Autómatas y Lenguajes Formales 2017-1
Facultad de Ciencias UNAM
Nota de Clase 4

Favio E. Miranda Perea A. Liliana Reyes Cabello Lourdes González Huesca

31 de agosto de 2016

1. Autómatas No-Deterministas

El determinismo de un autómata, deseable desde el punto de vista teórico, puede provocar complicaciones en la práctica. Veamos las diferencias entre estos conceptos:

- **Determinismo:** dado un estado q y un símbolo a existe una única transición $\delta(q, a) = p$, es decir δ es una función total.
- **No-determinismo:** no hay una transición única al leer un símbolo a en un estado dado q.

El no-determinismo se traduce en que hay más de una transición al leer un símbolo, es decir, $\delta(q, a)$ deja de ser función. O bien no hay transición, es decir, $\delta(q, a)$ no está definida (δ se vuelve función parcial). Sin embargo la máquina funciona únicamente al leer un símbolo y podemos decir que existe el no-determinismo sin lectura de símbolos.

Veamos cómo se puede agregar el no-determinismo a la definción de autómata que vimos anteriormente:

Definición 1 *Un autómata finito **no** determinista (AFN) es una quintupla* $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ *donde*

- Q *es un conjunto finito de estados.*
- Σ *es el alfabeto de entrada.*
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ *es la función de transición.*
- $q_0 \in Q$ *es el estado inicial.*
- $F \subseteq Q$ *es el conjunto de estados finales.*

Obsérvese que la imagen de δ es ahora un elemento de $\mathcal{P}(Q)$, es decir es un subconjunto de estados de Q. Además $\delta(q, a) = \{q_1, q_2, \ldots, q_n\}$ indica que al leer el símbolo a en el estado q la máquina puede pasar a cualquiera de los estados $q_1, \ldots, q_n$. Si $\delta(q, a) = \varnothing$ entonces no hay transición posible desde el estado q al leer a, es decir, la máquina está bloqueada.
Veamos como las nociones vistas para los AFD se modifican:

Definición 2 *Sea $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ un AFN. La función de transición δ se extiende a cadenas mediante una función $\delta^\star : Q \times \Sigma^\star \to \mathcal{P}(Q)$ definida recursivamente como sigue:*

- $\delta^\star(q, \varepsilon) = q$
- $\delta^\star(q, wa) = \bigcup_{p \in \delta^\star(q,w)} \delta(p, a)$
  Alternativamente: $\delta^\star(q, aw) = \bigcup_{p \in \delta(q,a)} \delta^\star(p, w)$

El lenguaje de aceptación se define mediante $\delta^\star$ como sigue:

$$L(M) = \{w \in \Sigma^\star \mid \delta^\star(q_0, w) \cap F \neq \varnothing\}$$

Es decir, $w \in L(M)$ si y sólo si existe al menos un cómputo o un procesamiento de w que conduce a un estado final al iniciar la máquina en q_0.

1.1. Eliminación del no-determinismo

Una pregunta natural es comparar dos autómatas, uno determinista y el otro no. Así también buscar una forma de eliminar el no-determinismo y conservar una máquina que acepte un lenguaje dado.

Todo AFD es a la vez un AFN con la particularidad de que $\delta(p, a)$ consta de un único estado. La idea para transformar un AFN en un AFD es considerar a cada conjunto de estados $\delta(p, a)$ del AFN como un único estado del nuevo AFD. A este método se conoce como la *construcción de subconjuntos*.

Definición 3 *Dado un AFN $M = \langle Q, \Sigma, \delta_N, q_0, F\rangle$ definimos un AFD $M^d = \langle Q^d, \Sigma, \delta, q_0^d, F^d\rangle$ como sigue:*

- $Q^d = \mathcal{P}(Q)$
- $\delta(S, a) = \delta_N(S, a) = \bigcup_{q \in S} \delta_N(q, a)$
- $q_0^d = \{q_0\}$
- $F^d = \{S \subseteq Q \mid S \cap F \neq \varnothing\}$

Ambos autómatas son equivalentes, es decir, $L(M) = L(M^d)$.

2. AFN con ε-transiciones

Otra de las máquinas que son útiles para procesar cadenas son las que permiten procesar cadenas vacías, no sólo al tener como estado final a q_0 sino que permiten procesar (sub)cadenas vacías en una parte itermedia. De esta forma se definen los autómatas con transiciones etiquetadas con la cadena ε:

Definición 4 *Un autómata finito **no** determinista con ε-transiciones (AFNε) es una quintupla $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ donde*

- *Q es un conjunto finito de estados.*

- Σ *es el alfabeto de entrada.*
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ *es la función de transición.*
- $q_0 \in Q$ *es el estado inicial.*
- $F \subseteq Q$ *es el conjunto de estados finales.*

De la definición de $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ se observa que la única diferencia está en el dominio de δ. Es decir que las transiciones de la forma $\delta(q, \varepsilon) = a$ están permitidas e indican que la máquina puede cambiar de estado *sin* leer ningún símbolo. Esto causa también un no-determinismo, que es aun más complicado de modelar matemáticamente pero tiene grandes ventajas:

- Se permiten múltiples cómputos para una cadena de entrada.
- Pueden existir cómputos bloqueados.
- A diferencia de los AFD y AFN simples, pueden existir cómputos infinitos, es decir, surge la **no-terminación**.
- La presencia de $\boldsymbol{\varepsilon}$-transiciones permite mayor libertad en el diseño.

Para extender la definición de transiciones a procesamiento de cadenas es necesario introducir un concepto previo que considera los estados a los que la máquina puede llegar al incluir las cadenas vacías.

Definición 5 *Dado un estado q, definimos la $\boldsymbol{\varepsilon}$-cerradura de q como el conjunto de estados alcanzables desde q mediante cero o más $\boldsymbol{\varepsilon}$-transiciones. Es decir*

$$Cl_{\boldsymbol{\varepsilon}}(q) = \{s \in Q \mid \exists p_1, \ldots, p_n \text{ con } p_1 = q,\ p_n = s,\ p_i \in \delta(p_{i-1}, \boldsymbol{\varepsilon})\}$$

Recursivamente:

- $q \in Cl_{\boldsymbol{\varepsilon}}(q)$
- *Si* $r \in Cl_{\boldsymbol{\varepsilon}}(q)$ *y* $\delta(r, \boldsymbol{\varepsilon}) = s$ *entonces* $s \in Cl_{\boldsymbol{\varepsilon}}(q)$

Esta definición se extiende a conjuntos de estados como sigue:

$$Cl_{\boldsymbol{\varepsilon}}(S) = \bigcup_{q \in S} Cl_{\boldsymbol{\varepsilon}}(q)$$

Con la $\boldsymbol{\varepsilon}$-cerradura de estados se puede definir la extensión de δ:

Definición 6 *Sea $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ un AFN$\boldsymbol{\varepsilon}$. La función de transición δ se extiende a cadenas mediante una función*

$$\delta^\star : Q \times (\Sigma^\star \cup \{\boldsymbol{\varepsilon}\}) \to \mathcal{P}(Q)$$

definida recursivamente como sigue:

- $\delta^\star(q, \boldsymbol{\varepsilon}) = Cl_{\boldsymbol{\varepsilon}}(q)$
- $\delta^\star(q, wa) = Cl_{\boldsymbol{\varepsilon}}\Big(\bigcup_{q' \in \delta^\star(q,w)} \delta(q', a)\Big)$

2.1. Eliminación de ε-transiciones

Estudiaremos el proceso de eliminación de transiciones $\boldsymbol{\varepsilon}$. Esta eliminación implica que un AFN es también un AFN$\boldsymbol{\varepsilon}$. Es decir que cualquier AFN$\boldsymbol{\varepsilon}$ es inmediatamente un AFN con la particularidad de que no existen $\boldsymbol{\varepsilon}$-transiciones.

Definición 7 *Dado un AFN$\boldsymbol{\varepsilon}$, $M_1 = \langle Q_1, \Sigma, \delta_1, q_0, F_1\rangle$ existe un AFN equivalente $M = \langle Q, \Sigma, \delta, p_0, F\rangle$, definido mediante:*

- $Q := Q_1$
- $p_0 := q_0$
- $\delta(q,a) = \delta_1^\star(q,a) = Cl_{\boldsymbol{\varepsilon}}\Big(\bigcup_{p\in Cl_{\boldsymbol{\varepsilon}}(q)} \delta_1(p,a)\Big)$
- $F := F_1 \cup \{q_0\}$ *si* $Cl_{\boldsymbol{\varepsilon}}(q_0) \cap F_1 \neq \varnothing$
  $F := F_1$ *en caso contrario.*

3. Equivalencias

Nuevamente surgen preguntas como la equivalencia entre autómatas deterministas, no deterministas y con transiciones $\boldsymbol{\varepsilon}$.

En la subsección 1.1 se estudió la forma de eliminar el no-determinismo de un autómata, creando un AFD. Es decir AFN $\Rightarrow$ AFD.

En la subsección pasada se revisó la forma de eliminar $\boldsymbol{\varepsilon}$-transiciones y de esta forma obtener un AFN. Así también se obsevó que un AFN es exactamente un AFN$\boldsymbol{\varepsilon}$ ya que incluye el no-determinismo pero sin transiciones de la cadena vacía.

Esta equivalencia, AFN $\Leftrightarrow$ AFN$\boldsymbol{\varepsilon}$, cierra el ciclo de equivalencias de autómatas finitos. Cualquier tipo de autómata finito puede convertirse en un AFD y viceversa, es decir:

$$\text{AFD} \Leftrightarrow \text{ AFN} \Leftrightarrow \text{ AFN}\boldsymbol{\varepsilon}$$

Nuestra siguiente meta es probar el Teorema de Kleene, el cual es uno de los resultados más importantes en la Teoría de la Computación pues asegura la equivalencia entre dos de nuestros tres conceptos fundamentales: los autómatas finitos y los lenguajes regulares.