Travaux Dirigés n°9

Cours de Parallélisme

—Maîtrise Informatique—

Ce TD est une étude pratique du tri d'entiers en environnement distribué. Nous étudions la portabilité de l'algorithme de tri "pair-impair" sur un réseau de stations.

▸ **Exercice 1.** (tri pair-impair séquentiel) *Cet exercice est une étude pratique de l'algorithme de tri "pair-impair". Le répertoire* `/tmp/PVM/odd-even/seq` *contient le fichier* `odd-even_seq.c` *qui implémente cet algorithme. Calculer les temps d'exécution séquentielle du programme sur quelques milliers d'entiers à trier et tracer la courbe pratique.*

▸ **Exercice 2.** (tri pair-impair distribué) *Cet exercice est une étude des possibilités pratiques de distribution du tri pair-impair.*

1. *Le répertoire* `/tmp/PVM/odd-even/dis` *contient des sources incomplets d'une version distribuée sur $p \ll n$ processeurs de l'algorithme de tri "pair-impair" de n entiers. Le fichier* `odd-even_master.c` *initie le calcul en créant les différents processus alors que le fichier* `odd-even_slave.c` *contient l'algorithme de tri. La version qui est donnée ne permet de diviser le calcul qu'en 2 sous-calculs. Compléter le fichier* `odd-even_slave.c` *afin que le programme* `odd-even_master` *puisse couper le calcul en p sous-calculs réalisés par p processus.*
2. *Comparer les temps d'exécution avec la version séquentielle.*
3. *Expliquer le comportement du programme en pratique.*

▸ **Exercice 3.** (trier en distribué) *Cet exercice est la mise en application de tout ce que nous avons mis en évidence à l'aide des différents exercices précédents, au sujet de la distribution efficace d'un algorithme de tri.*

1. *Programmer l'algorithme de tri proposé précédemment, algorithme qui limite le nombre de messages échangés en s'appuyant sur le tri* quick-sort*. Pour cela, on mettra en place un des algorithmes proposés précédemment pour fusionner des tableaux triés.*
2. *Regarder les temps de calcul obtenus et expliquer le comportement du programme.*
3. *Est-il possible de complètement faire disparaître la phase de fusion des tableaux triés distribués sur la machine virtuelle ? Si oui, comment faire et que deviennent les temps théoriques d'exécution ?*