REDES Y SISTEMAS DISTRIBUIDOS

MiniProyecto Sistemas Distribuidos
Simulacion distribuida

Autor: Unai Arronategui

Límite entrega memoria y código fuente en moodle : 18 de noviembre de 2019

Resumen

Este trabajo plantea el diseño e implementación de un simulador distribuido de Redes de Petri.

La clase de redes de Petri a simular son las **binarias** (1 solo token por lugar) y **sin conflictos** (solo una transición como sucesor a cualquier lugar).

*Esta práctica incluye **redactar** una **memoria y escribir todo el código fuente (salvo el propocionado en la asignatura)**. El texto de la memoria y código debe ser original y bien explicado. Se puede comentar sobre la práctica con otros alumnos, pero no está permitido mirar o copiar el código de otros.*

Recursos asociados

1. Simulador centralizado

Adjunto al guión, en un directorio centralsim, teneis disponible el código fuente de una librería básica (package *centralsim*) de simulación centralizada de redes de petri utilizando LEFs. Esta librería está constituida de 4 fichero de código funcional y 2 ficheros de test. Además, en el directorio cmd/centralSim disponeis de un pequeño fichero main.go que permite obtener un comando ejecutable de simulación centralizada.

Como podeis observar en los ficheros “simulation_test.go” y “main.go”, la Red d Petri a simular se pasa como una estructura de datos literal que codifica ya de forma interna una subred, en un formato acorde con el mecanismo de LEFs, como una lista de transiciones que contienen los datos necesarios para definir su activación y la propagación de una constante que representa los efectos de su disparo a las LEFs de sus transiciones sucesoras.

En el documento anexo “Ejemplo1ParaTests.pdf” se explica tanto esta estructura interna de una subred como la evolución de su ejecución a lo largo del tiempo.

Comprobar que la librería de simulación centralizada funcione correctamente y explicar su desarrollo y ejecución.

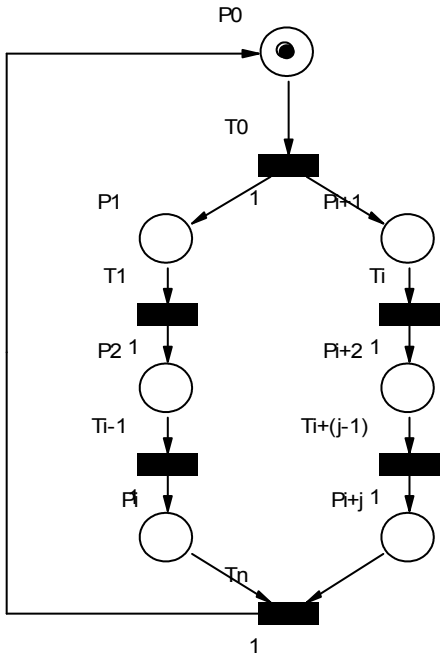
2. Simulador distribuido con sincronización conservativa

Diseñar e implementar una versión distribuida de simulador de RdP basada en sincronización pesimista mediante el mecanismo de LEFs y apoyandose en la implementación de la versión centralizada de simulador de la sección 1.

Para ello desarrollar las extensiones necesarias al simulador centralizado. En particular representar los lugares de entrada y salida de cada subred y la información de sincronización de lugares comunes entre subredes representada por la entrada “SYNCHRONIZATION” de la representación distribuida de RdP vista en clase. Y los elementos de comunicación necesarios para propagar las simulaciones entre subredes de Petri que se encuentren en diferentes máquinas.

Definir valores de look-up que permitan avanzar a la simulación distribuida conservativa.

Utilizar como ejemplo base de simulación distribuida el siguiente :



Definir las subredes a particionar y ampliar con ejemplos que incrementen el n.º de ramas a ejecutar en paralelo par incrementar el número de subredes a particionar hasta 5. Cambiar los tiempos de transiciones para observar diferentes comportamientos temporales de la simulación distribuida. Ampliar la longitud de las ramas paralelas a 4 transiciones por rama y efectuar alguna prueba adicional con tiempos dispares en las tansiciones para observar el comportamiento distribuido

3. Opcional : Simulación distribuida con sincronización optimista.

4. Requisitos de implementación

- El código implementado debe ser compatible con la versión 1.13.1 de Go.
- La solución planteada debe utilizar únicamente los paquetes de la **librería estándar** de Go y, eventualmente, el paquete ssh (golang.org/x/crypto/ssh) y los paquetes de código provistos junto a este guión.
- El estilo del código desarrollado debe ser comprobado o adaptado mediante la herramienta **gofmt**.

5. Memoria

En la memoria de prácticas a entregar debe aparecer claramente explicado el diseño de alto nivel de vuestras soluciones, los protocolos de interacción detallados entre los procesos distribuidos y los elementos más importantes de la implementación en el código.

6. Evaluación

Para la evaluación de la práctica, se debe entregar una memoria de su diseño e implementación y todo el código fuente necesario para ejecutar vuestro programa, el cual debera pasar los tests que hayais habilitado a tal efecto. Estos tests deberan ejecutarse con éxito en máquinas del laboratorio 1.02.

La entrega se realizará, mediante un solo fichero comprimido (tar.gz o zip), en una sección de **entrega** que se habilitará en la página moodle de la asignatura, siendo la fecha **límite el 18 de noviembre a las 9h**.