Chapitre 3 :
L'ordonnancement

Les objectifs du cours

- Comprendre la problématique du multitâche
- Connaître la notion d'ordonnancement des processus et l'utilité d'un ordonnanceur
- Connaître les différents critères d'ordonnancement
- Connaître les algorithmes d'ordonnancement préemptifs
- Connaître les algorithmes d'ordonnancement non préemptifs

1. Introduction

Un ordinateur possède forcément plusieurs processus en concurrence pour l'obtention du temps processeur, cette situation se produit lorsque 2 ou plusieurs processus sont en état prêt simultanément.

L'Ordonnanceur (planificateur) est la partie (un programme) du système d'exploitation responsable de régler les états des processus (Prêt, Actif,...etc.) et de gérer les transitions entre ces états;

⇨ c'est l'allocateur du processeur aux différent processus, il alloue le processeur au processus en tête de file des Prêts

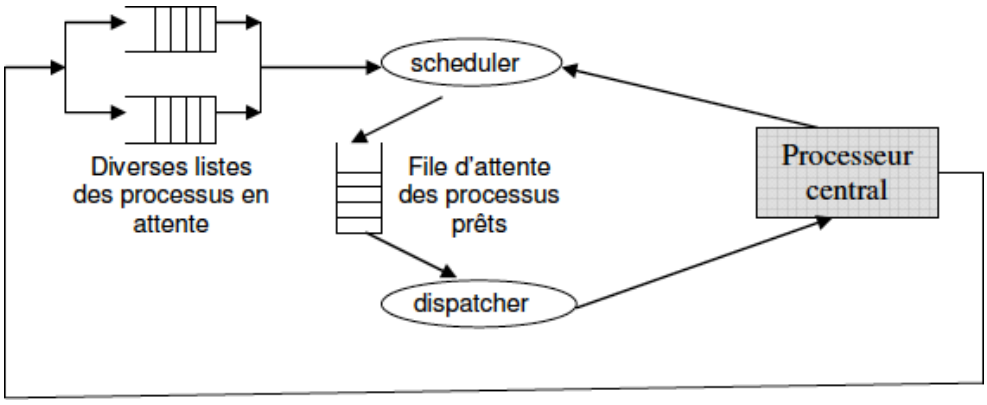
2. Objectifs d'un Ordonnanceur

Les objectifs d'un Ordonnanceur sont :

-Maximiser l'utilisation du processeur

-Présenter un temps de réponse acceptable

-Respecter l'équité entre les processus selon le critère d'ordonnancement utilisé.



3. Critères d'ordonnancement

L'objectif d'un algorithme d'ordonnancement consiste à identifier le processus qui conduira à la meilleure performance possible du système.

La politique d'ordonnancement détermine l'importance de chaque critère.

Utilisation de l'UC : Pourcentage de temps pendant lequel l'UC exécute un processus. L'importance de ce critère varie généralement en fonction du degré de partage du système.

Utilisation répartie : Pourcentage du temps pendant lequel est utilisé l'ensemble des ressources (outre l'UC, mémoire, périphérique d'E/S...)

Débit : Nombre de processus pouvant être exécutés par le système sur une période de temps donnée.

Temps de rotation : durée moyenne qu'il faut pour qu'un processus s'exécute. Le temps de rotation d'un processus comprend tout le temps que celui-ci passe dans le système. Il est inversement proportionnel au débit.

Temps d'attente : durée moyenne qu'un processus passe à attendre. Mesurer la performance par le temps de rotation présente un inconvénient : Le temps de production du processus accroît le temps de rotation ; Le temps d'attente représente donc une mesure plus précise de la performance.

Temps de réponse : Temps moyen qu'il faut au système pour commencer à répondre aux entrées de l'utilisateur.

Equité : degré auquel tous les processus reçoivent une chance égale de s'exécuter.

Priorités : attribue un traitement préférentiel aux processus dont le niveau de priorité est supérieur.

Un bon algorithme d'ordonnancement doit:

- Maximiser le taux d'utilisation de l'UC et le débit;
- Minimiser le temps moyen de traitement;
- Minimiser le temps moyen d'attente;
- Minimiser le temps de réponse.

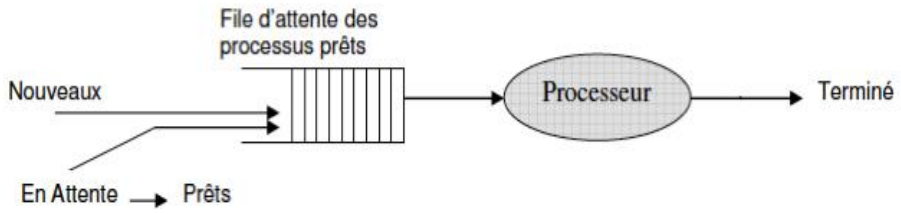
4. Algorithmes d'ordonnancement

Diagramme de Gantt

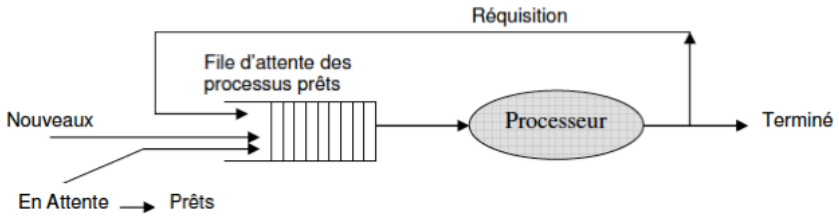
Représentation schématique de l'évolution dans le temps des processus.

Il existe 2 types d'ordonnancement :

Les algorithmes non-préemptifs (sans réquisition) empêchent l'appropriation du processeur par un processus avant la fin du processus courant



Les algorithmes préemptifs (avec réquisition) : possibilité d'appropriation du processeur par un processus avant la fin du processus courant.



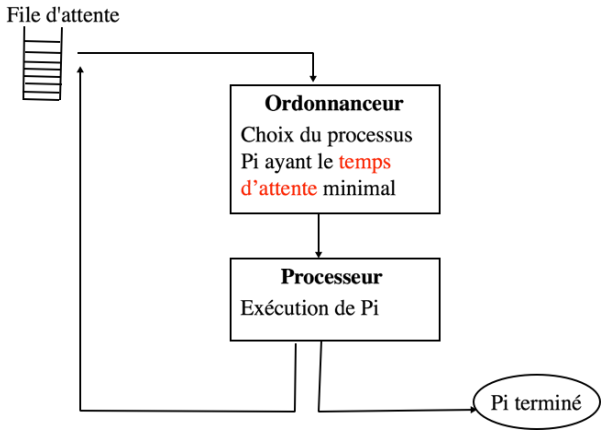
Algorithme non-préemptif

a. FIFO

L'organisation de la file d'attente des processus prêts est donc tout simplement du "**First In First Out".**

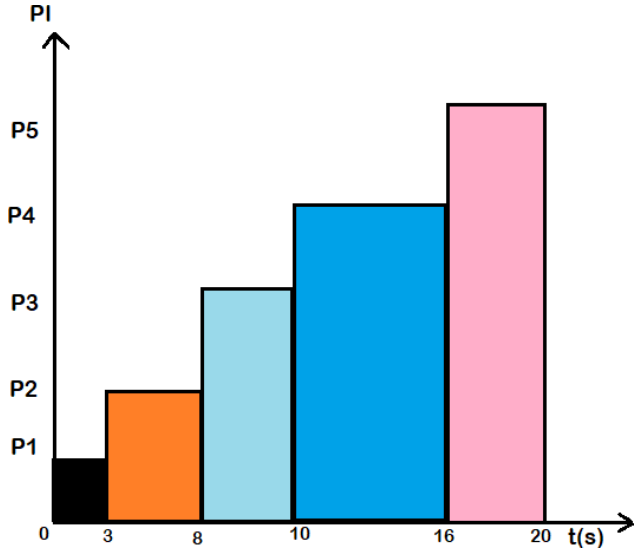
FIFO traite les processus dans l'ordre de leur soumission (date d'arrivée) sans aucune considération de leur temps d'exécution.

*L'algorithme **FIFO** consiste à choisir à un instant donné, le processus qui est depuis le plus longtemps dans la file d'attente, ce qui revient à choisir celui disposant du temps d'arrivée minimal et l'exécuter pendant un temps d'exécution bien définit. Ce procédé est répété jusqu'à épuisement des processus dans la file d'attente.



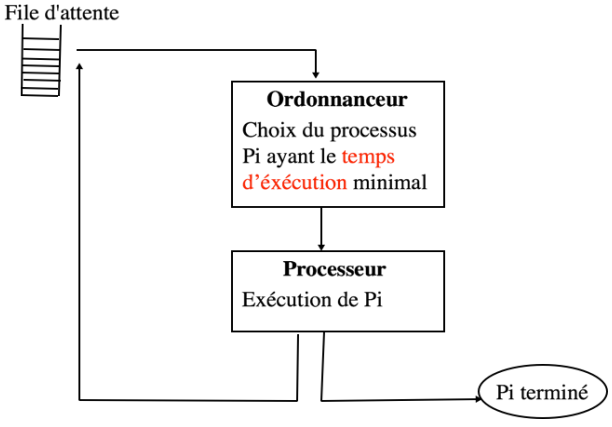
Exemple : FIFO

Processus	Durée d'exécution
P1	3
P2	5
P3	2
P4	6
P5	4



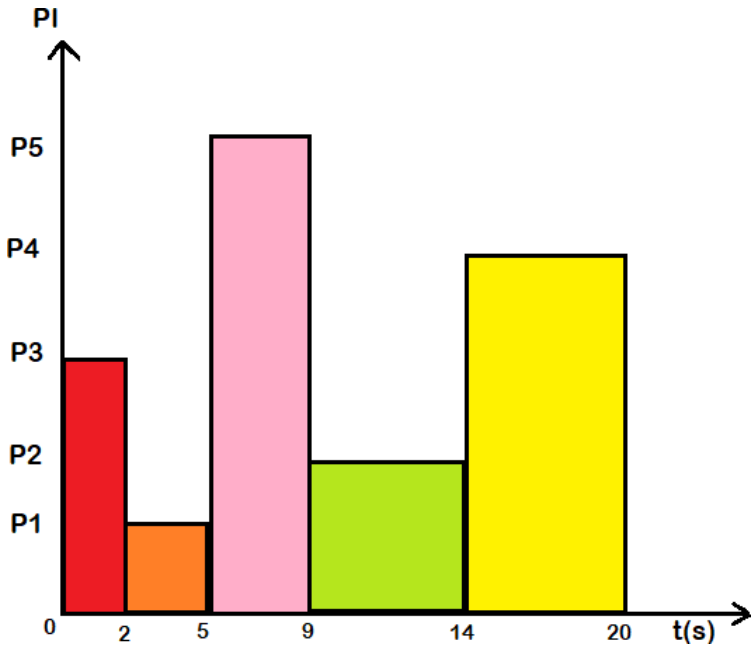
b. SJF

SJF: **S**hortest **J**ob **F**irst choisit de façon prioritaire les processus ayant **le plus court temps d'exécution** sans réellement tenir compte de leur date d'arrivée Ce procédé est répété jusqu'à épuisement des processus dans la file d'attente.



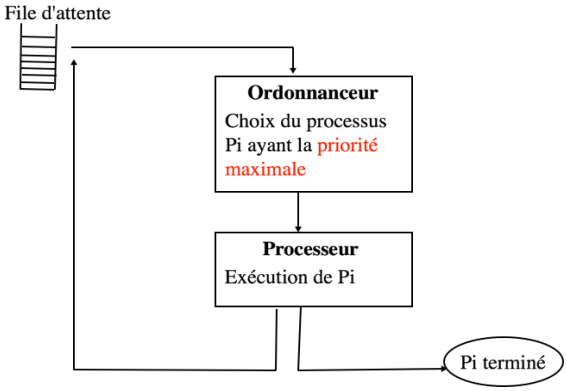
Exemple : SJF

Processus	Durée d'exécution
P1	3
P2	5
P3	2
P4	6
P5	4



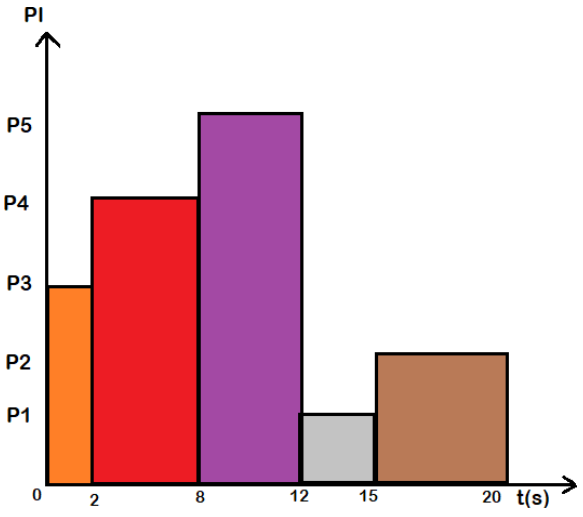
c. Algorithme basé sur la priorité

Les algorithmes fondés sur les priorités attribuées par le système d'exploitation aux processus choisissent les processus les plus prioritaires sans prise en considération d'une manière générale des données durée d'exécution et date d'arrivée des processus



Exemple :

Processus	Durée d'exécution	Priorité
P1	3	4
P2	5	4
P3	2	1
P4	6	2
P5	4	3



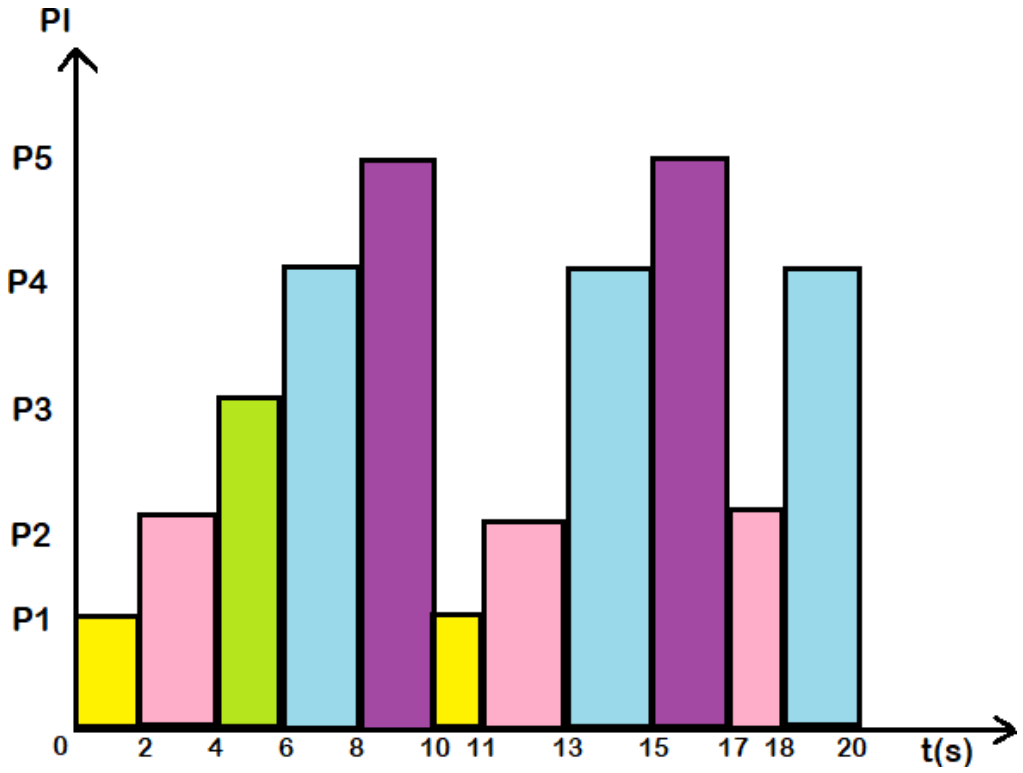
Algorithme préemptif

a. algorithme du tourniquet

Le Round Robin (RR) décrit une stratégie dite du tourniquet où on procède à un recyclage des processus sur le processeur tant que ceux-ci ne se sont pas terminés.

Lorsqu'un processus est élu, on lui attribue une tranche de temps fixe, appelé quantum, pendant laquelle il s'exécute. Au bout de ce temps, on ne poursuit plus l'exécution du processus, on lui retire donc le processeur et on le réinsère dans la file des processus prêts, il devra attendre sa prochaine élection. Ainsi , le processus se voit plusieurs tranches de temps avant d'atteindre sa terminaison attribuer successivement

Processus	Durée d'exécution
P1	3
P2	5
P3	2
P4	6
P5	4



Exemple :