Bacharelado em Ciência da Computação - DCC/IM-UFRJ
Programação Paralela e Distribuída
Prof. Gabriel P. Silva
2º Lista de Exercícios – 8/11/2008

1.Quais são os principais objetivos do padrão MPI?

2.O que são comunicadores? Qual o comunicador padrão no MPI?

3.Qual a função da rotina MPI_Comm_rank?

4.Qual o valor retornado pela rotina MPI_Comm_size no trecho de programa a seguir?

5.Para que servem os tipos padrão MPI_BYTE e MPI_PACKED?

6.Para que serve o handle "status"?

7.Como saber o número total de elementos recebido pela rotina abaixo?

```
MPI_Recv(&vector, 100, MPI_FLOAT, source, tag, MPI_COMM_WORLD, &status);
```

8.Quais as restrições existentes no uso das operações de comunicação coletivas?

9.Enumere e descreva sucintamente o funcionamento de cinco operações de comunicação coletivas.

10.Enumere e descreva quatro tipos de operações de redução utilizadas na função MPI_Reduce?

11.Qual o resultado da execução do seguinte trecho de código a seguir?

```
MPI_Reduce(&integral, &total, 1, MPI_FLOAT, MPI_SUM, 0, MPI_COMM_WORLD);
```

12.Quais os mecanismos que o MPI fornece para agrupamento de vários itens de dados em uma única mensagem?

13.Faça uma rotina que construa um tipo de dados derivado MPI com os seguintes tipos em C:

```
int a;
double b, c[4];
int d[2];
```

14.Faça uma rotina que construa um tipo de dados derivado MPI com os tipos abaixo em C. Utilize MPI_Type_contiguous primeiro e então MPI_Type_Struct:

```
#define MAX 65536
typedef struct {
        int     numero;
        float  vetor[MAX];
        } VETOR_LOCAL;
```

15.Faça uma rotina que envie e receba os tipos de dados da questão 14 utilizando MPI_Pack e MPI_Unpack.

16.Faça uma rotina que envie e receba os tipos de dados da questão 15 utilizando MPI_Pack e MPI_Unpack.

17.Quais os objetivos principais do uso de grupos e comunicadores?

18.Considere que um comunicador inicialmente é constituído por 12 processos. Elabore um programa que construa dois comunicadores com os processos cujo rank seja múltiplo de 2; outro com os ranks múltiplos de 3; um terceiro que com os ranks múltiplos de 2 e de 3; finalmente um quarto com os processos que não sejam nem múltiplos de 2 nem múltiplos de 3.

19.Qual é o comportamento das rotinas MPI_Send e MPI_Recv no modo de comunicação bloqueante?

20.Reescreva o trecho de programa a seguir utilizando rotinas de comunicação não-bloqueantes.

```
#define MAXSIZE 65536

successor = (my_rank + 1) % p;
predecessor = (my_rank - 1 + p) % p;

MPI_Send(a, 1, MPI_FLOAT, sucessor, 1, MPI_COMM_WORLD);
MPI_Send(vetor, MAXSIZE, MPI_FLOAT, successor, 1, MPI_COMM_WORLD);

MPI_Recv(a, 1, MPI_FLOAT, predecessor, 1, MPI_COMM_WORLD);
MPI_Recv(vetor, MAXSIZE, MPI_FLOAT, predecessor, 1, MPI_COMM_WORLD);
```

21.Enumere e descreva sucintamente os modos de comunicação das rotinas de envio de mensagem do MPI.

22.Descreva as vantagens e desvantagens do uso do modo de comunicação síncrona.

23.Quais as vantagens e desvantagens do uso do modo de comunicação padrão.

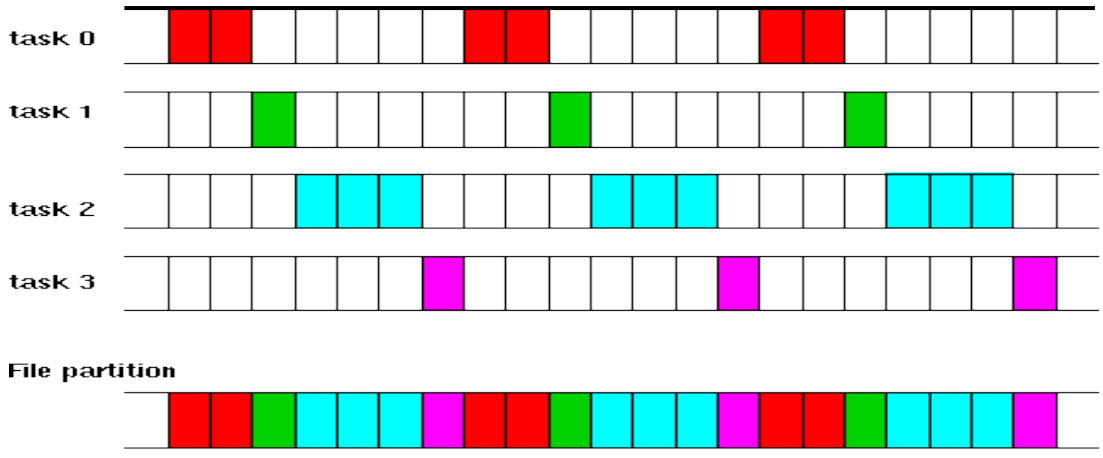
24.Reescreva o trecho de código da questão 20 utilizando rotinas de envio bufferizadas.

25.Reescreva o trecho de código da questão 20 utilizando rotinas de envio síncronas.

26.Escreva um trecho de programa com envio e recepção de mensagens entre 4 processos em anel no qual garantidamente vá acontecer “deadlock”.

27.Escreva um trecho de programa com envio e recepção de mensagens entre 4 processos em anel no qual garantidamente **não** vá acontecer “deadlock”.

28. Escreva um trecho de código usando rotinas do MPI I/O que configure uma vista de acordo com o diagrama abaixo:



29.Considere o seguinte trecho de código:

```
numt = MPI_Group_size(MPI_WORLD_COMM);
do {
   MPI_Send(msg,strlen(msg)+1,MPI_CHAR,numt-1,1,MPI_COMM_WORLD);
   }
while (--numt);
```

Indique quais trechos de código poderiam ser utilizados para recepção desta mensagem com sucesso. Explique o porquê.

a)	**MPI_Recv(msg, 100, MPI_CHAR, origem, MPI_ANY_TAG, MPI_COMM_WORLD, &status);**
b)	**MPI_Recv(msg, 100, MPI_PACKED, MPI_ANY_SOURCE, 0, MPI_COMM_WORLD, &status);**
c)	**MPI_Recv(msg, 100, MPI_CHAR, 0, MPI_ANY_TAG, MPI_COMM_WORLD, &status);**
d)	**MPI_Recv(msg, 100, MPI_CHAR, 0, 1, MPI_COMM_WORLD, &status);**
e)	**MPI_Recv(msg, 100, MPI_CHAR, MPI_ANY_SOURCE, 1, MPI_COMM_WORLD, &status);**
f)	**MPI_Recv(msg, 100, MPI_BYTE, MPI_ANY_SOURCE, MPI_ANY_TAG, MPI_COMM_WORLD, &status);**

30.Considere três tarefas executando o seguinte código:

task1	task2	task3
MPI_Isend(...,task3, 1,...); **xxx** **MPI_Isend(...,task2, 2,...);** **xxx** **MPI_Ssend(...,task3, 3,...);** **xxx** **MPI_Recv(...,MPI_ANY_TAG,MPI_ANY_SRC,...);**	**MPI_Recv(...,MPI_ANY_SOURCE, MPI_ANY_TAG,...);** **xxx** **MPI_Recv(...,MPI_ANY_SOURCE, MPI_ANY_TAG, ...);** **xxx** **MPI_Ssend(...,task1,5,...)**	**MPI_Recv(...,MPI_ANY_SOURCE,1, ...);** **xxx** **MPI_Recv(...,task1,3,...);** **xxx** **MPI_Isend(...,task2,4,...);** **xxx** **MPI_Ssend(...,task1,6,...);**

Qual será a ordem das mensagens recebidas por task2 e task3? Qual será a mensagem recebida por task1?

31.Escreva uma rotina para calcular o produto escalar de dois vetores utilizando rotinas do MPI. Considere cada vetor com 100 posições e divida entre 10 tarefas distintas. Utilize rotinas de comunicação coletivas para envio do vetor e recepção dos valores parciais. Não utilize as rotinas MPI_Pack e MPI_Unpack. Os vetores de entrada e os resultados deverão ser lidos de arquivos utilizando rotinas do MPI I/O.

32. Qual a diferença entre as rotinas MPI_File_write_at e MPI_write_at_all?