Universitat d'Alacant
Universidad de Alicante

Práctica Guiada y Consolidada – Introducción a MPI (Ejemplos completos)

Objetivos de la práctica

Esta práctica guiada está diseñada para afianzar los conceptos básicos de MPI.

Cada ejercicio incluye:

- Un ejemplo resuelto y comentado (programa completo).
- Preguntas de comprensión.
- Un ejercicio similar para que el alumnado lo resuelva.

La evaluación será conjunta con la práctica anterior y con la siguiente, siguiendo la rúbrica adjunta.

Ejercicio 1 - Comunicación punto a punto

Ejemplo comentado:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
  MPI_Init(&argc, &argv);
  int rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  if (rank == 0) {
    int dato = 42; // Valor a enviar
    MPI_Send(&dato, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
  } else if (rank == 1) {
    int recibido;
    MPI_Recv(&recibido, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    printf("Proceso 1 recibió: %d\n", recibido);
  }

  MPI_Finalize();
  return 0;
}
```

Preguntas de comprensión:

- ¿Qué sucede si cambiamos los tags?

- ¿Qué ocurre si el proceso 1 ejecuta MPI_Recv antes de que el proceso 0 envíe el mensaje?

- ¿Qué diferencia habría si el proceso 0 intentase enviar el dato a un proceso inexistente?

Realiza:

Haz que el proceso 0 envíe un número al proceso 2 y este lo multiplique por 3 antes de imprimirlo.

Ejercicio 2 - Comunicación en cadena

Ejemplo comentado:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
 MPI_Init(&argc, &argv);
 int rank, size;
 MPI_Comm_rank(MPI_COMM_WORLD, &rank);
 MPI_Comm_size(MPI_COMM_WORLD, &size);

 if (rank == 0) {
  int dato = 0;
  MPI_Send(&dato, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
 } else {
  int recibido;
  MPI_Recv(&recibido, 1, MPI_INT, rank - 1, 0, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
  recibido++;
  if (rank < size - 1) {
   MPI_Send(&recibido, 1, MPI_INT, rank + 1, 0, MPI_COMM_WORLD);
  } else {
   printf("Valor final: %d\n", recibido);
  }
 }

 MPI_Finalize();
 return 0;
}
```

Preguntas de comprensión:

- ¿Qué sucede si un proceso omite el envío?

- ¿Qué ocurre si ejecutamos el programa con solo un proceso?

- ¿Cómo podríamos hacer que el proceso 0 también imprima el dato inicial?

Realiza:

Haz que el dato inicial sea igual a 10 y cada proceso lo incremente en 2 antes de pasarlo al siguiente.

Ejercicio 3 - Comunicación colectiva

Ejemplo comentado:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
  MPI_Init(&argc, &argv);
  int rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  int dato;
  if (rank == 0) dato = 99; // El root inicializa el dato
  MPI_Bcast(&dato, 1, MPI_INT, 0, MPI_COMM_WORLD); // Difusión a todos
  printf("Proceso %d recibió: %d\n", rank, dato);

  MPI_Finalize();
  return 0;
}
```

Preguntas de comprensión:

- ¿Qué pasa si el root es diferente?

- ¿Qué problema evitaríamos usando MPI_Bcast en lugar de MPI_Send?

- ¿Qué ocurre si un proceso no participa en la operación colectiva?

Ahora tú:

Haz que el proceso 2 envíe un número a todos los procesos con MPI_Bcast y que todos lo incrementen en 1.

Ejercicio 4 - Reducción de datos

Ejemplo comentado:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
  MPI_Init(&argc, &argv);
```

```
 int rank, suma;
 MPI_Comm_rank(MPI_COMM_WORLD, &rank);

 MPI_Reduce(&rank, &suma, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD); // Suma de
ranks
 if (rank == 0) printf("Suma de ranks: %d\n", suma);

 MPI_Finalize();
 return 0;
}
```

Preguntas de comprensión:

- ¿Qué diferencia hay entre MPI_Gather y MPI_Reduce?

- ¿Qué pasaría si omitimos el proceso root en MPI_Reduce?

- ¿Qué otras operaciones podrías realizar en lugar de suma?

Ahora tú:

Haz que todos los procesos calculen y muestren el máximo de los ranks utilizando MPI_Reduce.

Ejercicio 5 - Sincronización y medición de tiempos

Ejemplo comentado:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv) {
 MPI_Init(&argc, &argv);
 int rank;
 MPI_Comm_rank(MPI_COMM_WORLD, &rank);

 double t1 = MPI_Wtime();
 MPI_Barrier(MPI_COMM_WORLD); // Sincronización
 double t2 = MPI_Wtime();
 printf("Proceso %d: Tiempo = %f\n", rank, t2 - t1);

 MPI_Finalize();
 return 0;
}
```

Preguntas de comprensión:

- ¿Por qué cada proceso puede registrar un tiempo diferente?

- ¿Qué pasaría si un proceso hace un sleep(3) antes de llegar a la barrera?

- ¿Qué utilidad tiene medir estos tiempos en aplicaciones reales?

Realiza:

Simula un desbalance de carga haciendo que los procesos con rank par esperen 2 segundos antes de la barrera y mide el tiempo total.

Instrucciones de la práctica

Esta práctica consolida los contenidos trabajados en las sesiones anteriores de Introducción a MPI.
Incluye ejemplos completos, preguntas de comprensión y ejercicios prácticos.
La práctica está dividida en dos bloques:
1. Comunicación punto a punto y comunicación en cadena.
2. Comunicación colectiva, reducción de datos y sincronización.

Instrucciones de entrega

Deberás comprimir en un único archivo .zip los siguientes elementos:

- Códigos fuente de todos los ejercicios de esta práctica (5 ejercicios).

- Códigos fuente de la práctica de la semana anterior.

- Capturas de pantalla que demuestren la ejecución correcta de cada ejercicio.

- Un documento PDF o RTF que incluya:

  - Las respuestas a las preguntas de comprensión de esta práctica.
  - Las respuestas de la práctica anterior.
  - Una breve reflexión final sobre las dificultades encontradas (opcional).

La práctica debe subirse a Moodle, en la actividad habilitada para tal efecto (que se habilitará la próxima y última session de introducción a MPI).

La evaluación será conjunta con la práctica anterior y con la siguiente, siguiendo la rúbrica adjunta.

Universitat d'Alacant
Universidad de Alicante

Rúbrica de Evaluación

Criterio	Descripción	Puntuación
Realización completa	Entrega de todos los ejercicios de esta práctica y de la práctica anterior.	3 puntos
Aplicación correcta de MPI	Uso adecuado de funciones MPI y estructuras paralelas en todos los ejercicios.	2 puntos
Calidad y comentarios	Claridad, limpieza y comentarios explicativos en los códigos fuente.	2 puntos
Respuestas y reflexión	Contestación adecuada a las preguntas de comprensión y reflexión final.	2 puntos
Presentación clara	Buena organización de los archivos y formato de entrega correcto.	1 punto

Justificación Académica

Esta práctica permite al alumnado consolidar los conceptos fundamentales de comunicación paralela con MPI:

- Comunicación punto a punto y colectiva.

- Sincronización y medición de tiempos.

- Gestión básica de errores.

Además, da continuidad a la práctica de la semana anterior, favoreciendo la reflexión progresiva.

Los contenidos de esta práctica contribuyen a los resultados de aprendizaje relacionados con la implementación de aplicaciones paralelas sobre arquitecturas multicomputador, la comprensión del diseño y rendimiento de sistemas de computación de alto rendimiento y la preparación para abordar problemas complejos que requieren paralelización eficiente, tal y como establecen los objetivos formativos de la asignatura 'Computación de Alto Rendimiento'.