Jornada sobre Técnicas de Modelado y Simulación

Sistemas de Eventos Discretos

Ernesto Kofman y Federico Bergero

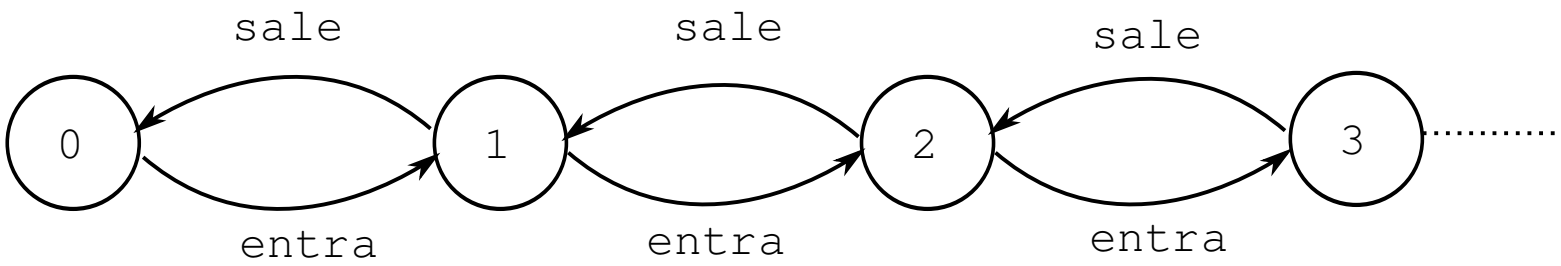
Laboratorio de Sistemas Dinámicos y Procesamiento de la Información
FCEIA - Universidad Nacional de Rosario.
CIFASIS – CONICET. Argentina

Organización de la Presentación

Organización de la Presentación

Ejemplo Introductorio

Consideremos un sistema que cuenta el número de personas que hay en una habitación sensando las personas que entran y salen.



- El estado del sistema es el número de personas en la habitación.
- Cada vez que entra o sale una persona, ocurre un evento y se produce una transición de estado.
- No hay una representación explícita del tiempo en este modelo.

Representación de Modelos de Eventos Discretos

Existen varios formalismos de representación de sistemas por eventos discretos:

- Redes de Petri
- State Graphs (el ejemplo visto anteriormente)
- Grafcet
- DEVS (Discrete EVent system Specification)

DEVS es el formalismo más general, ya que cualquier modelo de eventos discretos puede representarse mediante DEVS.

Organización de la Presentación

Formalismo DEVS

El formalismo DEVS, formulado por Bernard Zeigler a mediados de los 70, permite representar cualquier sistema que tenga un número finito de cambios en un intervalo finito de tiempo.

Un modelo DEVS procesa una secuencia de eventos de entrada y de acuerdo a su condición inicial produce una secuencia de eventos de salida.

DEVS – Modelo Atómico

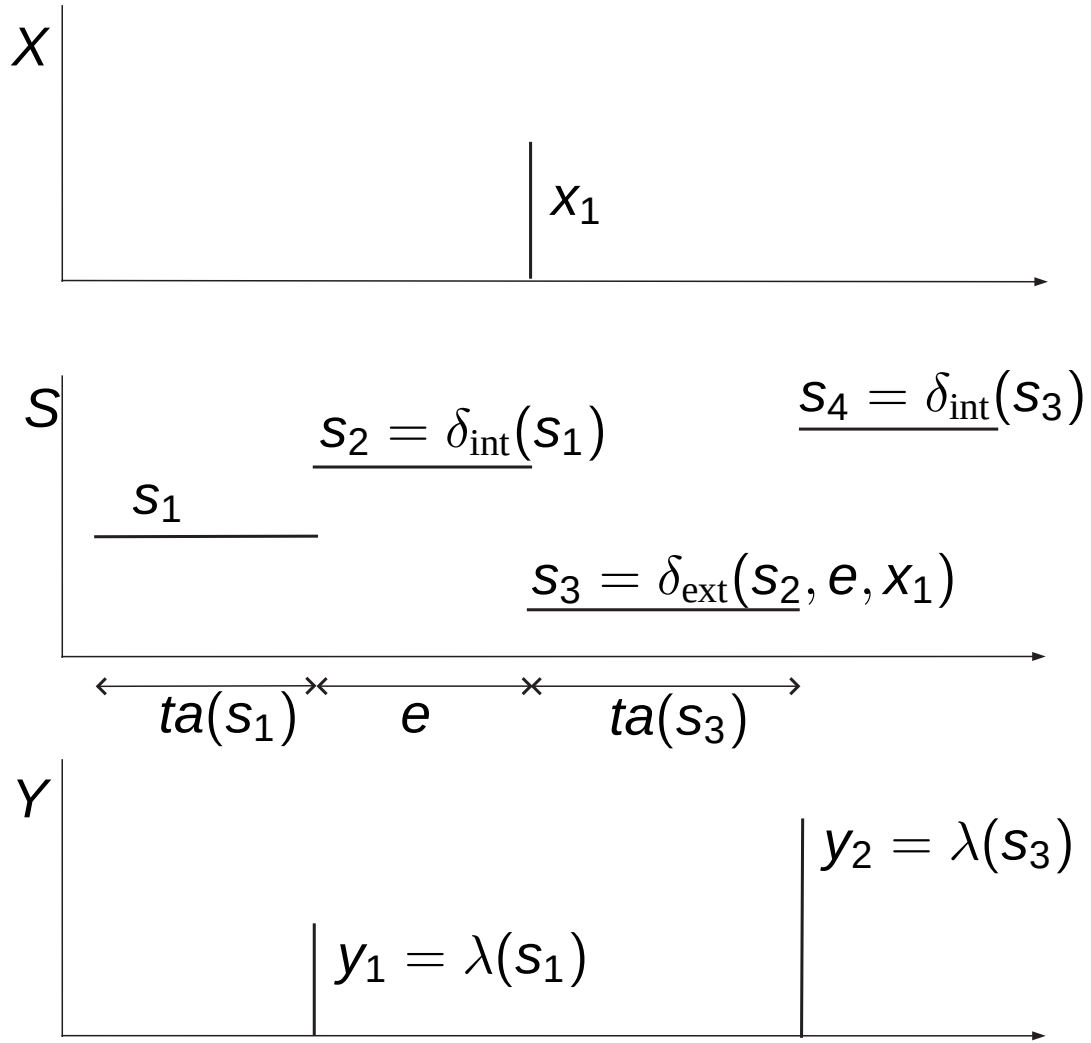
Un modelo atómico DEVS se define por la estructura:

$$M = (X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta)$$

donde:

- X es el conjunto de valores de entrada.
- Y es el conjunto de valores de salida.
- S es el conjunto de valores de estado.
- δ_{int}, δ_{ext}, λ y ta son las funciones que definen la dinámica del sistema.

DEVS – Trayectorias



DEVS – Ejemplo 1

Un sistema (que llamaremos generador) produce eventos que representan trabajos a realizar. El sistema produce eventos según la siguiente secuencia: $t = 0, y = 1$; $t = 1, y = 2$; $t = 3, y = 1$; $t = 4, y = 2$; etc.

$$G_1 = < X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta >$$
$$X = Y = S = \mathbb{R}^+$$
$$\delta_{\text{int}}(s) = 3 - s$$
$$\lambda(s) = 3 - s$$
$$ta(s) = s$$

DEVS – Ejemplo 2

Un procesador recibe trabajos identificados por un número real positivo que indica cuanto tiempo demora en procesarse dicho trabajo. Transcurrido el tiempo de procesamiento, el procesador emite un evento con valor 0.

$$P_1 =< X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta >$$
$$X = Y = S = \mathbb{R}^+$$
$$\delta_{\text{ext}}(s, e, x) = x$$
$$\delta_{\text{int}}(s) = \infty$$
$$\lambda(s) = 0$$
$$ta(s) = s$$

Este modelo se olvida del trabajo que estaba procesando al recibir uno nuevo

DEVS – Ejemplo 2

Para ignorar los eventos de entrada mientras se está procesando un trabajo, podemos modificar el modelo anterior como sigue:

$$P_2 =< X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \lambda, ta >$$

$$X = Y = \mathbb{R}^+$$

$$S = \mathbb{R}^+ \times \{true, false\}$$

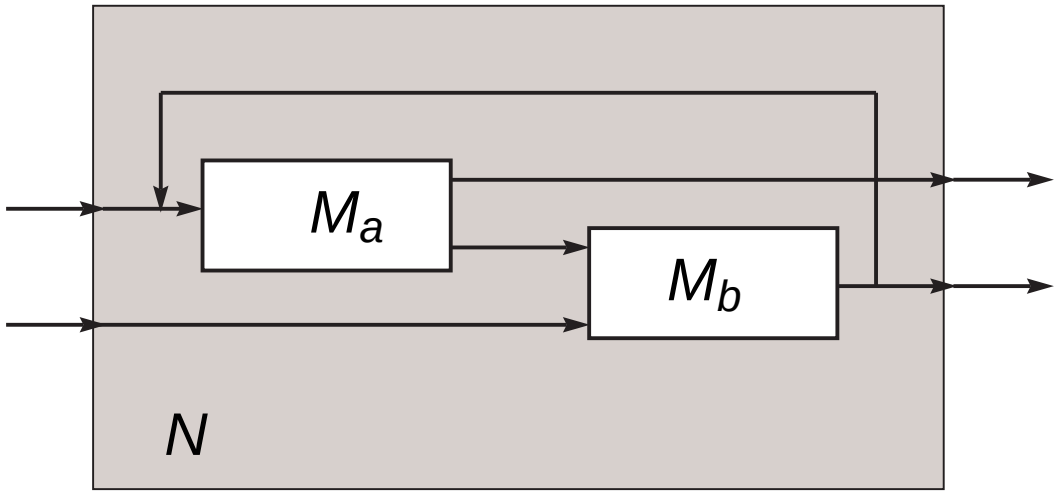
$$\delta_{\text{ext}}(s, e, x) = \delta_{\text{ext}}((\sigma, busy), e, x) = \begin{cases} (\sigma - e, true) & \text{si } busy = true \\ (x, true) & \text{en otro caso} \end{cases}$$

$$\delta_{\text{int}}(s) = \infty$$

$$\lambda(s) = 0$$

$$ta((\sigma, busy)) = \sigma$$

DEVS – Modelos Acoplados



En el acoplamiento modular los eventos de salida de un modelo DEVS se convierten en eventos de entrada de otro.

Clausura bajo acoplamiento

El acoplamiento de modelos atómicos DEVS define un modelo atómico equivalente.

Organización de la Presentación

Simulación de Modelos DEVS

Un simulador genérico de modelos DEVS funciona como sigue:

1. Para cada atómico $d \in N$, calculamos $tn_d = ta_d(s_d) - e_d$ (tiempo del próximo evento interno en el atómico d).
2. Buscamos el modelo atómico d^* que tiene el mínimo tn_d (d^* es el próximo atómico en realizar una transición interna).
3. Avanzamos el tiempo de la simulación t haciendo $t = tn_{d^*}$.
4. Calculamos el evento de salida $\lambda_{d^*}(s_{d^*})$.
5. Para cada modelo atómico d conectado a la salida del modelo d^*, ejecutamos la función de transición externa y recalculamos tn_d.
6. Ejecutamos la transición interna de d^* y recalculamos tn_{d^*}.
7. Volvemos al punto 2.

Software de Simulación de Modelos DEVS

Actualmente, existen varias herramientas de software basadas en DEVS, desarrollada por los distintos grupos de investigación que trabajan en el tema:

- ADEVS, DEVS/C++ y DEVSJAVA (University of Arizona).
- DEVSSim++ (KAIST, Corea).
- CD++ (Carleton University y UBA).
- LSIS-DME (LSIS, Marsella, Francia)
- VLE (Université du Littoral Côte d'Opale, Francia).
- SmallDEVS (Brno University of Technology, República Checa).
- JDEVS (Université de Corse).
- PowerDEVS (Universidad Nacional de Rosario).

Bibliografía


Christos Cassandras.
Discrete Event Systems: Modeling and Performance Analysis.
Irwin and Aksen, 1993.


B. Zeigler.
Theory of Modeling and Simulation.
John Wiley & Sons, New York, 1976.


B. Zeigler, T.G. Kim, and H. Praehofer.
Theory of Modeling and Simulation. Second edition.
Academic Press, New York, 2000.