Travaux Pratique 10

Ce sujet est en lien avec le cinquième chapitre du cours, et concerne la programmation répartie. Les mêmes commentaires que ceux des derniers TP s'appliquent ici aussi.

Dans ce sujet, vous poursuivez votre découverte enchantée de MPI en utilisant une topologie en **anneau unidirectionnel avec p processeurs** au total, numérotés depuis 0. Ainsi, chaque processeur de numéro q peut envoyer un message au processeur de numéro $q + 1$ (modulo p), et recevoir du processeur $q - 1$ (modulo p).

Toute communication vers un processeur qui n'est pas le suivant, ou depuis un processeur qui n'est pas le précédent est strictement interdite … Pour éviter les erreurs, **utilisez un `OPP::MPI::Ring`** !

Votre travail est à rendre (il le sera à chaque fois) sous la forme du code (et uniquement la partie « student ») accompagnée d'un rapport au format PDF, le tout dans une archive compressée au format ZIP. Ne respecter pas ces contraintes, et votre note en sera diminuée de quelques points.

Vous ne devez modifier que ce qui se trouve dans le répertoire « `./student/` » !

Notez que chaque exercice vient avec un squelette, qui s'occupe de la ligne de commande, du lancement de votre code (défini dans une classe particulière), et d'une vérification sommaire du résultat (lorsque c'est possible). Utilisez l'option « `-h` » ou « `--help` » pour connaître le fonctionnement de la ligne de commande …

Exercice 1 : Résolution d'un système triangulaire

Dans cet exercice, vous utiliserez une topologie en **anneau unidirectionnel** de p processeurs. Autrement dit, chaque processeur de numéro q peut envoyer un message au processeur de numéro $q+1$, et recevoir du processeur $q-1$.

Réalisez la résolution du système $L \cdot X = B$, dans lequel la matrice L est triangulaire inférieure de taille $n \times n$, le vecteur B est donné, et le vecteur X est inconnu (c'est lui qu'il faut calculer).

Pour cela vous utiliserez une distribution de L par **bloc de lignes**, et de B et X par **bloc**. Ainsi, cela diffère quelque peu de l'exercice 1 du TD 3.

Notez que vous sont fournies les classes permettant d'accéder aux éléments de L, B et X (en lecture et en écriture), avec la distribution désirée (en gros, elles chargent depuis un fichier les données attendues, par bloc de lignes pour la matrice et par bloc de valeurs pour les vecteurs).

Exemple d'utilisation :

```
TP10\correction> mpiexec -np 20 .\build\Debug\Resolution.exe -e 15
*** Number of Processors: 20 ***
Build matrix L(32768,32768) ...
Build vector B(32768) ...
Solve the system ...
check the result ...
Well done, it seems to work!
```

Exercice 2 : Le retour du seigneur

Dans cet exercice, vous utiliserez une topologie en **anneau unidirectionnel** de p processeurs. Autrement dit, chaque processeur de numéro q peut envoyer un message au processeur de numéro $q+1$, et recevoir du processeur $q-1$.

Ecrire un algorithme de multiplication d'une matrice A de taille $m \times n$ par un vecteur X de taille n. Ici aussi les données sont **distribuées par bloc** (de lignes ou de valeurs).

Les données sont sauvegardées dans des fichiers dont les noms sont donnés en ligne de commande. Cela permettra notamment de vérifier le résultat avec la version séquentielle, gracieusement fournie (programme `Check`).