CC4302: Sistemas Operativos
Tarea 1 - Semestre Primavera 2018
Prof.: Luis Mateu

Los transbordadores

Mientras se construye el puente entre Chiloé y el continente se le ha encomendado a Ud. que implemente un sistema de *p* transbordadores de bajo costo entre Chacao y Pargua. Cada transbordador puede llevar un solo vehículo. Para llevar el vehículo *v* hacia Chacao en el transbordador *i* se debe ejecutar *haciaChacao(i, v)*, y para traerlo de regreso en el transbordador *j* se invoca *haciaPargua(j, v)*. Estas dos operaciones toman un tiempo considerable. Un número *indeterminado* de automovilistas, representados mediante threads que ejecutan las funciones *isleño* o *norteño*, invocan concurrentemente estas operaciones como se indica en el lado izquierdo del cuadro. Además el lado derecho del cuadro muestra una implementación de referencia para un administrador que utiliza un solo transbordador y de manera muy ineficiente.

<table>
<tr><th>Automovilistas</th><th colspan="2">Administrador</th></tr>
<tr><td rowspan="2"><pre>
int norteño(int v) {
  transbordoAChacao(v);
  ...
  transbordoAPargua(v);
}
int isleño(int v) {
  transbordoAPargua(v);
  ...
  transbordoAChacao(v);
}
</pre></td><td colspan="2"><pre>
    nSem mutex; /*=nMakeSem(1);*/
    int enPargua= 1;
</pre></td></tr>
<tr><td><pre>
void transbordoAChacao(int v)
{
  nWaitSem(mutex);
    if (!enPargua)
      haciaPargua(0, -1);
    haciaChacao(0, v);
    enPargua= 0;
  nSignalSem(mutex);
}
</pre></td><td><pre>
void transbordoAPargua(int v)
{
  nWaitSem(mutex);
    if (enPargua)
      haciaChacao(0, -1);
    haciaPargua(0, v);
    enPargua= 1;
  nSignalSem(mutex);
}
</pre></td></tr>
</table>

Requerimientos

Implemente un administrador para un sistema de *p* transbordadores. Es decir Ud. debe programar las funciones ***transbordoAChacao***, ***transbordoAPargua*** e ***inicializar***. La función *inicializar* recibe como único argumento el número de transbordadores disponibles *p*. Los transbordadores se enumeran de 0 a *p*-1. Suponga que inicialmente todos los transbordadores se encuentran en Pargua. Desde luego Ud. debe considerar que no se puede usar simultáneamente un mismo transbordador para dos trayectos simultáneos. Para invocar *haciaChacao(i, v)*, el transbordador *i* debe estar en Pargua (y viceversa). Si no lo está, deberá invocar *haciaPargua* antes de invocar *haciaChacao* para ese transbordador. Un vehículo no puede quedar esperando indefinidamente un transbordador por lo que deberá trasladar alguno de la otra orilla, aún cuando no haya ningún vehículo que transbordar en el sentido opuesto. El procedimiento *transbordoAChacao(v)* no puede retornar hasta que se haya invocado *haciaChacao(i, v)* para algún *i*.

Por razones de eficiencia, se requiere que Ud. aproveche de la mejor forma los *p* transbordadores disponibles. Si va a desplazar un transbordador, Ud. debe llevar un vehículo si hay alguno esperando en la orilla de origen. Los transbordadores que sobren deben detenerse.

Resuelva esta tarea usando nSystem y monitores. Baje de U-cursos los archivos adjuntos para esta tarea. Se incluye un *Makefile* para compilar, el archivo *transbordo.h* con los encabezados de las funciones pedidas y el test de prueba *test-transbordo.c* para verificar que su tarea funciona correctamente. Programe la solución de esta tarea en el archivo *transbordo.c*.

Test de prueba adicional

Además del test de prueba publicado, Ud. debe implementar un nuevo test de prueba original en el archivo test-*transbordo2.h*.

Entrega

Entregue por medio de U-cursos un archivo .zip con *transbordo.c* y *test-transbordo2.c*. Se descontará medio punto por día hábil de atraso. Se recomienta resolver esta tarea antes del control 1.