Yang Chen

Portafolio FilosofoCamarero

En este caso, para solucionar el problema de interbloqueo, se ha añadido un nuevo proceso con el ID=10, el camarero. El camarero informara a los filosofos si hay sitio libre o no, y los filosofos enviaran un mensaje al camarero cuando terminen de comer.
Las acciones que realiza los filosofos son: pide al camarero pra sentarse, se sienta, coge el tenedor izquierdo, coge el tenedor derecho, come, suelta el tenedor izquierdo, suelta el tenedor derecho, informa al camarero para levantarse.

Codigo fuente:

```
#include <iostream>
#include <time.h>          // incluye "time"
#include <unistd.h>        // incluye "usleep"
#include <stdlib.h>        // incluye "rand" y "srand"
#include <mpi.h>
using namespace std;

#define soltar         0
#define coger          1
#define sentarse       2
#define levantarse 3
#define camarero    10

void Filosofo( int id, int nprocesos);
void Tenedor ( int id, int nprocesos);
void Camarero ( int id, int nprocesos);
// ---------------------------------------------------------------------
int main( int argc, char** argv )
{
    int rank, size;

    srand(time(0));
    MPI_Init( &argc, &argv );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    MPI_Comm_size( MPI_COMM_WORLD, &size );

    if( size!=11)
    {
        if( rank == 0)
            cout<<"El numero de procesos debe ser 11" << endl << flush ;
        MPI_Finalize( );
        return 0;
```

```
    }

    if (rank == 10)
          Camarero(rank, size);
    else if ((rank%2) == 0)
        Filosofo(rank,size); // Los pares son Filosofos
    else
        Tenedor(rank,size);    // Los impares son Tenedores

    MPI_Finalize( );
    return 0;

}
// ----------------------------------------------------------------------

void Filosofo( int id, int nprocesos )
{
    int izq = (id+1) % (nprocesos-1);
    int der = ((id+nprocesos)-2) % (nprocesos-1);
    MPI_Status status;

    while(true)
    {

      cout << "Filosofo "<<id<< " solicita sentarse"<<endl<<flush;
      MPI_Send(NULL, 0, MPI_INT, camarero, sentarse, MPI_COMM_WORLD);

      MPI_Recv(NULL, 0, MPI_INT, camarero, sentarse, MPI_COMM_WORLD, &status);
           cout << "Filosofo " << id << " se sienta " << endl;

        // Solicita tenedor izquierdo
        cout << "Filosofo "<<id<< " solicita tenedor izq ..." << izq << endl << flush;
      MPI_Ssend(NULL, 0, MPI_INT, izq, coger, MPI_COMM_WORLD);
        // ...

        // Solicita tenedor derecho
        cout <<"Filosofo "<<id<< " coge tenedor der ..." << der << endl << flush;
      MPI_Ssend(NULL, 0, MPI_INT, der, coger, MPI_COMM_WORLD);
        // ...

        cout<<"Filosofo "<<id<< " COMIENDO"<<endl<<flush;
        sleep((rand() % 3)+1);    //comiendo

        // Suelta el tenedor izquierdo
```

```
        cout <<"Filosofo "<<id<< " suelta tenedor izq ..." << izq << endl << flush;
      MPI_Ssend(NULL, 0, MPI_INT, izq, soltar, MPI_COMM_WORLD);
        // ...

        // Suelta el tenedor derecho
        cout <<"Filosofo "<<id<< " suelta tenedor der ..." << der << endl << flush;
      MPI_Ssend(NULL, 0, MPI_INT, der, soltar, MPI_COMM_WORLD);
        // ...


      cout    << "Filosofo " << id << " se levanta "    << endl;
            MPI_Ssend(NULL, 0, MPI_INT, camarero, levantarse, MPI_COMM_WORLD );


        // Piensa (espera bloqueada aleatorio del proceso)
        cout << "Filosofo " << id << " PENSANDO" << endl << flush;

        // espera bloqueado durante un intervalo de tiempo aleatorio
        // (entre una d 茅 cima de segundo y un segundo)
        usleep( 1000U * (100U+(rand()%900U)) );

  }

}
// ----------------------------------------------------------------------
void Tenedor(int id, int nprocesos)
{
   int buf;
   MPI_Status status;
   int Filo;

   while( true )
   {
      // Espera un peticion desde cualquier filosofo vecino ...
       MPI_Recv(&buf, 1, MPI_INT, MPI_ANY_SOURCE, coger, MPI_COMM_WORLD, &status);
      // ...

      // Recibe la peticion del filosofo ...
      // ...
      Filo=status.MPI_SOURCE;
      cout << "Ten. " << id << " recibe petic. de " << Filo << endl << flush;

      // Espera a que el filosofo suelte el tenedor...
            MPI_Recv(&Filo, 1, MPI_INT, Filo, soltar, MPI_COMM_WORLD, &status);
```

```
        // ...
        cout << "Ten. " << id << " recibe liberac. de " << Filo << endl << flush;
    }

}

void Camarero(int id, int nprocesos){

        int sentados=0;
        int Filo;
        MPI_Status status;

      while(true){
        if(sentados < 4)
                MPI_Probe(MPI_ANY_SOURCE, MPI_ANY_TAG, MPI_COMM_WORLD, &status);
        else
                MPI_Probe(MPI_ANY_SOURCE, levantarse, MPI_COMM_WORLD, &status);

        Filo = status.MPI_SOURCE;

        if(status.MPI_TAG == sentarse){

                //Recibe peticion de sentarse de un filosofo
                MPI_Recv(NULL, 0, MPI_INT, Filo, sentarse, MPI_COMM_WORLD, &status);
                cout << "El filosofo "<< Filo << " ha solicitado asiento al camarero." << endl <<
flush;
                sentados ++;

                //Deja que se siente
                MPI_Send(NULL, 0, MPI_INT, Filo, sentarse, MPI_COMM_WORLD);
                cout << "El filosofo "<< Filo << " se sienta, ahora hay "<< sentados<< " filosofos
sentados." << endl << flush;

        }else if(status.MPI_TAG == levantarse){

                //El filosofo ha terminado de comer y desea levantarse
                MPI_Recv(NULL, 0, MPI_INT, Filo, levantarse, MPI_COMM_WORLD, &status);
                sentados --;
                cout << "El filosofo "<< Filo << " se levanta, ahora quedan " <<sentados<<"
filosofos en la mesa."<< endl << flush;

        }
      }
}
```

Salida:

```
yang@yang-VirtualBox:~/Escritorio/scdprac3$ mpicxx -o filosofosCamarero filosofosCamarero.cpp
yang@yang-VirtualBox:~/Escritorio/scdprac3$ mpirun -np 11 filosofosCamarero
Filosofo 6 solicita sentarse
Filosofo 8 solicita sentarse
Filosofo 2 solicita sentarse
Filosofo 0 solicita sentarse
El filosofo 6 ha solicitado asiento al camarero.
El filosofo 6 se sienta, ahora hay 1 filosofos sentados.
Filosofo 6 se sienta
Filosofo 6 solicita tenedor izq ...7
El filosofo 8 ha solicitado asiento al camarero.
Filosofo 8 se sienta
Filosofo 4 solicita sentarse
Ten. 5 recibe petic. de 6
Ten. 7 recibe petic. de 6
Filosofo 6 coge tenedor der ...5
El filosofo 8 se sienta, ahora hay 2 filosofos sentados.
El filosofo 2 ha solicitado asiento al camarero.
Filosofo 6 COMIENDO
Ten. 3 recibe petic. de 2
Filosofo 8 solicita tenedor izq ...9
Filosofo 8 coge tenedor der ...7
Filosofo 4 se sienta
Filosofo 4 solicita tenedor izq ...5
Ten. 9 recibe petic. de 8
El filosofo 2 se sienta, ahora hay 3 filosofos sentados.
El filosofo 4 ha solicitado asiento al camarero.
El filosofo 4 se sienta, ahora hay 4 filosofos sentados.
Filosofo 2 se sienta
Filosofo 2 solicita tenedor izq ...3
Filosofo 2 coge tenedor der ...1
Filosofo 2 COMIENDO
Ten. 1 recibe petic. de 2
Filosofo 6 suelta tenedor izq ...7
Filosofo 2 suelta tenedor izq ...3
Ten. 3 recibe liberac. de 2
Filosofo 2 suelta tenedor der ...1
Ten. 1 recibe liberac. de 2
Filosofo 2 se levanta
El filosofo 2 se levanta, ahora quedan 3 filosofos en la mesa.
Filosofo 2 PENSANDO
El filosofo 0 ha solicitado asiento al camarero.
El filosofo 0 se sienta, ahora hay 4 filosofos sentados.
Filosofo 0 se sienta
Filosofo 0 solicita tenedor izq ...1
Ten. 1 recibe petic. de 0
```