Fundamentos de Ciencias de la Computación

Clase 20:
Redes de Petri

Primer Cuatrimestre de 2005
Departamento de Cs. e Ing. de la Computación
Universidad Nacional del Sur
Bahía Blanca, Argentina

Redes de Petri

Las redes de Petri son un modelo inherentemente no determinista de la noción de "computación".

Fueron desarrolladas por Carl Petri en 1962, y pensadas como mecanismo para formalizar sistemas.

Utilidad principal: situaciones en las que más de un proceso se desarrolla en paralelo, o con cierta interdependencia temporal.

Algo sobre Carl Adam Petri

- Carl Petri nació el 12 de julio de 1926, en Leipzig (Alemania). Estudió Matemática en la Univ. de Hannover, donde fue profesor.
- Fue el primero en definir el lenguaje de las redes de Petri. Lo hizo en su Tesis Doctoral "Kommunikation mit Automaten" (Comunicación con autómatas), en 1962 (Univ. de Darmstadt).
- Fue nombrado miembro de la Academia de Ciencias de N.York, y recibió la distinción Werner Von Siemens (Alemania).

Ejemplo: concurrencia de ejecución

Considérese el siguiente trozo de código:

```
A:=1;
B:=2;
C:=3;
A:=A+1;
C:=B+C;
B:=A+C;
```

Normalmente, las instrucciones se procesan secuencialmente.

Sin embargo, no hay razones para pensar en ejecutar las tres primeras instrucciones en cualquier orden (o las tres simultáneamente) pues son independientes.

Aplicaciones de Redes de Petri

- Lenguajes de programación concurrentes
- Bases de datos compartidas o distribuidas
- Diseño de arquitecturas de computadoras

Red de Petri: definición

Una Red de Petri (RP) es una 4-upla (P,T,IF,OF) donde:

$P = \{p_1,p_2 \ldots p_m\}$ es un conjunto finito de "lugares"

$T=\{t_1,t_2,\ldots t_n\}$ es un conj. finito de "transiciones"

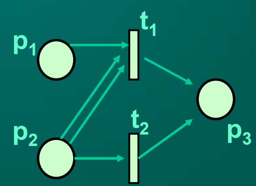
$IF: P \times T \rightarrow Nat$ es una "función de entrada a las transiciones" (*Input Function*)

$OF : T \times P \rightarrow Nat$ es una "función de salida de las transiciones" (*Output Function*)

Red de Petri: ejemplo

Sea RP = (P,T,IF,OF) :
$P = \{p_1, p_2, p_2\}$ *(“lugares”)*
$T=\{t_1,t_2\}$ *(“transiciones”)*

IF: P x T → Nat
Ej: $IF(p_2, t_1)=2$
OF : T x P → Nat
Ej: $IF(t_2, p_3)=1$



Usualmente IF y OF se definen con un cuadro de *doble entrada*, para considerar cada posibilidad.

7

Red de Petri: Representación

RP ⇒ representable con un multidigrafo bipartito G=(N,A), donde N=cjto. nodos, y A=cjto. arcos, tal que $N = P \cup T$, y $P \cap T = \emptyset$.

- Lugares P → nodos circulares
- TransicionesT → nodos barrita
- Para cada par (p,t), trazamos tantos arcos desde p a t como indica IF(p,t).
- Para cada par (t,p), trazamos tantos arcos desde t a p como indica OF(p,t).

8

Red de Petri: Ejemplo

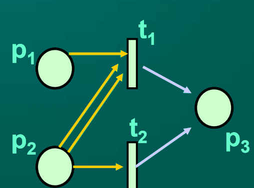
La red de Petri M=(P,T,IF,OF), con
$P=\{p_1,p_2, p_3\}$, y
$T =\{t_1,t_2\}$,
donde IF: PxT→Nat y OF:TxP→ Nat se define como

IF:

P\T	t_1	t_2
p_1	1	0
p_2	2	1
p_3	0	0

OF:

T\P	p_1	p_2	p_3
t_1	0	0	1
t_2	0	0	1



9

Red de Petri: Otro ejemplo

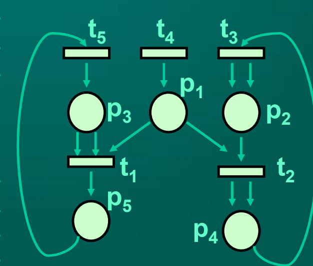
Sea M=(P,T,IF,OF), con
$P=\{p_1,p_2, p_3\, p_4\, p_5\}$
$T =\{t_1,t_2,t_3,t_4,t_5\}$

IF:

P\T	t_1	t_2	t_3	t_4	t_5
p_1	1	1	0	0	0
p_2	0	1	0	0	0
p_3	2	0	0	0	0
p_4	0	0	1	0	0
p_5	0	0	0	0	1

OF:

T\P	p_1	p_2	p_3	p_4	p_5
t_1	0	0	0	0	1
t_2	0	0	0	2	0
t_3	0	2	0	0	0
t_4	1	0	0	0	0
t_5	0	0	1	0	0



10

Red de Petri: Comportamiento

Idea intuitiva: los *lugares* se pueden “cargar” con marcas o “tokens”, que representan la cantidad de recursos disponibles.

Def.: Un marcado de una red de Petri es un mapeo M: P→Nat ∪ {0}

Con n pequeño: marcamos lugar p con M(p) puntos,
Con n grande: se indica el valor n asociado.

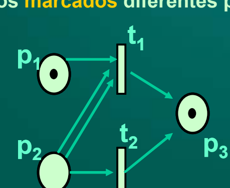
n=3 n=27

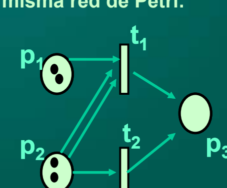
11

Red de Petri: Comportamiento

Def.: Un marcado de una red de Petri es un mapeo M: P→Nat ∪ {0}

Ejemplo:
Dos marcados diferentes para la misma red de Petri:





12

Red de Petri: Comportamiento

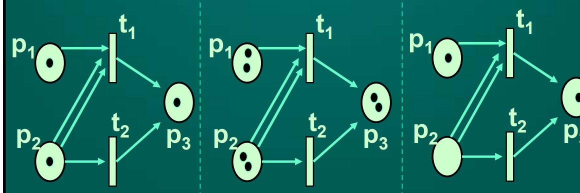
Lugares ⇔ condiciones
Transiciones ⇔ eventos

- Lugar de entrada para t: existe un arco *(p,t)*
- Lugar de salida para t: existe un arco *(t,p)*
- Activación (o habilitación) de t: Si cada lugar de entrada p para una transición t tiene <u>al menos</u> tantos tokens como arcos (p,t), entonces decimos que t está activada.
- Descarga (o disparo) de t: elimina un token de cada lugar de *entrada* de t y agrega un token en cada lugar de *salida* de t.

13

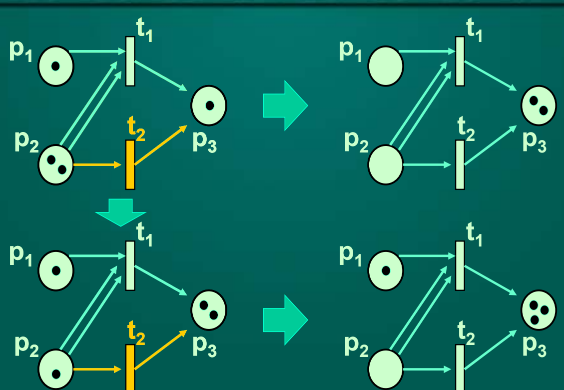
Red de Petri:Comportamiento

t_1 no activada	t_1 activada	t_1 no activada
t_2 activada	t_2 activada	t_2 no activada



14

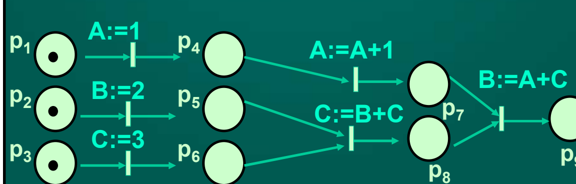
Comportamiento - *Descarga de Transiciones*



15

Ejemplo: Condiciones para ejecución de asignaciones

A:=1;
B:=2;
C:=3;
A:=A+1;
C:=B+C;
B:=A+C;



16

RP: Definiciones formales

Def.: Para un marcado **M** de una RP, y una transición **t** ∈T, **t** se dirá **activada/habilitada** por **M** sssi para cada lugar **p** de entrada a **t**, se verifica **M(p) ≥IF(p,t).**

Def.: Sean **M**, **M'** dos marcados para una RP. Diremos que **M** **evoluciona en M'**, **M|-M'**, sssi existe una transición **t** habilitada por **M** tal que al descargar **t** se obtiene el marcado **M'**.

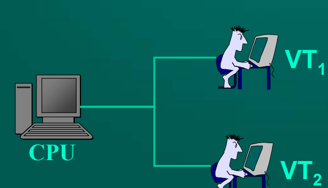
Si M_1|-M_2 |-... |- M_k ent. M_k es **alcanzable** desde M_1

El **disparo** de **t** consiste en el cambio de marcado siguiente:

$$M'(p)=M(p)-IF(p,t)+OF(t,p)$$

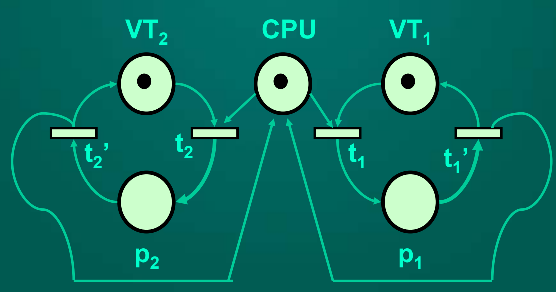
Ej: modelando un sistema de cómputos

- Se tiene una CPU y dos terminales VT_1 y VT_2.
- Cada terminal introduce una tarea por vez en el sistema.
- La CPU sólo puede correr una tarea de alguna terminal.



Atención: hay algunos errores en ejemplos del apunte.

Representación Gráfica de un sistema de cómputos



19

Representación Gráfica de un sistema de cómputos

IF:

P\T	t_1	t_1'	t_2	t_2'
CPU	1	0	1	0
VT_1	1	0	0	0
VT_2	0	0	1	0
p_1	0	1	0	0
p_2	0	0	0	1

OF:

T\P	c	vt_1	vt_2	p_1	p_2
t_1	0	0	0	1	0
t_1'	1	1	0	0	0
t_2	0	0	0	0	1
t_2'	1	0	1	0	0

Lo anterior puede modelarse con una RP M=(P,T,IF,OF,M), con

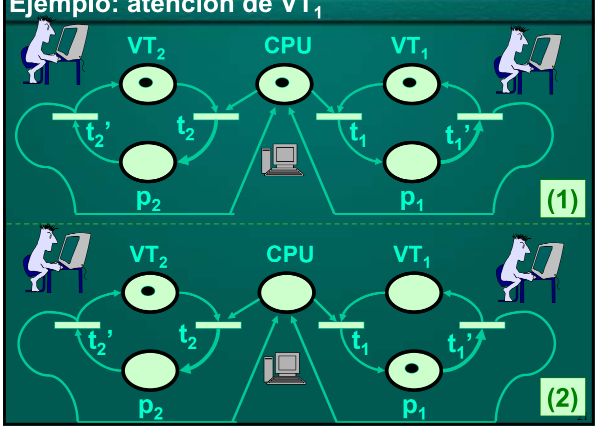
P={CPU,VT_1,VT_2,p_1,p_2}

T ={t_1,t_1' ,t_2 ,t_2'},

$$M(p)=_{def}\begin{cases} M(CPU)=1 \\ M(VT_1)=1 \\ M(VT_2)=1 \\ M(p_1)=M(p_2)=0 \end{cases}$$

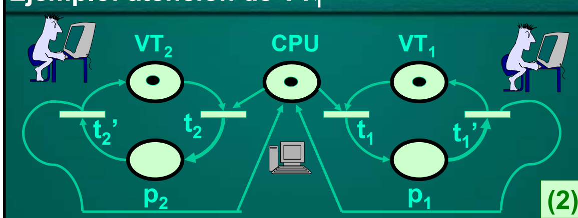
20

Ejemplo: atención de VT_1



21

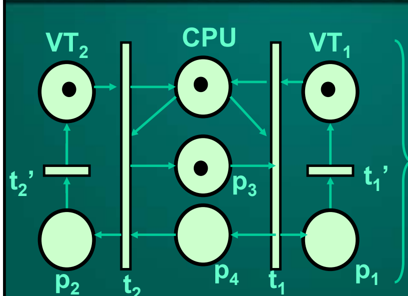
Ejemplo: atención de VT_1



Problema: el sistema operativo puede preferir una terminal, y la otra dejarla desatendida siempre (inanición o *starvation*). *¿Puede lograrse un criterio más justo?*

22

Ejemplo: atención equitativa



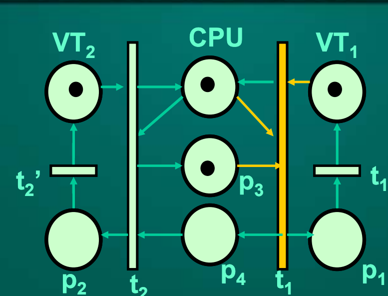
Lugares p_3 y p_4 sincronizan el uso de la CPU

Ejecutar en SIMPRES

Una opción: este modelo de sistema operativo logra mayor equidad para ambas terminales.

23

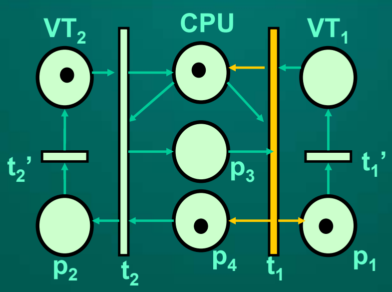
Ejemplo: atención equitativa



Transición habilitada : t_1

24

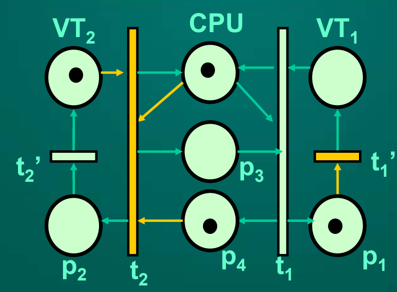
Ejemplo: atención equitativa



Se dispara t_1

25

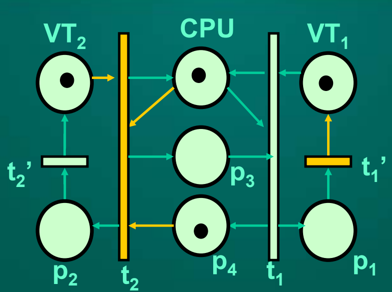
Ejemplo: atención equitativa



Transiciones habilitadas:
t_2 ,t_1'

26

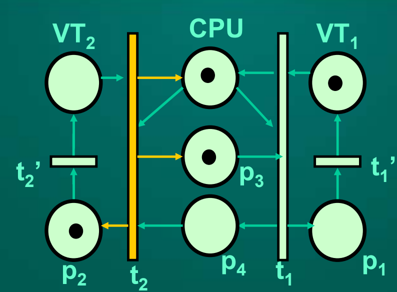
Ejemplo: atención equitativa



Disparada t_1'
Habilitadas:
t_2

27

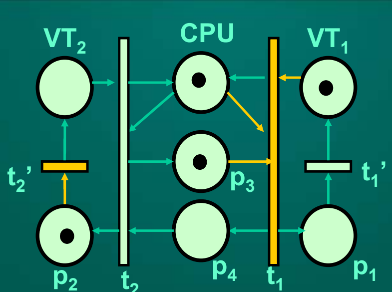
Ejemplo: atención equitativa



Disparada t_2

28

Ejemplo: atención equitativa



Trancisiones habilitadas:
t_1,t_2'

29

Simuladores de RP

En la página web de la materia está disponible SIMPRES, un simulador de RP.

***Mayor información sobre RP*:**

***"Petri Net Theory and the modeling of systems"* (James Peterson, Prentice Hall, 1981).**

30

RPs como reconocedoras de lenguajes

- Las redes de Petri pueden usarse para reconocer lenguajes de varias formas.
- Una forma usual: utilizar "*secuencias de disparos*".

Extendemos la definición anterior:

Def.: Una red de Petri **aceptadora** es una 6-upla (P,T,IF,OF,M_0,M_1), donde P,T,IF y OF se definen como antes, y M_0 y M_1 son marcados denominados inicial y final, respectivamente.

31

RP y lenguajes

En una RP, las acciones se modelan con transiciones. *Ejemplo: t_1, t_3, t_5.*

El conjunto T* de todas las secuencias de transiciones permitibles caracteriza una RP, y por tanto al sistema que modela.

Podemos establecer una analogía:

- Sec. de transiciones $S \in T^* \Leftrightarrow$ cadena
- Conjunto de cadenas = lenguaje
- Dos RPs serán **equivalentes** si denotan lenguajes equivalentes.

32

Secuencia de disparos

Def.: Una **secuencia de disparos S** para una red de Petri aceptadora $R=\{P,T,IF,OF,M_0,M_1\}$ es un elemento de T*.

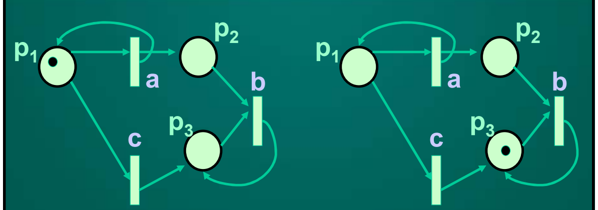
Una secuencia $S= t_1, t_2, \ldots t_n$ es aceptada por R sssi existen n-1 marcados $M_1,M_2,\ldots M_{n-1}$ de P tal que $M_0 \mid\!- M_1 \mid\!- \ldots M_{n-1} \mid\!- M_f$ por el disparo de $t_1 \ldots t_n$ respectivamente.

El lenguaje de secuencias de disparo de R es

$$L(R) = \{S|S \in T^* \text{ y S es aceptada por R}\}$$

33

Ejemplo: Secuencia de disparos



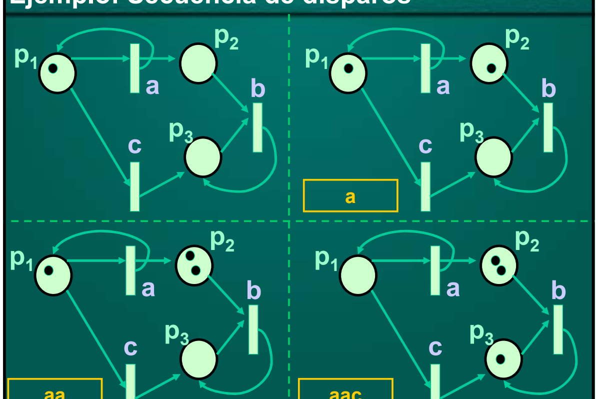
Marcado inicial de R ---- Marcado final de R

Transiciones = {a,b,c}

Sec. de disparos aceptada: c

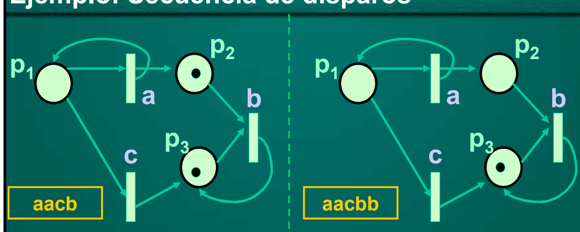
34

Ejemplo: Secuencia de disparos



35

Ejemplo: Secuencia de disparos



Disparamos en este caso: aacbb

Sec. de disparos aceptadas: c, acb, aacbb, etc.

Puede generalizarse $L(R) = \{a^ncb^n|n>=0\}$

36

Etiquetado de una Red de Petri

El etiquetado de una RP permite vincular transiciones y elementos asociados a ellas mediante una correspondencia. Etiquetados más comunes:

- Etiquetado arbitrario: mapeo Φ: $T \rightarrow \Sigma \cup \{\lambda\}$
- Etiquetado libre de λ: no permite usar λ como etiq.
- Etiquetado libre: tal que si $\Phi(t_i) = \Phi(t_j)$, entonces $t_i = t_j$, es decir Φ es inyectiva.

Informalmente, todas las transiciones deben recibir etiquetas distintas

Clase L^λ

Los lenguajes reconocibles via Redes de Petri involucran distintas clases, cuyo análisis detallado escapa al alcance de este curso.

Nos focalizaremos en la clase L^λ (etiquetado arbitrario). La misma constituye la clase más general de los lenguajes reconocibles con RP.

Lema: $L^\lambda \supseteq L \supseteq L^f$

donde

L^f= clase de lenguajes obtenibles via etiquetado libre y

L = clase de lengs. c/etiq. arbitrario libre de λ.

Propiedades de clausura de L^λ

Teo.: los lenguajes de la clase L^λ son cerrados bajo unión, concatenación, intersección, homomorfismo finito y sustitución finita. No lo son bajo estrella de Kleene, complemento, y sustitución generalizada.

Notar que sólo consideramos sustitución finita, que permite reemplazar elementos del lenguaje por lenguajes finitos.

Ser cerrado bajo sustitución finita es un hecho relevante al momento de modularizar problemas.

Expresividad de Redes de Petri

Una pregunta natural: ¿cómo se relacionan las RPs con la Jerarquía de Chomsky?

Teo.: todo lenguaje regular es aceptable mediante una red de Petri.

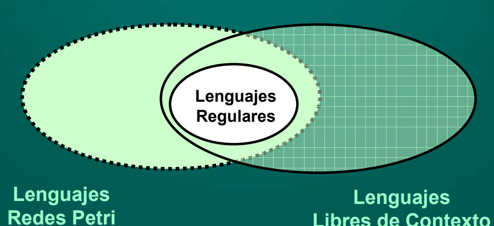
Si bien hay lenguajes LC que son reconocibles via una red de Petri, también hay otros que no lo son. Ejemplo: $L=\{ww^r | w \in \Sigma^*\}$

Ejercicio: mostrar que $L = \{a^n b^n | n>1\}$ puede reconocerse via una red de Petri.

Teo.: existen lenguajes $\in$ Clase LLC que no son aceptables mediante una red de Petri.

Expresividad de Redes de Petri

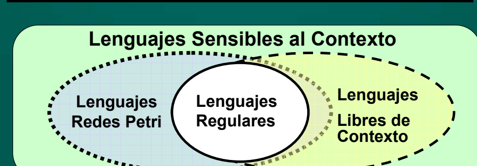
Los resultados anteriores pueden ser graficados como sigue:



Lenguajes de RP y Lenguajes SC

Puede probarse que $L = \{a^n b^n c^n | n>1\}$ (que es sensible al contexto) es reconocible por una Red de Petri (ejercicio).

Teo.: todo lenguaje de Red de Petri es sensible al contexto.



Aplicaciones

Las RP se usan ampliamente en Computación cuando se intenta modelar sistemas concurrentes. Algunas situaciones típicas:

- Lenguajes de programación concurrentes
- Bases de Datos distribuidas
- Diseño de Arquitecturas
- Diversas aplicaciones industriales.

43

Aplicaciones y extensiones

Existen modelos de RP que extienden la propuesta original. Mencionaremos algunos brevemente:

- Redes de Petri coloreadas (permiten distinguir tokens por variedades)
- Redes de Petri con probabilidades (asociadas al disparo de transiciones)
- Redes de Petri con demora en las transiciones

44

Aplicaciones y extensiones

Existen programas (disponibles en Internet) para determinar si una red dada puede alcanzar o no un cierto marcado final a partir de uno inicial.

Redes de Petri para simulación: construir primero un modelo a través de una red, y cuando se obtienen las propiedades deseadas, pasar a la construcción del sistema definitivo.

45