Autómatas y Lenguajes Formales
Nota 05. Minimización de un AFD*

Noé Salomón Hernández S.

1. La construcción del autómata cociente

Desarrollaremos en esta nota un método mecánico para encontrar estados equivalentes en un AFD, los cuales colapsaremos. Cada lenguaje regular tiene un AFD mínimo que es único a excepción de isomorfismos, y hay un método mecánico para construirlo.

Son dos las condiciones por la que podemos colapsar estados:

1. Nunca colapsamos a un estado final p y un estado de rechazo q porque si $p = \widehat{\delta}(q_0, x) \in F$ y $q = \widehat{\delta}(q_0, y) \notin F$, entonces x debe ser aceptada y y rechazada después de colapsar los estados, así que no hay manera de establecer al estado colapsado como final o de rechazo sin incurrir en un error.

2. Si colapsamos p y q, entonces debimos haber colapsado $\delta(p, a)$ y $\delta(q, a)$ para mantener el determinismo.

Formalmente tenemos la siguiente definición.

Definición 1.1 Dado el AFD $M = (Q, \Sigma, \delta, q_0, F)$ decimos que $p, q \in Q$ están relacionados, y escribimos $p \approx q$, si y sólo si:

$$\forall x \in \Sigma^*,\ \ \widehat{\delta}(p, x) \in F \iff \widehat{\delta}(q, x) \in F$$

Dicha relación es de equivalencia, por lo que el conjunto de estados se particiona en *clases de equivalencia*:

$$[p] = \{q \in Q \mid q \approx p\}$$

Así definimos el AFD $M_{/\approx} = (Q', \Sigma, \delta', q_0', F')$, conocido como *autómata cociente*, donde:

$$\begin{aligned} Q' &= \{[p] \mid p \in Q\}, \\ \delta'([p], a) &= [\delta(p, a)], \\ q_0' &= [q_0], \\ F' &= \{[p] \mid p \in F\}. \end{aligned}$$

*Esta nota se basa en el libro: D. C. Kozen. *Automata and Computability*, Springer-Verlag, Inc., New York, NY, 1997.

Vemos que los estados de $M_{/\approx}$ son las clases de equivalencia de $\approx$, ésta es la forma matemática de *colapsar* estados equivalentes. Es concebible que una elección diferente del representante de la clase $[p]$ pueda producir algo distinto a lo que se tiene en el lado derecho de $\delta'([p], a) = [\delta(p, a)]$. El siguiente lema indica que esto no puede ocurrir.

Lema 1.2 *Si $p \approx q$, entonces $\delta(p, a) \approx \delta(q, a)$, con $a \in \Sigma$. Es decir, si $[p] = [q]$, entonces $[\delta(p, a)] = [\delta(q, a)]$.*

Demostración. Supongamos que $p \approx q$. Queremos demostrar que $\delta(p, a) \approx \delta(q, a)$. Sea $x \in \Sigma^*$, ocuparemos la siguiente definición de $\widehat{\delta}$: $\boxed{\widehat{\delta}(q, \varepsilon) = q \text{ y } \widehat{\delta}(q, ax) = \widehat{\delta}(\delta(q, a), x)}$. Tenemos:

$$\begin{aligned} \widehat{\delta}(\delta(p, a), x) \in F &\Leftrightarrow \widehat{\delta}(p, ax) \in F \\ &\Leftrightarrow \widehat{\delta}(q, ax) \in F \qquad \text{pues } p \approx q \\ &\Leftrightarrow \widehat{\delta}(\delta(q, a), x) \in F. \end{aligned}$$

Como la x es cualquiera, $\delta(p, a) \approx \delta(q, a)$ por definición de $\approx$. $\dashv$

Lema 1.3 $p \in F \Leftrightarrow [p] \in F'$.

Demostración. La demostración de $\Rightarrow$ es inmediata a partir de la definición de F', ya que si $p \in F$, entonces la definición nos dice que la clase de equivalencia de $[p]$ es un estado final de F'.

Para demostrar la dirección $\Leftarrow$, tenemos que demostrar que si $[p]$ está en F', entonces cualquier representante de tal clase de equivalencia está en F. Esto nos dice que cualquier clase de equivalencia inducida por $\approx$ es un subconjunto de F, o su intersección con F es vacía. Supongamos $p \in F$ y $p \approx q$ para algún estado q. Por demostrar que $q \in F$. Esto se sigue inmediatamente tomando $x = \varepsilon$ en la definición de $p \approx q$. $\dashv$

Lema 1.4 *Para toda $x \in \Sigma^*$, $\widehat{\delta'}([p], x) = [\widehat{\delta}(p, x)]$. (Observe la prima en el lado izquierdo, la cual está ausente del lado derecho.)*

Demostración. Por inducción estructural sobre x. Ocuparemos la siguiente definición de $\widehat{\delta}$: $\boxed{\widehat{\delta}(p, \varepsilon) = p \text{ y } \widehat{\delta}(p, wa) = \delta(\widehat{\delta}(p, w), a)}$.

Base. Tomamos $x = \varepsilon$, tenemos

$$\begin{aligned} \widehat{\delta'}([p], \varepsilon) &= [p] && \text{definición de } \widehat{\delta'} \\ &= [\widehat{\delta}(p, \varepsilon)] && \text{definición de } \widehat{\delta}. \end{aligned}$$

Hipótesis de inducción. El lema se cumple para $x = w$, es decir, $\widehat{\delta'}([p], w) \approx [\widehat{\delta}(p, w)]$.

Paso inductivo. Debemos demostrar que se satisface también para $x = wa$, con $a \in \Sigma$.

$$\begin{aligned}
\widehat{\delta'}([p], wa) &= \delta'(\widehat{\delta'}([p], w), a) && \text{definición de } \widehat{\delta'} \\
&= \delta'([\widehat{\delta}(p, w)], a) && \text{hipótesis de inducción} \\
&= [\delta(\widehat{\delta}(p, w), a)] && \text{definición de } \delta' \\
&= [\widehat{\delta}(p, wa)] && \text{definición de } \widehat{\delta}.
\end{aligned}$$

⊣

Teorema 1.5 $L(M_{/\approx}) = L(M)$.

Demostración. Para toda $x \in \Sigma^*$, se tiene

$$\begin{aligned}
x \in L(M_{/\approx}) &\Leftrightarrow \widehat{\delta'}(q_0', x) \in F' && \text{definición de aceptación} \\
&\Leftrightarrow \widehat{\delta'}([q_0], x) \in F' && \text{definición de } q_0' \\
&\Leftrightarrow [\widehat{\delta}(q_0, x)] \in F' && \text{Lema 1.4} \\
&\Leftrightarrow \widehat{\delta}(q_0, x) \in F && \text{Lema 1.3} \\
&\Leftrightarrow x \in L(M) && \text{definición de aceptación.}
\end{aligned}$$

⊣

1.1. $M_{/\approx}$ no puede colapsarse más

Es factible que después de hacer la construcción del autómata cociente en una ocasión, podamos colapsar aún más al repetir el algoritmo. Resulta que una vez es suficiente.

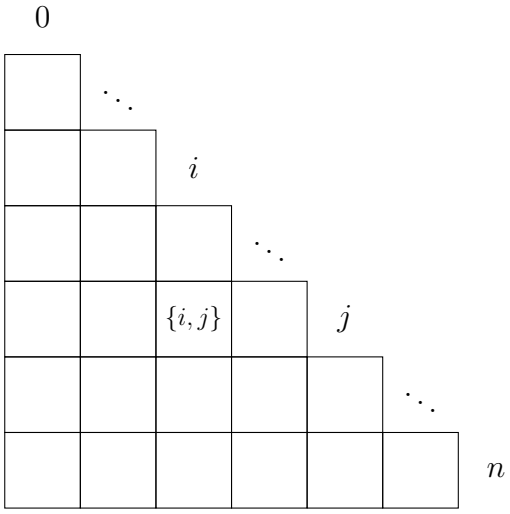
1.2. Un algoritmo de minimización

En seguida se presenta un algoritmo para calcular la relación de colapso $\approx$ dado un AFD M sin estados inaccesibles. Este algoritmo marcará pares (no ordenados) de estados $\{p, q\}$. Un par $\{p, q\}$ se marcará en el momento en el que se descubra una razón por la cual p y q *no* sean equivalentes. El algoritmo es:

1. Escribir una tabla para cada uno de los pares $\{p, q\}$, inicialmente sin marca alguna.
2. Marcar $\{p, q\}$ si $p \in F$ y $q \notin F$, o vice versa.
3. Repetir lo siguiente hasta que no se presenten cambios: si existe un par sin marca $\{p, q\}$ tal que $\{\delta(p, a), \delta(q, a)\}$ está marcado para alguna $a \in \Sigma$, entonces marcar $\{p, q\}$.
4. Al terminar, $p \approx q$ syss $\{p, q\}$ no tiene marca.

Aquí hay algunas observaciones sobre el algoritmo recién descrito:

- Sea $Q = \{q_0, q_1, \ldots, q_n\}$. La tabla que construye el paso 1 del algoritmo se muestra abajo. La intersección de la columna i con el renglón j determina el par de estados $\{i, j\}$.

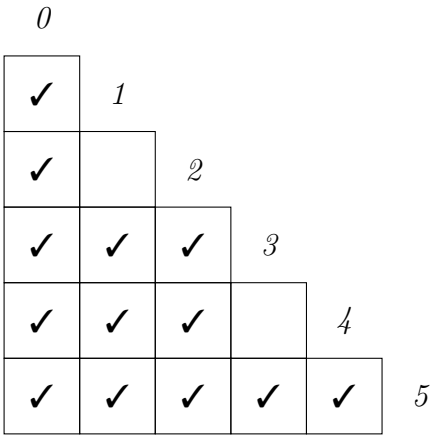


- Si $\{p, q\}$ se marca en el paso 2, entonces p y q no son equivalentes: tomemos $x = \varepsilon$ en la definición de $\approx$ (ver Definición 1.1).
- Quizá sea necesario considerar el par $\{p, q\}$ varias veces en el paso 3, ya que cualquier cambio en la tabla puede hacer que $\{p, q\}$ se marque. Nos detenemos únicamente después de haber hecho una ejecución del paso 3 sin que ocurra cambio alguno en la tabla.
- El algoritmo corre durante un número finito de pasos, puesto que sólo se pueden hacer $\binom{|Q|}{2}$ posibles marcas, y tenemos que hacer al menos una en cada paso del algoritmo para seguir ejecutándolo.
- El paso 4 es un enunciado del teorema que afirma que el algoritmo calcula $\approx$ correctamente.

Ejemplo 1.6 *Considere el AFD representado por la tabla siguiente:*

		a	b
$\rightarrow$	*0*	*1*	*2*
	1F	*3*	*4*
	2F	*4*	*3*
	3	*5*	*5*
	4	*5*	*5*
	5F	*5*	*5*

Al terminar la ejecución del algoritmo de minimización la tabla queda como:



Los pares que quedan sin marcar son $\{1,2\}$ *y* $\{3,4\}$*, indicando que* $1 \approx 2$ *y* $3 \approx 4$.

1.3. El algoritmo anterior que colapsa estados es correcto

Teorema 1.7 *El par* $\{p,q\}$ *es marcado por el algoritmo descrito arriba syss existe* $x \in \Sigma^*$ *tal que* $\widehat{\delta}(p,x) \in F$ *y* $\widehat{\delta}(q,x) \notin F$ *o vice versa, es decir, syss* $p \not\approx q$.

Demostración. La demostración de $\Leftarrow$ se deja como ejercicio. Demostraremos la parte $\Rightarrow$ por inducción fuerte sobre el número de corridas que hace el algoritmo.

Base El par $\{p,q\}$ es marcado durante la primer corrida que hace el algoritmo, es decir, el par $\{p,q\}$ lo marca el punto 2 del algoritmo, por lo que $p \in F$ y $q \notin F$ o vice versa. Entonces, escogemos $x = \varepsilon$ de modo que $\widehat{\delta}(p,\varepsilon) = p \in F$ y $\widehat{\delta}(q,\varepsilon) = q \notin F$ o vice versa.

Hipótesis de inducción Si el par $\{p,q\}$ es marcado durante un número de corrida $\leq m$ del algoritmo, entonces existe $y \in \Sigma^*$ tal que $\widehat{\delta}(p,y) \in F$ y $\widehat{\delta}(q,y) \notin F$ o vice versa.

Paso inductivo Que el par $\{p',q'\}$ sea marcado durante la corrida $m+1$ del algoritmo como dice el paso 3, indica que para alguna $a \in \Sigma$ se tiene que $\{\delta(p',a), \delta(q',a)\}$ fue marcado en alguna corrida previa ($\leq m$) del algoritmo. Por la hipótesis de inducción, existe $y \in \Sigma^*$ tal que $\widehat{\delta}(\delta(p',a),y) \in F$ y $\widehat{\delta}(\delta(q',a),y) \notin F$ o vice versa. Esto significa que $\widehat{\delta}(p',ay) \in F$ y $\widehat{\delta}(q',ay) \notin F$ o vice versa. Así que al considerar $x = ay$ tenemos que $\widehat{\delta}(p',x) \in F$ y $\widehat{\delta}(q',x) \notin F$ o vice versa.

$\dashv$