Rapport – Travaux Pratique 9

Sommaire

Exercice 1

```
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 4 .\win32\Release\PremierPas.exe
Hello world from processor LAPTOP-8LVQVU34, rank 0 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 1 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 3 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 2 out of 4 processors
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 4 .\win32\Release\PremierPas.exe
Hello world from processor LAPTOP-8LVQVU34, rank 3 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 0 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 2 out of 4 processors
Hello world from processor LAPTOP-8LVQVU34, rank 1 out of 4 processors
```

L'application *PremierPas* vérifie l'aléa temporel de MPI.

Exercice 2

```
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 2 .\win32\Release\Communication.exe
Processor 0 init data
Processor 0 send 42 to the next
Processor 1 receive
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 4 .\win32\Release\Communication.exe
Processor 0 init data
Processor 0 send 42 to the next
Processor 1 receive
Processor 1 send 42 to the next
Processor 2 receive
Processor 2 send 42 to the next
Processor 3 receive
```

Ce programme effectue une communication en anneau entre les processeurs, ces processeurs se communiquent la réponse à la grande question sur la vie, l'univers et le reste. C'est le processeur 0 qui envoie la données jusqu'au processeur n – 1 avec n le nombre de processeur.

Exercice 3

```
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 4 .\win32\Release\NaiveBroadcast.exe -e 20
Broadcast from 0 an array of 1048576 integers ...
[0] Done in 6.8673ms.
[3] Done in 7.0046ms.
[2] Done in 6.8875ms.
[1] Done in 7.0424ms.
[1] Good, your broadcast seems to work ;-)
[3] Good, your broadcast seems to work ;-)
[0] Good, your broadcast seems to work ;-)
[2] Good, your broadcast seems to work ;-)
Broadcast from 1 an array of 1048576 integers ...
[1] Done in 7.2819ms.
[0] Done in 7.3014ms.
```

Pour l'envoi de 1 000 000 de données en communication synchrone, il faut environ 7ms par processeur.

```
(base) PS C:\Users\sauze\OneDrive - Université de Poitiers\Documents\2021-2022_M1_IM_S8\APR\TP
9> mpiexec -n 4 .\win32\Release\PipeBroadcast.exe -en 20 -em 10
Broadcast from 0 an array of 1048576 integers ...
[1] Done in 32.6567ms.
[0] Done in 32.7525ms.
[3] Done in 32.5634ms.
[2] Done in 32.7917ms.
[0] Good, your broadcast seems to work ;-)
[2] Good, your broadcast seems to work ;-)
[1] Good, your broadcast seems to work ;-)
[3] Good, your broadcast seems to work ;-)
Broadcast from 1 an array of 1048576 integers ...
[1] Done in 37.1297ms.
[3] Done in 36.7699ms.
```

Pour l'envoi de 1 000 000 de données par paquet de 1024, il faut environ 32ms par processeur.

En testant avec d'autres valeurs pour les deux variantes du broadcast, le broadcast en pipeline est toujours plus lent que la version naïve, cependant plus au augmente le nombre de données à envoyer, moins cette différence se fait ressentir.