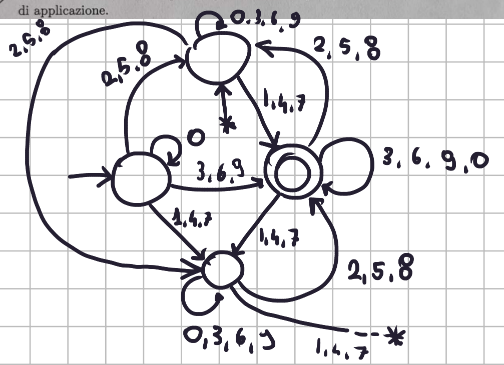
1 Automi — 10 Points

- Progettare un DFA che riconosca il linguaggio di stringhe su alfabeto $\Sigma = \{0, 1, \dots, 9\}$ tali che la somma delle cifre sia un multiplo di 3. Provare la correttezza del DFA proposto.
- Enunciare e dimostrare il *pumping lemma* per linguaggi regolari. Fornire un esempio di applicazione.



$$x + y \% 3 = 0 \iff \begin{cases} x \in [1,4,7] \land y \in [2,5,8] \\ \lor \\ y \in [1,4,7] \land x \in [2,5,8] \end{cases}$$

Se $x \% 3 = 0 \land y \% 3 = 0 \Rightarrow x + y \% 3 = 0$

Teo (Pumping Lemma): Sia $L \in REG$ e sia $w \in L$, $\exists p \le |w|$ tale che:

per una scomposizione $w = xyz$ tale che

$\bullet\ |y| \neq 0 \quad \bullet\ |xy| \le p \quad \bullet\ xy^i z \in L$

Dim: Sia p il numero di stati del DFA che accetta L, sia $w = w_1 w_2 \cdots w_n$

e sia $R = \{r_1, r_2 \cdots r_{n+1}\}$ la successione di stati: $\forall i \ \delta(r_i, w_i) = r_{i+1}$

oss $n \ge p \Rightarrow n+1 > p \Rightarrow$ in R c'e' uno stato ripetuto nei primi $p+1$ stati di R

$r_1 \cdots, r_j \cdots r_\ell \cdots r_{n+1}$ e $r_j = r_\ell$, pongo la scomposizione $w = xyz$ con

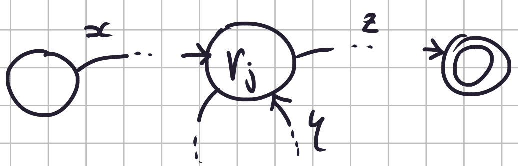
$\ell \le p+1$

$x = w_1 \cdots\cdots w_{j-1}$

$y = w_j \cdots\cdots w_{\ell-1} \Rightarrow |xy| = \ell - 1 \le p$ inoltre $j \neq \ell \Rightarrow |y| \neq 0$

$z = w_\ell \cdots\cdots w_n$

y porta da r_j a r_ℓ quindi $xy^i z \in L$ perche'



Esempio di applicazione

$w \in L = \{0^n 1^n \mid n \in \mathbb{N}\}$

sia $p = n \Rightarrow w = xyz$ ci sono diversi modi di scomporre

$w = 0^n 1^{n-q} 1^q \Rightarrow 0^n 1^{(n-q)^i} 1^q \notin L$

$w = 0^q\ 0^{n-q}\ 1^n \Rightarrow 0^q\ 0^{(n-q)^i}\ 1^n \notin L$

$w = 0^q (0^{n-q} 1^{n-q}) 1^q \Rightarrow 0^q (0^{n-q} 1^{n-q})^i 1^q = \underbrace{00\cdots0}_{q}\ 0101\cdots01\ \underbrace{11\cdots1}_{q} \notin L$

2 Calcolabilità — 10 Points

- Si consideri il linguaggio W_{TM} delle stringhe della forma $\langle M, w, a\rangle$, dove M è una TM, w è una stringa appartenente all'alfabeto di input di M, ed a è un simbolo appartenente all'alfabeto di nastro di M, tale che M scrive a sul nastro quando è eseguita su input w. Mostrare che W_{TM} è indecidibile.
- Dimostrare che un linguaggio è Turing-riconoscibile se e solo se esiste una TM multi-nastro che lo riconosce.

Devo ridurre un linguaggio non decidibile a W_{TM}

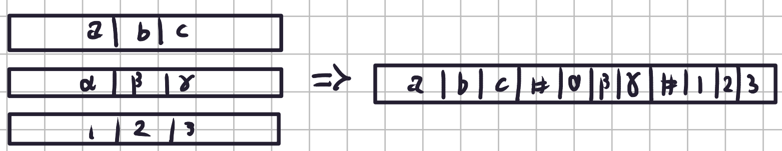
$\langle M, w\rangle \in A_{TM} \iff R(\langle M, w\rangle) \in W_{TM}$ Definisco R:

- Su input $\langle M, w\rangle$
- Esegue $M(w)$
- Se M accetta, prende a: ultimo carattere scritto sul nastro da M
- Se M accetta, ritorna $\langle M, w, a\rangle$
- Se M rifiuta, crea D tale che "D rifiuta su ogni input"
- ritorna $\langle D, w, \#\rangle$

R e' una riduzione da A_{TM} a W_{TM}.

Una TM multinastro (MTM) e' equivalente ad una singolo nastro, per def. Un linguaggio e' riconoscibile se una TM lo riconosce.
Una MTM puo' naturalmente simulare una TM usando solo un nastro.

D'altra parte, una TM può simulare una MTM utilizzando il carattere speciale # come separatore sul nastro singolo, per identificare i differenti nastri



La TM farà uso di marcatori sui caratteri per identificare i caratteri sui nastri nei quali sono riposte le testine

Se  $\xrightarrow{\text{nella TM}}$  quindi $\forall a \in \Sigma(MTM) \Rightarrow a, \dot{a} \in \Sigma(TM)$

in tal modo la TM può scannerizzare l'intero nastro cercando i simboli marcati ed operare su questi seguendo le regole della funzione di transizione originale della MTM.

Quindi, essendo una MTM equivalente ad una TM (in termini di computabilità), queste possono riconoscere lo stesso insieme di linguaggi.

3 **Complessità** 10 Points

- Mostrare che il seguente problema, detto 1/2-*CLIQUE*, è *NP*-completo: Dato un grafo G con un numero pari di vertici, stabilire se esiste una *clique* in G di taglia uguale alla metà del numero di vertici nel grafo.
- Definire il problema 2-*SAT* e mostrare che esso appartiene a P.

Sappiamo che CLIQUE e' NP-completo, possiamo definire K-CLIQUE con una TM $C(G,K) = accept \iff$ K-CLIQUE in G. Definisco R:

- R su input G
- conta $n = |V(G)|$ $O(n)$
- scrive in output $\langle G, K \rangle$

R e' una riduzione polinomiale da 1/2-CLIQUE a K-CLIQUE.

2-SAT = $\{ \phi \mid \phi$ e' una 2-CNF soddisfacibile $\}$

Teo: 2-SAT e' in P

Dim: Si puo' ridurre 2-SAT al problema della ricerca delle componenti fort. connesse. data $\phi(x_1 \dots, x_n)$: Definisco G_ϕ: grafo tale che i nodi sono le variabili x_i e $\forall (\bar{x}_i \vee x_j) \in \phi$ esiste $(x_i, x_j) \in E(G_\phi)$

$$(x_i \vee \bar{x}_j) \in \phi \quad \text{"} \quad (\bar{x}_i, \bar{x}_j) \in E(G_\phi)$$

Lemma 1): Se x e $\bar{x}$ sono nella stessa comp. ϕ non e' SAT

Dim 1): Se (x_i, x_j) e (x_j, x_k) $\exists$ cammino $x_i \rightsquigarrow x_k$ in $G_\phi \Rightarrow \bar{x}_i \vee x_k \in \phi$

Quindi se x e $\bar{x}$ sono nella stessa comp: $\begin{cases} x \to \bar{x} \\ \bar{x} \to x \end{cases} \Rightarrow \begin{matrix} \bar{x} \vee \bar{x} = \bar{x} \\ x \vee x = x \end{matrix} \Rightarrow x \wedge \bar{x} \in \phi \Rightarrow$ NO SAT

Lemma 2) Se x e $\bar{x}$ non sono nella stessa componente, ϕ e' SAT, dato il seguente assegnamento:

- Si fa l'ordine topologico del grafo delle componenti $\{C_1, C_2 \dots, C_m\}$
- $x = 1 \iff x \in C_i \wedge \bar{x} \in C_{i-K}$

A questo punto NON esistono archi (a,b) dove $a=1$ e $b=0$:

- Se (a,b) e $a=1$ e $b=0$
- $\exists$ $(\bar{b}, \bar{a})$ con $\bar{b}=1$ e $\bar{a}=0$
- $a \in C_i$ $b \in C_{i+K}$ per qualche $K>0$
- $b=0 \Rightarrow \bar{b} \in C_{i+K+l}$ per qualche $l>0$
- $\bar{a} \in C_{i-K}$ ma se $(\bar{b}, \bar{a})$ allora $C_{i+K+l} < C_i \Rightarrow$ Contraddizione.

Non essendoci archi di tal tipo e' garantita la soddisfacibilita'.

si può costruisre il grafo ed applicare l'algoritmo di Tarjan per risolvere 2SAT.