INF155 - Informática Teórica

Ayudantía #4

NP-Complete Warriors

Semestre 2022-2

UNIVERSIDAD TECNICA FEDERICO SANTA MARIA
DEPARTAMENTO DE INFORMÁTICA

Motivación

Los autómatas son una herramienta simple pero poderosa para poder representar lenguajes. Aunque los NFAs nos permiten aprovechar los beneficios del no-determinismo, los computadores en la realidad son inherentemente deterministas. Si tenemos un DFA gigante, consumiremos muchos más recursos de los necesarios. Es por ello que surge la necesidad de optimizar la cantidad de estados de los mismos, ya que de esta forma nos aseguramos que el costo computacional requerido para realizar tareas con él es el **mínimo** posible.

DFA mínimo - Algoritmo de Moore

El teorema de Myhill-Nerode permite obtener el DFA mínimo a partir de análisis de clases de equivalencias y relaciones (lo cual es bonito pero muy complejo e induce errores). Sin embargo, Moore diseñó un algoritmo que se basa directamente (más o menos) en dicho teorema, y es mucho más comprensible. Los pasos a seguir son los siguientes:

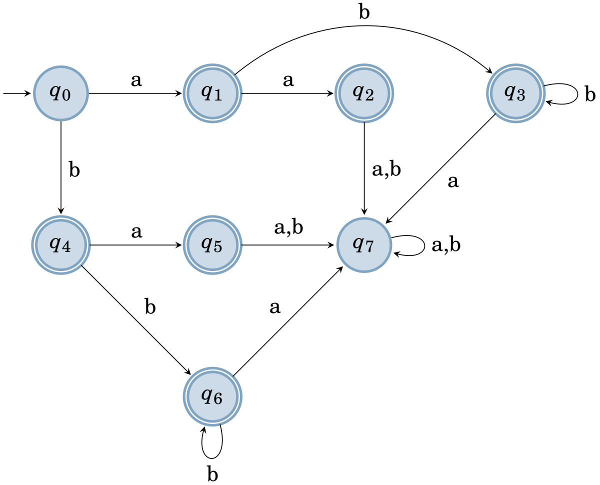
1. Dividir el conjunto de estados Q en dos conjuntos: uno con los estados no finales, y el otro con los estados finales.
2. Construir la tabla de transición de estados. El resultado de salida será el conjunto de estados al que se accede desde un estado consumiendo un símbolo.

DFA mínimo - Algoritmo de Moore

3. Comparar las filas de cada grupo en la tabla. Si en un grupo hay filas diferentes, debe ser subdividido en tantos grupos como filas distintas existan y se repite el paso 2. Las filas iguales corresponden a un subgrupo.
4. Si para cada grupo todas las filas que lo componen son iguales, entonces el algoritmo termina su ejecución y se obtuvo el DFA mínimo. El nuevo estado inicial será aquel grupo que contiene el estado inicial del DFA original, mientras que los nuevos estados finales serán aquellos grupos que contengan algún estado final del autómata original.

Algoritmo de Moore - Ejemplo

Minimicemos el siguiente DFA:



Algoritmo de Moore - Ejemplo

1. Separamos los estados finales de los no finales:

$$A = \{q_0, q_7\}$$
$$B = \{q_1, q_2, q_3, q_4, q_5, q_6\}$$

2. Reescribimos la tabla de transiciones:

Conjunto	Estado	a	b	Tipo
A	q_0	B	B	Inicial
	q_7	A	A	
B	q_1	B	B	Final
	q_2	A	A	Final
	q_3	A	B	Final
	q_4	B	B	Final
	q_5	A	A	Final
	q_6	A	B	Final

Algoritmo de Moore - Ejemplo

Ahora tenemos que analizar cada conjunto por separado. En el conjunto A tenemos solo dos combinaciones posibles: BB y AA. Luego, del conjunto A podemos definir dos nuevos conjuntos:

$$A = \{q_0\}$$
$$B = \{q_7\}$$

Aplicamos el mismo procedimiento al conjunto B, obtenemos los siguientes conjuntos:

$$C = \{q_1, q_4\}$$
$$D = \{q_2, q_5\}$$
$$E = \{q_3, q_6\}$$

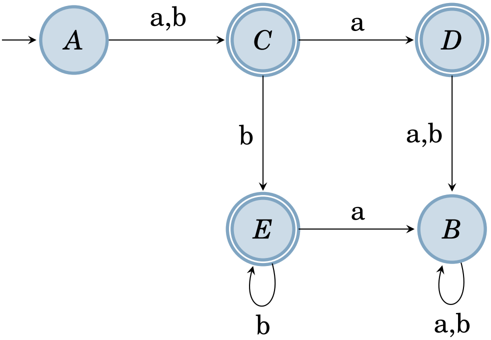
Algoritmo de Moore - Ejemplo

Reescribimos la tabla de transiciones:

Conjunto	**Estado**	**a**	**b**	**Tipo**
A	q_0	C	C	Inicial
B	q_7	B	B	
C	q_1	D	E	Final
	q_4	D	E	Final
D	q_2	B	B	Final
	q_5	B	B	Final
E	q_3	B	E	Final
	q_6	B	E	Final

Algoritmo de Moore - Ejemplo

Notamos que cada conjunto (A, B, C, D, E) tiene una única combinación, por lo que terminamos de aplicar el algoritmo y pasamos a dibujar nuestro DFA mínimo:



Recordatorio

Sea $\mathscr{L}$ un lenguaje regular, y $N > 0$ una constante

Escogemos $\sigma \in \mathscr{L}$, tal que $|\sigma| \geq N$,
entonces $\sigma = \alpha\beta\gamma, |\alpha\beta| \leq N$ y $\beta = \varepsilon$

$\Rightarrow \forall k \geq 0, \alpha\beta^k\gamma \in \mathscr{L}$

Recordatorio

El lema del bombeo es una herramienta que busca demostrar que un lenguaje **NO** es regular. Esto nos deja un conjunto de herramientas para analizar la regularidad de un lenguaje, pero aún quedan otras más.

Las propiedades sobre operaciones vistas previamente sobre conjuntos y las siguientes son otra herramienta que nos ayuda a probar la regularidad de un lenguaje.

Sustitución

Sea $\mathscr{L}$ un lenguaje sobre Σ y sean $\mathscr{L}_a$ lenguajes sobre Δ, para todo $a \in \Sigma$. La *sustitución* σ dada por $\mathscr{L}_a$ se define para la palabra $\alpha = a_1 a_2 \ldots a_n$ por:

$$a_1 a_2 \ldots a_n \mapsto \mathscr{L}_{a_1} \mathscr{L}_{a_2} \ldots \mathscr{L}_{a_n}$$

Esto puede extenderse a lenguajes en forma natural:

$$\sigma(\mathscr{L}) = \bigcup_{\alpha \in \mathscr{L}} \sigma(\alpha)$$

Homomorfismo

Un *homomorfismo* de Σ^* a Δ^* es una función $h\colon \Sigma^* \to \Delta^*$ tal que $h(\alpha\beta) = h(\alpha)h(\beta)$ para todo $\alpha, \beta \in \Sigma^*$

Observaciones

- Un homomorfismo $h\colon \Sigma^* \to \Delta^*$ reemplaza cada símbolo de Σ por una palabra sobre Δ.
- Los homomorfismos conservan la operación de concatenación.
- Se cumple que $h(\varepsilon) = \varepsilon$, y por la observación anterior, también se cumple que $h(a_1 a_2 \ldots a_n) = h(a_1)h(a_2)\cdots h(a_n)$.
- Se debe definir h para cada $a \in \Sigma$.

Clausura

- Un conjunto $\mathscr{U}$ se llama *cerrado* respecto de una operación si siempre que se operan entre sí elementos del conjunto el resultado nuevamente pertenece al conjunto.

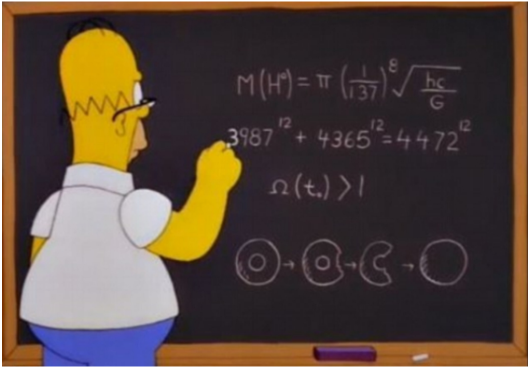
→ Los números enteros son cerrados respecto a la suma (es decir, si sumo dos enteros, el resultado es un número entero).

→ Los números enteros NO son cerrados respecto a la división. Por ejemplo, 1/3 es racional, pero no entero (*“nos salimos del conjunto de origen”*).

Propiedades de clausura

- Si $\mathscr{L}_1$, $\mathscr{L}_2$ son regulares, entonces $\mathscr{L}_1 \cup \mathscr{L}_2$, $\mathscr{L}_1 \cdot \mathscr{L}_2$, $\mathscr{L}_1^*$ y $\mathscr{L}_1^+$ son regulares.
- Si $\mathscr{L}$ es regular, entonces $\overline{\mathscr{L}}$ es regular.
- Si $\mathscr{L}$ es regular, entonces $\mathscr{L}^R$ es regular.
- Si $\mathscr{L}_1$ y $\mathscr{L}_2$ son regulares, entonces $\mathscr{L}_1 \cap \mathscr{L}_2$ es regular.
- Si $\mathscr{L}_1$ y $\mathscr{L}_2$ son regulares, entonces $(\mathscr{L}_1 \smallsetminus \mathscr{L}_2)$ es regular.
- Si $\mathscr{L}_1$ y $\mathscr{L}_2$ son regulares, entonces $(\mathscr{L}_1 \triangle \mathscr{L}_2)$ es regular.
- Si $\mathscr{L}$ es regular, entonces la sustitución $\sigma(\mathscr{L})$ de lenguajes regulares es regular
- En consecuencia del punto anterior, si $\mathscr{L}$ es regular, entonces el homomorfismo $h(\mathscr{L})$ también lo es.
- Si $\mathscr{L}$ es regular, entonces el homomorfismo inverso $h^{-1}(\mathscr{L})$ es regular.

Ejercicios


$M(H^0) = \pi \left(\frac{1}{137}\right)^8 \sqrt{\frac{hc}{G}}$
$3987^{12} + 4365^{12} = 4472^{12}$
$\Omega(t_0) > 1$

Ejercicio 1

Pruebe que si $\mathscr{L}$ es regular, entonces el lenguaje

$$\mathscr{L}_1 = \{u \cdot v : u \in \mathscr{L}, v \in \mathscr{L}^R\}$$

también es regular.

Solución Ejercicio 1

- $\mathscr{L}$ es regular, por lo que existe un DFA que acepta a u.
- $\mathscr{L}^R$ también es regular, pues existe un NFA que lo reconoce. Este se construye a partir del DFA que reconoce a $\mathscr{L}$, invirtiendo el sentido de los arcos y agregando un nuevo estado (que será nuestro nuevo estado inicial) donde convergen todos los estados finales a través de transiciones ε. El nuevo estado final es el estado inicial del DFA original. Por lo tanto, v es aceptado por un NFA.

Solución Ejercicio 1

Por lo tanto, $\mathscr{L}_1$ es simplemente $\mathscr{L} \cdot \mathscr{L}^R$, ambos lenguajes regulares y por propiedad de clausura la concatenación de dos lenguajes regulares da como resultado un lenguaje regular.

Ejercicio 2

¿Cuál es las siguientes proposiciones es verdad para todo lenguaje regular y homomorfismo?

a) $h(\mathscr{L}_1\mathscr{L}_2) = h(\mathscr{L}_1)h(\mathscr{L}_2)$
b) $h(\mathscr{L}_1 \cup \mathscr{L}_2) = h(\mathscr{L}_1) \cup h(\mathscr{L}_2)$
c) $h(\mathscr{L}_1 \cap \mathscr{L}_2) = h(\mathscr{L}_1) \cap h(\mathscr{L}_2)$

Solución Ejercicio 2 a

$h(\mathscr{L}_1\mathscr{L}_2) = h(\mathscr{L}_1)h(\mathscr{L}_2)$ es básicamente la definición de homomorfismo extendida a lenguajes en lugar de palabras, por lo tanto se cumple para todo lenguaje regular y homomorfismo.

Parte de la definición: $h(\alpha\beta) = h(\alpha)h(\beta)$ para todo $\alpha, \beta \in \Sigma^*$.

Si $\alpha \in \mathscr{L}_1$ y $\beta \in \mathscr{L}_2$, obtendremos lo pedido.

Solución Ejercicio 2 b

Nuevamente justificando en base a la definición de homomorfismo, un homomorfismo h para lenguajes es $h(\mathscr{L}) = \{h(\alpha) : \alpha \in \mathscr{L}\}$.

Podemos reescribir $\mathscr{L}$ como $\mathscr{L}_1 \cup \mathscr{L}_2$, por lo tanto:

$$
\begin{aligned}
h(\mathscr{L}) &= h(\mathscr{L}_1 \cup \mathscr{L}_2) \\
h(\mathscr{L}_1 \cup \mathscr{L}_2) &= \{h(\alpha) : \alpha \in \mathscr{L}_1 \cup \mathscr{L}_2\} \\
&= \{h(\alpha) : \alpha \in \mathscr{L}_1\} \cup \{h(\alpha) : \alpha \in \mathscr{L}_2\} \\
&= h(\mathscr{L}_1) \cup h(\mathscr{L}_2)
\end{aligned}
$$

Solución Ejercicio 2 c

Este es el único caso en donde no se cumple. Demostración por contraejemplo:

Sean $\mathscr{L}_1 = \mathscr{L}(a^*)$, $\mathscr{L}_2 = \mathscr{L}(b^*)$ y sea el homomorfismo h definido como:

$$h : \begin{cases} h(a) = b \\ h(b) = b \end{cases}$$

Note que $h(\mathscr{L}_1) = \mathscr{L}_2$ y $h(\mathscr{L}_2) = \mathscr{L}_2$

Solución Ejercicio 2 c

- $h(\mathscr{L}_1) \cap h(\mathscr{L}_2) = \mathscr{L}_2$
- $\mathscr{L}_1 \cap \mathscr{L}_2 = \{\varepsilon\} \therefore h(\mathscr{L}_1 \cap \mathscr{L}_2) = \{\varepsilon\}$
- $h(\mathscr{L}_1) \cap h(\mathscr{L}_2) \neq h(\mathscr{L}_1 \cap \mathscr{L}_2)$

Ejercicio 3

Demuestre si el lenguaje $\mathscr{L} = \{a^n c^k b^{n+k}, n, k \geq 0\}$ es regular o no.

Solución Ejercicio 3

Podemos demostrar que el lenguaje no es regular usando nuestro querido lema del bombeo de forma trivial pero también hemos aprendido nuevas herramientas que son tan útiles como la primera.

Solución Ejercicio 3

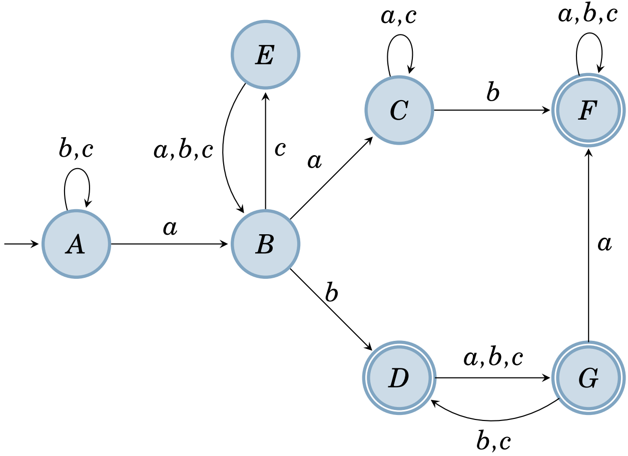
Suponemos $\mathscr{L}$ regular y definimos el homomorfismo h como:

$$h : \begin{cases} h(a) = a \\ h(b) = b \\ h(c) = a \end{cases}$$

Entonces $h(\mathscr{L}) = \{a^{n+k}b^{n+k}, n, k \geq 0\} = \{a^k b^k, k \geq 0\}$ debería ser regular por propiedades de clausura pero sabemos que no es regular, por lo tanto suponer que $\mathscr{L}$ era incorrecto. Esto significa que $\mathscr{L}$ no es regular.

Ejercicio 4

Minimice el siguiente DFA aplicando el algoritmo de Moore:



Solución Ejercicio 2

Separamos los estados en **estados no finales** (α) y **estados finales** (β):

$$\alpha = \{A, B, C, E\}$$
$$\beta = \{D, F, G\}$$

Reescribimos la tabla de transiciones:

		a	b	c	Tipo
α	A	α	α	α	Inicial
	B	α	β	α	
	C	α	β	α	
	E	α	α	α	
β	D	β	β	β	Final
	F	β	β	β	
	G	β	β	β	

Solución Ejercicio 2

El grupo β está listo, mientras que el grupo α debe ser descompuesto (ya que hay diferentes transiciones para los estados que pertenecen a él). Catalogamos los nuevos grupos:

$$\gamma = \{A, E\}$$
$$\delta = \{B, C\}$$
$$\beta = \{D, F, G\}$$

Solución Ejercicio 2

Actualizamos la tabla de transiciones:

		a	b	c	Tipo
γ	A	δ	γ	γ	Inicial
	E	δ	δ	δ	
δ	B	δ	β	γ	
	C	δ	β	δ	
	D	β	β	β	
β	F	β	β	β	Final
	G	β	β	β	

Solución Ejercicio 2

El grupo γ y δ deben descomponerse:

$$\mu = \{A\}$$
$$\lambda = \{E\}$$
$$\omega = \{B\}$$
$$\psi = \{C\}$$
$$\beta = \{D, F, G\}$$

Solución Ejercicio 2

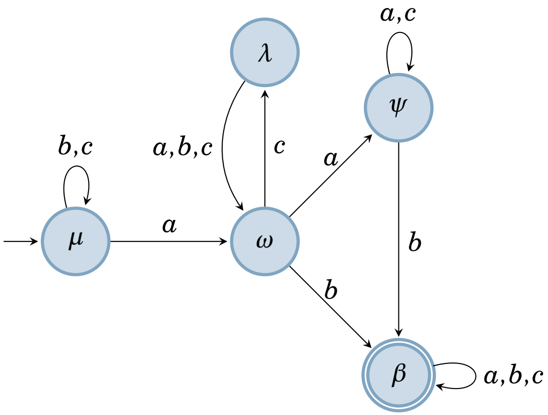
Actualizamos la tabla de transiciones:

		a	b	c	Tipo
μ	A	ω	μ	μ	Inicial
λ	E	ω	ω	ω	
ω	B	ψ	β	λ	
ψ	C	ψ	β	ψ	
	D	β	β	β	
β	F	β	β	β	Final
	G	β	β	β	

Todos los grupos tienen solo una combinación, por lo que el algoritmo finaliza.

Solución Ejercicio 2

En forma gráfica, el DFA mínimo obtenido es:



¿Dudas?

INF155 - Informática Teórica

Ayudantía #4

NP-Complete Warriors

Semestre 2022-2

UNIVERSIDAD TECNICA FEDERICO SANTA MARIA

DEPARTAMENTO DE INFORMÁTICA