Tarea 2
'I Like Trains'
INF246-Sistemas Operativos

Francisco Tobar, Javier Cañas, Javier Campos,

12 de septiembre de 2016

1. Objetivo

El alumno deberá familiarizarse con la programación de threads y acceso concurrente a elementos compartidos a través del uso de locks y variables de condición.

2. Problema

A Chile ha llegado una nueva empresa de objetos tecnológicos **"Mapple.Inc"**. Para ingresar en el mercado nacional la compañía trajo a la ciudad de Valparaíso un cargamento de 25 Containers con sus nuevos productos que deben ser distribuidos a otras 5 ciudades del país (Santiago, Temuco, Antofagasta, Concepción y Puerto Montt).

Los contaniners serán transportados en tren, de uno en uno, desde la ciudad de Valparaíso hacia los almacenes de cada locación. Al ser una nueva empresa en el país, sus recursos aún son escasos y cuentan con solo 5 tipos de locomotoras, cada tipo puede ir a solo una ciudad en especifico y solo 5 rieles (cada riel une una de las ciudades con Valparaíso).

En cada almacén hay un supervisor que chequea los containers para asegurarse que todo ha llegado en perfectas condiciones además de supervisar el transporte de los objetos del container a las bodegas.

Para asegurar el éxito de estas operaciones se les pide a los estudiantes que simulen estas situaciones con programación concurrente.

3. Consideraciones

1. Los containers, por comodidad será enumerado del 1 al 15.
2. Se deben llevar 5 containers a cada locación.
3. Los trenes salen de forma paralela a distintos lugares.
4. Los trenes se demoran distintos tiempo en llegar a cada ciudad :
   - Santiago: 2 segundos
   - Temuco: 8 segundos
   - Antofagasta: 6 segundos
   - Concepción: 4 segundos
   - Puerto Montt: 10 segundos
5. Las locomotoras pueden dejar el container sin retardo significativo.
6. La locomotora después de dejar el container se queda es esa ciudad y es reemplazada, inmediatamente, por otra del mismo tipo en Valparaíso lista para partir.

   Hint: Para su comodidad puede considerar que la locomotora reemplazante es la misma que la sale de circulación para evitar la constante creación y eliminación de locomotoras.

7. Solo pueden haber 5 locomotoras asignadas o en funcionamiento al mismo tiempo.
8. A un container se le pueden asignar un riel y después una locomotora o vice versa, pero no se pueden asignar al mismo tiempo (no son acciones atómicas).
9. El supervisor solo puede chequear 1 containers o supervisar el transporte de productos al mismo tiempo.
10. Existen suficientes trabajadores para poder siempre descargar todos los containers a la vez.
11. Solo se descargan los containers si es que no quedan ninguno otro por chequear.
12. Toma 11 segundos descargar un container.
13. Toma 7 segundos chequear un container.
14. Debe imprimir por salida estándar cada vez que a un container se le asigna un riel, se le asigna una locomotora, sale de Valparaíso, llega a otra ciudad, es chequeado y es descargado.

4. Acerca de la entrega

- La tarea debe ser realizada en C++ en parejas o forma individual.
- Se deben usar las librerías sthread, lock y Cond suministrada en el example.tar.gz y ninguna otra para programar los thread, locks o las variables de condición. Fuera de lo anterior pueden ocupar cualquier otra librería que estimen conveniente.
- La entrega es un archivo de la forma rol1-rol2.tar.gz que contenga:
  - Un archivo de texto plano README explicando como se resolvió la tarea.
  - Un Makefile que limpie los archivos intermedios creados y que permita compilar la tarea. **Hint:** Tome como base el makefile de Example.tar.gz
  - Los códigos necesarios para compilar la tarea.
- Se descontará puntaje por no respetar la lógica del lenguaje (no modularizar), por código no comentado o poco explicativo.
- Su código **DEBE** asegurar que no ocurran deadlocks.
- La tarea debe poder ser compilada y corrida en el **labComp**.
- La entrega será el día miércoles 5 de octubre y se descontarán 15 puntos por cada día de atraso hasta un máximo de 3 días.