Parallélisation à l'aide de processus et de threads

Exercice 5.1 Il s'agit de compléter le code de la commande "`hello-goodbye` *`entier`*" qui lance en parallèle *entier* threads effectuant la fonction `HelloGoodbye()`.

1. Faire une version préliminaire qui se contente de lancer un seul thread.
2. Faire une version qui lance le nombre de threads correspondant à l'entier passé en paramètre.
3. Faire en sorte de numéroter les threads à partir de 0, chaque thread affichera son numéro à la place de son tid.
4. En utilisant une barrière, faire en sorte que tous les affichage de type `"hello"` se déroulent avant tous les affichages de type `"goodbye"`.

Exercice 5.2 La commande "`for-en-parallele x`" lance en parallèle `x` threads réalisant la boucle suivante :

```
for(unsigned long i=0; i < MAX; i++)
  k++;
```

1. Sachant que `k` est une variable globale initialement nulle, quelle(s) valeur(s) peut-elle avoir à la suite de l'exécution de `x` threads ?
2. Conforter votre intuition en exécutant plusieurs fois le programme `for-en-parallele` tout en faisant varier le nombre de threads. Quelle valeur minimale peut avoir `k` ?
3. Comment faire en sorte que toutes les incrémentations de `k` soient prises en compte ?

Exercice 5.3 Le module `distibuteur` implémente un distributeur d'entier. Proposer une version multithreadée de ce programme.

Exercice 5.4 Il s'agit d'apporter une solution multithreadée au programme `suivre-commandes.c` [1]. L'objectif est de suivre *au plus près* l'évolution des processus fils en délégant le traitement des zombies à un thread auxilaire. Pour simplifier le utiliser le mécanisme de déroutement des signaux. On traitera soigneusement l'accès aux variables partagées.

Exercice complémentaire

Exercice 5.5 Il s'agit de paralléliser la résolution du problème des n-reines dont l'énnoncé est le suivant : *«soit un échiquier de dimensions $N \times N$, combien y'a-t-il configurations où N reines sont disposées sur cet échiquier sans qu'aucune d'elles ne se menace.»* Une version séquentielle est disponible dans le programme `nreine.c`, cette version est basée sur un algorithme récursif qui passe en revue toutes les solutions possibles en progressant ligne par ligne.
Pour paralléliser à l'aide de threads ce calcul il s'agit de répartir la charge de calcul entre les différents threads, chaque thread travaillant sur son propre échiquier.
Proposer une solution en créant autant de threads qu'il y a de colonnes et en initialisant l'échiquier du k-ième thread en posant une reine sur la k-ième case de la première ligne ;

1. le code a été remodelé depuis le TP sur les signaux afin de mettre en valeur le temps mis pour prendre en compte la mort du processus fils