Automi e Linguaggi Formali
Università degli Studi di Padova
Marzo 2024

Come dimostrare che un certo linguaggio sia regolare

Gabriel Rovesti

Tendenzialmente si usano:

- automi a stati finiti (DFA)
- automi non deterministici (NFA)
- espressioni regolari
- grammatiche context-free

Tendenzialmente, la via più semplice è usare un automa, rispettandone la definizione formale e quindi costruirne uno in grado di rispettare le specifiche del linguaggio.
Si tratta di:

- definire l'insieme di stati
- definire l'alfabeto
- definire la funzione di transizione rispettando le specifiche del linguaggio fornito
- definire lo stato iniziale
- definire l'insieme degli stati finali

Alcuni esempi utili per la comprensione:

$$L^R = \{w^R : w \in \Sigma^*\}.$$

(Here $(w_1 \dots w_n)^R = w_n \dots w_1$.) This is one of the standard closure operations, and closure under reversal easily follows from manipulation of regular expressions (which may be regarded as the counterpart of finite automaton transformation to regular expressions) – just reverse the regular expression. But you can also prove closure using NFAs. Suppose that L is accepted by a DFA $\langle \Sigma, Q, F, \delta, q_0 \rangle$. We construct an NFA $\langle \Sigma, Q', F', \delta', q_0' \rangle$, where

- The set of states is $Q' = Q \cup \{q_0'\}$.
- The initial state is q_0'.
- The unique accepting state is q_0.
- The transition function is defined as follows: $\delta'(q_0', \epsilon) = F$, and for any state $q \in Q$ and $\sigma \in \Sigma$, $\delta(q', \sigma) = \{q : \delta(q, \sigma) = q'\}$.

(We can get rid of q_0' if we allow multiple initial states.) The guessing component here is the final state of the word after reversal.

Figure 1: Esempio 1 - Dimostrazione linguaggio regolare

$$R(L) = \{yx \in \Sigma^* : xy \in L\}.$$

Suppose that L is accepted by the DFA $\langle \Sigma, Q, F, \delta, q_0 \rangle$. We construct an NFA $\langle \Sigma, Q', F', \delta', q_0' \rangle$, which operates as follows. The NFA first guesses $q = \delta(q_0, x)$. It then verifies that $\delta(q, y) \in F$ and that $\delta(q_0, x) = q$, moving from y to x non-deterministically. This can be formalized as follows:

- The states are $Q' = \{q_0'\} \cup Q \times Q \times \{1, 2\}$. Apart from the initial state q_0', the states are $\langle q, q_{curr}, s \rangle$, where q is the state that we guessed, q_{curr} is the current state, and s specifies whether we are at the y part of the input (when 1) or at the x part of the input (when 2).
- The final states are $F' = \{\langle q, q, 2 \rangle : q \in Q\}$: we accept when $\delta(q_0, x) = q$.
- The transitions $\delta'(q_0', \epsilon) = \{\langle q, q, 1 \rangle : q \in Q\}$ implement guessing q.
- The transitions $\delta'(\langle q, q_{curr}, s \rangle, \sigma) = \langle q, \delta(q_{curr}, \sigma), s \rangle$ (for every $q, q_{curr} \in Q$ and $s \in \{1, 2\}$) simulate the original DFA.
- The transitions $\delta'(\langle q, q_f, 1 \rangle, \epsilon) = \langle q, q_0, 2 \rangle$, for every $q \in Q$ and $q_f \in F$, implement moving from the y part to the x part. This is only allowed if we've reached a final state on the y part.

Figure 2: Esempio 2 - Dimostrazione linguaggio regolare

1. *Dimostra che se L ed M sono linguaggi regolari sull'alfabeto $\{0,1\}$, allora anche il seguente linguaggio è regolare:*

$$L \sqcap M = \{x \sqcap y \mid x \in L, y \in M \text{ e } |x| = |y|\},$$

*dove $x \sqcap y$ rappresenta l'*and bit a bit *di x e y. Per esempio,* $0011 \sqcap 0101 = 0001$.

Poiché L e M sono regolari, sappiamo che esiste un DFA $A_L = (Q_L, \Sigma, \delta_L, q_L, F_L)$ che riconosce L e un DFA $A_M = (Q_M, \Sigma, \delta_M, q_M, F_M)$ che riconosce M.

Costruiamo un NFA A che riconosce il linguaggio $L \sqcap M$:

- L'insieme degli stati è $Q = Q_L \times Q_M$, che contiene tutte le coppie composte da uno stato di A_L e uno stato di A_M.
- L'alfabeto è lo stesso di A_L e di A_M, $\Sigma = \{0,1\}$.
- La funzione di transizione δ è definita come segue:

  $$\delta((r_L, r_M), 0) = \{(\delta_L(r_L, 0), \delta_M(r_M, 0)), (\delta_L(r_L, 1), \delta_M(r_M, 0)), (\delta_L(r_L, 0), \delta_M(r_M, 1))\}$$
  $$\delta((r_L, r_M), 1) = \{(\delta_L(r_L, 1), \delta_M(r_M, 1))\}$$

  La funzione di transizione implementa le regole dell'and tra due bit: l'and di due 1 è 1, mentre è 0 se entrambi i bit sono 0 o se un bit è 0 e l'altro è 1.
- Lo stato iniziale è (q_L, q_M).
- Gli stati finali sono $F = F_L \times F_M$, ossia tutte le coppie di stati finali dei due automi.

Figure 3: Esempio 3 - Dimostrazione linguaggio regolare