

## Exemples d'échanges de messages dans un code hybride MPI - OpenMP

Chacun des 3 codes crée 2 processus contenant 1 ou 2 threads chacun.

Le processus 0 envoie des messages vers le processus 1

**Dans la variante 1:**

- un message MPI depuis le thread principal du processus 0 vers le thread principal du processus 1 (dans une partie non multithreadée du code)
- un message MPI depuis le thread 0 du processus 0 vers le thread 0 du processus 1 et en même temps un message MPI depuis le thread 1 du processus 0 vers le thread 1 du processus 1

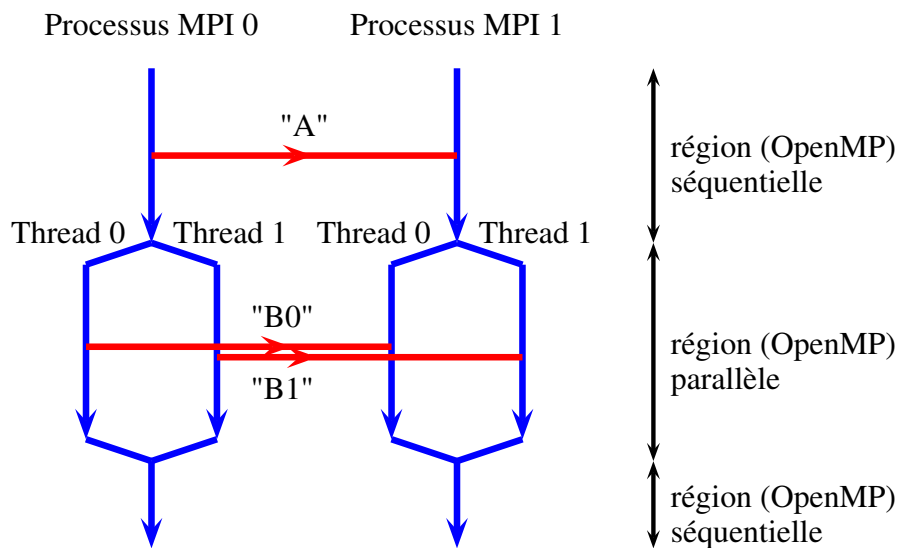


Figure 1: Exécution variante 1

**Dans la variante 2:**

- un message MPI depuis le thread principal du processus 0 vers le thread principal du processus 1 (dans une partie non multithreadée du code)
- un message MPI depuis le thread 0 du processus 0 vers le thread 1 du processus 1 et en même temps un message MPI depuis le thread 1 du processus 0 vers le thread 0 du processus 1

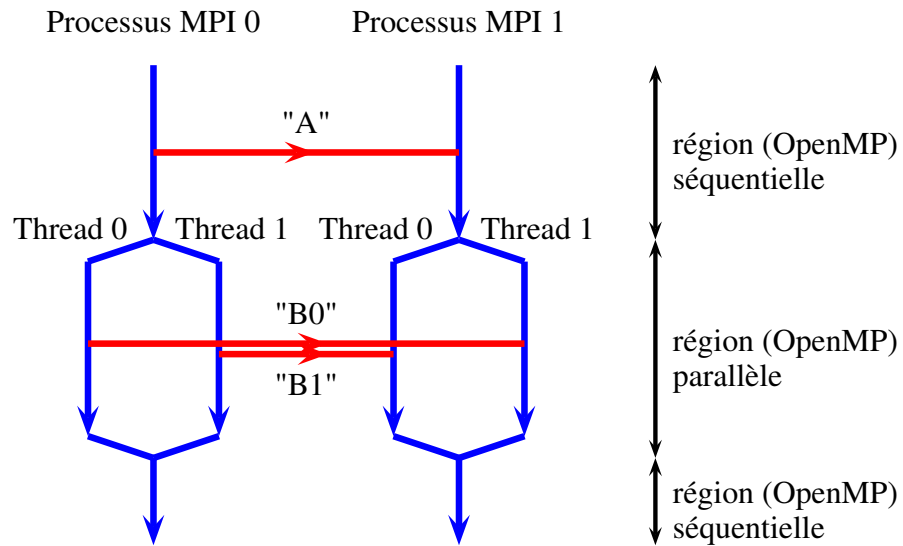


Figure 2: Exécution variante 2

### Dans la variante 3:

- un message MPI depuis le thread principal du processus 0 vers le thread principal du processus 1 (dans une partie non multithreadée du code)
- un message MPI depuis le thread principal du processus 0 vers le thread 0 du processus 1
- un message MPI depuis le thread principal du processus 0 vers le thread 1 du processus 1

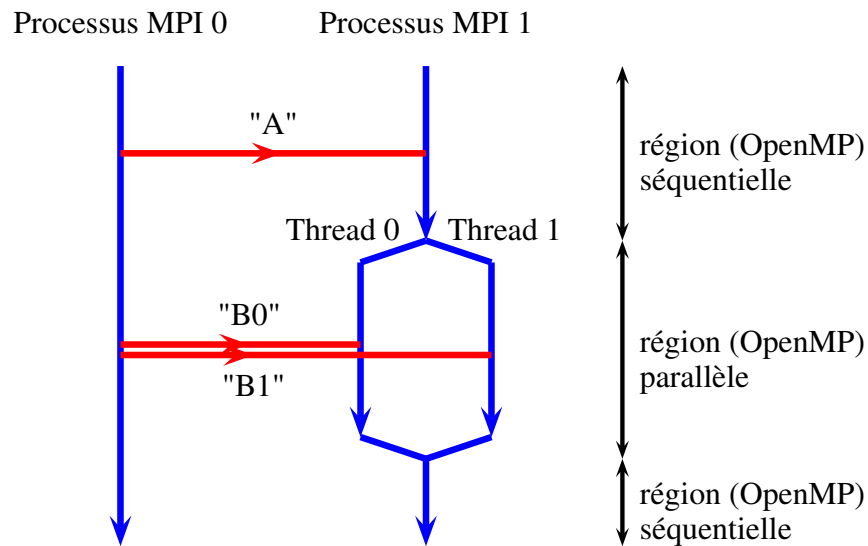


Figure 3: Exécution variante 3

Bien voir que ce sont les tags des messages qui permettent de spécifier le thread de départ et celui d'arrivée !

### Pour compiler et exécuter sur la machine Cholesky:

```
sbatch run.sh
```

### Pour compiler et exécuter sur une machine locale:

```
./run.sh
```

### Voir les résultats

Les affichages sont dans les fichiers out\_0\_on\_2.txt (affichage du processus 0) et out\_1\_on\_2.txt (processus 1).

Taper la commande

```
paste out_0_on_2.txt out_1_on_2.txt
```

qui affiche le contenu de ces 2 fichiers côte à côte pour mieux les comparer.