Tema 8: MAQUINAS DE ESTADO FINITO

Ingeniería de Protocolos y Servicios

8.1. Introducción

En un nivel bajo de abstracción, un protocolo puede verse como una máquina de estado finito.

El estado de un protocolo define qué acciones se permiten a cada proceso, qué eventos esperan, y cómo responden a dichos eventos.

El modelo de máquina de estado finito comunicante es muy importante en tres áreas de diseño de protocolos: validación formal, síntesis de protocolos y test de conformidad.

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

8.2. Descripción informal

Normalmente una máquina de estado finito se especifica en forma de tabla de transiciones, p.ej.

Condición		Efecto	
Estado Actual	IN	OUT	Siguiente Estado
Q0	-	1	Q2
Q1	-	0	Q0
Q2	0	0	Q3
Q2	1	0	Q1
Q3	0	0	Q0
Q3	1	0	Q1

8.2. Descripción informal

La tabla especifica el conjunto de transiciones válidas entre estados.

Las 2 primeras columnas son las condiciones que deben ser satisfechas para que la transición pueda llevarse a cabo, que son el estado actual en que la máquina debe estar y una condición del entorno (p.ej. una señal de entrada).

8.2. Descripción informal

Las 2 últimas columnas definen los efectos de la transición, especificando cómo cambiaría el entorno (p. ej. señal de salida) y el nuevo estado que alcanzaría la máquina.

El entorno de la máquina consiste en 2 conjuntos de señales finitos y disjuntos: señales de entrada y señales de salida.

8.2. Descripción informal

En cada estado hay cero o más reglas que son ejecutables.

Si no hay reglas ejecutables la máquina se dice que está en un estado de terminación.

Si hay una regla ejecutable, la máquina realiza una transición -determinista- hacia el siguiente estado.

8.2. Descripción informal

Si hay dos reglas o más que son ejecutables se selecciona de manera no determinista una opción. Este tipo de máquinas (criterio de elección no definido) se llaman máquinas no-deterministas. P.ej.

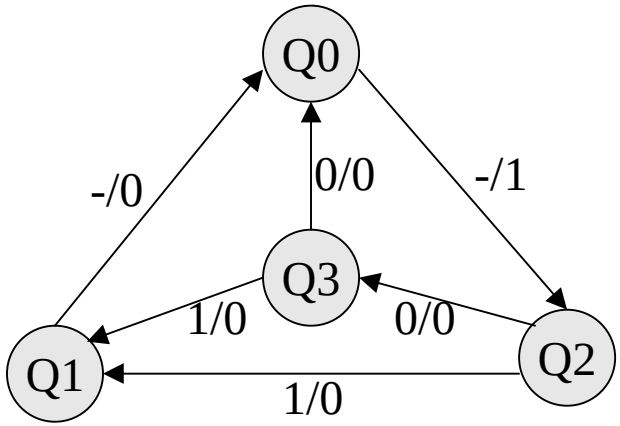
Estado Actual	IN	OUT	Siguiente Estado
Q1	-	0	Q0
Q1	0	0	Q3

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.2. Descripción informal

Una máquina de estado finito también puede representarse gráficamente mediante un diagrama de transición de estados (ejemplo determinista)



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.2. Descripción informal

Los círculos representan los estados y las flechas las transiciones. Las etiquetas de las flechas son del tipo *c/e*, donde *c* representa una condición de transición (p.ej. un valor de entrada) y *e* el correspondiente efecto (p.ej. un nuevo valor de salida).

8.2. Descripción informal

MÁQUINAS DE TURING

Si el entorno ya no se considera finito se obtiene el modelo de máquina de Turing. Éste consiste en una cinta de longitud infinita, compuesta por una secuencia de cuadrados que a su vez contienen un conjunto finito de símbolos. La máquina puede leer y escribir de manera secuencial en un cuadrado, y moverse a izquierda o derecha. Inicialmente la cinta está vacía y se apunta a un cuadrado cualquiera.

8.2. Descripción informal
MÁQUINAS DE TURING
Ejemplo: *Busy Beaver* (castor atareado)

Condición		Efecto	
Estado Actual	IN	OUT/ MOVE	Siguiente Estado
Q0	0	1/L	Q1
Q0	1	1/R	Q2
Q1	0	1/R	Q0
Q1	1	1/L	-
Q2	0	1/R	Q1
Q2	1	1/L	Q3
Q3	-	-	-

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.2. Descripción informal

MÁQUINAS DE TURING

Ejemplo: *Busy Beaver* (castor atareado). Esta máquina tiene 2 señales de salida: Una sobreescribe el cuadrado de la cinta y la otra reposiciona la cinta. El estado Q3 es un estado de terminación.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.2. Descripción informal

MÁQUINAS DE ESTADO FINITO COMUNICANTES

Resultan de interconectar las salidas y entradas de una máquina de estado. Dichas máquinas ejecutan un algoritmo en 2 pasos:

- Primero, se inspecciona el valor de entrada y se selecciona una regla ejecutable.
- Segundo, la máquina cambia de estado y actualiza las salidas en concordancia con las reglas de transición.

8.2. Descripción informal

MÁQUINAS DE ESTADO FINITO COMUNICANTES

Dicho algoritmo se ejecuta para siempre. Un ejemplo serían las transiciones Q0 Q2 Q1 para siempre de la tabla inicial.

Se pueden construir sistemas elaborados utilizando la interacción entre máquinas de estado finito comunicantes, conectando las salidas de una con las entradas de otra.

8.2. Descripción informal

ACOPLAMIENTO ASÍNCRONO

Las máquinas están acopladas mediante colas de mensajes FIFO.

Si una cola de entrada está vacía la transición será no ejecutable.

Si una cola de salida está llena la transición que provocaría un mensaje hacia dicha cola es no ejecutable.

8.2. Descripción informal

ACOPLAMIENTO ASÍNCRONO

Se restringe el modelo a un evento de sincronismo por regla de transición: una regla especifica una entrada o una salida (no ambas).

Ejemplo: Protocolo de bit alternado con *timeouts*.

DTIC

Área Ingeniería Telemática

francesc.burrull@upct.es

8.2. Descripción informal
ACOPLAMIENTO ASÍNCRONO

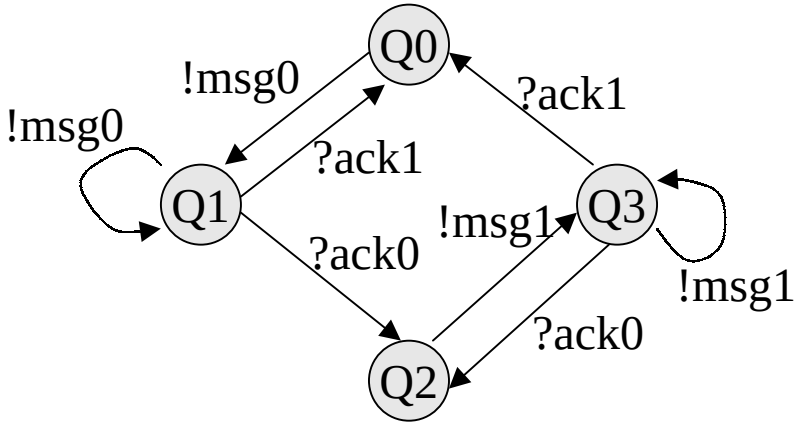
Estado	IN	OUT	Siguiente
Q0	-	msg0	Q1
Q1	ack1	-	Q0
Q1	ack0	-	Q2
Q2	-	msg1	Q3
Q3	ack0	-	Q2
Q3	ack1	-	Q0
Q1	-	msg0	-
Q3	-	msg1	-

Emisor

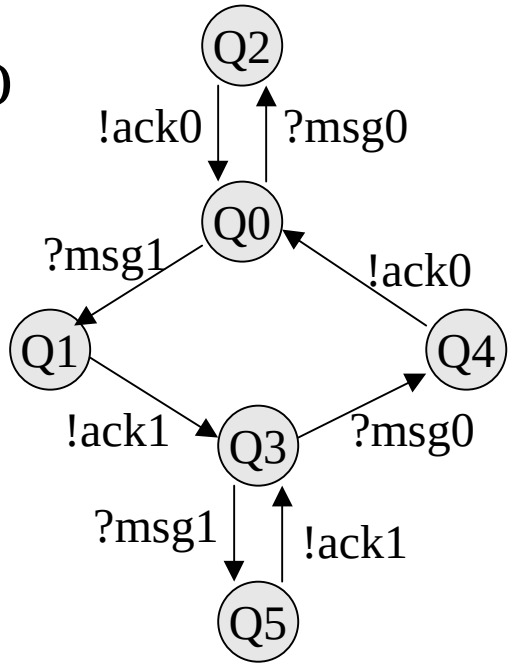
Estado	IN	OUT	Siguiente
Q0	msg1	-	Q1
Q0	msg0	-	Q2
Q1	-	ack1	Q3
Q2	-	ack0	Q0
Q3	msg0	-	Q4
Q3	msg1	-	Q5
Q4	-	ack0	Q0
Q5	-	ack1	Q3

Receptor

8.2. Descripción informal
ACOPLAMIENTO ASÍNCRONO



Emisor



Receptor

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

8.2. Descripción informal

ACOPLAMIENTO SÍNCRONO

Las condiciones de transición son “selecciones” que la máquina puede hacer para comunicarse.

La máquina, de nuevo, puede seleccionar o bien una salida o bien una entrada.

Para hacer un movimiento una señal debe ser seleccionada por 2 máquinas simultáneamente, como entrada por una y salida por la otra.

8.2. Descripción informal
ACOPLAMIENTO SÍNCRONO
Ejemplo: Semáforos.

Estado	IN	OUT	Siguiente
Q0	P	-	Q1
Q1	-	V	Q0

Usuario

Estado	IN	OUT	Siguiente
Q0	-	P	Q1
Q1	V	-	Q0

Servidor

Nótese que si se crean 1 servidor y 2 usuarios, éstos nunca podrán de manera simultánea estar en Q1

8.3. Descripción formal

Una cola de mensajes es la terna (S, N, C) donde:

- S es un conjunto finito, llamado vocabulario.
- N es un entero que indica el número de posiciones de la cola
- C es el contenido de la cola, un conjunto ordenado de elementos de S.

Los elementos de S y C se llaman mensajes.

8.3. Descripción formal

Sea M el conjunto de todas las colas de mensajes, sea el superíndice $1 \leq m \leq M$ el identificador de una cola y el subíndice $1 \leq n \leq N$ la posición n-ésima de una cola.

C_n^m es el mensaje n de la cola m.

El vocabulario del sistema V será la unión de todos los vocabularios más el elemento nulo ε.

Para un conjunto de M colas (de 1 a M) se define V como:

$$V = \bigcup_{m=1}^{M} S^m \bigcup \varepsilon$$

8.3. Descripción formal

Una máquina de estado finito comunicante es una tetra (Q, q_0, M, T) donde:

- Q es un conjunto finito no vacío de estados,
- q_0 es un elemento de Q, el estado inicial,
- M es un conjunto de colas de mensajes, y
- T es una relación de transición de estados.

T(q,a): q es el estado actual. a es una acción.

Se permiten 3 tipos de acciones: entradas, salidas y acción nula ε.

8.3. Descripción formal

La relación de transición T define un conjunto de cero o más posibles estados sucesores del estado q del conjunto Q. Si no se modela el no-determinismo, habrá un estado sucesor. Si T(q,a) no está explícitamente definido se supone $T(q,a)=\emptyset$.

$T(q,\varepsilon)$ indica una transición espontánea.

francesc.burrull@upct.es

8.4. Ejecución de máquinas

• Poner todas las máquinas al estado inicial con las colas de mensajes vacías.

$\forall i,\ 1 \leq i \leq P \Rightarrow q^i = q_0^i$ y $\forall i,\ 1 \leq i \leq M \Rightarrow C^i = \emptyset$

• Seleccionar una máquina arbitraria *i* y una regla de transición arbitraria T^i tal que $T^i(q,a) \neq \emptyset$, con a ejecutable, y ejecutarla.

• Si no quedan reglas ejecutables, el algoritmo termina.

8.4. Ejecución de máquinas

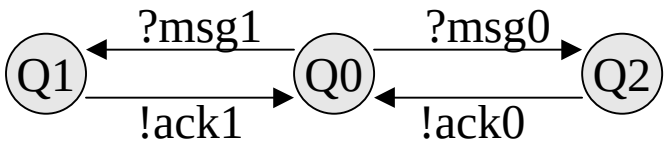
La acción *a* puede ser una entrada, una salida o ε. Sea $1 \leq d(a) \leq M$ la cola destino de una acción *a*, y sea *m(a)* el mensaje enviado o recibido, $m(a) \in S^{d(a)}$. Sea N^i el número de posiciones en la cola de mensajes *i*. Para ver si *a* es ejecutable basta con cumplir una regla:

- $a = \varepsilon$
- *a* es entrada y $m(a) = C_1^{d(a)}$
- *a* es salida y $C^{d(a)} < N^{d(a)}$

8.5. Minimización de máquinas

Considérese la siguiente máquina:

Condición		Efecto	
Estado	IN	OUT	Siguiente
Q0	msg1	-	Q1
Q0	msg0	-	Q2
Q1	-	ack1	Q0
Q2	-	ack0	Q0



8.5. Minimización de máquinas

Esta máquina es equivalente a la del receptor del protocolo de bit alternado porque PUEDE generar la misma secuencia de símbolos de salida si se le ofrece la misma secuencia de símbolos de entrada.

Dos estados dentro de una máquina son equivalentes si la máquina puede iniciarse en cualquiera de estos 2 estados y puede generar el mismo conjunto de símbolos de salida para cualquier secuencia test de símbolos de entrada.

8.5. Minimización de máquinas

Para la minimización de máquinas se estudiará Q (conjunto de estados) y T (conjunto de transiciones), considerando M (colas de mensajes) como parte del entorno, no de la máquina.

Este planteamiento genera restricciones mayores al comportamiento de la máquina (impone a los estados restricciones de comportamiento mayores), generando además máquinas equivalentes con menos estados.

8.5. Minimización de máquinas

ALGORITMO DE MINIMIZACIÓN DE *FSM*

• Definir un array *E* de QxQ valores booleanos. Inicialmente poner a *true* todos los elementos *E[i,j]* que cumplan: $\forall a,\ T(i,a)\neq\emptyset \Leftrightarrow T(j,a)\neq\emptyset$

• De los elementos *E[i,j] true*, cambiar su valor a: $\forall a,\ E[T(i,a),E(j,a)]$ Es decir, 2 estados no son equivalentes si sus sucesores no lo son.

• Repetir el punto anterior hasta que no se pueda incrementar el número de *false*.

8.5. Minimización de máquinas

Ejemplo: Algoritmo recepción bit alternado

	Q0	Q1	Q2	Q3	Q4	Q5
Q0	1	0	0	1	0	0
Q1	0	1	0	0	0	1
Q2	0	0	1	0	1	0
Q3	1	0	0	1	0	0
Q4	0	0	1	0	1	0
Q5	0	1	0	0	0	1

8.6. El problema del test de conformidad

A través de la equivalencia entre dos estados se puede determinar la equivalencia de dos máquinas: Hay que determinar que cada estado de una máquina tiene un estado equivalente en la otra, y viceversa.

Esto es muy importante cuando se aplica a 2 máquinas, siendo una un modelo y la otra una implementación.

Vistas como “cajas negras”, ambas máquinas deben responder igual a señales de entrada.

Este problema se conoce comotest de conformidad.

8.7. Combinación de máquinas

Se trata de obtener una tetra (Q, q_0, M, T) para la máquina combinada de (Q^1, q_0^1, M^1, T^1) y (Q^2, q_0^2, M^2, T^2).

• Definir el producto de estados, $Q = Q^1 \times Q^2$, con $q_0^1 q_0^2$ el estado inicial de la nueva máquina.

• El conjunto de colas de mensajes M será $M^1 \cup M^2$. El vocabulario V será la unión de vocabularios, y *a* la unión de acciones.

• $\forall q^1 q^2$ y $\forall a$, $\Rightarrow T(q^1 q^2, a) = T(q^1, a) \cup T(q^2, a)$

8.8. Máquinas de estado finito extendidas

Mediante el modelo de máquina de estado finito se han obtenido máquinas que intercambian objetos abstractos utilizando el acceso a colas como único mecanismo de sincronismo.

Las máquinas de estado finito extendidas introducen 3 modificaciones: Uso de variables, colas de enteros y operadores aritmético-lógicos sobre variables.

8.8. Máquinas de estado finito extendidas

Una variable puede definirse de manera parecida a una cola, con la diferencia que una variable tiene un único valor, dentro de un rango finito, en un instante dado. El valor de una variable será el último valor introducido. Las variables disponen de nombres simbólicos y contienen, en este caso, enteros (en lugar de objetos abstractos).

8.8. Máquinas de estado finito extendidas

La segunda modificación es el uso de colas para transferir valores enteros (en lugar de objetos abstractos)

La tercera modificación es la introducción de operadores aritméticos y lógicos para manipular el contenido de las variables.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.8. Máquinas de estado finito extendidas

Una variable puede simularse con una máquina de estado finito, siempre que su rango de valores sea finito.

El siguiente ejemplo es una máquina de estado finito que simula una variable con 3 posibles valores.

8.8. Máquinas de estado finito extendidas

Estado	IN	OUT	Siguiente
Q0	S0	-	-
Q0	S1	-	Q1
Q0	S2	-	Q2
Q0	RV	-	R0
R0	-	0	Q0
Q1	S0	-	Q0
Q1	S1	-	-
Q1	S2	-	Q2
Q1	RV	-	R1
R1	-	1	Q1
Q2	S0	-	Q0
Q2	S1	-	Q1
Q2	S2	-	-
Q2	RV	-	R2
R2	-	2	Q2

8.8. Máquinas de estado finito extendidas

La extensión de la máquina de estado finito mediante variables no es más que una conveniencia para el modelado (el precio es un incremento importante del número de estados).

Ahora las transiciones incluyen en las condiciones expresiones booleanas sobre el valor de las variables y en los efectos asignaciones a variables.

8.8. Máquinas de estado finito extendidas

Se puede definir una máquina de estado finito extendida como la quintupla (Q, q_0, M, A, T) donde:

- A es un conjunto de nombres de variables, y
- Q, q_0, M, T como antes.

Se han añadido 2 tipos de acciones extra (a entradas, salidas y acción nula ε): condiciones booleanas sobre elementos de A y asignaciones a elementos de A.

8.8. Máquinas de estado finito extendidas

E/S EXTENDIDA

Se puede definir las acciones de E/S como un conjunto de valores finito y ordenado. Los valores serán expresiones sobre variables de A o simples constantes.

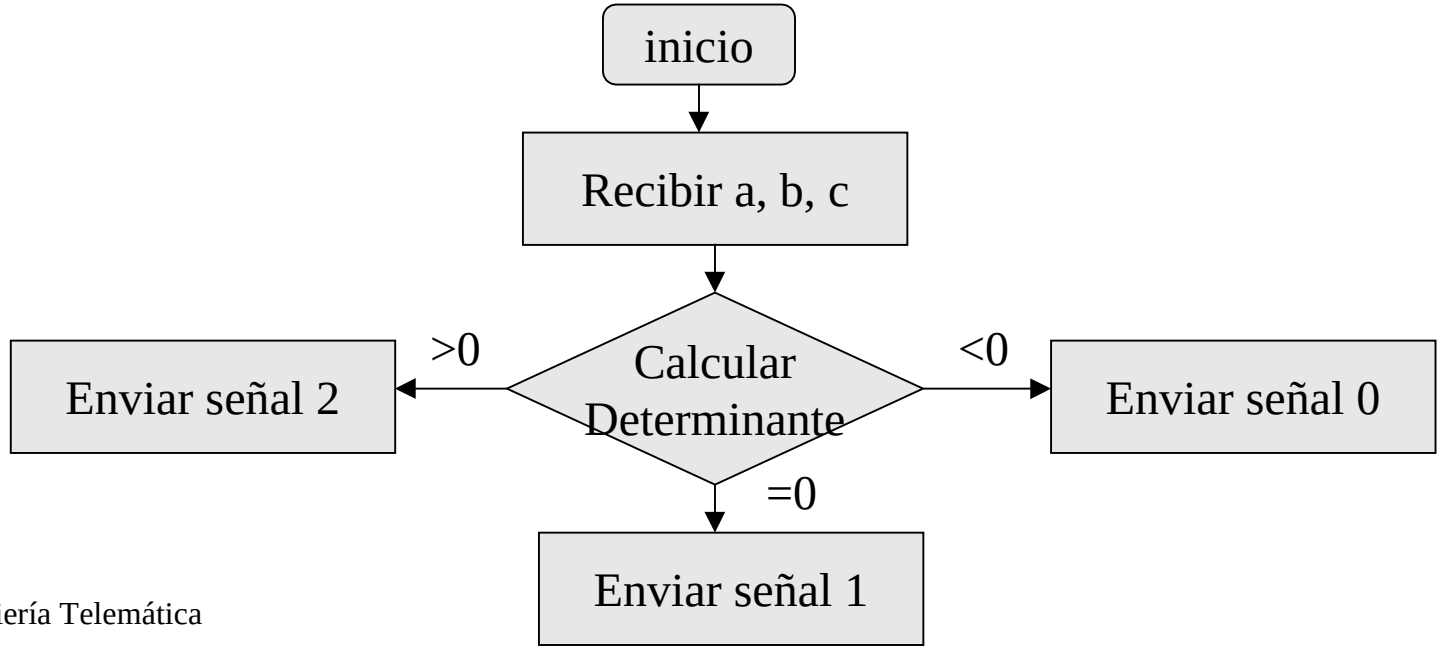
8.8. Máquinas de estado finito extendidas

E/S EXTENDIDA

Ejemplo: obtención máximo común divisor (Euclides)

Estado	Acción	Siguiente
Q0	In?x,y	Q1
Q1	x>y	Q2
Q1	x<y	Q3
Q1	x=y	Q4
Q2	x:=x-y	Q1
Q3	y:=y-x	Q1
Q4	Out!x	Q5
Q5	-	-

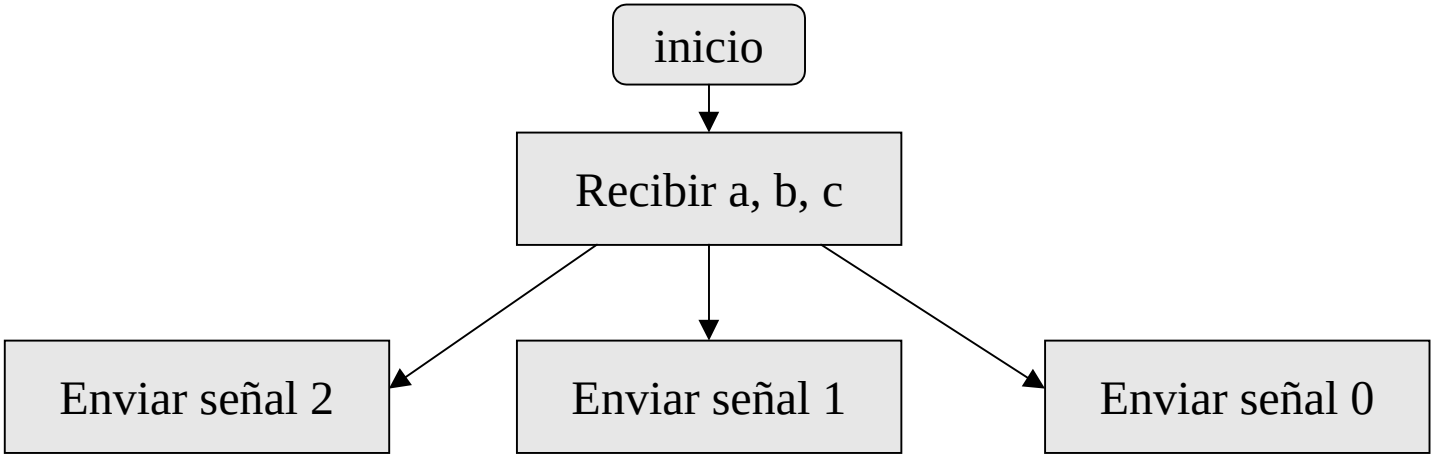
8.9. Generalización de máquinas



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.9. Generalización de máquinas



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.9. Generalización de máquinas

La generalización de máquinas consiste en eliminar el comportamiento interno que no interesa.

La máquina generalizada puede hacer lo mismo que la original, pero puede hacer más.

La máquina generalizada es más general sólo en la manera de generar salida.

8.9. Generalización de máquinas

La utilidad de la generalización de máquinas es el análisis de problemas complejos.

Supóngase un protocolo formado por los módulos A y B, y que se quiere validar el módulo A. Se puede simplificar el módulo B combinando, reduciendo, generalizando y minimizando máquinas. El nuevo módulo B es capaz de comportarse como el módulo B original, pero puede hacer más.

8.9. Generalización de máquinas

Si se puede demostrar que el módulo A es correcto en presencia del nuevo módulo B (más fácil) también será correcto en presencia del módulo B original, ya que su comportamiento es un subconjunto del nuevo comportamiento.

En la validación del módulo A de esta manera se han obtenido 2 beneficios: Se ha realizado un test más duro (se validan condiciones más generales que en el protocolo original) y ha sido más fácil de realizar.

8.10. Modelos restringidos

Para el análisis de protocolos se usan distintas variaciones del modelo de máquina de estado finito, tanto extensiones (mayor poder de modelado) como restricciones (mayor poder analítico) del modelo.

Se verá un caso modelo restringido: Redes de Petri.

8.10. Modelos restringidos

REDES DE PETRI

Son una colección de sitios, transiciones y aristas dirigidas (*places*, *transitions and edges*).

Gráficamente, los sitios son dibujados como círculos y las transiciones como barras o rectángulos. Las aristas del grafo son conocidas como *arcos*.

8.10. Modelos restringidos

REDES DE PETRI

El estado del sistema que la red esté modelando es representado con la asignación de enteros no-negativos a los sitios. Esta asignación es conocida como una *marcación*, la cual es representada gráficamente mediante unos pequeños círculos negros dentro de un sitio *p*, llamados *tokens* . Si el número de tokens es demasiado grande, los *k* tokens son representados con un número no-negativo dentro del correspondiente sitio.

DTIC
Área Ingeniería Telemática
francesc.burrull@upct.es

8.10. Modelos restringidos

REDES DE PETRI

Típicamente, los estados representan algún tipo de condición en el sistema, y una transición representa un evento. Un sitio de entrada (salida) a una transición representan las pre- (post-) condiciones. Los tokens pueden tener muchas interpretaciones, como una condición satisfecha, como un punto de control de flujo del programa, y como un privilegio para proceder la ejecución.

DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

Tema 8: MAQUINAS DE ESTADO FINITO

Ingeniería de Protocolos y Servicios

8.10. Modelos restringidos
REDES DE PETRI

Un token modela un recurso compartido, que puede ser solicitado por más de una transición. Esta abstracción simboliza el cumplimiento de orden parcial dentro de las posibles ejecuciones del problema modelado

DTIC
Área Ingeniería Telemática

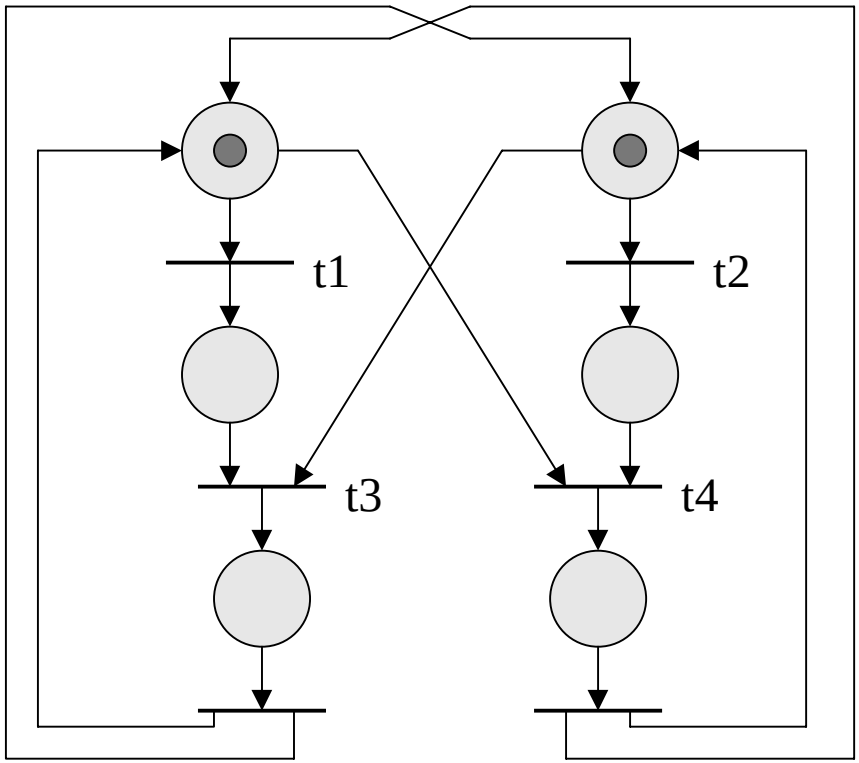
francesc.burrull@upct.es

8.10. Modelos restringidos

REDES DE PETRI

El siguiente ejemplo modela un bloqueo utilizando redes de Petri. Inicialmente las 2 transiciones t1 y t2 están habilitadas. Después de disparar t1 la transición t3 está permitida. Si se dispara t3 no hay problema. Si se dispara t2 la red bloquea.

8.10. Modelos restringidos



DTIC
Área Ingeniería Telemática

francesc.burrull@upct.es

8.11. Resumen

Hay 3 criterios para evaluar lo adecuado de un modelo formal: poder analítico, poder de modelado, y claridad descriptiva.

El objetivo fundamental del modelado es el poder analítico (modelo más fácil de analizar que sistema).

Para alcanzar dicho objetivo se ha escogido el modelo básico de máquina de estado finito, avanzando hacia variaciones del mismo para obtener mayor claridad y poder de modelado.