Real-Time End-to-End Scheduling

Embedded System Software Design

Prof. Ya-Shu Chen
National Taiwan University of Science and Technology

End-to-End Scheduling

任務模型

–每個任務需要按一定順序在一組處理器上執行

–每個任務可能需要不同的順序

- Task model
  - Each task needs to execute on a set of processors in a certain order
  - Each task may require a different order
- Problems in End-to-End scheduling 端到端調度中的問題
  - Priority assignment
    –優先分配
    •為任務分配固定的優先級，以便系統可調度
    - Assign fixed priorities to tasks so that the system is schedulable
  - Synchronization of tasks
    –任務同步
    •控制子任務實例（非第一個子任務）的釋放
    - Control the releases of subtask instances (non-first subtasks)
  - Schedulability analysis
    - For a given priority assignment and a given synchronization protocol, whether every instance of each task meets its deadline

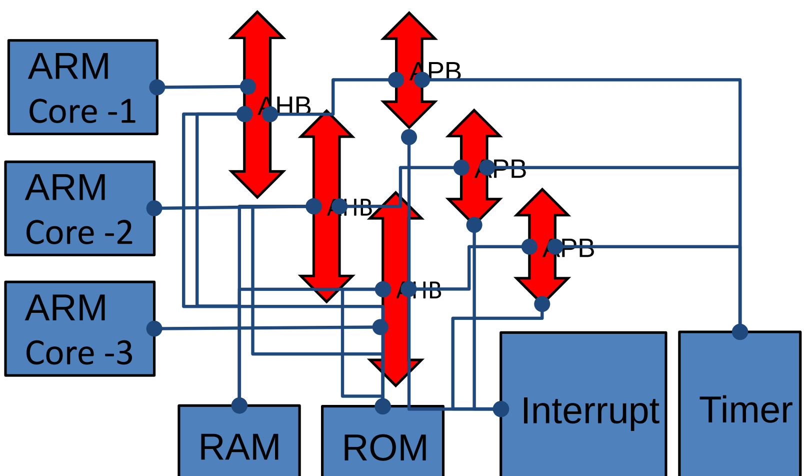
–可調度性分析

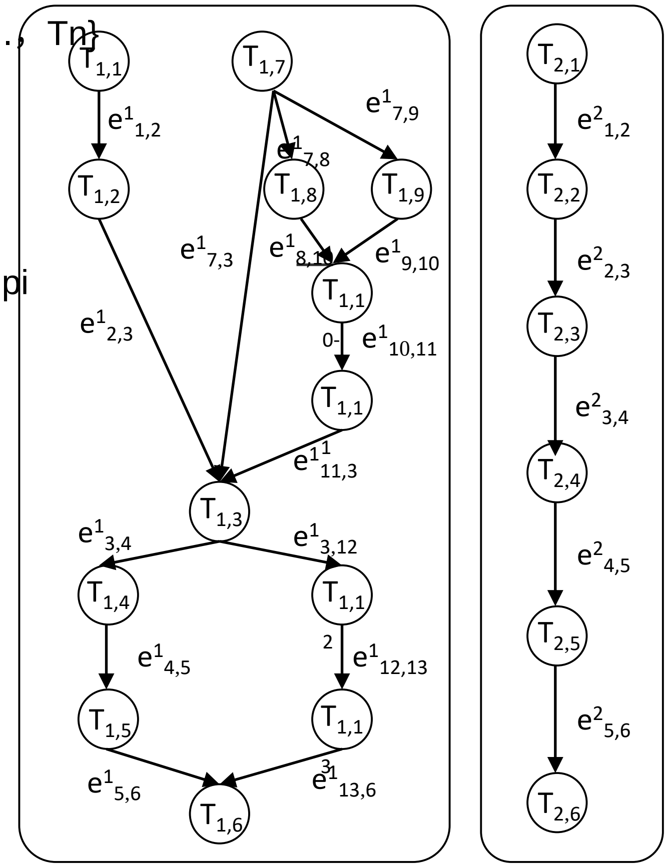
•對於給定的優先級分配和給定的同步協議，每個任務的每個實例是否都滿足其截止日期

所有工作都要在 Deadline 之前就
最長的那一條路徑需要滿足

System Model

- Platform: A set of processors 平台：一組處理器
- Task graph, $G = \{T_1, T_2, \ldots, T_n\}$ 任務圖，G = {T1，T2，…，Tn}
  - Sink node –下沉節點
  - Deadline: d_i –截止日期：di
  - Precedence edge: $e^i_{j,k}$ –優先權邊緣：eij，k
  - Predecessors and Sucessors –前任和後任
  - Period or Minimum separation time: p_i –週期或最短分離時間：pi
- 特徵 Characteristics of $T_{i,j}$:
  - Execution time (on processor m): $c^m_{i,j}$





T_1, p_1, d_1 T_2, p_2, d_2

Priority Assignment 優先分配

•如果事先知道所有執行的任務，則可以離線找到可行的優先級分配

- To find feasible priority assignments off-line if all tasks executed are known in prior
- NP-hard problem •NP難題
- Algorithms
  - Branch and bound
  - Search algorithm
    - Simulated annealing
    - Generic algorithm
  - Heuristic
    - Deadline assignment

•算法
–分支定界
–搜索算法
•模擬退火
•通用算法
- 啟發式
•截止日期分配

Deadline Assignment 截止日期分配

- Ultimate deadline 最終期限
  - $UD_{i,k} = D_i$
- Effective deadline 有效期限
  - $ED_{i,k} = D_i - \sum_{l=k+1}^{n(i)} e_{i,l}$
- Proportional deadline 比例截止
  - $PD_{i,k} = D_i e_{i,k} / e_i$
- Normalized Proportional deadline 歸一化比例截止期限
  - $NPD_{i,k} = D_i \frac{e_{i,k} U(V_{i,k})}{\sum_{l=1}^{n(i)} e_{i,l} U(V_{i,l})}$ 多考慮了 U
  - $U(V_{i,l})$ is the total utilization of the all the subtasks that execute on the processor $V_{i,l}$ U是在處理器V上執行的所有子任務的總利用

Example

$T_{i,k}$	$V_{i,k}$	p_i	$e_{i,k}$	$UD_{i,k}$	$ED_{i,k}$	$PD_{i,k}$	$NPD_{i,k}$
$T_{1,1}$	$\boldsymbol{P_1}$	15	1	15	11	3	2.0
$T_{1,3}$	$\boldsymbol{P_1}$	15	2	15	15	6	4.1
$T_{2,1}$	$\boldsymbol{P_1}$	20	4	20	20	20	20
$T_{3,1}$	$\boldsymbol{P_2}$	2	1	2	2	2	2
$T_{1,2}$	$\boldsymbol{P_2}$	15	2	15	13	6	8.9
$T_{4,1}$	$\boldsymbol{P_2}$	20	5	20	20	20	20

$PD_{i,k} = \frac{D_i \times e_{ik}}{\bar{e_i}}$

11: $\frac{15\times 1}{1+2+2}$ 41: $\frac{20\times 5}{5}$

12 = $\frac{15\times 2}{1+2+2}$

13: $\frac{15\times 2}{1+2+2}$

21: $\frac{20\times 4}{4}$

31: $\frac{2\times 1}{1}$

$ED_{i,k} = D_i - \sum_{l>k+1}^{n(i)} e_{il}$

11: $15 - (2+2)^{\to 12,13}$

12: $15 - (2)^{\to 13}$

13: 15

21: 20

31: 2

41: 20

$NPD_{i,k}$ $U_{p1} = \frac{1}{15} + \frac{2}{15} + \frac{4}{20}$ $U_{p2} = \frac{1}{2} + \frac{2}{15} + \frac{5}{20}$

11: $15 \times \frac{1\times 0.4}{(1+2)\times 0.4 + 2\times 0.88}$

12 = $15 \times \frac{2\times 0.88}{(1+2)\times 0.4 + 2\times 0.88}$

13 = $15 \times \frac{2\times 0.4}{(1+2)\times 0.4 + 2\times 0.88}$

21: $20 \times \frac{4\times 0.4}{4\times 0.4}$

31 = $2 \times \frac{1\times 0.88}{1\times 0.88}$ 41: $20 \times \frac{5\times 0.88}{5\times 0.88}$

同步問題

The Synchronization Problem

- Given that 鑑於
  –使用某些固定優先級分配算法將優先級分配給任務鏈中的子任務
  – Priorities are assigned to subtasks in a task chain using some fixed priority assignment algorithm
- How do we coordinate the release of subtasks in a task chain so that 我們如何協調任務鏈中子任務的釋放，以便
  – Precedence constraints among subtasks are satisfied –滿足子任務之間的優先約束
  – Subtask deadlines are met –滿足子任務的截止日期
  – End-to-end deadlines are met –達到了端到端的截止日期

同步協議

Synchronization Protocols

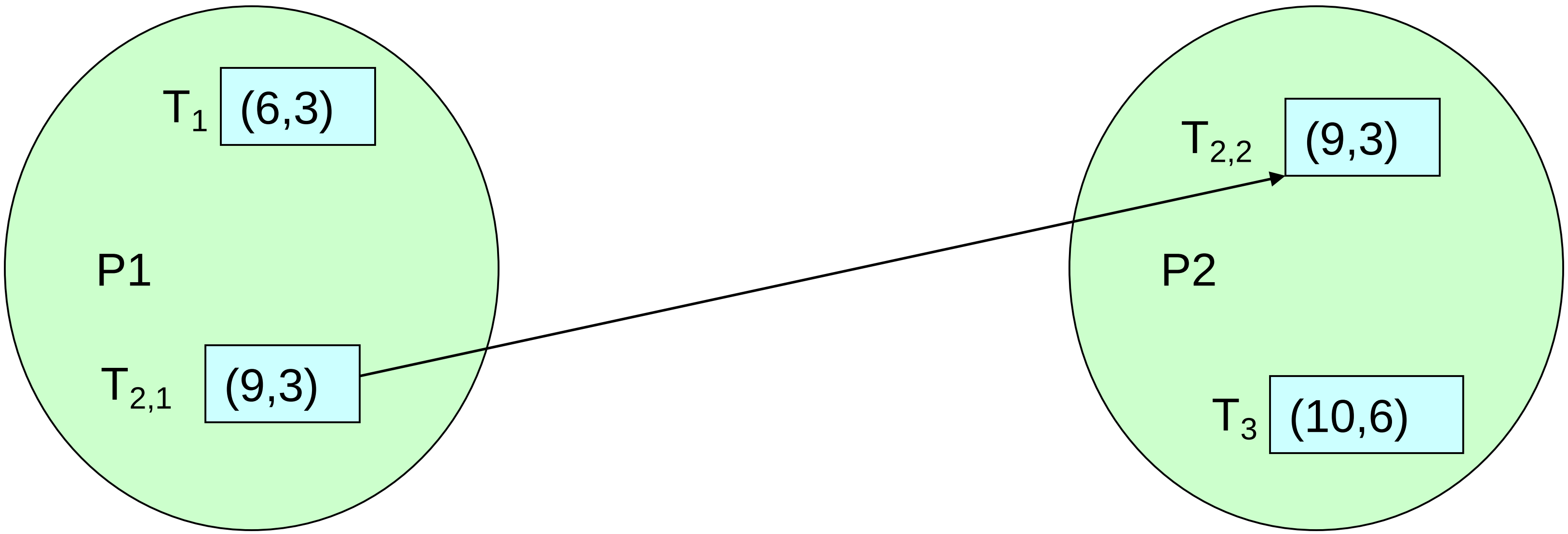
•直接同步（DS）協議
–簡單明了

- Direct Synchronization (DS) Protocol
  – Simple and straightforward
- Phase Modification (PM) Protocol
  – Used by flow-shop tasks
  – Extension called Modified Phase Modification (MPM) Protocol
- Release Guard Protocol
  – Reclaim the idle time

•相位修改（PM）協議
–用於流水車間任務
–稱為修改相位修改（MPM）協議的擴展

•Release Guard協議
–回收空閒時間

Example



$T_{i,j}$ – j^{th} subtask of task T_i

Task T3 releases at 6

(period,execution time)

Period = relative deadline of parent task

期間=上級任務的相對期限

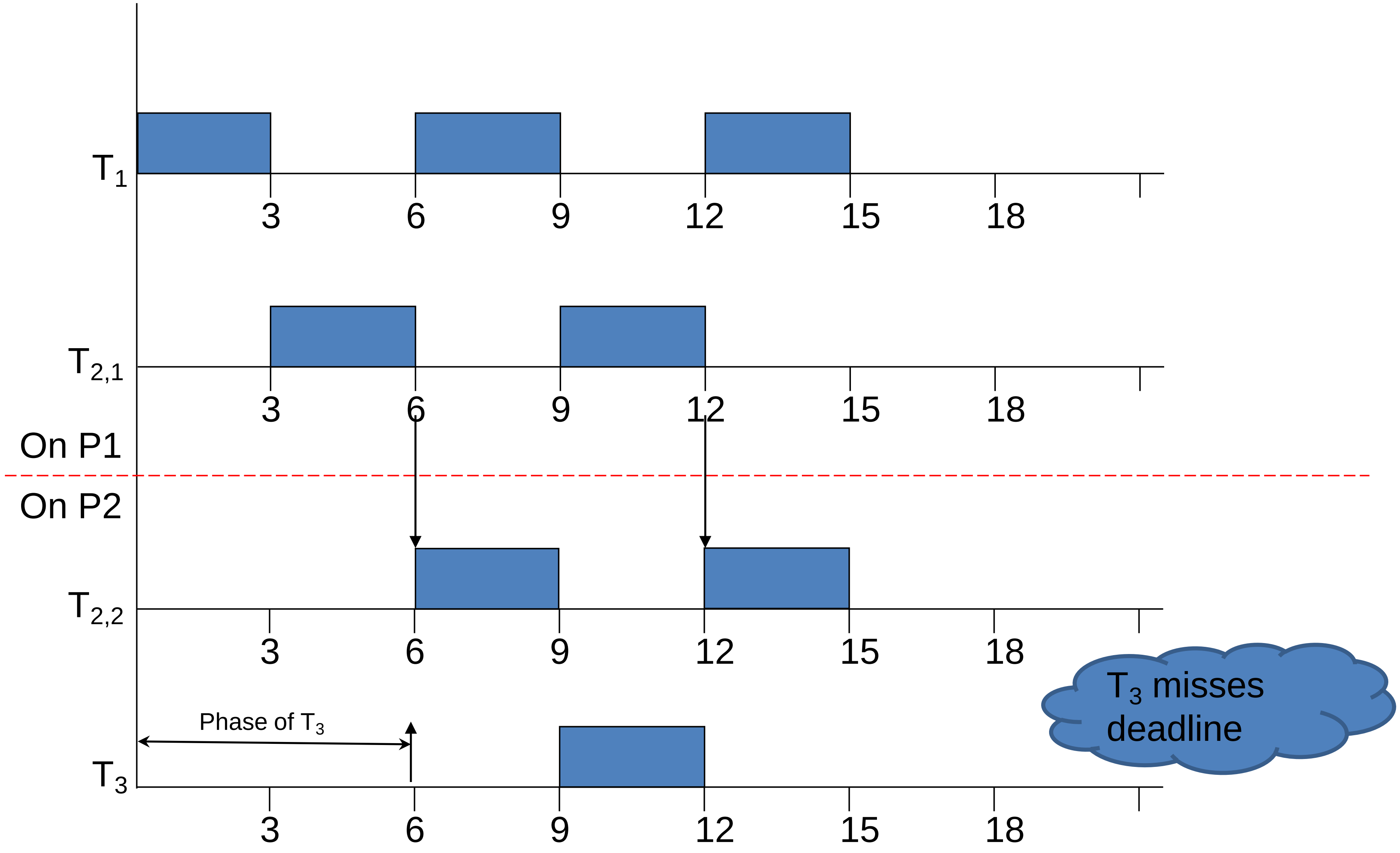
直接同步協議

Direct Synchronization Protocol

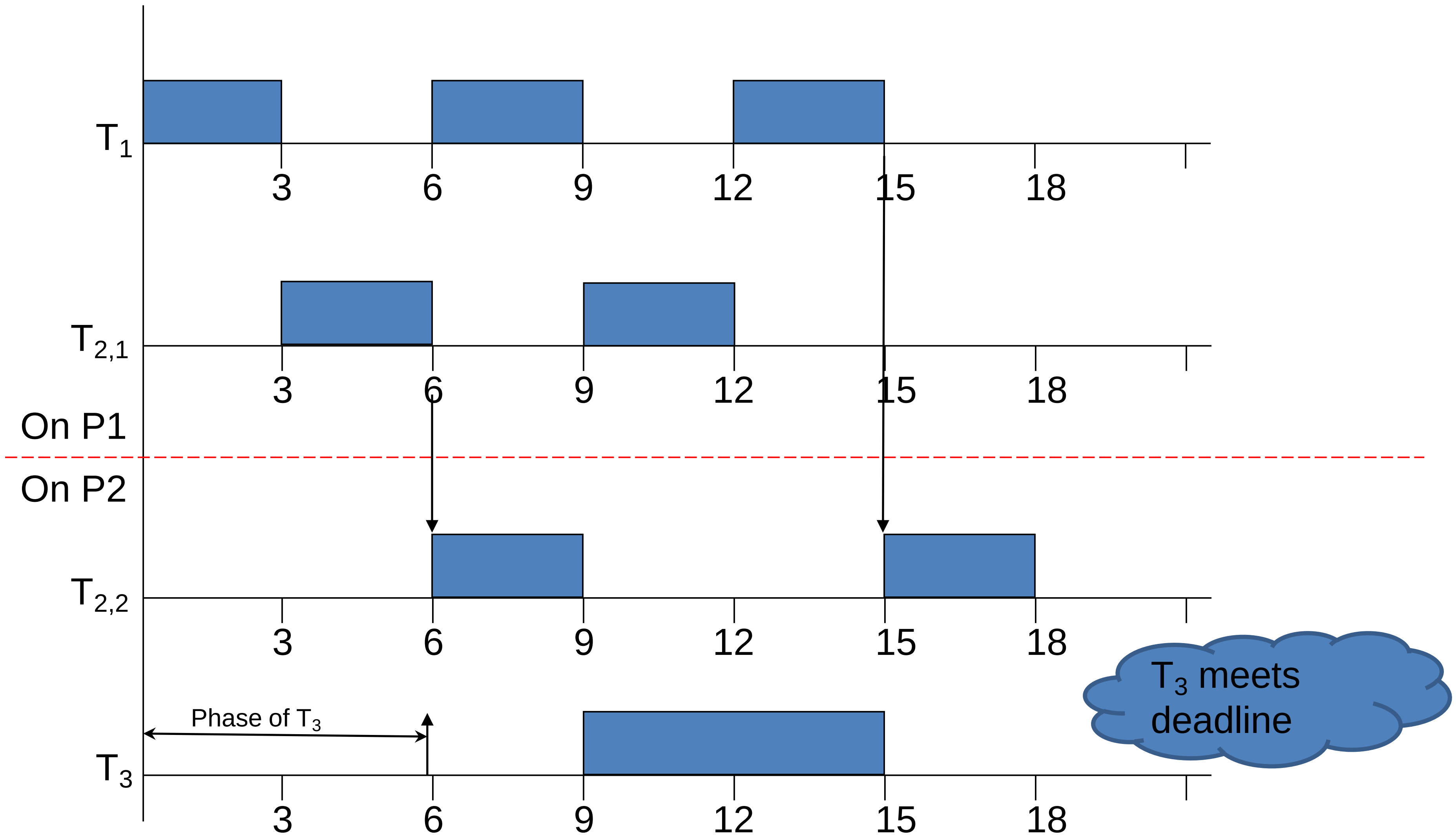
- Greedy strategy 貪婪策略
- On completion of subtask 完成子任務
  - A synchronization signal sent to the next processor 同步信號發送到下一個處理器
  - Successor subtask competes with other tasks/subtasks on the next processor

    後繼子任務與下一處理器上的其他任務/子任務競爭

Greedy Example



Non-greedy Example

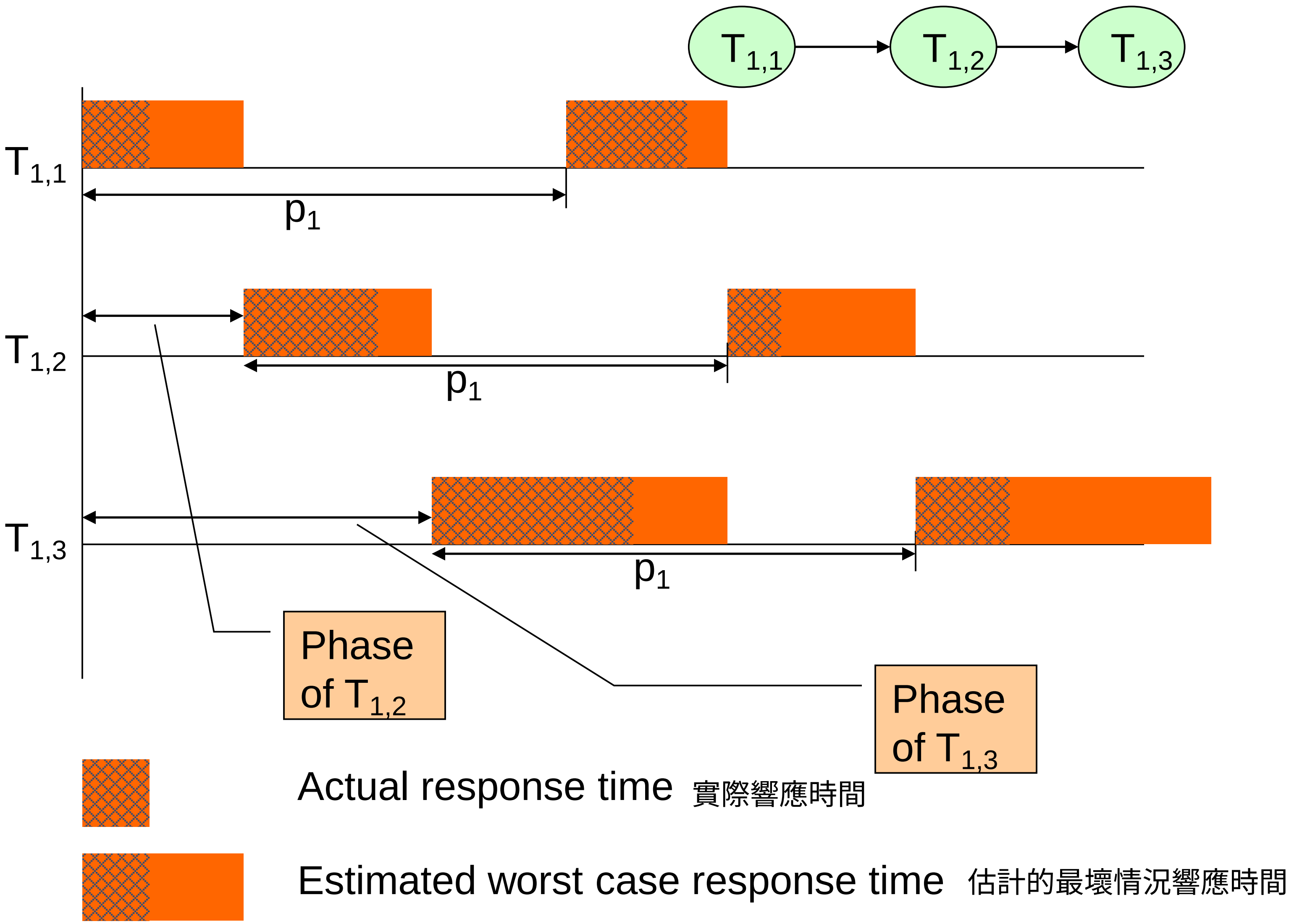


相位修改協議

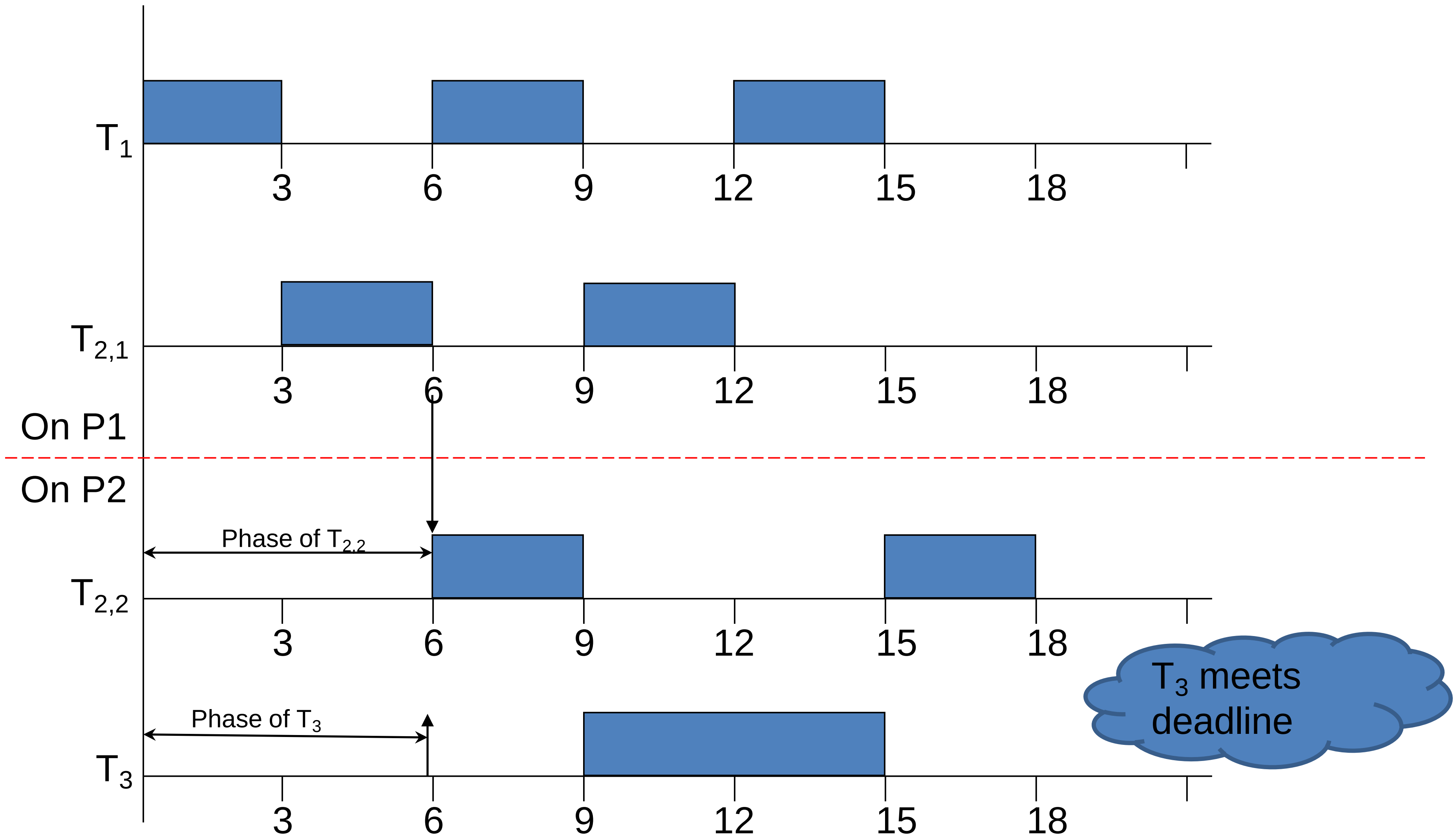
Phase Modification Protocol

- Release subtasks periodically 定期釋放子任務
  –根據其上級任務的期限
  - According to the periods of their parent tasks
- Each subtask given its own phase 每個子任務都有自己的階段
- Phase determined by subtask precedence constraints 由子任務優先級約束確定的階段

Phase Modification Protocol (1/2)



Phase Modification Protocol (2/2)



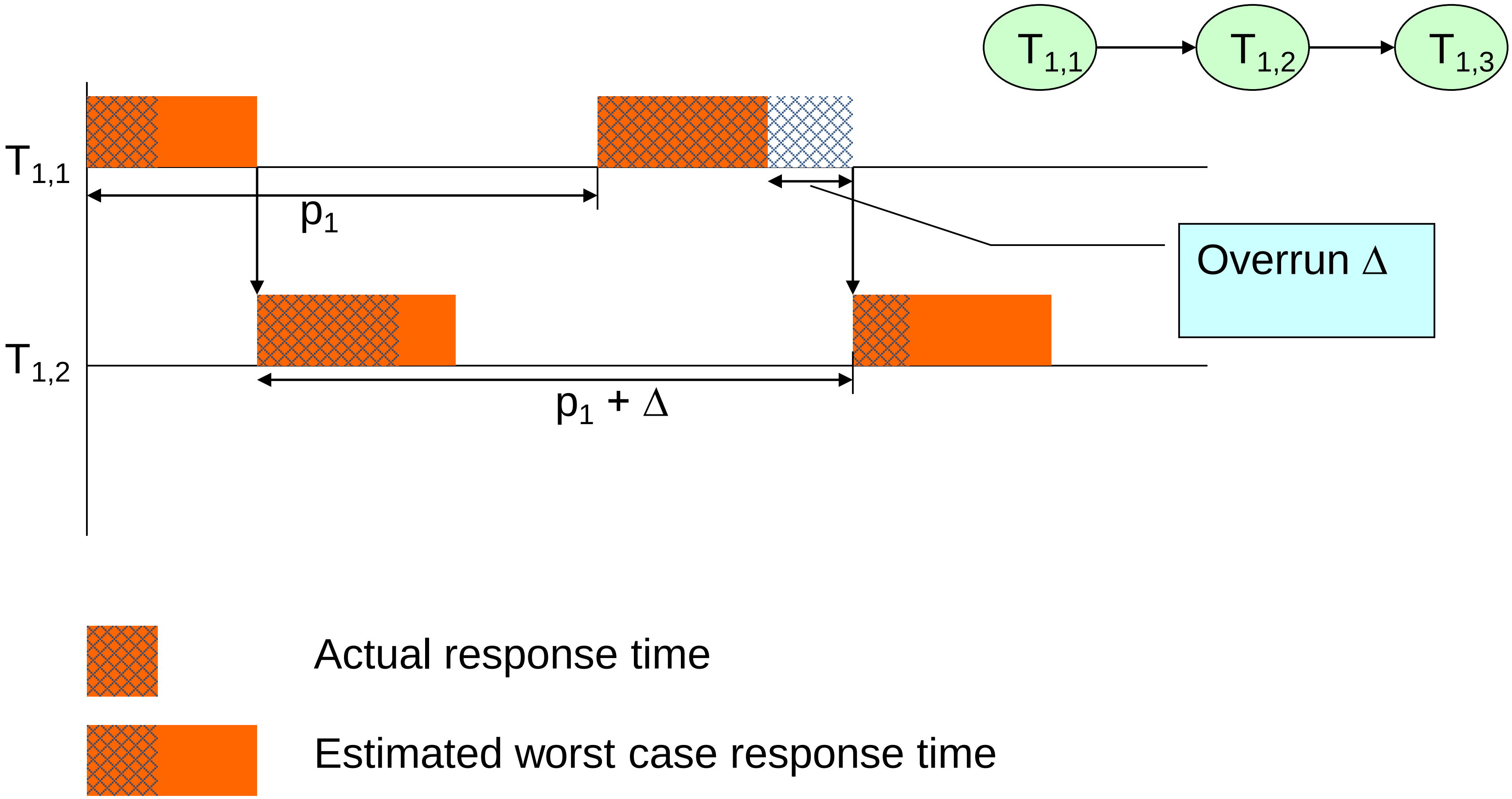
Phase Modification Protocol - Analysis

相變協議-分析

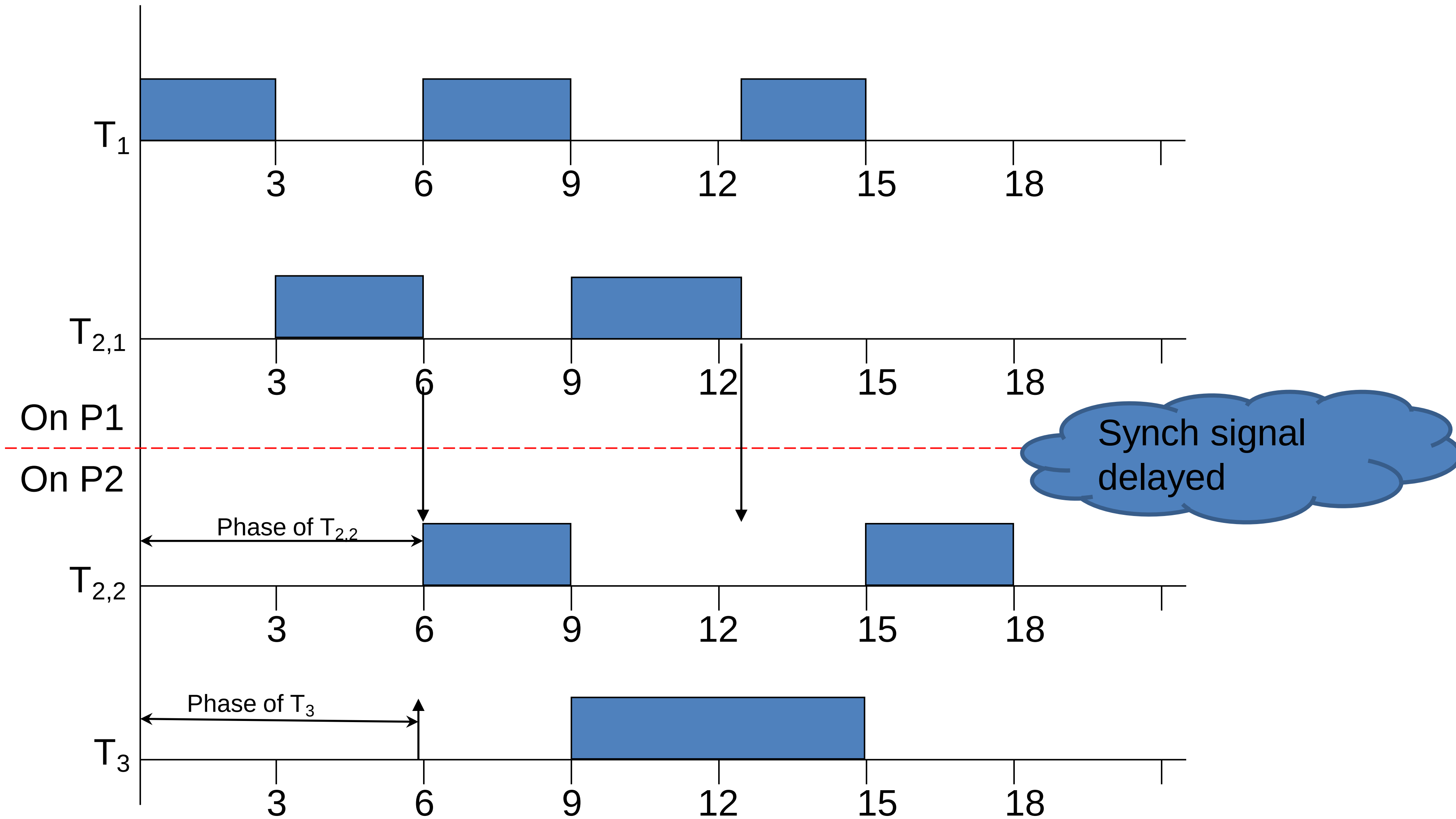
定期計時器中斷以釋放子任務

- Periodic timer interrupt to release subtasks
- Centralized clock or strict clock synchronization 集中式時鐘或嚴格的時鐘同步
- Task overruns could cause precedence constraint violations 任務超限可能會導致違反優先約束

Modified PM Protocol (1/2)



Modified PM Protocol (2/2)

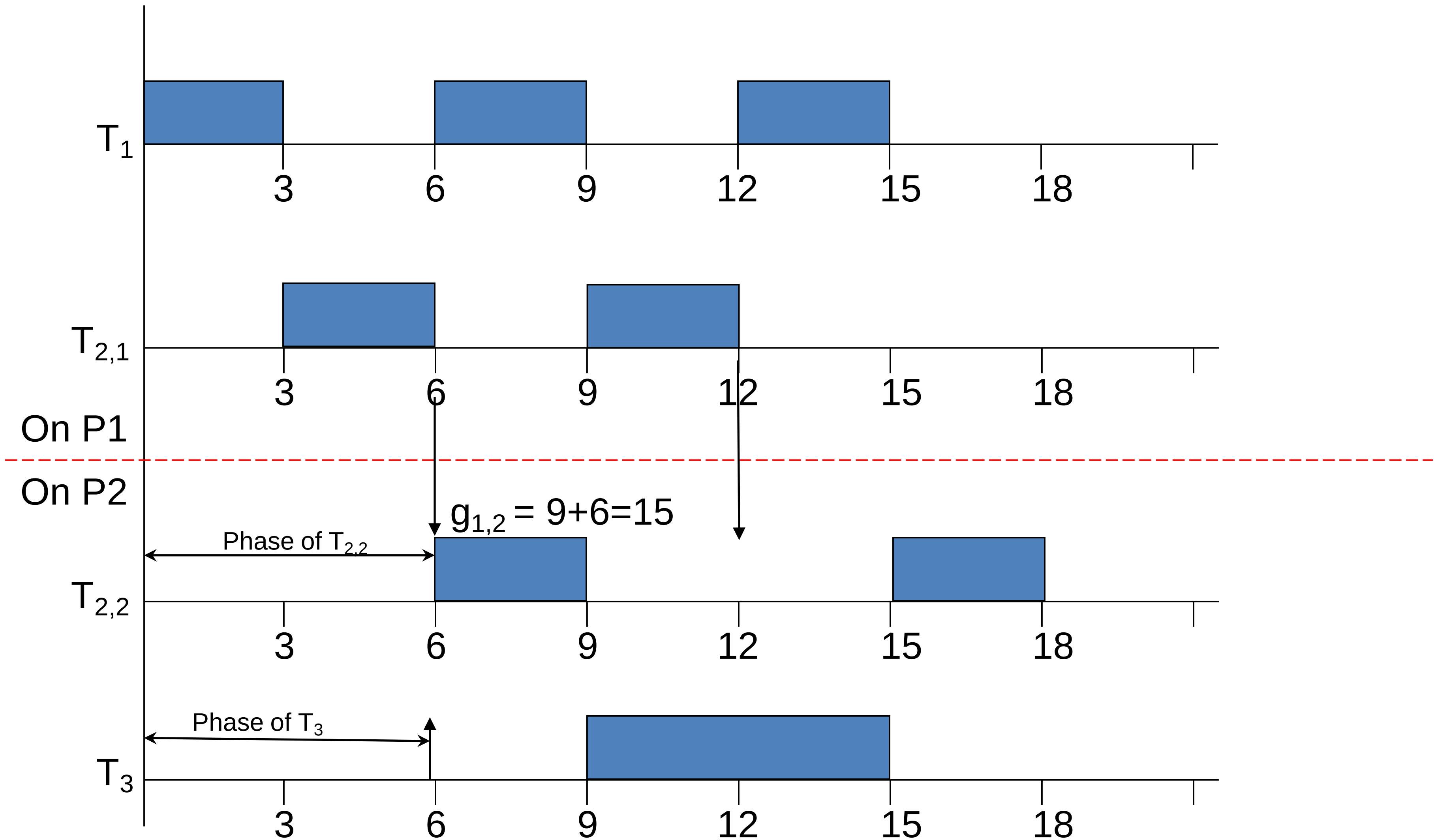


釋放防護協議

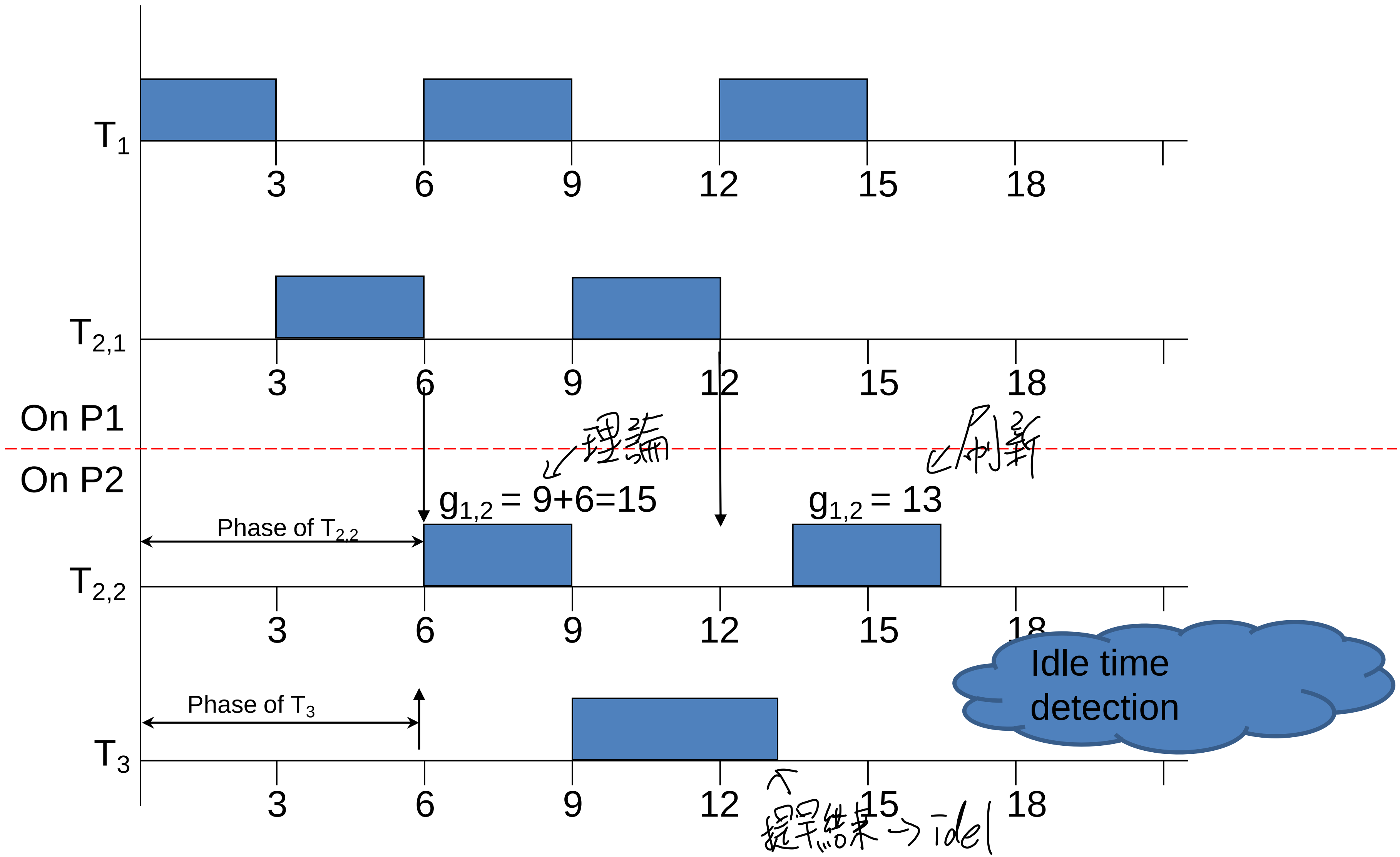
Release Guard Protocol

- A guard variable – *release guard* - associated with each subtask 保護變量–釋放與每個子任務相關的保護
- Release guard used to control release of each subtask 釋放防護用於控制每個子任務的釋放
  –包含子任務的下一個發佈時間
  - Contains next release time of subtask
- Synchronization signals as MPM 同步信號作為MPM
- Release guard updated 發布保護已更新
  - On getting synchronization signal –在獲得同步信號時
  - During idle time –空閒時間

Release Guard Protocol



Release Guard Protocol



釋放防護協議-分析

Release Guard Protocol - Analysis

- Shares the same advantages as MPM 與MPM具有相同的優勢

EER的上限仍與MPM相同

- Upper bound on EER still the same as MPM

–由於發佈時間的上限由發布保護者強制執行

  - Since upper bound on release time enforced by release guard

EER的下限小於MPM的下限

- Lower bound on EER less than that of MPM

–如果有空閒時間

  - If there are idle times

–降低平均EER（端到端響應時間）

  - Results in lower average EER (end-to-end response time)

Schedulability Analysis

An upper bound W_i to the end-to-end response time of any periodic task T_i in a fixed-priority system synchronized according to the MPM protocol or the RG protocol is given by 固定優先級系統中根據MPM協議或RG協議同步的任何週期性任務Ti的端到端響應時間的上限Wi

$$W_i = \sum_{k=1}^{n(i)} W_{i,k}$$

High pro 擋次數

同一顆 processer preement 時間

and

$$W_{i,k} = \frac{e_{i,k} + b_{i,k} + \sum_{\phi_{j,l} \le \phi_{i,k} \text{ and } \tau_{j,l} \in V_{i,k}} e_{j,l}}{1 - \sum_{\phi_{j,l} < \phi_{i,k} \text{ and } \tau_{j,l} \in V_{i,k}} u_{j,l}}$$

High pro Total U 都減掉

where $n(i)$ is the number of subtasks in T_i, $\phi_{i,k}$ is the priority of $\tau_{i,k}$, and the upper bound $W_{i,k}$ to the response time of every subtask $T_{i,k}$ is obtained by considering only subtasks on the same processor $V_{i,k}$, and by treating every such subtask $T_{j,l}$ as periodic task whose period is equal to the period p_j of the parent task T_j.

其中n（i）是Ti中子任務的數量，Oik是ti.k的優先級，通過僅考慮同一處理器Vih上的子任務並通過處理來獲得每個子任務Tik響應時間的上限Wik 每個這樣的子任務Tji，作為周期任務，其周期等於父任務Tj的周期pj。

通用計劃測試（GST）

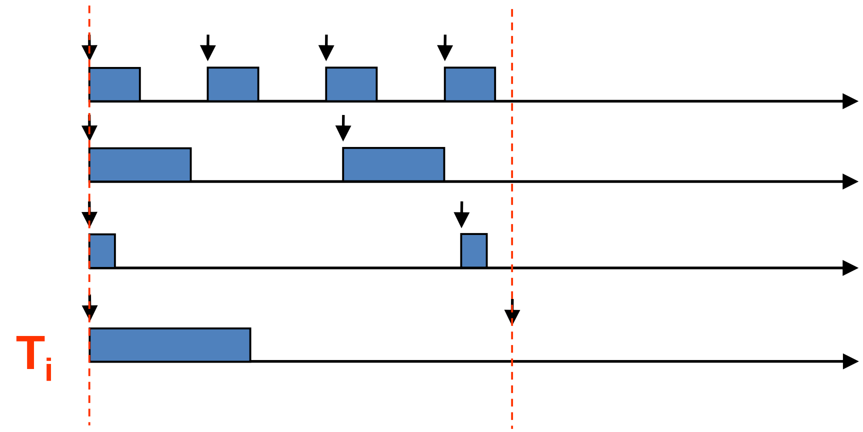
General Scheduling Test (GST)

響應時間分析

- Response time analysis
  - The response time of the job of Ti at critical instant can be calculated by the following recursive function –通過以下遞歸函數可以計算出關鍵時刻的Ti作業的響應時間

$$r_0 = \sum_{\forall i} c_i$$

$$r_n = \sum_{\forall i} c_i \left\lceil \frac{r_{n-1}}{p_i} \right\rceil$$



  - Observation: the sequence of r_x, x>=0 may or may not converge 觀察：rx的序列，x> = 0可能會收斂，也可能不會收斂

General Scheduling Test (GST)

- Example: T1=(2,5), T2=(2,7), T3=(3,8)
  - T1:
    - $R_0=2 \leq 5$ ok
  - T2:
    - $R_0=2+2=4 \leq 7$
    - $R_1=2*\lceil 4/5 \rceil+2*\lceil 4/7 \rceil=4 \leq 7$ ok
  - T3:
    - $R_0=2+2+3=7 \leq 8$
    - $R_1= 2*\lceil 7/5 \rceil+2*\lceil 7/7 \rceil+ 3*\lceil 7/8 \rceil =9> 8$ failed

  - Note: each task succeeds→the task set succeeds

Example

$P_1: T_{11}: \quad W_{11} = \frac{1+0+2}{1-0} = 3$

$W_{13} = \frac{2+1+1}{1-0} = 4$

$W_{21} = \frac{4+0+(1+2)}{1-(\frac{1}{15}+\frac{2}{15})} = 8.75$

$W_{11}: +2 \Rightarrow W_{13}$

$W_{13}: +1 \Rightarrow W_{11}$

∵ pro 一樣

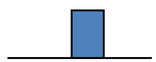
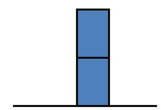
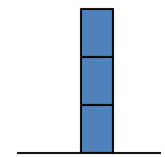
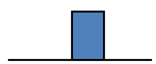
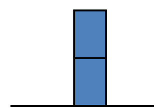
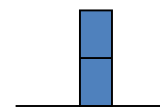
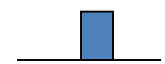
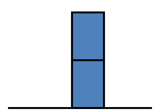
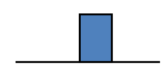
$P_2: T_{31} \Rightarrow W_{31} = \frac{e_{31}+b_{31}+0}{1-0} = \frac{1+0}{1} = 1$

$T_{12} \Rightarrow W_{12} = \frac{e_{12}+b_{12}+e_{31}}{1-\frac{e_{31}}{2}} = \frac{2+1+1}{1-\frac{1}{2}} = 8$

$T_{41} \Rightarrow W_{41} = \frac{e_{41}+b_{41}+(e_{31}+e_{12})}{1-(\frac{e_{31}}{2}+\frac{e_{12}}{15})} = \frac{5+0+(1+2)}{1-(\frac{1}{2}+\frac{2}{15})} = 21.8$

$T_{i,k}$	$V_{i,k}$	p_i	$e_{i,k}$	$UD_{i,k}$	$b_{i,k}$	$W_{i,k}$	$W_{i,k}(GST)$
$T_{1,1}$	P_1	15	1	15	0	3	3
$T_{1,3}$	P_1	15	2	15	1	4	3(4)
$T_{2,1}$	P_1	20	4	20	0	8.75	7
$T_{3,1}$	P_2	2	1	2	0	1	1
$T_{1,2}$	P_2	15	2	15	1	8	4(6)
$T_{4,1}$	P_2	20	5	20	0	21.8	14

Comparison of Protocols

	DS	PM	MPM	RG
Implementation complexity	Synch interrupts	Timer interrupts clock synchronization	Synch & timer interrupts	Synch & timer interrupts
Run-time overhead				
Average EER				
Estimated worst case EER				
Inherently missed deadlines	Yes	No		

Reference

- Real-time Systems, Jane Liu
- Bettati,R.,``End-to-end scheduling to meet deadlines in distributed systems," Ph.D. thesis, University of Illinois at Urbana-Champaign
- Sun, J., `` Fixed-Priority Scheduling of Periodic Tasks With End-to-End Deadlines," Ph.D. thesis, University of Illinois at Urbana-Champaign

4/21