Análisis y Complejidad de Algoritmos

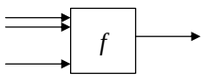
Modelos Computacionales

Arturo Díaz Pérez

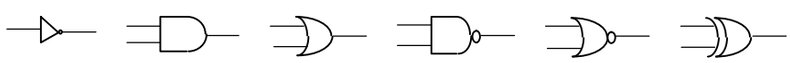
- El circuito lógico
- La máquina de estados finitos
- La máquina de acceso aleatorio
- La máquina de Turing

Compuertas Lógicas

- Compuerta Lógica
  - Un dispositivo físico que realiza una función booleana
    - f: **N** ® **{ 0, 1 }**
    - Frecuentemente n se codifica como una número en base 2
    - $n = x_{k-1}2^{k-1} + x_{k-2}2^{k-2} + ... + x_1 2^1 + x_0 2^0 +$, $x_i = 0, 1$



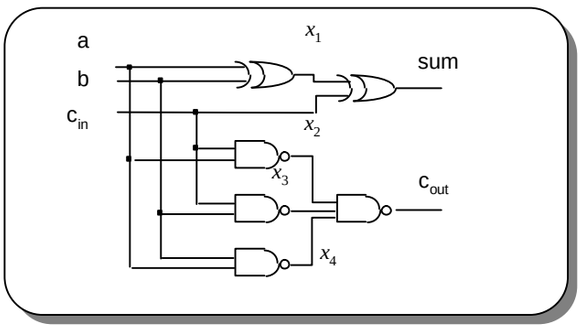
Las compuertas lógicas pueden ser construidas por tecnologías diferentes
Se acostumbra utilizar un símbolo lógico para cada compuerta
Los símbolos se utilizan para dibujar **circuitos lógicos**



Circuitos Lógicos

- Circuito Lógico
  - Una DAG en la cual todos sus vértices, excepto los vértices de entrada y salida, están etiquetados con compuertas lógicas
  - Los circuitos lógicos ejecutan programas estrictamente secuenciales, esto es, programas que consisten únicamente de asignmientos
  - No contienen ciclos ni saltos
  - Los circuitos lógicos constituyen los bloques básicos para la construcción de computadoras digitales.
  - Cuando se combinan con celdas de memoria binarias, se pueden construir máquinas con memoria (máquinas de estados finitos)

Circuitos Lógicos

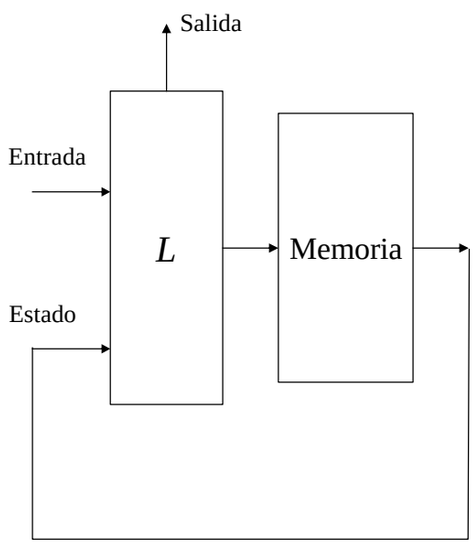


x_1 := a **xor** b
x_2 := a **nand** c_{in}
x_3 := b **nand** c_{in}
x_4 := a **nand** b
sum := x_1 **xor** c_{in}
c_{out} := (x_2 **nand** x_3) **nand** x_4

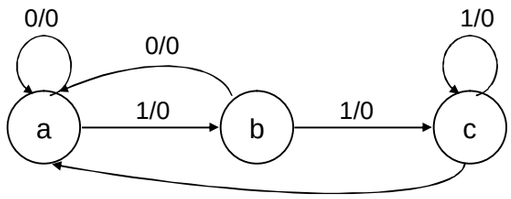
Máquinas de Estados Finitos

- Máquinas de estados finitos
  - Es una máquina con memoria
  - Ejecuta una serie de pasos en los cuales
    - toma su estado actual de un conjunto de estados *Q*
    - define su salida a partir de un conjunto de símbolos de entrada Σ
  - FSM = < Σ, Ψ, *Q*, δ, λ>
    - Σ, alfabeto de símbolos de entrada
    - Ψ, alfabeto de símbolos de salida
    - *Q*, conjunto finito de estados
    - δ, función de transición
      - $\delta : \Sigma \times Q \rightarrow Q$
    - λ, función de transición
      - $\lambda : \Sigma \times Q \rightarrow \Psi$

Máquinas de Estados Finitos



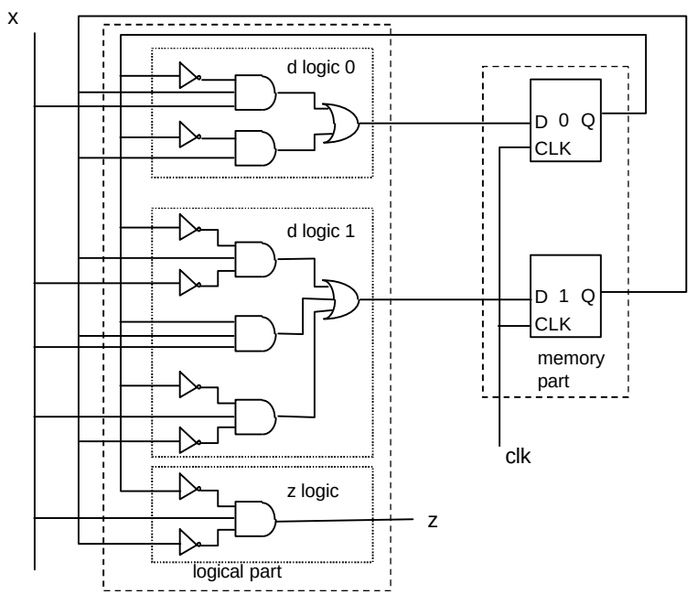
FSM: Ejemplo



ESTADO	V0	V1	0	x_{in} 1
a	0	0	00,0	01,0
b	0	1	00,0	10,0
c	1	0	00,$\neg y_{in}$	10, $\neg y_{in}$
-	1	1	--,-	--,-

Análisis y Diseño de Algoritmos *Models*

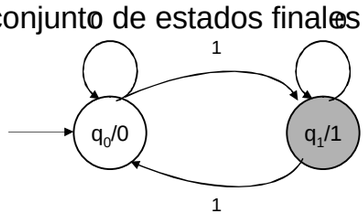
FSM: Estructura General



Análisis y Diseño de Algoritmos *Models*

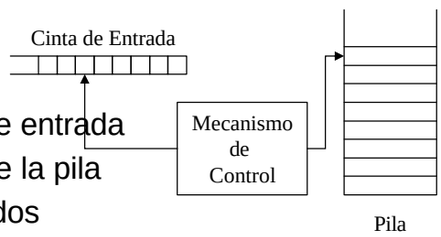
Autómata Finito

- Autómata Finito
  - AF = < Σ, *Q*, δ, q_0, *F*>
    - Σ, alfabeto de símbolos de entrada
    - *Q*, conjunto finito de estados
    - δ, función de transición
      - $\delta : \Sigma \times Q \rightarrow Q$
    - q_0, estado inicial
    - $F \subset Q$, conjunto de estados finales



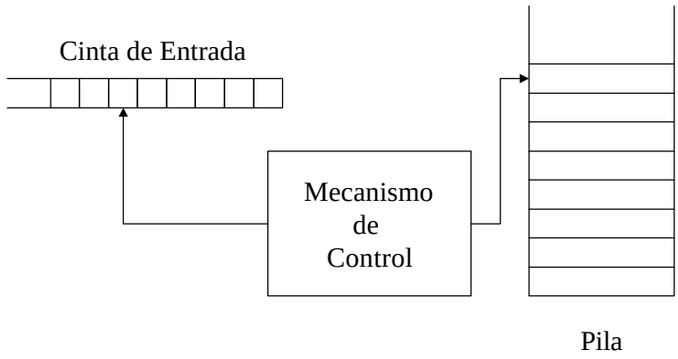
Autómata de Pila

- Autómata de Pila
  - Consiste de
    - una *cinta de entrada*,
    - un *control finito* y
    - una *pila*
  - AF = < Σ, Γ, *Q*, δ, q_0, Z_0, *F*>
    - Σ, alfabeto de símbolos de entrada
    - Γ, alfabeto de símbolos de la pila
    - *Q*, conjunto finito de estados
    - δ, función de transición
      - $\delta : Q \times \Sigma \times \Gamma \rightarrow Q \times \Gamma$
    - q_0, estado inicial
    - Z_0, símbolo inicial de la pila
    - $F \subset Q$, conjunto de estados finales



Autómata de Pila

- Autómata de Pila
  - Consiste de
    - una *cinta de entrada*,
    - un *control finito* y
    - una *pila*



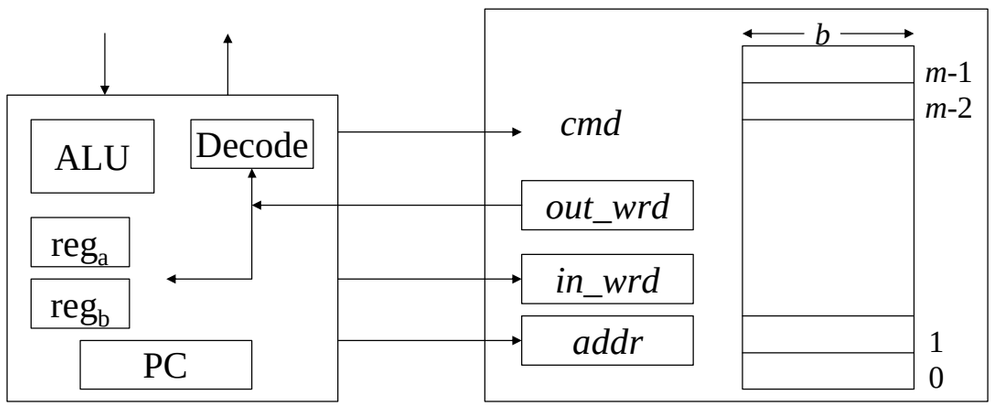
Análisis y Diseño de Algoritmos *Models*

La Máquina de Acceso Aleatorio: RAM

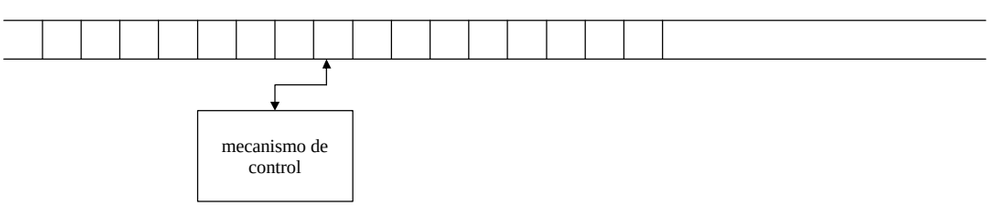
- RAM: Random Access Machine
  - Es modelada como dos máquinas de estados finitos:
    - **unidad central de procesamiento** (CPU)
    - **memoria de acceso aleatorio**
    - *cmd:* READ, WRITE, NO-OP
    - instrucciones:
      - LOAD, STORE, ADD, SUB, MULT, DIV,
      - READ, WRITE,
      - JUMP, JZ, JNZ, HALT

Análisis y Diseño de Algoritmos *Models*

La Máquina de Acceso Aleatorio: RAM



Máquina de Turing



Máquina de Turing

- Máquina de Turing
  - Cinta no acotada formada por *casillas*
  - *mecanismo de control*, i. e., autómata con diversos estados
  - *cabeza de lectura* con la que examina, en cada instante, una casilla en la cinta
  - *programa*, lista de quíntuples de la forma
    - *qs* $\rightarrow$ *ptm*
    - *q,p* $\in$ *Q* (estados), *s,t* $\in \Sigma$ (símbolos del alfabeto), *m* $\in$ { I, D, A} (movimiento)
    - "*Si en el estado **q** se lee en la cinta el símbolo **s**, entonces, cambia a **s** por **t**, luego pasa al estado **p** y pasa a examinar la casilla adyacenta en la dirección **m**"*

Máquina de Turing: Ejemplo

1, *x*	$\rightarrow$ 1, *x*, D
1,)	$\rightarrow$ 2, B, I
1, *b*	$\rightarrow$ 3, *b*, I
2, *y*	$\rightarrow$ 2, *y*, I
2, (	$\rightarrow$ 1, A, D
2, *b*	$\rightarrow$ 4, No, H
3, *x*	$\rightarrow$ 3, *x*, I
3, (	$\rightarrow$ 3, No, H
3, *b*	$\rightarrow$ 3, Si, H

**(**)(())	AB(**1**())	AB(**2**ABB	ABAAB**3**B	**3***b*ABAABB
1()(())	AB((**1**))	AB**2**(ABB	ABAA**3**BB	**[Si]**ABAABB
(1)(())	AB(**2**(B)	ABA**1**ABB	ABA**3**ABB	
(2B(())	AB(A**1**B)	ABAA**1**BB	AB**3**AABB	
A**1**B(())	AB(AB**1**)	ABAAB**1**B	A**3**BAABB	
AB**1**(())	AB(A**2**BB	ABAABB**1***b*	**3**ABAABB	

Máquina de Turing: Ejemplo

☞ Reconocimiento de cadenas equilibradas de paréntesis

1, *x* → 1, *x*, D	con cualquier cosa salvo “)”, avánce se a la derecha, *x* ∈ { (, A, B }
1,) → 2, B, I	con el primer “)”, márquese y retrocédase
1, *b* → 3, *b*, I	si la lista se acaba, revísese que todo haya quedado marcado
2, *y* → 2, *y*, I	con cualquier cosa salvo “(”, retrocédase a la izquierda, *y* ∈ {), A, B }
2, (→ 1, A, D	márquese el “(“ que empata y repítase el ciclo
2, *b* → 4, No, H	se termina la cadena y no existe el correspondiente “(”. No hay equilibrio.
3, *x* → 3, *x*, I	retrocédase ignorando las marcas, *x* ∈ { A, B }
3, (→ 3, No, H	si quedara un “(”, éste ya no podrá empatarse. No hay equilibrio.
3, *b* → 3, Si, H	acabó la cadena y todo quedó marcado. Si hay equilibrio.

Análisis y Diseño de Algoritmos *Models*