SYS2

Système d'exploitation

M.Bastreghi (mba)

Haute École Bruxelles Brabant — École Supérieure d'Informatique

Année académique 2020 / 2021

ordonnanceur

L'ordonnanceur (scheduler) doit élire un nouveau processus.

Plusieurs processus sont dans l'état prêt.

Lequel choisir ?

ordonnanceur

L'algorithme de l'ordonnanceur doit être conçu pour prendre des décisions en fonction de différents points de vue :

- équité (votre voisin compile 10 fois plus vite ...)
- temps de réponse (le calcul de pi vous empêche de taper)
- temps d'exécution (chaque E/S vous pénalise)
- rendement/efficacité (90% du processeur est pour le scheduler)
- équilibre (toutes les parties du système sont utilisées)
- ...

ordonnanceur

L'ordonnanceur a un rôle très important dans un S.E. multitâche.

Privilégier un point de vue se fait au détriment d'un autre...

Pour les systèmes intéractifs on se souciera plus d'améliorer les **temps de réponse**

ordonnanceur

Compromis à trouver en tâchant que

- chaque processus ait sa part
- le processeur soit utilisé à 100%
- le temps de réponse soit minimum
- le nombre de travaux achevés/unité de temps soit maximum
- ...

Il n'existe pas de solution idéale pour toutes les situations : nous allons étudier certaines techniques...

tourniquet

Tourniquet

Chaque processus **prêt** reçoit, à tour de rôle, un quantum de temps processeur.

Un processus perd le CPU :

- parce qu'il a terminé
- parce qu'il a besoin d'une E/S (élu -> bloqué)
- parce qu'il a épuisé son quantum de temps

L'ordonnanceur élit le suivant de la liste...

tourniquet - quantum

Quantum

- si on utilise **5ms** pour échanger de contexte
- si il y a 10 processus prêts en même temps

quantum	temps max attente	efficacité CPU
5 ms	95 ms	$50\% = 5/10$
100 ms	+/- 1 sec	$95\% = 100/105$
1 sec	+/- 10 sec	$99{,}5\% = 1000/1005$

Souvent utilisé avec un compromis de 100 ms pour le quantum.

priorité

Priorité statique et famine

Tourniquet => on place les processus sur le même niveau mais certains processus sont plus importants que d'autres

=> associer une priorité aux processus

Le sheduler élit les processus par priorité décroissante

=> risque de **famine** pour petite priorité

prio dynamique(1)

Priorité dynamique

solution -1

-1 à chaque accès au processeur par exemple
Désavantage : on oublie sa priorité de départ

Idée suivante : on favorise les processus qui demandent beaucoup d'E/S en leur attribuant tout de suite le processeur.

prio dynamique(2)

solution f(quatum utilisé)

Un processus qui demande beaucoup d' E/S doit être favorisé (car les E/S se font en // avec le processeur et interactivité)

priorité = quantum / temps du quantum utilisé

exemples :

- un processus a un quantum de 100 ms et est bloqué par demande d'E/S après 5 ms –

files multiples ou classes

solution files multiples ou classes

- utilise beaucoup le CPU => performance augmente pour quantum grand.
- utilise beaucoup les E/S => meilleur temps de réponse pour quantum petit.

=> donner un grand quantum à un processus qui demande beaucoup de calcul

=> donner une priorité plus grande et un petit quantum à un processus qui demande beaucoup

classes (1)

Classe	Quantum	priorité	remarque
1	40 ms	5	pour un processus d'E/S
2	80 ms	4	...
3	160 ms	3	intermédiaire
4	320 ms	2	...
5	640 ms	1	pour un processus de calcul

On a, dans cet exemple, 5 tourniquets, un par classe.
Tous les processus prêts de chacune des classes reçoivent un quantum.
Un processus de classe inférieure est élu quand aucun processus d'une classe supérieure n'est prêt.

classes (2)

Comment connaître la classe d'un processus ?
En la fixant au départ.
Désavantage : tous vont se mettre en classe 1 !
En faisant évoluer : au départ : classe 1 et chaque fois que le processus consomme son quantum, il descend de classe.

- Les processus avec beaucoup d'E/S restent en classe 1
- Les processus avec beaucoup de CPU ont des grands quanta
- Un processus qui demande beaucoup d'E/S après un long calcul n'est pas à sa place

classes (3)

- Un processus qui demande beaucoup d'E/S après un long calcul n'est pas à sa place

Pour remédier à cet inconvénient : à chaque E/S, le processus remonte en classe 1

mais on peut tricher ...

classes (3)

- Un processus qui demande beaucoup d'E/S après un long calcul n'est pas à sa place

Pour remédier à cet inconvénient : à chaque E/S, le processus remonte en classe 1

exercice théorique

exercice

Des mesures sur un système ont montré que les processus s'exécutent en moyenne pendant T secondes avant de se bloquer sur une E/S. Une commutation de processus se fait en S secondes. Pour un ordonnancement circulaire avec un quantum de Q secondes, donnez l'efficacité du processeur si (a) Q = infini, (b) Q>T, (c) S<Q<T, (d) Q=S, (e) Q=0.

questions

questions

- L'ordonnancement du tourniquet utilise le plus souvent un quantum fixe. Donnez un argument en faveur d'un petit quantum et un argument en faveur d'un grand quantum.
- soit un quantum de 92ms et un temps d'ordonnancement de 8ms calculez l'efficacité du processeur.

questions

questions

- Donner une haute priorité à un processus favorise son temps de réponse [V-F]
- Avec un ordonnancement par tourniquet un processus s'interrompt pour une seule raison [V-F]
- un ordonnanceur à priorités statiques améliore le temps de réponse de tous les processus [V-F]

Questions ?

remerciements

remerciements à P.Bettens et M.Codutti
pour la mise en page :-) Mba

Crédits

Ces slides sont le support pour la présentation orale de l'activité d'apprentissage **SYS2** à la HE2B-ÉSI

Crédits Crédits

La distribution opensuse
du système d'exploitation GNU Linux.
LaTeX/Beamer comme système d'édition.
GNU make, rubber, pdfnup, ... pour les petites tâches.

Images et icônes

deviantart, flickr, The Noun Project