1. **(9 punti)** Considera il linguaggio

$$L = \{0^m 1^n \mid 2m > 3n + 1\}.$$

Dimostra che L non è regolare.

$\{PL = xy^iz, i \geq 0, y \neq \varepsilon\}$

$xy^0z = \nearrow z = 0^{k-p-q}$

$xy^2z = \searrow 0^{k+p+q}$

$\{0^m 1^n \mid 2m > 3n+1\}$

$m = 1$,	$n = 0$
2	1
$m = 2$,	$n = 1$
4 ,	4
$m = 3$,	$n = 2$
6	> 7

$w = xy^iz = xy^0z$

$x = 0^p \quad y = 0^q \quad z = 0^{k-p-q} 1^n$

$xy^0z = 0^p 0^q 0^{k-p-q} 1^n$

$= 0^k 1^n$

$\{0^m 1^n \mid 2m > 3n+1\}$

$m = 1, n = 0$

$2 > 1$

$(xy \leq \cancel{k} \to m \cdot n)$

FUNZIONA GRAZIE AD ESEMPIO INDUTTIVO

L NON È REGOLARE

2. **(9 punti)** Considera la seguente funzione da $\{0,1\}^*$ a $\{0,1\}^*$:

FARLO → 2021

$$\text{stutter}(w) = \begin{cases} \varepsilon & \text{se } w = \varepsilon \\ aa.\text{stutter}(x) & \text{se } w = ax \text{ per qualche simbolo } a \text{ e parola } x \end{cases}$$

Dimostra che se L è un linguaggio context-free sull'alfabeto $\{0,1\}$, allora anche il seguente linguaggio è context-free:

$$\text{stutter}(L) = \{\text{stutter}(w) \mid w \in L\}.$$

$G = \text{STUTTER} \to G' = \underline{CF}$

2. (9 punti) Considera la seguente funzione da $\{0,1\}^*$ a $\{0,1\}^*$:

$$\text{stutter}(w) = \begin{cases} \varepsilon & \text{se } w = \varepsilon \\ aa.\text{stutter}(x) & \text{se } w = ax \text{ per qualche simbolo } a \text{ e parola } x \end{cases}$$ $\Rightarrow \exists G = CFG$

Dimostra che se L è un linguaggio context-free sull'alfabeto $\{0,1\}$, allora anche il seguente linguaggio è context-free:

$$\text{stutter}(L) = \{\text{stutter}(w) \mid w \in L\}.$$

CFG $\rightarrow$ CHOMSKY

$\exists G$, $\exists G'$ EQUIVALENTE

$G = (\Sigma, R, V, S)$

$\downarrow$

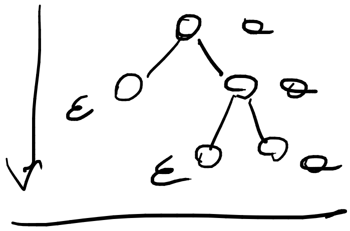
$G' = (\Sigma', R', V', S')$

$\Sigma = \Sigma'$

$S = S'$

$$\text{stutter}(w) = \begin{cases} \varepsilon & \text{se } w = \varepsilon \\ aa.\text{stutter}(x) & \text{se } w = ax \text{ per qualche simbolo } a \text{ e parola } x \end{cases}$$

0 ε



$\forall \ldots$ REGOLE

$\left\{ \begin{array}{l} S \to aA \\ A \to aB \\ B \to b \end{array} \right\} \Rightarrow$ CFG

$\left\{ \begin{array}{l} A \to BC \\ B \to b \end{array} \right\}$ CHOMSKY

DERIVAZIONE = A LBERO COME SI SVILUPPA

$(\cancel{S} \to aA \to aaB \to aab)$

$\forall$ VARIABILE V, RIMPIAZZO $\forall$ REGOLA CON VARIABILE V'

STUTTER $(W) = \begin{cases} \varepsilon, \quad w = \varepsilon \to \{S \to aA\} \\ aa.\text{STUTTER}(X) \to \{\ldots\ ax\} \end{cases}$

2. (8 punti) *Per ogni linguaggio L, sia* prefix$(L) = \{u \mid uv \in L$ *per qualche stringa* $v\}$. *Dimostra che se L è un linguaggio context-free, allora anche* prefix(L) *è un linguaggio context-free.*

Se L è un linguaggio context-free, allora esiste una grammatica G *in forma normale di Chomski* che lo genera. Possiamo costruire una grammatica G' che genera il linguaggio prefix(L) in questo modo:

- per ogni variabile V di G, G' contiene sia la variabile V che una nuova variabile V'. La variabile V' viene usata per generare i prefissi delle parole che sono generate da V;
- tutte le regole di G sono anche regole di G';
- per ogni variabile V di G, le regole $V' \to V$ e $V' \to \varepsilon$ appartengono a G;
- per ogni regola $V \to AB$ di G, le regole $V' \to AB'$ e $V' \to A'$ appartengono a G';
- se S è la variabile iniziale di G, allora S' è la variabile iniziale di G'.

$$\text{stutter}(w) = \begin{cases} \varepsilon & \text{se } w = \varepsilon \\ aa.\text{stutter}(x) & \text{se } w = ax \text{ per qualche simbolo } a \text{ e parola } x \end{cases}$$ → PDA

CFG → CHOMSKY (1) ✓

CFG → PDA (2)

(PUSH)

POP

a

O ~ O

memoria push

A (A) → A → STACK

O

```
stutter("abc")
= a.a.stutter("bc")                 [applicando la definizione]
= a.a.b.b.stutter("c")              [applicando ricorsivamente]
= a.a.b.b.c.c.stutter("ε")         [applicando ricorsivamente]
= a.a.b.b.c.c.ε                    [caso base]
= "aabbcc"
```

$$\begin{cases} S \to aAbBcC \\ A \to aAa \\ B \to bBb \\ C \to cCc \\ \varepsilon \to \varepsilon \end{cases}$$

3. **(9 punti)** Chiamiamo k-PDA un automa a pila dotato di k pile. In particolare, uno 0-PDA è un NFA e un 1-PDA è un PDA convenzionale. Sappiamo già che gli 1-PDA sono più potenti degli 0-PDA (nel senso che riconoscono una classe più ampia di linguaggi). Mostra che i 2-PDA sono equivalenti alle Turing Machine.

PILA = PUSH/POP (NO PILA) 0-PDA = NFA (NON MEMORIA)

(1 NASTRO) 1-PDA = PDA (PUSH/POP) STACK

(2 NASTRI) 2-PDA = TM (TESTINA + NASTRO)

2-PDA ⇒ 2 NASTRI # ... # → ?

1 NASTRO → PUSH (NASTRO DI LAVORO) → ?

1 NASTRO → POP (NASTRO AGGIUNTIVO PER ALTRE OPERAZIONI)

TM (NASTRO SINGOLO / MULTINASTRO / SEMI-INFINITO)

CLASSE TURING-RICONOSCIBILE

SEMI-INFINITO # ☐ → . . . . (∞)

(K-PDA ≡ TM CON K NASTRI) → GOAL!

3. (9 punti) Dimostra che un linguaggio è decidibile se e solo se esiste un enumeratore che lo enumera seguendo l'ordinamento standard delle stringhe.

→ ORDINAMENTO LESSICOGRAFICO = {a...z, A...Z}

(ENUMERATORE) → Σ^* (STAMPA)

ABACO

ABBECEDARIO

L DECIDIBILE SSE ∃ ENUMERATORE CON ORDINE STANDARD

1 ⇒ SE L DECIDIBILE, ALLORA ∃ ENUMERATORE IN GRADO DI PRODURRE COME OUTPUT UNA STRINGA W = |K| → "INPUT CAOTICO" = STESSA CARDINALITÀ

W = $a_1 \to a_2 \ldots \to a_K$ $\{1 \ldots K\}$ = ORDINE LESSICOGRAFICO

STAMPA SEGUENDO ORDINE LESSICOGRAFICO → ARRIVI ALLA FINE (DECIDIBILE)

② SE ∃ ENUMERATORE, $\exists\, w = |k|$

$\downarrow$

L = DECIDIBILE

RIUSCIAMO A CONTARE LE STRINGHE FINITAMENTE CON ALG. DECIDIBILE

4. **(9 punti)** Considera il seguente problema: data una TM M a nastro semi-infinito, determinare se esiste un input w su cui M sposta la testina a sinistra partendo dalla cella numero 2023 (ossia se in qualche momento durante la computazione la testina si muove dalla cella 2023 alla cella 2022).

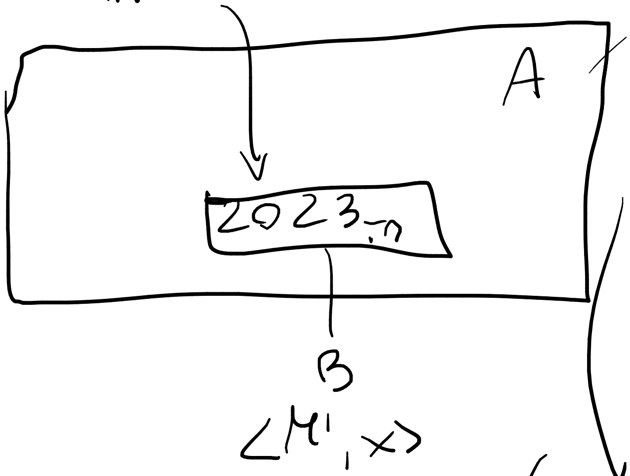
   (a) Formula questo problema come un linguaggio 2023_{TM}.

   (b) Dimostra che il linguaggio 2023_{TM} è indecidibile.

$2023_{TM} = \{ \langle M, w \rangle \mid M$ è una TM, w è una stringa, M accetta w se M si sposta a sx PARTENDO DA 2023 $\}$

(b) $A \leq_m B \Rightarrow A_{TM} \leq_m 2023_{TM}$

$\langle M, w \rangle$



FUNZIONE DI RIDUZIONE

$x \in A \iff f(x) \in B$

F = SU INPUT $\langle M, w \rangle$ =

1. SIMULA M' SU x $\rightarrow A_{TM}$ ($w \in L$)
2. SE M' ACCETTA, M SI SPOSTA A SX DA 2023 ALLA CELLA 2022 ALLORA ACCETTA
3. ALTRIMENTI VA IN LOOP (SEMI-INFINITO) (FINALE) q_{END}
4. RITORNA $\langle M' \rangle$

$HALT_{TM} \leq_m 2023_{TM}$

```
TM M'(input x):
1. Ignora l'input x
2. Simula M su w mantenendo il nastro di lavoro a destra della posizione 2024
3. Se M raggiunge uno stato finale:
   a. Sposta la testina alla posizione 2023
   b. Scrivi un simbolo dummy
   c. Sposta la testina a sinistra (posizione 2022)
   d. Entra in uno stato finale
4. Se M non termina, continua la simulazione indefinitamente
```