Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, bianco e nero

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, bianco, algebra

Descrizione generata automaticamenteImmagine che contiene testo, Carattere, bianco, bianco e nero

Descrizione generata automaticamenteImmagine che contiene testo, Carattere, schermata, documento

Descrizione generata automaticamente

Immagine che contiene testo, carta, lettera, Carattere

Descrizione generata automaticamenteImmagine che contiene testo, ricevuta, Carattere, bianco

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, ricevuta, Carattere, bianco

Descrizione generata automaticamente

Per dimostrare che stutter(L) è un linguaggio regolare se L è regolare, possiamo procedere nel seguente modo:

1. Dato che L è un linguaggio regolare, esiste un DFA A = (Q, Σ, δ, q0, F) che lo riconosce.
2. Costruiamo un nuovo DFA A' = (Q', Σ, δ', q0', F') che riconosce stutter(L): Q' = Q × {0,1} (gli stati di A' sono coppie formate da uno stato di A e un bit) Σ = {0,1} q0' = (q0, 0) F' = {(q,1) | q ∈ F} La funzione di transizione δ' è definita come: δ'((q,0), a) = (δ(q,a), 1) δ'((q,1), a) = (δ(q,a), 0)
3. L'idea è che il bit aggiuntivo tiene traccia se stiamo leggendo il primo o il secondo simbolo di una coppia "stutterata".
4. Quando il bit è 0, stiamo leggendo il primo simbolo di una coppia, quindi passiamo allo stato corrispondente in A e settiamo il bit a 1.
5. Quando il bit è 1, stiamo leggendo il secondo simbolo di una coppia, quindi passiamo allo stato corrispondente in A e resettiamo il bit a 0.
6. L'automa accetta solo quando si trova in uno stato finale di A e ha appena letto il secondo simbolo di una coppia (bit = 1).

Questo DFA A' riconosce esattamente le parole nella forma aa.stutter(x) dove ax ∈ L, che corrisponde alla definizione di stutter(L).

Poiché abbiamo costruito un DFA che riconosce stutter(L), abbiamo dimostrato che stutter(L) è un linguaggio regolare.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Per dimostrare che ROR(L) è regolare se L è regolare, costruiamo un automa a stati finiti deterministico (DFA) che riconosce ROR(L) a partire dal DFA che riconosce L.

Sia A = (Q, Σ, δ, q0, F) il DFA che riconosce L. Costruiamo A' = (Q', Σ, δ', q0', F') che riconosce ROR(L) come segue:

Q' = Q × Σ ∪ {q0'} q0' è un nuovo stato iniziale F' = {(q, a) | δ(q, a) ∈ F} δ' è definita come: δ'(q0', a) = (q0, a) per ogni a ∈ Σ δ'((q, b), a) = (δ(q, a), b) per ogni q ∈ Q, a,b ∈ Σ

L'idea è che A' simula A tenendo traccia nell'ultimo simbolo letto nello stato. Quando A' raggiunge uno stato finale di A, accetta se l'ultimo simbolo corrisponde a quello memorizzato nello stato.

Poiché A' è un DFA finito, ROR(L) è regolare.

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Sia A = (Q, Σ, δ, q0, F) il DFA che riconosce L. Costruiamo A' = (Q', Σ, δ', q0', F') che riconosce ROL(L) come segue:

Q' = Q × Σ ∪ {q0'} q0' è un nuovo stato iniziale F' = {(q, a) | q ∈ F, a ∈ Σ} δ' è definita come:

* δ'(q0', a) = (q0, a) per ogni a ∈ Σ
* δ'((q, b), a) = (δ(q, a), b) per ogni q ∈ Q, a,b ∈ Σ

L'idea è che A' simula A, ma "ritarda" la lettura del primo simbolo, memorizzandolo nello stato. Quando A' raggiunge uno stato finale di A, accetta se l'ultimo simbolo letto porta A in uno stato finale.

A' accetta wa se e solo se aw ∈ L. Quindi, L(A') = ROL(L).

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamentePoiché A' è un DFA finito, ROL(L) è regolare.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamenteImmagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteImmagine che contiene testo, Carattere, schermata, lettera

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, numero

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, Carattere, schermata, numero

Descrizione generata automaticamente

Dimostrazione: Sia A = (Q, Σ, δ, q0, F) un automa a stati finiti deterministico (DFA) che riconosce L. Costruiamo un nuovo DFA A' = (Q, Γ\*, δ', q0, F) che riconosce T(L) come segue:

* Gli stati Q e lo stato iniziale q0 rimangono gli stessi
* L'insieme degli stati finali F rimane lo stesso
* Per ogni transizione δ(q, a) = p in A, definiamo in A' la transizione δ'(q, T(a)) = p

A' simula A, ma invece di leggere un simbolo a ∈ Σ, legge la sua traslitterazione T(a).

Per dimostrare che A' riconosce T(L), mostriamo che per ogni w ∈ Σ\*: w ∈ L se e solo se T(w) ∈ T(L)

(⇒) Se w = a1a2...an ∈ L, allora esiste una sequenza di stati q0, q1, ..., qn in A tale che: δ(qi-1, ai) = qi per i = 1, ..., n, e qn ∈ F

Per costruzione di A', avremo: δ'(qi-1, T(ai)) = qi per i = 1, ..., n

Quindi, T(w) = T(a1)T(a2)...T(an) è accettato da A', e T(w) ∈ T(L)

(⇐) Se T(w) ∈ T(L), allora T(w) è accettato da A'. Seguendo il ragionamento inverso, w deve essere accettato da A, quindi w ∈ L.

Poiché A' è un DFA, T(L) è regolare.

Immagine che contiene testo, schermata, Carattere, informazione

Descrizione generata automaticamente

Dimostrazione: Poiché L ed M sono regolari, esistono DFA A\_L = (Q\_L, {0,1}, δ\_L, q\_0L, F\_L) e A\_M = (Q\_M, {0,1}, δ\_M, q\_0M, F\_M) che li riconoscono rispettivamente.

Costruiamo un NFA N = (Q, {0,1}, δ, q\_0, F) che riconosce L ⊔ M:

* Q = Q\_L × Q\_M × {0, 1, 2}
* q\_0 = (q\_0L, q\_0M, 0)
* F = {(q\_L, q\_M, 2) | q\_L ∈ F\_L, q\_M ∈ F\_M}
* δ è definita come segue:
  1. δ((q\_L, q\_M, 0), 0) = {(δ\_L(q\_L, 0), q\_M, 0), (q\_L, δ\_M(q\_M, 1), 1)}
  2. δ((q\_L, q\_M, 0), 1) = {(δ\_L(q\_L, 1), q\_M, 0), (q\_L, δ\_M(q\_M, 0), 1)}
  3. δ((q\_L, q\_M, 1), 0) = {(q\_L, δ\_M(q\_M, 1), 1)}
  4. δ((q\_L, q\_M, 1), 1) = {(q\_L, δ\_M(q\_M, 0), 1)}
  5. δ((q\_L, q\_M, 1), ε) = {(q\_L, q\_M, 2)} se |x| = |y|

N simula A\_L e A\_M in parallelo, usando il terzo componente dello stato per tenere traccia della fase: 0 per leggere x, 1 per leggere y, e 2 per accettare se le lunghezze sono uguali.

Poiché N è un NFA e riconosce L ⊔ M, L ⊔ M è regolare.

Immagine che contiene testo, Carattere, schermata, informazione

Descrizione generata automaticamente

Dimostrazione: Poiché L ed M sono regolari, esistono DFA A\_L = (Q\_L, {0,1}, δ\_L, q\_0L, F\_L) e A\_M = (Q\_M, {0,1}, δ\_M, q\_0M, F\_M) che li riconoscono rispettivamente.

Costruiamo un NFA N = (Q, {0,1}, δ, q\_0, F) che riconosce L ⊕ M:

* Q = Q\_L × Q\_M × {0,1}
* q\_0 = (q\_0L, q\_0M, 0)
* F = {(q\_L, q\_M, i) | q\_L ∈ F\_L, q\_M ∈ F\_M, i ∈ {0,1}}
* δ è definita come segue:
  1. δ((q\_L, q\_M, 0), 0) = {(δ\_L(q\_L, 0), δ\_M(q\_M, 0), 0), (δ\_L(q\_L, 0), δ\_M(q\_M, 1), 1)}
  2. δ((q\_L, q\_M, 0), 1) = {(δ\_L(q\_L, 1), δ\_M(q\_M, 1), 0), (δ\_L(q\_L, 1), δ\_M(q\_M, 0), 1)}
  3. δ((q\_L, q\_M, 1), 0) = {(δ\_L(q\_L, 0), δ\_M(q\_M, 1), 1), (δ\_L(q\_L, 0), δ\_M(q\_M, 0), 0)}
  4. δ((q\_L, q\_M, 1), 1) = {(δ\_L(q\_L, 1), δ\_M(q\_M, 0), 1), (δ\_L(q\_L, 1), δ\_M(q\_M, 1), 0)}

N simula A\_L e A\_M in parallelo, usando il terzo componente dello stato per tenere traccia del risultato dello XOR bit a bit. Poiché N è un NFA e riconosce L ⊕ M, L ⊕ M è regolare.

Immagine che contiene testo, schermata, Carattere, algebra

Descrizione generata automaticamente

Dimostrazione: Sia A = (Q, Σ, δ, q0, F) un DFA che riconosce L. Costruiamo un NFA A' = (Q', Σ, δ', q0', F') che riconosce SWAP(L):

Q' = Q × {0, 1, 2} q0' = (q0, 0) F' = {(q, 2) | q ∈ F} δ' è definita come segue:

1. δ'((q, 0), ε) = {(q, 2)} (per gestire le stringhe vuote o di lunghezza 1)
2. δ'((q, 0), a) = {(δ(q, a), 1), (δ(q, a), 2)} per ogni a ∈ Σ
3. δ'((q, 1), a) = {(δ(q, a), 2)} per ogni a ∈ Σ
4. δ'((q, 2), a) = {(δ(q, a), 0)} per ogni a ∈ Σ

L'idea è che lo stato 0 rappresenta l'inizio della stringa o dopo uno scambio, lo stato 1 rappresenta dopo aver letto il primo carattere di una coppia, e lo stato 2 rappresenta dopo aver completato uno scambio o alla fine della stringa.

Poiché A' è un NFA che riconosce SWAP(L), SWAP(L) è regolare