TD2 – **Ordonnancement de processus**

{nollinge,esposito,godard}@cmi.univ-mrs.fr

13 octobre 2004

☞ *Ce TD est consacré à la notion de processus et plus particulièrement à leur ordonnancement. Après avoir proposé une mise en oeuvre des processus et de leurs changements de contextes sur Ant-32, on expérimente et étudie plusieurs politiques d'ordonnancement vues en cours.*

1 Mise en oeuvre des processus

Dans cette partie, on considère un système d'exploitation fictif qui fonctionne comme décrit dans le précédant TD.

1.1. Énumérer les évènements qui peuvent engendrer un changement de contexte. Dans quel mode se trouve alors le processeur ?

1.2. Quels sont les opérations à réaliser lors d'un changement de contexte ? Quels sont celles qui sont dépendantes de l'architecture ?

1.3. Proposer un squelette, pour la partie dépendante de l'architecture, du changement de contexte (sur l'architecture de référence).

2 Traitement par lots

2.1. Politique d'ordonnancement « Premier arrivé, premier servi ! »

(a) Proposer un algorithme correspondant à cette politique.

(b) Traiter l'exemple suivant : $I(P_1) = 8$, $I(P_2) = 2$, $I(P_3) = 5$, $I(P_4) = 2$, $I(P_5) = 1$. Calculer la capacité, le temps de traitement moyen et le temps d'attente moyen.

2.2. Politique d'ordonnancement « Le plus court d'abord ! »

(a) Proposer un algorithme correspondant à cette politique.

(b) Traiter l'exemple suivant : $I(P_1) = 8$, $I(P_2) = 2$, $I(P_3) = 5$, $I(P_4) = 2$, $I(P_5) = 1$. Calculer la capacité, le temps de traitement moyen et le temps d'attente moyen.

2.3. L'algorithme précédant est optimal pour le temps d'attente moyen :

(a) Montrer qu'une transposition « descendante » fait diminuer le temps d'attente moyen ;

(b) Montrer que toute permutation se decompose en un produit de transpositions descendantes (on pourra s'intéresser tout d'abord au problème du tri en place d'une liste) ;

(c) Conclure.

2.4. Montrer que cette optimalité n'est plus garantie dans le cas où certains processus se présentent en cours d'exécution (*i.e.* à des temps différents de 0).

3 Systèmes interactifs

3.1. Politique d'ordonnancement « Tourniquet »

(a) Proposer un algorithme correspondant à cette politique.

(b) Traiter l'exemple suivant : $\Delta = 3$, $I(P_1) = 8$, $I(P_2) = 2$, $I(P_3) = 5$, $I(P_4) = 2$, $I(P_5) = 1$. Calculer la capacité, le temps de traitement moyen et le temps d'attente moyen.

3.2. Politique d'ordonnancement « Files de priorité avec tourniquet »

(a) Proposer un algorithme correspondant à cette politique.

(b) Traiter l'exemple suivant : $\Delta = 3$, $I(P_1) = 8$, $I(P_2) = 2$, $I(P_3) = 5$, $I(P_4) = 2$, $I(P_5) = 1$ avec $p(P_1) = 2$, $p(P_2) = 1$, $p(P_3) = 2$, $p(P_4) = 3$, $p(P_5) = 2$.

(c) Montrer qu'il y a des risques de famine dans le cas où certains processus se présentent en cours d'exécution (*i.e.* à des temps différents de 0).

3.3. On propose de mettre en place un mécanisme de vieillissement pour éviter les problèmes de famine. La priorité du processus actif est augmentée de 1 à chaque temps $p \leftarrow p + 1$. Tous les n temps, la priorité est transformée linéairement : $p \leftarrow \alpha p + \beta$.

(a) Discuter les valeurs possibles pour α et leurs effets.

(b) Comment β permet-il de mettre en place l'idée de priorité ?

(c) Montrer que pour des choix raisonnables de α, β et n, on évite les problèmes de famine.