Ordonnancement dans les STR

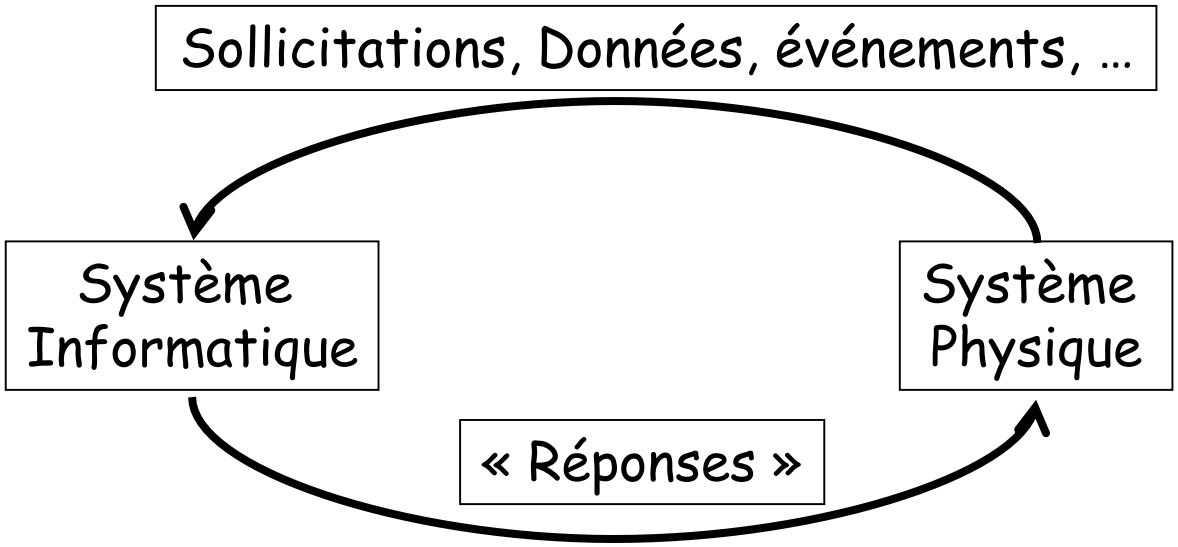
Borislav Vidolov

UMR CNRS 7253 HeuDiaSyC
Département Génie Informatique

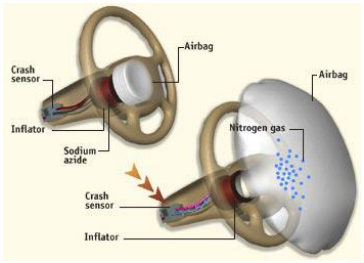
bvidolov@utc.fr

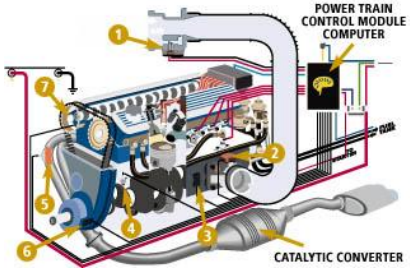
Systèmes informatiques en interaction avec des systèmes physiques réels. (Interactions, pilotages, E/S, …)

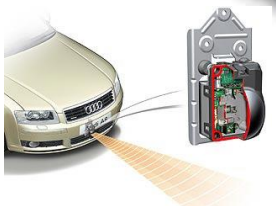
La dynamique est imposée par le « monde réel ».



Système Temps Réel Embarque



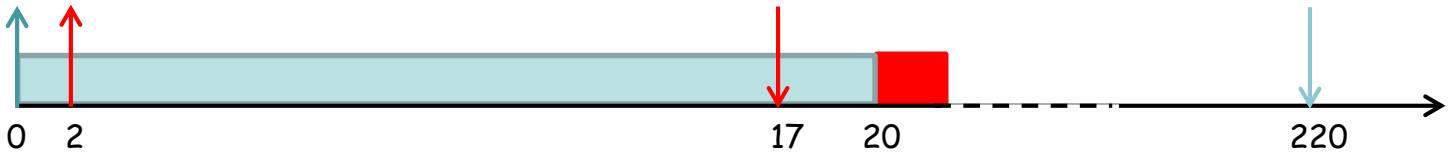




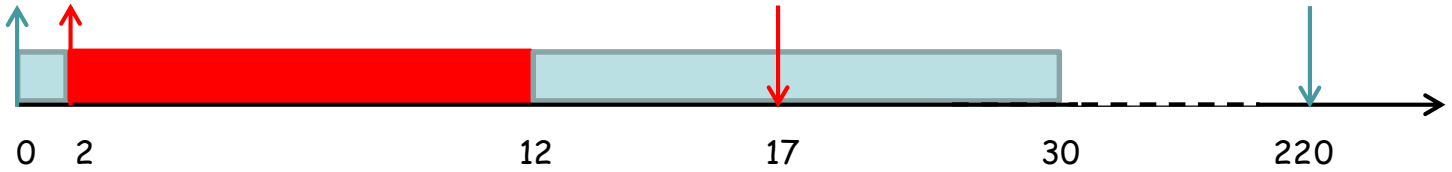
Système Temps Réel Embarque

Le lièvre et la tortue

- T1 (r = 0, C = 20, D = 220)
- T2 (r = 2, C = 10, D = 15)



FIFO – T2 ne respect pas son échéance



EDF – les échéances sont respectées

Types de tâches

- **Périodiques** : elles ont des échéances strictes
- **Apériodiques** (sporadiques) : le début de l'exécution de la tâche est irrégulier (asynchrone)
- **Souple** : échéance relative (mieux tard que jamais)
- **Non temps réel** : aucune contrainte de temps

Caractéristique des tâches

Les tâches sont exécutées et terminées en respectant un certain nombre de contraintes :

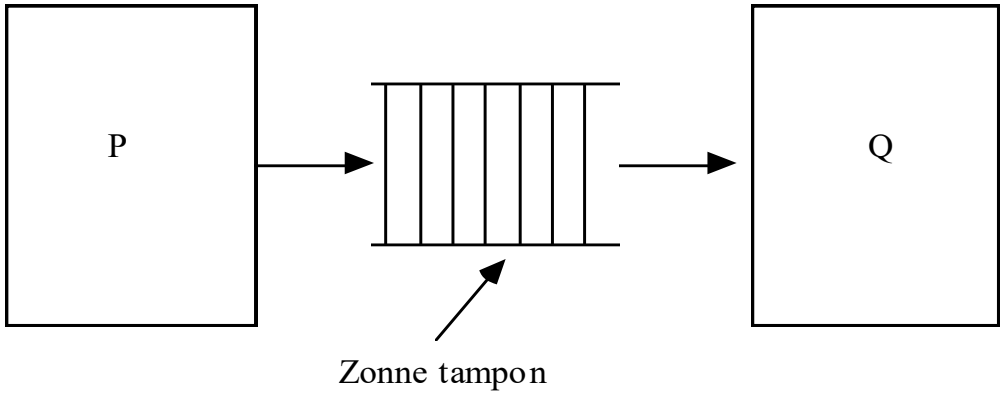
- des contraintes temporelles
- des priorités
- des interruption des tâches
- de la terminaison des tâches
- des relations entre les tâches
- …

Ressources du processus

• Ressources : procédures, données, mémoire, processeur, fichiers, périphériques

• Ressources communes (partagées entre les différents processus) => L'évolution des processus sera donc dépendant de la disponibilité de la ressource:

- **bloqué**
- **actif**



Types de ressources :

- **ressource locale à un processus** : elle lui appartient
- **ressource commune** : partageable à n points d'accès,

  critique – 1 point d'accès

En fonction des ressources, les processus sont :

- **indépendants**
- **parallèles**
- **en exclusion mutuelle**

Contraintes temporelles

- aucune contrainte temporelle
- contraintes de temps essentiellement de trois types :

  - instant au plus tôt pour commencer une tâche
  - instant au plus tard pour la terminer
  - durée maximale d'exécution de la tâche.

Activation des tâches:

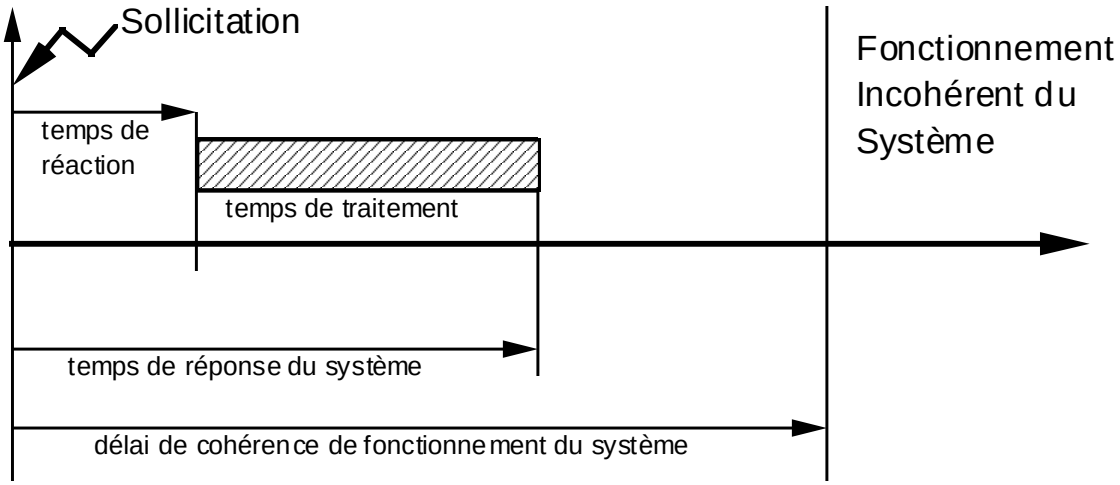
  - périodiquement (périodique)
  - à des instants fixes
  - de manière aléatoire(apériodique).

Interruption des tâches

+ Plus de liberté à l'ordonnanceur

- Charge d'exécution induite par les changements de contextes

- Ressources en exclusion mutuelle, la préemption peut conduire à des situations d'interblocage.



Relations entre tâches

- les tâches sont indépendantes - sans relation
- les tâches s'exécutent selon un ordre fixé - des contraintes de précédence
- les tâches partagent une (ou plusieurs) ressource(s) - contraintes de ressources

Des algorithmes d'ordonnancement qui tiennent compte de :

Relations de précédence
Contraintes de ressources

Priorités des tâches

- Critères pour l'allocation du processeur

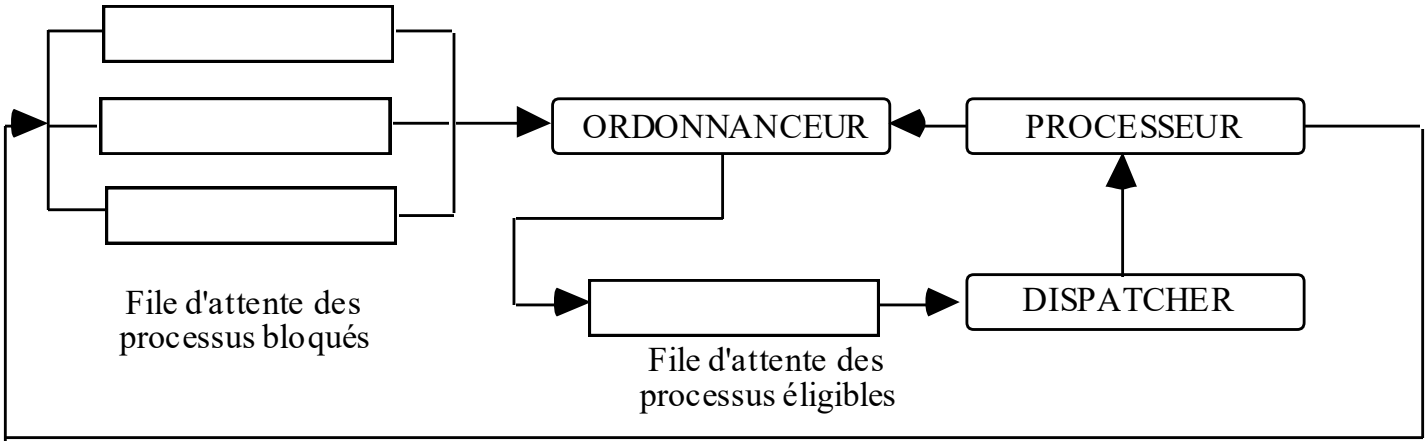
- Priorités affectées aux tâches :
  - à la conception (priorités externes)
  - à partir d'un calcul fait par l'algorithme d'ordonnancement (priorités internes).

Priorités statiques
Priorités dynamiques

Algorithmes d'ordonnancement

- assurer la gestion des commutations de tâches de l'état bloqué à l'état éveillé

- effectuer le choix d'une tâche dans l'ensemble des tâches éligibles.



Classification

- Exécution
  - hors ligne
  - en ligne
  - combinaison des deux : tâches périodiques – hors ligne
    tâches apériodiques – en ligne
- A contraintes strictes ou souples
- Statiques ou dynamiques
- Préemptifs ou non préemptifs
- Ordonnancement local – site isolé
- Ordonnancement global – ensemble de sites constituants un système repartie
- Garantir ou non le respect de toutes les contraintes
- Optimal ou non
- Efficace - solution dans un temps borné par une fonction polynomiale

Critères de qualité d'un algorithme d'ordonnancement

- **efficacité** : le processeur doit consacrer le maximum temps à l'application

- **temps de réponse** :
  doit être faible de façon à réagir rapidement à un événement externe

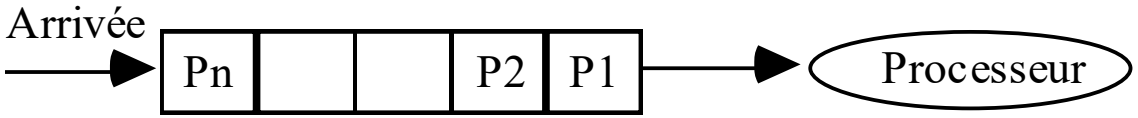
- **impartialité** : l'ordonnanceur doit effectuer un partage équitable du processeur

- **débit** : l'ordonnanceur doit effectuer le plus de tâches possibles dans un intervalle de temps donné

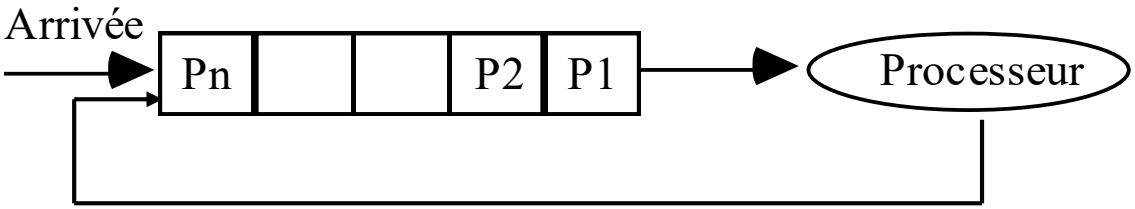
- **déterminisme et prédictibilité**

Sans priorités

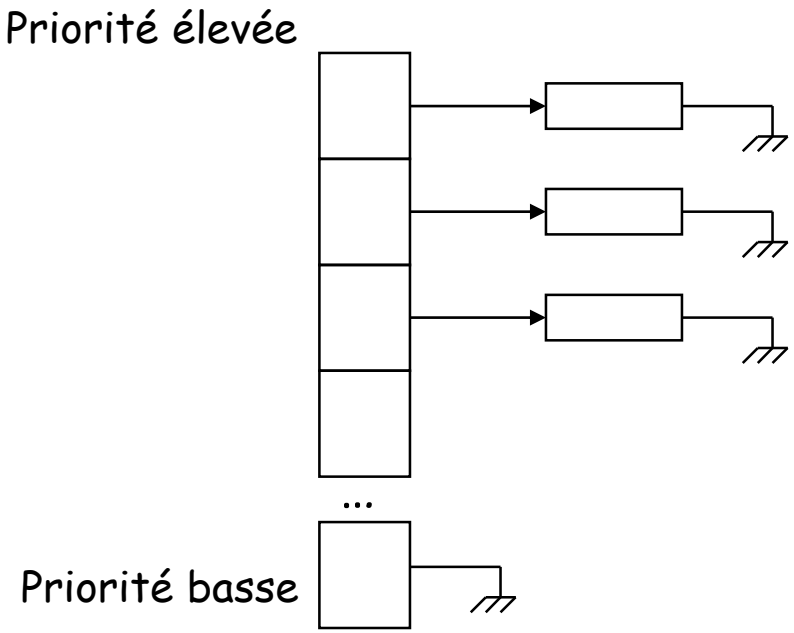
- Premier arrivé-premier servi



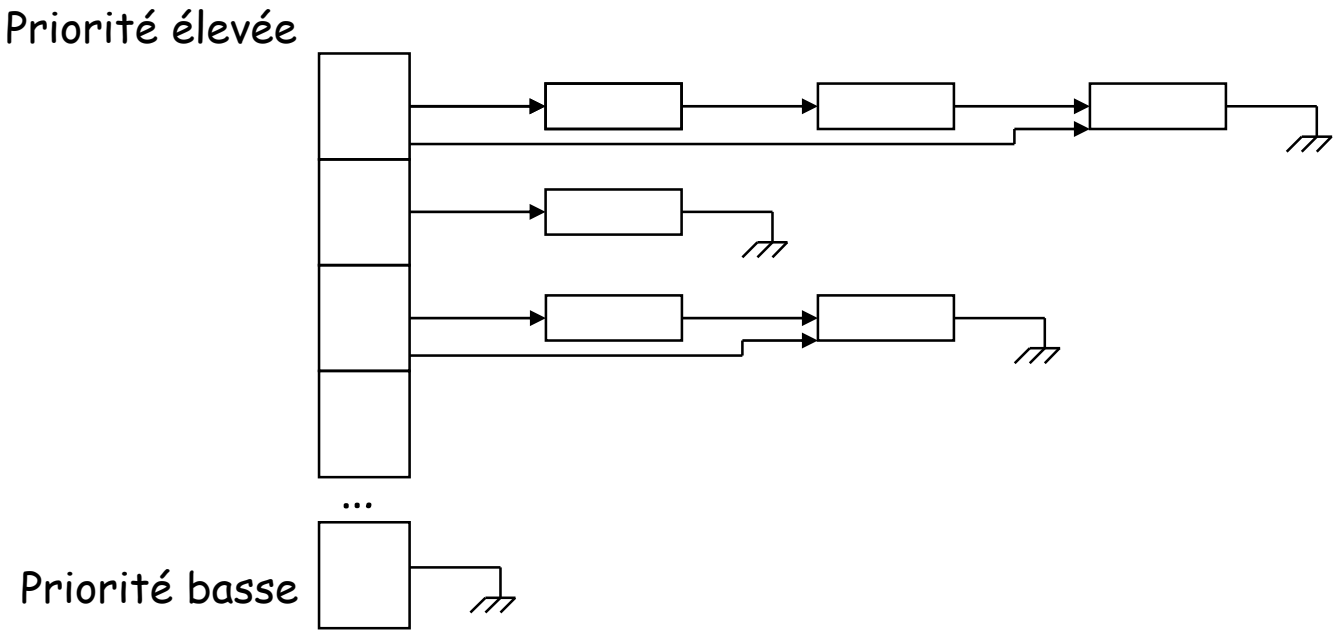
- Tourniquet ou round-robin



Ordonnanceur à priorité



Ordonnanceur préemptif à priorité



Ordonnanceur préemptif à priorité avec vieillissement

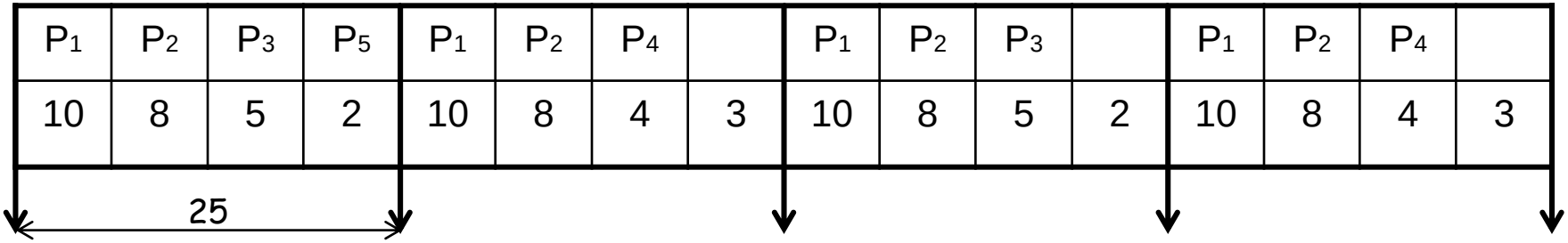
T3(64) T2(127) T1(128)	exécution de T1
T3(65) T2(128) **T1(128)**	exécution de T1
T3(66) **T1(128)** T2(129)	exécution de T2
T3(67) **T2(127)** T1(128)	exécution de T1
T3(68) T2(128) **T1(128)**	exécution de T1
T3(69) **T1(128)** T2(129)	exécution de T2

Ordonnancement cyclique

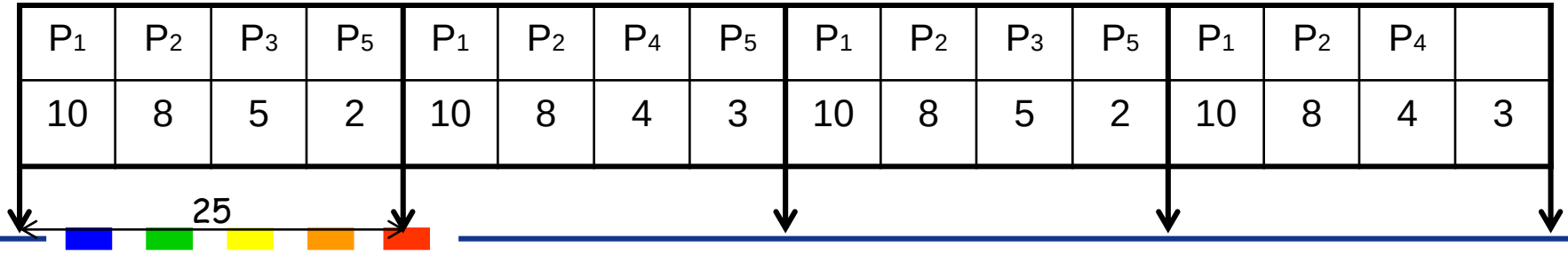
- découpage de l'application en processus
- définition de séquences élémentaires jamais interrompues
- construction d'une table des processus à activer cycliquement
- activation d'une séquence à chaque tic d'horloge
- cycle mineur - <= T_i
- cycle majeur - PPCM (T_i)

Exemple :

	C_i	T_i
P_1	10	25
P_2	8	25
P_3	5	50
P_4	4	50
P_5	2	100



Modification de la capacité de traitement de P_5 :
$C_5=7 \Rightarrow C_{51} = 2;\ C_{52} = 3;\ C_{53} = 2;$



Avantages –

- maîtrise du temps processeur
- pas de mécanismes de synchronisation et d'exclusion mutuelle
- preuve « hors ligne » du fonctionnement

Inconvénients –

- manque de flexibilité
- prise en compte de tâches apériodiques
- modification et évolution de l'application
- complexité algorithmique
- politique d'ordonnancement basé sur le partage du temps
- affectation de priorités et garanties du respect des contraintes temporaires

Application –

systèmes critiques (aéronautique, nucléaire, défense)

Stratégies d'ordonnancement de tâches indépendantes

Ordonnanceurs TR classiques :

- le respect des contraintes TR des processus est laissé à la responsabilité du concepteur de l'application
- la prise en compte d'un nouveau processus doit avoir été prévue au niveau des spécifications
- des algorithmes d'ordonnancement peu adaptatifs

Stratégies d'ordonnancement de tâches indépendantes

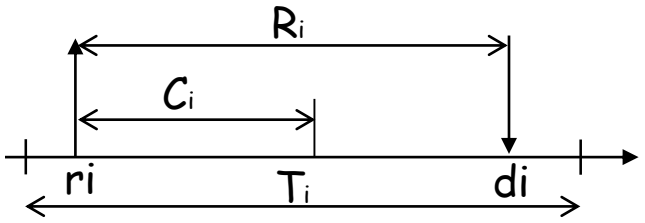
Ordonnanceur temps réel par échéance :

- calculent eux-mêmes les priorités des processus
- meilleur rendement des processeurs
- des algorithmes d'ordonnancement moins spécifiques

Hypothèses : des processus indépendants, pas de contraintes de précédente ni contraintes de ressources, tâches périodiques prêtes au début de chaque période, préemptées, on ignore le temps de chargement des vecteurs d'état, les tests d'ordonnancement se font hors-ligne.

Caractéristiques d'un processus τ_i périodique

τ_i : (r_i, C_i, R_i, T_i)



Ti : période de τ_i

ri : première date de disponibilité de τ_i.

date de début de la k^{ème} période : $r_i^{(k)} = r_i + (k-1)T_i ; \forall k \geq 0$

Ci : durée d'exécution sans préemption de τ_i sur un processeur

Ri : délai critique de τ_i

0 < Ci < Ri < Ti

Pi : priorité du processus

ui : facteur d'utilisation de τ_i : u_i = Ci/Ti < 1

di : prochaine date d'échéance du processus : $d_i = r_i^{(k)} + R_i$

Ri(t) : temps de réponse dynamique : Ri (t) = di - t

Ci(t) : durée d'exécution dynamique : Ci (t) = Ci - t

Li : laxité maximale : Li = di -Ci

Li(t) : laxité dynamique : Li (t) = di -Ci (t)

Rate Monotonic

- algorithme préemptif à priorité statique
- le processus le plus prioritaire est celui disponible avec la plus petite période
- les instants de prise en compte des priorités sont les dates de disponibilités et les dates de terminaisons des processus.

Théorème : Pour n processus périodiques à échéance sur requête (Ti = Ri), une condition suffisante d'acceptation d'une configuration par l'algorithme Rate Monotonic est :

$$U \leq n\left(2^{1/n} - 1\right)$$

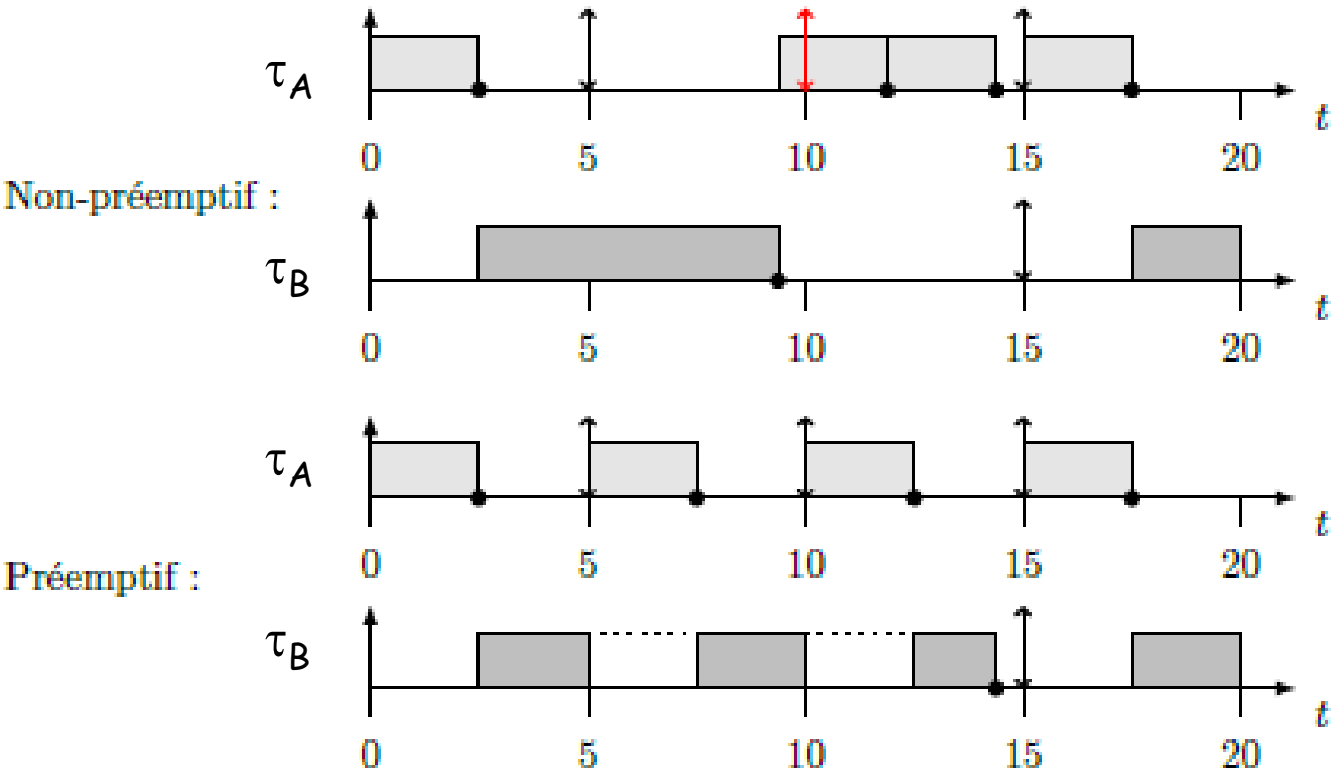
$U = \sum_{i=1}^{n} u_i = \sum_{i=1}^{n} C_i / T_i$ facteur d'utilisation d'une configuration.

Exemple : Trois processus à échéance sur requête
(ri, Ci, Ti) : A(0, 3, 20), B(0, 2, 5), C(0, 2, 10).

$U \leq n\left(2^{1/n} - 1\right)$ => $0.75 < 0.78$

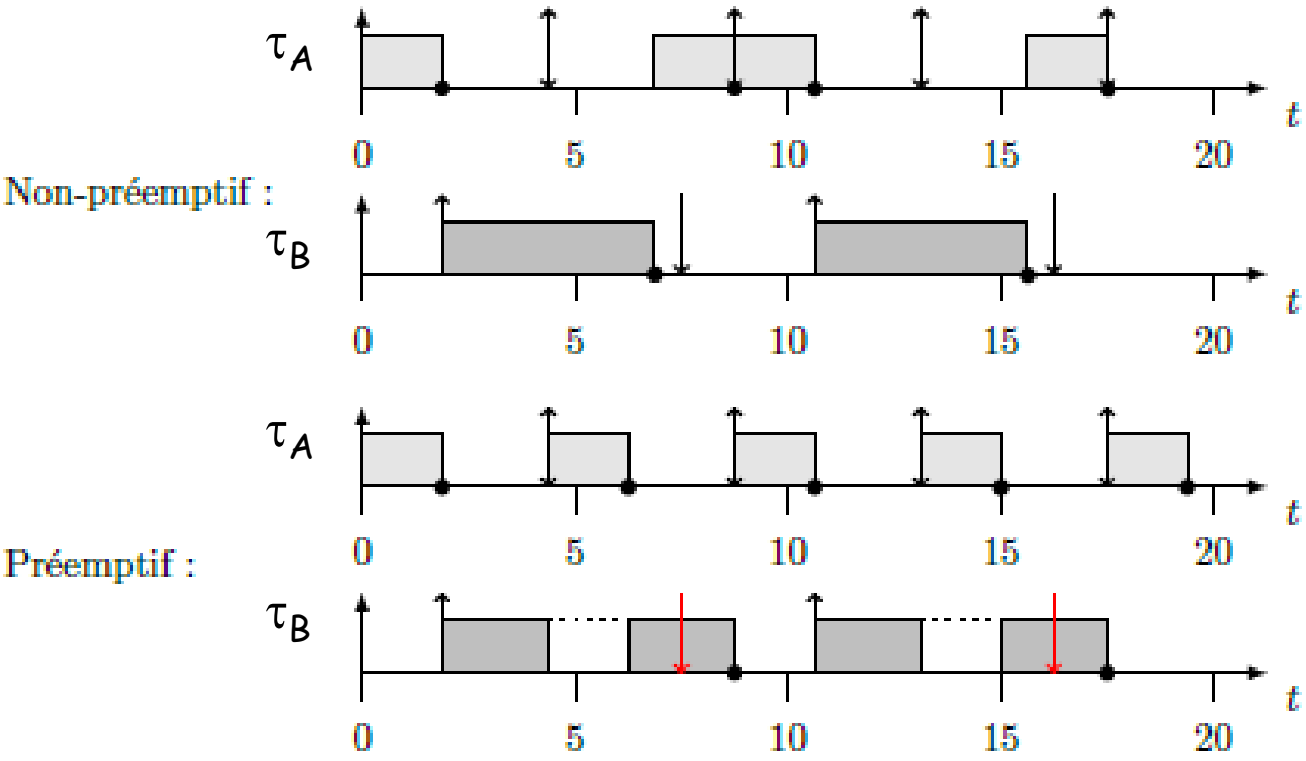
Préemptif ou Non-préemptif

Exemple : Deux processus à échéance sur requête
(ri, Ci, Ti) : A(0, 2.5, 5), B(0, 6.8, 15)



Préemptif ou Non-préemptif

Exemple : (ri, Ci, Ri, Ti) : A(0, 1.8, 4.3, 4.3), B(0, 5, 7.4, 8.6)



A suivre …