Exámenes

MPI

Volver a la Lista de Exámenes

Parte 1 de 1 - / 1.0 Puntos

Preguntas 1 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Indica cuál de las siguientes afirmaciones es correcta:

- ✔ ◯ A. Una llamada a la función `MPI_Bsend` puede terminar antes de que el receptor haya hecho la llamada correspondiente a `MPI_Recv`.
- ✖ ◯ B. El receptor de un mensaje debe conocer el tamaño exacto del mismo antes de su recepción, ya que debe indicarlo como argumento de la llamada a `MPI_Recv`
- ✔ ◯ C. Los mensajes enviados con una determinada etiqueta (*tag*) solo pueden ser recibidos si se ha especificado la misma etiqueta en la llamada a la operación de recepción (no es posible indicar que se reciba cualquier etiqueta)
- ✔ ◯ D. En el receptor es necesario indicar siempre el identificador (*rank*) del proceso emisor del que se desea recibir (no es posible recibir de cualquier proceso)

Respuesta correcta: A

Preguntas 2 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Dado el siguiente fragmento de código:

```
MPI_Comm_rank(MPI_COMM_WORLD, &rango);
if (rango==0) {
   x=100;
   MPI_Isend(&x, 1, MPI_INT, 1, tag, MPI_COMM_WORLD, &request);
   x=200;
   MPI_Wait(&request, &status);
} else if (rango==1) {
   MPI_Recv(&x, 1, MPI_INT, 0, tag, MPI_COMM_WORLD, &status);
}
```

- ✖ ◯ A. El valor final de x será 200 en el proceso de rango 0 y 100 en el proceso de rango 1.

- ✔ ◯ B. El código es erróneo porque se debe utilizar `MPI_Irecv` para recibir un mensaje enviado con `MPI_Isend`.
- ✔ ◯ C. El valor final de x será 200 en el proceso de rango 0, e indefinido en el proceso de rango 1.
- ✔ ◯ D. El valor final de x será indefinido tanto en el proceso de rango 0 como en el de rango 1.

Respuesta correcta: C

Preguntas 3 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

La llamada `MPI_Send`

- ✔ ◯ A. Dependiendo del tipo de datos del mensaje es bloqueante o no.
- ✔ ◯ B. Nunca es bloqueante.
- ✔ ◯ C. Dependiendo del tamaño del mensaje es bloqueante o no.
- ✖ ◯ D. Es siempre bloqueante.

Respuesta correcta: C

Preguntas 4 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Sea el siguiente fragmento de código:

```
dest = (rank+1) % p;
src = (rank+p-1) % p;
if (rank%2) {
  MPI_Recv(&tmp, 1, MPI_DOUBLE, src, 0, MPI_COMM_WORLD, &status);
  MPI_Send(&val, 1, MPI_DOUBLE, dest, 0, MPI_COMM_WORLD);
  val = tmp;
} else {
  MPI_Send(&val, 1, MPI_DOUBLE, dest, 0, MPI_COMM_WORLD);
  MPI_Recv(&val, 1, MPI_DOUBLE, src, 0, MPI_COMM_WORLD, &status);
}
```

en donde las variables enteras `rank` y `p` contienen el identificador de proceso y el número de procesos, respectivamente. ¿Qué tipo de transferencia de la variable `val` se realiza?

- ✔ ◯ A. Se trata de un reparto de datos desde el proceso 0
- ✔ ◯ B. Se trata de una difusión desde el proceso 0

- ✔ ◯ C. Se trata de un desplazamiento en anillo
- ✖ ◯ D. Se trata de un desplazamiento en malla 1-D

Respuesta correcta: C

Preguntas 5 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

El siguiente fragmento de código MPI, suponiendo que se han lanzado 10 procesos,

```
if (my_rank < 5)
   MPI_Bcast(buf, count, MPI_INT, 0, MPI_COMM_WORLD);
```

- ✔ ◯ A. Sólo recogerían los datos los primeros 5 procesos en llegar al `MPI_Bcast`
- ✔ ◯ B. Sólo recogerían los datos los procesos con índice 0, 1, 2, 3, y 4
- ✔ ◯ C. Habría que diferenciar entre las llamadas a `MPI_Bcast` del proceso 0 y del resto para que funcionara
- ✔ ◯ D. No funcionaría correctamente, ya que no todos los procesos ejecutan la operación colectiva

Respuesta correcta: D

Preguntas 6 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

La función `MPI_Scatter` ...

- ✔ ◯ A. Permite distribuir un vector entre varios procesos en fragmentos de igual tamaño.
- ✔ ◯ B. Permite hacer una reducción del contenido de un vector y difundir el resultado a todos los procesos.
- ✔ ◯ C. Permite replicar el contenido de un vector a varios procesos del sistema.
- ✔ ◯ D. Permite recoger los fragmentos de un vector de varios procesos y componer con ellos un vector único.

Respuesta correcta: A

Preguntas 7 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Si np es el número de procesos y k es el identificador del proceso actual, ¿a qué operación colectiva equivale el siguiente código?

```
if (k==0) {
   for (i=1;i<np;i++) MPI_Send(&n,1,MPI_INT,i,12,MPI_COMM_WORLD);
} else {
   MPI_Recv(&n,1,MPI_INT,0,12,MPI_COMM_WORLD,&status);
}
```

- ✔ ◯ A. `MPI_Scatter`
- ✔ ◯ B. `MPI_Bcast`
- ✔ ◯ C. `MPI_Reduce`
- ✔ ◯ D. `MPI_Allreduce`

Respuesta correcta: B

Preguntas 8 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Se tienen 4 procesos MPI con las siguientes variables enteras en cada uno de los procesos:

```
rank=0, a=12, b=-1
rank=1, a=18, b=-1
rank=2, a=22, b=-1
rank=3, a=8, b=-1
```

¿Cuál es el resultado de la siguiente operación?

```
MPI_Reduce(&a, &b, 1, MPI_INT, MPI_MAX, 1, MPI_COMM_WORLD);
```

- ✖ ◯ A.
  ```
  rank=0, a=12, b=22
  rank=1, a=18, b=-1
  rank=2, a=22, b=-1
  rank=3, a=8, b=-1
  ```
- ✔ ◯ B.
  ```
  rank=0, a=12, b=-1
  rank=1, a=18, b=22
  rank=2, a=22, b=-1
  rank=3, a=8, b=-1
  ```
- ✔ ◯ C.
  ```
  rank=0, a=12, b=18
  ```

```
rank=1, a=18, b=18
rank=2, a=22, b=18
rank=3, a=8, b=18
```

- ✔ ○ D.

```
rank=0, a=12, b=22
rank=1, a=18, b=22
rank=2, a=22, b=22
rank=3, a=8, b=22
```

Respuesta correcta: B

Preguntas 9 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Dado el siguiente fragmento de código, donde rango contiene el identificador de proceso:

```
double x[6]={1,2,3,4,5,6};
double y[6]={0,0,0,0,0,0};
MPI_Datatype T;
MPI_Type_vector(3, 1, 2, MPI_DOUBLE, &T);
MPI_Type_commit(&T);
if (rango==0)
   MPI_Send(x, 1, T, 1, tag, MPI_COMM_WORLD);
else
   MPI_Recv(y, 3, MPI_DOUBLE, 0, tag, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
MPI_Type_free(&T);
```

Tras ejecutarse, el contenido de y en el proceso 1 será:

- ✔ ○ A. {2,4,6,0,0,0}
- ✔ ○ B. {1,2,3,0,0,0}
- ✔ ○ C. {1,0,3,0,5,0}
- ✔ ○ D. {1,3,5,0,0,0}

Respuesta correcta: D

Preguntas 10 de 10

0.1 Puntos. Puntos descontados por fallo: 0.025

Deseamos definir un nuevo tipo de datos en MPI, llamado `fil_doble`, de manera que podamos realizar envíos eficientes de dos filas separadas entre sí por una fila (por ejemplo, filas 1 y 3, o filas 2 y 4, etc.) de una matriz declarada como `double a[N][N]`. Esto se puede hacer mediante la llamada:

- ✔ ○ A. `MPI_Type_vector(N, 2, 2*N, MPI_DOUBLE, &fil_doble)`
- ✔ ○ B. `MPI_Type_vector(2, 2*N, N, MPI_DOUBLE, &fil_doble)`
- ✔ ○ C. `MPI_Type_vector(2, N, 2*N, MPI_DOUBLE, &fil_doble)`
- ✖ ○ D. `MPI_Type_vector(2*N, N, 2, MPI_DOUBLE, &fil_doble)`

Respuesta correcta: C