CATEGORIA: DIMOSTRA L DECIDIBILE

=

DIMOSTRA CHE ESISTE UNA MDT PER IL PROBLEMA

Il problema dell'accettazione

UNIVERSITÀ DEGLI STUDI DI PADOVA

UNICO LINGUAGGIO = AUTOMA A STATI FINITI DETERMINISTICO

- **Problema dell'accettazione:** testare se un DFA accetta una stringa
- $A_{DFA} = \{\langle B, w\rangle \mid B \text{ è un DFA che accetta la stringa } w\}$
- B accetta w se e solo se $\langle B, w\rangle$ appartiene ad A_{DFA}
- Mostrare che il linguaggio è decidibile equivale a mostrare che il problema computazionale è decidibile

Il problema dell'accettazione

UNIVERSITÀ DEGLI STUDI DI PADOVA

- **Problema dell'accettazione:** testare se un DFA accetta una stringa
- $A_{DFA} = \{\langle B, w\rangle \mid B \text{ è un DFA che accetta la stringa } w\}$
- B accetta w se e solo se $\langle B, w\rangle$ appartiene ad A_{DFA}
- Mostrare che il linguaggio è decidibile equivale a mostrare che il problema computazionale è decidibile

Teorema: A_{DFA} è decidibile

Idea: definire una TM che decide A_{DFA}

M = "Su input $\langle B, w\rangle$, dove B è un DFA e w una stringa:

1. Simula B su input w
2. Se la simulazione termina in uno stato finale, accetta. Se termina in uno stato non finale, rifiuta."

Dimostrazione:

- la codifica di B è una lista delle componenti Q, Σ, δ, q_0 e F
- fare la simulazione è facile

1° [DFA / NFA / REGEX]

Teorema: A_{NFA} è decidibile

$A_{NFA} = \{\langle B, w\rangle \mid B \text{ è un } \varepsilon\text{-NFA che accetta la stringa } w\}$

Idea: usiamo la TM M che decide A_{DFA} come subroutine

Dimostrazione:

N = "Su input $\langle B, w\rangle$, dove B è un ε-NFA e w una stringa:

1. Trasforma B in un DFA equivalente C usando la costruzione per sottoinsiemi
2. Esegui M con input $\langle C, w\rangle$
3. Se M accetta, accetta; altrimenti, rifiuta."

N è un decisore per A_{NFA}, quindi A_{NFA} è decidibile

Teorema: A_{REX} è decidibile

AD ALTO LIVELLO

$A_{REX} = \{\langle R, w\rangle \mid R \text{ è una espressione regolare che genera la stringa } w\}$

Idea: usiamo la TM N che decide A_{NFA} come subroutine

Dimostrazione:

P = "Su input $\langle R, w\rangle$, dove R è una espressione regolare e w una stringa:

1. Trasforma R in un ε-NFA equivalente C usando la procedura di conversione
2. Esegui N con input $\langle C, w\rangle$
3. Se N accetta, accetta; altrimenti, rifiuta."

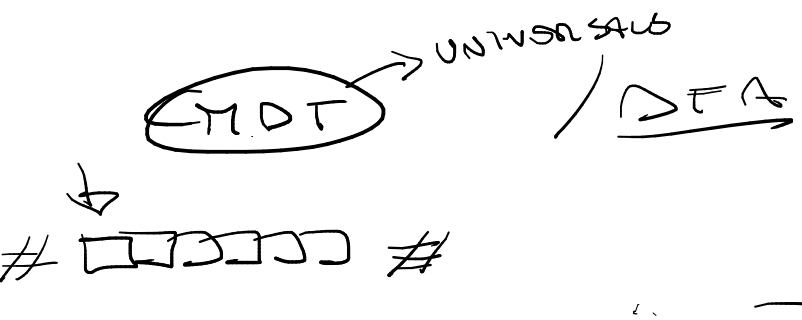
P è un decisore per A_{REX}, quindi A_{REX} è decidibile

MDT DECIDE L'E DECISORE = N

Altri:

$$EQ_{DFA} = \{\langle A, B\rangle \mid A \text{ e } B \text{ sono DFA e } L(A) = L(B)\}$$

$$A_{CFG} = \{\langle G, w\rangle \mid G \text{ è una CFG che genera la stringa } w\}$$



Consegna: "Verifica che 10 + 20 = 30."

1. Esprimi la consegna come Linguaggio L.
2. Dimostra che L sia decidibile → DFA INFA LGR → TERMINANO

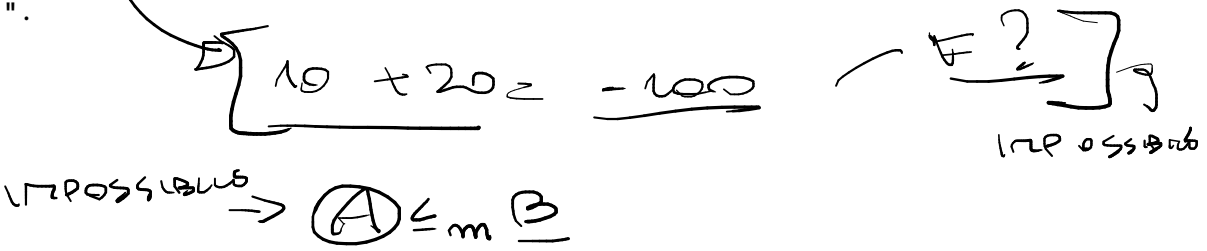
DESCRIVO UN PROBLEMA

Soluzione:

1. L = { <A, w> | A è un DFA, w è il mio input}

Osservazioni:
- 10 + 20 = 30 diventa la condizione di accettazione del DFA (w diventa uguale a 30)
  - x + y = 30 → Se SI, vai alla fine, Se NO, vai in loop o rifiuta
- In questo modo, la MdT "M" che risolve il problema SOLO SE accetta il DFA

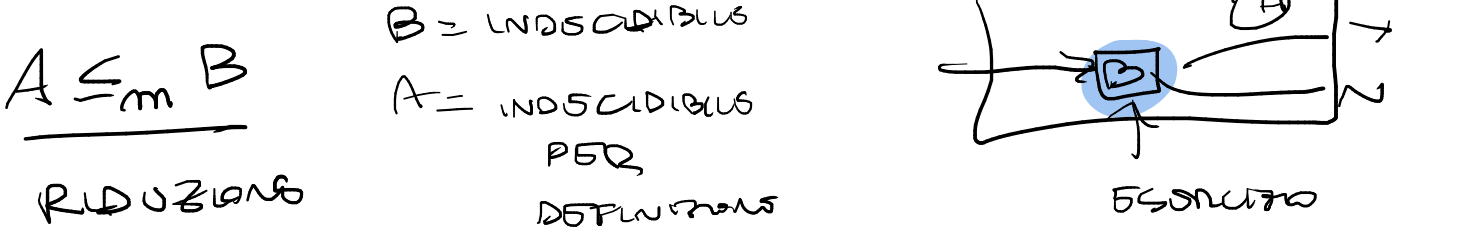
2. Il punto (2) ti chiede "esiste una MdT che esprime il tuo problema sia che vada bene sia che vada male?".



Dimostrazione:

M = "Su input <A, w>, dove "A" è un DFA e "w" è una stringa:

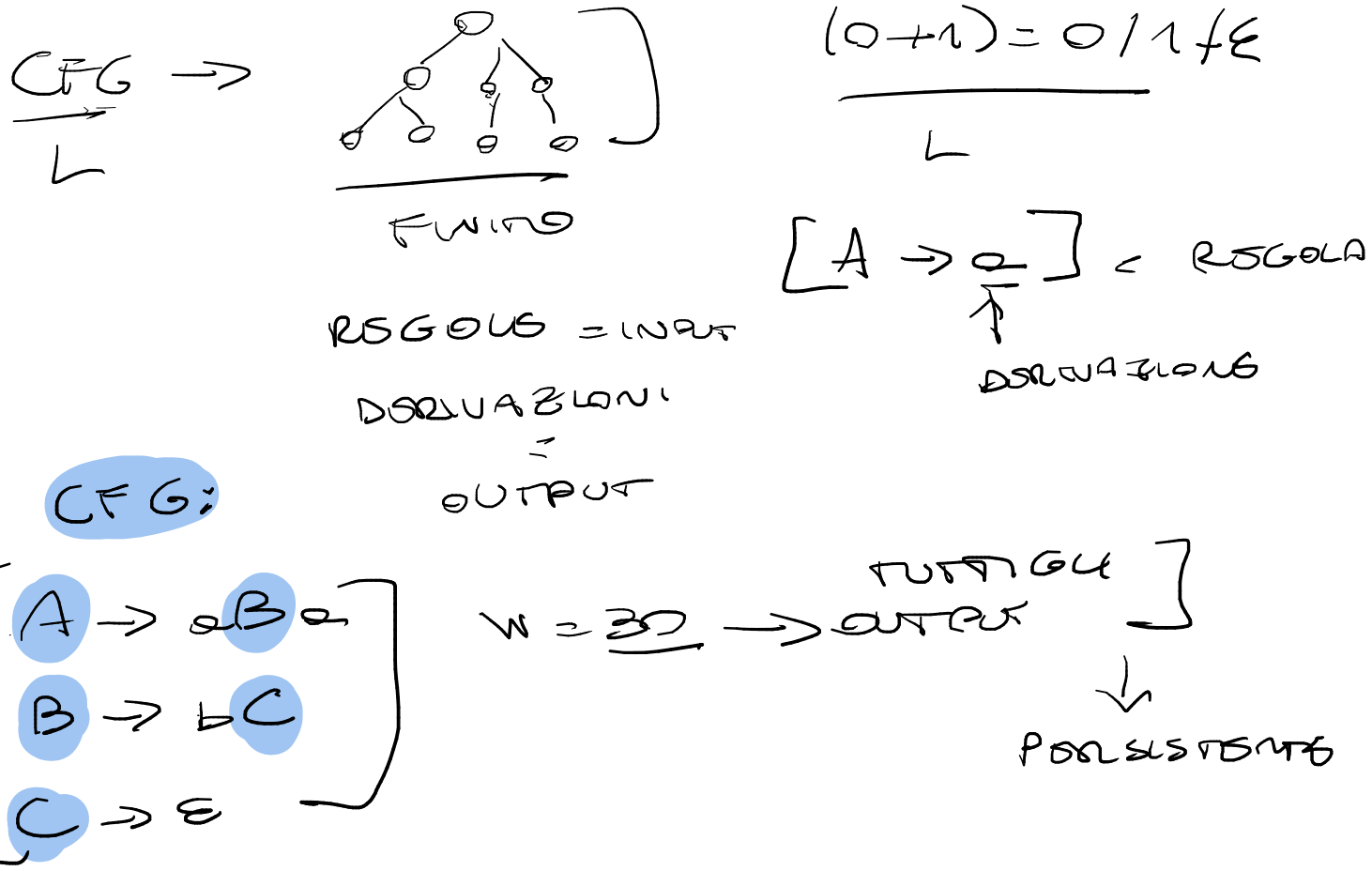
- Simula A su w ed esegui ⟶ fai la calcolatrice avendo il problema
- Se A termina, allora M accetta, altrimenti rifiuta → Ti do output si SE E SOLO trovo 30



2. (12 punti) Una variabile A in una grammatica context-free G è *persistente* se compare in ogni derivazione di ogni stringa w in $L(G)$. Data una grammatica context-free G e una variabile A, considera il problema di verificare se A è persistente.

(a) Formula questo problema come un linguaggio $PERSISTENT_{\text{CFG}}$.

(b) Dimostra che $PERSISTENT_{\text{CFG}}$ è decidibile.



2. (12 punti) Una variabile A in una grammatica context-free G è *persistente* se compare in ogni derivazione di ogni stringa w in $L(G)$. Data una grammatica context-free G e una variabile A, considera il problema di verificare se A è persistente.

(a) Formula questo problema come un linguaggio $PERSISTENT_{\text{CFG}}$.

(b) Dimostra che $PERSISTENT_{\text{CFG}}$ è decidibile.

(a) $PERSISTENT_{\text{CFG}} = \{\langle G, A\rangle \mid G \text{ è una CFG, } A \text{ è una variabile persistente}\}$

(b) La seguente macchina N usa la Turing machine M che decide E_{CFG} per decidere $PERSISTENT_{\text{CFG}}$

N = "su input $\langle G, A\rangle$, dove G è una CFG e A una variabile:

1. Verifica che A appartenga alle variabili di G. In caso negativo, rifiuta.
2. Costruisci una CFG G' eliminando tutte le regole dove compare A dalla grammatica G.
3. Esegui M su input $\langle G'\rangle$, e ritorna lo stesso risultato di M."

Mostriamo che N è un decisore dimostrando che termina sempre e che ritorna il risultato corretto. Verificare che una variabile appartenga alle variabili di G è una operazione che si può implementare scorrendo la codifica di G per controllare se A compare nella codifica. Il secondo passo si può implementare copiando la codifica di G senza riportare le regole dove compare A. Di conseguenza, il primo ed il secondo step terminano sempre. Anche il terzo step termina sempre perché sappiamo che E_{CFG} è un linguaggio decidibile. Quindi N termina sempre la computazione.

Vediamo ora che N dà la risposta corretta:

- Se $\langle G, A \rangle \in PERSISTENT_{\text{CFG}}$ allora A è una variabile persistente, quindi compare in ogni derivazione di ogni stringa $w \in L(G)$. Se la eliminiamo dalla grammatica, eliminando tutte le regole dove compare A, allora otteniamo una grammatica G' dove non esistono derivazioni che permettano di derivare una stringa di soli simboli terminali, e di conseguenza G' ha linguaggio vuoto. Quindi $\langle G' \rangle \in E_{\text{CFG}}$, e l'esecuzione di M terminerà con accettazione. N ritorna lo stesso risultato di M, quindi accetta.
- Viceversa, se $\langle G, A \rangle \in PERSISTENT_{\text{CFG}}$ allora A non è una variabile persistente, quindi esiste almeno una derivazione di una parola $w \in L(G)$ dove A non compare. Se eliminiamo A dalla grammatica, eliminando tutte le regole dove compare, allora otteniamo una grammatica G' che può derivare w, e di conseguenza G' ha linguaggio vuoto. Quindi $\langle G' \rangle \notin E_{\text{CFG}}$, e l'esecuzione di M terminerà con rifiuto. N ritorna lo stesso risultato di M, quindi rifiuta.