CC4302 Sistemas Operativos – Tarea 3
Semestre Otoño 2020 – Prof.: Luis Mateu

Programe un sistema *batch* que permita solicitar la ejecución de *jobs* por medio de la función *submitJob* y esperar hasta que un job termine con *waitJob*. El sistema ejecutará a lo más *n* jobs en paralelo. Los jobs se atenderán por orden de llegada. Concretamente se pide implementar la siguiente API:

- *void startBatch(int n)* : Lanza el sistema batch, siendo *n* el máximo grado de paralelismo. En esta función Ud. debe lanzar *n* threads que se usarán para ejecutar los jobs en paralelo. nEmitTask
- *Job* sumbitJob(void* (*f)(void *ptr), void *ptr)* : Solicita la invocación de *(*f)(ptr)* en el sistema batch. Entrega un descriptor del job que debe ser usado posteriormente al llamar a *waitJob*. Los jobs se asignan a los threads del sistema batch por orden de llegada.
- *void *waitJob(Job *job)* : Espera si es necesario a que termine el job especificado, entregando el resultado de la invocación de *f*. Con la invocación de *waitJob* se dice que el job está completo y el descriptor es liberado.
- *void stopBatch()* : Detiene los *n* threads del sistema batch. Una vez iniciada su invocación, no habrán más llamadas a *submitJob*, pero todavía se pueden realizar invocaciones de *waitJob* asociados a los jobs pendientes a partir de otros threads de la aplicación. La función debe esperar hasta que se ejecuten todos los jobs pendientes con sus respectivas invocaciones de *waitJob*.

Esta tarea es similar a la pregunta 1 del control 1 del semestre primavera de 2014. Lea el enunciado de esa pregunta. Para la espera en *stopBatch* cuente los jobs creados con *submitJob* y los jobs completados con *waitJob*. Ud. podrá terminar los threads una vez que los creados igualen a los completados. Está garantizado que no habrán más invocaciones de *submitJob* una vez iniciada la invocación de *stopBatch*.

Note que una vez que se detuvo el sistema batch, se puede volver a lanzar con un número *n* distinto, es decir con más o menos threads.

Requerimientos

- Ud. debe programar las 4 funciones pedidas en el archivo *batch.c*.
- Ud. debe resolver esta tarea usando los monitores de nSystem como herramienta de sincronización.
- Su tarea debe pasar todos los tests de prueba incluidos en *test-batch.*c.
- En la llamada a *startBatch* Ud. debe lanzar exactamente *n* threads. Ud. no puede crear threads adicionales para ejecutar los jobs.
- Ningún thread puede permanecer ocioso si existen jobs pendientes. Camiones
- En la llamada a *stopBatch* Ud. debe enterrar los *n* threads lanzados llamando a *nWaitTask*.
- Sea eficiente: use múltiples condiciones para la espera en *waitJob*. Debe evitar que un *nNotifyAll* despierte a todas las tareas que esperan en *waitJob*. Use una *FifoQueue* para encolar los jobs pendientes. nWaitCondition()

Archivos

En los archivos adjuntos Ud. encontrará:

- *Makefile*: Archivo con reglas para compilar su tarea. Ud. debe definir la variable de ambiente NSYSTEM con la ubicación de nSystem en su computador.
- *batch.h*: archivo de encabezados de la API solicitada.
- *test-batch.c*: el test de prueba.
- *batch.c.plantilla*: plantilla para programar *batch.c*.

Entrega

Ud. debe entregar mediante U-cursos el archivo *batch.c*. Se descontará medio punto por día hábil de atraso.