UNIVERSITE SAINT-JOSEPH
FACULTE D'INGENIERIE
E.S.I.B.

Cursus Ingénieur Informatique et Réseaux
Semestre 4

TRAVAUX PRATIQUES DE PROGRAMMATION SYSTEME

Séance 6 – Durée 01H15

Une manière de déterminer les nombres premiers dans un intervalle est de découper l'espace de recherche en plusieurs sous-intervalles et d'affecter la recherche dans chaque intervalle à un processus créé par le père. Ainsi si on découpe l'intervalle *[1, N]* en *p* sous-intervalles, le processus père lance *p* fils et le fils *k* recherche l'intervalle $[kN/p + 1, (k + 1)N/p]$, $k = 0,...,p - 1$. Cette technique permet, si on dispose d'une machine équipée de *p* processeurs, de paralléliser la recherche (et d'essayer de retourner le résultat *p* fois plus vite). Elle a cependant l'inconvénient d'affecter des espaces de recherche dont les temps d'exploration sont inégaux. Ainsi le processus 0 terminera la recherche sur *[1, N/p]* bien avant le processus $p - 1$. Il (et donc un des processeurs de la machine) sera donc oisif jusqu'à la fin du programme. La charge de travail entre les processus (et donc les processeurs) est inégalement répartie.
Une solution (celle que vous devez programmer) pour répartir la charge de travail, consiste à utiliser un « pool » de *p* processus travailleurs à qui un processus maître affecte successivement des travaux de recherche de nombres premiers dans de petits intervalles (taille $T << N/p$). Quand un travailleur a fini sa recherche, le maître lui affecte la recherche dans un intervalle encore inexploré.

Pour votre implémentation:

1. le maître est le processus père;
2. les *proc* travailleurs sont des fils créés par le père
3. Les données partagées entre père et fils se trouvent dans un segment partagé.
4. Les données partagées sont les suivantes:
   a. limite maximale de l'intervalle des nombres à tester (Max),
   b. taille du petit intervalle (T),
   c. premier nombre non vérifié (p),
   d. Un tableau de *Max* éléments initialisé à 0 (Premier). Ce tableau sera utilisé pour savoir si le nombre est premier ou non.
   e. Un tableau de *proc* éléments initialisé à 0 (Occurrence). Ce tableau sera utilisé pour savoir combien de nombres premiers sont déterminés par chaque travailleur.
5. **L'algorithme** est le suivant:
   a. le père crée un ensemble de sémaphores ainsi qu'un segment partagé.
   b. le père initialise les deux mécanismes IPC précédents.
   c. le père crée *proc* processus qui vont jouer le rôle de travailleurs et attend la fin de tous ses fils.

d. Tant que p est inférieure à Max, chaque fils lit la valeur de *p* (qui va jouer le rôle de la borne inférieure du petit intervalle, ajoute *T* à cette valeur. *p+T-1* est la borne supérieure du petit intervalle (attention aux cas limites).
e. Chaque travailleur détermine, pour chaque nombre de l'intervalle *[p , p+T-1]*, s'il est premier ou non.
f. Pour chaque nombre premier trouvé, le travailleur met à jour l'entrée correspondante dans le tableau *Premier* et incrémente l'entrée qui lui correspond dans le tableau *Occurrence*.
g. Lorsque tous les fils sont terminés, le père affiche les nombres premiers trouvés ainsi que le résultat de chaque travailleur.
h. N'oubliez pas de libérer et de détruire toutes les ressources à la fin du programme.

Travail demandé:

1. Quel genre de sémaphores faut-il utiliser? Pourquoi?
2. Implémentez cet algorithme en langage C.
   Les variables *Max*, *T* et *proc* doivent être fournies sur la ligne de commande.
3. Est-ce que l'algorithme est équitable? Justifiez.