

DISEÑO LÓGICO



Objetivo:

Tener una idea general del impacto que los circuitos digitales tienen en los sistemas informáticos en software y/o hardware en la actualidad.

| FICHA TÉCNICA | |
|----------------------|--|
| ESCUELA: | Universidad del Sureste |
| CARRERA: | Ingeniería en Sistemas Computacionales |
| ALUMNO(A): | Jirem Madali Jiménez Trejo |
| DOCENTE: | Emmanuel Fabio Santiago Aguilar |
| CUATRIMESTRE: | 6to |
| MATERIA: | Diseño Lógico |
| PARCIAL: | 1ero |
| TEMA: | Autómata Finito |



AUTÓMATAS DE ESTADOS FINITOS: MELAY VS. MOORE

En la teoría de la computación, una **Máquina de Mealy** es un tipo de máquina de estados finitos que genera una salida basándose en su estado actual y una entrada. Esto significa que el Diagrama de estados incluirá ambas señales de entrada y salida para cada línea de transición. En contraste, la salida de una máquina de Moore de estados finitos (el otro tipo) depende solo del estado actual de la máquina, dado que las transiciones no tienen entrada asociada. Sin embargo, para cada Máquina de Mealy hay una máquina de Moore equivalente cuyos estados son la unión de los estados de la máquina de Mealy y el Producto cartesiano de los estados de la máquina de Mealy y el alfabeto de entrada.

El nombre "Máquina de Mealy" viene del promotor del concepto: George H. Mealy, un pionero de las máquinas de estados, quien escribió *Un Método para sintetizar Circuitos Secuenciales*, Bell System Tech. J. vol 34, pp. 1045–1079, September 1955.

Las máquinas de Mealy suministran un modelo matemático rudimentario y eficiente para las máquinas de cifrado. Considerando el alfabeto de entrada y salida del alfabeto Latino, por ejemplo, entonces una máquina de Mealy puede ser diseñada para darle una cadena de letras (una secuencia de entradas), esto puede procesarlo en un string cifrado (una secuencia de salidas). Sin embargo, aunque se podría probablemente usar un modelo de Mealy para describir una Máquina Enigma, el diagrama de estados sería demasiado complejo para suministrar medios factibles de diseñar máquinas de cifrado complejas.

Una máquina de Mealy es una 6-tupla, $M=(S, S_0, \Sigma, \Lambda, T, G)$:

S es un conjunto finito de estados.

S_0 es un estado inicial, el cual es un elemento de S . $S_0 \in S$ Σ es

un conjunto finito, llamado alfabeto de entrada.

Λ es un conjunto finito, llamado alfabeto de salida.

T es una función de transiciones ($T : S \times \Sigma \rightarrow S$)

G es una función de salida ($G : S \times \Sigma \rightarrow \Lambda$)

En la 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 Moore incluirá una señal de salida para cada estado. Comparada con la Máquina de Mealy, la cual mapea *transiciones* en la máquina a salidas.

El nombre **Moore machine** 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, Princeton University Press, Princeton, N. J., 1956.

La mayoría de las máquinas electrónicas están diseñadas como sistemas secuenciales síncronos. Los sistemas secuenciales síncronos son una forma restringida de máquinas de Moore donde el estado cambia solo cuando la señal de reloj global cambia. Normalmente el estado actual se almacena en Flip-flops, y la señal de reloj global está conectada a la entrada "clock" de los flip-flops. Los sistemas secuenciales síncronos son una manera de resolver problemas de meta estabilidad.

Una máquina electrónica de Moore típica incluye una cadena de Lógica combinacional para decodificar el estado actual en salidas (Λ). El instante en el cual el estado actual cambia, aquellos cambios se propagan a través de la cadena. y casi instantáneamente las salidas cambian (o no cambian). Hay técnicas de diseño para asegurar que no ocurran errores de corta duración en las salidas durante el breve periodo mientras esos cambios se están propagando a través de la cadena, pero la mayoría de los sistemas están diseñados para que los glitches durante el breve tiempo de transición sean ignorados. Las salidas entonces permanecen igual indefinidamente (por ejemplo, los LEDs permanecen brillantes, la batería permanece conectada a los motores, etc.), hasta que la máquina de Moore cambia de estado otra vez.

Una máquina de Moore puede ser definida como una 6-tupla $\{ S, S_0, \Sigma, \Lambda, T, G \}$ consistente de un conjunto finito de estados (S) un estado inicio (también llamado estado inicial) S_0 el cual es un elemento de (S) un conjunto finito llamado alfabeto entrada (Σ) un conjunto finito llamado el alfabeto salida (Λ) una función de transición ($T : S \times \Sigma \rightarrow S$) mapeando un estado y una entrada al siguiente estado una función salida ($G : S \rightarrow \Lambda$) mapeando cada estado al alfabeto salida.

El número de estados en una máquina de Moore será mayor o igual al número de estados en la Máquina de Mealy correspondiente.

AUTOMATAS FINITOS

Un autómata finito es un modelo matemático de una máquina que acepta cadenas de un lenguaje definido sobre un alfabeto A . Consiste en un conjunto finito de estados y un conjunto de

transiciones entre esos estados, que dependen de los símbolos de la cadena de entrada. El autómata finito acepta una cadena x si la secuencia de transiciones correspondientes a los símbolos de x conduce desde el estado inicial a un estado final.

Si para todo estado del autómata existe como máximo una transición definida para cada símbolo del alfabeto, se dice que el autómata es determinístico (AFD). Si a partir de algún estado y para el mismo símbolo de entrada, se definen dos o más transiciones se dice que el autómata es no determinístico (AFND).

Formalmente un autómata finito se define como una 5-upla

$$M = \langle E, A, \delta, e_0, F \rangle \quad \text{donde}$$

E : conjunto finito de estados

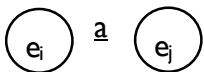
A : alfabeto o conjunto finito de símbolos de entrada δ : función de transición de estados, que se define como

- $\delta: E \times A \rightarrow E$ si el autómata es determinístico

- $\delta: E \times A \rightarrow P(E)$ si el autómata es no determinístico ($P(E)$ es el conjunto potencia de E , es decir el conjunto de todos los subconjuntos de E) e_0 : estado inicial; $e_0 \in E$

F : conjunto de estados finales o estados de aceptación; $F \subseteq E$

Generalmente se asocia con cada autómata un grafo dirigido, llamado diagrama de transición de estados. Cada nodo del grafo corresponde a un estado. El estado inicial se indica mediante una flecha que no tiene nodo origen. Los estados finales se representan con un círculo doble. Si existe una transición del estado e_i al estado e_j para un símbolo de entrada a , existe entonces un arco rotulado a desde el nodo e_i al nodo e_j ; es decir que $\delta(e_i, a) = e_j$, se representa en el diagrama




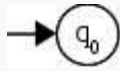


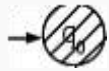
ESPECIFICACIÓN DEL SISTEMA MEDIANTE DIAGRAMAS Y TABLAS DE ESTADOS

Diagrama de Estado: Esta muestra la secuencia de estados por los que pasa bien un caso de uso, un objeto a lo largo de su vida, o bien todo el sistema. Es una forma de representación gráfica más intuitiva de los autómatas finitos basadas en dígrafos con arcos acotados llamados transiciones en los cuales se ponen los símbolos de tránsito entre un vértice (estado) y otro y se identifican los estados de partida y los de aceptación del resto. Los diagramas de estados finitos son también representaciones más cómodas para su elaboración, legibilidad y comprensión de distintos tipos de abstracciones computacionales de reconocimiento como los autómatas de pila y las máquinas de Turing.

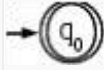
Sea un autómata finito definido por la 5-tupla $A = \langle Q, T, g, F, q_0 \rangle$, donde Q es el conjunto de estados, T el alfabeto de símbolos terminales, la relación de transiciones $g \subseteq Q \times (T \cup \{\epsilon\}) \times Q$, F son los estados finales o de llegada dentro de Q , q_0 es el estado inicial o de partida; se le llama **diagrama de estados de A** al grafo orientado con aristas y vértices acotados de la forma siguiente:

- Todos los estados de Q se representan por círculos en cuyo interior se designa el nombre del estado que representa.
- El estado inicial q_0 se indica agregando una saeta corta a su izquierda que no tiene origen alguno y concluye en el borde de la circunferencia, normalmente en los 180 grados del mismo.
- Los estados finales se indican o bien sombreando el círculo o poniendo un borde doble al estado.
- Pueden existir estados iniciales finales, simplemente se aplican los dos casos anteriores.
- A cada transición entre los estados q_i y q_j con el símbolo terminal x o la cadena vacía ϵ se representa como un arco etiquetado con x ó ϵ según corresponda.

- Si entre los estados y en el mismo sentido hay varias transiciones con los terminales x_1, x_2, \dots, x_n entonces se indican en el mismo arco pero separados por espacio o comas según convenga.

| Tipo | Diagrama de estado | Descripción |
|----------------|---|--|
| Estado |  | Círculo con el nombre del estado etiquetado dentro |
| Estado inicial |  | q_0 : Flecha corta sin origen que apunta al estado |
| Estado final |  | $q \in F$: Estado sombreado. |
| Estado final |  | $q \in F$: Estado con doble circunferencia. |
| Estado inicial |  | $q_0 \in F$: Flecha sin origen que apunta al estado |
| y final | | sombreado. |

Estado inicial y final



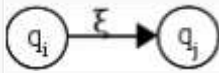
$q_0 \in F$: Flecha sin origen que apunta al estado doblemente circulado.

Transición



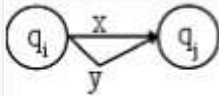
$\langle q_i, x, q_j \rangle$ ó $g(q_i, x) = q_j$: Arco con origen en q_i y destino en q_j y acotado con el terminal x .

Transición vacía



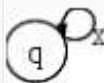
$\langle q_i, \epsilon, q_j \rangle$ ó $g(q_i, \epsilon) = q_j$: Arco con origen en q_i y destino en q_j y acotado con la cadena vacía ϵ .

Transición múltiple



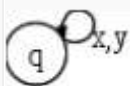
$g(q_i, x) = q_j, g(q_i, y) = q_j$: Arco con origen en q_i y destino en q_j y acotado con los terminales x, y , separados por coma o espacio en blanco.

Lazo



$\langle q, x, q \rangle$ ó $g(q, x) = q$: Arco circular con origen y destino en q , acotado con el terminal x .

Lazo múltiple



$g(q, x) = q, g(q, y) = q$: Arco circular con origen y destino en q , acotado con los terminales x, y .

Función

En el diagrama de estados se indica qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera. También ilustra qué eventos pueden cambiar el estado de los objetos de la clase. En cuanto a la representación, un diagrama de estados es un grafo cuyos nodos son estados y cuyos arcos dirigidos son transiciones etiquetadas con los nombres de los eventos. Normalmente contienen: estados y transiciones. Como los estados y las transiciones incluyen, a su vez, eventos, acciones y actividades. Al igual que otros diagramas, en los diagramas de estado pueden aparecer notas explicativas y restricciones.

Definición de Estado

Identifica un periodo de tiempo del objeto (no instantáneo) en el cual el objeto está esperando alguna operación, tiene cierto estado característico o puede recibir cierto tipo de estímulos. Se representa mediante un rectángulo con los bordes redondeados, que puede tener tres compartimientos: uno para el nombre, otro para el valor característico de los atributos del objeto en ese estado y otro para las acciones que se realizan al entrar, salir o estar en un estado. También en casos más simples se usan círculos con textos dentro para la representación de los estados, como para los autómatas finitos.

Partes que conforman el Diagrama de Estados

Estado

Un estado se representa como una caja redondeada con el nombre del estado en su interior. Una transición se representa como una flecha desde el estado origen al estado destino. La caja de un estado puede tener 1 o 2 compartimientos. En el primer compartimiento aparece el nombre del estado. El segundo compartimiento es opcional, y en él pueden aparecer acciones de entrada, de salida y acciones internas.

Eventos

- Es una ocurrencia que puede causar la transición de un estado a otro de un objeto.
Esta ocurrencia puede ser una de varias cosas:

- Condición que toma el valor de verdadero o falso
- Recepción de una señal de otro objeto en el modelo

Recepción de un mensaje

Paso de cierto período de tiempo, después de entrar al estado o de cierta hora y fecha particular. El nombre de un evento tiene alcance dentro del paquete en el cual está definido, no es local a la clase que lo nombre.

Envío de mensajes

Además de mostrar y transición de estados por medio de eventos, puede representarse el momento en el cual se envían mensajes a otros objetos. Esto se realiza mediante una línea punteada dirigida al diagrama de estados del objeto receptor del mensaje.

Transición simple

Una transición simple es una relación entre dos estados que indica que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones, cuando un evento ocurre y si ciertas condiciones son satisfechas. Se representa como una línea sólida entre dos estados, que puede venir acompañada de un texto con el siguiente formato:

Transición interna

Es una transición que permanece en el mismo estado, en vez de involucrar dos estados distintos. Representa un evento que no causa cambio de estado. Se denota como una cadena adicional en el compartimiento de acciones del estado.

Acciones

Se puede especificar la solicitud de un servicio a otro objeto como consecuencia de la transición. Se puede especificar el ejecutar una acción como consecuencia de entrar, salir, estar en un estado, o por la ocurrencia de un evento.

Generalización de Estados

Se puede reducir la complejidad de estos diagramas usando la generalización de estados. Se distingue así entre superestado y subestados. Un estado puede contener varios subestados disjuntos. Los subestados heredan las variables de estado y las transiciones externas. La agregación de estados es la composición de un estado a partir de varios estados independientes. La composición es concurrente por lo que el objeto estará en alguno de los estados de cada uno de los subestados concurrentes. La destrucción de un objeto es efectiva cuando el flujo de control del autómata alcanza un estado final no anidado. La llegada a un estado final anidado implica la subida al superestado asociado, no el fin del objeto.

Subestados

Un estado puede descomponerse en subestados, con transiciones entre ellos y conexiones al nivel superior. Las conexiones se ven al nivel inferior como estados de inicio o fin, los cuales se suponen conectados a las entradas y salidas del nivel inmediatamente superior.

Transacción Compleja

Una transición compleja relaciona tres o más estados en una transición de múltiples fuentes y/o múltiples destinos. Representa la subdivisión en threads del control del objeto o una sincronización. Se representa como una línea vertical de la cual salen o entran varias líneas de transición de estado.

Transición a estados anidados

Una transición de hacia un estado complejo (descrito mediante estados anidados) significa la entrada al estado inicial del subdiagrama. Las transiciones que salen del estado complejo se entienden como transiciones desde cada uno de los subestados hacia afuera (a cualquier nivel de profundidad).

Transiciones temporizadas

Las esperas son actividades que tienen asociada cierta duración. La actividad de espera se interrumpe cuando el evento esperado tiene lugar. Este evento desencadena una transición que

permite salir del estado que alberga la actividad de espera. El flujo de control se transmite entonces a otro estado.

Ventajas y Desventajas

Ventajas

El Diagrama de Estados tiene éxito en sistemas interactivos, ya que expresa la intención que tiene el actor (su usuario) al hacer uso del sistema.

Como técnica de extracción de requerimiento permite que el analista se centre en las necesidades del usuario, qué espera éste lograr al utilizar el sistema, evitando que la gente especializada en informática dirija la funcionalidad del nuevo sistema basándose solamente en criterios tecnológicos. A su vez, durante la extracción (elicitation en inglés), el analista se concentra en las tareas centrales del usuario describiendo por lo tanto los casos de uso que mayor valor aportan al negocio. Esto facilita luego la priorización del requerimiento.

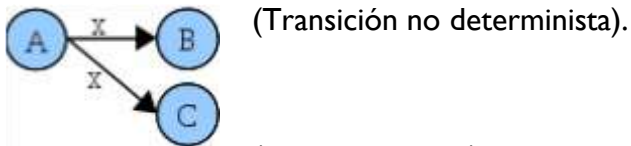
Desventajas

La inclusión de estas relaciones hace que los diagramas sean más difíciles de leer, sobre todo para los clientes.

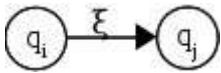
Importancia

Los diagramas de estado en el caso de los automatas finitos, además de mejorar su legibilidad, comprensión, e incluso visualizar una especie de primera aproximación material a su implementación física o computacional; también ayudan a visibilizar las propiedades del AF más intuitivamente que en la notación de la 5-tupla o la de la tabla de transiciones.

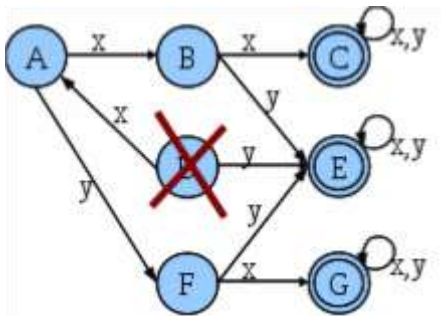
Por ejemplo, un autómata finito no determinista se reconoce con más prontitud viendo si en su esquema aparecen las transiciones de la forma:



(Transición vacía).



○ si existe un estado inaccesible desde el inicio como en el caso:



En teoría de autómatas y lógica secuencial, una **tabla de transición de estados** es una tabla que muestra qué estado se moverá un autómata finito dado, basándose en el estado actual y otras entradas. Una *tabla de estados* es esencialmente una tabla de verdad en la cual algunas de las entradas son el estado actual, y las salidas incluyen el siguiente estado, junto con otras salidas.

Una tabla de estados es una de las muchas maneras de especificar una *máquina de estados*, otras formas son un diagrama de estados, y una *ecuación característica*.

Cuando se trata de un autómata finito no determinista, entonces la tabla de transición muestra todos los estados que se moverá el autómata.

Tablas de estados de una dimensión

También llamadas **tablas características**, las tablas de estados de una dimensión son más como tablas de verdad que como las versiones de dos dimensiones. Las entradas son normalmente colocadas a la izquierda, y separadas de las salidas, las cuales están a la derecha. Las salidas representarán el siguiente estado de la máquina. Aquí hay un ejemplo sencillo de una máquina de estados con dos estados, y dos entradas combinacionales:

| A | B | Estado Actual | Siguiente Estado | Salida |
|----------|----------|----------------------|-------------------------|---------------|
| 0 | 0 | S ₁ | S ₂ | 1 |
| 0 | 0 | S ₂ | S ₁ | 0 |
| 0 | 1 | S ₁ | S ₂ | 0 |
| 0 | 1 | S ₂ | S ₂ | 1 |
| 1 | 0 | S ₁ | S ₁ | 1 |
| 1 | 0 | S ₂ | S ₁ | 1 |
| 1 | 1 | S ₁ | S ₁ | 1 |
| 1 | 1 | S ₂ | S ₂ | 0 |

S₁ y S₂ representarían probablemente los bits individuales 0 y 1, dado que un simple bit solo tiene dos estados.

- Tablas de Estados de dos dimensiones
- Las tablas de transición de estados son normalmente tablas de dos dimensiones. Hay dos formas comunes para construirlas.

La dimensión vertical indica los Estados Actuales, la dimensión horizontal indica eventos, y las celdas (intersecciones fila/columna) de la tabla contienen el siguiente estado si ocurre un evento (y posiblemente la acción enlazada a esta transición de estados).

Tabla de Transición de Estados

| Events | E ₁ | E ₂ | ... | E _n |
|----------------|--------------------------------|--------------------------------|-----|--------------------------------|
| State | | | | |
| S ₁ | - | A _y /S _j | ... | - |
| S ₂ | - | - | ... | A _x /S _i |
| ... | ... | ... | ... | ... |
| S _m | A _z /S _k | - | ... | - |

(S: estado, E: evento, A: acción, -: transición ilegal)

La dimensión vertical indica los Estados Actuales, la dimensión horizontal indica los siguientes estados, y las intersecciones fila/columna contienen el evento el cual dirigirá al siguiente estado particular.

Tabla de Transición de Estados

| next current | S ₁ | S ₂ | ... | S _m |
|-----------------|--------------------------------|--------------------------------|-----|--------------------------------|
| S ₁ | A _y /E _j | - | ... | - |
| S ₂ | - | - | ... | A _x /E _i |
| ... | ... | ... | ... | ... |
| S _m | - | A _z /E _k | ... | - |

(S: estado, E: evento, A: acción, -: transición imposible)

BIBLIOGRAFÍA BÁSICA Y COMPLEMENTARIA

- <http://www.uco.es/~el1momoc/BibEIDig.html>
- <https://ocw.ehu.eus/mod/page/view.php?id=1793>
- <http://www3.uji.es/~mmarques/f47/teoria/tema7.pdf>
- <https://web-argitalpena.adm.ehu.es/pdf/UCWEB142021.pdf>
- <http://jagarza.fime.uanl.mx/general/notas/FDDSC.pdf>
- https://ocw.unican.es/pluginfile.php/313/course/section/261/tema_03.pdf
- Diseño Digital /por M. Morris Mano y traducción de Julio Fournier González., Mano, M. Morris., DISEÑO LOGICO ; CIRCUITOS INTEGRADOS DIGITALES ; CIRCUITOS LOGICOS ; COMPUTADORES ELECTRONICOS DIGITALES - CIRCUITOS.
- Principios de Diseño Digital /por Daniel D. Gajski, traducción de Carlos Garcia Puntonet y otros., Gajski, Daniel D.