Modelación de software usando autómatas

Marcelo Arenas

Sistemas de transición (autómata)

En un sistema de transición tenemos:

- Estados
- Propiedades: Pueden ser verdaderas o falsas en un estado del sistema.

  “la luz está encendida”, “la temperatura es menor que 50 grados Celsius”, . . .

- Acciones: Permiten ir de un estado del sistema a otro.

  “apagar la luz”, “subir la temperatura en 1 grado Celsius”, “subir la temperatura a 60 grados Celsius”, . . .

Sistemas de transición: Formalización

Definición

$\mathcal{A} = (Q, E, q_0, \delta, \lambda)$ es un sistema de transición sobre un conjunto finito de propiedades P si:

- *Q es un conjunto finito de estados;*
- *E es un conjunto finito de acciones;*
- *$q_0 \in Q$ es un estado inicial;*
- *$\delta : Q \times E \to 2^Q$ es una función de transición;*
- *$\lambda : Q \to 2^P$ es una función que indica que propiedades son ciertas en cada estado.*

Sistemas de transición: Un ejemplo

Suponga que tiene un sistema con un proceso y dos recursos A y B. Para acceder a cada uno de estos recursos, el proceso debe solicitarlo y esperar que le sea asignado. El proceso puede dejar de usar cada uno de estos recursos cuando lo estime conveniente.

¿Cómo se puede modelar este proceso usando un sistema de transición?

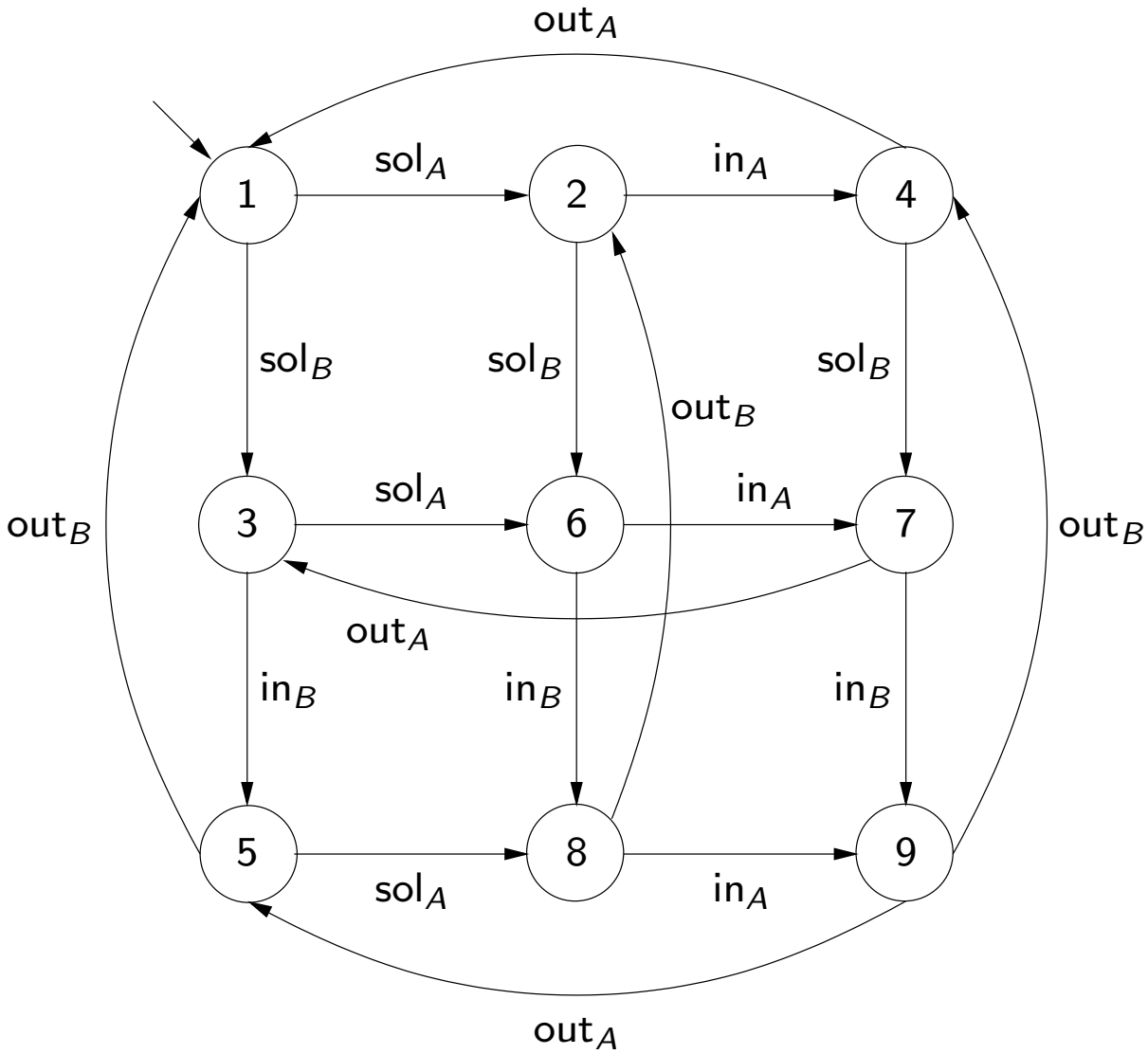
Sistemas de transición: Un ejemplo

Tenemos que:

- $P = \{E_A, E_B, T_A, T_B\}$
- $Q = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- $E = \{\text{sol}_A, \text{sol}_B, \text{in}_A, \text{in}_B, \text{out}_A, \text{out}_B\}$
- $q_0 = 1$
- $\lambda(1) = \emptyset$ $\quad \lambda(4) = \{T_A\}$ $\quad \lambda(7) = \{T_A, E_B\}$
  $\lambda(2) = \{E_A\}$ $\quad \lambda(5) = \{T_B\}$ $\quad \lambda(8) = \{T_B, E_A\}$
  $\lambda(3) = \{E_B\}$ $\quad \lambda(6) = \{E_A, E_B\}$ $\quad \lambda(9) = \{T_A, T_B\}$

Sistemas de transición: Un ejemplo

Función de transición:



Sincronización de sistemas de transición

- En general, un sistema puede estar compuesto por varios procesos que interactúan.
- Cada uno de estos procesos es representado como un sistema de transición.
- ¿Cómo construimos un sistema de transición que represente la interacción de estos procesos?
- Construimos el producto de los sistemas de transición.

Producto de sistemas de transición

Definición

Dados sistemas de transición $\mathcal{A}_i = (Q_i, E_i, q_0^i, \delta_i, \lambda_i)$, *para* $1 \leq i \leq k$, *el producto* $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ *se define como* $\mathcal{A} = (Q, E, q_0, \delta, \lambda)$, *donde:*

- $Q = Q_1 \times \cdots \times Q_k$
- $E = (E_1 \cup \{\star\}) \times \cdots \times (E_k \cup \{\star\})$
- $q_0 = (q_0^1, \ldots, q_0^k)$
- $\delta((q_1, \ldots, q_k), (e_1, \ldots, e_k)) =$

$$\{(q_1', \ldots, q_k') \mid \text{ para todo } i \in \{1, \ldots, k\}, \\ (e_i = \star \text{ y } q_i' = q_i) \text{ o } (e_i \neq \star \text{ y } q_i' \in \delta_i(q_i, e_i))\}.$$

- $\lambda((q_1, \ldots, q_k)) = (\lambda_1(q_1), \ldots, \lambda_k(q_k))$

Producto de sistemas de transición: Algunas observaciones

- El símbolo $\star$ representa la acción "hacer nada".

  Esta acción es usada para representar el hecho de que no todos los sistema $\mathcal{A}_i$ tienen que actuar al mismo tiempo.

- La función λ también puede ser definida como:

$$\lambda((q_1, \ldots, q_k)) = \bigcup_{i=1}^{k} \lambda_i(q_i).$$

  En este caso suponemos que todos los sistemas $\mathcal{A}_i$ están definidos sobre el mismo grupo de propiedades *universales*.

Producto de sistemas de transición: Un ejemplo

Ejercicio: Sean $\mathcal{A}_1$ y $\mathcal{A}_2$ copias del sistema de transición en la página 5. Construya el producto de estos autómatas.

¿Puede el producto $\mathcal{A}_1 \times \mathcal{A}_2$ representar el siguiente sistema?

Suponga que tiene un sistema con **dos** procesos y dos recursos A y B. Para acceder a cada uno de estos recursos, cada proceso debe solicitarlo y esperar que le sea asignado. Cuando un proceso está usando un recurso, el otro no lo puede utilizar. Cada proceso puede dejar de usar cada uno de los recursos cuando lo estime conveniente.

Producto de sistemas de transición: Sincronización

Para sincronizar el producto eliminamos las tuplas de acciones que representen ejecuciones incorrectas.

Sea $\mathcal{A} = (Q, E, q_0, \delta, \lambda)$ el producto $\mathcal{A}_1 \times \cdots \times \mathcal{A}_k$. Para sincronizar los componentes de $\mathcal{A}$, reemplazamos E por:

$$\text{Sinc} \subseteq (E_1 \cup \{\star\}) \times \cdots \times (E_k \cup \{\star\}).$$

Ejercicio: En la transparencia anterior, ¿qué eliminaría del producto $\mathcal{A}_1 \times \mathcal{A}_2$ para sincronizar correctamente los sistemas $\mathcal{A}_1$ y $\mathcal{A}_2$?

Producto de sistemas de transición: Sincronización

La sincronización también puede mejorarse utilizando estados adicionales.

Ejercicio: En el sistema sincronizado construido en la transparencia anterior es posible que un proceso nunca reciba permiso para usar un recurso.

1. ¿Qué algoritmo puede ser utilizado para resolver este problema?
2. ¿Cómo se puede codificar este algoritmo en el producto de los sistemas de transición?

Uso de variables

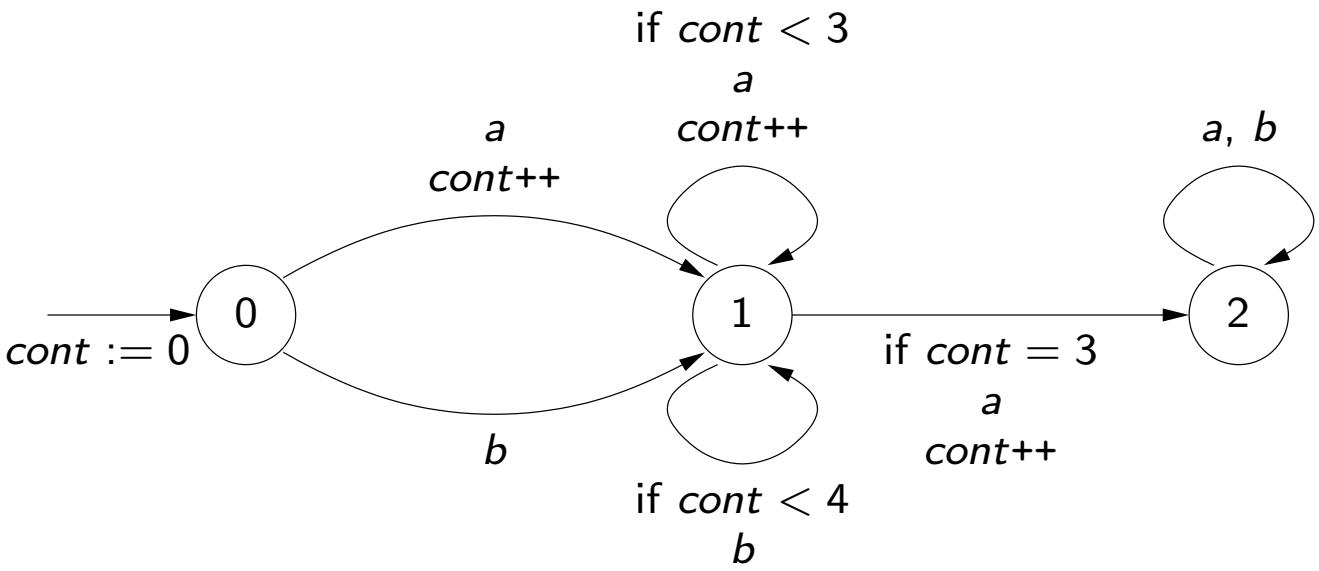
Un sistema de transición es usado para representar el funcionamiento de un programa.

- El proceso de codificación del programa puede ser complicado.

Para simplificar el proceso de codificación se han agregado algunas funcionalidades a los sistemas de transición.

- El uso de variables es una de estas funcionalidades.

Uso de variables: Un primer ejemplo



$$\lambda(0) = \{OK\}$$
$$\lambda(1) = \{OK\}$$
$$\lambda(2) = \{error\}$$

Eliminación de variables

Importante: Para hacer verificación formal tenemos que usar autómatas normales.

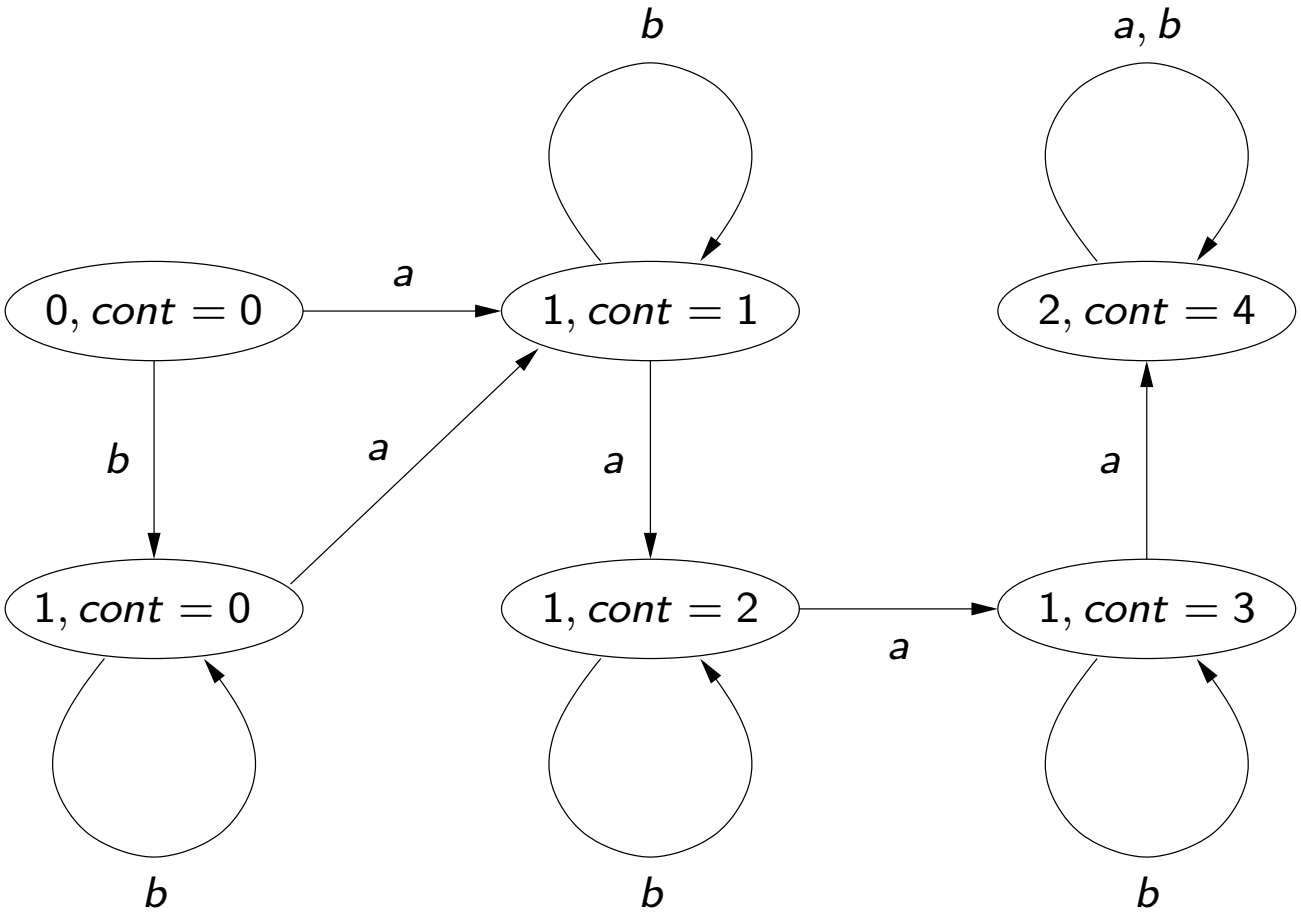
Es necesario eliminar las variables del sistema de transición.

- Este proceso se llama **desenrollar el sistema**.

  Cada estado q es reemplazado por un estado *global* $(q, x_1 = v_1, \ldots, x_n = v_n)$, donde x_1, ..., x_n son las variables del sistema y v_1, ..., v_n son valores concretos para ellas.

- El sistema resultante puede ser de tamaño exponencial en el tamaño del sistema original.

Uso de variables: Desenrollando el primer ejemplo

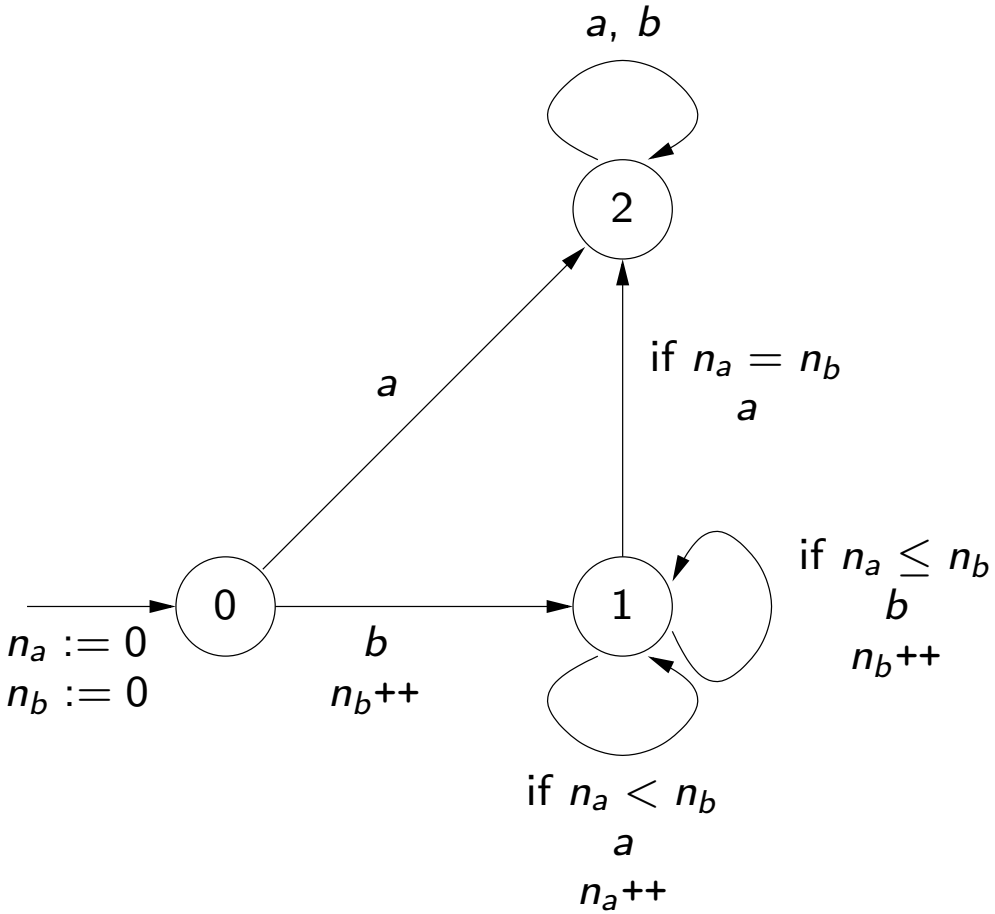


$\lambda((0, cont = 0)) = \{OK\}$

$\lambda((1, cont = i)) = \{OK\}$, para $0 \leq i \leq 3$

$\lambda((2, cont = 4)) = \{error\}$

Uso de variables: Un segundo ejemplo



$\lambda(0) = \{OK\}$

$\lambda(1) = \{OK\}$

$\lambda(2) = \{error\}$

¿Cómo se puede desenrollar este sistema?