8INF856
Programmation sur architectures parallèles
Devoir 3

(À remettre au plus tard mercredi le 27 novembre 2019)

INSTRUCTIONS

- Les étudiants doivent travailler en équipe de 2 ou 3.
- Vous serez évalué pour la rectitude et la qualité de vos réponses.
- Les programmes des questions 1 et 2 devront être mis dans votre répertoire maison sur `dim-openmpi0.uqac.ca`
- Le programme de la question 1 devra être mis dans le fichier d3-1.c et celui de la question 2 devra être mis dans le fichier d3-2.c.
- Les deux programmes devron pouvoir être exécuté sans erreur à l'aide de la commande `mpicc`.
- Vous devez me faire parvenir un rapport écrit contenant les informations suivantes:
  1. Le nom des coéquipiers
  2. Le nom d'usager et le mot de passe du compte ou se trouve le programme
  3. Une explication détaillée de vos solutions
  4. Une analyse du temps d'exécution de vos solutions en fontions de la taille de l'entrée et du nombre de processus utilisés.

1. Vous devez d'abord concevoir un programme MPI en C qui reçoit sur la ligne de commande un entier N et qui calcule le nombre de couples de nombres premiers jumeaux plus petits ou égaux à N. On dit que deux nombre premier p et q sont jumeaux si $q = p + 2$. Par exemple si $N = 15$ alors il y a trois couples: $(3, 5), (5, 7)$ et $(11, 13)$. Votre programme doit équilibrer du mieux possible la charge de travail sur les différents noeuds du cluster.

2. Dans cette question, vous devez modifier l'exemple du diaporama `Send et receive` qui apparaît dans la section MPI de la page Web du cours. Cet exemple est un programme MPI en C qui utilise l'algorithme de Floyd pour trouver la longueur du chemin le plus court entre toutes les paires de noeuds. Le programme actuel fait un appel à la fonction `read_row_striped_matrix` afin de lire la matrice d'adjacence du graphe et de la distribuer à l'aide des fonction `MPI_Send` et `MPI_Recv`. Vous devez modifier ce programme (ou le réécrire, si vous le jugez préférable) de la façon suivante: Le processus 0 doit lire la matrice MAT et la mettre dans un tableau. Tous les processus doivent utiliser `MPI_Get` pour obtenir les lignes de la matrice qui leurs sont assignées par `BLOCK_LOW` et `BLOCK_HIGH`. À la fin de l'algorithme, les processus doivent aller modifier la matrice MAT du processus 0 à l'aide de la fonction `MPI_Put` afin qu'elle contienne la longueur du plus court chemin entre toutes les paires de noeuds. Finalement, le processus 0 affiche la matrice MAT.