Implementación de un procesador de lenguajes para Autómatas de Moore

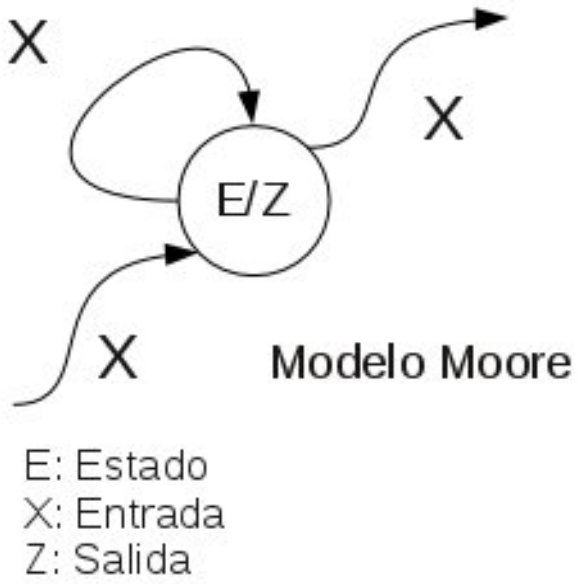
Julián García Sánchez Iván Illán Barraya
Alejandro Medina Jiménez Javier Monescillo Buitrón

26 de diciembre de 2018

Índice

1. ¿Qué es un autómata de Moore?

En Teoría de la computación una **Máquina de Moore** es un autómata de estados finitos para el cual la salida en un momento dado sólo depende de su estado en ese momento, mientras la transición al siguiente estado depende del estado en que se encuentre y de la entrada introducida. El diagrama de estados para una máquina de Moore incluirá una señal de salida para cada estado. [1]



Ejemplo de Máquina de Moore simple

El nombre de **Máquina de Moore** viene de su promotor Edward F. Moore, un pionero de las máquinas de estados, quien escribió *Gedanken-experiments on Sequential Machines*, pp 129-153, Estudios de Autómatas, Anuales de los Estudios Matemáticos, no. 34, Princenton Universityy Press, Princenton, N.J., 1956. [1]

1.1. Definición formal

Una máquina de Moore se define como una 6-tupla:

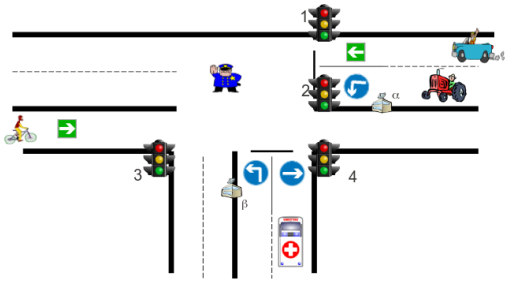
$$Mmor = (S, S_0, \Sigma, \Lambda, T, G)$$

donde definimos los siguientes elementos:

- S: es un conjunto finito de estados.
- S_0: es el estado inicial, y además es un elemento de S.
- Σ: un conjunto finito llamado alfabeto de entrada.
- Λ: un conjunto finito llamado el alfabeto de salida.
- T : una función de transición $T : S \times \Sigma \rightarrow S$ mapeando un estado y una entrada al siguiente estado.
- G función de salida $G : S \rightarrow \Lambda$ mapeando cada estado al alfabeto de salida.

1.2. Ejemplo propuesto

Definimos una Máquina de Moore para regular el tráfico en el siguiente cruce con los cuatro semáforos:



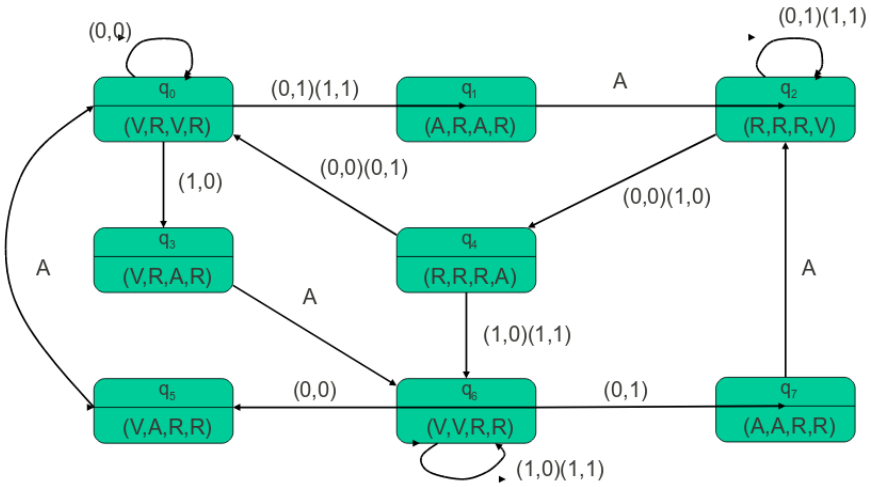
El alfabeto de entrada Σ: (0,0),(0,1),(1,0),(1,1)

- (a,b), donde a es el estado para el sensor α y b para el sensor β
- 0 indica que no hay coches en la cola
- 1 indica que hay coches en la cola

El alfabeto de salida Λ: (a_1, a_2, a_3, a_4 donde $a_i \epsilon$ (A,V,R))

- a_1 es el estado del semáforo 1
- a_2 es el estado del semáforo 2
- a_3 es el estado del semáforo 2
- a_4 es el estado del semáforo 2

El diagrama de estados sería el siguiente



2. Presentación del problema

La práctica consiste en el diseño de un procesador de lenguajes cuya entrada estará formada por una o varias máquinas de Moore y su salida será código en un lenguaje de alto nivel que lo represente, en nuestro caso Java.

Los conocimientos en la materia serán útiles en las distintas etapas del proceso, en primer lugar, se necesitará definir el léxico del lenguaje de entrada, para los cuál emplearemos los diagramas T que hemos aprendido durante los primeros temas.

También utilizaremos los conocimientos de la asignatura en la elaboración de la tabla de tokens, lexemas y patrones del lenguaje, y el EBNF de este.

2.1. Lenguaje definido

```
c1      # System.out.println("Activado sensor 1"); #

c2      # System.out.println("Activado sensor 2"); #

c3      # System.out.println("Activado sensor 3"); #

/*
Comentario

*/

moore Automata_de_Ejemplo{

        estados q0,q1,q2;
        estado_in q0 ;
        alf_in e1,e2,e3;
        alf_out c1,c2,c3;

        transicion {
                (q0,e1,q1),
                (q1,e2,q1),
                (q1,e3,q2);
        }

        comportamiento {
                (q0,c1),
                (q1,c2),
                (q2,c3);

        }
}
```

3. EBNF

PROGRAMA	::=	DEC_COMP AUTOMATA {AUTOMATA}
DEC_COMP	::=	CMP CODIGO { CMP CODIGO }
CODIGO	::=	′#′ ASCII ′#′
AUTOMATA	::=	**moore** ID CUERPO_AUTOMATA
CUERPO_AUTOMATA	::=	′{′ ESTADOS ESTADO_INI ALF_IN ALF_OUT TRANSICION COMPORTAMIENTOS ′}′
ESTADOS	::=	**estados** { ID ′,′ } ID ′;′
ESTADO_INI	::=	**estado_in** ID ′;′
ALF_IN	::=	**alf_in** { EVENTOS ′,′ } EVENTOS ′;′
ALF_OUT	::=	**alf_out** { CMP ′,′ } CMP ′;′
TRANSICION	::=	**transicion** ′{′ TRANSICION_DEF { ′,′ TRANSICION_DEF } ′;′ ′}′
TRANSICION_DEF	::=	′(′ ID ′,′ ID ′,′ ID ′)′
COMPORTAMIENTOS	::=	**comportamientos** ′{′ COMP_DEF { ′,′ COMP_DEF } ′;′ ′}′
COMP_DEF	::=	′(′ ID ′,′ ID ′,′ ID ′)′
CMP	::=	′c′NUMEROS
NUMEROS	::=	0 | 1 | .. | 9
COMENTARIOS	::=	′/∗′ ASCII ′ ∗ /′

4. Tablas de Tokens

Token	Lexema	Patrón
moore	moore	m·o·o·r·e
estados	estados	e·s·t·a·d·o·s
estado_in	estado_in	e·s·t·a·d·o·_·i·n
alf_in	alf_in	a·l·f·_i·n
alf_out	alf_out	a·l·f·_o·u·t
transicion	transicion	t·r·a·n·s·i·c·i·o·n
comportamiento	comportamiento	c·o·m·p·o·r·t·a·m·i·e·n·t·o
Paréntesis abierto	(	(
Paréntesis cerrado	)	)
Llave abierta	{	{
Llave cerrada	}	}
Punto y coma	;	;
Coma	,	,
Asterisco barra	*/	*·/
Barra asterisco	/*	/·*
ID	hola	[A-Za-z][A-Zaz0-9_]*
CMP	c1	c[1-9][0-9]*
CODIGO	#codigo aquí #	#· codigo aquí ·#

5. ¿Cómo se va a construir el procesador del lenguaje diseñado?

En el siguiente diagrama tipo T se explica el funcionamiento del procesador que se pretende implementar, tomamos **Mor** como lenguaje fuente, y Java como lenguaje objeto, y además el lenguaje que implementa es Java, es decir, nuestro compilador compila a Java, y esta escrito a Java.

Para utilizar ese compilador, utilizamos un compilador auxiliar, que está escrito en código máquina para dar lugar a bytecode, el típico archivo **.class** que generamos al compilar un **.java**.

Por último tenemos que compilar el bytecode, con un compilador que acepta bytecode escrito en código máquina.

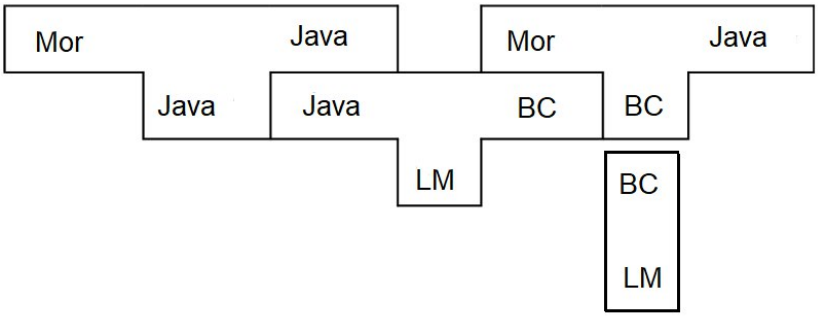


Diagrama tipo T asociado

Referencias

[1] Autómatas de Moore https://es.wikipedia.org/wiki/M%C3%A1quina_de_Moore