Institut Supérieur d'Informatique

Université de Tunis el Manar

TD1 : Rappel sur l'ordonnancement des processus

Systèmes d'Exploitation Avancés – 1ère année ING.

Année Universitaire : 2011/2012

MME. LILIA SFAXI

TD1 : Rappel sur l'ordonnancement des processus

Systèmes d'Exploitation Avancés

Exercice 1 : Généralités

1. Qu'est-ce qu'un processus ?

Le processus est un concept fondamental de tout système d'exploitation. Un processus est l'unité système qui permet l'exécution d'un programme.

Un processus est un programme en exécution. Il définit un objet dynamique tandis que le programme est un objet statique.

Un processus est caractérisé par son pointeur de pile, ses instructions et ses données,… Ces attributs définissent le contexte d'un processus.

2. Qu'est-ce qu'un PCB ?

Chaque processus est représenté dans le système d'exploitation par une structure de données contenant toute information décrivant le contexte du processus appelé bloc de contrôle (Process Control Bloc: PCB).

- ✓ *Attributs d'un PCB:*
  - *PID et PPID,*
  - *État,*
  - *Priorité,*
  - *Compteur ordinal,*
  - *Fichiers ouverts,*
  - *Pointeurs: seg. code, seg. données, seg. Pile,*
  - *Temps d'exécution.*

3. Qu'est-ce qu'un thread ?

Un thread ou encore processus léger (lightweight process) est une unité d'exécution de code. Il est issu d'un processus mais ne contenant que la pile d'exécution.

Un processus contient donc au moins un thread de contrôle unique en plus de l'espace d'adressage (segments code et données).

4. Quelle est la différence entre un processus et un thread ?

Un processus peut contenir un ou plusieurs threads.
Un processus a son propre PCB, alors que des threads dans le même processus partagent certaines informations du même PCB. (même espace d'adressage)

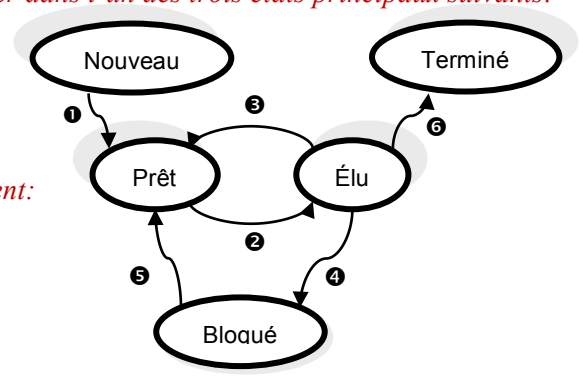
5. Qu'est ce qu'un PID ? PPID ?

Un processus est identifié par un PID (Process IDentifier) et un PPID (Parent Process IDentifier).

6. Quels sont les différents états d'un processus ? Quelles sont les transitions entre ces états ?

Lorsqu'un processus s'exécute; il change d'état. Il peut se trouver dans l'un des trois états principaux suivants:

- *Prêt (Ready) : le processus attend son tour pour s'exécuter*
- *Élu (Running) : les instructions sont en cours d'exécution.*
- *Bloqué (Sleep) : le processus bloqué en attente d'événement: signal, E/S, ...*



Graphe des états d'un processus

Transitions :

(1) Création du processus

(2) Allocation du processeur

(3) Fin du temps alloué sur le processeur, l'exécution du processus n'est pas terminée

(4) Opération E/S

(5) Fin opération E/S

(6) Exécution terminée

7. Qu'est-ce que la commutation de contexte ?

Sur un système multiprogrammé, le SE doit redonner le contrôle du processeur d'un processus à un autre en effectuant des ***commutations de contexte****.*
La commutation de contexte consiste à mémoriser le PCB du processus courant et charger le PCB du processus à élire.

8. Quels sont les différents types d'algorithmes d'ordonnancement ?

Il existe deux types d'algorithmes d'ordonnancement:

- ✓ *Ordonnancement sans réquisition (sans préemption): exécution d'un processus jusqu'à sa terminaison.*
- ✓ *Ordonnancement avec réquisition (avec préemption): suspension possible d'un processus élu même s'il n'a pas terminé son exécution.*

9. Quand est-ce que l'ordonnanceur est invoqué ?

Chaque fois que le processus exécutant est interrompu :

- ✓ *un processus exécutant devient bloqué (4)*
- ✓ *un processus change d'élu à prêt (3)*
- ✓ *un processus exécutant se termine (6)*

Chaque fois qu'un nouveau processus est prêt

- ✓ *un processus se présente en tant que nouveau (1)*
- ✓ *un processus change de bloqué à prêt (5)*

L'ordonnanceur choisit un processus parmi les processus prêts et lui alloue le processeur.

10. Comment juger de l'efficacité d'un algorithme d'ordonnancement ?

Un bon algorithme d'ordonnancement :

- ✓ *Chaque processus doit avoir sa part de temps CPU: **équité**.*
- ✓ *Utiliser le temps processeur a 100%: **efficacité**.*
- ✓ *Minimiser le **temps de réponse** en mode interactif.*

11. Discuter les différents algorithmes (FCFS, SJF) et leur mise en œuvre pratique.

- ***Premier arrivé, premier servi, FCFS**:*
  - *Temps moyen d'attente non-optimal*

- *Mauvaise utilisation des ressources s'il y a apport continu de processus aux cycles longs (v. effet d'accumulation)*

- ***Plus court servi, SJF:***
  - *Difficulté de prévoir la durée du prochain cycle*
  - *Famine possible des processus 'longs' s'il y a apport continu de processus aux cycles courts*

⇨ *Donc besoin d'une méthode systématiquement préemptive*

- ***Le tourniquet, RR :***
  - *Le temps processeur est divisé en intervalles de temps appelés Quantum Q, chaque processus s'exécutera exactement pendant son quantum.*
  - *Le processeur sera réquisitionné jusqu'à ce que:*
    - *Le processus élu ait épuisé son quantum,*
    - *Le processus élu ait fini son exécution avant la fin de son quantum,*
    - *Le processus élu demande une entrée/sortie.*

12. Définir interruption, préemption et déroutement et discuter la différence.

Interruption *: Une interruption est un signal produit par un périphérique et envoyé vers le processeur pour l'informer de la fin d'une E/S, la production d'une erreur …*

Préemption *: La préemption est la capacité d'un système d'exploitation multitâche à exécuter ou stopper une tâche planifiée en cours en faveur d'une tâche de priorité supérieure.*

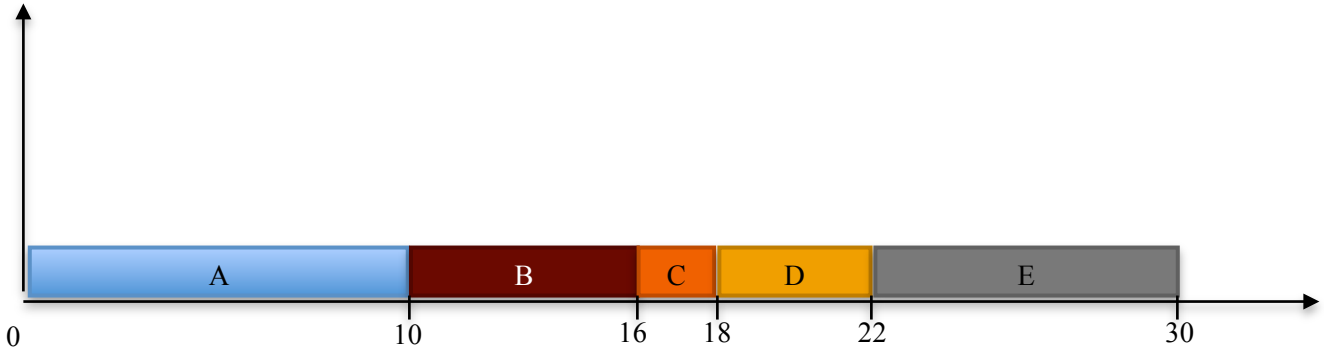
Déroutement *: Erreur interne du processus. Toujours prévisible (exemple : division par zéro)*

Exercice 2 : Ordonnancement des processus

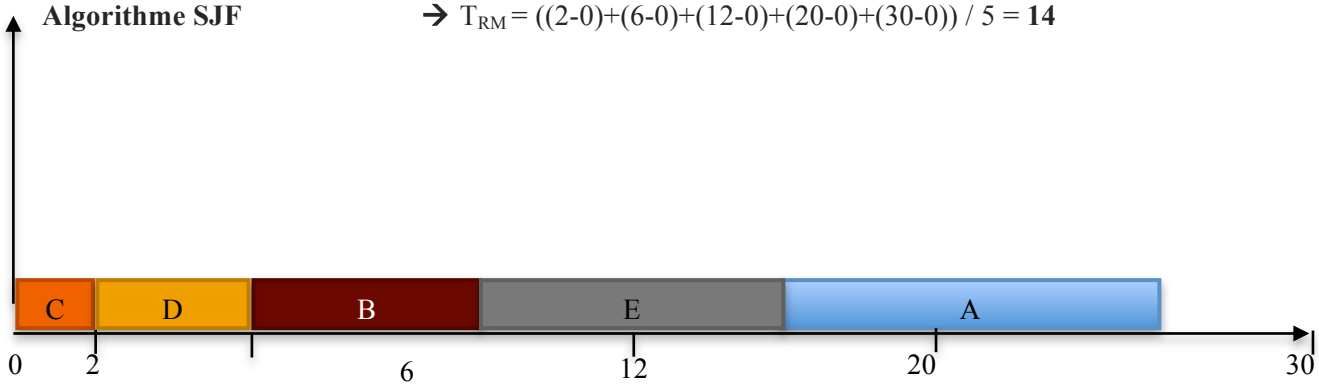
Cinq travaux A, B, C, D et E arrivent pratiquement en même temps dans un centre de calcul. Leur temps d'exécution respectif est estimé à 10, 6, 2, 4 et 8 secondes. Tracez le digramme de Gantt et déterminez le temps moyen de rotation pour chacun des algorithmes d'ordonnancement suivants. Ne tenez pas compte du temps perdu lors de la commutation des processus.

- Premier arrivé, premier servi FCFS (exécution dans l'ordre 10, 6, 2, 4, 8) ;
- Plus court d'abord SJF ;
- Tourniquet (quantum q = 4 s).

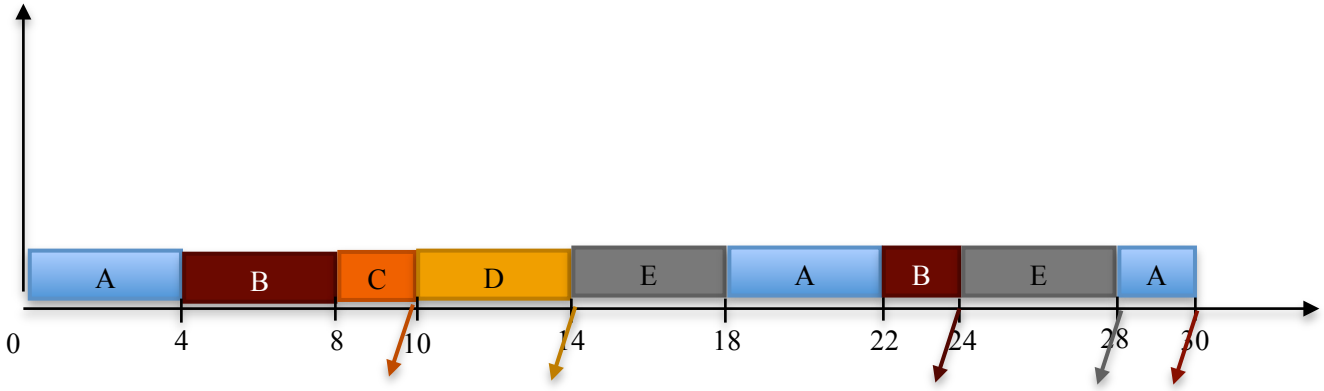
Algorithme FCFS → T_{RM} = ((10-0)+(16-0)+(18-0)+(22-0)+(30-0)) / 5 = **19,2**



Algorithme SJF → T_{RM} = ((2-0)+(6-0)+(12-0)+(20-0)+(30-0)) / 5 = **14**



Algorithme RR (quantum=4) → T_{RM} = ((10-0)+(14-0)+(24-0)+(28-0)+(30-0)) / 5 = **21,2**



Exercice 3 : Questions de compréhension

1. Quel est l'effet de la diminution du quantum sur les performances de l'algorithme RR (tourniquet)? Et son augmentation ?

La diminution du quantum entraîne la dégradation des performances de l'algorithme RR, car le temps de commutation/changement de contexte augmente. L'augmentation du quantum entraîne le manque d'efficacité de l'algorithme → devient FCFS.

2. Les algorithmes d'ordonnancement basés sur des priorités peuvent engendrer la famine (non exécution) des processus à faible priorité. Comment peut-on éviter ce problème ?

On utilise un autre algorithme: tourniquet ou priorité dynamique.

3. Soit un système sur lequel les processus s'exécutent en moyenne pendant un temps de T secondes. La commutation de processus nécessite C secondes. L'ordonnancement est circulaire (RR ou Tourniquet) avec un quantum de Q secondes. Donnez une interprétation pour chacun des cas suivants :

Q > T	C < Q < T	Q = C	Q tend vers + ∞	Q tend vers 0

Q>T et Q tend vers l'infini : RR tend vers FiFo / C<Q<T : cas raisonnable / Q=C et Q tend vers 0 : dégradation de l'efficacité d'exploitation du processeur.

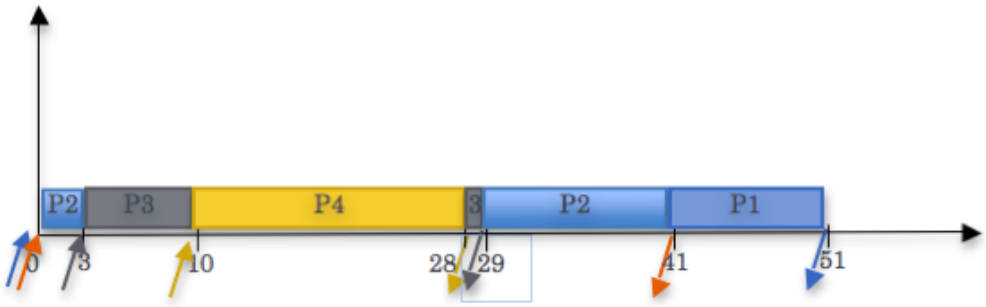
Exercice 4 : Ordonnancement avec priorité dynamique

On considère l'ensemble des processus suivants :

Processus	Date d'arrivée	Temps CPU	Priorité
P1	7h00	10 mn	2
P2	7h00	15 mn	3
P3	7h03	8 mn	4
P4	7h10	18 mn	5

A- On suppose qu'on utilise un algorithme d'ordonnancement basé sur la priorité. Donnez le diagramme de Gantt pour les processus donnés.

Priorité Statique → *TRM = ((28-10)+(29-3)+(41-0)+(51-0)) / 4 = 34*
→ *TAM = ((18-18)+(26-8)+(41-15)+(51-10)) / 4 = 21,25*

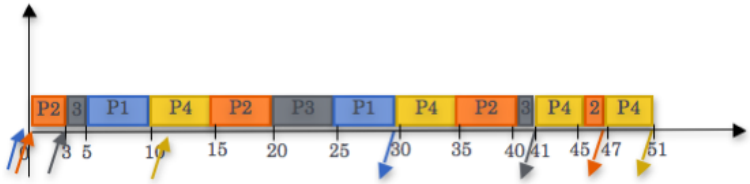


B- On voudrait que la priorité des processus soit dynamique au cours du temps. Ainsi, pour calculer la priorité d'un processus, on utilise la formule suivante :

$$Priorité = \frac{Temps\ d'attente + Temps\ CPU\ restant}{Temps\ CPU}$$

Remarque : Lors des calculs, on arrondira suivant l'exemple suivant : 3.5 ou 3.6 -> 4 ; 3.1 ou 3.4 -> 3.

1. Donnez le diagramme de Gantt sachant que la priorité est recalculée toutes les 5 minutes.



	0	3	5	10	15	20	25	30	35	40	41	45
P1	2	2	5+10/10 =2	0+5/10= 1	5+5/10=1	10+5/10 =2	15+5/10= 2					
P2	3	3	2+12/15 =1	7+12/15 =1	12+12/15 =2	0+7/15= 0	5+7/15=1	10+7/15= 1	15+7/15 =1	0+2/15 =0	0	5+2/15 =0
P3		4	0+6/8=1	5+6/8=1	10+6/8=2	15+6/8= 3	0+1/8=0	5+1/8=1	10+1/8= 1	15+1/8 =2		
P4				5	0+13/18= 1	5+13/18 =1	10+13/18 =1	15+13/18 =2	0+8/18= 0	5+8/18 =1	1	0+4/18 =0

Hypothèses :

- *Au démarrage, les priorités des processus sont égales à leurs priorités statiques (indiquées dans le tableau).*

- *Les priorités ne sont calculées que toutes les 5mn. Pour les autres temps, on prend la priorité précédente.*
- *Le temps d'attente d'un processus est son temps d'attente depuis sa dernière exécution.*
- *Si on a le choix entre deux processus de même priorité, on choisit celui qui attend depuis le plus longtemps.*

2. Calculez le temps d'attente moyen ainsi que le temps de rotation moyen.

Priorité Dynamique → TRM = ((30-0)+(47-0)+(41-3)+(51-10)) / 4 = 39

→ TAM = ((30-0-10)+(47-0-15)+(41-3-8)+(51-10-18)) / 4 = 26,25

3. Comparez les résultats obtenus par rapport à ceux obtenus avec l'algorithme de priorité classique.

Le temps de rotation pour l'ordonnancement à priorité dynamique est plus grand, mais l'algorithme est plus équitable.