“TECNOLÓGICO NACIONAL DE MEXICO”

INSTITUTO TECNOLÓGICO DE IZTAPALAPA

INTEGRANTES:

ISC-6AM

LENGUAJES Y AUTOMATAS I

M.C. ABIEL TOMÁS PARRA HERNÁNDEZ

SEP 2020 / FEB 2021

ACTIVIDAD SEMANA 7

Cuananemi Cuanalo Mario Alberto

DFA • El lenguaje de un DFA es el conjunto de todas las cadenas que el DFA acepta • Dada una cadena (e.g., s1, s2, . . . , sn) el DFA empieza en su estado inicial (e.g., q0), consulta si existe una transicion de ´ q0 con el primer s´ımbolo (s1) a otro estado (e.g., q1) y si existe (i.e., δ(q0, s1) = q1) se mueve al estado descrito en la transicion. ´ • Procesa el siguiente s´ımbolo de la cadena (i.e., s2) y as´ı continua. • Si logra procesar toda la cadena y el estado al que llega es uno de los estados finales, entonces se dice que el automata acepta esa cadena.

Ejemplo Un Automata A que acepta ´ L = {x01y|x ∧ y ∈ {0, 1} ∗} • El DFA acepta cadenas que tienen 01 en alguna parte de la cadena • El lenguaje del DFA es el conjunto de cadenas que acepta {w|w tiene la forma “x01y” para algunas cadenas x y y que consisten solo de 0’s y 1’s ´ }

Diferencia entre DFA y NFA:

NO SEÑOR.	DFA	NFA
1	DFA son las siglas de Deterministic Finite Automata.	NFA son las siglas de Nondeterministic Finite Automata.
2	Para cada representación simbólica del alfabeto, solo hay una transición de estado en DFA.	No es necesario especificar cómo reacciona la NFA según algún símbolo.
3	DFA no puede utilizar la transición de cadena vacía.	NFA puede utilizar la transición de cadena vacía.
4	DFA puede entenderse como una sola máquina.	NFA puede entenderse como múltiples pequeñas máquinas que computan al mismo tiempo.
5	En DFA, el siguiente estado posible se establece claramente.	En NFA, cada par de estado y símbolo de entrada puede tener muchos estados siguientes posibles.
6	DFA es más difícil de construir.	NFA es más fácil de construir.
7	DFA rechaza la cadena en caso de que termine en un estado diferente del estado de aceptación.	NFA rechaza la cuerda en caso de que todas las ramas mueran o rechacen la cuerda.
8	El tiempo necesario para ejecutar una cadena de entrada es menor.	El tiempo necesario para ejecutar una cadena de entrada es mayor.
9	Todos los DFA son NFA.	No todos los NFA son DFA.
10	DFA requiere más espacio.	NFA requiere menos espacio que DFA.

Teoria de Myhill-Nerode

Sea Ent un alfabeto finito. Según vimos anteriormente la noción de *lenguaje regular* puede presentarse mediante el reconocimiento por autómatas finitos, sean deterministas o no, o mediante la representación por expresiones regulares. Otras presentaciones equivalentes de esta noción se dan en la siguiente:

Proposición 6.1 (Teorema de Myhill-Nerode) Sea $L \subset Ent^*$ un lenguaje arbitrario. Las siguientes aseveraciones son equivalentes a pares:

1.
L es regular.

2.
L es la unión de algunas clases de equivalencia de una relación de equivalencia de índice finito, congruente por la derecha con la concatenación de palabras.

3.
La relación $1. \Rightarrow 2.$ tal que

$$L = L(AF$$

es una relación de índice finito.

Demostración:

Supongamos *L* regular. Sea $AF = (Q, Ent, tran, q_0, F)$ un

autómata finito $\equiv_K$ tal que . Consideremos la función

y su kernel:

$$L = \bigcup [\sigma]_{\equiv_K}.$$

Hemos visto que ``$2. \Rightarrow$.'' es una relación de equivalencia de índice finito (de hecho este índice está acotado por la cardinalidad de *Q*), congruente por la derecha. Se tiene además

$$R \subset (Ent^*)^2$$

Así pues, se cumple 2.

$\mathcal{F} \subset Q/R$ Sea $L = \bigcup_{[\sigma] \in \mathcal{F}} [\sigma]$ una relación de equivalencia de índice finito, congruente por la derecha con la concatenación de palabras, tal que para un subconjunto

se tiene que $\sigma R \tau$. Afirmamos que

$$\forall v \in Ent^* \tag{47}$$

En efecto, si $\sigma v R \tau v$ entonces, $(\sigma \cdot v \in L \Leftrightarrow$, al ser *R* congruente por la derecha, $\sigma \equiv_L \tau$.

Como *L* es la unión de algunas clases de *R*, esto da que $\equiv_L$. Así pues, $3 \Rightarrow 1$. De la relación 4.3 6 vemos que toda clase de equivalencia respecto a `` $Q =$ '' es la unión de clases de equivalencia respecto a *R*. Por tanto el índice de `` $Q =$ '' no puede exceder el de

R. Como este último es finito el de `` $Q =$ '' es también finito.

$q_0 = [nil]$ `` $Q =$ '' es una relación de equivalencia de índice finito, congruente por la derecha. Definamos $AF = (Q, Ent, tran, q_0, F)$ $F = \{[\sigma] | \sigma \in L\}$

$tran : ([\sigma],$, $q_1, q_2 \in Q$ y haciendo ,

. Es evidente que una palabra es

reconocida por AF si y sólo si esa palabra está en *L* . Así pues, $L = L(AF)$

y,consecuentemente, es regular.

Para un autómata finito $AF = (Q, Ent, tran, q_0, F)$ diremos que dos estados

son *indistinguibles* si para cualquier palabra $\sigma \in Ent^*$,

$\equiv_I$

Si dos estados

$\sigma \in Ent^*$

tal que no fueran indistinguibles, entonces habría una palabra

$\sigma, \tau \in Ent^*$

Diremos que tal palabra *distingue* a los estados q_1, q_2. La relación de indistinguibilidad es

una relación de equivalencia. La denotaremos como `` ". $L = L(AF)$ el

Sea

$AFM_L = (E$

lenguaje reconocido por el autómata AF . Para cualesquiera dos palabras

rige la equivalencia siguiente

$(3. \Rightarrow 1.)$

Por tanto el autómata $AFM_I = (Q^{con} / \equiv_I)$ construído en la demostración de la

$h : T(\sigma) \mapsto$

implicación del teorema de Myhill-Nerode 4.6. 1 es isomorfo al autómata

AFM_L

cociente y, por consiguiente, es una imagen homomorfa de la parte conexa del

AFM_I

AF ,

autómata es un homomorfismo. De hecho, $\{tran(q_1,$ es una imagen homomorfa de cualquier autómata que reconozca a *L* . Así pues, resulta de inmediato la

Proposición 6.2 El autómata mínimo que reconoce a un lenguaje regular *L* es

$$AFM_I = (Q^{con}/ \equiv_I)$$

.

Minimización de autómatas

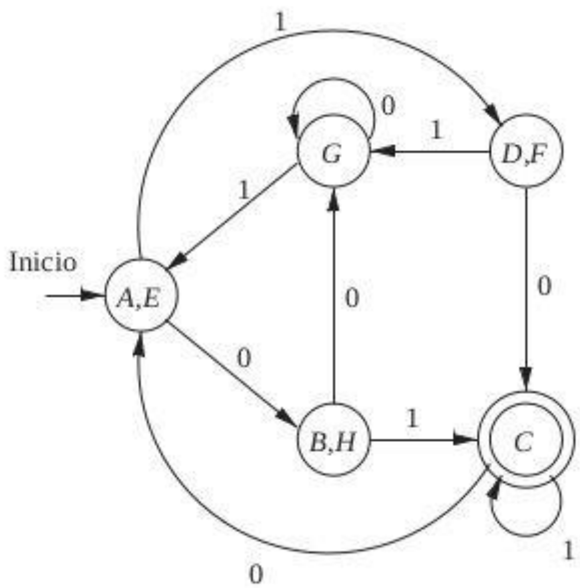
Construcción de un AFDt con un número de estados mínimo que sea equivalente a un AFDt dado. Definiciones previas: • Estados accesibles: q0 es accesible q accesible ⇒ ∀s∈Σ, δ(q, s) es accesible • Estados k-equivalentes o k-indistinguibles: p ≡k q ∀x∈Σ≤k (δ*(p, x)∈F ↔ δ*(q, x) ∈ F) • Estados equivalentes o indistinguibles: p ≡ q ∀k ∀x∈Σ≤k (δ*(p, x)∈F ↔ δ*(q, x) ∈F), es decir, ∀x∈Σ* (δ*(p, x)∈F ↔ δ*(q, x)∈F)

Construcción del AFDt mínimo N a partir del AFDt M = (Q, Σ, δ, q0, F) 1) Eliminar estados inaccesibles 2) Determinar las clases de estados equivalentes: p ≡0 q ⇔ (p∈F ↔ q∈F) p ≡k+1 q ⇔ (p ≡k q ∧ ∀s∈Σ δ(p, s) ≡k δ(q, s)) 3) Construcción del AFD N = (P, Σ, γ, p0, G) con P = Q/≡ siendo ≡ es la menor ≡k tal que ≡k coincide con ≡k+1 p0 = [q0] γ([p], s) = [δ(p, s)] G = {[p]: p∈F}

Minimización de un AFD

Dos pasos:

1. Eliminación de los estados no alcanzables
2. Búsqueda de estados equivalentes
3. Construcción de un autómata a partir de los grupos de estados equivalentes



Ejercicio 1

Considerar el automata siguiente:

```
        0   1
---------------
->  A | B | A
B | A | C
C | D | B
  * D | D | A
E | D | F
F | G | E
G | F | G
H | G | D
```

1. Dibuje la tabla de estados distinguibles para este autómata.
2. Construya el AFD equivalente con el número mínimo de estados.

Ejercicio 2

Mismo ejercicio con:

```
       0   1
---------------
->  A | B | E
    B | C | F
* C | D | H
      D | E | H
      E | F | I
* F | G | B
      G | H | B
      H | I | C
      I | A | E
```