Elementi di
Teoria della Computazione

Classe: Resto_2 - Prof.ssa Marcella Anselmo

Tutorato
24/06/2022 ore 15:00-17:00

Sesta Esercitazione

a cura della dott.ssa Manuela Flores

Preappello 15/06/2022: linguaggi regolari

1. Per ognuno dei seguenti linguaggi, indicare se sono regolari o non regolari, giustificando la risposta.

   (a) $X = \{ww \mid w \in \{a, b\}^*\}$.

   (b) $Pref(L) = \{x \in \Sigma^* \mid \exists y \in \Sigma^* \text{ tale che } xy \in L\}$, con $L \subseteq \Sigma^*$ regolare.

   (c) $Y = \{a^i b^j \mid i \neq j \;\; e \;\; i, j \geq 0\}$.

Lezione 13 pag. 91

Pumping lemma: dimostrare la non regolarità (esempio)

Dimostriamo che $L = \{a^n b^n \mid n \geq 0\}$ **non è regolare**!

Dimostrazione.

Supponiamo per assurdo che L sia regolare. Allora vale il pumping lemma.
Sia p la lunghezza del pumping.

Consideriamo la stringa $s = a^p b^p$.
Ovviamente $s \in L$ e $|s| = 2p$ (soddisfa le ipotesi $|s| \geq p$).

Consideriamo **TUTTE** le possibili fattorizzazioni di $s = a^p b^p$ in 3 stringhe x, y, z con le proprietà delle condizioni: $|xy| \leq p$ e $|y| \geq 1$.

$$\overbrace{a\ a\ a\ a\ a\ a\ a\ a\ a}^{p}\ \overbrace{b\ b\ b\ b\ b\ b\ b\ b\ b}^{p}$$

Lezione 13 pag. 97

Pumping lemma: dimostrare la non regolarità (esempio)

Dimostriamo che $L = \{a^n b^n \mid n \geq 0\}$ **non è regolare**!

Dimostrazione.

...

Consideriamo **TUTTE** le possibili fattorizzazioni di $s = a^p b^p$ in 3 stringhe x, y, z con le proprietà delle condizioni: $|xy| \leq p$ e $|y| \geq 1$.

Quindi $y = a^m$, per $1 \leq m \leq p$. Per $i = 2$, $xy^2z = a^{p+m}b^p \notin L$.

Contraddizione

a a a a a a a a a a a a b b b b b b b b b

x y y z

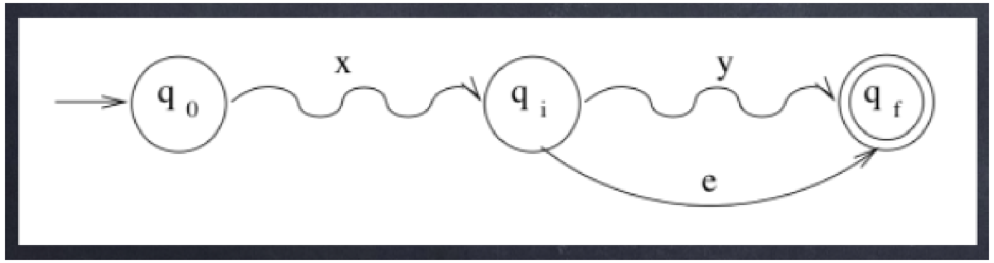
Lezione 11 pag. 24

Chiusura di REG rispetto a insieme **suffisso** e **prefisso**

Sia $A = (Q, \Sigma, \delta, q_0, \{q_f\})$ un NFA tale che non esista nessuna transizione entrante in q_0 e nessuna uscente da q_f (ipotesi sempre riproducibili usando ϵ-transizioni).

Descrivere il linguaggio accettato da ognuna delle seguenti varianti di A:

2. automa costruito da A aggiungendo una ϵ-transizione verso q_f da ogni stato che può raggiungere q_f (lungo cammini che possono comprendere sia simboli di Σ che ϵ)



L'automa ottenuto riconosce l'insieme dei **prefissi** di L: $\{x \in \Sigma^* \mid \exists y \in \Sigma^* : xy \in L\}$

REG è chiuso rispetto ai prefissi

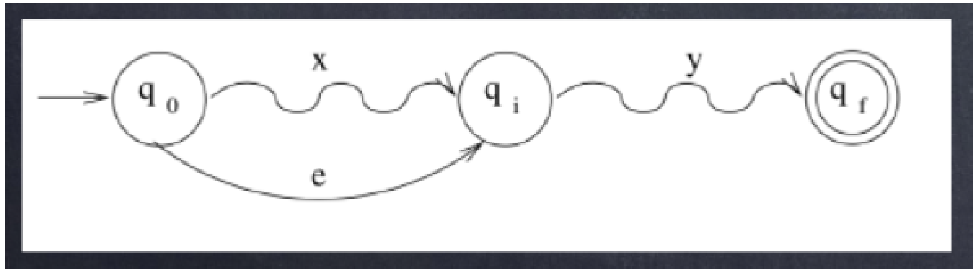
Lezione 11 pag. 19

Chiusura di REG rispetto a insieme **suffisso** e **prefisso**

Sia $A = (Q, \Sigma, \delta, q_0, \{q_f\})$ un NFA tale che non esista nessuna transizione entrante in q_0 e nessuna uscente da q_f (ipotesi sempre riproducibili usando ϵ-transizioni).

Descrivere il linguaggio accettato da ognuna delle seguenti varianti di A:

1. automa costruito da A aggiungendo una ϵ-transizione da q_0 verso ogni stato raggiungibile da q_0 (lungo cammini che possono comprendere sia simboli di Σ che ϵ)

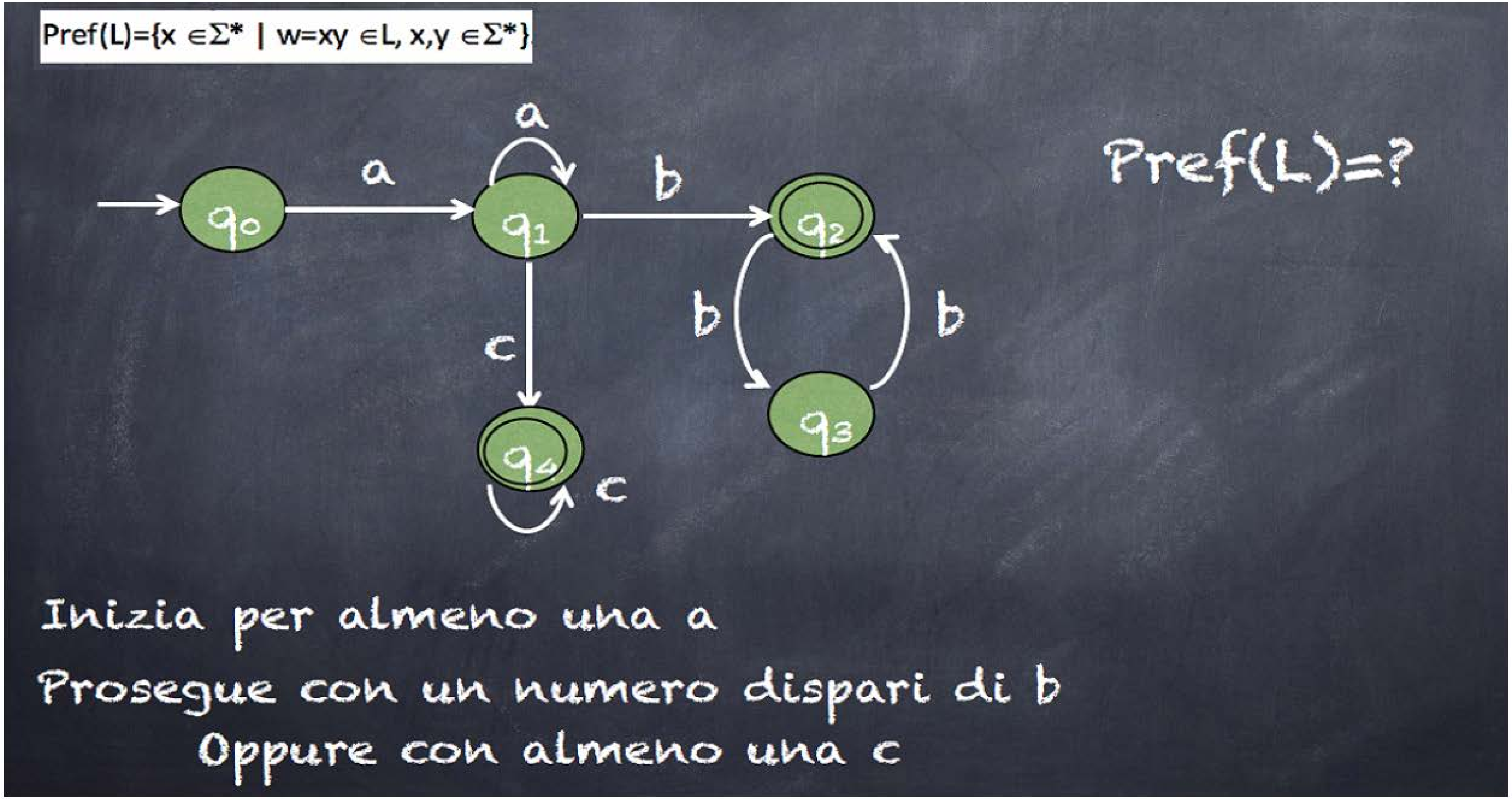


L'automa ottenuto riconosce l'insieme dei **suffissi** di L: $\{y \in \Sigma^* \mid \exists x \in \Sigma^* : xy \in L\}$

REG è chiuso rispetto ai suffissi

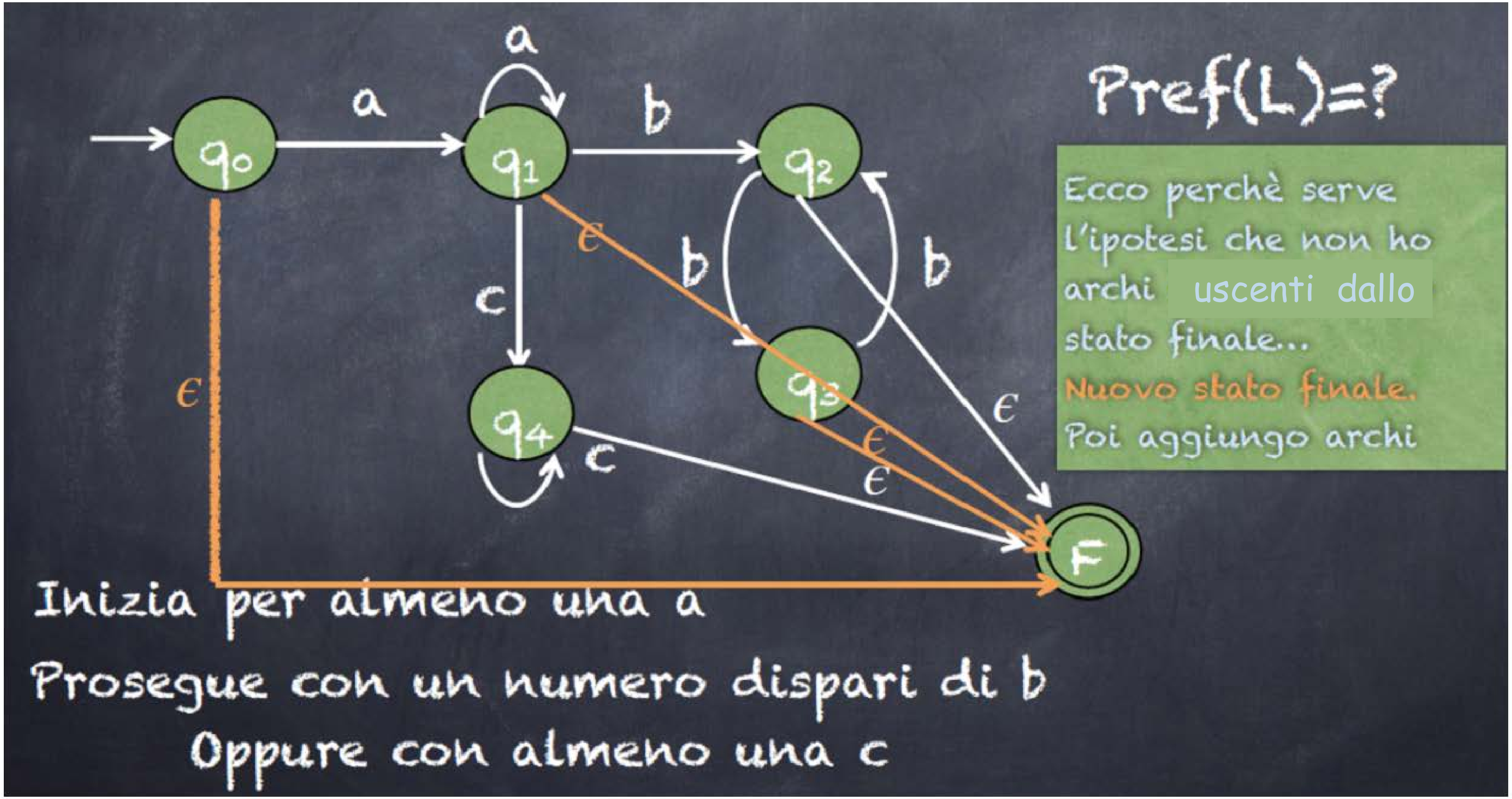
Lezione 11 pag. 26

Esercizi sui prefissi



Lezione 11 pag. 32

Esercizi sui prefissi



Lezione 6 pag. 34

Chiusura dei linguaggi regolari

Dimostreremo che i linguaggi regolari sono chiusi rispetto alle seguenti operazioni.

Siano L_1 e L_2 linguaggi regolari, allora:

- **unione**: $L_1 \cup L_2$ è regolare
- **concatenazione**: $L_1 \circ L_2$ è regolare
- **star**: L_1^*, L_2^* sono regolari
- **reversal**: L_1^R, L_2^R sono regolari
- **complemento**: $\overline{L_1}, \overline{L_2}$ sono regolari
- **intersezione**: $L_1 \cap L_2$ è regolare

Lezione 11 pag. 34

Cosa si può fare con gli automi finiti

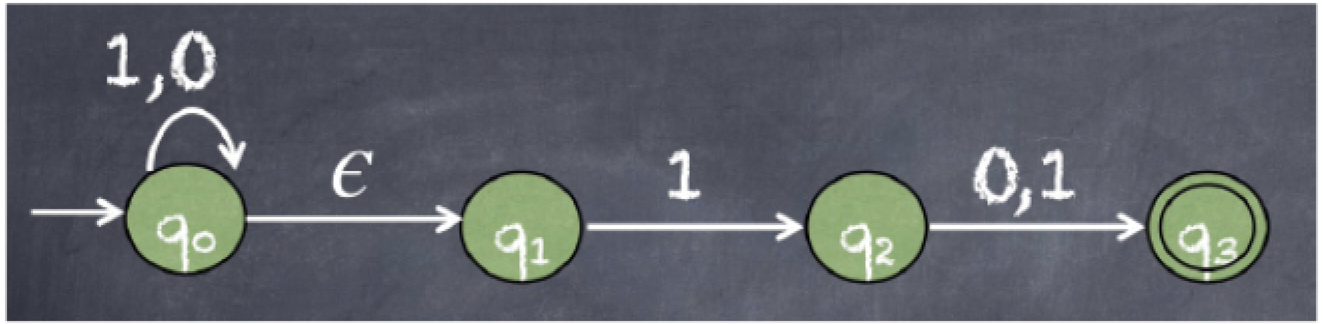
- Deterministici (REG)
- Nondeterministici
- Determinizzazione (equivalenza dei modelli)
- REG
- REG: proprietà di chiusura (funzioni su REG)

unione, concatenazione, star, complemento, intersezione, reverse, prefisso, suffisso, differenza

$$L - M = L \cap \overline{M}$$

Preappello 15/06/2022: da NFA a DFA

2. Trasformare il seguente NFA nel DFA equivalente utilizzando la costruzione presentata nella dimostrazione del Teorema sull'equivalenza NFA-DFA. Riportare con precisione la descrizione della funzione di transizione e produrre il diagramma di stato (limitandosi agli stati raggiungibili dallo stato iniziale del DFA). Fornire una espressione regolare che descrive il linguaggio accettato dall'automa.



Lezione 9 pag. 68

Subset construction: da NFA a DFA

Costruzione.

Sia $\mathbb{N} = (Q_N, \Sigma, \delta_N, q_N, F_N)$ un NFA, costruiamo il DFA $\mathbb{M} = (Q_M, \Sigma, \delta_M, q_M, F_M)$ come:

1. $Q_M = P(Q_N)$, **insieme potenza** di Q_N; osserviamo che $|P(Q_N)| = 2^{|Q_N|}$.
   - Q_M contiene tutti i "possibili stati" in cui può terminare una transizione di $\mathbb{M}$, cioè tutte le combinazioni possibili di stati di Q_N.
2. $q_M = E(q_N)$, lo stato iniziale di $\mathbb{M}$ non è solo q_N ma anche tutti gli stati raggiungibili da q_N utilizzando solo ϵ-transizioni, quindi $E(q_N)$.
3. $F_M = \{R \in Q_M \mid R \cap F_N \neq \emptyset\}$
4. $\forall R \in Q_M$, $\forall a \in \Sigma$:

$$\delta_M(R, a) = E(\cup_{r \in R} \delta_N(r, a)) = \cup_{r \in R} E(\delta_N(r, a))$$

Lezione 9 pag. 36

NFA: computazione (`HUM, 2.5.3–2.5.4`)

Come per i DFA, siamo interessati a definire le computazioni di δ in termini di stringhe.

Innanzitutto, definiamo l'insieme degli stati raggiungibili da uno stato usando solo ϵ-transizioni.

Sia $\mathbb{M} = (Q, \Sigma, \delta, q_0, F)$ un **NFA** e $q \in Q$. La ϵ-**chiusura** $E(q)$ di q è il sottinsieme di Q definito ricorsivamente come segue:

passo base: $q \in E(q)$

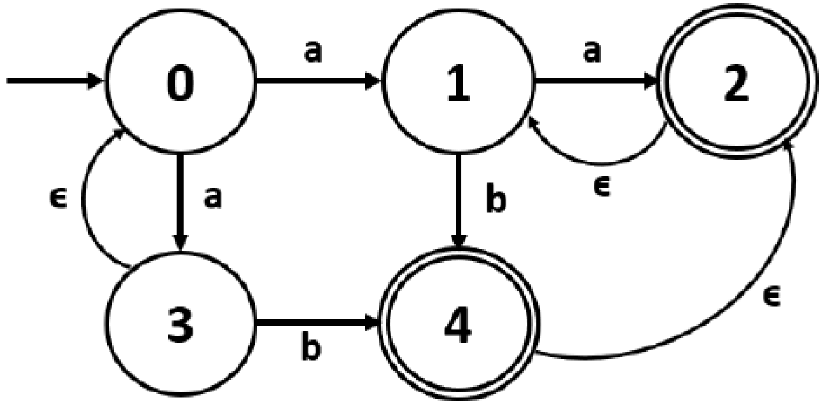
passo ricorsivo: $\forall p \in E(q)$, $\delta(p, \epsilon) \subseteq E(q)$

Sia $R \subseteq Q$. La ϵ-chiusura $E(R)$ di R è:

$$E(R) = \cup_{q \in R} E(q)$$

Subset construction: da NFA a DFA (esempio)

Consideriamo il seguente NFA $\mathbb{N} = (Q_N, \Sigma, \delta_N, 0, F_N)$

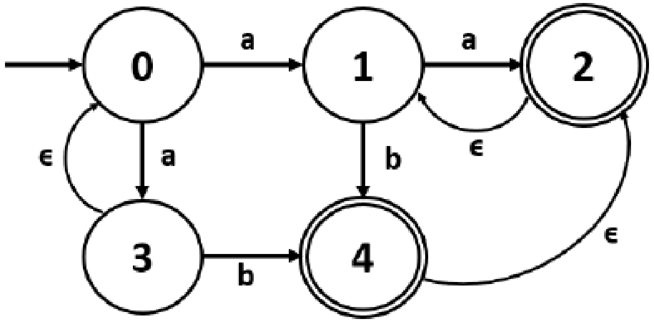


	a	b	ϵ
0	{1,3}	∅	∅
1	{2}	{4}	∅
2	∅	∅	{1}
3	∅	{4}	{0}
4	∅	∅	{2}

Costruiamo innanzitutto le ϵ-chiusure: $E(0) = \{0\}$, $E(1) = \{1\}$, $E(2) = \{2, 1\}$, $E(3) = \{3, 0\}$, e $E(4) = \{4, 2, 1\}$.

Lezione 9 pag. 111

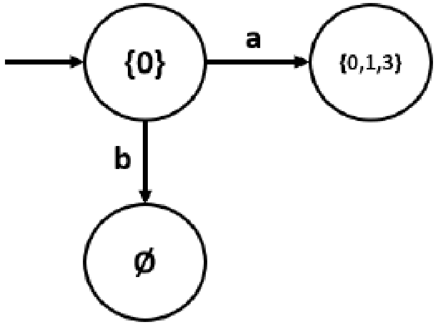
Subset construction: da NFA a DFA (esempio)



	a	b	ϵ
0	{1,3}	∅	∅
1	{2}	{4}	∅
2	∅	∅	{1}
3	∅	{4}	{0}
4	∅	∅	{2}

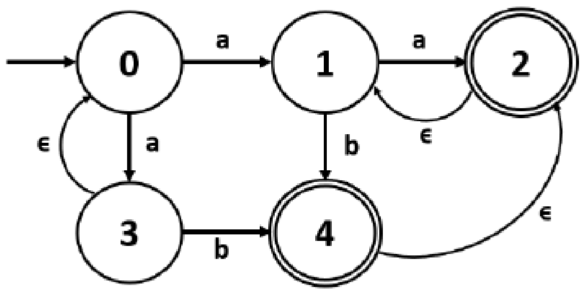
Costruiamo le transizioni per ogni stato:

- $\delta_M(\{0\}, a) = E(\delta_N(0, a)) = E(\{1, 3\}) = E(\{1\}) \cup E(\{3\}) = \{0, 1, 3\}$
- $\delta_M(\{0\}, b) = E(\delta_N(0, b)) = E(\emptyset) = \emptyset$



Lezione 9 pag. 115

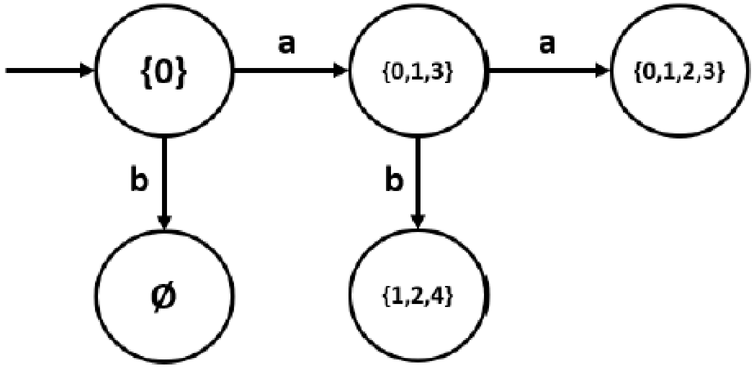
Subset construction: da NFA a DFA (esempio)



	a	b	ϵ
0	{1,3}	∅	∅
1	{2}	{4}	∅
2	∅	∅	{1}
3	∅	{4}	{0}
4	∅	∅	{2}

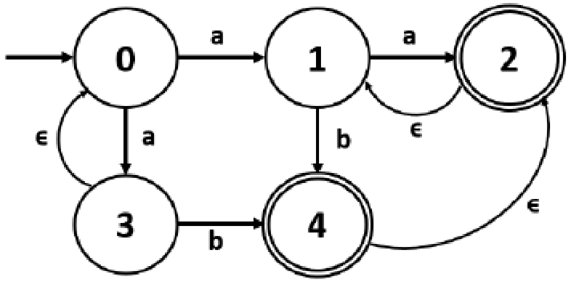
Costruiamo le transizioni per ogni stato:

- $\delta_M(\{0,1,3\},a) = E(\delta_N(0,a) \cup \delta_N(1,a) \cup \delta_N(3,a)) = E(\{1,3\} \cup \{2\} \cup \emptyset) = \{0,1,2,3\}$
- $\delta_M(\{0,1,3\},b) = E(\delta_N(0,b) \cup \delta_N(1,b) \cup \delta_N(3,b)) = E(\emptyset \cup \{4\} \cup \{4\}) = E(\{4\}) = \{1,2,4\}$



Lezione 9 pag. 122

Subset construction: da NFA a DFA (esempio)



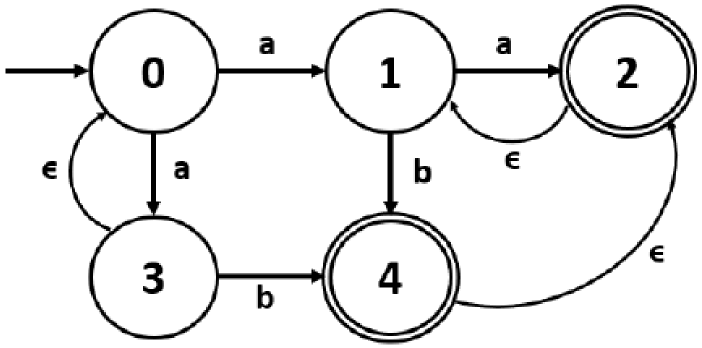
	a	b	ϵ
0	{1,3}	∅	∅
1	{2}	{4}	∅
2	∅	∅	{1}
3	∅	{4}	{0}
4	∅	∅	{2}

Costruiamo le transizioni per ogni stato:

- $\delta_M(\{1,2,4\},a) = E(\{2\} \cup \emptyset \cup \emptyset) = \{1,2\}$
- $\delta_M(\{1,2,4\},b) = E(\{4\} \cup \emptyset \cup \emptyset) = \{1,2,4\}$
- $\delta_M(\{0,1,2,3\},a) = E(\{1,3\} \cup \{2\} \cup \emptyset \cup \emptyset) = \{1,3,0,2\}$
- $\delta_M(\{0,1,2,3\},b) = E(\emptyset \cup \{4\} \cup \emptyset \cup \{4\}) = \{1,2,4\}$
- $\delta_M(\{1,2\},a) = E(\{2\} \cup \emptyset) = \{1,2\}$
- $\delta_M(\{1,2\},b) = E(\{4\} \cup \emptyset) = \{1,2,4\}$

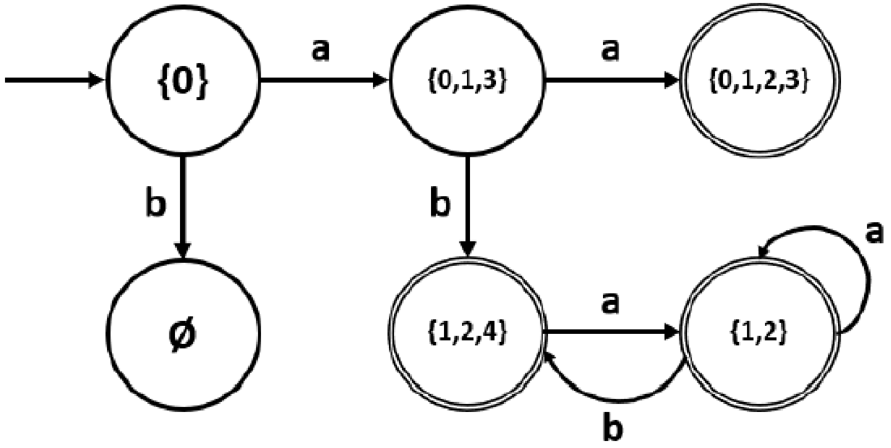
Lezione 9 pag. 123

Subset construction: da NFA a DFA (esempio)



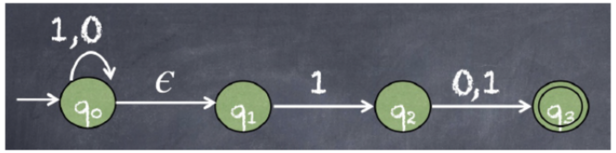
	a	b	ϵ
0	{1,3}	∅	∅
1	{2}	{4}	∅
2	∅	∅	{1}
3	∅	{4}	{0}
4	∅	∅	{2}

Consideriamo solo le **parti raggiungibili** dallo stato iniziale del DFA.



Preappello 15/06/2022: da NFA a DFA

2. Trasformare il seguente NFA nel DFA equivalente utilizzando la costruzione presentata nella dimostrazione del Teorema sull'equivalenza NFA-DFA. Riportare con precisione la descrizione della funzione di transizione e produrre il diagramma di stato (limitandosi agli stati raggiungibili dallo stato iniziale del DFA). Fornire una espressione regolare che descrive il linguaggio accettato dall'automa.



Preappello 15/06/2022: MdT che riconosce

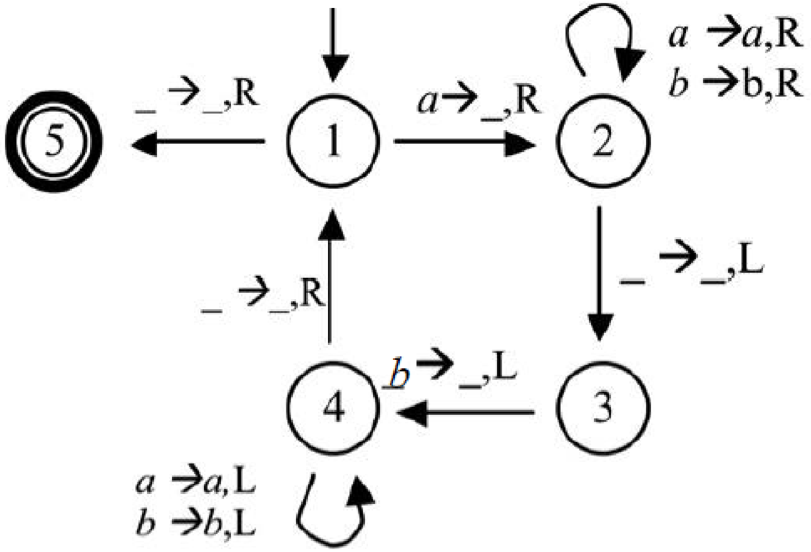
Esercizio 3 (*6 punti*)
Descrivere una **MdT** deterministica che **riconosce** il linguaggio $\boldsymbol{Y = \{\ a^i\, b^j \mid i \neq j \text{ e } i, j \geq 1\}}$ sull'alfabeto $\Sigma = \{a, b\}$.

La descrizione deve essere fornita tramite **settupla** o **diagramma di stato** e deve essere accompagnata da una descrizione **ad alto livello** che ne giustifichi il funzionamento.

Lezione 15 pag. 29

Cancella ripetutamente: prima occorrenza di (a) e ultima di (b)
se la stringa era del tipo $a^n b^n$, non rimangono simboli
Cinque passi

1. Se leggi _, vai a 5. Se leggi a, scrivi _ e vai a 2.
2. Spostati a destra (R) di tutti a e b. Al primo _, muovi a sinistra (L) e vai a 3
3. se leggi b, scrivi _ e vai a 4.
4. Spostati a sinistra (L) di tutti a e b. Leggendo _, muovi R e vai a 1.
5. Accept.



Le transizioni non indicate portano in uno stato di Reject (per esempio se in 1. leggi b).

Lezione 15 pag. 48

Altra strategia MdT per $\{ 0^n 1^n \mid n \geq 0 \}$.

Una macchina di Turing che accetta $L = \{0^n 1^n \mid n > 0\}$.
(Descrizione ad alto livello)

(passo 1) Se legge 0 lo sostituisce con X, se legge 1 rifiuta, se legge Y va al passo 4.

(passo 2) Scorre il nastro verso destra, se trova 1 lo sostituisce con Y, altrimenti rifiuta.

(passo 3) Scorre il nastro a sinistra fino a incontrare X, si sposta a destra e ripete dal passo 1.

(passo 4) Scorre il nastro a destra. Se legge solo delle Y e poi il carattere $\sqcup$, accetta. Altrimenti rifiuta.

$000111 \to X00111 \ldots \to X00Y11 \ldots \to XX0Y11 \ldots \to XXXYYY$

Lezione 20 pag. 16

Esempio: MdT a 2 nastri per $\{0^n1^n \mid n > 0\}$

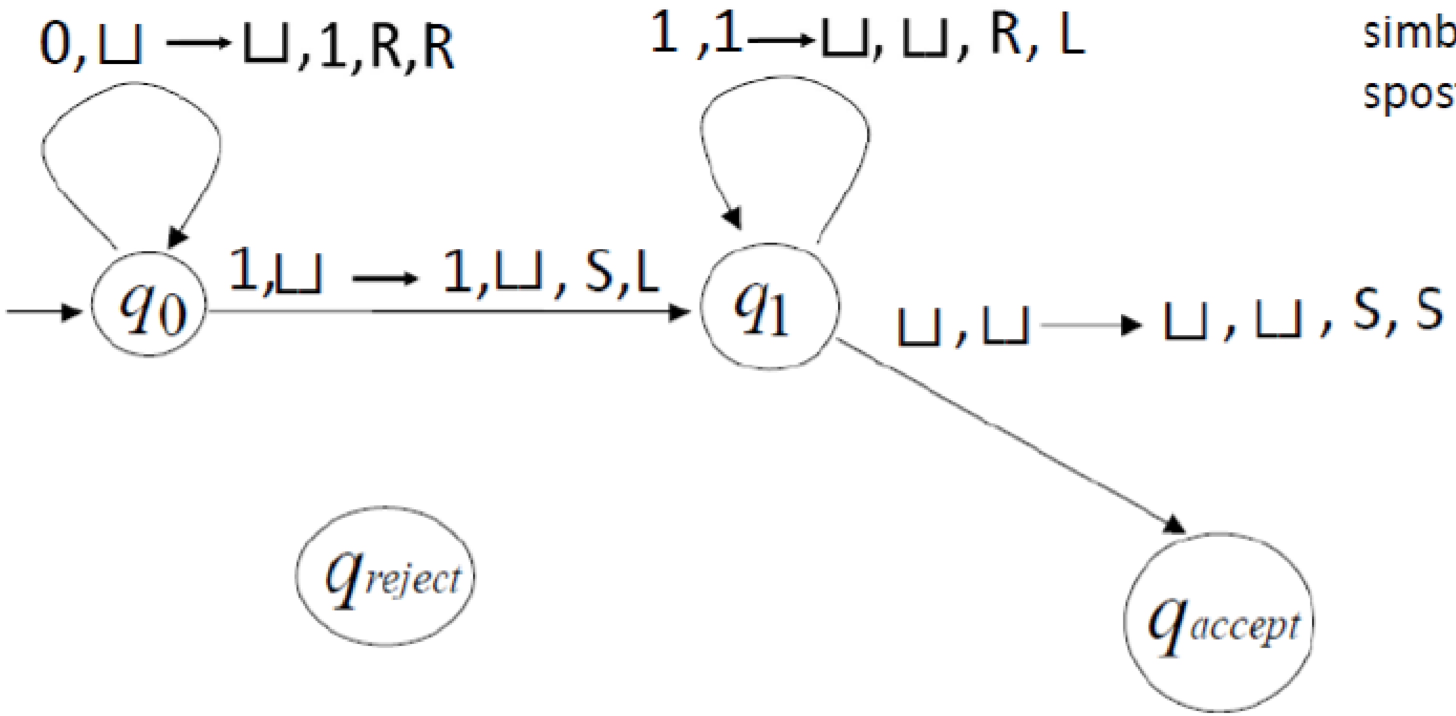
1. Scorre il primo nastro verso destra fino al primo 1: per ogni 0, scrive un 1 sul secondo nastro
2. Scorre il primo nastro verso destra e il secondo nastro verso sinistra: se i simboli letti non sono uguali, termina in q_{reject}
3. Se legge $\sqcup$ su entrambi i nastri, termina in q_{accept}

stato attuale	simbolo letto	Valore funzione δ
q_0	$(0, \sqcup)$	$q_0, (\sqcup, 1), (R, R)$
q_0	$(1, \sqcup)$	$q_1, (1, \sqcup), (S, L)$
q_1	$(1, 1)$	$q_1, (\sqcup, \sqcup), (R, L)$
q_1	$(\sqcup, \sqcup)$	$q_{accept}, (\sqcup, \sqcup), (S, S)$

Nota: Se $\delta(q, \gamma, \gamma')$ non è presente nella tabella, con $q \in Q \setminus \{q_{accept}, q_{reject}\}$ allora $\delta(q, \gamma, \gamma') = (q_{reject}, \gamma, \gamma', S, S)$.

Lezione 20 pag. 17

Esempio: MdT a 2 nastri per $\{0^n1^n \mid n > 0\}$



Transizioni omesse vanno in q_{reject} (senza cambiare i simboli letti o spostare la testina)

Preappello 15/06/2022: vero, falso, non si sa

Esercizio 4 (4 punti)
Per ognuna delle seguenti affermazioni dire se è **vera**, **falsa** oppure **non si sa**. In ogni caso, occorre **motivare** la risposta, citando eventuali risultati noti utilizzati.

a) Se L è riconosciuto da una MdT **non-deterministica** allora L è riconosciuto da una MdT deterministica.
b) Se L è riconosciuto in tempo **polinomiale** da una MdT **non-deterministica** allora L è riconosciuto in tempo polinomiale da una MdT deterministica.
c) Il linguaggio **3-SAT** è **indecidibile**.
d) Comunque si scelgano due linguaggi **NP- completi** A e B si ha che $\mathbf{A \leq_p B}$ e $\mathbf{B \leq_p A}$.

Lezione 22 pag. 38

Equivalenza tra il modello deterministico e quello non deterministico

Teorema

Per ogni macchina di Turing non deterministica N esiste una macchina di Turing deterministica D equivalente ad N, cioè tale che $L(N) = L(D)$.

Lezione 29 pag. 37

MdT non deterministiche: tempo di esecuzione

Il tempo di esecuzione di una MdT non deterministica N è la funzione $f : \mathbb{N} \to \mathbb{N}$ dove
$f(n) =$ massimo delle altezze degli alberi, ognuno dei quali rappresenta le possibili computazioni su input w, al variare di $w \in \Sigma^n$.

Teorema

Sia $t(n)$ una funzione tale che $t(n) \geq n$. Per ogni macchina di Turing a nastro singolo, non deterministica N avente tempo di esecuzione $t(n)$ esiste una macchina di Turing a nastro singolo, deterministica e di complessità di tempo $2^{O(t(n))}$, equivalente ad N.

Lezione 30 pag. 30

P, *NP* e *co-NP*

Le osservazioni precedenti si applicano a qualsiasi linguaggio in NP e pongono il problema del rapporto tra la classe NP e la classe $coNP = \{L \mid \overline{L} \in NP\}$.

Per ognuno di questi linguaggi non è noto se tale linguaggio appartenga o meno a NP.

Le risposte alle domande

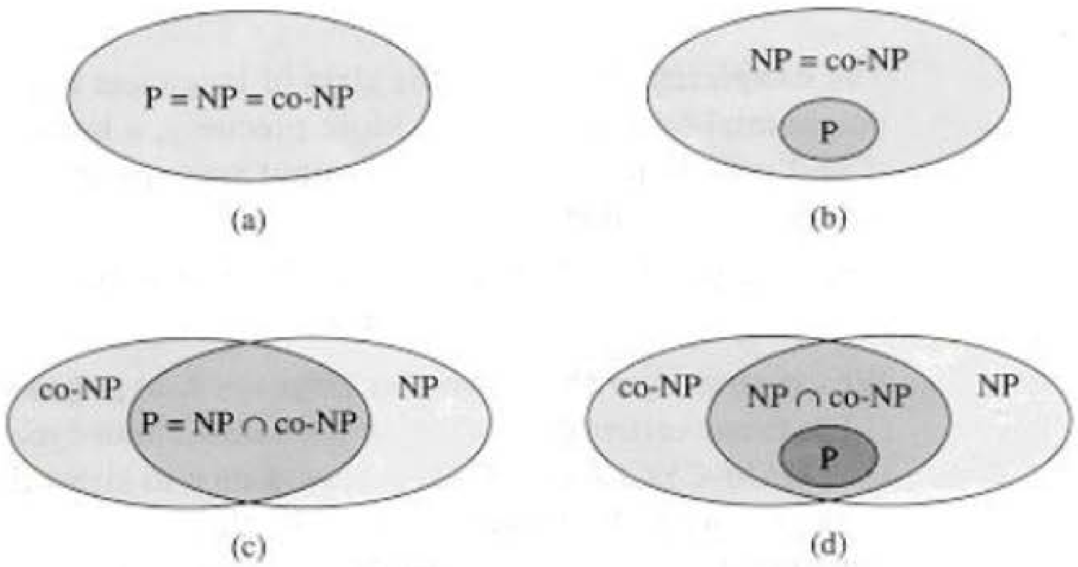
$$P = NP?$$

$$NP = coNP?$$

danno luogo ai seguenti quattro possibili scenari.

Lezione 30 pag. 31

P, NP e co-NP



Vi sono quattro possibilità:

1. $P = NP = coNP$
2. $P \subsetneq NP = coNP$
3. $NP \neq coNP$, $P = NP \cap coNP \subsetneq NP$ (e quindi $P = NP \cap coNP \subsetneq coNP$)
4. $NP \neq coNP$, $P \subsetneq NP \cap coNP \subsetneq NP$ (e quindi $P \subsetneq NP \cap coNP \subsetneq coNP$)

Lezione 32 pag. 24

3SAT è *NP* – completo

Occorre prima dimostrare che 3SAT è NP-completo.

- 3SAT $\in$ NP. Infatti 3SAT è verificabile in tempo polinomiale perché è un caso particolare di SAT (che sappiamo essere in NP).

- E' possibile dimostrare poi che:

$$\text{SAT} \leq_p \text{SAT}_{\text{CNF}} \leq_p 3\ \text{SAT}$$

Nota: 3SAT pur essendo un caso particolare di SAT è di «difficoltà maggiore o uguale» a SAT.

Lezione 30 pag. 16

La classe *NP*

Teorema 7.20
Un linguaggio L è in NP se e solo se esiste una macchina di Turing non deterministica che decide L in tempo polinomiale.

Dunque i linguaggi della classe NP sono decidibili:
3SAT è in NP, quindi è decidibile in tempo polinomiale da una MdT non deterministica $\Rightarrow$ 3SAT NON può essere indecidibile

Definizione
Sia $t : \mathbb{N} \to \mathbb{R}^+$ *una funzione. La classe di complessità in tempo non deterministico* $NTIME(t(n))$ *è*

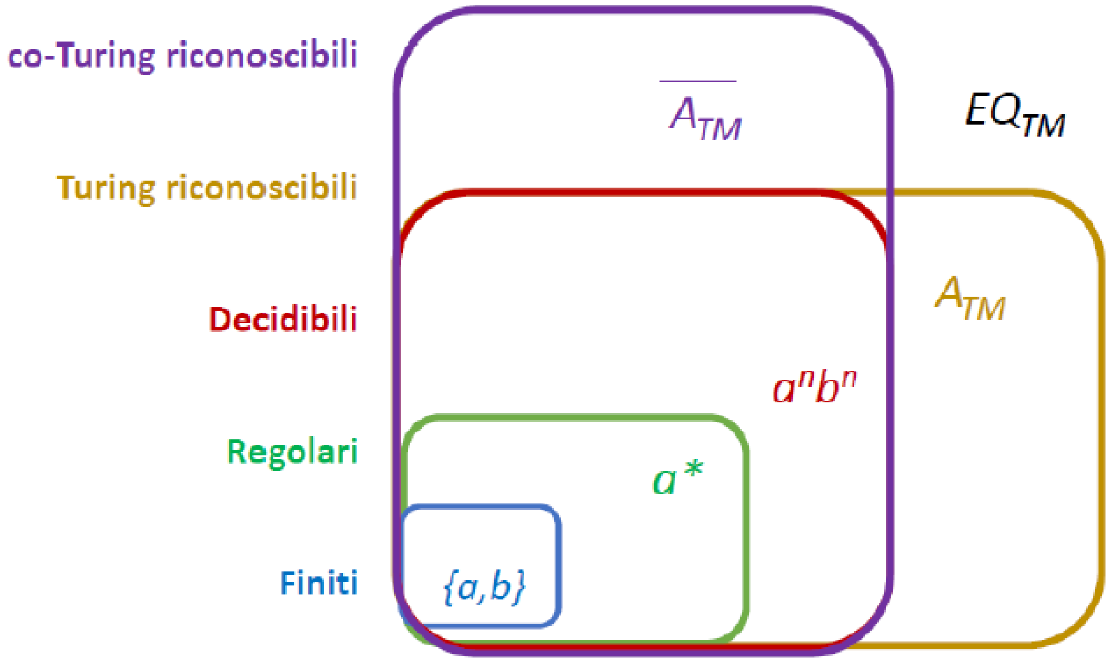
$$NTIME(t(n)) = \{L \mid \exists \text{ una macchina di Turing non deterministica } M \text{ che decide } L \text{ in tempo } O(t(n))\}$$

Corollario 7.22

$$NP = \bigcup_{k \geq 0} NTIME(n^k)$$

Lezione 28 pag. 22

The right picture



Decidibili = Turing riconoscibili ∩ co-Turing riconoscibili

Lezione 32 pag. 17

NP - completezza

Vogliamo definire quando un linguaggio B è uno dei linguaggi «più difficili» della classe NP.

Abbiamo visto un modo per definire quando B è «più difficile» di A, ovvero quando A è di difficoltà «minore o uguale» a B:

$$A \leq_p B$$

Quindi B è uno dei linguaggi «più difficili» della classe NP.....

Definizione

Un linguaggio B è *NP-completo* se soddisfa le seguenti due condizioni:

1. B appartiene a NP
2. Per ogni linguaggio A in NP, $A \leq_p B$ (ovvero B è NP-hard)

Preappello 15/06/2022: A_TM e HALT_TM

Esercizio 5 (5 punti)

a) **Definire** i linguaggi A_{TM} e $HALT_{TM}$.
b) Descrivere i **problemi decisionali** a cui sono associati i linguaggi A_{TM} e $HALT_{TM}$.
c) Siano A e B due linguaggi. **Definire** cosa significa che **$A \leq_m B$**, ovvero che A è **riducibile** mediante funzione a B.

Esercizio bonus*

d) Durante il corso abbiamo visto che $A_{TM} \leq_m HALT_{TM}$. Dimostrare adesso che **$HALT_{TM} \leq_m A_{TM}$**.

Lezione 25 pag. 36

Un problema indecidibile

$$\boxed{A_{TM} = \{\langle M, w\rangle \mid M \text{ è una MdT e } M \text{ accetta } w\}}$$

A_{TM} è il linguaggio associato al problema decisionale dell'accettazione di una macchina di Turing.

Teorema

Il linguaggio A_{TM} non è decidibile.

Lezione 27 pag. 25

Indecidibilità del problema della fermata

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una MdT e } M \text{ accetta } w\}$$

$$\boxed{HALT_{TM} = \{\langle M, w\rangle \mid M \text{ è una MdT e } M \text{ si arresta su } w\}}$$

Teorema
$A_{TM} \leq_m HALT_{TM}$.

Dimostrazione
Occorre definire una funzione calcolabile $f : \Sigma^* \to \Sigma^*$ tale che $f(\langle M, w\rangle) = \langle M', w\rangle$ e

$$\langle M, w\rangle \in A_{TM} \quad \text{sse} \quad \langle M', w\rangle \in HALT_{TM}$$

Lezione 27 pag. 17

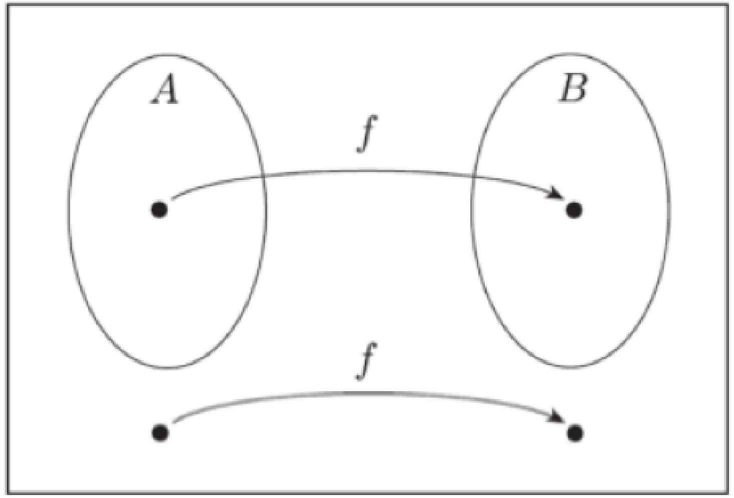
Riducibilità mediante funzione

Definizione

Un linguaggio $A \subseteq \Sigma^*$ *è riducibile mediante funzione a un linguaggio* $B \subseteq \Sigma^*$, *e scriveremo* $A \leq_m B$, *se esiste una funzione calcolabile* $f : \Sigma^* \to \Sigma^*$ *tale che* $\forall w \in \Sigma^*$

$$w \in A \Leftrightarrow f(w) \in B$$

La funzione f *è chiamata una riduzione da* A *a* B.



Preappello 15/06/2022: A_TM e HALT_TM

Esercizio 5 (5 punti)

a) **Definire** i linguaggi A_{TM} e $HALT_{TM}$.
b) Descrivere i **problemi decisionali** a cui sono associati i linguaggi A_{TM} e $HALT_{TM}$.
c) Siano A e B due linguaggi. **Definire** cosa significa che $\mathbf{A \leq_m B}$, ovvero che A è **riducibile** mediante funzione a B.

Esercizio bonus*

d) Durante il corso abbiamo visto che $A_{TM} \leq_m HALT_{TM}$. Dimostrare adesso che $\mathbf{HALT_{TM} \leq_m A_{TM}}$.

Lezione 27 pag. 26

Indecidibilità del problema della fermata

Consideriamo la MT F che, sull'input $\langle M, w \rangle$:

1. Costruisce la macchina M',
   M' = "sull'input x
   1. simula M su x
   2. se M accetta, accetta
   3. se M rifiuta, cicla"
2. Fornisce in output $\langle M', w \rangle$

Nota: la macchina M' si ferma su un input x se e solo se M accetta x.

La funzione f calcolata da F, che associa a $\langle M, w \rangle$ la stringa $\langle M', w \rangle$, è una riduzione da A_{TM} a $HALT_{TM}$.

$$\langle M, w \rangle \in A_{TM} \Leftrightarrow M \text{ accetta } w \Leftrightarrow M' \text{ si arresta su } w$$

$$\Leftrightarrow \langle M', w \rangle \in HALT_{TM}.$$

Prossimo tutorato

Prima del secondo appello di luglio:
data da definire, la troverete pianificata su questo canale del Team…
…buono studio
e in bocca al lupo
per l'appello
del 5 luglio ☺