Lezione 14 – da Macchina di Turing a Grammatica

Lezione del 03/04/2025

ESEMPIO 4

- Riprendiamo l'esempio introdotto la lezione precedente:
- Si chiede di progettare una grammatica G che generi il linguaggio
  $L = \{ xx : x \in \{a,b\}^+ \}$
- La grammatica $G = \langle V_T, V_N, P, S \rangle$ che serve allo scopo è la seguente:
  - $V_T = \{ a,b \}$,
  - $V_N = \{S, U_a, U_b, X\}$
  - P contiene le produzioni seguenti
    - 1) $S \to a\, U_a\, S \mid b\, U_b\, S \mid X$
    - 2) $U_a\, a \to a\, U_a$, $U_a\, b \to b\, U_a$
    - 3) $U_b\, a \to a\, U_b$, $U_b\, b \to b\, U_b$
    - 4) $U_a\, X \to Xa$, $U_b\, X \to Xb$
    - 5) $U_a\, X \to a$, $U_b\, X \to b$
- Abbiamo osservato che G è una grammatica di tipo 0

ESEMPIO 4

- Mostriamo ora che L(G) = { xx : x ∈ $\{a,b\}^+$}
  - abbiamo dimostrato la lezione precedente che 1) { xx : x ∈ $\{a,b\}^+$} ⊆ L(G)
- 2) mostriamo informalmente che L(G) ⊆ { xx : x ∈ $\{a,b\}^+$}, ossia che ogni parola non vuota generabile mediante G è una parola nella forma xx, per qualche parola x nell'alfabeto {a,b}:
  - osserviamo che il non terminale S può essere eliminato dalle parole in $(V_T \cup V_N)^*$ generate da G a partire da S solo quando viene utilizzata la produzione S → X
  - e osserviamo anche che una volta utilizzata la produzione S → X non possono più essere aggiunti caratteri a o b a sinistra di X
  - inoltre, se la produzione S → X è la prima ad essere utilizzata allora nessuna parola in $\{a,b\}^+$ può essere generata
  - supponiamo allora di aver utilizzato per n volte le produzioni S → a $\mathrm{U_a}$ S | b $\mathrm{U_b}$ S così da aver generato $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_n\mathrm{U}_{x_n}$ S e poi $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_n\mathrm{U}_{x_n}$ X
    - (dove U_{x_i}=0 se x_i =0, U_{x_i}=1 se x_i =1)

ESEMPIO 4

- Mostriamo ora che L(G) = { xx : x ∈ {a,b}$^+$}
- 2) mostriamo informalmente che L(G) ⊆ { xx : x ∈ {a,b}$^+$}, ossia che ogni parola non vuota generabile mediante G è una parola nella forma xx, per qualche parola x nell'alfabeto {a,b}:
  - supponiamo allora di aver utilizzato per n volte le produzioni $S \rightarrow a\ \mathrm{U_a}\ S \mid b\ \mathrm{U_b}\ S$ così da aver generato $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_n\mathrm{U}_{x_n}$ S e poi $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_n\mathrm{U}_{x_n}$ X
  - a questo punto, se applichiamo la produzione U_{x_n} X → x_n otteniamo la parola $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_{n-1}\mathrm{U}_{x_{n-1}}x_n\ x_n$ dalla quale non riusciamo ad eliminare i caratteri non terminali
  - pertanto, prima di applicare una produzione che rimuova X dalla parola generata dobbiamo eliminare dalla parola generata tutti i non terminali $\mathrm{U_a}$ e $\mathrm{U_b}$
  - a questo scopo, o applichiamo la produzione U_{x_n}X → X x_n così da ottenere la parola $x_1\mathrm{U}_{x_1}\ x_2\mathrm{U}_{x_2} \ldots x_{n-1}\mathrm{U}_{x_{n-1}}\ x_n$ X x_n oppure applichiamo una delle produzioni nella forma $\mathrm{U_i}$h → h$\mathrm{U_i}$ (con i,h ∈ {a,b}*) per ''avvicinare'' il non terminale $\mathrm{U_i}$ al non terminale X

ESEMPIO 4

- Mostriamo ora che L(G) = { xx : x ∈ {a,b}$^+$}
- 2) mostriamo informalmente che L(G) ⊆ { xx : x ∈ {a,b}$^+$}, ossia che ogni parola non vuotagenerabile mediante G è una parola nella forma xx, per qualche parola x nell'alfabeto {a,b}:
  - prima di applicare una produzione che rimuova X dalla parola generata dobbiamo eliminare dalla parola generata tutti i non terminali $\mathrm{U_a}$ e $\mathrm{U_b}$
  - a questo scopo, o applichiamo la produzione U_{x_n}X → X x_n così da ottenere la parola $x_1\mathrm{U}_{x_1}$ $x_2\mathrm{U}_{x_2}$ … $x_{n-1}\mathrm{U}_{x_{n-1}}$ x_n X x_n oppure applichiamo una delle produzioni nella forma $\mathrm{U_i}$h → h$\mathrm{U_i}$ (con i,h ∈ {a,b}*) per ''avvicinare'' il non terminale $\mathrm{U_i}$ al non terminale X
  - in ogni caso, possiamo eliminare un terminale $\mathrm{U_i}$ dalla parola solo quando esso è il carattere immediatamente a destra di X, così da usare la produzione $\mathrm{U_i}$X → Xi
  - ma il primo non terminale $\mathrm{U_i}$ a trovarsi immediatamente a destra di X è U_{x_n} così che la parola generata terminerà con X x_n
  - poi, il secondo non terminale $\mathrm{U_i}$ a trovarsi immediatamente a destra di X è $\mathrm{U}_{x_{n-1}}$, così che la parola generata terminerà con X$x_{n-1}x_n$
  - …

ESEMPIO 4

- Mostriamo ora che L(G) = { xx : x ∈ {a,b}$^+$}
- 2) mostriamo informalmente che L(G) ⊆ { xx : x ∈ {a,b}$^+$}, ossia che ogni parola non vuota generabile mediante G è una parola nella forma xx, per qualche parola x nell'alfabeto {a,b}:
  - prima di applicare una produzione che rimuova X dalla parola generata dobbiamo eliminare dalla parola generata tutti i non terminali $\mathrm{U_a}$ e $\mathrm{U_b}$ tranne uno applicando una delle produzioni $\mathrm{U_a}$X → X a oppure $\mathrm{U_a}$X → X a
  - in ogni caso, possiamo eliminare un terminale $\mathrm{U_i}$ dalla parola solo quando esso è il carattere immediatamente a destra di X, così da usare la produzione $\mathrm{U_i}$X → Xi
  - ma il primo non terminale $\mathrm{U_i}$ a trovarsi immediatamente a destra di X è U_{x_n} , il secondo è $\mathrm{U}_{x_{n-1}}$, e così via fino a U_{x_2}, così da ottenere una parola che termina con X x_2 … $x_{n-1}x_n$
  - ma, avendo eliminato come descritto tutti i non terminali $\mathrm{U_a}$ e $\mathrm{U_b}$ tranne U_{x_1} dalla parola $x_1\mathrm{U}_{x_1}$ $x_2\mathrm{U}_{x_2}$ … $x_{n-1}\mathrm{U}_{x_{n-1}}$ x_n X , la parola ottenuta al termine di questo procedimento è x_1x_2 … $x_{n-1}x_n\mathrm{U}_{x_1}$X x_2 … $x_{n-1}x_n$
  - da cui, applicando la produzione U_{x_1}X → x_1, otteniamo infine la parola in L(G) x_1x_2 … $x_{n-1}x_n$ x_1x_2 … $x_{n-1}x_n$

Grammatiche e macchine di Turing

- La Grammatica è un modello di calcolo
  - una grammatica descrive come si possono generare le parole appartenenti a un insieme di parole
- La Macchina di Turing è un modello di calcolo
  - una macchina di Turing descrive come si fa a riconoscere le parole appartenenti a un insieme di parole
- Mostriamo ora che i due modelli di calcolo sono equivalenti, ossia che
  - se un linguaggio è accettato da una macchina di Turing allora esiste una grammatica di tipo 0 che lo genera
  - se un linguaggio è generato da una grammatica (di tipo 0) allora esiste una macchina di Turing che lo accetta
- In altre parole ciò che ci accingiamo a fare è dimostrare che
  le grammatiche di tipo 0 soddisfano la Tesi di Church-Turing

Grammatiche e macchine di Turing

- Dimostreremo, innanzi tutto, il seguente teorema:
- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica G (di tipo 0) tale che L = L(G)

- Prima di farlo, abbiamo però bisogno di un risultato ancora relativo all'equivalenza di modelli di macchine di Turing che non abbiamo visto quando ci siamo occupati di equivalenza di modelli di macchine di Turing:

- <u>*TEOREMA*</u>: *per ogni macchina di Turing T a un nastro e con alfabeto {0,1} esiste una macchina di Turing T' con un solo nastro semi-infinito e che non scrive mai il blank ▱ tale che, per ogni x ∈ {0,1}* ,* $o_T(x)=o_{T'}(x)$
  - dove un nastro semi-infinito ha una cella iniziale e tutte le altre celle sono a destra della cella iniziale
- La dimostrazione del precedente teorema viene lasciata per esercizio

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica G (di tipo 0) tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile
  - sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing che accetta L
    - possiamo supporre che T sia una macchina ad un solo nastro e che esso sia semi-infinito
    - e che T non scriva mai il blank $\square$
- e sia $Q = \{q_0, q_1, \dots, q_k\ q_A, q_R\}$
- Definiamo una grammatica $G = \langle V_T, V_N, P_G, S \rangle$, con
  - $V_T = \{0, 1, a, \square\}$,
  - $V_N = \{S, A, C, D, X, U_0, U_1\} \cup \{Q_i : i = 0, \dots, k\} \cup \{Q_A, Q_R\}$
- che costruisce una parola x in tre fasi
  - prima costruisce la parola x a Q_0 x $\square$ con $x \in \{0,1\}^*$
  - poi "simula" il comportamento di T con input x e
  - se la computazione T(x) termina in q_A allora i caratteri a destra di 'a' e il carattere 'a' vengono cancellati lasciando la sola x sul nastro

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia L ⊆ {0,1}* un linguaggio accettabile e sia T = ⟨ {0,1}, Q, P_T, q_0, $\{q_A, q_R\}$ ⟩ la macchina di Turing a nastro semi-infinito che accetta L, con Q = $\{q_0, q_1, \ldots, q_n, q_A, q_R\}$
- Produzioni in P_G della fase 1 (costruzione della parola xaQ_0x□ con x ∈{0,1}*):
  - coincidono sostanzialmente con le produzioni della grammatica dell'esempio 4:
    - 1') S → 0 U_0 A □ | 1 U_1 A □ | a Q_0 □,   A → 0 U_0 A | 1 U_1 A | X
    - 2') U_0 0 → 0 U_0, U_0 1 → 1 U_0
    - 3') U_1 0 → 0 U_1, U_1 1 → 1 U_1
    - 4') U_0 X → X0, U_1 X → X1
    - 5') U_0 X → a$Q_0$0, U_1 X → a $Q_0$1
- una dimostrazione simile a quella dell'ESEMPIO 4 permette ora di dimostrare che:
  OSS1: per ogni x ∈{0,1}*****, le produzioni della fase 1 permettono di generare la parola xaQ_0x□ (dunque, anche aQ_0□)

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- DIMOSTRAZIONE: sia L ⊆ {0,1}* un linguaggio accettabile e sia T = ⟨ {0,1}, Q, P_T, q_0, {q_A, q_R} ⟩ la macchina di Turing a nastro semi-infinito che accetta L con Q = $\{q_0, q_1, \ldots, q_k, q_A, q_R\}$
  - senza perdita di generalità abbiamo assunto che T non scriva il carattere '$\square$'
- Produzioni in P_G della fase 2 (simulazione di T(x)):
  - per ogni quintupla ⟨ q_{i_1}, h_1, h_2, q_{i_2}, sx ⟩ in P_T, con h_1 ∈ {0,1}, h_2 ∈ {0,1} e q_{i_1}, q_{i_2} ∈ Q, P_G contiene le produzioni b $Q_{i_1}h_1 \rightarrow Q_{i_2}bh_2$ per ogni b ∈ {0,1, $\square$}
  - per ogni quintupla ⟨ q_{i_1}, $\square$, h_2, q_{i_2}, sx ⟩ in P_T, con h_2 ∈ {0,1} e q_{i_1}, q_{i_2} ∈ Q, P_G contiene le produzioni b $Q_{i_1}h_1 \rightarrow Q_{i_2}bh_2\square$ per ogni b ∈ {0,1}
  - [... quintuple con testina che resta ferma ...]
  - [... quintuple con testina che muove a destra]

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che accetta L con $Q = \{q_0, q_1, \dots, q_k, q_A, q_R\}$
  - senza perdita di generalità assumiamo che T non scriva il carattere '□'
- Produzioni in P_G della fase 2 (simulazione di T(x)):
  - [quintuple con testina che muove a sinistra ...]
  - per ogni quintupla $\langle q_{i_1}, h_1, h_2, q_{i_2}, f \rangle$ in P_T, con $h_1 \in \{0,1\}$, $h_2 \in \{0,1\}$ e $q_{i_1}, q_{i_2} \in Q$, P_G contiene la produzione $Q_{i_1} h_1 \rightarrow Q_{i_2} h_2$
  - per ogni quintupla $\langle q_{i_1}, □, h_2, q_{i_2}, f \rangle$ in P_T, con $h_2 \in \{0,1\}$ e $q_{i_1}, q_{i_2} \in Q$, P_G contiene la produzione $Q_{i_1} h_1 \rightarrow Q_{i_2} h_2$ □
  - [... quintuple con testina che muove a destra]

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- DIMOSTRAZIONE: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che accetta L con $Q = \{q_0, q_1, \dots, q_k, q_A, q_R\}$
  - senza perdita di generalità assumiamo che T non scriva il carattere '□'
- Produzioni in P_G della fase 2 (simulazione di T(x)):
  - [quintuple con testina che muove a sinistra ...]
  - [... quintuple con testina che rimane ferma ...]
  - per ogni quintupla $\langle q_{i_1}, h_1, h_2, q_{i_2}, dx \rangle$ in P_T, con $h_1 \in \{0,1\}$, $h_2 \in \{0,1\}$ e $q_{i_1}, q_{i_2} \in Q$, PG contiene la produzione $Q_{i_1} h_1 \rightarrow h_2 Q_{i_2}$
  - per ogni quintupla $\langle q_{i_1}, □, h_2, q_{i_2}, dx \rangle$ in P_T, con $h_2 \in \{0,1\}$ e $q_{i_1}, q_{i_2} \in Q$, PG contiene la produzione $Q_{i_1} □ \rightarrow h_2 Q_{i_2} □$

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- DIMOSTRAZIONE: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che non scriva il carattere '□' e che accetta L, con $Q = \{q_0, q_1, \dots, q_k, q_A, q_R\}$
- Produzioni in P_G della fase 2 (simulazione di T(x)): osservazioni
  - **OSS2**: le produzioni della fase 2 possono essere applicate solo a parole contenenti il non terminale Q_0 (dunque, solo dopo che il non terminale X è stato rimosso dalla parola generata e le produzioni della fase 1 non possono più essere applicate)
  - **OSS3**: ogni parola generata durante la fase 2 contiene uno e un solo non terminale appartenente all'insieme $\{Q_i : i = 0, \dots, k\} \cup \{Q_A, Q_R\}$
  - **OSS4**: poiché la porzione delle parole generate durante la fase 2 a destra del carattere 'a' rappresentano gli stati globali della computazione T(x), allora verrà generata una parola contenente il non terminale Q_A se e soltanto se la computazione T(x) termina nello stato di accettazione q_A

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- DIMOSTRAZIONE: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che non scriva il carattere '□' e che accetta L, con $Q = \{q_0, q_1, \dots, q_k, q_A, q_R\}$
- Produzioni in P_G della fase 2 (simulazione di T(x)): osservazioni
  - **OSS5**: inoltre, se la computazione T(x) non termina nello stato di accettazione q_A allora qualunque parola generata durante la fase 2 contiene un non terminale (appartenente all'insieme $\{Q_i : i = 0, \dots, k\} \cup \{Q_R\}$)
  - **OSS6**: qualunque parola generata durante la fase 2 ha come ultimo carattere il carattere □

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia L ⊆ {0,1}* un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che accetta L con $Q = \{ q_0, q_1, \ldots, q_k, q_A, q_R\}$
- Produzioni in P_G della fase 3 (cancellazione dei caratteri a destra di 'a'):
  - per ogni b ∈ {0,1}, P_G contiene la produzione $Q_A b \rightarrow b\, Q_A$
  - P_G contiene la produzione $Q_A \square \rightarrow C$
  - per ogni b ∈ {0,1}, P_G contiene la produzione $bC \rightarrow C$
  - P_G contiene la produzione $aC \rightarrow \varepsilon$
- Osservazioni relative alla fase 3
  - **OSS7**: le produzioni della fase 3 possono essere applicate solo a parole contenenti il non terminale Q_A
  - **OSS8**: è semplice verificare che, applicando le produzioni della fase 3 a una parola che contiene Q_A come unico carattere non terminale e che termina con il carattere $\square$, viene generata una parola costituita dai soli caratteri a sinistra del carattere 'a' più a destra

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia L ⊆ {0,1}* un linguaggio accettabile e sia T = ⟨ {0,1}, Q, P_T, q_0, {q_A, q_R} ⟩ la macchina di Turing a nastro semi-infinito che accetta L con Q = { q_0, q_1, …, q_n, q_A, q_R}
- Conclusioni: per ogni x ∈ L
  - in base all'**OSS1**, le produzioni della fase 1 permettono di generare la parola x a Q_0 x $\square$, ossia S $\Rightarrow_G^*$ x a Q_0 x $\square$
  - in base all'**OSS2**, le produzioni della fase 2 possono essere applicate a x a Q_0 x $\square$
  - sia xay la parola generata applicando le produzioni della fase 2 fino a quando non è più possibile applicarle (così che x a Q_0 x $\square$ $\Rightarrow_G^*$ xay) con y ∈ $(V_T \cup V_N)^*$
  - in base all'**OSS4**, poiché x ∈ L e T accetta L, y contiene il non terminale Q_A
  - in base all'**OSS7**, le produzioni della fase 3 possono essere applicate a xay
  - in base alle **OSS6** e **OSS8**, xay $\Rightarrow_G^*$ x
  - quindi, S $\Rightarrow_G^*$ x, ossia x ∈ L(G)

Grammatiche e macchine di Turing

- **TEOREMA G.4**: per ogni linguaggio accettabile L esiste una grammatica (di tipo 0) G tale che L = L(G)
- <u>DIMOSTRAZIONE</u>: sia $L \subseteq \{0,1\}^*$ un linguaggio accettabile e sia $T = \langle \{0,1\}, Q, P_T, q_0, \{q_A, q_R\} \rangle$ la macchina di Turing a nastro semi-infinito che accetta L con $Q = \{ q_0, q_1, \ldots, q_n, q_A, q_R\}$
- Conclusioni: per ogni $x \in L(G)$
  - se $x \in L(G)$ allora $S \Rightarrow_G^* x$
  - è possibile dimostrare che le sole parole derivate da S utilizzando solo produzioni della fase 1 e che contengono x e Q_0 sono nella forma $xaQ_0x\square$ con $x \in \{0,1\}^*$
  - in base all'**OSS2**, le produzioni della fase 2 non possono essere utilizzate fino a quando non viene generata la parola $xaQ_0x\square$ con $x \in \{0,1\}^*$
  - in base anche all'**OSS5**, tutte le parole generate utilizzando produzioni delle fasi 1 e 2 contengono almeno un carattere non terminale e quindi è necessario utilizzare le produzioni della fase 3 per ottenere una parola di L(G)
  - in base all'**OSS7**, per applicare le produzioni della fase 3 è necessario che la parola generata utilizzando le produzioni delle fasi 1 e 2 contenga il non terminale Q_A e, in base all'**OSS4**, questo avviene se e soltanto se x è accettato da T
  - questo dimostra che $x \in L$ QED