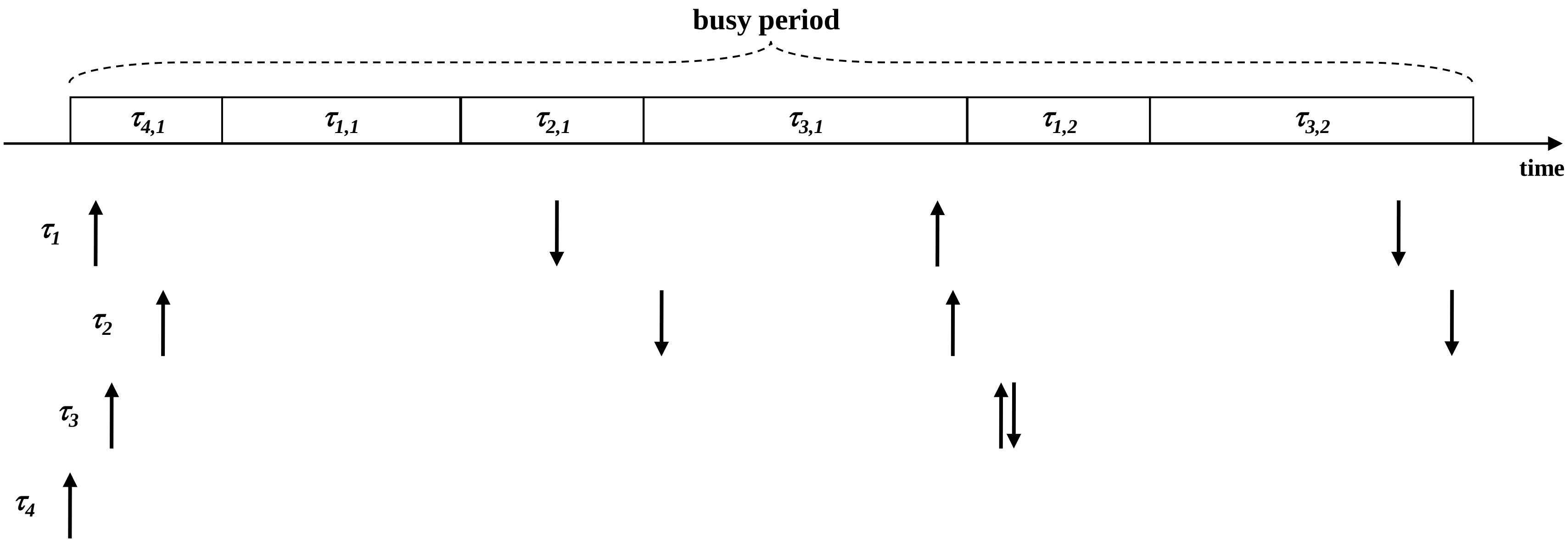
Ideas that do not work

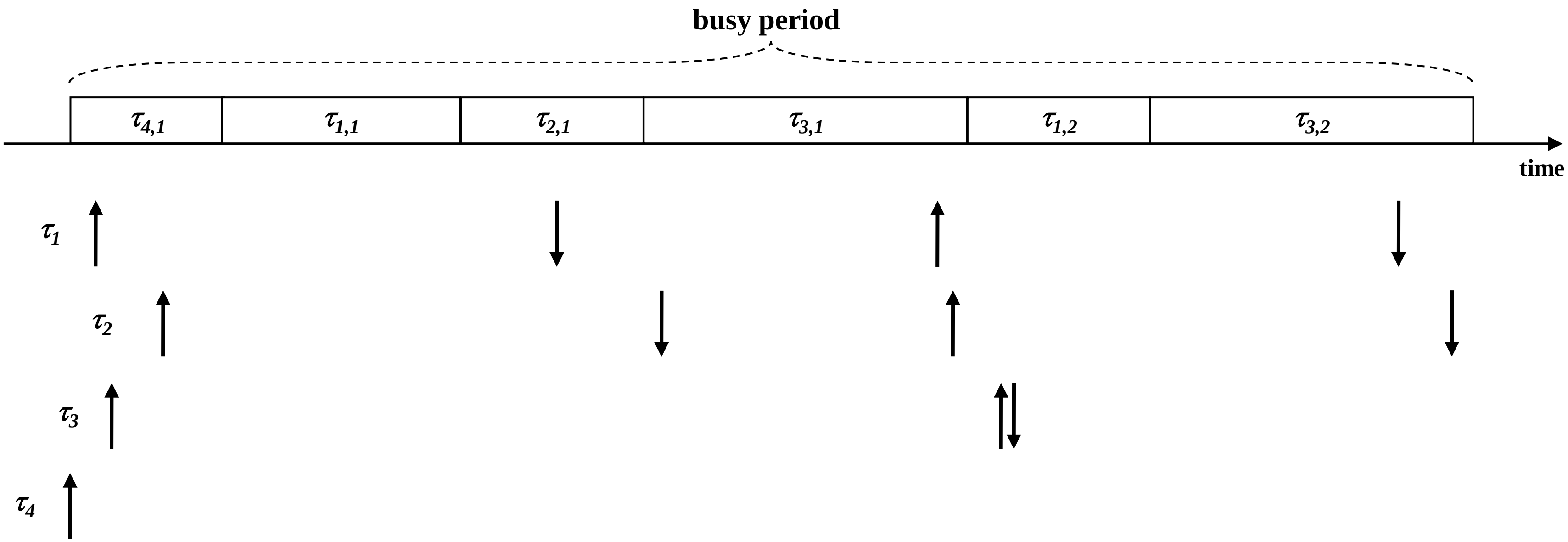
- L&L critical instant



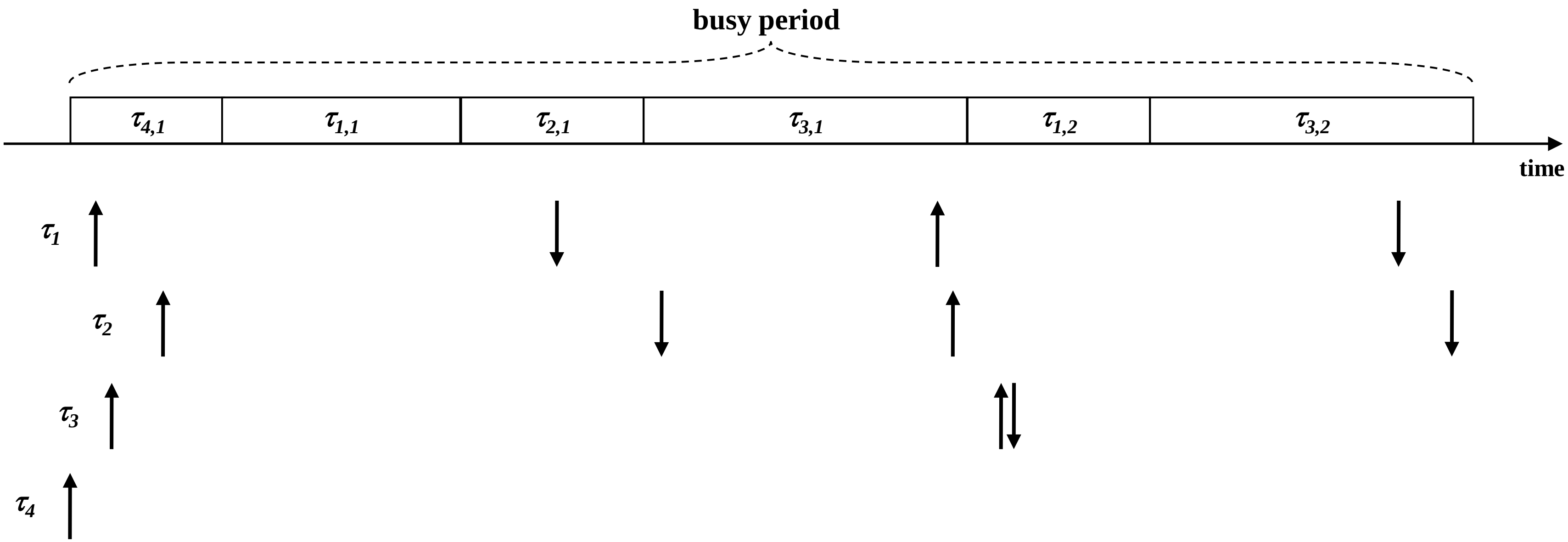
Ideas that do not work

- L&L critical instant
- Level-*i* busy period


busy period
$\tau_{4,1}$
$\tau_{1,1}$
$\tau_{2,1}$
$\tau_{3,1}$
$\tau_{1,2}$
$\tau_{3,2}$
time
τ_1
τ_2
τ_3
τ_4

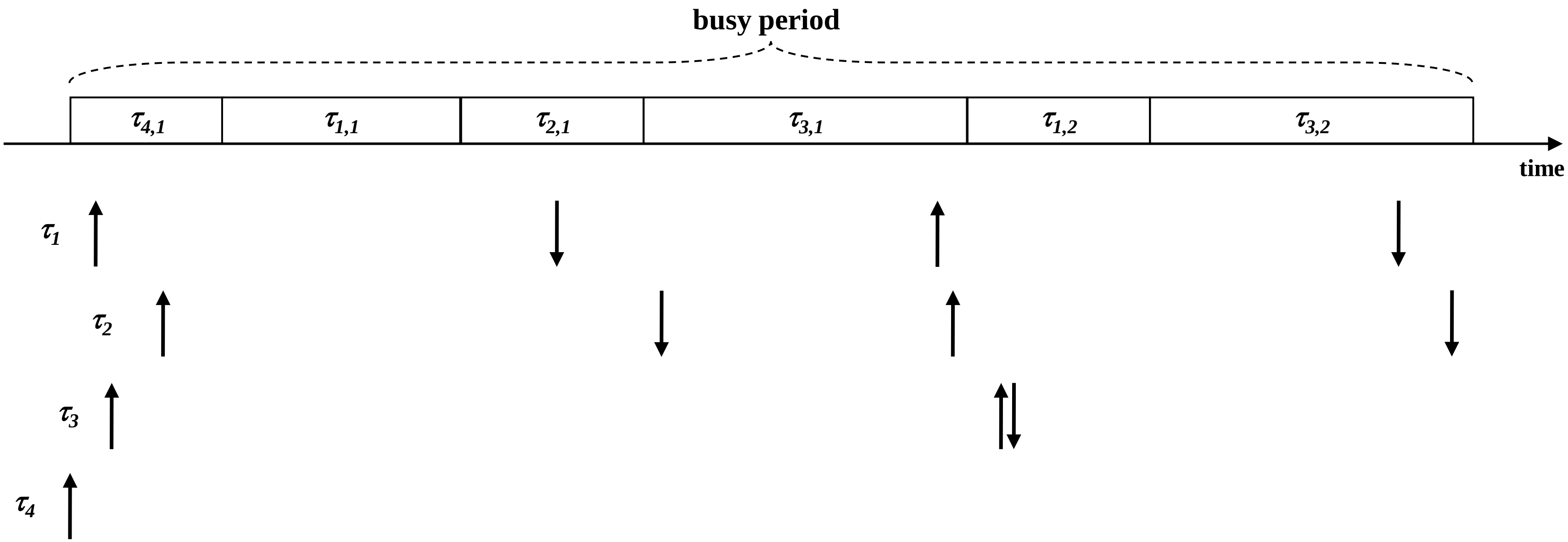


1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $R_{i,q}$:= compute the maximum response time of $\tau_{i,q}$ — the *q*:th job of task τ_i — in every busy period of length at most L
5. $R_i := \max_{q=1..Q} R_{i,q}$



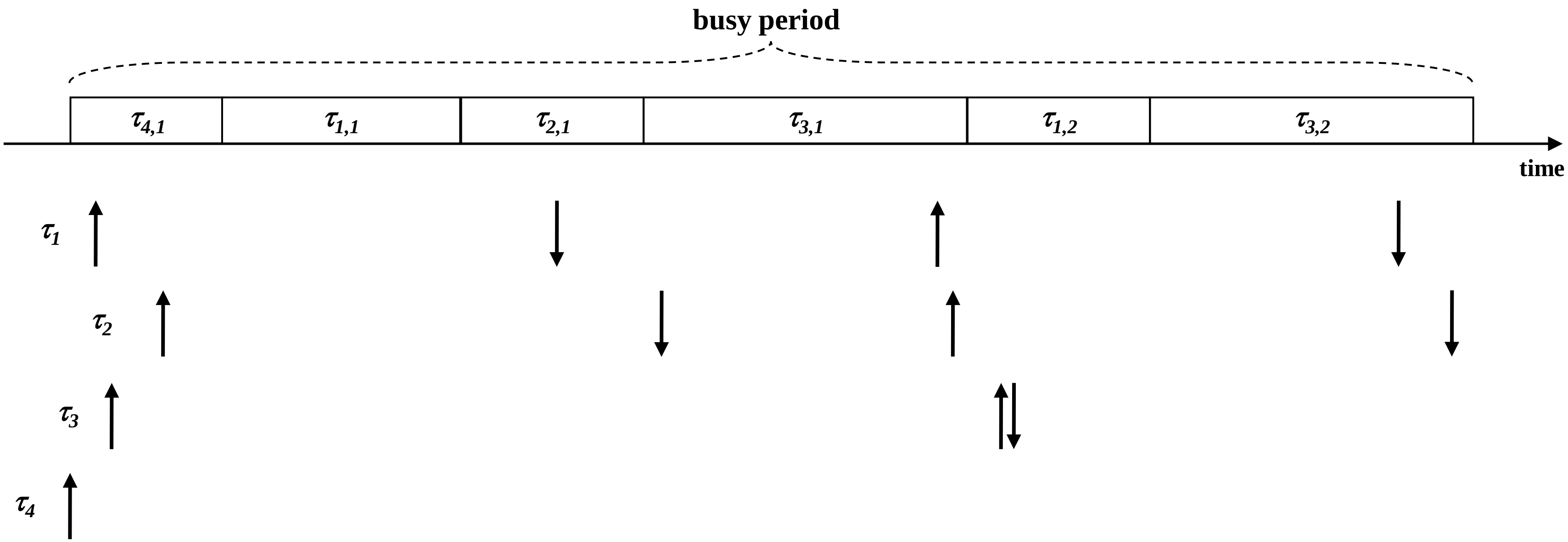
An idea that almost works for computing the response time of task τ_i

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $R_{i,q}$:= compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L
5. $R_i := \max_{q=1..Q} R_{i,q}$



An idea that works for computing the response time of task τ_i

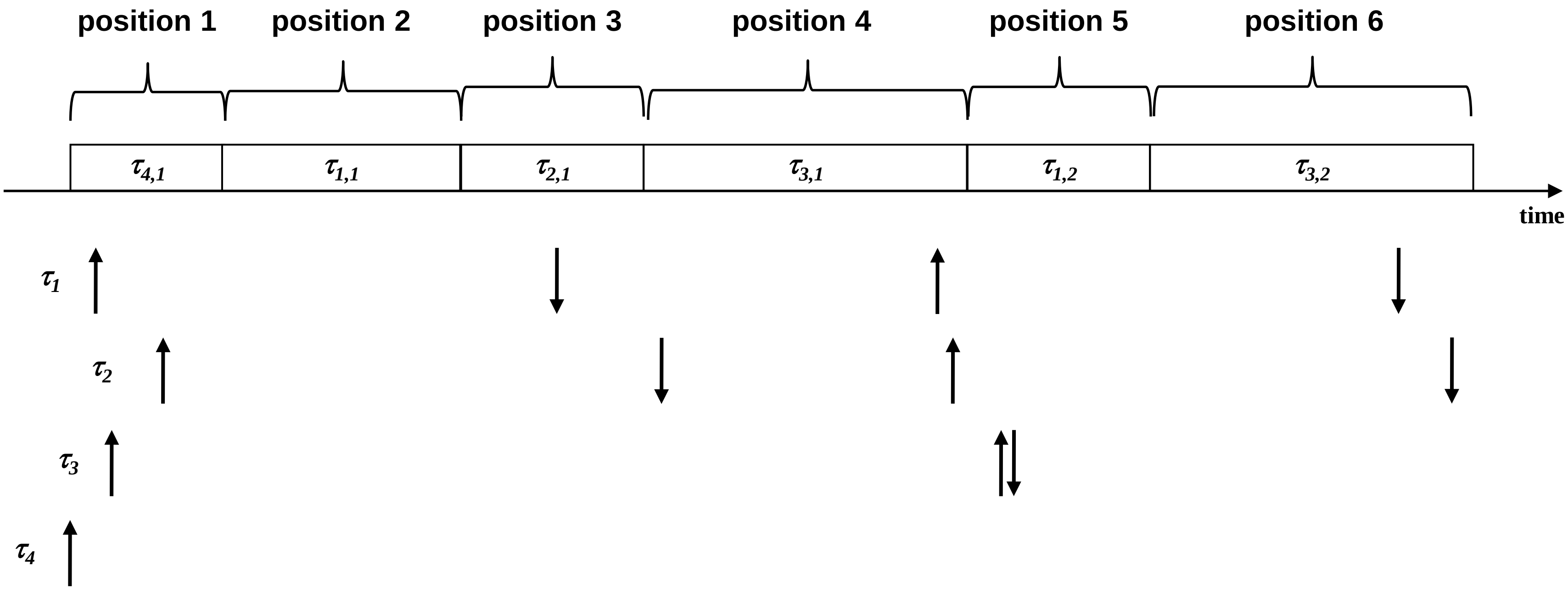
1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. <$R_{i,q}$,$valid_{i,q}$> := compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}}$



<u>An idea that works for computing the response time of task</u>

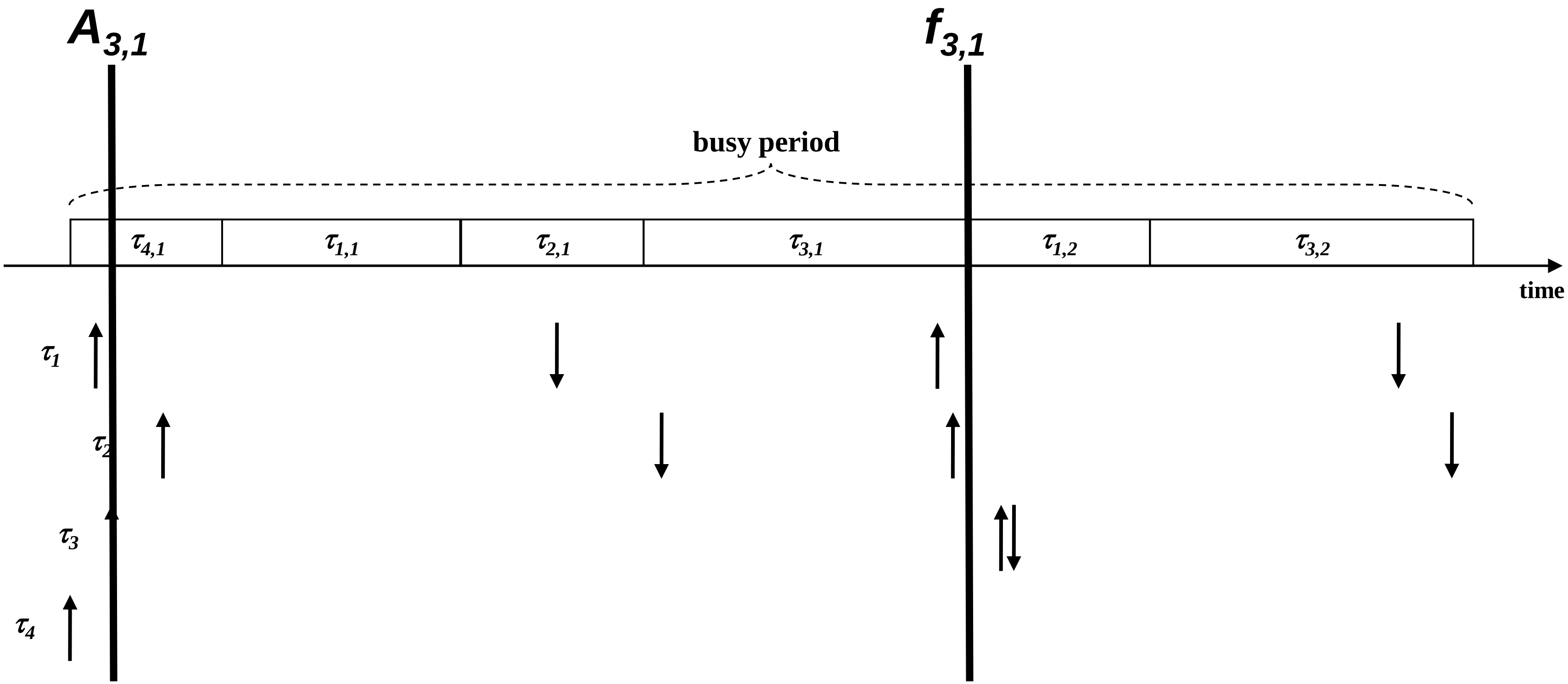
1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $\langle R_{i,q}, valid_{i,q} \rangle$:= compute the maximum response time of — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy peri
5. $R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}$

How to perform step 1 and step 4?



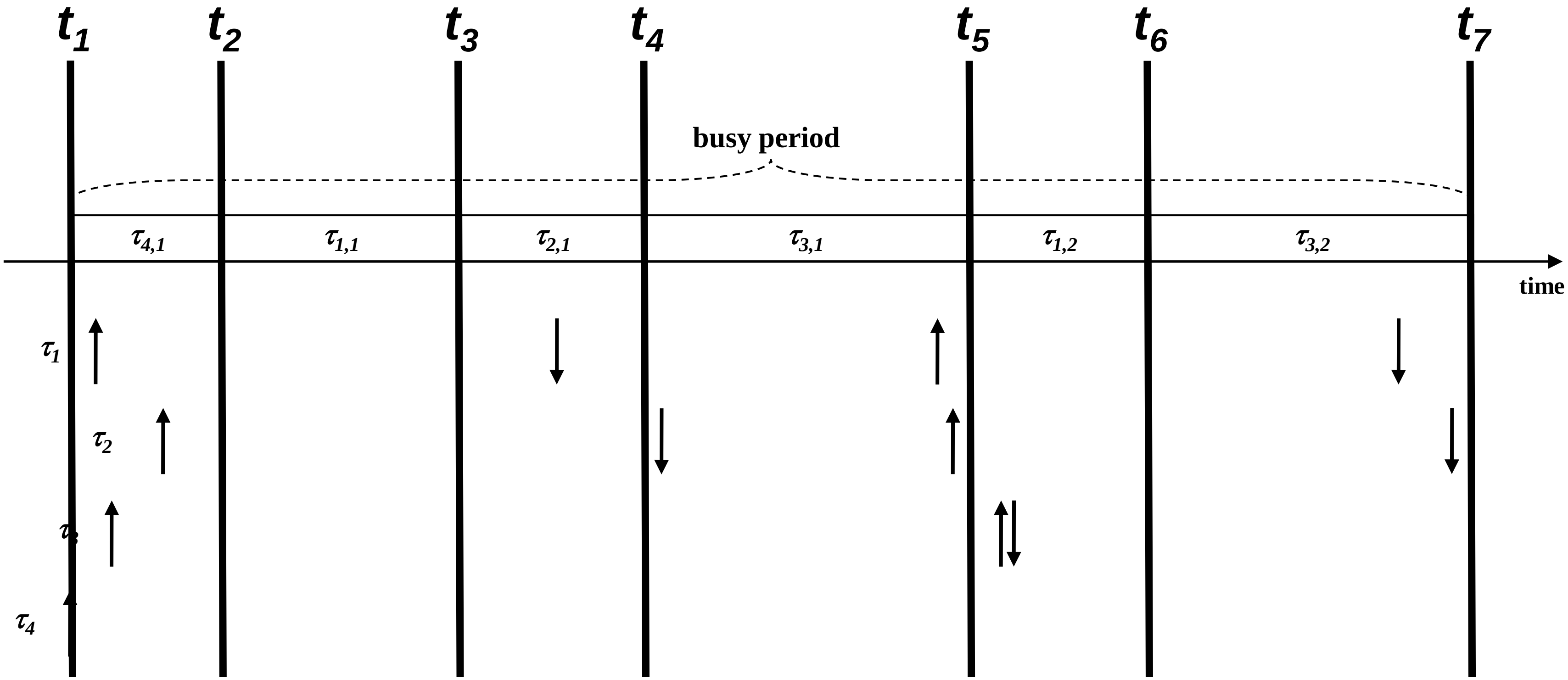
An idea that works for computing the response time of task τ_i

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $\langle R_{i,q}, valid_{i,q} \rangle$:= compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}$



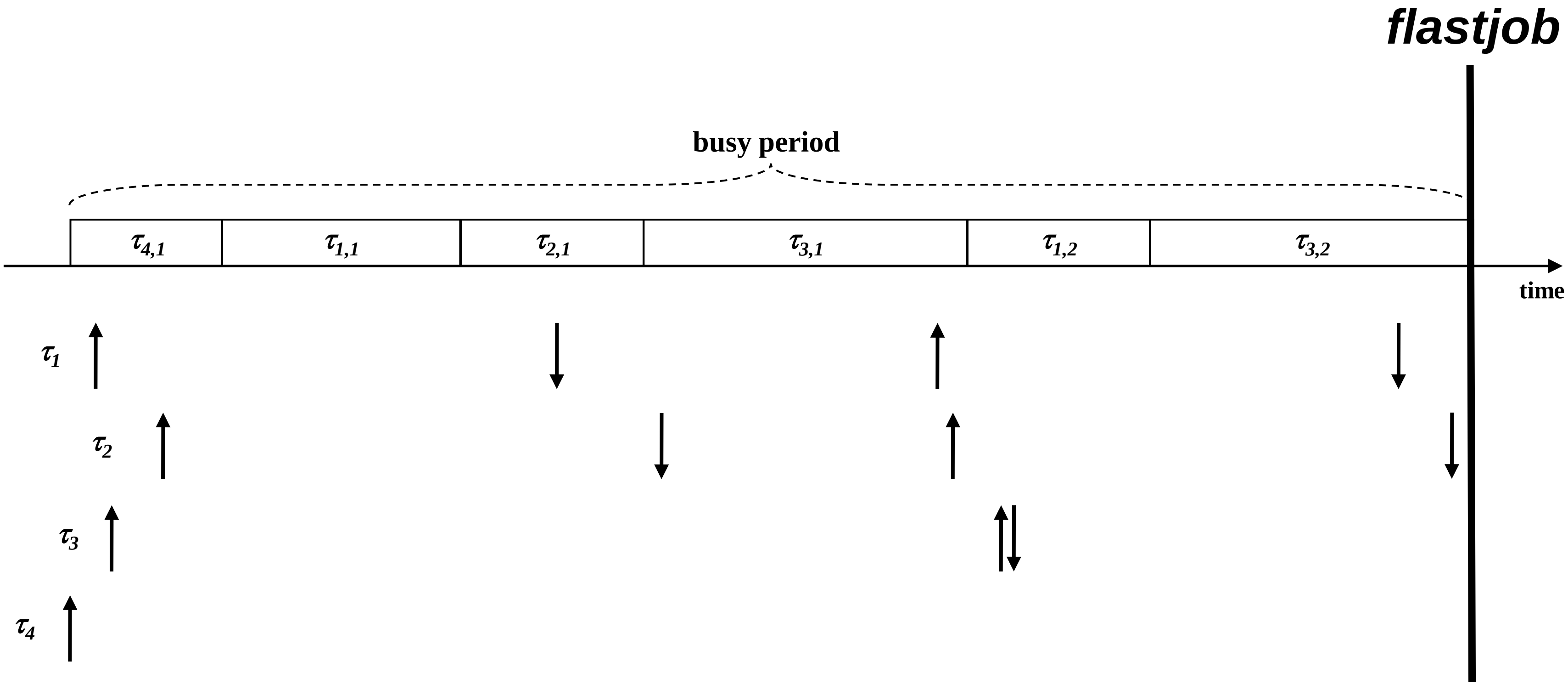
An idea that works for computing the response time of task τ_i

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $<R_{i,q}, valid_{i,q}>$:= compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q\ and\ valid_{i,q}} R_{i,q}}$



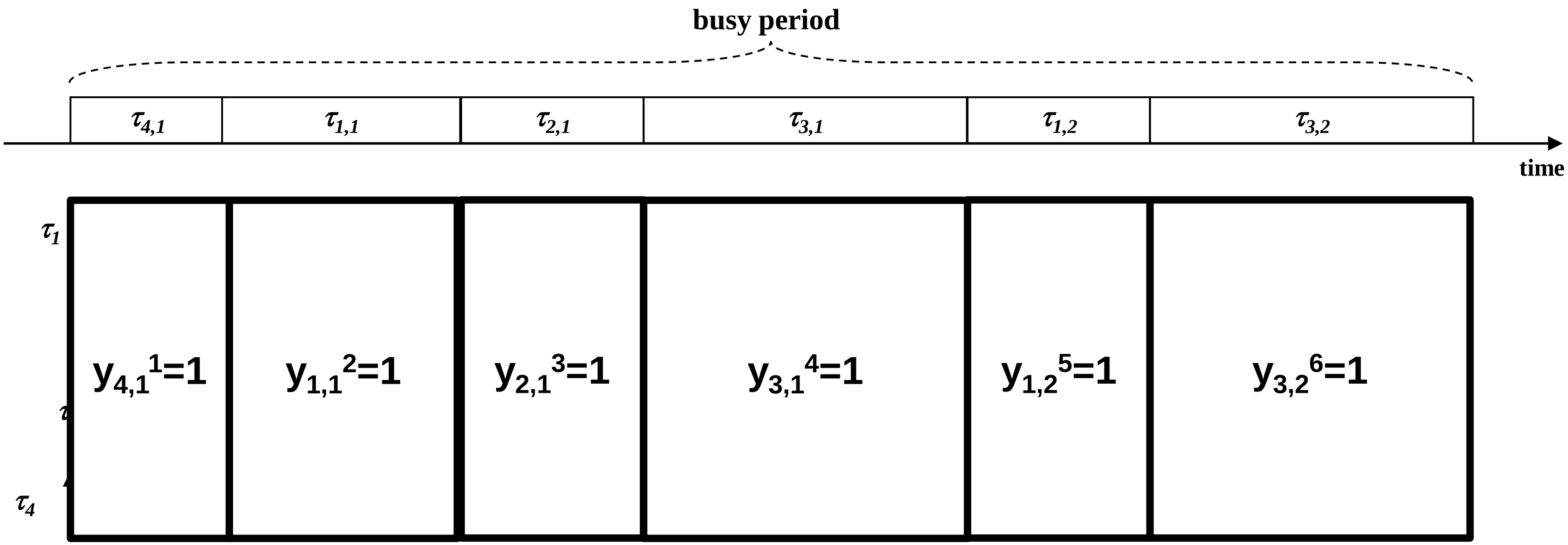
An idea that works for computing the response time of task τ_i

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $\langle R_{i,q}, valid_{i,q} \rangle$:= compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}$



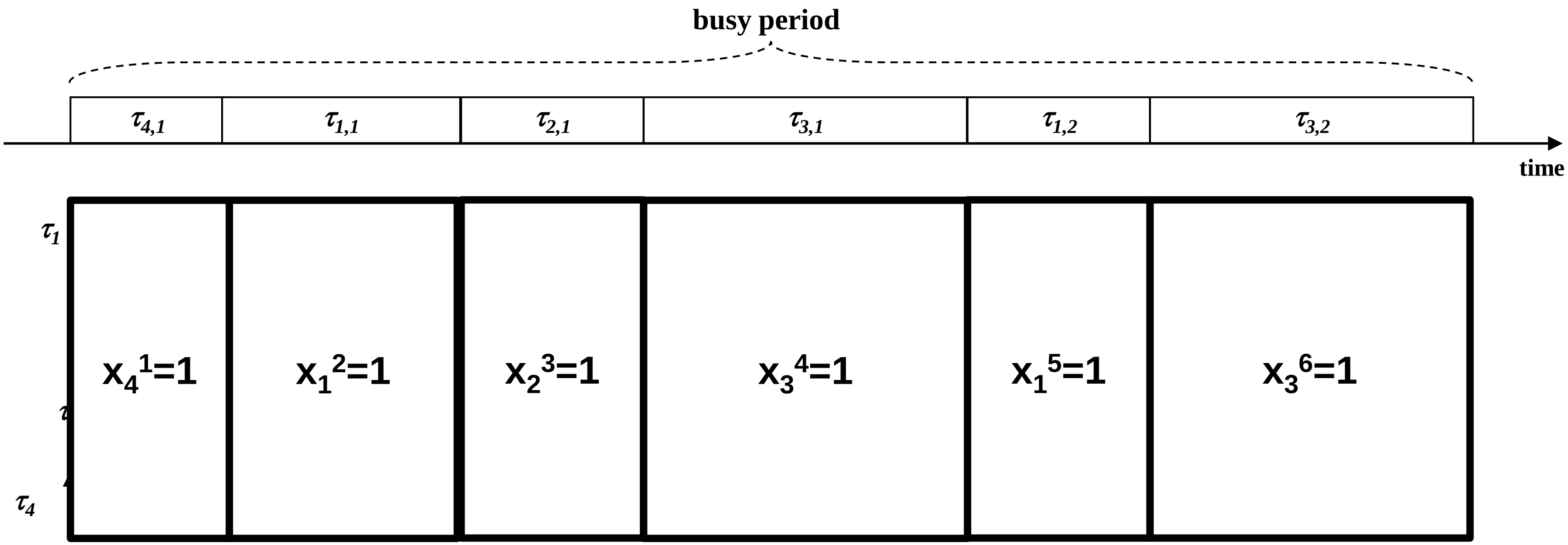
<u>An idea that works for computing the response time of task τ_i</u>

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. <$R_{i,q}$,$valid_{i,q}$> := compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}}$



<u>An idea that works for computing the response time of task τ_i</u>

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. <$R_{i,q}$,$valid_{i,q}$> := compute the maximum response time of $\tau_{i,q}$ — the *q*:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}}$



An idea that works for computing the response time of task τ_i

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. $\langle R_{i,q}, valid_{i,q} \rangle$:= compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \; and \; valid_{i,q}} R_{i,q}}$

Perform step 1: Compute the maximum duration of a busy period

Represent a schedule.

x_j^p = 1 iff a job of task τ_j executes in position p in busy period

$y_{j,k}^p$ = 1 iff job $\tau_{j,k}$ executes in position p in busy period

t_k = time of k:th context switch in busy period

Other variables

$A_{j,k}$ = arrival time of $\tau_{j,k}$

$f_{j,k}$ = finishing time of $\tau_{j,k}$

ftlastjob= time when the busy period ends

Maximize *ftlastjob*- t_1 subject to constraints …. [see paper]

---

1. L := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. <$R_{i,q}$,$valid_{i,q}$> := compute the maximum response time of $\tau_{i,q}$ — the q:th job of task τ_i — in every busy period of length at most L and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}}$

Perform step 4: Compute the maximum response time of $\tau_{i,q}$ during every busy period of duration at most *L*

Represent a schedule.

x_j^p = 1 iff a job of task τ_j executes in position *p* in busy period

$y_{j,k}^p$ = 1 iff job $\tau_{j,k}$ executes in position *p* in busy period

t_k = time of *k*:th context switch in busy period

Other variables

$A_{j,k}$ = arrival time of $\tau_{j,k}$

$f_{j,k}$ = finishing time of $\tau_{j,k}$

ftlastjob= time when the busy period ends

Maximize $f_{j,k} - A_{j,k}$ subject to constraints …. [see paper]

---

1. *L* := Compute the maximum duration of a busy period
2. $Q_i := \lceil L/T_i \rceil$
3. **for** q := 1 to Q_i **do**
4. <$R_{i,q}$,$valid_{i,q}$> := compute the maximum response time of $\tau_{i,q}$ — the *q*:th job of task τ_i — in every busy period of length at most *L* and compute whether $\tau_{i,q}$ exist in the busy period.
5. $\boldsymbol{R_i = max_{q=1..Q \text{ and } valid_{i,q}} R_{i,q}}$

Conclusion

It is possible to compute exact response times of tasks scheduled by non-preemptive fixed-priority scheduling where execution times depend on history.

Thanks for listening!