

Université des Sciences et de la Technologie Houari Boumediene
Faculté d'Informatique



ACP

TP3 : modèle de programmation à mémoire distribué (MPI)

Fait par :

Nom et prénom : ABDELMALEK BENMEZIANE

Matricule : 171731046778

Spécialité : M2 BIOINFO

Section : A

Contents

1	Exercice 1 (TD) : inter blocages de communication	1
1.1	Deadlock 1	1
1.2	Deadlock 2	1
2	Exercice 2	2
3	Exercice 3	3
4	Exercice 4	4
5	Exercice 5 : (bcast et gather) calcul pi	5

List of Figures

3.1	Output	3
4.1	Output	4
4.2	Output	4
5.1	Output	5

Chapter 1

Exercice 1 (TD) : inter blocages de communication

1.1 Deadlock 1

Il existe une situation d'interblocage au rank 0 et rank 1 à l'instruction **MPI_Recv**, la solution proposée est :

```
1 if(rank==0){
2 MPI_Send(&sendbuf, count, MPI_Float, 1, tag, MPI_COMM_WORLD);
3 MPI_Recv(&recvbuf, count, MPI_Float, 1, tag, MPI_COMM_WORLD, &status);
4 else if(rank==1){
5 MPI_Recv(&recvbuf, count, MPI_Float, 0, tag, MPI_COMM_WORLD, &status);
6 MPI_Send(&sendbuf, count, MPI_Float, 0, tag, MPI_CMM_WORLD, &status);
7 }
```

1.2 Deadlock 2

Il existe une situation d'interblocage au rank 0 et rank 1 à l'instruction **MPI_Ssend**, la solution proposée est :

```
1 if(rank==0){
2 MPI_Ssend(&sendbuf, count, MPI_Float, 1, tag, MPI_COMM_WORLD);
3 MPI_Recv(&recvbuf, count, MPI_Float, 1, tag, MPI_COMM_WORLD, &status);
4 else if(rank==1){
5 MPI_Recv(&recvbuf, count, MPI_Float, 0, tag, MPI_COMM_WORLD, &status);
6 MPI_Ssend(&sendbuf, count, MPI_Float, 0, tag, MPI_CMM_WORLD, &status);
7 }
```

Chapter 2

Exercice 2

Le code suivant ne contient aucune situation d'interblocage.

```
1 MPI_Comm_rank (comm, &myRank) ;
2 if(myRank == 0){
3 MPI_Send(sendbuf1 , count , MPI_INT, 2 , tag , comm);
4 MPI_Send(sendbuf2 , count , MPI_INT, 1 , tag , comm);
5 }else if(myRank == 1){
6 MPI_Recv(recvbuf1 , count , MPI_INT, 0 , tag , comm, &status);
7 MPI_Send(recvbuf1 , count , MPI_INT, 2 , tag , comm);
8 }else if(myRank == 2){
9 MPI_Recv(recvbuf1, count, MPI_INT, MPI_ANY_SOURCE, tag, comm, &status);
10 MPI_Recv(recvbuf2, count, MPI_INT, MPI_ANY_SOURCE, tag, comm, &status);
11 }
```

Chapter 3

Exercice 3

Le code source est dans le fichier "exercice3.py".

```
PS C:\Users\pc\Desktop\M2 BIOINFO\S1\ACP\TP\TP3> mpiexec -n 4 python exercice3.py
Process 0: Valeur generee aleatoirement = 79
Process 1: Message reçu = 79
Process 2: Message reçu = 79
Process 3: Message reçu = 79
Process 0: Message final reçu = 79
PS C:\Users\pc\Desktop\M2 BIOINFO\S1\ACP\TP\TP3> █
```

Figure 3.1: Output

Chapter 4

Exercice 4

Le code source est dans le fichier "exercice4.py".

```
PS C:\Users\pc\Desktop\M2 BIOINFO\S1\ACP\TP\TP3> mpiexec -n 4 python exercice4.py
Blocking Broadcast Time: 0.031982 seconds
```

Figure 4.1: Output

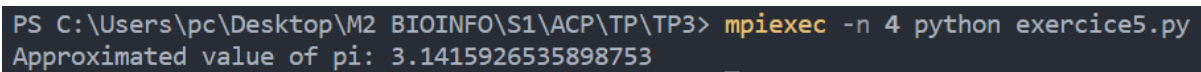
```
PS C:\Users\pc\Desktop\M2 BIOINFO\S1\ACP\TP\TP3> mpiexec -n 4 python exercice4.py
MPI_Bcast Time: 0.009996 seconds
```

Figure 4.2: Output

Chapter 5

Exercice 5 : (bcast et gather) calcul pi

Le code source est dans le fichier "exercice5.py".



```
PS C:\Users\pc\Desktop\M2 BIOINFO\S1\ACP\TP\TP3> mpiexec -n 4 python exercice5.py
Approximated value of pi: 3.1415926535898753
```

Figure 5.1: Output