TSE2531: Technique des Systèmes d'Eploitation

Les processus et leur ordonnancement

Par: Miguel Landry FOKO SINDJOUNG, PhD

Institut Universitaire de Technologie Fotso Victor de Bandjoun
Filière: Génie-Informatique
Niveau: DUT 2

Novembre 2020

Objectifs

- Définir la notion de processus ainsi que les différents états de ce dernier
- Comprendre les principes par lesquels le SE alloue le processeur aux processus
- Identifier les différents états d'un processus
- Créer les processus dans un SE
- Identifer la différence entre un processus léger et un processus lourd

Qu'est-ce qu'un processus ?

Un **processus** est un programme en cours d'exécution.

Qu'est-ce qu'un processus ?

Un **processus** est un programme en cours d'exécution.

Ses composantes sont les suivants :

- Les **données** (variables globales, pile et tas) stockées dans une zone de la mémoire qui a été allouée au processus ;
- La **valeur des registres** (généraux et spécialisés) de la CPU lors de l'exécution ;
- Les **ressources** qui lui ont été allouées par le système d'exploitation (mémoire principale, fichiers ouverts, périphériques utilisés, etc.) ;

Qu'est-ce qu'un processus ?

Un **processus** est un programme en cours d'exécution.

Ses composantes sont les suivants :

- Les **données** (variables globales, pile et tas) stockées dans une zone de la mémoire qui a été allouée au processus ;
- La **valeur des registres** (généraux et spécialisés) de la CPU lors de l'exécution ;
- Les **ressources** qui lui ont été allouées par le système d'exploitation (mémoire principale, fichiers ouverts, périphériques utilisés, etc.) ;

L'ensemble de ces composantes forme le **contexte d'exécution** d'un processus ou plus simplement le **contexte**

État d'un processus (1)

- Un processus n'est pas continuellement en train de s'exécuter

État d'un processus (1)

- Un processus n'est pas continuellement en train de s'exécuter
- Si la machine comporte n processeurs identiques, à un instant donné il y a au maximum n processus actifs

État d'un processus (1)

- Un processus n'est pas continuellement en train de s'exécuter
- Si la machine comporte n processeurs identiques, à un instant donné il y a au maximum n processus actifs
- En fait, parmi tous les processus qui sont susceptibles de s'exécuter, seulement un petit nombre s'exécutent réellement

État d'un processus (1)

- Un processus n'est pas continuellement en train de s'exécuter
- Si la machine comporte n processeurs identiques, à un instant donné il y a au maximum n processus actifs
- En fait, parmi tous les processus qui sont susceptibles de s'exécuter, seulement un petit nombre s'exécutent réellement
- L'allocation de la CPU aux processus qui la réclament est appelée **l'ordonnancement** de la CPU

État d'un processus (1)

- Un processus n'est pas continuellement en train de s'exécuter
- Si la machine comporte n processeurs identiques, à un instant donné il y a au maximum n processus actifs
- En fait, parmi tous les processus qui sont susceptibles de s'exécuter, seulement un petit nombre s'exécutent réellement
- L'allocation de la CPU aux processus qui la réclament est appelée **l'ordonnancement** de la CPU
- L'état opérationnel d'un processus est un moyen de représenter les différentes étapes de ce processus telles qu'elles sont gérées par le système d'exploitation

État d'un processus (2)

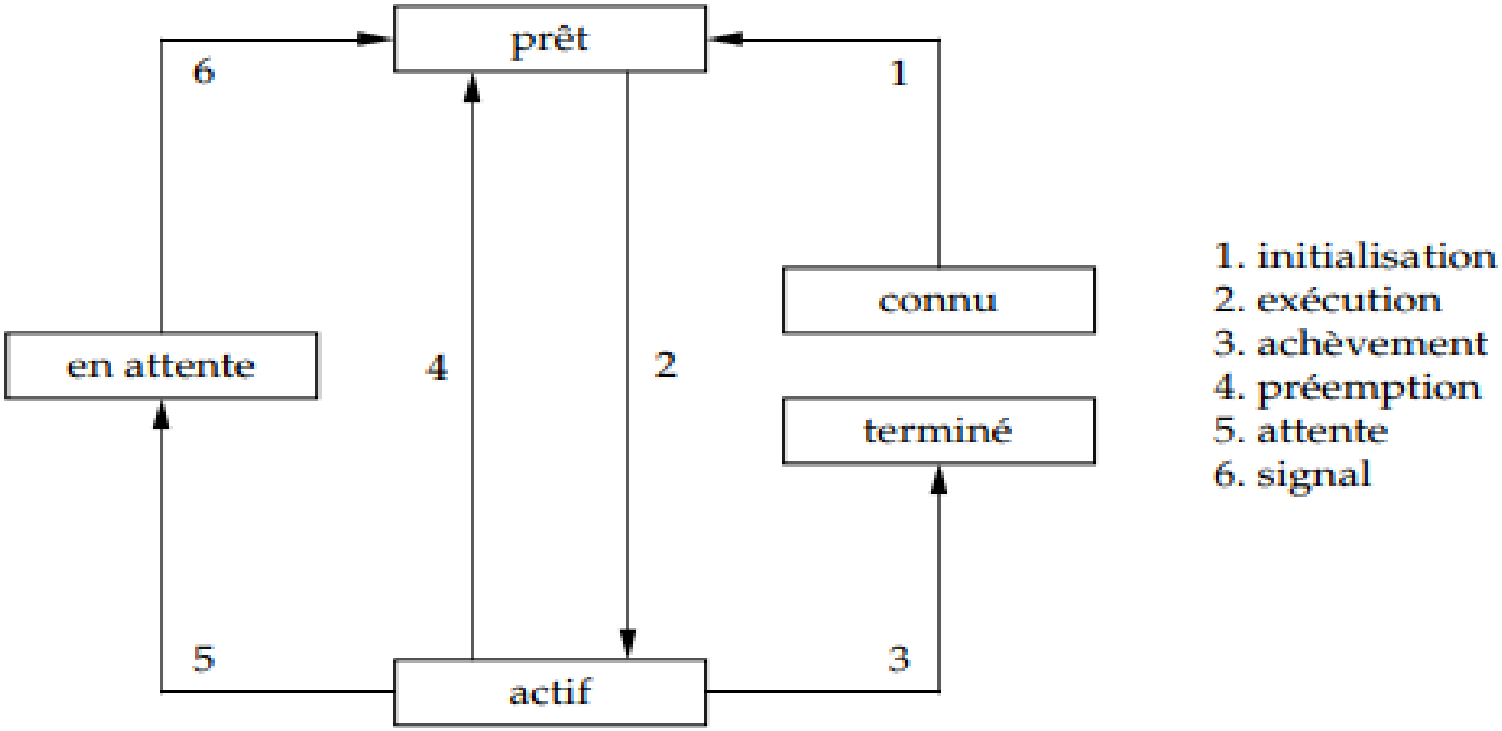


FIGURE – Schéma simplifié d'un CPU

État d'un processus (3)

1. Initialement, un processus est connu du système mais l'exécution n'a pas débuté.
2. Lorsqu'il est initialisé, il devient prêt à être exécuté (1).
3. Lors de l'allocation de la CPU à ce processus il devient actif (2). Trois cas peuvent alors se présenter :
    1. Le processus se termine (3)
    2. Le processus est en attente (5) d'un événement et dès sa réception il redeviendra prêt (6)
    3. Le processus est suspendu et se remet dans l'état prêt (4). Il y a réquisition ou préeemption de la CPU. Dans ce cas, le S.E. enlève la CPU au processus qui la détient.

Gestion des processus (1)

- Les processus sont les principaux éléments actifs du système

Gestion des processus (1)

- Les processus sont les principaux éléments actifs du système
- Un nouveau processus ne se crèe qu'à la demande d'un autre processus : **Que l'on qualifiera de processus père pour le processus nouvellement créé.**

Gestion des processus (1)

- Les processus sont les principaux éléments actifs du système
- Un nouveau processus ne se crèe qu'à la demande d'un autre processus : **Que l'on qualifiera de processus père pour le processus nouvellement créé.**
- Lors du démarrage de la machine, le S.E. lance un processus qui est orphelin puisqu'il n'a pas de père : Ce processus est souvent appelé **init**

Gestion des processus (1)

- Les processus sont les principaux éléments actifs du système
- Un nouveau processus ne se crèe qu'à la demande d'un autre processus : **Que l'on qualifiera de processus père pour le processus nouvellement créé.**
- Lors du démarrage de la machine, le S.E. lance un processus qui est orphelin puisqu'il n'a pas de père : Ce processus est souvent appelé **init**
- Le premier rôle de ce processus est de lancer des fils qui auront chacun une fonction dans l'organisation générale de la machine :
  - Un processus pour gérer les E/S asynchrones avec les terminaux
  - Un processus pour gérer les connexions au système avec demande et vérification d'un nom d'utilisateur et d'un mot de passe
  - Un processus pour gérer l'allocation de la CPU aux processus
  - etc.

Gestion des processus (2)

Le SE est donc composé d'un noyau résident qui ne s'exécute que sur demande explicite (interruptions et déroutements) et d'un ensemble de processus système qui ont chacun une fonction précise à assurer :

1. La partie résidente du système est réduite en taille ce qui permet d'éviter une trop grande consommation de mémoire par le système
2. Les processus systèmes ne sont pas forcément toujours prêts ou même toujours présents en mémoire ce qui permet encore une fois de réduire la mémoire et le temps CPU consommé par le S.E. au détriment des processus utilisateur.

Poids lourds et poids légers (1)

- Un thread (aussi appelé processus de poids léger ou lightweight process LWP) est un programme en cours d'exécution qui partage son code et ses données avec les autres threads d'un même processus
- les piles sont propres à chaque thread pour éviter que les appels de fonctions et les variables locales ne se mélangent.

Poids lourds et poids légers (2)

L'utilisation des threads présente plusieurs avantages :

- Si un processus ne comporte qu'un seul thread nous revenons au modèle classique ; les systèmes multi-threads sont donc plus généraux
- Il n'y a plus à mettre en place une communication entre les threads d'un même processus puisqu'ils agissent tous sur les mêmes données
- Le temps de commutation entre les threads d'un même processus est réduit car le contexte est le même, et seuls les registres de la CPU doivent être sauvegardés
- En associant un (ou plusieurs) thread(s) à chaque processeur on peut facilement exploiter une structure multi-processeurs.

Ordonnancement des processus (1)

- Le systeme d'exploitation doit donc gérer l'ensemble des processus, dans le but de maintenir le taux d'utilisation du processeur le plus élevé possible
- Pour cela, le systeme d'exploitation met en oeuvre une politique d'ordonnancement, algorithme régissant la commutation des tâches

Ordonnancement des processus (2)

- Utilisation de l'UC : il s'agit de prendre en compte le temps pendant lequel l'unité centrale exécute un processus
- Utilisation répartie : c'est le pourcentage de temps pendant lequel est utilisé l'ensemble des ressources.
- Débit : c'est le nombre de processus pouvant être exécutés par le système durant une période de temps donnée.
- Temps de rotation : durée moyenne de l'exécution d'un processus.
- Temps d'attente : durée moyenne d'attente des processus du proceseur
- Temps de réponse : le temps moyen pour répondre aux entrées de l'utilisateurs (systèmes interactifs)
- Equité : degré auquel tous les processeurs reçoivent une chance égale de s'exécuter

Ordonnancement des processus (3)

Principe :
Un processus est donc successivement en phases de calcul, et en phase d'entrées-sorties. L'idée est donc de tenter de se faire recouvrir une phase d'E/S d'un processus avec des phases de calcul d'un autre processus.

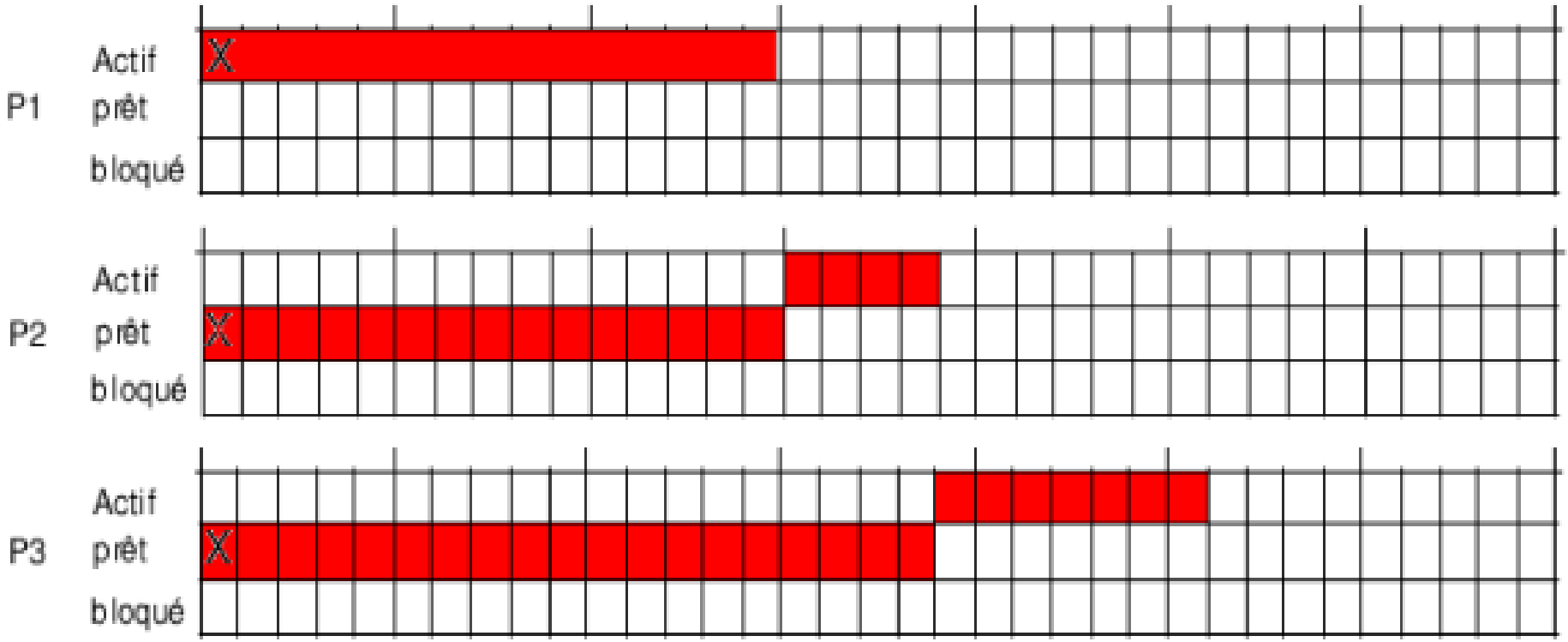
Algorithmes d'ordonnancement (4)

- Au début de l'informatique, l'ordonnancement était souvent non préemptif, ou coopératif
- Un processus conservait le contrôle de l'unité centrale jusqu'à ce qu'il se bloque ou qu'il se termine
- Pour des travaux par lots, le système convenait, le temps de réponse n'ayant pas grande importance
- Sur les sytèmes interactifs actuels, c'est l'ordonnancement préemptif qui est utilisé : le systeme d'exploitation peut préempter un processus avant qu'il ne se bloque ou ne se termine, afin d'attribuer le processeur à un autre processus

Algorithmes d'ordonnancement (5)

FIFO - First In First Out :

Ex : Soit trois processus P1,P2,P3 faisant uniquement du calcul. Les temps d'exécution des trois processus sont respectivement 15, 4 et 7 millisecondes



Algorithmes d'ordonnancement (6)

SJF - Shortest First Job :

En considérant le même exemple que précédement :

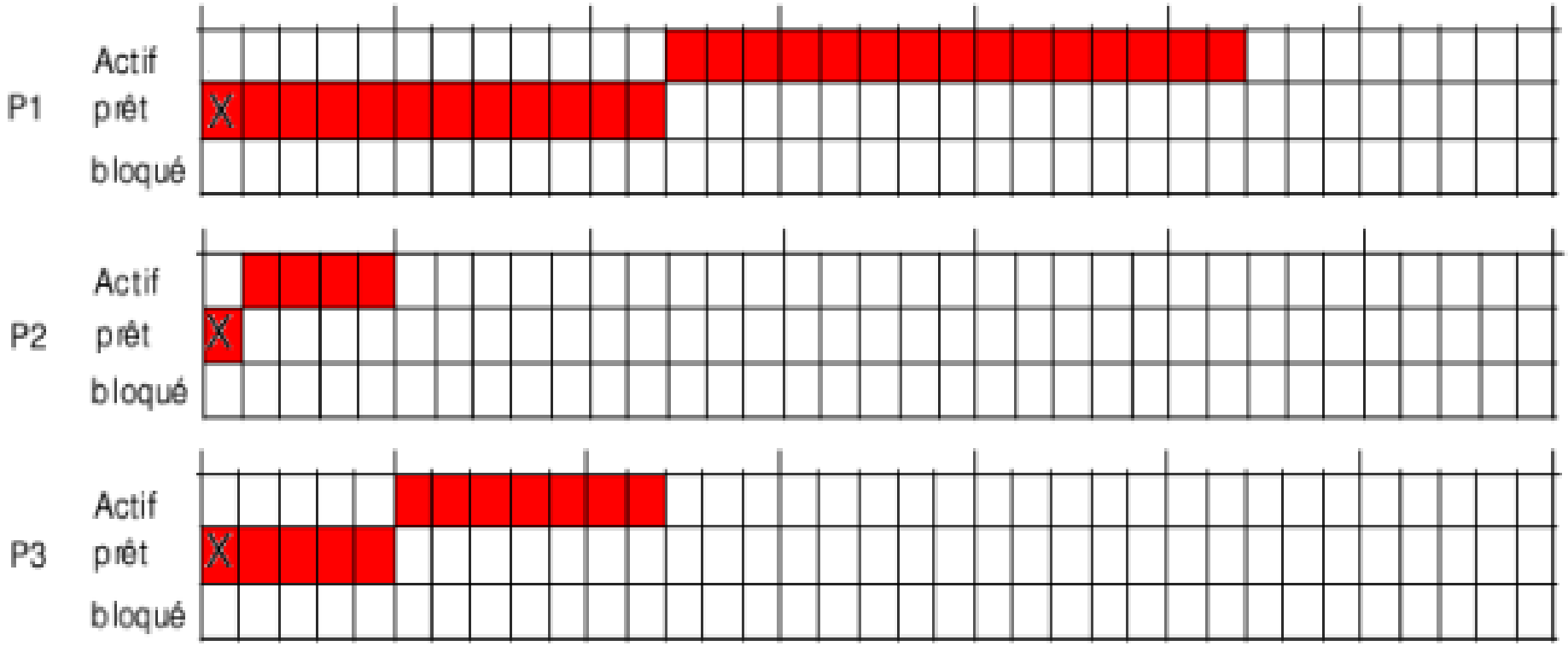
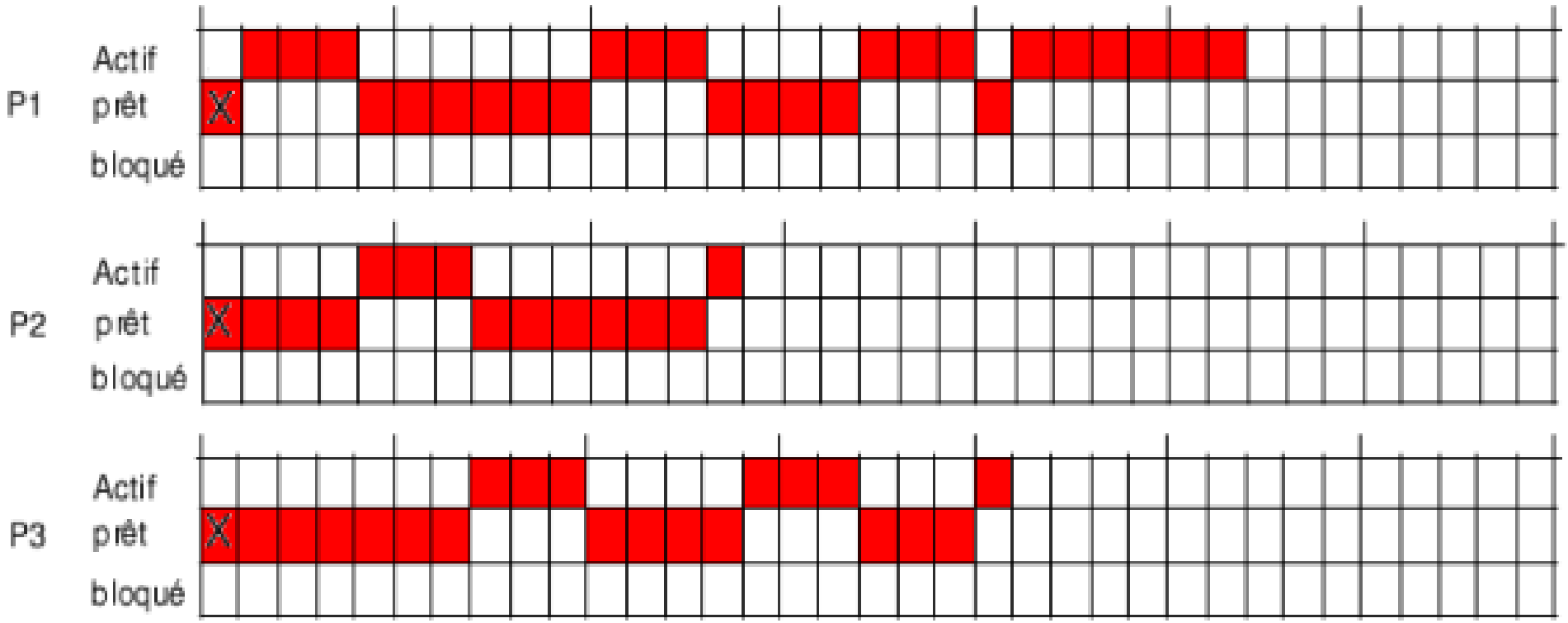


FIGURE – Exécution de l'algorithme d'ordonnancement SJF

Algorithmes d'ordonnancement (7)

RR - Round Robin (tourniquet) : est préemptif et sélectionne le processus attendant depuis le plus longtemps

En considérant le même exemple que précédement :



Algorithmes d'ordonnancement (8)

RR - Round Robin avec priorité (1)

- On affecte une valeur à chaque processus

Algorithmes d'ordonnancement (8)

RR - Round Robin avec priorité (1)

- On affecte une valeur à chaque processus
- Le processus actif est celui qui à la priorité la plus élevée (valeur la plus faible)

Algorithmes d'ordonnancement (8)

RR - Round Robin avec priorité (1)

- On affecte une valeur à chaque processus
- Le processus actif est celui qui à la priorité la plus élevée (valeur la plus faible)
- en cas d'égalité, on utilise FIFO

Algorithmes d'ordonnancement (8)

RR - Round Robin avec priorité (1)

- On affecte une valeur à chaque processus
- Le processus actif est celui qui à la priorité la plus élevée (valeur la plus faible)
- en cas d'égalité, on utilise FIFO
- Les priorités peuvent être fixées en fonction des caractéristiques du processus (beaucoup d'E/S, utilisation de la mémoire,...), de l'utilisateur, ou même d'une priorité fixée par l'administrateur ou l'utilisateur (commande nice sous Unix)

Algorithmes d'ordonnancement (8)

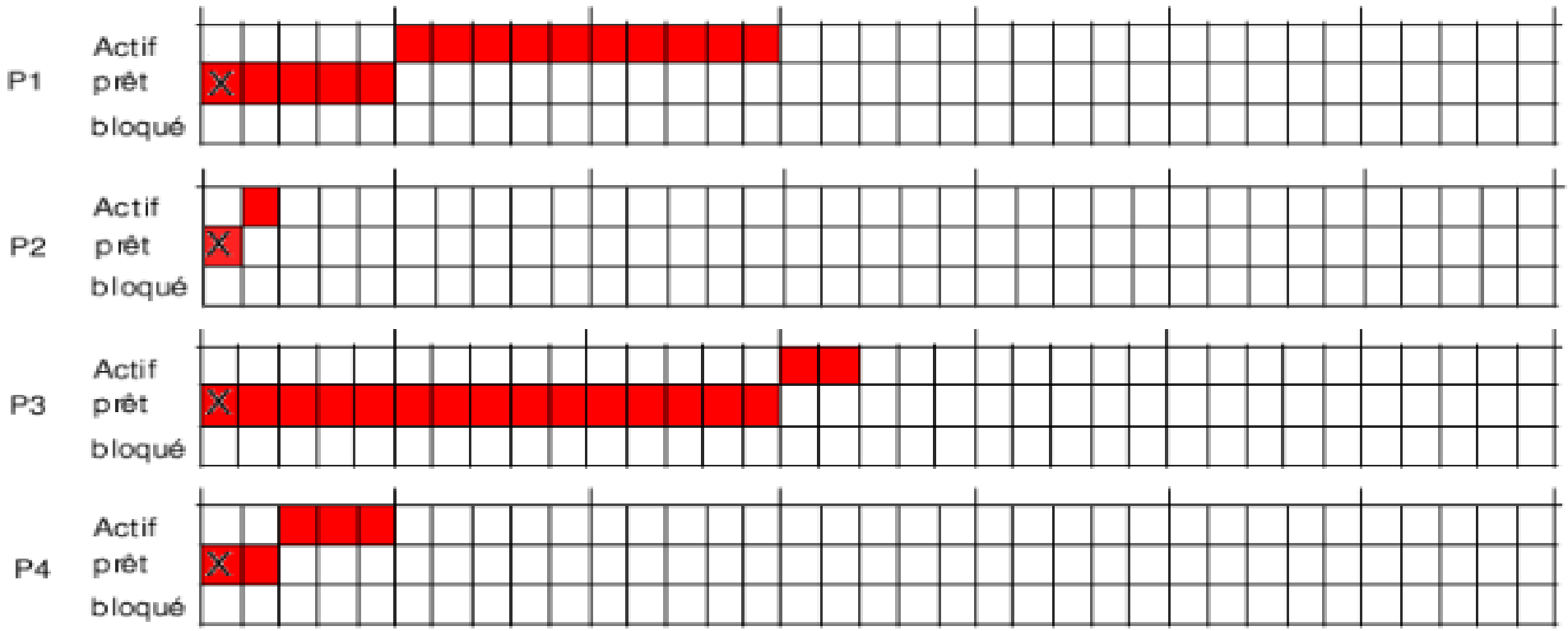
RR - Round Robin avec priorité (1)

- On affecte une valeur à chaque processus
- Le processus actif est celui qui à la priorité la plus élevée (valeur la plus faible)
- en cas d'égalité, on utilise FIFO
- Les priorités peuvent être fixées en fonction des caractéristiques du processus (beaucoup d'E/S, utilisation de la mémoire,...), de l'utilisateur, ou même d'une priorité fixée par l'administrateur ou l'utilisateur (commande nice sous Unix)
- Pour éviter le risque de famine des processus de trop faible priorité, on peut augmenter la priorité en fonction du temps d'attente écoulé

Algorithmes d'ordonnancement (9)

RR - Round Robin avec priorité (2)

Ex : Soit P1,P2,P3 et P4 des processus de temps d'exécution respectifs 10,1,2,3, et de priorité respectives 3,1,3,2



Algorithmes d'ordonnancement (10)

SR - Shortest Remaining Time :

- L'ordonnancement du temps restant le plus court est la version préemptif de l'algorithme SJF

Algorithmes d'ordonnancement (10)

SR - Shortest Remaining Time :

- L'ordonnancement du temps restant le plus court est la version préemptif de l'algorithme SJF
- Chaque fois qu'un processus passe dans l'état prêt (à sa création ou à la fin d'une E/S), on compare la valeur estimée du temps de traitement restant à celle du processus en cours

Algorithmes d'ordonnancement (10)

SR - Shortest Remaining Time :

- L'ordonnancement du temps restant le plus court est la version préemptif de l'algorithme SJF
- Chaque fois qu'un processus passe dans l'état prêt (à sa création ou à la fin d'une E/S), on compare la valeur estimée du temps de traitement restant à celle du processus en cours
- On préempte le processus en cours si son temps restant est plus long

Algorithmes d'ordonnancement (10)

SR - Shortest Remaining Time :

- L'ordonnancement du temps restant le plus court est la version préemptif de l'algorithme SJF
- Chaque fois qu'un processus passe dans l'état prêt (à sa création ou à la fin d'une E/S), on compare la valeur estimée du temps de traitement restant à celle du processus en cours
- On préempte le processus en cours si son temps restant est plus long
- L'algorithme privilégie les programmes courts, mais il peut y avoir un risque de famine pour les programmes longs

Algorithmes d'ordonnancement (11)

MQS - Multilevel Queue Scheduling (1)

- L'ordonnancement par files multi-niveaux est une extension de l'ordonnancement RR avec priorité
- Mais en plaçant les processus de même prioité dans des files d'attente distinctes
- Par exemple, les systèmes en temps partagé supportent généralement la notion de processus en premier (foreground) et en arrière plan (background)
- Les premiers sont souvent des processus interactifs, alors que les seconds n'ont besoin de s'exécuter que si le processeur est libre
- Donc les deux types de processus nécessitent deux ordonnancements différents

Algorithmes d'ordonnancement (12)

MQS - Multilevel Queue Scheduling (2)

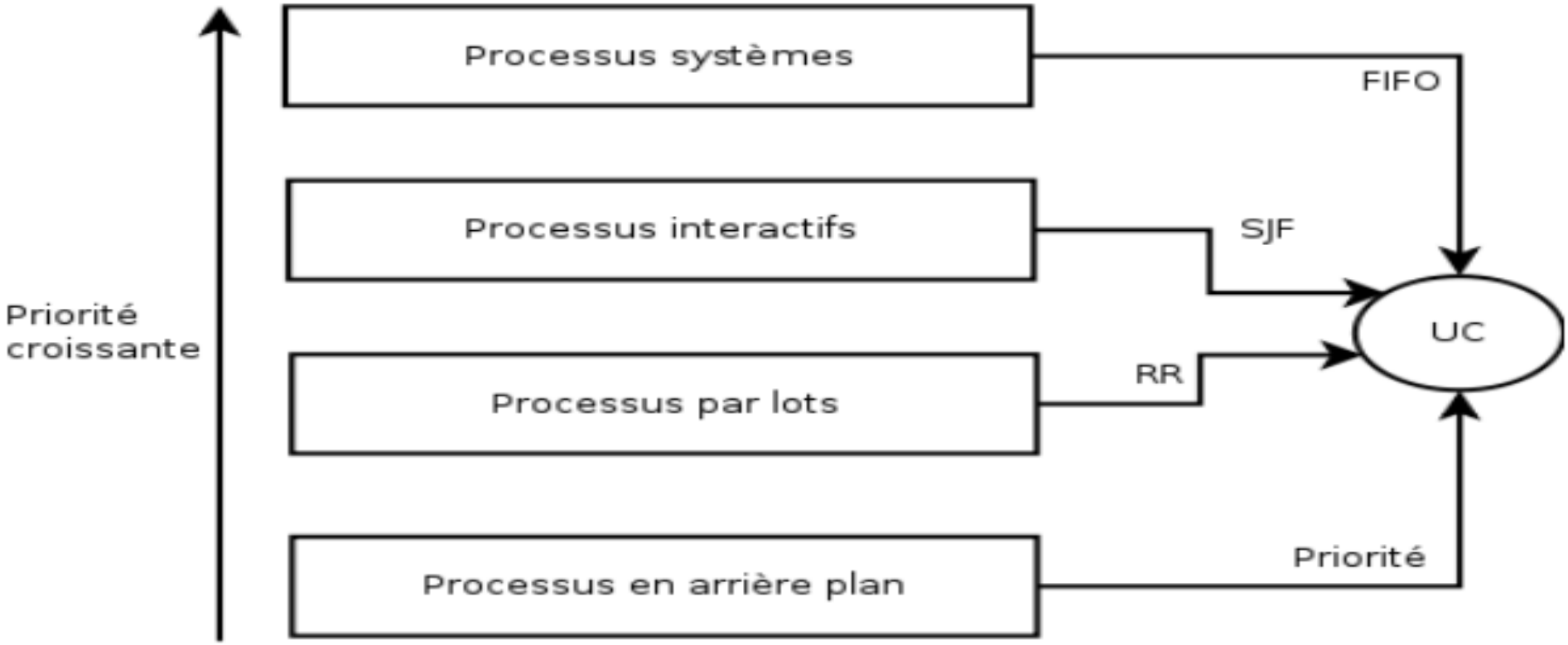


FIGURE – Exécution de l'algorithme d'ordonnancement MQS