Javier García Borrego

Practica 2 - MPI

Ampliación de sistemas operativos

Doble grado en Ing. Informática + Ing. Computadores

Contenido

[Introducción: 1](#_Toc106038199)

[Explicación del código Parking.c: 1](#_Toc106038200)

[Explicación de coche.c y camión.c: 4](#_Toc106038201)

[Makefile: 5](#_Toc106038202)

[Ejemplos de resultados: 6](#_Toc106038203)

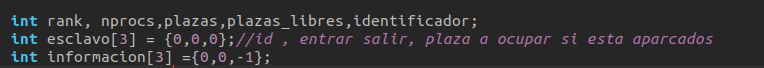
# Introducción:

La practica 2 consiste en crear un parking que se puedan aparcar coches y camiones los cuales serán procesados por separado y se comunicaran con la librería open-MPI.

No se ha realizado el apartado de tener múltiples plantas.

# Explicación del código Parking.c:

Parking es el encargado de gestionar la entrada/salida de los vehículos así que lo primero que se hace en el código es inicializar las variables que vamos a necesitar que son las siguientes:



Rank: es el id del proceso.

Nprocs: es el número de procesos.

Plazas: es el número de plazas totales.

Plazas\_libres: son el numero de plazas libres que quedan.

Identificador: es para guardar el tag que se recibe (0 si es coche o 1 si es camión)

Esclavo: es el mensaje que recibimos del proceso esclavo

Información: es el mensaje que enviamos al proceso esclavo

Luego procedemos a iniciar el entono MPI:

Texto

Descripción generada automáticamente

inicialización parecida a los ejemplos.

Texto

Descripción generada automáticamente

Se cogen el numero de plazas que se pasa como argumentos y se ponen las plazas libres iguales a las plazas totales además de que inicializamos todas las plazas a 0.

Texto

Descripción generada automáticamente

Entramos en bucle while que no parara hasta que hagamos ctrl+C y terminemos el proceso luego hacemos MPI\_Recv que se quedará esperando hasta que un coche o camión le envíen un mensaje cuando reciba un mensaje en identificador se guardará la TAG que será 0 si es un coche o 1 si es un camión a parte guardamos en información su id en la posición 0 y si quieres entrar o salir (0 o 1) en la posición 1.

Si el coche o camión quieren entrar se identificará si es un coche o un camión y se mirara si hay plazas disponibles si las hay se realizaran las operaciones necesarias para poner en el aparcamiento el numero del coche en la plaza adecuada y dejar las plazas disponibles además de cambiar la posición 1 del array información a 1 para saber que ha dicho coche ha aparcado y su próxima instrucción será que quiere salir al final se manda el mensaje información a coche o camión.

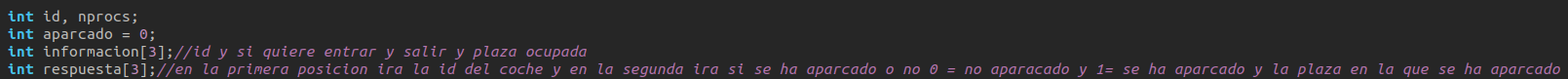
Texto

Descripción generada automáticamente

En caso de salir se realizara lo mismo la única diferencia en las operaciones que se realizaran dependiendo de si es un coche o un camion ya que uno ocupa una plaza solo y otro ocupa 2.

# Explicación de coche.c y camión.c:

Se inician las variables a usar:



Id: es el id del proceso.

Nprocs: el numero de procesos.

Aparcado: nos dice si el coche o camion ha sido aparcado.

Información: es la información que pasamos al proceso principal.

Respuesta: es la respuesta que recibimos del proceso principal.

Luego iniciamos el entorno MPI igual que en parking.c:

Texto

Descripción generada automáticamente

Entramos en el while:

Texto

Descripción generada automáticamente

Ponemos valores al mensaje información con su id si quiere entrar o salir y la plaza ocupada que en este caso es ninguna y entramos en un bucle do while que solamente saldremos de el cuando la posición 1 de respuesta sea 1 ya que quiere decir que se ha aparcado si no volveremos otra vez a hacer el bucle y así hasta encontrar sitio.

Cuando hemos encontrado sitio hacemos que el proceso se duerma un tiempo aleatorio comprendido entre 0 y 20 (un valor alto para ver si se realizan interbloqueos o inanición).

Para finalizar mandamos que queremos salir y ponemos un MPI\_Recv para que el proceso espere a la respuesta y pida volver a entrar antes de tiempo.

La diferencia entre camion.c y coche.c es la siguiente:

Camion.c:



Coche.c:



En la parte en la que iría la TAG se pone 0 o 1 dependiendo de si es un coche o un camion correspondiente.

# Makefile:

Se ha realizado un makefile para una ejecución de la practica el cual es el siguiente:

Texto

Descripción generada automáticamente

El localhost es un txt en el que pongo slots para poder usar la aplicación ya que si no saltaba este error:

Texto

Descripción generada automáticamente

# Ejemplos de resultados:

7 plazas 5 coches y 2 camiones:

Texto

Descripción generada automáticamente

5 plazas, 8 coches y 4 camiones:

Texto

Descripción generada automáticamente

Como se puede ver no se producen interbloqueos en ninguno de los 2 ejemplos.