Universidad Nacional de Córdoba
Facultad de Ciencias Exactas Físicas y Naturales

Trabajo práctico N° 2:

Implementación de un sistema productor/consumidor, haciendo uso de un monitor de concurrencia con una red de Petri.

Alumnos

- *Chiacchio Hails, Joaquin*
- *Gatica Carolina*
- *Mayol, Adelina*
- *Pavón, Diego*
- *Sarquis, Tomas*

Grupo

- Adderall

Córdoba, Junio 2019

Índice

Introducción

Se debe implementar un sistema productor/consumidor, haciendo uso de un monitor de concurrencia con una red de Petri. El sistema debe poseer 2 buffers acotados con diferentes capacidades: 10 y 15 lugares respectivamente. Existen 5 productores y 8 consumidores.

Considerar:

- Cada buffer como un recurso que solo puede ser usado por un hilo al mismo tiempo.
- Cada productor inserta un producto en cualquiera de los buffers que esté disponible (no esté ocupado y tenga lugares libres). Si no hay buffer disponible, esperar hasta que uno esté disponible.
- Cada consumidor extrae un producto de cualquiera de los buffers que esté disponible (no esté ocupado y tenga productos).Si no hay ningún buffer disponible, esperar a que uno esté disponible.
- Una vez desarrollado el proyecto (funcionando) implemente los siguientes casos y extraiga conclusiones:
    A. El tiempo de inserción y extracción de productos es insignificante
    B. El tiempo de inserción y extracción de productos es de 50ms
    C. Opcional: analice e implemente el proyecto de modo que para el caso b., el tiempo de ejecución completa no supere los 3 minutos.

Objetivos

1. Modelar el sistema con una red de Petri y verificar todas sus propiedades haciendo uso de la herramienta PIPE.
2. Hacer el diagrama de clases que modele el sistema.
3. Hacer el diagrama de secuencias que muestre la interacción entre un productor, un consumidor y el monitor.
4. Modelar el sistema con objetos en Java.
5. Realizar una ejecución con todos los productores y consumidores trabajando concurrentemente, donde cada productor inserta 10.000 productos.
6. Verificar los resultados haciendo uso de un archivo de log.
7. Debe hacer una clase Main que al correrla, inicie el programa.

Desarrollo

1. Modelado del sistema con una red de Petri y verificación todas sus propiedades.

Con la red de Petri utilizaremos para su análisis la herramienta PIPE, que es la que se usó en el transcurso del cursado; la misma nos calcula fácilmente las matrices de incidencia, la existencia de interbloqueos y nos permite ver las posibles transiciones en forma de animación.

1.1. Modelado del sistema con una red de Petri

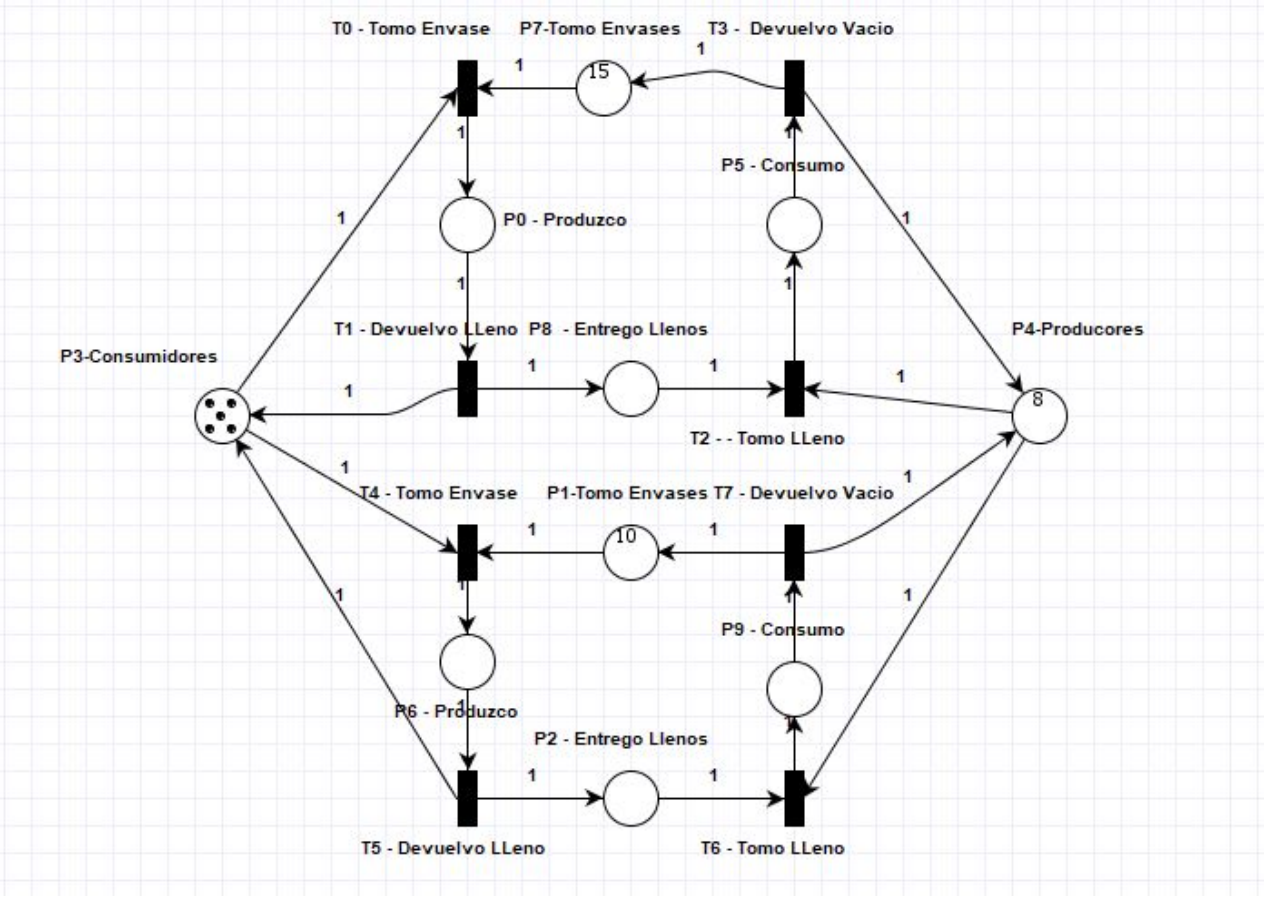

figura 1. Red de petri del sistema

1.2. Plazas Invariantes

Petri net invariant analysis results

T-Invariants

T0PrB1	T1PrB1	T0CoB1	T1CoB1	T0PrB2	T1PrB2	T0CoB2	T1CoB2
1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1

The net is covered by positive T-Invariants, therefore it might be bounded and live.

P-Invariants

P0	PProducB1	PConsB2	P11	P1B1	P0B1	PConsB1	P5	P6	PProducB2	P1B2	P0B2
1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	0
0	1	0	0	1	1	1	0	0	0	0	0
0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	1	1	1

The net is covered by positive P-Invariants, therefore it is bounded.

P-Invariant equations

M(P0) + M(PProducB1) = 1
M(PConsB2) + M(P11) = 1
M(PProducB1) + M(P1B1) + M(P0B1) + M(PConsB1) = 10
M(PConsB1) + M(P5) = 1
M(P6) + M(PProducB2) = 1
M(PConsB2) + M(PProducB2) + M(P1B2) + M(P0B2) = 15

1.3. Matrices de Marcado e Incidencia.

Forwards incidence matrix I^+

	T0	T1	T2	T3	T4	T5	T6	T7
P0	1	0	0	0	0	0	0	0
P1	0	0	0	0	0	0	0	1
P2	0	0	0	0	0	1	0	0
P3	0	1	0	0	0	1	0	0
P4	0	0	0	1	0	0	0	1
P5	0	0	1	0	0	0	0	0
P6	0	0	0	0	1	0	0	0
P7	0	0	0	1	0	0	0	0
P8	0	1	0	0	0	0	0	0
P9	0	0	0	0	0	0	1	0

Backwards incidence matrix *f*

	T0	T1	T2	T3	T4	T5	T6	T7
P0	0	1	0	0	0	0	0	0
P1	0	0	0	0	1	0	0	0
P2	0	0	0	0	0	0	1	0
P3	1	0	0	0	1	0	0	0
P4	0	0	1	0	0	0	1	0
P5	0	0	0	1	0	0	0	0
P6	0	0	0	0	0	1	0	0
P7	1	0	0	0	0	0	0	0
P8	0	0	1	0	0	0	0	0
P9	0	0	0	0	0	0	0	1

Combined incidence matrix *I*

	T0	T1	T2	T3	T4	T5	T6	T7
P0	1	-1	0	0	0	0	0	0
P1	0	0	0	0	-1	0	0	1
P2	0	0	0	0	0	1	-1	0
P3	-1	1	0	0	-1	1	0	0
P4	0	0	-1	1	0	0	-1	1
P5	0	0	1	-1	0	0	0	0
P6	0	0	0	0	1	-1	0	0
P7	-1	0	0	1	0	0	0	0
P8	0	1	-1	0	0	0	0	0
P9	0	0	0	0	0	0	1	-1

Marking

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
Initial	0	10	0	5	8	0	0	15	0	0
Current	0	10	0	5	8	0	0	15	0	0

Enabled transitions

T0	T1	T2	T3	T4	T5	T6	T7
yes	no	no	no	yes	no	no	no

Representando anteriormente el conjunto de matrices de marcado, incidencia y transiciones sensibilizadas.

1.4. Sifones y Trampas

A continuación se muestra una captura de pantalla de los sifones y trampas que la herramienta Pipe nos brinda:

T-Invariants

T0	T1	T2	T3	T4	T5	T6	T7
1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1

The net is covered by positive T-Invariants, therefore it might be bounded and live.

P-Invariants

P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
0	1	1	0	0	0	1	0	0	1
1	0	0	1	0	0	1	0	0	0
0	0	0	0	1	1	0	0	0	1
1	0	0	0	0	1	0	1	1	0

The net is covered by positive P-Invariants, therefore it is bounded.

P-Invariant equations

M(P1) + M(P2) + M(P6) + M(P9) = 10
M(P0) + M(P3) + M(P6) = 5
M(P4) + M(P5) + M(P9) = 8
M(P0) + M(P5) + M(P7) + M(P8) = 15

1.5. Análisis del resultado de estados Classification:

De la siguiente captura de pantalla de la herramienta PIPE podemos ver que nuestra red es limitada, no es segura y nunca llegaremos a una marca tal que no se pueda disparar ninguna transición.

Petri net classification results

State Machine	false
Marked Graph	false
Free Choice Net	false
Extended Free Choice Net	false
Simple Net	true
Extended Simple Net	true

State Machine (false): Nuestra red no tiene exactamente una entrada y una salida
Marked Graph (false): Nuestra red no es marcada; los grafos marcados pueden representar paralelismo, concurrencia y sincronización, pero no conflicto o decisiones dependientes de datos.
Free Choice Net (false): nuestra red no tiene una unica plaza de entrada de una transición o no hay como mucho una transición que tiene p como una plaza de entrada.
Extended Free Choice Net (false): Nuestra red no es de libre elección, debido a que ni los productores ni los consumidores pueden elegir a qué buffer ir.
Simple Net (True): nuestra red es simple; es aquella en la que cada transición tiene como máximo una plaza de entrada compartida con otra transición.

Petri net state space analysis results

Bounded	true
Safe	false
Deadlock	false

Boundead (true): nuestra red es limitada
Safe (false): Nuestra red de Petri es segura o binaria si en los marcados del árbol de alcanzabilidad sólo aparecen 0 y 1.
Deadlock (false): nuestra red no se va a bloquear de manera permanetne; deadlock es el bloqueo permanente de un conjunto de procesos en un sistema concurrente que compiten por recursos del sistema o bien se comunican entre ellos.

2. Diagrama de clases se adjunta en un archivo

3. Diagrama de secuencias se adjunta en un archivo