

Ordonnancement des processus

Christophe Viroulaud

Terminale - NSI

Archi 02

Ordonnancement des processus

Christophe Viroulaud

Terminale - NSI

Archi 02

Les processus

Définition

Création d'un processus

Ordonnancement

Le chef d'orchestre

Le scheduling

Quelques algorithmes
d'ordonnancement

Un processeur ne peut exécuter qu'une seule instruction à la fois. Pourtant sur un ordinateur, il est possible d'écouter de la musique tout en surfant sur le web.

Un processeur ne peut exécuter qu'une seule instruction à la fois. Pourtant sur un ordinateur, il est possible d'écouter de la musique tout en surfant sur le web.

Comment exécuter plusieurs activités en même temps
sur une machine ?

Comment exécuter plusieurs activités en même temps
sur une machine ?

Sommaire

1. Les processus

Définition

1.1 Création d'un processus

2. Ordonnancement

À retenir

Un programme est un fichier en mémoire qui ne fait rien.
Un processus est l'exécution d'un programme.

Les processus - définition

À retenir

Un *programme* est un fichier en mémoire qui ne fait rien.
Un *processus* est l'exécution d'un programme.

Activité 1 :

1. Ouvrir un terminal.
2. Écrire la commande

1

top

Code 1 – Visualiser les processus en cours

3. Dans Debian, ouvrir le logiciel *Firefox* et observer l'apparition de nouveaux processus.
4. Depuis le terminal, utiliser la combinaison de touche **Ctrl+c** pour stopper la surveillance des processus.

Activité 1 :

1. Ouvrir un terminal.
2. Écrire la commande

1

top

Code 1 – Visualiser les processus en cours

3. Dans Debian, ouvrir le logiciel *Firefox* et observer l'apparition de nouveaux processus.
4. Depuis le terminal, utiliser la combinaison de touche **Ctrl+c** pour stopper la surveillance des processus.

Les processus

Définition

Création d'un processus

Ordonnancement

Le chef d'orchestre

Le scheduling

Quelques algorithmes
d'ordonnancement

- Ordonnancement des processus
 - Les processus
 - Création d'un processus
 - Création d'un processus

À retenir

Chaque processus possède un identifiant unique, le **PID** (**P**rocess **ID**entifier). Au démarrage de la machine un premier processus spécial (*init*) est lancé. Ce processus crée d'autres processus fils. Ainsi chaque processus possède un (et un seul) parent, le **PPID** (**P**arent **P**rocess **ID**entifier).

Création d'un processus

À retenir

Chaque processus possède un identifiant unique, le **PID** (**P**rocess **ID**entifier). Au démarrage de la machine un premier processus spécial (*init*) est lancé. Ce processus crée d'autres *processus fils*. Ainsi chaque processus possède un (et un seul) parent, le **PPID** (**P**arent **P**rocess **ID**entifier).

d'autres infos : nom de l'utilisateur qui a créé, utilisation CPU, état (STAT) :

- R en cours d'exécution.
- T processus stoppé.
- I processus endormi (>20s).
- S processus endormi (<20s).
- Z processus zombie.
- D processus non interruptible.
- W processus swappé (échangé) sur disque.

Activité 2 :

1. Afficher la liste de tous les processus :

1 `ps all`

2. Retrouver le PID du processus de Firefox. Tuer le processus avec l'instruction :

1 `kill numéro_PID`**Activité 2 :**

1. Afficher la liste de tous les processus :

1 `ps all`

2. Retrouver le PID du processus de Firefox. Tuer le processus avec l'instruction :

1 `kill numéro_PID`

Sommaire

1. Les processus

2. Ordonnancement

2.1 Le chef d'orchestre

2.2 Le scheduling

2.3 Quelques algorithmes d'ordonnancement

Ordonnancement des processus

└ Ordonnancement

└ Le chef d'orchestre

└ Le chef d'orchestre

1. chaque processus considère qu'il a le processeur pour lui tout seul.
multiprogrammation
2. scheduleur = 1 module du système d'exploitation

Le chef d'orchestre

Dans le système plusieurs processus sont en cours simultanément, mais le processeur ne peut exécuter qu'une seule instruction à la fois. Le processeur travaille donc en *temps partagé*. Il bascule constamment d'un processus à l'autre.

Le chef d'orchestre

Dans le système plusieurs processus sont en cours simultanément, mais le processeur ne peut exécuter qu'une seule instruction à la fois. Le processeur travaille donc *en temps partagé*. Il bascule constamment d'un processus à l'autre.

Ordonnancement
des processus

Les processus

Définition

Création d'un processus

Ordonnancement

Le chef d'orchestre

Le scheduling

Quelques algorithmes
d'ordonnancement

À retenir

L'**ordonnanceur** (**scheduleur**) sélectionne le prochain processus prêt (*Ready*) qui sera exécuté par le processeur. L'objectif est d'obtenir un *temps de traitement moyen* le plus court possible.

À retenir

L'**ordonnanceur** (**scheduleur**) sélectionne le prochain processus prêt (*Ready*) qui sera exécuté par le processeur. L'objectif est d'obtenir un *temps de traitement moyen* le plus court possible.

Sommaire

1. Les processus

2. Ordonnancement

2.1 Le chef d'orchestre

2.2 Le scheduling

2.3 Quelques algorithmes d'ordonnancement

Les algorithmes d'ordonnancement peuvent être classés en deux catégories :

- **Non pré emptif** : Sélectionne un processus, puis le laisse s'exécuter jusqu'à ce qu'il bloque (soit sur une E/S, soit en attente d'un autre processus) où qu'il libère volontairement le processeur.

Le scheduling

Les algorithmes d'ordonnancement peuvent être classés en deux catégories :

- **Non pré emptif** : Sélectionne un processus, puis le laisse s'exécuter jusqu'à ce qu'il bloque (soit sur une E/S, soit en attente d'un autre processus) où qu'il libère volontairement le processeur.

Les algorithmes d'ordonnancement peuvent être classés en deux catégories :

- ▶ **Non pré emptif** : Sélectionne un processus, puis le laisse s'exécuter jusqu'à ce qu'il bloque (soit sur une E/S, soit en attente d'un autre processus) où qu'il libère volontairement le processeur.
- ▶ **Pré emptif** : Sélectionne un processus et le laisse s'exécuter pendant un délai déterminé.

Le scheduling

Les algorithmes d'ordonnancement peuvent être classés en deux catégories :

- ▶ **Non pré emptif** : Sélectionne un processus, puis le laisse s'exécuter jusqu'à ce qu'il bloque (soit sur une E/S, soit en attente d'un autre processus) où qu'il libère volontairement le processeur.
- ▶ **Pré emptif** : Sélectionne un processus et le laisse s'exécuter pendant un délai déterminé.

Ordonnancement des processus

Ordonnancement

Quelques algorithmes d'ordonnancement

Sommaire

Sommaire

1. Les processus

2. Ordonnancement

2.1 Le chef d'orchestre

2.2 Le scheduling

2.3 Quelques algorithmes d'ordonnancement

Sommaire

1. Les processus

2. Ordonnancement

2.1 Le chef d'orchestre

2.2 Le scheduling

2.3 Quelques algorithmes d'ordonnancement

Ordonnancement des processus

Les processus

Définition

Création d'un processus

Ordonnancement

Le chef d'orchestre

Le scheduling

Quelques algorithmes d'ordonnancement

Ordonnancement des processus

Ordonnancement

Quelques algorithmes d'ordonnancement

Quelques algorithmes d'ordonnancement

Quelques algorithmes d'ordonnancement

- **First Come First Served** : Une fois que le CPU a été alloué à un processus, celui-ci le garde jusqu'à ce qu'il décide de le libérer.
- **Shortest Job First** : Quand le CPU est disponible, elle est assignée au processus qui possède le prochain cycle le plus petit.
- **Round Robin** : Chaque processus a une petite unité de temps appelée *quantum* (en général de 10 à 100 ms). L'ordonnanceur parcourt la file d'attente des processus prêts et alloue le CPU à chaque processus pendant un *quantum*.

1. **First Come** : Cet algorithme est particulièrement incommode pour le temps partagé où il est important que chaque utilisateur obtienne le CPU à des intervalles réguliers.
2. **Shortest** : La difficulté est pouvoir connaître la longueur de la prochaine requête du CPU.
3. La performance du *tourniquet* dépend fortement du choix du quantum de base.
4. Linux propose ces 3 politiques d'ordonnancement. Un système de priorité est également mis en place. Par défaut, un processus est associé à la politique de temps partagé. Seul root peut associer un processus à une des classes d'ordonnancement en temps réel.

Quelques algorithmes d'ordonnancement

- **First Come First Served** : Une fois que le CPU a été allouée à un processus, celui-ci le garde jusqu'à ce qu'il décide de le libérer.
- **Shortest Job First** : Quand le CPU est disponible, elle est assignée au processus qui possède le prochain cycle le plus petit.
- **Round Robin** : Chaque processus a une petite unité de temps appelée *quantum* (en général de 10 à 100 ms). L'ordonnanceur parcourt la file d'attente des processus prêts et alloue le CPU à chaque processus pendant un *quantum*.