Chapitre IV : Ordonnancement des processus

Cours Système Exploitation II

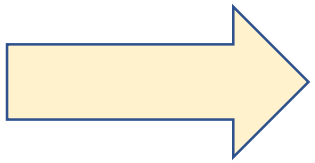
1 Linfo

ISIMM

2022/2023

Motivations

Lorsqu'un ordinateur est multiprogrammé, il possède fréquemment plusieurs processus/threads en concurrence pour l'obtention de temps processeur.

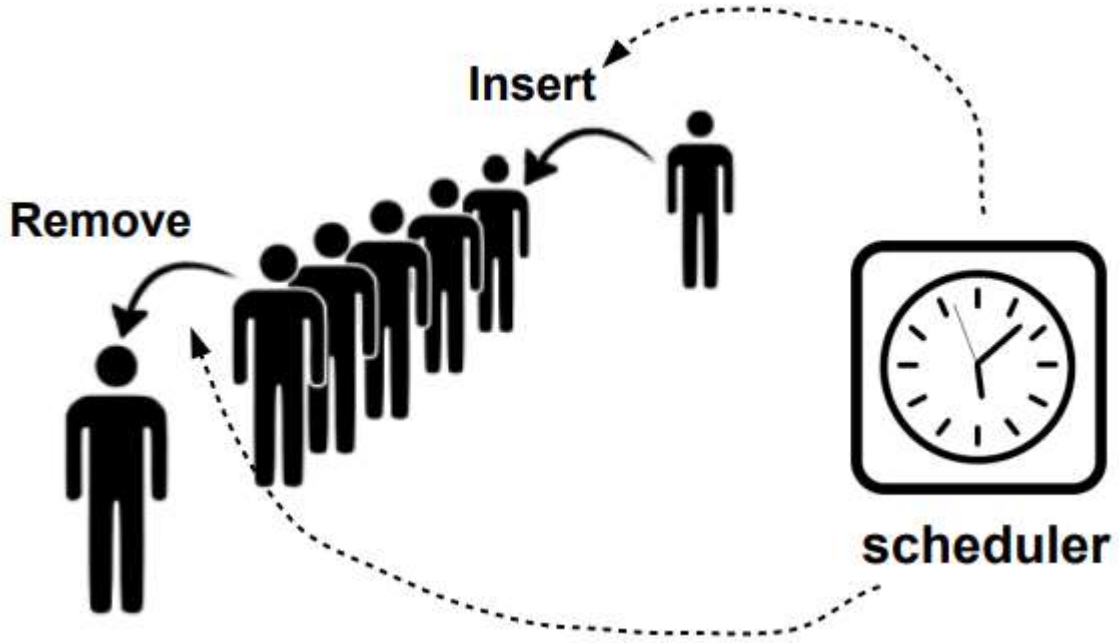


→ S'il n'y a qu'un seul processeur, un choix doit être fait quant au prochain processus à exécuter

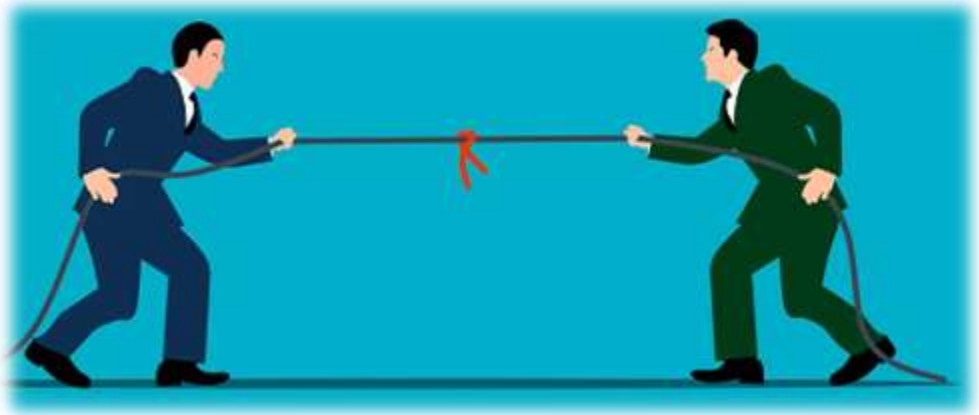
Définitions

La partie du système d'exploitation qui effectue ce choix se nomme l'ordonnanceur (scheduler) et l'algorithme qu'il emploie s'appel algorithme d'ordonnancement (scheduling algorithm).

Outre le fait de sélectionner le bon processus à exécuter, l'ordonnancement doit également se soucier de faire un usage efficace du processeur, car le passage d'un processus à l'autre sont coûteux en termes de temps de traitement



Défis

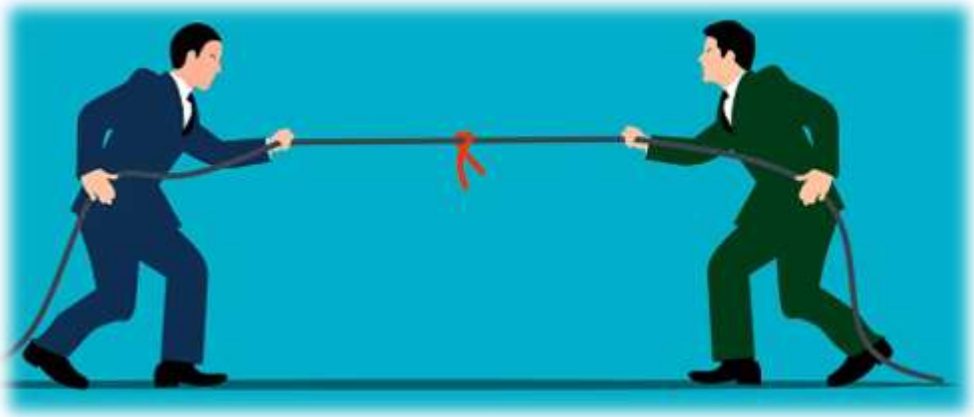


Temps de changement de contexte

Temps d’exécution

Efficacité : Utiliser au mieux le processeur

Objectifs de l'ordonnanceur



Les objectifs d'un ordonnanceur d'un système multi-utilisateur sont entre autres :

– S'assurer que chaque processus en attente d'exécution reçoive sa part de temps processeur.

– Minimiser le temps de réponse.

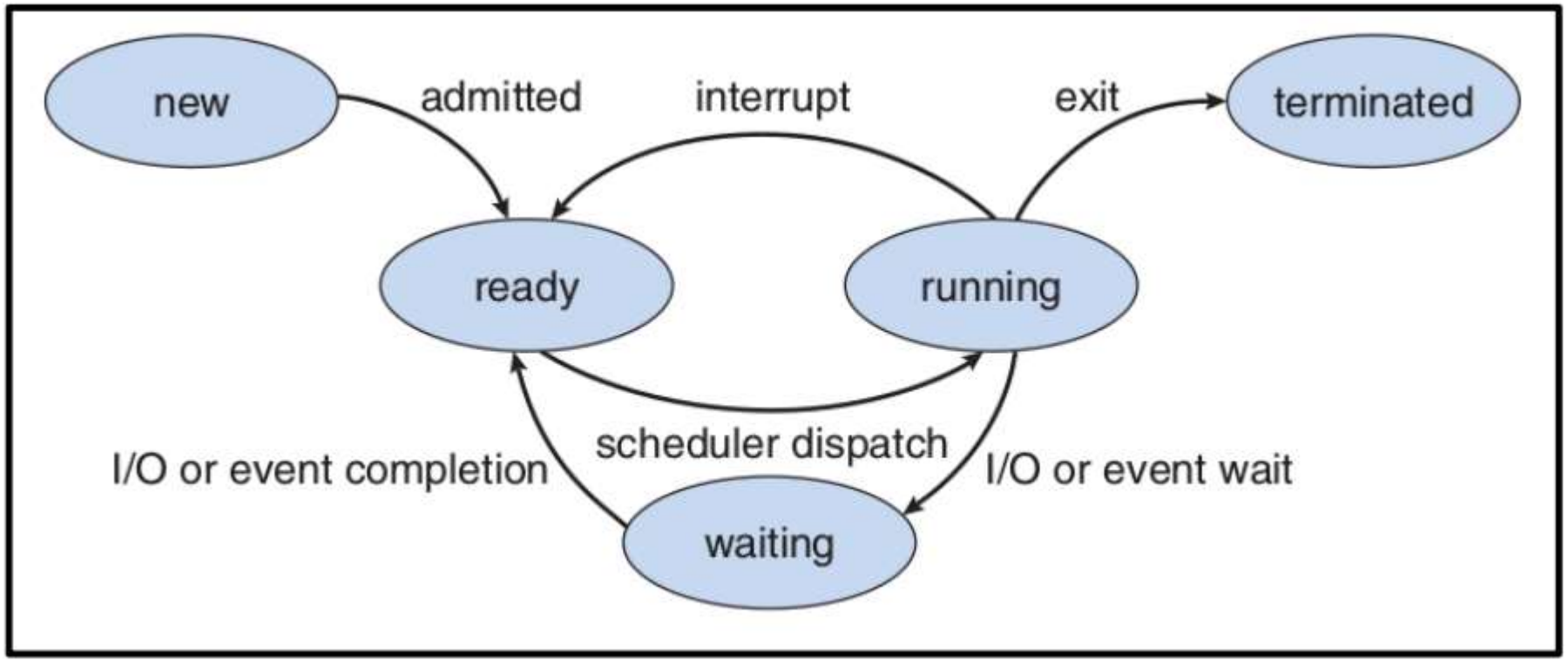
– Utiliser le processeur à 100%.

– Prendre en compte des priorités.

– Être prédictible.

États de processus et ordonnancement

Quand ordonnancer ?



États de processus et ordonnancement

Quand ordonnancer ?

- Lorsqu'un nouveau processus est créé :

→ il faut se décider s'il faut exécuter d'abord le processus parent ou le processus enfant.

- Lorsqu'un processus se termine

→ un autre processus doit être choisi parmi les processus prêts

États de processus et ordonnancement

Quand ordonnancer ?

- Lorsqu'un processus se bloque

→ un autre processus doit être sélectionné pour être exécuter

- Lorsqu'une interruption d'E/S se produit

→ il faut prendre une décision d'ordonnancement parmi les processus qui étaient bloqué en attente d'E/S

Qu’est ce qu’une interruption ?

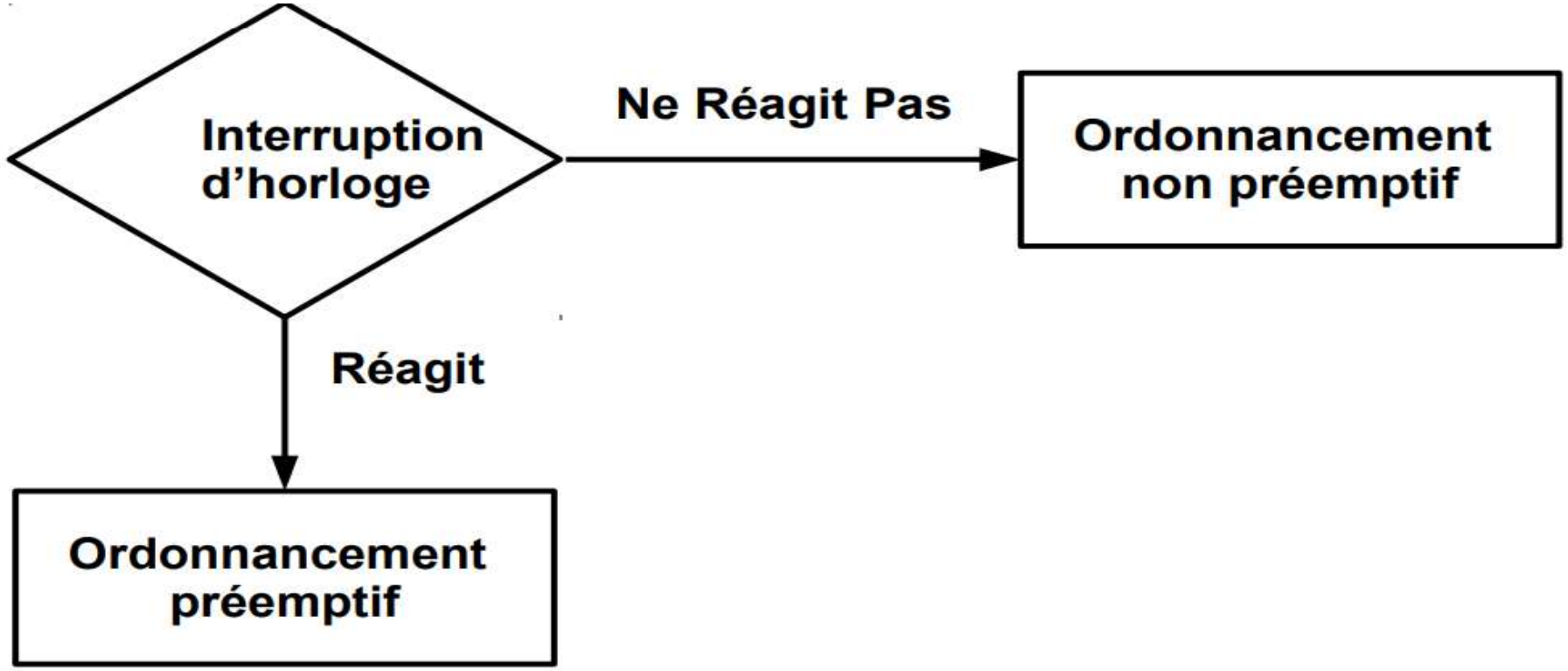
- **Interruption**
  - arrêt temporaire de l’exécution
- **Catégorie** : **Matérielle** vs. **Logicielle**
- **Interruption Logicielle : Trappe ou déroutement**
  - Si le CPU détecte une erreur dans le traitement d’une instruction (division par zéro par exemple)
  - Arrêt de l’exécution pour exécuter une routine particulière ISR (*Interrupt Service Routine*) par type d’erreur rencontrer (débordement de mémoire, division par zéro,...)

Interruption Matérielle IRQ : Interrupt Request

- **Générées par les périphériques (clavier, disque, USB,…)**
- **Peuvent être masquées (interdites ou autorisés)**
- **Interruption : le périphérique signale au CPU les événements par interruption**
  - Éviter au CPU **d'attendre un délai supplémentaire (boucler)** pour que les données soient émises par le périphérique (technique de polling ou questionnement)
- **Le CPU arrête momentanément l'exécution d'un processus pour exécuter la requête du périphérique.**
  - Déclenche l'ordonnancement entre les processus bloqué en l'attente de l'E/S en question

Classification des algorithmes d'ordonnancement

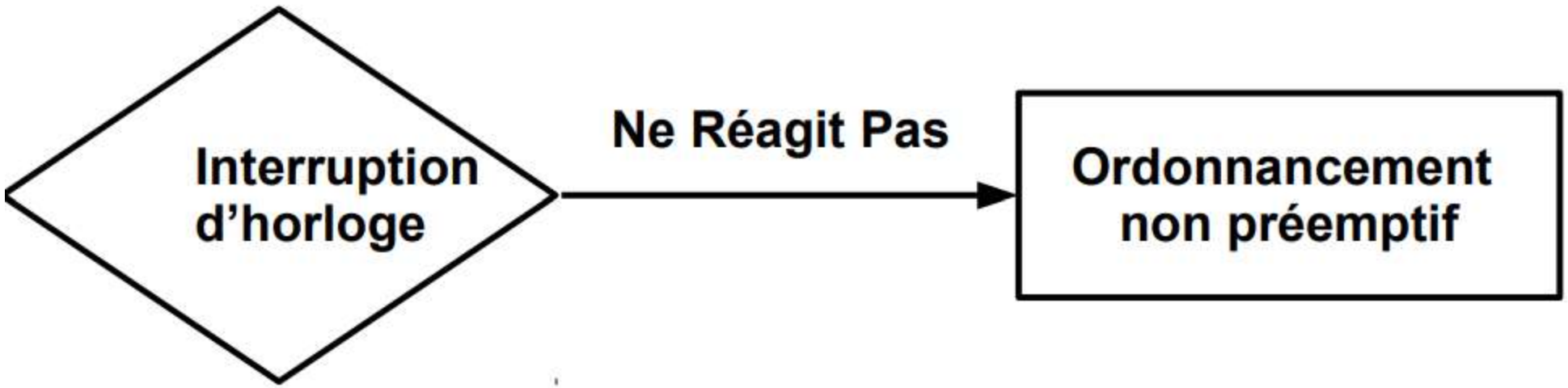
On utilise les interruptions d’horloge pour commuter les tâches dans les systèmes multitâches.



Algorithmes d'ordonnancement non préemptif

- **Sélectionne un processus, puis le laisse s'exécuter jusqu'à ce qu'il se bloque, ou qu'il libère volontairement le processeur**
  - Même s'il s'exécute pendant des heures, il ne sera pas suspendu de force
  - Aucune décision d'ordonnancement n'intervient pendant les interruptions d'horloge

Ordonnanceurs non préemptifs



Ordonnanceurs non préemptifs

- **Dans un système à ordonnancement non préemtif ou sans réquisition, le système d'exploitation choisit le prochain processus à exécuter:**
  - le Premier Arrivé est le Premier Servi PAPS (ou First-Come First-Served FCFS)
  - ou le plus court d'abord (Shortest Job First SJF)

- **Il lui alloue le processeur jusqu'à ce qu'il se termine ou qu'il se bloque (en attente d'un événement).**

  → Il n'y a pas de réquisition

Critères spécifiques d’ordonnancement

Débit = Nombre de processus terminés / Temps total écoulé

Temps d’exécution (Burst Time) : Le temps nécessaire au processeur ou à un périphérique d'E/S pour être occupé par le processus.

Temps d’arrive (Arrival Time): Le moment où un processus apparaît pour la première fois dans la file d'attente des processus prêts.

Critères spécifiques d’ordonnancement:

- **Temps de rotation ou temps de séjour (Turn Around Time (TAT))**:

TAT = temps de terminaison - temps d'arrivée, où le temps de terminaison est le moment où le processus se termine.

- **Temps d’attente (Waiting Time (WAT)):**

WAT = temps de début d'exécution - temps d'arrivée, où le temps de début d'exécution est le moment où le processus commence à être traité.

WAT = temps de séjour – temps d’exécution

- **Temps d'attente moyen** (**Average Waiting Time):** moyenne de tous les temps d'attente des processus.

First-Come, First-Served

- Implémenter via une file d'attente FIFO → **code très facile à implémenter**
- Par contre, c'est une stratégie qui peut engendrer des **temps d'attentes moyens importants et très variables**
- Exemple : considérons les processus P1, P2, P3 avec les temps d'exécutions suivants. Ces processus arrivent tous au temps 0.

First-Come, First-Served

P1	24
P2	3
P3	3

First-Come, First-Served

- Si les processus arrivent dans l'ordre P1, P2, P3, on aura le diagramme de Gantt suivant

- Le temps d'attente de P1 est de 0ms, celui de P2 est de 24ms et enfin celui de P3 est 27ms

- Le temps d'attente moyen est de (0 +24 +27)/3 = 17ms

First-Come, First-Served

- Utilisation UCT = 100%
- Débit = 3/30 = 0,1
  - 3 processus complétés en 30 unités de temps
- Temps de rotation moyen: (24+27+30)/3 = 27

First-Come, First-Served

- Supposons que les processus arrivent dans l'odre P2, P3, P1, on aura la diagramme de Gantt suivant :

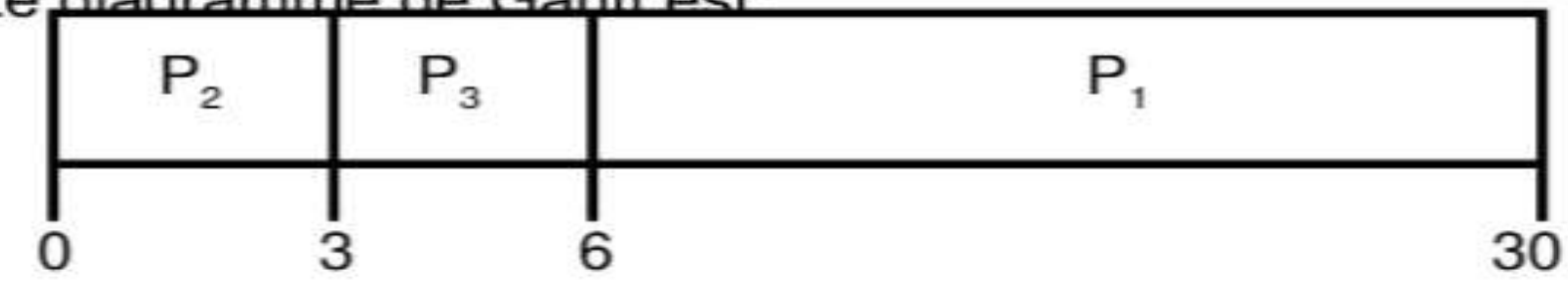
- Le temps moyen d'attente devient (0+3+6)/3 = 3ms

First-Come, First-Served

Si les mêmes processus arrivent à 0 mais dans l'ordre

P_2 , P_3 , P_1 .

Le diagramme de Gantt est:



- Temps d'attente pour $P_1 = 6$ $P_2 = 0$ $P_3 = 3$
- Temps moyen d'attente: (6 + 0 + 3)/3 = 3
- Temps de rotation moyen: (3+6+30)/3 = 13
- Beaucoup mieux!
- Donc pour cette technique, les temps peuvent varier grandement par rapport à l'ordre d'arrivée de différent processus

Shortest Job First

- **L'ordonnanceur choisit, parmi le lot de processus à exécuter, le plus court (plus petit temps d'exécution).**

- **Stratégie offre le temps moyen d'attente minimale**

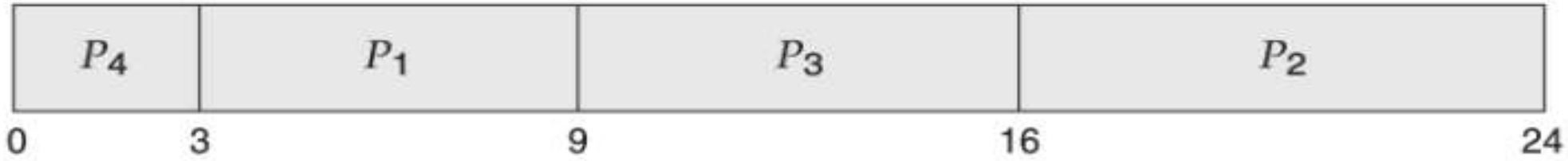
Shortest Job First

- Exemple les processus P1, P2, P3 et P4 arrivent tous au temps 0 et ont les temps d'exécution suivant

P1	**6**
P2	**8**
P3	**7**
P4	**3**

Shortest Job First

- Avec SJF, on aura le diagramme de Gantt suivant :



- P1 attends 3ms, P3 9ms, P2 16ms et P4 0ms. Le temps moyen d'attente est (3+16+9+0)/4 = 7ms

Temps de rotation = (3+9+16+24)/4 = 13 ms

Ordonnanceurs non préemptifs : Exercice I

Considérons cinq processus A, B, C, D et E dont les temps d'exécution et leurs temps d'arrivée respectifs sont les suivants:

Processus	Temps d'exécution	Temps d'arrivée
A	3	0
B	6	1
C	4	4
D	2	6
E	1	7

Ordonnanceurs non préemptifs : Exercice I

Faire un schéma qui illustre l'exécution (diagramme de Gantt) et calculer le temps de rotation de chaque processus, le temps moyen de rotation, le temps d'attente et le temps moyen d'attente et le débit en utilisant :

– Premier arrivé premier servi (FCFS)

– Le plus court d'abord (SJF)

Ordonnanceurs non préemptifs : Correction Exercice I : FCFS

- **A t= 0, seulement le processus A est dans le système et il s'exécute.**
- **A t=1 le processus B arrive mais il doit attendre que le processus A termine son exécution car il a encore 2 unités de temps.**
- **Ensuite B s'exécute pendant 4 unités de temps.**
- **A t= 4, 6, et 7 les processus C, D et E arrivent mais B a encore 2 unités de temps.**
- **Une fois que B a terminé, C, D et E entrent au système dans l'ordre**

Ordonnanceurs non préemptifs : Correction Exercice I : FCFS

- **Temps de séjour = temps de terminaison – temps d'entrée**

Processus	Temps de terminaison	Temps d'arrivée	Temps de séjour
A	3	0	3
B	9	1	8
C	13	4	9
D	15	6	9
E	16	7	9

- **Le temps moyen de séjour est**
  - (3+8+9+9+9)/5=7.6

Ordonnanceurs non préemptifs : Correction Exercice I : FCFS

- **Temps d'attente = temps de séjours – temps d'exécution**

Processus	Temps de séjour	Temps d'exécution	Temps d'attente
A	3	3	0
B	8	6	2
C	9	4	5
D	9	2	7
E	9	1	8

- **Le temps moyen d'attente est : (0+2+5+7+8)/5=4.4**

Ordonnanceurs non préemptifs : Correction Exercice I : SJF

- **Pour la stratégie SJF nous aurons la séquence d'exécution A,B,E,D,C, et le temps de séjour est : temps de terminaison – temps d'entrée**

Processus	Temps de terminaison	Temps d'arrivée	Temps de séjour
A	3	0	3
B	9	1	8
E	10	7	3
D	12	6	6
C	16	4	12

- **Le temps moyen de séjour est (3+8+3+6+12)/5 = 6.4**

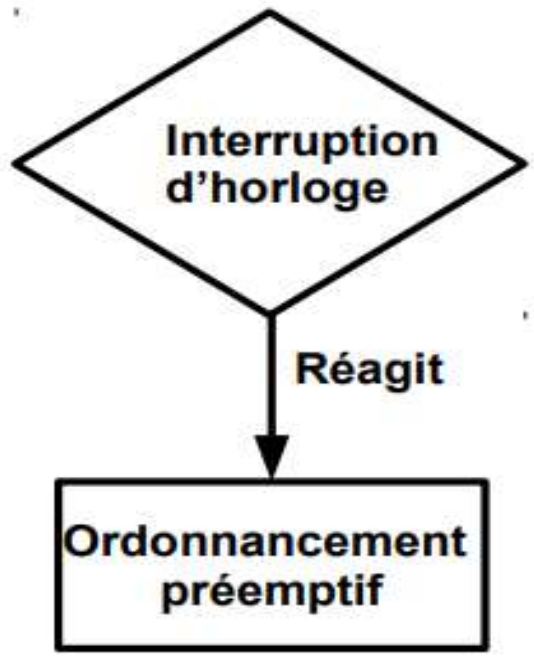
Ordonnanceurs non préemptifs : Correction Exercice I : SJF

- **Temps d'attente = temps de séjours – temps d'exécution**

Processus	Temps de séjour	Temps d'exécution	Temps d'attente
A	3	3	0
B	8	6	2
E	3	1	2
D	6	2	4
C	12	4	8

- **Le temps moyen d'attente est (0+2+2+4+8)/5 = 3.2**

Ordonnanceurs préemptifs



Ordonnanceurs préemptifs

- **Si le processus courant doit suspendre son exécution au profit d'un autre :**
  - l'OS doit d'abord sauvegarder le contexte du processus
  - avant de charger le contexte du processus à lancer.

- **C'est qu'on appelle la commutation de contexte ou le changement de contexte**
  - Cette sauvegarde est nécessaire pour pouvoir poursuivre ultérieurement l'exécution du processus suspendu.

Ordonnanceurs préemptifs

- Le temps d'allocation du processeur au processus est appelé quantum.

- La commutation entre processus doit être rapide, c'est-à-dire, exiger un temps nettement inférieur au quantum

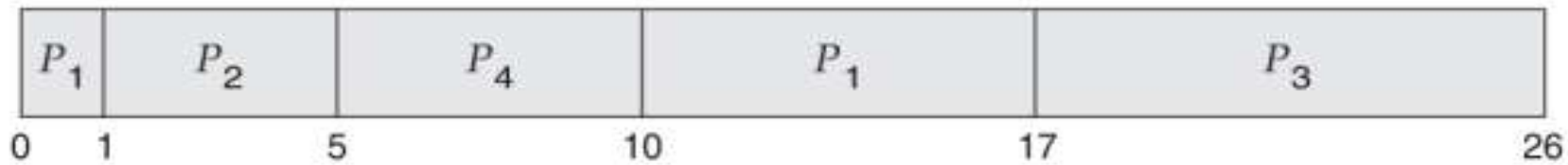
Ordonnancement du plus petit temps de séjour Shortest Remaining Time (SRT)

- SRT= la version préemptive de l'algorithme SJF.

- Si un processus dont le temps d'exécution est plus court que le reste du temps d'exécution du processus en cours de traitement, alors il prendra sa place.

Shortest Remaining Time (SRT)

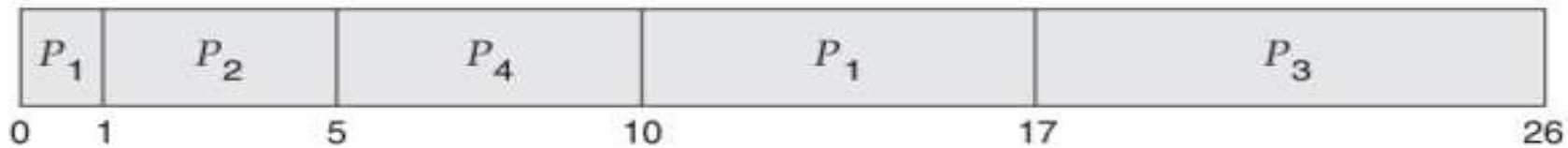
Processus	Temps d'arrivée	Temps d'exécution
P1	0	8
P2	1	4
P3	2	9
P4	3	5



- P1 commence à t=0, il est le seul processus dans la file d'attente
- P2 arrive à t=1ms
- Le temps restant à P1 est de 7ms > temps demandé par P2, 4ms
  → P2 est exécuté et P1 retourne dans le file d'attente
- Le temps moyen d'attente = [(10 -1) + (1-1) +(17-2) +(5-3)] /4
  =6.5ms < 7.75ms le temps moyen d'attente de SJF

Shortest Remaining Time (SRT)

Processus	Temps d'arrivée	Temps d'exécution	Temps de séjour : Terminaison-Entrée	Temps d'attente : Séjour - Exécution
P1	0	8	17-0=**17**	17 -8 = **9**
P2	1	4	5-1=**4**	4 -4 =**0**
P3	2	9	26-2=**24**	24 -9 = **15**
P4	3	5	10-3=**7**	7-5=**2**

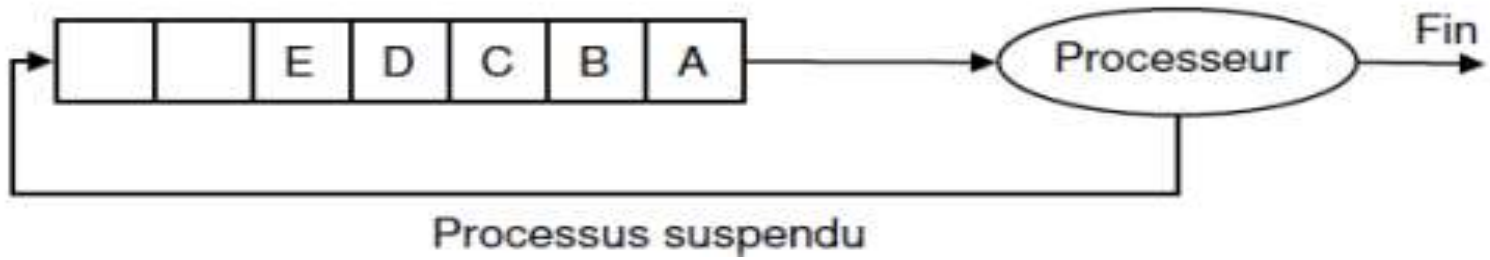


- **Le temps moyen d'attente = (9+0+15+2) /4 =6.5ms < 7.75ms le temps moyen d'attente de SJF**

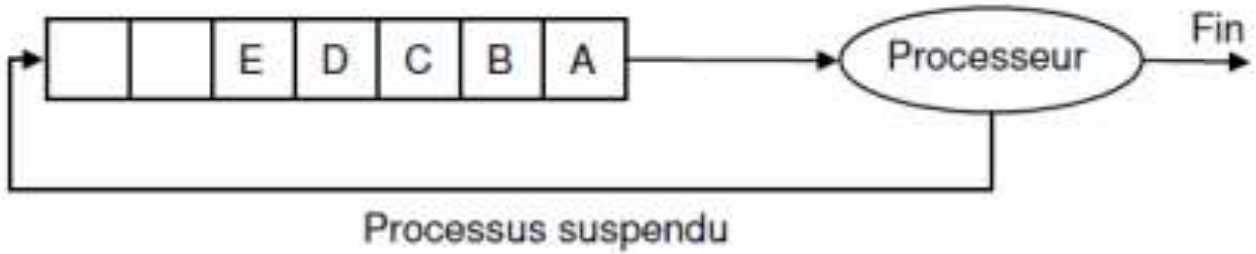
Ordonnancement circulaire
Round Robin (RR)

- **L'algorithme du tourniquet, circulaire ou round robin est un algorithme ancien, simple, fiable et très utilisé.**

- **Il mémorise dans une file du type FIFO (First In First Out) la liste des processus prêts, c'est-à-dire en attente d'exécution.**



Choix du processus à exécuter dans RR

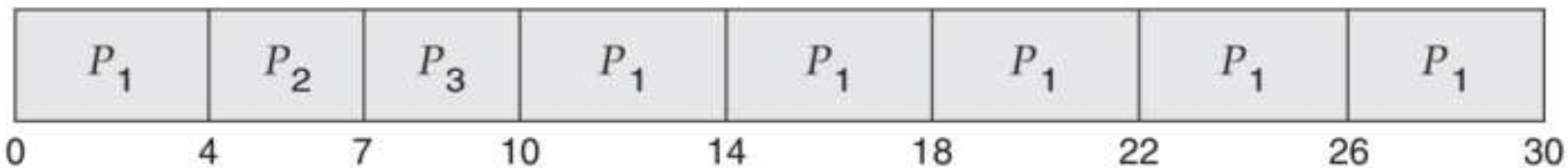


- **Il alloue le processeur au processus en tête de file, pendant un quantum de temps.**
- **Si le processus se bloque ou se termine avant la fin de son quantum, le processeur est immédiatement alloué à un autre processus (celui en tête de file).**
- **Si le processus ne se termine pas au bout de son quantum, son exécution est suspendue.**
  - **Le processeur est alloué à un autre processus (celui en tête de file).**
  - **Le processus suspendu est inséré en queue de file.**
  - **Les processus qui arrivent ou qui passent de l'état bloqué à l'état prêt sont insérés en queue de file.**

Exemple de RR

P1	24
P2	3
P3	3

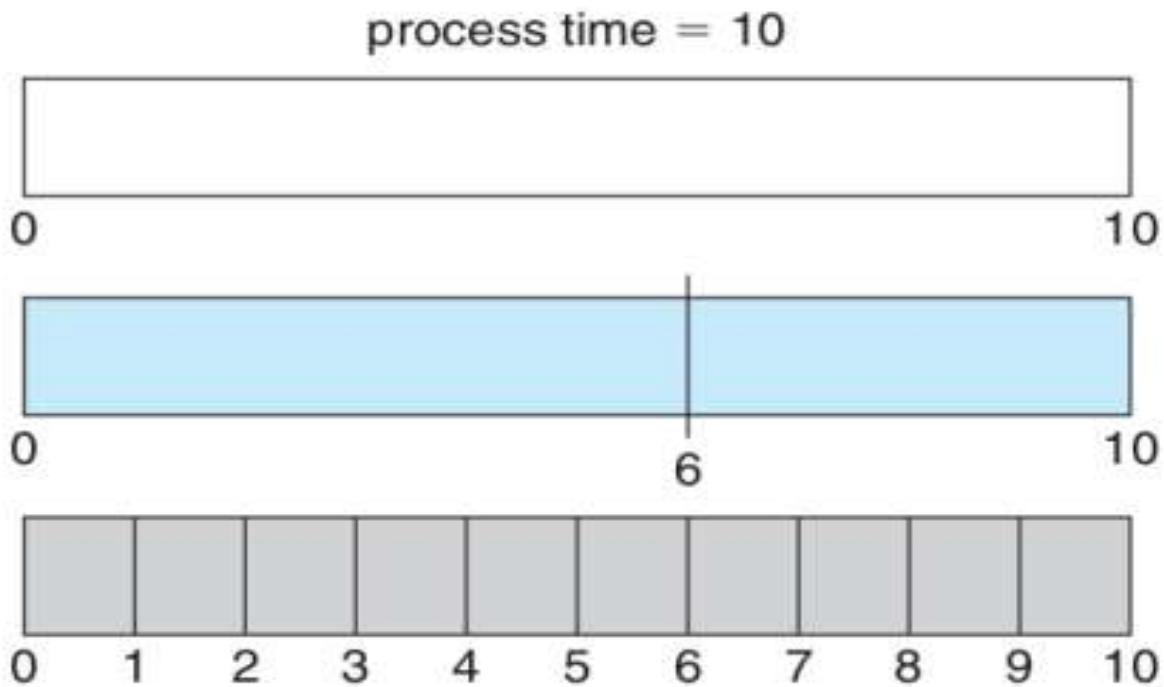
- Quantum = 4ms



Impact de la valeur du quantum dans RR

- Un quantum trop petit provoque trop de commutations de processus et abaisse l'efficacité du processeur.
- Un quantum trop élevé augmente le temps de réponse des courtes commandes en mode interactif.
- Un quantum entre 20 et 50 ms est souvent un compromis raisonnable

Impact de la valeur du quantum dans RR



quantum	context switches
12	0
6	1
1	9

Ordonnanceurs préemptifs
Exercice II

- **Soient deux processus A et B prêts tels que A est arrivé en premier suivi de B, 2 unités de temps après. Les temps de processeur nécessaires pour l'exécution des processus A et B sont respectivement 15 et 4 unités de temps.**

Processus	Temps d'arrivée	Temps d'exécution
A	0	15
B	2	4

- **Le temps de commutation est supposé nul. Calculer le temps de séjour de chaque processus A et B, le temps moyen de séjour, le temps d'attente, le temps moyen d'attente, et le nombre de changements de contexte pour:**
  - SRT
  - Round robin (quantum = 10 unités de temps)
  - Round robin (quantum = 3 unités de temps)

Correction exercice II : SRT

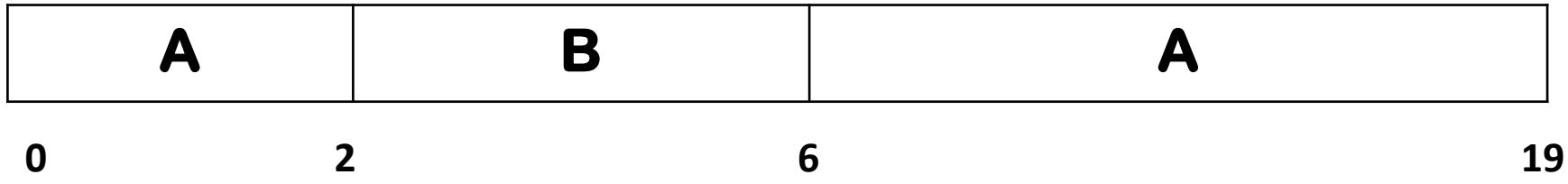
Processus	Temps de séjour
A	19-0 = 19
B	6-2 = 4

Temps moyen de séjour = $\frac{19+4}{2} = 11,5$

Processus	Temps d'attente
A	19-15 = 4
B	4-4 = 0

Le temps moyen d'attente = $\frac{4+0}{2} = 2$

Il y a 3 changements de contexte.



Correction exercice II : RR avec quantum=10

Processus	Temps de séjour
A	19-0 = 19
B	14-2 = 12

Temps moyen de séjour = $\frac{19+12}{2} = 15,5$

Processus	Temps d'attente
A	19-15 = 4
B	12-4 = 8

Le temps moyen d'attente = $\frac{4+8}{2} = 6$

Il y a 3 changements de contexte.

Correction exercice II : RR avec quantum=3

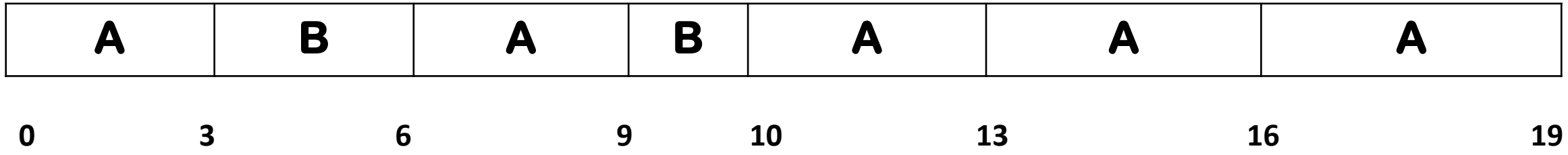
Processus	Temps de séjour
A	19-0 = 19
B	10-2 = 8

Temps moyen de séjour = $\frac{19+8}{2} = 13,5$

Processus	Temps d'attente
A	19-15 = 4
B	8-4 = 4

Le temps moyen d'attente = $\frac{4+4}{2} = 4$

Il y a 5 changements de contexte.



Ordonnanceurs préemptifs
Exercice III – Round Robin

- Considérons cinq Processus P1, P2, P3, P4, P5, dont les temps d'exécution et leurs temps d'arrivée respectifs sont les suivants

Processus	Temps d'arrivée	Temps d'exécution
P1	0	5
P2	1	3
P3	3	6
P4	5	1
P5	6	4

- Le temps de commutation est supposé nul. Dessiner le diagramme de GANTT pour l'ordonnancement Round Robin (quantum = 3 unités de temps)

Correction exercice III : RR avec quantum=3

T=0
P1

T=1
P2

T=3
P3
P1(2)

T=5
P3
P1(2)
P4

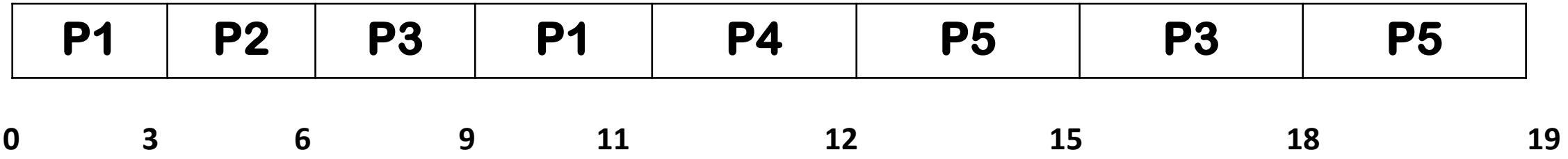
T=6
~~P3~~
P1(2)
P4
P5

T=9
~~P1(2~~)
P4
P5
P3(3)

T=11
~~P4~~
P5
P3(3)

T=12
~~P5~~
P3(3)

T=15
~~P3(3~~)
P5(1)



Correction exercice III : RR avec quantum=3

processus	T. Arrivée	Exécution =BT	Terminaison	T.séjour=T.rotation	T.attente
P1	0	5	11	11-0=11	11-5=6
P2	1	3	6	6-1=5	5-3=2
P3	3	6	18	18-3=15	15-6=9
P4	5	1	12	12-5=7	7-1=6
P5	6	4	19	19-6=13	13-4=9

Ordonnanceurs préemptifs
Exercice IV – Round Robin

- Considérons cinq Processus A, B, C, D, e, dont les temps d'exécution et leurs temps d'arrivée respectifs sont les suivants

Processus	Temps d'arrivée	Temps d'exécution
A	0	3
B	2	6
C	4	4
D	6	5
E	8	2

- Le temps de commutation est supposé nul. Dessiner le diagramme de GANTT pour l'ordonnancement Round Robin (quantum = 1 unités de temps ensuite quantum = 3)

Ordonnancement avec priorité

- Certains processus sont plus importants ou urgents que d'autres.

- L'ordonnanceur à priorité attribue à chaque processus une priorité. Le choix du processus à élire dépend des priorités des processus prêts.

Attribution et évolution des priorités

- Pour empêcher les processus de priorité élevée de s'exécuter indéfiniment, l'ordonnanceur diminue régulièrement la priorité du processus en cours d'exécution.

- La priorité du processus en cours est comparée régulièrement à celle du processus prêt le plus prioritaire (en tête de file). Lorsqu'elle devient inférieure, la commutation a lieu.

- Dans ce cas, le processus suspendu est inséré en queue de le correspondant à sa nouvelle priorité. L'attribution et l'évolution des priorités dépendent des objectifs fixés et de beaucoup de paramètres.

Exercice V : Ordonnancement avec priorité

Processus	Temps d'arrivée	Temps d'exécution	Priorité
P1	0	5	4
P2	2	4	2
P3	2	2	6
P4	4	4	3

Le nombre de priorité élevé correspond à une priorité plus importante :

prio(P3)>prio(P1)>prio(P4)>prio(P2)

correction Exercice V : Ordonnancement avec priorité

T=0
~~P1~~

T=2
~~P3~~
P1(3)
P2

T=4
~~P1(3)~~
P4
P2

T=7
~~P4~~
P2

T=11
P2

correction Exercice V : Ordonnancement avec priorité

processus	T. Arrivée	Exécution =BT	Terminaison	T.séjour=T.rotation	T.attente
P1	0	5	7	7-0=7	7-5=2
P2	2	4	15	15-2=13	13-4=9
P3	2	2	4	4-2=2	2-2=0
P4	4	4	11	11-4=7	7-4=3