UNIVERSIDAD
DE GRANADA

2ºA - Grupo A1

Grado en Ingeniería Informática

Sistemas Concurrentes y Distribuidos
Práctica 3 - Implementación de algoritmos distribuidos con MPI

Autor:
Jose Luis Gallego Peña

10 de diciembre de 2018

Índice

1. Productor-Consumidor con buffer acotado

1.1. Cambios realizados sobre el programa de partida

Partiendo de *prodcons2.cpp* se ha creado una versión para múltiples productores y consumidores llamada *prodcons2-mu.cpp* con los siguientes cambios:

- Se han cambiado los valores de las constantes id_productor, id_buffer e id_consumidor, cada uno al valor máximo de número de procesos asociados a cada uno de los roles. Además se han creado dos etiquetas, una para indicar productores y otra para consumidores a la hora de comunicar con el buffer.
- La función del buffer ahora determina si puede enviar solo prod, cons o todos según las etiquetas, siendo el source de MPI_Recv ahora cualquier proceso (MPI_ANY_SOURCE), para así poder englobar todos los procesos consumidor en una etiqueta y todos los procesos productor en otra, ya que no se puede usar el id al tener cada proceso uno distinto pero ser del mismo rol.
- En el main se ejecuta la función productor si el id propio es menor que el id productor, ya que deben ser los procesos desde el 0 hasta el número de productores - 1. Además se calculan unos números de orden para el productor y consumidor, siendo cada uno de 0 al número de productores/consumidores, y siendo pasados como parámetros a las funciones de productor y consumidor para así indicar con los mensajes por pantalla cual es el proceso actuando.

1.2. Listado parcial de la salida del programa

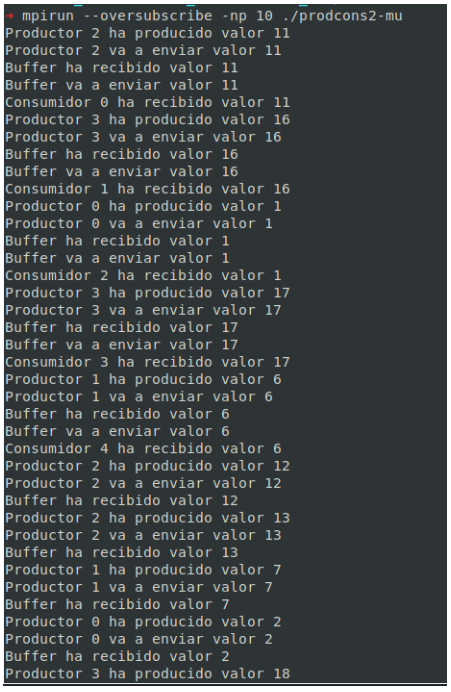

Figura 1: Ejecuciones prodcons2-mu.cpp

2. Cena de los Filósofos

2.1. Aproximación inicial

2.1.1. Aspectos destacados de la solución

La versión del problema de los filósofos aportada en *filosofos-interb.cpp* puede conducir a interbloqueo ya que cada filósofo coge primero el tenedor de su izquierda y después el de la derecha, y si dos filósofos que están juntos cogen a la vez su tenedor izquierdo, dejan al otro sin su derecho. Por tanto para solucionar esto se ha optado por hacer que el primer proceso coja los tenedores al revés: primero el derecho y luego el izquierdo, lo que soluciona el interbloqueo.

Los filosofos usan Ssend para pedir cada tenedor y para soltar cada tenedor. Los tenedores reciben estas peticiones de coger y soltar. Se ha usado una etiqueta para la acción de coger un tenedor, y otra para la acción de soltar, para así no producir interbloqueo al tener dos Recv seguidos.

2.1.2. Listado parcial de la salida del programa

```
→ mpirun --oversubscribe -np 10 ./filosofos
Filósofo 4 solicita ten. izq.5
Filósofo 2 solicita ten. izq.3
Filósofo 6 solicita ten. izq.7
Filósofo 4 solicita ten. der.3
Ten. 5 ha sido cogido por filo. 4
Filósofo 0 solicita ten. der.9
Filósofo 4 comienza a comer
Ten. 3 ha sido cogido por filo. 4
Filósofo 0 solicita ten. izq.1
Filósofo 6 solicita ten. der.5
Ten. 9 ha sido cogido por filo. 0
Filósofo 0 comienza a comer
Ten. 1 ha sido cogido por filo. 0
Ten. 7 ha sido cogido por filo. 6
Filósofo 8 solicita ten. izq.9
Filósofo 0 suelta ten. izq. 1
Filósofo 0 suelta ten. der. 9
Ten. 1 ha sido liberado por filo. 0
Filósofo 8 solicita ten. der.7
Ten. 9 ha sido liberado por filo. 0
Ten. 9 ha sido cogido por filo. 8
Filosofo 0 comienza a pensar
Filósofo 0 solicita ten. der.9
Filósofo 4 suelta ten. izq. 5
Filósofo 4 suelta ten. der. 3
Ten. 5 ha sido liberado por filo. 4
Ten. 5 ha sido cogido por filo. 6
Ten. 3 ha sido liberado por filo. 4
Ten. 3 ha sido cogido por filo. 2
Filosofo 4 comienza a pensar
Filósofo 2 solicita ten. der.1
Filósofo 6 comienza a comer
Ten. 1 ha sido cogido por filo. 2
Filósofo 2 comienza a comer
Filósofo 4 solicita ten. izq.5
Filósofo 2 suelta ten. izq. 3
Ten. 3 ha sido liberado por filo. 2
Filósofo 2 suelta ten. der. 1
Ten. 1 ha sido liberado por filo. 2
```

Figura 2: Ejecuciones prodcons2-mu.cpp

2.2. Uso del proceso camarero con espera selectiva

2.2.1. Solución al problema de los filósofos con camarero central

Se aporta la solución al problema de los filósofos con camarero central en el archivo *filosofos-camarero.cpp* con los siguientes cambios:

- Se han creado dos nuevas etiquetas: para levantarse y sentarse, al igual que teníamos las de coger y soltar tenedor. Además, ahora el número de procesos es uno más al tener el camarero.
- La nueva función del camarero sigue un bucle infinito en el cual comprueba si el número de filosofos sentados es 4, y entonces es cuando recibe la petición de sentarse. Además, va comprobando en cada iteración si hay algun filosofo esperando para levantarse, y entonces le deja levantarse y se resta uno a la cuenta de filósofos sentados.
- En la función de los filósofos se añaden envíos de mensaje con las etiquetas de sentarse y levantarse respectivamente para pedirle al camarero estas acciones.

2.2.2. Listado parcial de la salida del programa

```
→ mpirun --oversubscribe -np 11 ./filosofos-camarero
Filósofo 8 solicita ten. izq.9
Camarero: Filósofo 8 se sienta en la mesa
Camarero: Filósofo 2 se sienta en la mesa
Camarero: Filósofo 4 se sienta en la mesa
Camarero: Filósofo 0 se sienta en la mesa
Filósofo 8 solicita ten. der.7
Filósofo 0 solicita ten. der.10
Ten. 9 ha sido cogido por filo. 8
Filósofo 4 solicita ten. izq.5
Filósofo 4 solicita ten. der.3
Ten. 5 ha sido cogido por filo. 4
Ten. 7 ha sido cogido por filo. 8
Filósofo 8 comienza a comer
Ten. 1 ha sido cogido por filo. 2
Filósofo 2 solicita ten. izq.3
Filósofo 2 solicita ten. der.1
Filósofo 2 comienza a comer
Ten. 3 ha sido cogido por filo. 2
Ten. 1 ha sido liberado por filo. 2
Filósofo 2 suelta ten. izq. 3
Filósofo 2 suelta ten. der. 1
Filosofo 2 comienza a pensar
Filósofo 4 comienza a comer
Filósofo 6 solicita ten. izq.7
Camarero: Filósofo 2 se levanta de la mesa
Camarero: Filósofo 6 se sienta en la mesa
Ten. 3 ha sido liberado por filo. 2
Ten. 3 ha sido cogido por filo. 4
Filósofo 8 suelta ten. izq. 9
Filósofo 8 suelta ten. der. 7
Ten. 7 ha sido liberado por filo. 8
Camarero: Filósofo 8 se levanta de la mesa
Camarero: Filósofo 2 se sienta en la mesa
Ten. 9 ha sido liberado por filo. 8
Filosofo 8 comienza a pensar
Ten. 7 ha sido cogido por filo. 6
Filósofo 2 solicita ten. izq.3
Filósofo 6 solicita ten. der.5
Filósofo 4 suelta ten. izq. 5
```

Figura 3: Ejecuciones prodcons2-mu.cpp