Introducción a Autómatas Finitos

El Lenguaje Aceptado por un Autómata e Indeterminismo.

Universidad de Cantabria

Esquema

El Problema

Podemos interpretar un autómata como un evaluador de la función característica de un subconjunto de $L \subseteq \Sigma^*$:

$$\chi_L : \Sigma^* \longrightarrow \{0, 1\}$$

El Problema

Por lo tanto siempre nos vamos a centrar en el siguiente problema:

Dado un lenguaje L y una palabra, ¿está esa palabra en el lenguaje?

El Problema

Entrada: Una palabra $\omega \in \Sigma^*$
Salida:

- 1 si el autómata llega a una configuración final aceptadora (i.e., $\delta(q_0, \omega) \in F$).
- 0 si el autómata llega a una configuración final no aceptadora (i.e., $\delta(q_0, \omega) \in Q \setminus F$).

Lenguaje Aceptado por un Autómata

Definición (Lenguaje Aceptado por un Autómata)

Dado un autómata A se define el lenguaje aceptado por A como el conjunto de palabras al que el anterior algoritmo devuelve 1.

El Problema en Forma de Programa

Entrada: $x \in \Sigma^*$ (una palabra sobre el alfabeto).

Inicializar: $I := (q_0, x)$ (la configuración inicial sobre x)

mientras $I \notin F \times \{\lambda\}$ **hacer**
  si $I = (q, x'x_1)$, $x'x_1 \neq \lambda$, **entonces** $I := (\delta(q, x'), x_1)$
    en otro caso devolver NO
  finaliza si
finaliza mientras
devolver YES

Diversos Ejemplos

Hallar un autómata que acepte solamente las palabras que empiezan por 01, sobre el alfabeto $\Sigma = \{0, 1\}$.

Diversos Ejemplos

Hallar un autómata que acepte solamente las palabras que contengan la subpalabra 00, sobre el alfabeto $\Sigma = \{0, 1\}$.

El Indeterminismo

Este código es fácil de interpretar si no tenemos en cuenta:

- La presencia de $\lambda-$transiciones.
- La indefinición de $I = (\delta(q, x'), x_1)$ por no estar definido $\delta(q, x')$ o por tener más de un valor asociado.

Autómatas con $\lambda-$transiciones

Definición

Se denominan $\lambda-$transiciones a las transiciones de una autómata $A := (Q, \Sigma, q_0, F, \delta)$ dadas por igualdades de la forma:

$$\delta(q, \lambda) = \{p_1, \ldots, p_s\}.$$

Interpretación del Indeterminismo

Una $\lambda-$transición puede interpretarse como que el autómata "adivina" cuales son los siguientes símbolos que van a aparecer.

Interpretación del Indeterminismo

- *NO lee* el contenido de la cinta.
- *Modifica* el estado en la unidad de control.
- *NO borra* el contenido de la celda señalada por la unidad de control.
- *NO se mueve* a la izquierda.

Grafo de $\lambda-$Transiciones

Recordemos que un grafo es un par (V, E), donde V es el conjunto de vertices y E es el conjunto de aristas.

Grafo de $\lambda-$Transiciones

Queremos saber que cual como opera el autómata sin leer de la cinta. Queremos saber cual es el la configuración al acabar de leer la palabra (si es que termina de leer la palabra).

Grafo de $\lambda-$Transiciones

- $V = Q$.
- Dados $p, q \in V$, decimos que $(p, q) \in E$ si $q \in \delta(p, \lambda)$, i.e.

$$E := \{(p, q) \;:\; q \in \delta(p, \lambda)\}.$$

Clausura Transitiva

La clausura transitiva muestra que nodos se pueden alcanzar de otros nodos en el grafo de $\lambda-$Transiciones.

Clausura Transitiva

Definición

La clausura transitiva de un nodo (estado) se define del modo siguiente:

$$\lambda - clausura(p) := \{q \in V \ : \ (p, \lambda) \vdash (q, \lambda)\}.$$

Objetivo

Teorema

Dado cualquier lenguaje L que sea aceptado por un autómata con $\lambda-$transiciones, entonces existe un autómata sin $\lambda-$transiciones que acepta el mismo lenguaje. Más aun, la transformación de un autómata a otro se puede dar algorítmicamente.

Algoritmo de Transformación

Entrada: Autómata $A := (Q, \Sigma, q_0, F, \delta)$.

Inicializar: $\bar{Q} := Q$ y $\overline{q_0} := q_0$.

para cada $p \in Q$ **encontrar** $\lambda - clausura(p)$ **finaliza para**

$\bar{F} := F \cup \{p \ : \ \lambda - clausura(p) \cap F \neq \emptyset\}$.

para cada $p \in Q$ **hacer**

  si $\lambda - clausura(p, a) \neq \emptyset$ **, entonces**

  $$\bar{\delta}(p, a) := \bigcup_{q \in \lambda - clausura(p)} \lambda - clausura(\delta(q, a)).$$

  finaliza si

finaliza para cada

Salida $\bar{A} := (\bar{Q}, \Sigma, \overline{q_0}, \bar{F}, \bar{\delta})$

Determinismo e Indeterminismo

Todavía no hemos acabado con el indeterminismo. Queremos saber que papel juega en nuestros lenguajes. Esta claro que los autómatas indeterministas son útiles pero no son fáciles de utilizar.

Determinismo e Indeterminismo

Sorprendentemente, es posible encontrar autómatas deterministas que acepten los mismos lenguajes que los indeterministas (aunque pagando un precio).

Algoritmo

Empezaremos con un autómata sin $\lambda-$transiciones $A = (Q, \Sigma, q_0, F, \delta)$.

Algoritmo

$\overline{Q} := \mathcal{P}(Q)$ (el espacio de estados es el conjunto de las partes de Q).

Algoritmo

$\overline{F} := \{X \in \overline{Q} \ : \ X \cap F \neq \emptyset\}$ (las configuraciones finales aceptadoras son aquellas que contienen algún estado del espacio F de estados finales aceptadores).

Algoritmo

$\overline{q_0} := \{q_0\}$ (el conjunto formado por la antigua configuración inicial).

Algoritmo

La función de transición

$$\overline{\delta} : \overline{Q} \times \Sigma \longrightarrow \overline{Q}$$

definida mediante:

$$\overline{\delta}(X, a) := \{q \in Q \ : \ \exists q' \in X, \ q = \delta(q', a)\}.$$

Sugerencia

Este es la forma más cómoda de programar el algoritmo, aunque para lápiz y papel sea mejor no añadir todos los estados de una vez, sino ir añadiendo estados a medida que se generen.

Ejemplo

Sea el siguiente autómata indeterminista $A = (Q, \Sigma, q_0, F, \delta)$, donde $Q = \{q_0, q_1, q_2\}$, $F = \{q_2\}$ y δ viene dada por esta tabla:

	a	b
q_0	q_1, q_2	
q_1	q_1	q_1
q_2	q_1	

Solución

El nuevo autómata tiene dos estados adicionales, $q_\emptyset$ y $q_{1,2}$ esta es la nueva función de transición:

	a	b
q_0	$q_{1,2}$	$q_\emptyset$
q_1	q_1	q_1
q_2	q_1	$q_\emptyset$
$q_{1,2}$	q_1	q_1
$q_\emptyset$	$q_\emptyset$	$q_\emptyset$

¿Cuales son los estados finales?