Laboratoire 3

Simulation de chaleur

Écrit par Gabriel-Andrew Pollo-Guilbert

Idée original de Francis Giraldeau

Pour helene.jiang@polymtl.ca

Pour francois-philippe.ossim-belias@polymtl.ca

Lorsque la résolution d'un problème demande beaucoup plus de ressources que ce qu'un seul ordinateur peut fournir, il est souvent idéal de distribuer le calcul sur plusieurs machines, communément appelé superordinateur ou une grappe de calcul de nos jours. Les simulations scientifiques (par exemple le repliement des protéines, la dynamique des fluides et autres) font souvent appel à des superordinateurs pour accélérer le calcul.

Ce laboratoire a pour but de vous familiariser avec la technologie Message Passing Interface (MPI) développée et utilisée par et pour le domaine du calcul haute performance. Pour ce faire, vous allez complèter la communication entre les noeuds d'une simulation de chaleur distribuée.

Code

- `source/main.c`
  - Contient le point d'entrée du programme qui traite les arguments en ligne de commande et démarre la simulation de chaleur.
- `source/image.c` `include/image.h` `source/pixel.h` `source/color.h` `include/color.h`
  - Contiennent les structures et le code permettant la lecture/écriture d'images de format PNG.
- `source/grid.c` `include/grid.h`
  - Contiennent la grille de calcul utilisée par la simulation de chaque noeud.
- `source/cart.c` `include/cart.h`
  - Contiennent une structure de données permettant de diviser le problème initial en plusieurs grilles de calcul et de fusionner les grilles résultantes afin de créer l'image finale.
- `source/heatsim.c` `include/heatsim.h`
  - Contiennent la simulation de chaleur utilisant la méthode Runge-Kutta.
- `source/heatsim-mpi.c` **(À COMPLÉTER)**
  - Contient la communication MPI demandée de la simulation de chaleur.
- `scripts/*`
  - Contiennent des scripts pour exécuter la simulation sur la grappe de calcul.

Algorithme

La simulation de diffusion de chaleur s'effectue sur une image en 2D. Pour paralléliser la simulation, on divise cette image en N×M blocs (possible de taille différente). La diffusion de chaleur à chaque itération est calculée localement pour chaque bloc. Pour chaque noeud, l'algorithme est le suivant:

1. Initialisation du noeud MPI (**À COMPLÉTER**) — `heatsim_init`.
2. Chargement de l'image et division en sous-grilles.
    - Le rang 0 envoit les grilles aux autres rangs (**À COMPLÉTER**) — `heatsim_send_grids`.
    - Les autres rangs recoivent leur grille du rang 0 (**À COMPLÉTER**) — `heatsim_receive_grid`.
3. Pour chaque itération, les noeuds calculent localement la diffusion de chaleur et échange les bordures de leurs grilles.
    - Chaque rang envoit et reçoit les 4 bordures de leur grille à leurs 4 voisins (**À COMPLÉTER**) — `heatsim_exchange_borders`.
4. Fusion des sous-grilles en une seule et enregistrement de l'image résultante.
    - Le rang 0 recoit les grilles des autres rangs (**À COMPLÉTER**) — `heatsim_receive_results`.
    - Les autres rangs envoient leur grille au rang 0 (**À COMPLÉTER**) — `heatsim_send_result`.

L'information spécifique à chaque étape est écrite en commentaire dans `heatsim-mpi.c`.

Spécifications

Les spécifications décrites dans cette section diffèrent d'une équipe à une autre.

L'envoit/réception des grilles initiales doit être effectué avec `MPI_Isend` et `MPI_Irecv`. Vous devez envoyé les paramètres `width`, `height` et `padding` en une seule requête de type définit avec `MPI_Type_struct`. Les données (`data`) de la grille doivent être envoyé en une seule requête de type définit avec `MPI_Type_struct`.

L'échange des bordures doit être effectué avec `MPI_Isend` et `MPI_Irecv`. Les bordures nord et sud doivent être de type `MPI_DOUBLE`. Les bordures est et ouest doivent être de type définit avec `MPI_Type_vector`.

L'envoie et la réception de la grille finale doit être effectué avec `MPI_Send` et `MPI_Recv`. Les données (`data`) de la grille doit être de type définit avec `MPI_Type_struct`.

- La compilation ne doit pas lancer d'avertissements.
- Le programme ne doit pas avoir de fuite de mémoire durant son exécution autre que MPI.

Compilation

Pour compiler l'application, il est recommandé de créer un dossier `build/` à la racine du projet afin de bien séparer les fichiers générés.

```
$ mkdir build && cd build
```

Ensuite, on configure le projet avec `cmake`. Celui-ci peut donc être compilé avec `make`.

```
$ cmake ..
$ make
```

Il n'est pas nécessaire de re-exécuter toutes les commandes ci-dessus pour recompiler le binaire, seulement la dernière. Vous pouvez exécuter `./heatsim --help` pour voir les options du programme.

La configuration de base du projet compile le programme avec Address Sanitizer afin d'inspecter les fautes d'accès mémoires. Il est possible de désactiver ce comportement en ajoutant `-DCMAKE_BUILD_TYPE=Release` à la commande `cmake`.

Commandes

Le `Makefile` généré par `cmake` contient les commandes spéciales ci-dessous.

- `make format`
  - Utilise `clang-format` pour formatter le code source.
- `make remise`
  - Crée une archive ZIP contenant les fichiers pour la remise.

Exécution Locale

Pour exécuter la simulation, il faut utiliser `mpirun`. Ce programme offert par MPI permet de démarrer plusieurs processus, localement ou à distance, chaque exécutant le même code, mais avec un rang différent. Sans aller dans les détails, on peut exécuter une simulation sur 16 noeuds répartis sur une grille 4×4 avec la commande suivante:

```
$ mpirun -n 16 ./heatsim --dim-x 4 --dim-y 4 --input ../image/earth-xlarge.png
--iterations 100
```

Seulement le canal rouge de l'image d'entrée est utilisé. Si le fichier de sortie n'est pas spécifier avec `--output`, la valeur par défaut sera le fichier d'entrée avec le suffix `.output.png`.

La taille de la grille (`-n`) doit être égale au nombre de noeud (`--dim-x` × `--dim-y`).

Exécution sur la Grappe de Calcul

En raison du long temps de calcul, exécutez seulement votre code sur la grappe lorsque vous êtes certain que celui-ci fonctionne localement pour plusieurs noeuds et dimensions différentes. Cela évite de monopoliser la grappe pendant que d'autres étudiants veulent l'utiliser. De plus, il est recommandé de ramasser les données quelques journées avant la remise afin de s'assurer que vos calculs aient le temps de s'exécuter. En cas de problème, contacter votre chargé.

Pour plus d'informations concernant l'exécution sur la grappe, suivez les instruction sur Moodle.