ESERCIZI DI INFORMATICA TEORICA

Maria Silvia Pini e Cinzia Pizzi

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Padova
AA. 2015-2016

1 Automi a Stati Finiti Deterministici (DFA)

Esercizio 1.1

Definire, se esso esiste, l'automa deterministico a stati finiti A che riconosce il linguaggio $L = \{w | w \in \{0,1\}^*, w[i] = 1, \forall i \text{ dispari }, i > 0\}$. Dimostrare rigorosamente che $L = L(A)$ oppure che un tale automa non esiste.

L'automa proposto è riportato in Fig.1.

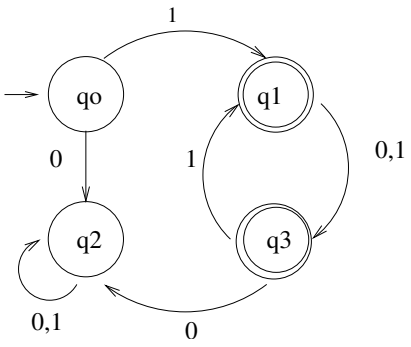

Figure 1: Automa a stati finiti che riconosce L.

Se è possibile costruire l'automa significa che Σ^* è partizionabile in un insieme finito di classi, ciascuna associata a uno stato. È utile descrivere l'insieme delle stringhe associate a ciascuno stato, anche se poi la dimostrazione formale viene richiesta solamente in riferimento agli stati finali. Nel caso specifico q_0 è lo stato iniziale a cui è associata solo la stringa vuota. Agli stati q_1 e q_3 sono associate tutte le stringhe del linguaggio che hanno lunghezza rispettivamente dispari e pari. Allo stato q_2 sono associate tutte le stringhe che non appartengono a L, ad esclusione della stringa vuota.

Per dimostrare che $L = L(A)$ si deve dimostrare che $L \subseteq L(A)$ e $L(A) \subseteq L$.

1. Se $w \in L$ allora $w \in L(A)$ (tutte le stringhe del linguaggio portano in stati finali dell'automa).

   Si procede per induzione sulla lunghezza delle stringhe.

   - Base: $w = 1 \in L$ è la più piccola stringa di L. Verifichiamo che appartenga a $L(A)$. Poiché $\hat{\delta}(q_0, 1) = \delta(q_0, 1) = q_1 \in F$ possiamo concludere che $w = 1 \in L(A)$.
   - Ipotesi induttiva: "se $w \in L$ allora $w \in L(A)$" è vera per tutte le stringhe di lunghezza $|w| \leq n$.
   - Consideriamo ora $w = xa \in L$, con $|w| = n + 1$, e dimostriamo che $w \in L(A)$.
     Poiché $w \in L$ in tutte le posizioni dispari ci sarà necessariamente un simbolo 1. Consideriamo i due casi possibili:
     - $|w|$ pari: questo implica che $|x|$ è dispari, e il simbolo a può assumere indifferentemente valore 0 oppure 1. Per quanto riguarda la struttura di x possiamo dire che tale stringa dovrà necessariamente avere degli 1 nelle posizioni dispari (altrimenti w non potrebbe essere stringa di L). Quindi, poiché $x \in L$ e $|x| \leq n$, possiamo far valere l'ipotesi induttiva e concludere che $x \in L(A)$. In particolare, essendo di lunghezza dispari avremo che $\hat{\delta}(q_0, x) = q_1$. Per definizione di funzione di transizione estesa:

       $$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, xa) = \delta(\hat{\delta}(q_0, x), a) = \delta(q_1, a) = q_3 \in F$$

       Quindi possiamo concludere che $w \in L(A)$.
     - $|w|$ dispari: questo implica che $|x|$ è pari, e il simbolo a, in posizione dispari, deve necessariamente essere 1. Per quanto riguarda la struttura di x possiamo dire che tale stringa

dovrà necessariamente avere degli 1 nelle posizioni dispari (altrimenti w non potrebbe essere stringa di L). Quindi poiché $x \in L$ e $|x| \leq n$ possiamo far valere l'ipotesi induttiva e concludere che $x \in L(A)$. In particolare, essendo di lunghezza pari avremo che $\hat{\delta}(q_0, x) = q_3$. Per definizione di funzione di transizione estesa:

$$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, x1) = \delta(\hat{\delta}(q_0, x), 1) = \delta(q_3, 1) = q_1 \in F$$

Possiamo concludere che $w \in L(A)$.

2. Se $w \in L(A)$ allora $w \in L$ (solo le stringhe del linguaggio portano in stati finali dell'automa).

   Si procede per induzione sulla lunghezza delle stringhe.

   - Base: dal diagramma delle transizioni si vede che il percorso più breve che conduce in uno stato finale dell'automa è etichettato con il simbolo 1. Pertanto $w = 1$ è la più piccola stringa di $L(A)$. Tale stringa ha in tutte le posizioni dispari un 1, pertanto $w = 1 \in L$.
   - Ipotesi induttiva: "se $w \in L(A)$ allora $w \in L$" è vera per tutte le stringhe di lunghezza $|w| \leq n$.
   - Consideriamo ora $w = xa \in L(A)$, con $|w| = n + 1$, e dimostriamo che $w \in L$.
     Poichè $w \in L(A)$, deve essere $\hat{\delta}(q_0, w) \in F$. Ci sono due possibilità.
     - $\hat{\delta}(q_0, w) = q_1$ (caso $|w|$ dispari)
       Gli archi entranti in q_1 sono etichettati solamente con il simbolo 1, quindi possiamo affermare che l'ultimo simbolo di w doveva essere necessariamente 1: $w = x1$.
       Gli stati di provenienza sono q_0 o q_3. Considerando q_0 si ritorna al caso base già esaminato. Quindi lo stato di provenienza è lo stato q_3, stato in cui ci si trova dopo aver letto x. Poiché $\hat{\delta}(q_0, x) = q_3 \in F$, si ha $x \in L(A)$ e $|x| \leq n$. Possiamo far valere l'ipotesi induttiva e concludere che $x \in L$.
       Riguardo la struttura di $w = x1$ possiamo affermare che nella porzione di w coperta da x tutte le posizioni dispari dovranno avere degli 1 ($x \in L$). Inoltre, l'ultima posizione, dispari, contiene anch'essa un 1, rispettando la definizione di L. Pertanto $w \in L$.
     - $\hat{\delta}(q_0, w) = q_3$ (caso $|w|$ pari)
       Gli archi entranti in q_3 provengono solo da q_1 e sono etichettati con 0 o 1. Quindi $w = xa$, con a qualsiasi. Poiché $\hat{\delta}(q_0, x) = q_1 \in F$ e $|x| \leq n$ vale l'ipotesi induttiva e $x \in L$. Possiamo concludere che $w = xa \in L$ poiché il vincolo sulle posizioni dispari è garantito dalla struttura derivata per x.

Si è dimostrato che $L \equiv L(A)$ quindi l'automa proposto è corretto.

Esercizio 1.2

Dimostrare che l'automa A riconosce il linguaggio $L = \{w | w \in \{0,1\}^*, \#1(w) \text{ dispari }\}$.

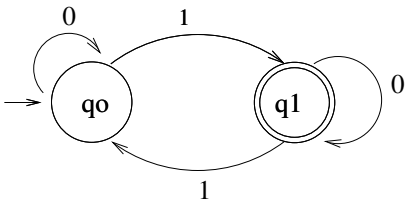

Figure 2: Diagramma delle transizioni dell'automa A.

Anche in questo caso per dimostrare che $L = L(A)$ si devono dimostrare che le due affermazioni

1. $L \subseteq L(A)$
2. $L(A) \subseteq L$

sono vere. Proviamo a procedere come all'esercizio precedente.

1. Se $w \in L$ allora $w \in L(A)$ (tutte le stringhe del linguaggio portano in stati finali dell'automa).

   Si procede per induzione sulla lunghezza delle stringhe.

   - Base: $w = 1 \in L$ è la più piccola stringa di L. Poiché $\hat{\delta}(q_0, 1) = \delta(q_0, 1) = q_1 \in F$ si conclude che $w = 1 \in L(A)$.
   - Ipotesi induttiva: "se $w \in L$ allora $w \in L(A)$" è vera per tutte le stringhe di lunghezza $|w| \leq n$.
   - Sia $w = xa \in L$ ($\#1(w)$ è dispari) e $|w| = n + 1$. Ci sono due casi da considerare:
     - $w = x0$. Poiché $\#1(x) = \#1(w)$ è dispari abbiamo che $x \in L$. Inoltre $|x| \leq n$, quindi vale l'ipotesi induttiva e $x \in L(A)$. In particolare, essendoci un unico stato finale avremo $\hat{\delta}(q_0, x) = q_1$. Quindi:

       $$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, x0) = \delta(\hat{\delta}(q_0, x), 0) = \delta(q_1, 0) = q_1 \in F$$

       e possiamo concludere che $w \in L(A)$.
     - $w = x1$. Poiché $\#1(x) = \#1(w) - 1$ è pari, possiamo concludere che $x \notin L$. A questo punto sappiamo che $|x| \leq n$ e che $x \notin L$. L'ipotesi induttiva ci da informazioni solamente riguardo stringhe di L di lunghezza inferiore o uguale a n ma non dice nulla riguardo a stringhe che non appartengono al linguaggio in esame. Si rimane quindi "bloccati" nella dimostrazione.

In casi come questo è necessario procedere per *mutua induzione*. Si inizia dimostrando che la base è vera per entrambe le affermazioni 1) e 2). Si prosegue supponendo che entrambe le affermazioni siano vere per valori fino a un certo n, e ci conclude dimostrando che entrambe sono ancora vere per stringhe di lunghezza $n + 1$.

1. Se $w \in L$ allora $w \in L(A)$ (tutte le stringhe del linguaggio portano in stati finali dell'automa).
2. Se $w \in L(A)$ allora $w \in L$ (solo le stringhe del linguaggio portano in stati finali dell'automa).

Si procede per mutua induzione sulla lunghezza delle stringhe.

- Base:
  1. $w = 1 \in L$ è la più piccola stringa di L. Poiché $\hat{\delta}(q_0, 1) = \delta(q_0, 1) = q_1 \in F$ possiamo concludere che $w = 1 \in L(A)$.
  2. $w = 1 \in L(A)$ è la più piccola stringa accettata dall'automa. Poiché il numero di 1 di tale stringa è dispari possiamo concludere che $w = 1 \in L$.
- Ipotesi induttiva:
  1. "se $w \in L$ allora $w \in L(A)$" è vera per tutte le stringhe di lunghezza $|w| \leq n$
  2. "se $w \in L(A)$ allora $w \in L$" è vera per tutte le stringhe di lunghezza $|w| \leq n$
- Sia $w = xa$, $|w| = n + 1$.

1. Dimostriamo prima che vale l'affermazione 1: se $w = xa \in L$ allora $w = xa \in L(A)$.
   Se $w = x0 \in L$, si ha $\#1(w)$ dispari e $\#1(x) = \#1(w)$ dispari. Poiché $x \in L$ e $|x| \leq n$, vale l'ipotesi induttiva 1) e possiamo affermare che $x \in L(A)$. Come in precedenza, da $\hat{\delta}(q_0, x) = q_1$ si ottiene:

   $$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, x0) = \delta(\hat{\delta}(q_0, x), 0) = \delta(q_1, 0) = q_1 \in F$$

   Se invece $w = x1 \in L$, si ha $\#1(w)$ dispari ma $1(x) = \#1(w) - 1$ pari. Quindi $x \notin L$ e $|x| \leq n$. Poichè l'ipotesi induttiva 2) afferma che solo le stringhe di L conducono in stati finali dell'automa, la lettura di x deve condurre in uno stato non finale: $\hat{\delta}(q_0, x) = q_0$. Quindi:

   $$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, x1) = \delta(\hat{\delta}(q_0, x), 1) = \delta(q_0, 1) = q_1 \in F$$

2. Dimostriamo ora che vale l'affermazione 2: se $w = xa \in L(A)$ allora $w = xa \in L$.
   Se $w = x0 \in L(A)$, si ha $\hat{\delta}(q_0, w) = q_1$. L'ultimo simbolo letto è uno 0 e l'unico arco entrante in q_1 con etichetta 0 proviene da q_1 stesso. Questo significa che leggendo x a partire da q_0 ci si ritrova nello stato $q_1 \in F$. Poiché $x \in L(A)$ e $|x| \leq n$, vale l'ipotesi induttiva 2 e $x \in L$. Da $w = x0$ ricaviamo che $\#1(w) = \#1(x)$ dispari, da cui $w \in L$.
   Se invece $w = x1 \in L(A)$, si ha ancora $\hat{\delta}(q_0, w) = q_1$. Tuttavia l'ultimo simbolo letto è un 1 e l'unico arco entrante in q_1 con etichetta 1 proviene da q_0. Questo significa che leggendo x a partire da q_0 ci si ritrova nello stato $q_0 \notin F$. Per l'ipotesi induttiva 1) tutte le stringhe di lunghezza minore o uguale a n che appartengono a L devono portare in stati finali. Poiché x non porta in uno stato finale possiamo concludere che $x \notin L$, quindi $\#1(x)$ è pari. Da $w = x1$ ricaviamo che $\#1(w) = \#1(x) + 1$ è dispari, da cui $w \in L$.

Si noti che la dimostrazione può essere suddivisa nei due casi $a = 0$ e $a = 1$ e all'interno di ciascuna situazione dimostrare le due affermazioni 1) e 2). I due procedimenti sono equivalenti.

Esercizi proposti - DFA

1. Definire gli elementi che costituiscono la quintupla che descrive un DFA.
2. Sia $L = \{w \in \{0,1\}^*$ tale che w ha esattamente una sola occorrenza della sottostringa $00\}$
   (a) Disegnare il diagramma di transizione di un DFA che riconosce L.
   (b) Caratterizzare gli stati con una breve descrizione a parole.
   (c) Assumendo di aver già dimostrato che gli stati non accettanti sono caratterizzati dalle stringhe descritte al punto precedente, dimostrare per mutua induzione che:
   (1) se $w \in L$ allora $w \in L(A)$
   (2) se $w \in L(A)$ allora $w \in L$.
   In particolare svolgere entrambi i punti (1) e (2) per la base e l'ipotesi induttiva e dimostrare poi il passo induttivo di (1) se si ha matricola pari, oppure il passo induttivo di (2) se si ha matricola dispari.
3. Sia $L = \{w \in \{0,1\}^*$ tale che il simbolo 0 compare in w solo in blocchi di lunghezza dispari$\}$. Ad esempio, $0 \in L$, $1000110 \in L$, $100011001 \notin L$.
   (a) Disegnare il diagramma di transizione di un DFA che riconosce L con 5 stati (compresi eventuali stati trappola).
   (b) Caratterizzare brevemente a parole gli stati, cioè per ogni stato descrivere le stringhe che portano ad esso.
   (c) Assumendo di aver già dimostrato che gli stati non accettanti sono caratterizzati dalle stringhe descritte al punto precedente, si considerino i seguenti enunciati:
   (1) se $w \in L$ allora $w \in L(A)$
   (2) se $w \in L(A)$ allora $w \in L$.
   Dimostrare entrambi i punti (1) e (2) per la base e l'ipotesi induttiva e SOLO il passo induttivo di (1).
4. (a) Costruire il **diagramma** di transizione per un DFA con 3 stati (compresi gli stati trappola) che riconosce il linguaggio $L = \{w \in \{a,b\}^* : n_a(w) \bmod 3 > 1\}$ e caratterizzare in termini delle stringhe accettate i vari stati.
   (b) Si indichi se $L_1 \subset L_2$, $L_1 \supset L_2$ o $L_1 = L_2$ motivando brevemente la risposta.
   L_1: il linguaggio della CGF con produzioni $S \to 0S1|1S0|\epsilon$
   L_2: il linguaggio delle espressioni regolari $(0+1)^*$.
   (c) Sia A un DFA e $a \in \Sigma$ tale che $\delta(q,a) = q$. Dimostrare per induzione su n che $\forall n \geq 0$, $\hat{\delta}(q,a^n) = q$.
5. (a) Costruire il **diagramma** di transizione per un DFA A che riconosce il linguaggio $L = \{w \in \{0,1,2\}^* : s_w \bmod 2 = 1\}$, dove s_w è la somma del valore corrispondente ai caratteri di w.
   (b) Caratterizzare in termini delle stringhe accettate i vari stati.
   (c) Dimostrare per induzione che se $w \in L$ allora $w \in L(B)$, dove B è il DFA minimo che si ottiene dall'automa A trovato al punto 1.

2 Automi a Stati Finiti Non Deterministici (NFA e ϵ-NFA)

Esercizio 2.1

Dato un NFA con la seguente tabella di transizione:

δ_N	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_1\}$
q_1	$\{q_1, q_2\}$	$\emptyset$
$*q_2$	$\emptyset$	$\{q_2\}$

ottenere un DFA equivalente.

Si utilizza la costruzione lazy evaluation. Si parte dall'unico stato raggiungibile q_0 e si calcola l'insieme degli stati da esso raggiungibili nell'NFA leggendo, rispettivamente, i simboli 0 e 1. L'insieme degli stati ottenuti rappresenta un *unico* stato nel DFA, per il quale andrà calcolata la funzione di trasferimento. Questo unico stato rappresenta il fatto che nell'NFA possiamo trovarci contemporaneamente in ciascuno degli stati coinvolti. Si ricorda che la formula per ottenere lo stato successivo a uno stato $\{q_1, q_2, \ldots, q_r\}$ del DFA, leggendo il simbolo a, è:

$$\delta_D(q_1, q_2, \ldots, q_r, a) = \bigcup_{i=1}^{r} \delta_N(q_i, a)$$

$$\begin{aligned}
\delta_D(\{q_0\}, 0) &= \delta_N(q_0, 0) = \{q_0, q_1\} \\
\delta_D(\{q_0\}, 1) &= \delta_N(q_0, 1) = \{q_1\} \\
\delta_D(\{q_0, q_1\}, 0) &= \delta_N(q_0, 0) \cup \delta_N(q_1, 0) = \{q_0, q_1\} \cup \{q_1, q_2\} = \{q_0, q_1, q_2\} \\
\delta_D(\{q_0, q_1\}, 1) &= \delta_N(q_0, 1) \cup \delta_N(q_1, 1) = \{q_1\} \cup \emptyset = \{q_1\} \\
\delta_D(\{q_1\}, 0) &= \delta_N(q_1, 0) = \{q_1, q_2\} \\
\delta_D(\{q_1\}, 1) &= \delta_N(q_1, 1) = \emptyset
\end{aligned}$$

In modo analogo si ottengono i valori per i rimanenti stati (per i quali si omettono i passaggi intermedi):

$$\begin{aligned}
\delta_D(\{q_0, q_1, q_2\}, 0) &= \{q_0, q_1, q_2\} \\
\delta_D(\{q_0, q_1, q_2\}, 1) &= \{q_1, q_2\} \\
\delta_D(\{q_1, q_2\}, 0) &= \{q_1, q_2\} \\
\delta_D(\{q_1, q_2\}, 1) &= \{q_2\} \\
\delta_D(\{q_2\}, 0) &= \emptyset \\
\delta_D(\{q_2\}, 1) &= \{q_2\}
\end{aligned}$$

L'automa ottenuto è rappresentato in Figura 3.

Esercizio 2.2

Dato l'NFA di Figura 4, enumerare tutti i percorsi seguiti avendo in input la stringa 01001.

Il non determinismo dell'NFA fa sì che più strade possano essere seguite contemporaneamente. Alla lettura del primo 0, l'automa si porterà sia nello stato q che nello stato p. Il primo percorso si svilupperà in modo deterministico attraverso gli stati r, s, s, s in successione. Nell'altro percorso, alla lettura del simbolo 1, si rimane nello stato p. La successiva lettura di uno 0 causa la generazione di due percorsi come nel

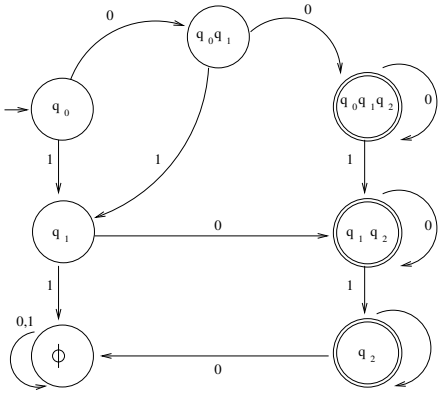

Figure 3: Il DFA equivalente ottenuto con Lazy evaluation.

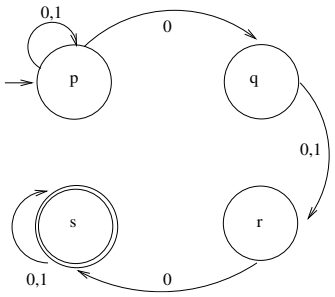

Figure 4: NFA.

caso iniziale. Il primo percorso prosegue in q e da qui, alla lettura dello 0 successivo si porterà in r. A questo punto però l'ultimo simbolo da leggere è un 1 ma non esistono archi uscenti da r con etichetta 1. Il percorso viene abbandonato. L'ultimo percorso attivo ci fa trovare nello stato p dopo aver letto il prefisso 010. La lettura dello 0 successivo crea nuovamente due percorsi. Il primo prosegue nello stato q e alla lettura dell'ultimo simbolo, 1, si sposta nello stato r. Il secondo percorso rimane in p e alla lettura dell'ultimo 1 rimane in questo stato.

Riassumendo, gli stati raggiunti terminando la lettura dell'input sono gli stati s, r, p (vedi Figura 15). Poiché lo stato s è stato finale la stringa 01001 viene accettata.

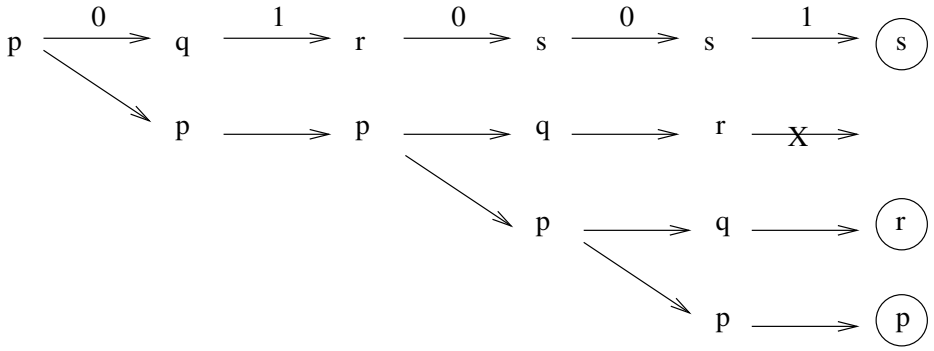

Figure 5: Percorsi seguiti dall'NFA di Figura 4 con input 01001

Esercizio 2.3

Data la tabella di transizione di un ϵ-NFA:

δ	ϵ	a	b	c
$\rightarrow p$	$\emptyset$	$\{p\}$	$\{q\}$	$\{r\}$
q	$\{p\}$	$\{q\}$	$\{r\}$	$\emptyset$
$*r$	$\{q\}$	$\{r\}$	$\emptyset$	$\{p\}$

calcolare la funzione ϵ-close per tutti gli stati.

La ϵ-close si calcola in modo induttivo. In partenza fa parte della ϵ-close di uno stato p, lo stato stesso. Successivamente si includono gli stati raggiungibili da esso con una ϵ-transizione e si itera il procedimento fino a quando non ci sono altri stati da aggiungere. I vari passaggi sono illustrati in Tabella 1.

$ECLOSE(p) = \{p\}$	$\delta(p, \epsilon) = \emptyset$	non ci solo altri stati
$ECLOSE(q) = \{q\}$	$\delta(q, \epsilon) = \{p\}$	$\Rightarrow p \in ECLOSE(q)$
$ECLOSE(q) = \{q, p\}$	$\delta(p, \epsilon) = \emptyset$	non ci solo altri stati
$ECLOSE(r) = \{r\}$	$\delta(r, \epsilon) = \{q\}$	$\Rightarrow q \in ECLOSE(r)$
$ECLOSE(r) = \{r, q\}$	$\delta(q, \epsilon) = \{p\}$	$\Rightarrow p \in ECLOSE(r)$
$ECLOSE(r) = \{r, q, p\}$	$\delta(p, \epsilon) = \emptyset$	non ci solo altri stati

Table 1: Calcolo della epsilon chiusura

Esercizio 2.4

Sia $A = (Q, \Sigma, \delta, q_0, \{q_f\})$ un f-NFA tale che non esista nessuna transizione entrante in q_0 e nessuna uscente da q_f. Descrivere il linguaggio accettato da ognuna delle seguenti varianti di A, in termini di $L = L(A)$.

a) Automa costruito da A aggiungendo una ϵ-transizione da q_f a q_0.

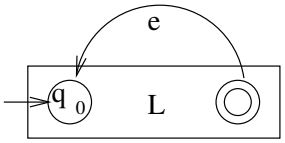

Figure 6: Automa del caso a).

Il linguaggio ottenuto è L^+. Infatti per raggiungere q_f dobbiamo aver letto una stringa $w \in L$. L'aggiunta della ϵ-transizione ci riporta eventualmente allo stato iniziale e ci permette di concatenare altre stringhe di L.

b) 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).

Uno stato può essere raggiunto in uno o più passi. La situazione ottenuta è rappresentata in Figura 17.

Prima si aveva che la stringa $w = xy$ apparteneva a L, ora una qualsiasi stringa y che parte da un punto qualsiasi di w appartiene a L. La stringa y è di fatto un qualsiasi suffisso di $w \in L$. Il linguaggio accettato pertanto è quello di tutti i suffissi (anche impropri) di stringhe di L:

$$\{y \in \Sigma^* | \text{ there is } x \in |\Sigma^* : xy \in L\}$$

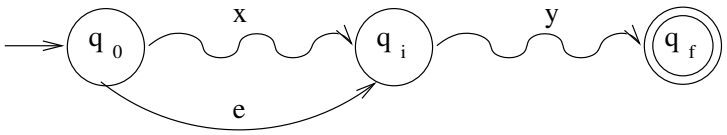

Figure 7: Illustrazione del caso b).

Si faccia attenzione a come l'ipotesi che non ci siano archi entranti in q_0 è cruciale per ottenere solo stringhe di L e loro suffissi. Infatti supponiamo che ci sia un arco entrante in q_0. Questo potrebbe rappresentare l'ultimo step di un ciclo che parte da q_0 e attraverso altri stati dell'automa ritorna allo stato inizile. Sia v la stringa associata a questo percorso. Si veda la Figura 18 come riferimento.

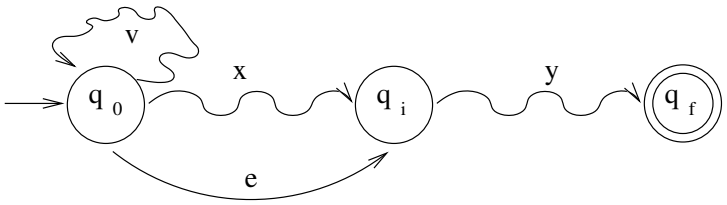

Figure 8: Illustrazione del caso b) se ci fossero archi entranti in q_0.

Ora supponiamo che la stringa $v^*xy \in L$ nell'automa di partenza, ma che $vy \notin L$. Supponiamo inolste che y sia suffisso solo di xy. Aggiungendo una ϵ-transizione da q_0 a q_i si ottiene l'accettazione di vy. Ma vy non è suffisso di nessuna stringa di L.

c) 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).

La situazione ottenuta è rappresentata in Figura 19.

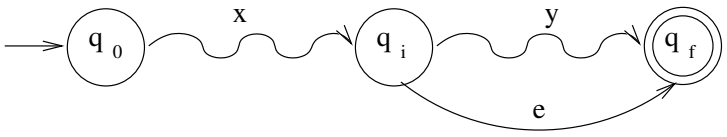

Figure 9: Illustrazione del caso c).

Con un ragionamento duale al precedente si ottiene:

$$\{x \in \Sigma * \mid \text{there is } y \in |\Sigma^* : xy \in L\}$$

d) Automa costruito con le regole dei punti b) e c).

Il linguaggio ottenuto è costituito da suffissi, prefissi e infissi di L.

Esercizi proposti - NFA e ϵ-NFA

1. "Un ϵ-NFA accetta la stringa vuota se e solo se lo stato iniziale è anche stato di accettazione". Commentare la veridicità di questa affermazione: se è vera spiegando a parole perché, se è falsa portando un controesempio.

2. Dato il seguente NFA: N=($\{q_0, q_1\}$,$\{0,1\}$,δ_N, q_0, $\{q_1\}$), dove $\delta_N(q_0, 0) = \{q_0, q_1\}$, $\delta_N(q_0, 1) = \{q_1\}$, $\delta_N(q_1, 0) = \emptyset$, $\delta_N(q_1, 1) = \{q_0, q_1\}$.

   (a) Verificare se le stringhe $w_1 = 101$ e $w_2 = 0010$ vengono o meno accettate dall'automa mostrando tutti i passaggi.

   (b) Dimostrare che dato un NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, con $L(N) = L$ è possibile costruire un DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$, tale che $L(D) = L$.

   (c) Costruire il diagramma di transizione del DFA equivalente all'NFA dato.

   (d) Descrivere a parole il linguaggio accettato dall'automa (suggerimento: è più facile osservando il DFA).

3. (a) Dare la definizione di stati equivalenti.

   (b) Costruire il **diagramma** di transizione dell'automa che minimizza l'automa seguente: ($\{q_0, q_1, q_2, q_3, q_4, q_5\}$ $\{0,1\}$, $\delta, q_0, \{q_3, q_5\}$) dove δ è descritta dalla seguente tabella di transizione:

	0	1
$\rightarrow q_0$	q_1	q_3
q_1	q_0	q_3
q_2	q_1	q_4
$^*q_3$	q_5	q_5
q_4	q_3	q_3
$^*q_5$	q_5	q_5

   Spiegare **sinteticamente** il procedimento seguito.

3 Espressioni e Linguaggi Regolari

Esercizio 3.1

Scrivere l'espressione regolare per i seguenti linguaggi di $\{0,1\}^*$. In generale è possibile costruire un DFA e da questo ricavare l'espressione regolare corrispondente agli stati finali. Tuttavia in questo esercizio si chiede di ricavare l'espressione regolare ragionando sulla struttura delle stringhe che appartengono al linguaggio. Si lascia come esercizio al lettore l'utilizzo dell'altro metodo. Ovviamente si possono scrivere diverse espressioni regolari per rappresentare uno stesso linguaggio, quindi il risultato in generale non è detto che sia identico (potrebbero ad esempio essere necessarie delle semplificazioni per ottenere l'uguaglianza).

1. Tutte le stringhe che terminano in 01.

   L'espressione regolare che rappresenta tutte le stringhe è $(0+1)^*$. Tra queste stringhe si vogliono selezionare solo quelle che terminano in 01. Poiché non ci sono vincoli sul prefisso le stringhe saranno del tipo $w01$, dove $w \in (0+1)^*$. L'espressione regolare risulta quindi:

   $$(0+1)^*01$$

2. Tutte le stringhe che *non* terminano in 01.

   Se le stringhe non terminano in 01, termineranno in 00, 11, 10. Facendo lo stesso ragionamento di prima, e aggiungendo le stringhe di lunghezza minore di 2 si ottiene:

   $$(0+1)^*(00+11+10)+\epsilon+0+1$$

3. Tutte le stringhe con un numero pari di 0.

   L'idea è quella di ripetere un numero arbitrario di volte le stringhe che contengono coppie di zeri. Si consideri la stringa 00. È possibile inserire un numero arbitrario di 1 (eventualmente zero) prima del primo 0, tra i due 0 e dopo il secondo 0 mantenendo la stringa nel linguaggio: $1^*01^*01^*$. Questo tipo di stringa può essere quindi ripetuta un numero arbitrario di volte per rappresentare tutte le stringhe con un numero pari di 0. A questo insieme si deve aggiungere l'insieme delle stringhe composte da soli 1, in quanto in questo caso il numero di 0 è zero.

   $$(1^*01^*01^*)^*+1^*$$

4. Tutte le stringhe con almeno 2 occorrenze di 00.
   La base da cui partire in questo caso è 00 00. Tra le due occorrenze di 00 possiamo avere quasiasi stringa, quindi: $(0+1)^*00(0+1)^*00(0+1)^*$ sono stringhe valide. A queste bisogna aggiungere l'insieme delle stringhe in cui le due occorrenze di 00 si sovrappongono, ossia è presente una sottostringa del tipo 000:

   $$(0+1)^*00(0+1)^*00(0+1)^*+(0+1)^*000(0+1)^*$$

5. Tutte le stringhe con al più 2 occorrenze di 00.

   Anche in questo caso si parte da 00 00, solo che questa volta le stringhe che posso inserire tra le due occorrenze non possono essere qualsiasi. Infatti è necessario evitare di concatenare accidentalmente uno 0 a una delle due occorrenze di 00, altrimente se ne creerebbe una terza. Quindi, prima della

prima occorrenza di 00 si potranno inserire solo stringhe che non contengono 00 e terminano con 1, cioé del tipo $(1+01)^*$. In modo analogo a seguire la prima occorrenze di 00 si potranno mettere solo stringhe che non contengono 00 e che iniziano con 1: $(1+10)^*$. In questo modo però si rischia di avere una stringa che termina con 0 e che, concatenandosi con la seconda occorrenza di 00, creerebbe una terza occorrenza (inserendo una sottostringa del tipo 000). Per evitare questo imponiamo che la seconda occorrenza di 00 sia preceduta da un 1. La seconda occorrenza di 00 può quindi essere seguita solamente da stringhe che non contengono coppie di 0 e che iniziano con 1 (per evitare di concatenare 0 all'occorrenza che le precede). Arrivati a questo punto l'espressione regolare sarebbe: $(1+01)^*00(1+10)^*100(1+10)^*$. Tuttavia è richiesto che ci siano al più 2 occorrenze di 00, quindi potremo anche eliminarne una o entrambe. Questo risultato si ottiene rendendo possibile la scelta di ϵ al posto delle due occorrenze di 00. Infine, non bisogna dimenticare il caso in cui le due occorrenze si sovrappongono, cioé le stringhe che contengono 000 e i casi con 00 o 0. Queste stringhe sono rappresentate da $(1+01)^*(\epsilon+00)(0+\epsilon)(1+10)^*$ Riassumendo otteniamo:

$$(1+01)^*(\epsilon+00)(1+10)^*(\epsilon+100)(1+10)^* + (1+01)^*(\epsilon+00)(0+\epsilon)(1+10)^*$$

6. Tutte le stringhe che non contengono 101.
   La condizione in questo caso è che ogni occorrenza di 10 non sia mai seguita da un 1. Quindi, partendo da 10 dobbiamo far seguire questa stringa da un numero arbitrario di 0 (ma deve essercene almeno uno!). Possiamo anche farla precedere da un eventuale numero arbitrario di 1. Il risultato è 1^*1000^* Questa struttura può essere ripetuta un numero arbitrario di volte. Le stringhe così ottenute cominciano però tutte con 1 e terminano con 0. L'espressione regolare deve quindi essere modificata per tener conto anche di altre possibilità. In particolare possiamo farla precedere da stringhe di 0 di lunghezza arbitraria: $0^*(1^*1000^*)^*$. Ma anche stringhe che terminano con un numero arbitrario di 1 sono lecite: $0^*(1^*1000^*)^*1^*$. Si noti inoltre che le stringhe generate dall'espressione tra parentesi tonde terminano necessariamente con almeno una coppia di zeri. Questo va bene finchè si considerano stringhe che devono essere ripetute, ma alla fine della stringa stessa possiamo ammettere che un eventuale 1 sia seguito da un numero arbitrario (anche zero o uno) di 0. L'espressione diviene per tanto:

   $$0^*(1^*1000^*)^*1^*0^*$$

   Un'altra possibile espressione è $0^*(1^*000^*)^*1^*0^*$, dove l'1 esplicitato in parentesi viene assorbito da 1^*. Di fatto questo permette di esplicitare eventuali stringhe di almeno due 0 consecutivi, che comunque anche nel caso precedente sarebbero state rappresentate attraverso lo 0^* nella parentesi.

Esercizio 3.2

A partire dall'espressione regolare $(ab+a)^*$ costruire un ϵNFA equivalente.

Si ricorda che se $L = L(R)$ per una regexp R, allora esiste un $\epsilon NFAE$ tale che $L(E) = L(R)$ con:

1. esattamente uno stato accettante
2. nessun arco entrante in q_0
3. nessun arco uscente dalla stato finale

Le strutture di base sui simboli ($\epsilon = e$ in questo e negli esercizi successivi) e sugli operatori di unione, concatenzione e chiusura sono:

A partire dalle strutture di base per i simboli $R = a$ e $S = b$ si ottiene l'automa per la concatenazione ab:

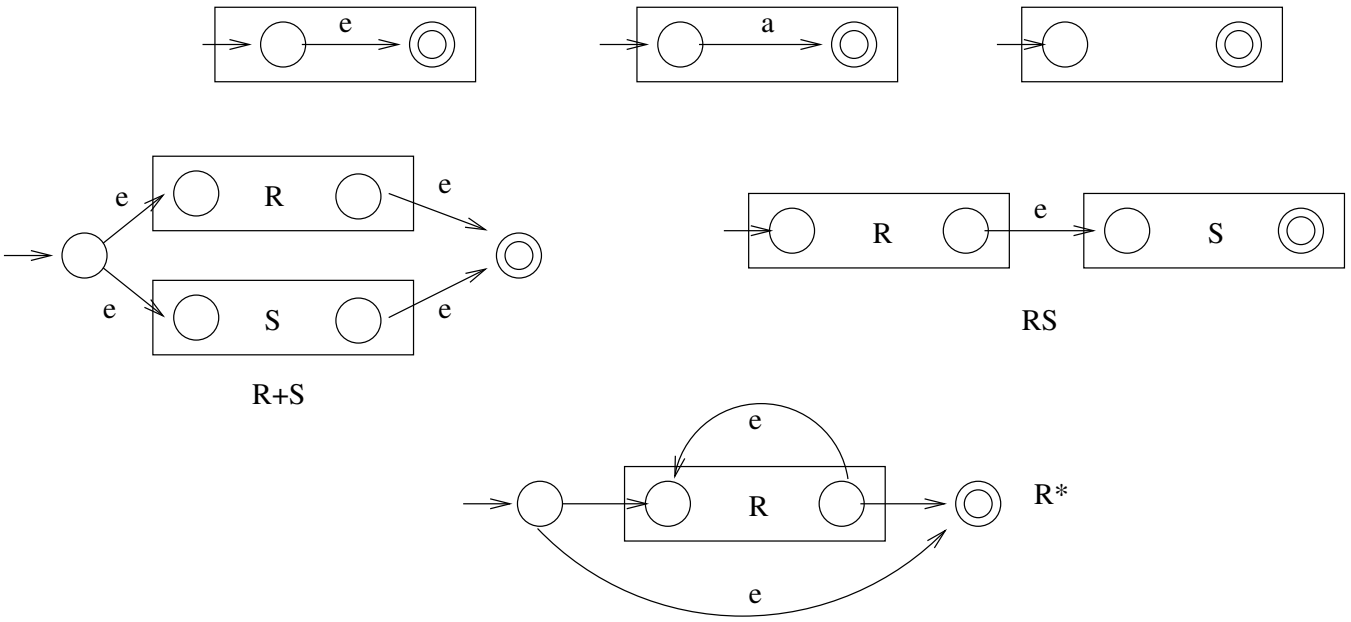

Figure 10: Strutture base di ϵ-NFA per espressioni regolari.

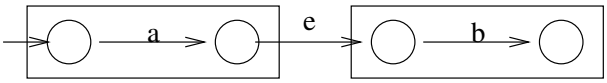

Figure 11: L'ϵ-NFA per ab.

Quest'ultimo viene utilizzato come blocco R per farne l'unione con l'espressione $S = a$:

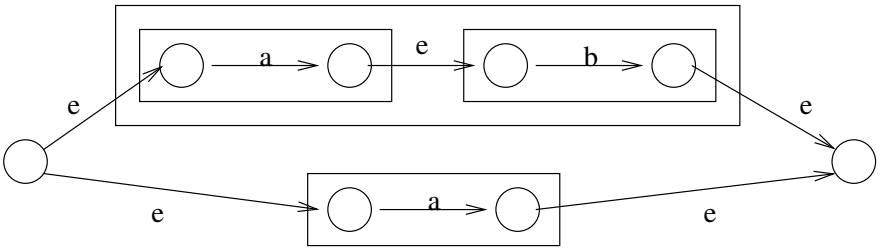

Figure 12: L'ϵ-NFA per $ab + a$.

L'automa risultante rappresenta il blocco R di una chiusura, il che permette di ottenere l'automa corrispondente all'espressione $(ab + a)^*$:

Si potrebbe osservare che è possibile semplificare ciascuna delle operazioni sulle regexp singolarmente, ottenendo un automa corretto (Figura 10).

Tuttavia alcune combinazioni di questi formati possono portare a risultati errati. Ad esempio, si consideri il linguaggio di un'espressione regolare R^*S^*. Le stringhe di questo linguaggio si ottengono concatenando un numero arbitrario di stringhe di R e facendole seguire da un numero arbitrario di stringhe di S. L'automa per R^*S^* viene riportato in Figura 11.

Questo automa riconosce anche stringhe del tipo vxy dove $v \in R$, $x \in S$ e $y \in R$. Queste però sono stringhe che terminano con una stringa del linguaggio R invece che con una del linguaggio S (e in generale i due linguaggi possono contenere stringhe diverse).

Tale stringa verrebbe riconosciuta a partire dallo stato iniziale raggiungendo lo stato finale di R ($v \in R$).

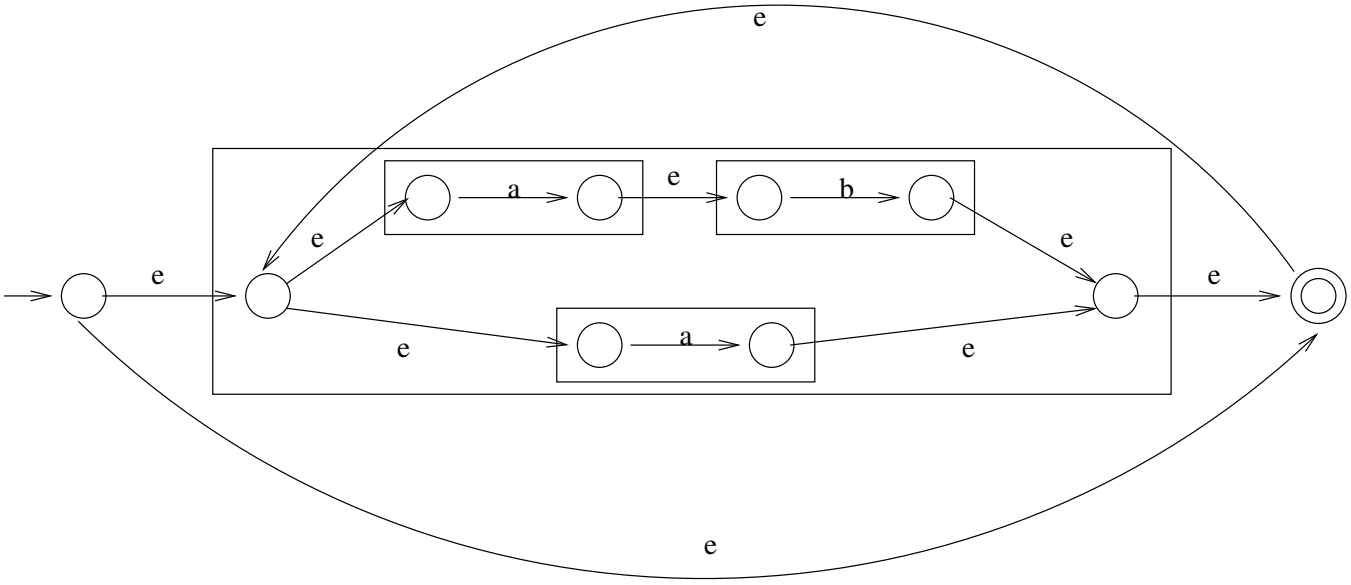

Figure 13: L'ϵ-NFA per $(ab + a)^*$.

Lo stato finale di R coincide con lo stato iniziale di S, quindi leggendo una stringa $x \in S$ ci portiamo nello stato finale di S. Da qui seguendo i due percorsi etichettati con ϵ ci riportiamo nello stato iniziale di R. Una stringa $y \in R$ ci porta nuovamente nello stato finale di R. Da qui si segue l'arco etichettato ϵ e si arriva allo stato finale di S che è anche stato finale dell'automa.

Quindi la stringa vxy viene accettata pur avendo una struttura in $R^*S^*R \neq R^*S^*$.

Esercizio 3.3

Dimostrare che le espressioni regolari $(R + S)^*$ e $(R^*S^*)^*$ sono equivalenti.

Per dimostrare l'equivalenza delle due espressioni, si dimostra che i linguaggi da esse descritti $L_1 = L((R + S)^*)$ e $L_2 = L((R^*S^*)^*)$ coincidono. Si scompone la dimostrazione in due passi: 1) $L_1 \subseteq L_2$ e 2) $L_2 \subseteq L_1$.

1. $L_1 \subseteq L_2$, cioé se $w \in L_1$ allora $w \in L_2$

   Per definizione di linguaggio associato alla chiusura transitiva di un'espressione regolare si ha:

   $$w \in L((R + S)^*) \Rightarrow w \in [L(R + S)]^*$$

   Questo significa che w è costituita dalla concatenazione di zero o più stringhe di $L(R + S)$. Senza perdita di generalità supponiamo sia composta da k stringhe concatenate:

   $$w = w_1 w_2 \dots w_k \text{ con } w_i \in L(R + S)$$

   Per definizione di linguaggio associato all'unione di due espressioni regolari si ottiene:

   $$w_i \in L(R + S) \Rightarrow w_i \in L(R) \cup L(S)$$

   Quindi ciascuna w_i deve appartenere a $L(R)$ o a $L(S)$.

   Supponiamo che $w_i \in L(R)$. A maggior ragione possiamo affermare che $w_i \in [L(R)]^*$. Infatti $[L(R)]^*$ contiene stringhe ottenute dalla concatenzazione di zero o più stringhe di $L(R)$, quindi sicuramente tutte le stringhe di $L(R)$. Per definizione avremo inoltre che $\epsilon \in [L(S)]^*$ e $w_i = w_i\epsilon$.

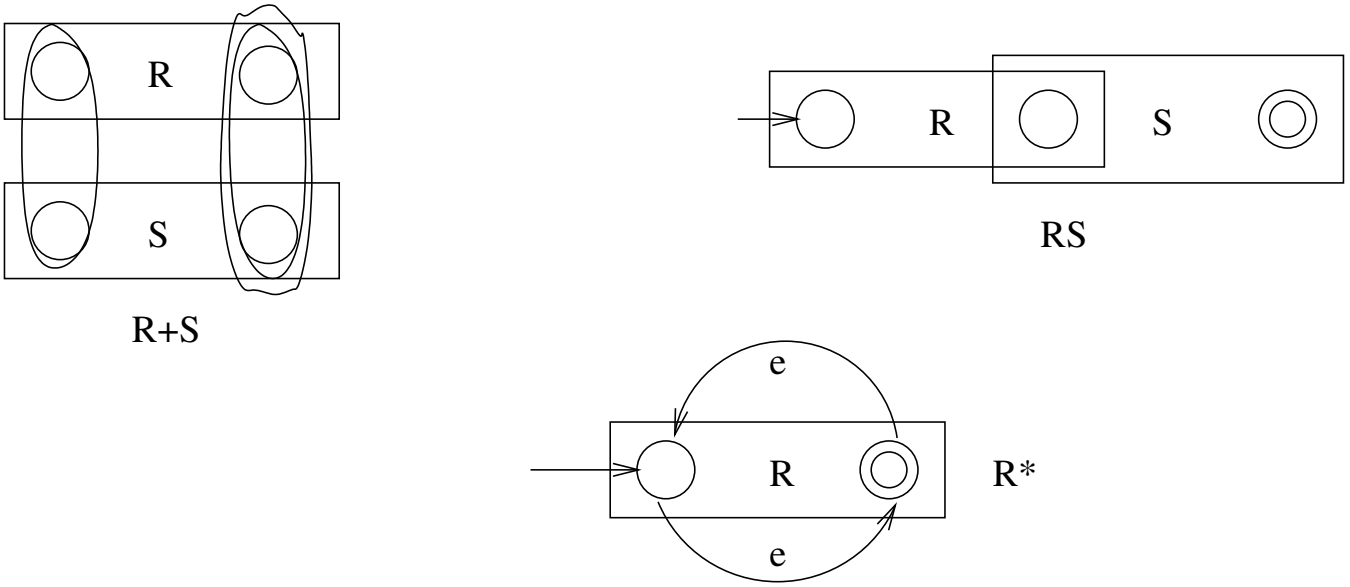

Figure 14: ϵ-NFA alternativi per espressioni regolari di base.

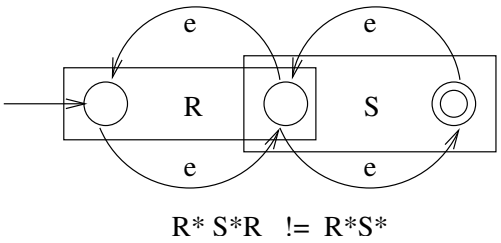

Figure 15: L'ϵ-NFA per R^*S^* risulta errato.

In conclusione, $w_i = w_i\epsilon \in [L(R)]^*[L(S)]^* = L(R^*)L(S^*) = L(R^*S^*)$.

Se invece $w_i \in L(S)$ si ragiona in modo duale tenendo presente che ϵ è elemento neutro per la concatenazione sia a destra che a sinistra: $w_i = \epsilon w_i$.

Abbiamo dimostrato che se $w_i \in L(R + S)$ allora $w_i \in L(R^*S^*)$. Ritornando alla struttura di w questo implica che:

$$w \in [L(R^*S^*)]^* = L[(R^*S^*)^*]$$

2. $L_2 \subseteq L_1$, cioé se $w \in L_2$ allora $w \in L_1$

   Applicando la definizione di linguaggio associato alla chiusura si ottiene:

$$w \in L[(R^*S^*)^*] = [L(R^*S^*)]^* = [L(R^*)L(S^*)]^* = [L(R)^*L(S)^*]^*$$

   Possiamo affermare che w è costituita dalla concatenazione di zero o più stringhe di $L(R)^*L(S)^*$. Senza perdita di generalità assumiamo che sia concatenazione di k stringhe:

$$w = w_1w_2 \dots w_k \text{ con } w_i \in L(R)^*L(S)^*$$

   Questo significa che ciascuna stringa w_i è composta da zero o più stringhe di $L(R)$ seguite da zero o più stringhe di $L(S)$.

Per come è costruita possiamo affermare che a maggior ragione $w_i \in [L(R) \cup L(S)]^*$. Infatti di questo insieme fanno parte tutte le stringhe ottenute per concatenazione di zero o più stringhe prese da $L(R)$ o da $L(S)$, *in ordine qualsiasi*. In particolare quindi ci saranno quelle ottenute concatenando ϵ e stringhe di $L(R)$ seguite da ϵ e stringhe di $L(S)$.

Poiché $w_i \in [L(R) \cup L(S)]^*$ si avrà $w \in \{[L(R) \cup L(S)]^*\}^*$. Per la proprietà di idempotenza $w \in [L(R) \cup L(S)]^*$, per definizione di espressione regolare associata all'unione dei linguaggi di due espressioni, e per definizione di chiusura transitiva si ottiene:

$$w \in [L(R) \cup L(S)]^* = [L(R+S)]^* = L[(R+S)^*]$$

Avendo dimostrato che $L_1 \subseteq L_2$ e $L_2 \subseteq L_1$, possiamo concludere che $L_1 \equiv L_2$.

Esercizio 3.4

Dimostrare attraverso le proprietà di chiusura dei linguaggi regolari se le seguenti affermazioni sono vere o false.

- Se $L = L_1 \cup L_2$ è regolare, allora L_1 è regolare e L_2 è regolare. Falso. Per dimostrarlo basta portare un controesempio.

  Si consideri un qualsiasi linguaggio NON regolare L_1 e si faccia l'unione con $L_2 = \Sigma^*$. Il risultato è sempre Σ^*, linguaggio regolare, ma i due linguaggi di cui si è fatta l'unione non sono entrambi regolari.

- Se $L = L_1$ è non regolare e L_2 è regolare, allora $L_1 \cap L_2$ è sempre regolare. Falso. Per dimostrarlo basta portare un controesempio.

  Si consideri un qualsiasi linguaggio non regolare L_1 e si faccia l'intersezione con $L_2 = \Sigma^*$. Il risultato è L_1, linguaggio non regolare.

Esercizio 3.5

Dimostrare attraverso le proprietà di chiusura dei linguaggi regolari che $L = \{a^n b^m, n \neq m, n, m \geq 0\}$ non è regolare.

Si dimostra per assurdo. Supponiamo che L sia regolare. Per le proprietà di chiusura dei linguaggi regolari anche il complementare $\bar{L}$ dovrà esserlo. Il complementare comprende due sottoinsiemi di stringhe: le stringhe con un certo numero di a seguite da uno stesso numero di b; ma anche tutte le stringhe che non rispettano la struttura a^*b^* (si ricordi che l'unione di un linguaggio e del suo complementare devono dare Σ^*). Pertanto:

$$\bar{L} = \{a^n b^n, n \geq 0\} \cup \{w : w \notin a^*b^*\}$$

Il linguaggio a^*b^* è chiaramente regolare in quanto rappresentato da un'espressione regolare (equivalentemente è banale costruire l'automa a stati finiti che lo riconosce). Per le proprietà di chiusura dei linguaggi regolari l'intersezione di due linguaggi regolari è ancora regolare.

$$\bar{L} \cap a^*b^* = \{a^n b^n, n \geq 0\}$$

Ma il linguaggio $\{a^n b^n, n \geq 0\}$ si può facilmente dimostrare essere non regolare facendo fallire il pumping lemma (provare per esercizio). Quindi $\bar{L} \cap a^*b^* = \{a^n b^n, n \geq 0\}$ **non** è regolare.

Siamo giunti ad una contraddizione, quindi assumere che L fosse regolare è stato un errore. L non è pertanto regolare.

Esercizio 3.6

Stabilire se il linguaggio $L = w \in \{a, b\}^* : w \neq w^R$ dei non-palindromi è regolare. In caso affermativo costruire un DFA A e dimostrare formalmente che L(A) = L. Altrimenti dimostrare formalmente che L non è regolare.

Il linguaggio L non è regolare. Si può dimostrare in due modi diversi: con le proprietà di chiusura e con il pumping lemma.

Supponiamo per assurdo che L sia regolare. Allora per le proprietà di chiusura dei linguaggi regolari anche $\bar{L}$, linguaggio dei palindromi deve esserlo. Ma è dimostrabile facilmente con il pumping lemma che quest'ultimo non è regolare. Ad esempio:

- P_2) fissa n
- P_1) sceglie $w = a^n b a^n$
- P_2) scompone w in xyz, con $|xy| \leq n$ e $|y \neq \epsilon$

  Per come P_1) ha fissato w, la stringa y dovrà necessariamente essere composta da a e appartenere al prefisso di lunghezza n di w.
- P_1) prende $k = 0$: $xz = a^{n-|y|} b a^n$ che non è palindroma.

Quindi non può esserlo nemmeno L.
Dimostrazione diretta di non regolarità di L con il pumping lemma:

- P_2) fissa n
- P_1) sceglie $w = a^n b a^{n+n!}$
- P_2) scompone w in xyz, con $|xy| \leq n$ e $|y \neq \epsilon$

  Per come P_1) ha fissato w, la stringa y dovrà necessariamente essere composta da a e appartenere al prefisso di lunghezza n di w.
- P_1) cerca un k per cui risulti la stringa pompata sia palindroma, cioé tale per cui il numero di a alla sinistra di$b$ sia uguale a $n + n!$ in $xy^k z$. Dovrà verificarsi:

$$n - |y| + k|y| = n + n!$$
$$(k - 1)|y| = n!$$
$$k = 1 + \frac{n!}{|y|}$$

  Poiché $1 \leq |y| \leq n$ a secondo membro ho un numero intero, quindi esiste un k per cui la stringa diventa palindroma, qualsiasi sia la partizione scelta da P_2.

Esercizi proposti - Espressioni e Linguaggi Regolari

1. "Esistono linguaggi accettati da un ϵ-NFA che non sono esprimibili tramite espressioni regolari". Commentare la veridicità di questa affermazione: se è vera portare un esempio, se è falsa motivare a parole la risposta.

2. Dato il seguente DFA: N=($\{q_0, q_1\}$,$\{0,1\}$,δ_N, q_0, $\{q_1\}$), dove $\delta_N(q_0, 0) = \{q_0, q_1\}$, $\delta_N(q_0, 1) = \{q_1\}$, $\delta_N(q_1, 0) = \emptyset$, $\delta_N(q_1, 1) = \{q_0, q_1\}$. Calcolare l'espressione regolare che rappresenta il linguaggio accettato dall'automa.

3. Scrivere l'espressione regolare per il linguaggio su $\Sigma = \{0, 1, 2\}$ in cui le stringhe hanno somma dei numeri che le compongono dispari. Spiegare il ragionamento con cui si è costruita l'espressione.

4. Scrivere l'espressione regolare accettata dall'automa $(\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_0, q_1, q_3\})$ con $\delta(q_0, a) = q_1$, $\delta(q_0, b) = q_2$, $\delta(q_1, a) = q_1$, $\delta(q_2, b) = q_2$ e $\delta(q_2, a) = q_3$.

5. Se E e F sono espressioni regolari $L(E \cup F)$ è il linguaggio di un espressione regolare? Dimostrarlo.

6. Scrivere l'espressione regolare che rappresenta il linguaggio accettato dal seguente NFA: $Q = (\{q_0, q_1, q_2\}$, $\{a, b, c\}$, $\delta, q_0, \{q_2\})$ dove $\delta(q_0, a) = \{q_0\}$, $\delta(q_0, b) = \{q_1\}$, $\delta(q_1, c) = \{q_1\}$, $\delta(q_1, a) = \{q_2\}$, $\delta(q_2, a) = \{q_2\}$, $\delta(q_2, b) = \{q_1\}$.

7. Dimostrare con il Pumping Lemma che il seguente linguaggio non è regolare $L = \{ww^R$, dove w^R è la stringa w rovesciata$\}$.

8. Dimostrare che il linguaggio delle palindromi di lunghezza dispari su alfabeto {0,1} non è regolare.

9. Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) Se L è non regolare e $L' \subseteq L$ allora L' non è regolare (2) Se L_1 è regolare e L_2 è un linguaggio qualsiasi, allora $L_1 \cap L_2$ è regolare?

10. Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non regolare L_1 e di un linguaggio regolare L_2 puo' essere non regolare; (2) L'intersezione di un linguaggio non regolare e di un linguaggio finito è sempre regolare; (3) Ogni sottoinsieme di un linguaggio non regolare è non regolare.

4 Grammatiche e Linguaggi Liberi dal Contesto (CFG e CFL)

Esercizio 4.1

Si descriva una grammatica che generi il linguaggio $L = \{0^n 1^m, m > n\}$.

La grammatica procederà generando inizialmente uno stesso numero di 0 e 1 che andranno a porsi ai lati del simbolo iniziale. Per chiudere la stringa dovrò passare attraverso un'ulteriore variabile A che permetterà di generare solo degli 1: $G = (\{A, S\}, \{0, 1\}, P, S)$, dove P:

$$S \to 0S1|1A$$
$$A \to 1A|\epsilon$$

Esercizio 4.2

Si descriva una grammatica che generi il linguaggio $L = \{a^i b^j c^k, i \neq j\}$.

Idea: faccio generare stringhe del tipo $a^* b^* c^*$ in cui $n_a > n_b$ e n_c qualsiasi e stringhe in cui $n_a < n_b$ e n_c qualsiasi: $S \to AC|BC$, dove A genera stringhe con $n_a > n_b$, B genera stringhe con $n_b > n_a$ e C genera stringhe con un numero qualsiasi di c.

$$A \to aAb|aA|a$$
$$B \to aBb|Bb|b$$
$$C \to cC|\epsilon$$

Esercizio 4.3

Si costruisca una grammatica per il linguaggio $L = w \in \{a, b\}^* : w \neq w^R$ dei non-palindromi. Descrivere brevemente il processo di generazione delle stringhe di L.

Idea: perché sia $w \neq w^R$ è sufficiente che ci sia un simbolo diverso in posizione corrispondente a destra e a sinistra di metà stringa. Quindi è possibile partire generando dei palindromi (questa porzione della stringa è facoltativa), per avvicinarsi alla chiusura della stringa devo passare attraverso una variabile A che imponga caratteri diversi ai suoi lati (quindi alla stessa distanza dall'inizio e dalla fine della stringa). La variabile A poi potrà generare una stringa qualsiasi per chiudere la stringa.

$$S \to aSa|bSb|aAb|bAa$$
$$A \to aA|bA|a|b|\epsilon$$

Esercizio 4.4 (Es. 3 dell'appello del 16/07/2012)

Sia $T' = \{a, b\}$ e sia L il linguaggio dei palindromi con un numero di b multiplo di 4.

1. Si proponga una grammatica libera dal contesto G che generi L, motivandone brevemente la correttezza.
2. Si disegni un albero di derivazione della stringa $w = ababaaababa$ nella grammatica da voi proposta.
3. Si descriva, mediante il diagramma di transizione di stato, un automa a pila che accetti L per stato finale.

1. La variabile S deve generare stringhe in cui il numero di b è uguale a zero oppure è multiplo di 4. Può invece generare un numero arbitrario di a, purché si rispetti la palindromicità della stringa. Si può partire dalla grammatica che genera i palindromi ($P' = \{S' \to aS'a|bS'b|\epsilon|a|b\}$) e modificarla opportunamente. Nella grammatica richiesta posso chiudere la stringa solo con una ϵ (lunghezza pari) oppure con una a (lunghezza dispari). Non posso chiudere con una b altrimenti avrei sempre un numero dispari di b. Quindi sicuramente la nuova grammatica avrà produzioni: $S \to aSa|\epsilon|a$. Vediamo ora come gestire l'inserimento dei simboli b. Nella grammatica richiesta devo garantire che se vengono aggiunte due b, sicuramente prima di chiudere la stringa ne devo aggiungere altre due. Per fare questo introduciamo una nuova variabile T che viene derivata da S con la produzione $S \to bTb$. La variabile T potrà produrre solo coppie di a, oppure, se aggiunge due b dovrà richiamare il simbolo S. In questo modo posso ripetere il procedimento di inserimento di multipli di 4 del simbolo b un numero arbitrario di volte. La grammatica risultante sarà pertanto:

$$G = (\{S, T\}, \{a, b\}, P, S) \;\; dove \;\; P = \{S \to aSa|bTb|\epsilon|a; T \to bSb|aTa\}$$

2. Un albero di derivazione della stringa $w = ababaaababa$ è:

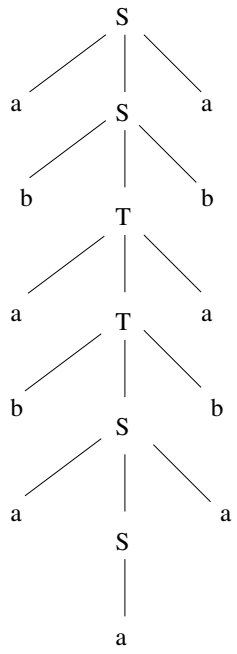

Figure 16: Parse tree per $w = ababaaababa$

3. Il PDA che accetta per stato finale può essere ottenuto in due passi: prima costruisco il PDA che accetta per stack vuoto la grammatica del punto 1; poi trasformo tale PDA in un PDA che accetta per stack finale.

Passo 1. Definisco $P_V = (\{q\}, T', V \cup T', \delta, q, S)$, in cui per ogni produzione del tipo $A \to \beta$ della grammatica, definisco $\delta(q, \epsilon, A) = \{(q, \beta)\}$ e per ogni terminale a della grammatica, definisco $\delta(q, a, a) = \{(q, \epsilon)\}$

Il PDA costruito in questo modo ha δ:

- $\delta(q, \epsilon, S) = \{(q, aSa), (q, bTb), (q, \epsilon), (q, a)\}$;
- $\delta(q, \epsilon, T) = \{(q, bSb), (q, aTa)\}$;
- $\delta(q, a, a) = \{(q, \epsilon)\}$;
- $\delta(q, b, b) = \{(q, \epsilon)\}$;

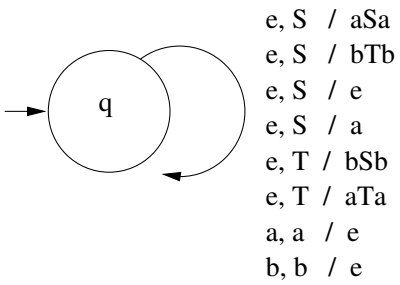

Figure 17: PDA che accetta per stack vuoto

Ora per passare all'accettazione per stato finale introduco due nuovi stati q_0 e q_f e delle ϵ-transizioni che inseriscano un opportuno simbolo di stack all'inizio, e che svuotino lo stack alla fine, rispettivamente. Il PDA definitivo è:

$P_F = (\{q, q_0, q_f\}, \{a, b\}, \{a, b, S, T, X_0\}, \delta_F, q_0, X_0, \{q_f\})$, dove

- $\delta_F(q_0, \epsilon, X_0) = \{(q, SX_0)\}$;
- $\delta_F(q, \epsilon, S) = \{(q, aSa), (q, bTb), (q, \epsilon), (q, a)\}$;
- $\delta_F(q, \epsilon, T) = \{(q, bSb), (q, aTa)\}$;
- $\delta_F(q, a, a) = \{(q, \epsilon)\}$;
- $\delta_F(q, b, b) = \{(q, \epsilon)\}$;
- $\delta_F(q, \epsilon, X_0) = \{(p_f, \epsilon)\}$.

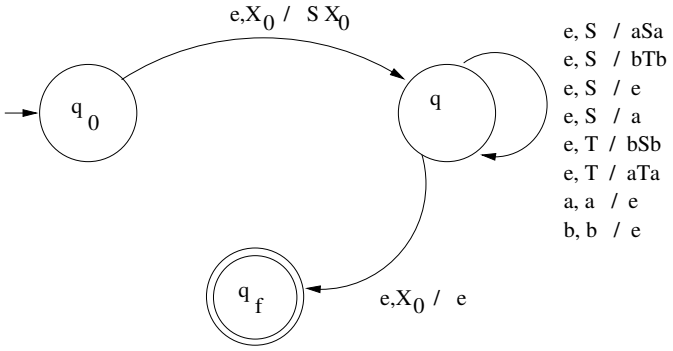

Figure 18: PDA che accetta per stato finale

Esercizio 4.5

Dimostrare che il linguaggio $L = \{0^i 1^j, j = i^2\}$ non è CFL utilizzando il pumping lemma. Utilizziamo il"gioco tra due avversari":

- P_2 fissa n
- P_1 prende una stringa in L di lunghezza almeno n: $z = 0^n 1^{n^2}$
- P_2 scompone la stringa in $z = uvwxy$, dove $|vwx| \leq n$ e $vx \neq \epsilon$

- Per ogni possibile scomposizione P_1 deve trovare un k per cui $uv^kwx^ky \notin L$

  1. vwx si trova nella prima parte della stringa composta da soli 0. Per $k = 0$, il numero di 0 diminuisce ($vx \neq \epsilon$) ad un valore minore di n mentre il numero di 1 rimane invariato. Quindi $uwy \notin L$.
  2. vwx si trova nella seconda parte della stringa composta da soli 1. Si dimostra per $k = 0$ in modo similare.
  3. Se v o x hanno sia 0 che 1, allora per $k = 2$ la struttura della stringa non è più del tipo 0^*1^*, quindi $uv^2wx^2y \notin L$
  4. L'ultimo caso prende in considerazione la possibilità che v consista di soli 0, e x di soli 1 (e siano entrambe diverse dalla stringa vuota). Per un generico k sia ha che il numero di 0 è $n + k|v|$ e il numero di 1 è $n^2 + k|x|$. Per avere una stringa che sia in L deve valere:

     $$(n + k|v|)^2 = n^2 + k|x| \Rightarrow 2nk|v| + k^2|v|^2 = k|x|$$

     Ma a sinistra abbiamo un termine che cresce in modo quadratico in k, mentre a destra cresce in modo lineare, per cui l'uguaglianza non è mai possibile.

  Avendo individuato un k per cui $uv^kwx^ky \notin L$ in corrispondenza di ogni possibile partizione della stringa z il giocatore P_1 dimostra che L non è libero dal contesto.

Esercizi proposti - CFG e CFL

1. Descrivere una context free grammar che genera il linguaggio $L = \{a^i b^j c^k, j \neq k\}$ e descrivere per ogni variabile le stringhe generate.

2. Indicare la grammatica che genera il linguaggio $L = \{w \in \{0,1,2\}^+ | w = x2x', \}, x, x' \in (0+1)^*, x' = x^R\}$. $x' = x^R$ significa che x' è la stringa di x rovesciata, per esempio se $x = 011$ allora $x' = 110$. Descrivere l'insieme delle stringhe accettate dalla variabili della grammatica proposta.

3. Data una grammatica in Forma Normale di Chomsky, quante derivazioni sono necessarie per ottenere una forma sentenziale di lunghezza 9 con 2 variabili e 7 terminali? Giustificare la risposta.

4. Dimostrare con un esempio che la seguente grammatica è ambigua (suggerimento: è sufficiente una stringa di lunghezza 4).
$$S \to aSbS|bSaS|\epsilon$$

5. Indicare i tipi di produzioni di una grammatica in Forma Normale di Chomsky.

6. Ridurre in CNF la grammatica con le seguenti produzioni:
$$S \to aB|bA;\ A \to bAA|aS|a;\ B \to aBB|BS|b$$

7. Indicare la grammatica che genera il linguaggio $L = \{w \in \{0,1,2\}^+ | w = x2x', \}, x, x' \in (0+1)^*, x' = x^R\}$. $x' = x^R$ significa che x' è la stringa di x rovesciata, per esempio se $x = 011$ allora $x' = 110$. Descrivere l'insieme delle stringhe accettate dalla variabili della grammatica proposta.

8. Dimostrare con il Pumping Lemma che il seguente linguaggio non è context-free $L = \{2^n 0^n 1^n, n \geq 1\}$.

9. Dimostrare con il Pumping Lemma che il seguente linguaggio non è context-free $L = \{a^i b^j c^k, k = \max\{i,j\},\ i,j,k \geq 0\}$

10. Commentare la veridicità delle seguenti affermazioni, giustificando la risposta:

    (1) Il linguaggio $L = \{2^i 0^p 1^p, p \geq 1, i \geq 1\}$ è context free.

    (2) $L_1 \subseteq L_2$ e L_1 regolare $\Rightarrow L_2$ è regolare.

11. Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non context-free L_1 e di un linguaggio context-free L_2 può essere non context-free; (2) L'intersezione di un linguaggio non context-free e di un linguaggio finito è context-free.

5 Automi a Pila (PDA)

Esercizio 5.1

1. Costruire un PDA che riconosca il linguaggio $L = \{0^n 1^n, n \geq 1\}$ per stato finale.

   Idea: parto da q_0 e ci rimango fino a quando non leggo un 1. Man mano che leggo uno 0 dall'input, faccio il push di una X nello stack. Quando inizio a leggere 1, cambio stato e in corrispondenza di ogni 1 che leggo faccio il pop di una X dallo stack. Se svuoto lo stack vuol dire che ho letto lo stesso numero di 0 e 1. A questo punto, se accetto per stato finale mi porto in un nuovo stato dove non prevedo ulteriori transizioni (se l'input non è esaurito, tale percorso "muore").

   $P = (\{q_0, q_1, q_2\}, \{0, 1\}, \{X, Z_0\}, \delta, q_0, Z_0, \{q_2\})$

   - Leggo 0, aggiungo X allo stack

   $$\delta(q_0, 0, Z_0) = \{(q_0, XZ_0)\}$$
   $$\delta(q_0, 0, X) = \{(q_0, XX)\}$$

   - Cancello dallo stack X in corrispondenza di un 1 in ingresso

   $$\delta(q_0, 1, X) = \{(q_1, \epsilon)\}$$
   $$\delta(q_1, 1, X) = \{(q_1, \epsilon)\}$$

   - Entro nello stato finale (per accettare non devono esserci ulteriori simboli da leggere, quindi q_2 non ha transizioni definite)

   $$\delta(q_1, \epsilon, Z_0) = \{(q_2, \epsilon)\}$$

2. Descrivere la successione di ID dell'automa alla lettura della stringa 0011

   $(q_0, 0011, Z_0) \vdash (q_0, 011, XZ_0) \vdash (q_0, 11, XXZ_0) \vdash (q_1, 1, XZ_0) \vdash (q_1, \epsilon, Z_0) \vdash (q_2, \epsilon, \epsilon)$

   $w = 0011$ viene accettata.

3. Descrivere la successione di ID dell'automa alla lettura della stringa 0110

   $(q_0, 0110, Z_0) \vdash (q_0, 110, XZ_0) \vdash (q_1, 10, Z_0) \vdash (q_2, 10, \epsilon)$

   Non esisteno transizioni a partire da q_2. Sono in uno stato di accettazione ma non ho esaurito l'input, non accetto. Si noti che se anche avessi avuto una transizione uscente da q_2 il fatto di aver svuotato lo stack mi impedirebbe di continuare il percorso (ho bisogno della tripletta stato-input-stack), pertanto la stringa non verrebbe accettata neanche in questo caso.

Esercizio 5.2

Sia $\Sigma = \{-1, 0, 1\}$ un alfabeto di tre simboli (-1 va considerato come un unico simbolo). Si specifichi la funzione di transizione δ di un PDA $P = (\{q_0, \Sigma, \{Z_0, X, Y\}, \delta, q_0, Z_0)$ che accetti (per pila vuota) tutte e sole le stringhe di Σ^* per le quali la somma dei simboli è nulla. Spiegare brevemente il ruolo svolto dai simboli di pila.

Idea: lo stack tiene conto della somma complessiva calcolata fino a quel punto. Utilizzo due simboli X e Y per indicare un eccesso di -1 o 1, rispettivamente. Z_0 indica che finora ho visto uno stesso numero di 1 e -1 (quindi la somma è zero).

Parto dalla configurazione iniziale e gestisco i vari casi inserendo X se il primo simbolo è -1, oppure Y se il primo simbolo è 1, oppure lasciando invariato lo stack se il primo simbolo è 0.

$$\delta(q_0, -1, Z_0) = \{(q_0, XZ_0)\}$$

$$\delta(q_0, 0, Z_0) = \{(q_0, Z_0)\}$$
$$\delta(q_0, 1, Z_0) = \{(q_0, YZ_0)\}$$

Se leggo -1 e in cima allo stack ho X (che rappresenta -1) aggiungo una X allo stack. Se invece leggo -1 ma in cima allo stack ho una Y (che rappresenta 1) la cancello perchè la somma di questi due simboli è zero.

$$\delta(q_0, -1, X) = \{(q_0, XX)\}$$
$$\delta(q_0, -1, Y) = \{(q_0, \epsilon)\}$$

Se leggo 1 e in cima allo stack ho Y (che rappresenta 1) aggiungo una Y allo stack. Se invece leggo 1 ma in cima allo stack ho una X (che rappresenta -1) la cancello perchè la somma di questi due simboli è zero.

$$\delta(q_0, 1, Y) = \{(q_0, YY)\}$$
$$\delta(q_0, 1, X) = \{(q_0, \epsilon)\}$$

Se leggo 0 lascio invariato lo stack.

$$\delta(q_0, 0, X) = \{(q_0, X)\}$$

$$\delta(q_0, 0, Y) = \{(q_0, Y)\}$$

Se ho in cima allo stack vedo Z_0 significa che la somma di quanto ho letto finora è nulla. Con una ϵ-transizione svuoto lo stack. A questo punto se non ho ulteriori simboli da leggere posso accettare.

$$\delta(q_0, \epsilon, Z_0) = \{(q_0, \epsilon)\}$$

Esercizi proposti - PDA

1. Definire gli elementi che costituiscono la settupla di un PDA.
2. Dare la definizione del linguaggio accettato da un PDA per stato finale.
3. Costruire un PDA che accetta per stato finale il linguaggio $L = \{a^n b^n c^i, n \geq 1, i \geq 1\}$ e spiegarne brevemente il funzionamento.
4. Costruire un PDA che accetta per pila vuota il linguaggio $L = \{c^i a^n b^n, n \geq 1, i \geq 1\}$ e spiegarne brevemente il funzionamento.
5. Costruire un PDA che accetta per pila vuota il linguaggio $L = \{w \in \{0,1,2\}^+ | w = x2x', \}, x, x' \in (0+1)^*, x' = x^R\}$. $x' = x^R$ significa che x' è la stringa di x rovesciata, per esempio se $x = 011$ allora $x' = 110$. Descrivere brevemente **(max mezza pagina)** il funzionamento del PDA. La funzione di transizione del PDA dovrà essere rappresentata mediante il **diagramma** di transizione.

6 Macchine di Turing (TM)

Esercizio 6.1

Progettare una macchina di Turing che riconosca il linguaggio L delle stringhe con un numero uguale di 0 e 1. Si indichi la successione di ID su input 0110.

La macchina di Turing utilizza 6 stati. Lo stato q_5 è stato finale, ed è privo di transizioni così da garantire la fermata della macchina. Lo stato iniziale è q_0. Da qui leggendo il simbolo Blank ci si sposta direttamente nello stato finale, in quanto $\epsilon \in L$. Lo stato q_0 infatti è lo stato che identifica la condizione di equilibrio tra numero di 0 e numero di 1 visti finora. Alla lettura di un simbolo questo viene marcato con una X, indipendentemente dal fatto che sia 0 o 1. Il simbolo X più a sinistra nel nastro indica l'ultimo simbolo già considerato a partire dall'inizio della stringa di input. Per questo motivo se troviamo una Y la sostituiamo con una X.

Se il simbolo letto in q_0 è 0, ci si sposta nello stato q_1. Qui si rimane, spostandosi verso destra, e senza modificare il nastro, fino a quando non si trova un 1. In questo caso ci si sposta in q_2 e si sostituisce l'1 con una Y. Se si incontra un Blank in q_1 la macchina si ferma in uno stato non accettante e rifiuta l'input (di fatto ho più 0 che 1). Lo stato q_2 viene utilizzato per tornare indietro nel nastro fino all'ultimo simbolo considerato, lasciando invariato il contenuto delle altre celle. Quando troviamo una X torniamo in q_0 e iteriamo il procedimento.

Gli stati q_3 e q_4 svolgono un ruolo analogo a quello di q_1 e q_2 nel caso in cui il simbolo che sbilancia l'equilibrio sia un 1 e non uno 0.

	0	1	X	Y	B
q_0	(q_1, X, R)	(q_3, X, R)		(q_0, X, R)	(q_5, B, R)
q_1	$(q_1, 0, R)$	(q_2, Y, L)		(q_1, Y, R)	
q_2	$(q_2, 0, L)$	$(q_2, 1, L)$	(q_0, X, R)	(q_2, Y, L)	
q_3	(q_4, Y, L)	$(q_3, 1, R)$		(q_3, Y, R)	
q_4	$(q_4, 0, L)$	$(q_4, 1, L)$	(q_0, X, R)	(q_4, Y, L)	
q_5					

Table 2: Tabella di transizione per una TM che riconosce il linguaggio delle stringhe binarie con un eguale numero di 0 e 1.

Se l'input è la stringa $w = 0110$, la sequenza di ID risulta essere:

$$q_0 0110 \vdash X q_1 110 \vdash q_2 XY10 \vdash X q_0 Y10 \vdash XX q_0 10 \vdash XXX q_3 0 \vdash$$
$$\vdash XX q_4 XY \vdash XXX q_0 Y \vdash XXXX q_0 B \vdash XXXXB q_5$$

Esercizio 6.2

Progettare una macchina di Turing per ognuno dei seguenti linguaggi.

1. Punto a
   $L = \{w \in \{0,1\}^* : \#0 = \#1\}$

   Un prima strategia per risolvere questo esercizio è stata proposta nella soluzione dell'esercizio precedente. Un'altra possibile strategia prevede di leggere la stringa nel punto in cui attualmente si trova la testina, marcare il simbolo corrente (che sarà 0 o 1) e cercare il simbolo complementare successivo (1 se all'inizio ho letto zero e viceversa), proseguendo finchè non ci sono più simboli da marcare nella stringa. Se ne rimangono la TM non deve accettare.

   Una possibile soluzione è la TM

   $$M = (\{q_0, q_1, q_2, q_3, q_f\}, \{0,1\}, \{0,1,X,B\}, \delta, q_0, B, \{q_f\})$$

con δ definita nel diagramma delle transizioni in Figura 19.

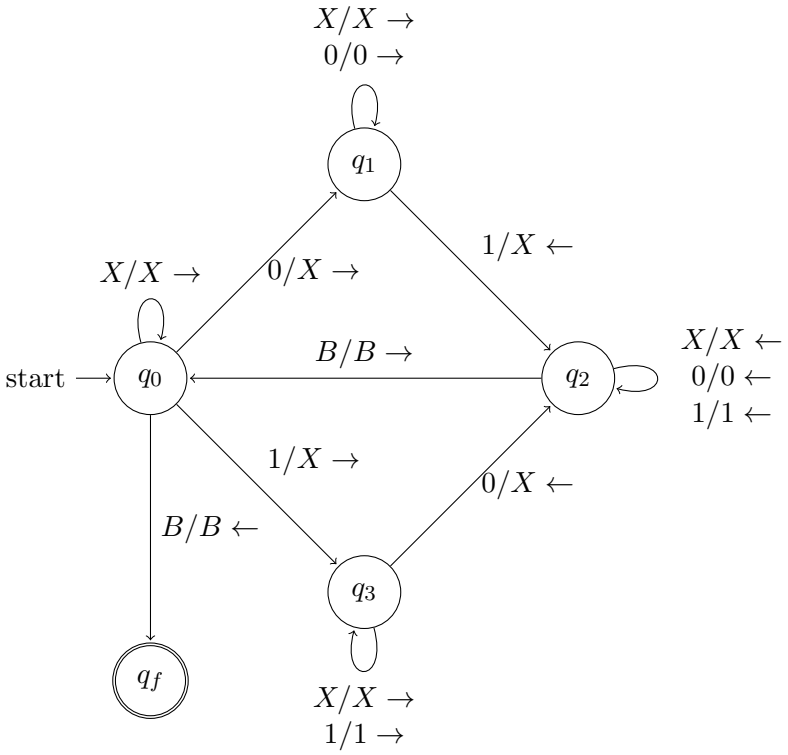

Figure 19: Diagramma delle transizioni della TM del punto a.

Il ciclo $q_0 - q_1 - q_2 - q_0$ quando trova 0 lo marca e cerca il corrispondente 1, tornando poi a inizio stringa. Allo stesso modo il ciclo $q_0 - q_3 - q_2 - q_0$, quando trova un 1, cancella uno 0 e torna all'inizio. La transizione $X/X \rightarrow$ permette di scorrere gli X iniziali, e se riesce ad arrivare a un B significa che ad ogni 0 è corrisposto un 1 e viceversa, accettando quindi la stringa in input.

2. Punto b
   $L = \{a^n b^n c^n : n \geq 1\}$.

   Data una stringa, la strategia proposta in questa soluzione prevede di

   (a) cancellare la prima a ($q_0 \rightarrow q_a$)
   (b) saltare le rimanenti a (q_a) e cancellare la prima b ($q_a \rightarrow q_b$)
   (c) saltare le rimanenti b (q_b) e cancellare la prima c ($q_b \rightarrow q_c$)
   (d) saltare le rimanenti c (q_c)
   (e) riavvolgere la stringa (q_r) tornando all'inizio ($q_r \rightarrow q_0$) e ripetere il ciclo.

   Ognuno degli stati q_0, q_a, q_b, q_r deve essere in grado di scorrere il simbolo di nastro X, non presente tra i simboli di input ed usato per cancellare gli input letti. Si consideri una stringa di L, ad esempio *aaabbbccc*. Gli stati finora specificati modificano la stringa *aaabbbccc* in $XaaXbbXcc$, $XXaXXbXXc$, fino a $XXXXXXXXX$, con la macchina che si trova in q_0 e la testina a inizio stringa. Nel caso in cui ci siano a, b o c fuori posto la macchina muore senza accettare, dato che le transizioni gestiscono solo le stringhe dove tutte le a precedono tutte le b, che a loro volta precedono tutte le c.

   In q_0 c'è bisogno di una transizione $X/X \rightarrow$ che scorra le a in testa alla stringa già cancellate. Questa permette anche, quando tutte le a sono state cancellate, di portare la testina nel $Blank$ che si trova a fine stringa. Ciò avviene solo nel caso in cui tutta la stringa sia costituita da X, situazione che avviene solo se le lettere in input sono presenti in egual numero, portando la macchina in q_1.

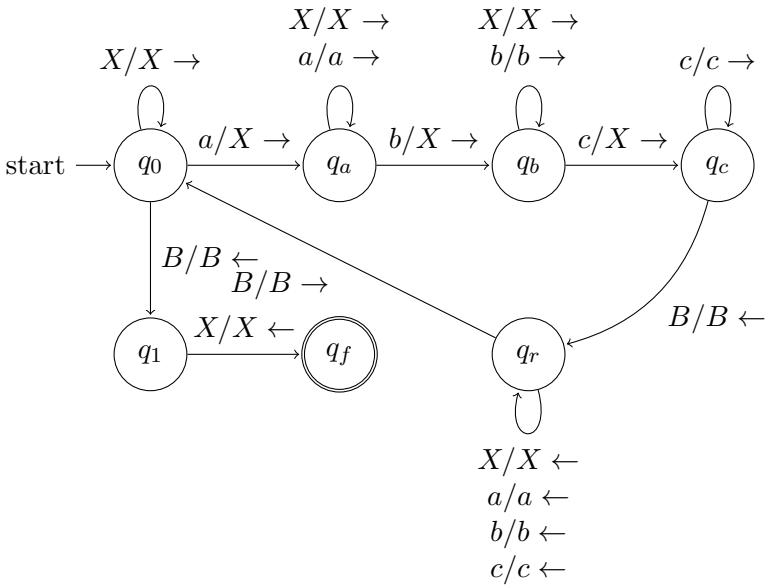

Figure 20: Diagramma delle transizioni per la TM del punto b.

Non siamo ancora nella condizione di accettare, in quanto potremmo aver ricevuto in input la stringa vuota. Se l'ultimo carattere della stringa era una X significa che la stringa non era vuota, la TM va in q_f e accetta, altrimenti la stringa era vuota e muore senza accettare.

Una TM che riconosce L utilizzando la strategia precedente, fermandosi in ogni caso, è quindi

$$M = (\{q_0, q_a, q_b, q_c, q_r, q_1, q_f\}, \{a, b, c\}, \{a, b, c, X, B\}, \delta, q_0, B, \{q_f\})$$

dove δ è specificata dal diagramma delle transizioni in Figura 20.

Si noti che semplificando la TM, togliendo q_r e riavvolgendo la stringa subito in q_c (Figura 21) non si ottiene la TM desiderata, in quanto non si effettua mai il controllo che non ci sia il $Blank$ dopo le c e ottenendo così una TM che accetta stringhe del linguaggio L^+ (ad esempio $aabbccabc$ viene trasformata in $XXXXXXXabc$, q_0 salta poi le X iniziali e infine cancella anche la parte finale accettando la stringa).

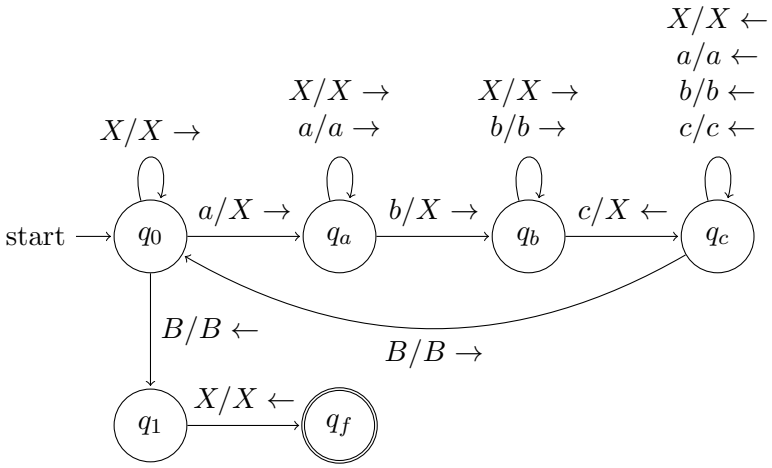

Figure 21: Soluzione sbagliata

3. Punto c
   $L = \{ww^R : w \in \{0,1\}^*\}$.

Una possibile strategia per una TM che riconosce L è la seguente:

(a) leggo e cancello il carattere in testa alla stringa

(b) vado in coda alla stringa (se il primo carattere dopo quello appena cancellato è un B significa che la stringa ha lunghezza dispari e quindi non è nella forma voluta)

(c) cancello il carattere corrispondente a quello letto in testa (la TM muore se è diverso) e torno a inizio stringa

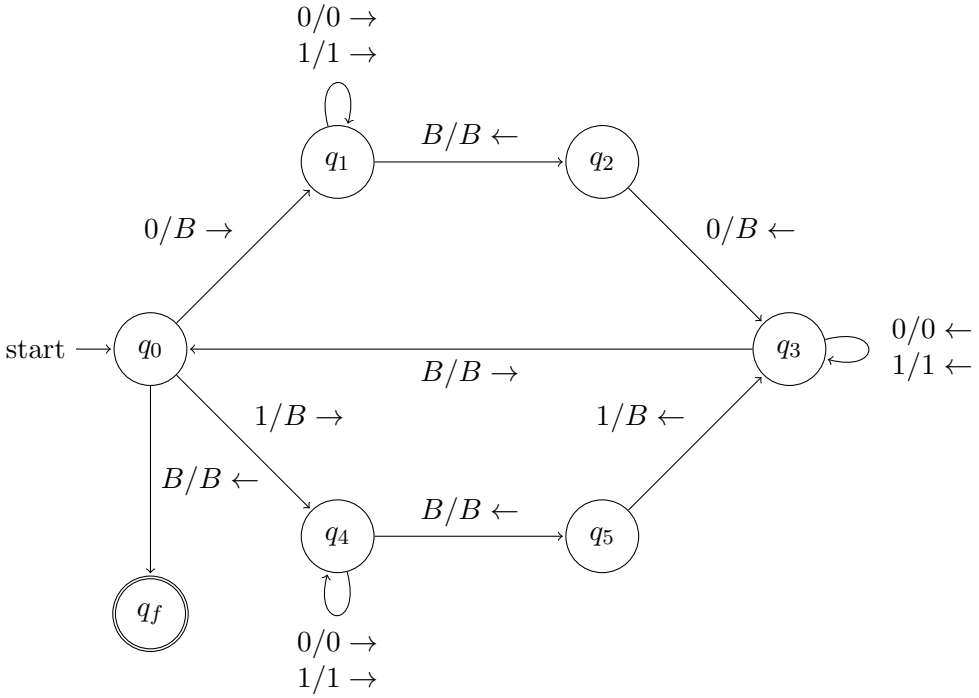

Figure 22: Diagramma delle transizioni della TM del punto c.

Nel diagramma in Figura 22, il ciclo $q_0 - q_1 - q_2 - q_3 - q_0$ per ogni 0 in testa alla stringa cancella uno 0 in coda, mentre $q_0 - q_4 - q_5 - q_3 - q_0$ per ogni 1 in testa cancella un 1 in coda. Quando in testa c'è un B, l'input è stato esaurito correttamente e quindi la TM accetta la stringa.

Esercizi proposti - TM

1. Dare la definizione di linguaggio accettato da una TM M=(Q, Σ, Γ, δ, q_0, B, F).
2. Dare la definizione della funzione di transizione δ di una Turing Machine (TM) M=(Q, Σ, Γ, δ, q_0, B, F).
3. La Turing Machine $M = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, B\}, \delta, q_0, B, \{q_2\})$ ha le seguenti transizioni e nessun'altra:

   $$\delta(q_0, 0) = (q_1, 1, R) \quad \delta(q_1, 1) = (q_2, 0, L) \quad \delta(q_2, 1) = (q_0, 1, R)$$

   Specificare la sequenza di ID su input "0100".
4. Costruire una TM per il linguaggio $L = \{a^n b^{2n}, n \geq 1\}$ e spiegare brevemente a parole il ragionamento seguito nella costruzione. NB: Ci sono diverse soluzioni possibili, si può (ma non è necessario) assumere che a sinistra dell'input ci sia un Blank.
5. Siano L e $\bar{L}$ due linguaggi complementari. Dire quali fra le seguenti situazioni sono impossibili giustificando la risposta:
   (a) L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile ma non ricorsivo
   (b) L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile
   (c) L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile, $L \cap \bar{L}$ non è ricorsivo
6. Costruire una Turing Machine (rappresentando la funzione di transizione mediante il **diagramma** di transizione) che accetta il linguaggio delle stringhe su $\{a, b, c\}$ tali che $\#a = \#b = \#c$ e spiegare brevemente a parole il ragionamento seguito nella costruzione.

Temi d'esame 2013

Informatica Teorica	**I Compitino**
Pini-Pizzi	**03 Maggio 2013**

Si svolgano, anche in parte, i seguenti problemi. Si prega di scrivere la soluzione a ciascun problema in un foglio protocollo separato e di intestare ciascun foglio con

COGNOME, Nome, No. Matricola, Canale

dove Canale $\in$ {Pini, Pizzi}.

1. Problema 1 [6 punti]
   (a) Definire gli elementi che costituiscono la quintupla che descrive un DFA.
   (b) "Esistono linguaggi accettati da un ϵ-NFA che non sono esprimibili tramite espressioni regolari". Commentare la veridicità di questa affermazione: se è vera portare un esempio, se è falsa motivare a parole la risposta.
   (c) "Un ϵ-NFA accetta la stringa vuota se e solo se lo stato iniziale è anche stato di accettazione". Commentare la veridicità di questa affermazione: se è vera spiegando a parole perché, se è falsa portando un controesempio.
2. Problema 2 [6 punti]
   (a) Dimostrare con il Pumping Lemma che il seguente linguaggio non è regolare $L = \{ww^R,$ dove w^R è la stringa w rovesciata$\}$.
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non regolare L_1 e di un linguaggio regolare L_2 puo' essere non regolare; (2) L'intersezione di un linguaggio non regolare e di un linguaggio finito è sempre regolare; (3) Ogni sottoinsieme di un linguaggio non regolare è non regolare.
3. Problema 3 [8 punti]
   Dato il seguente NFA: N=($\{q_0, q_1\}$,$\{0,1\}$,δ_N, q_0, $\{q_1\}$), dove $\delta_N(q_0, 0) = \{q_0, q_1\}$, $\delta_N(q_0, 1) = \{q_1\}$, $\delta_N(q_1, 0) = \emptyset$, $\delta_N(q_1, 1) = \{q_0, q_1\}$.
   (a) Verificare se le stringhe $w_1 = 101$ e $w_2 = 0010$ vengono o meno accettate dall'automa mostrando tutti i passaggi.
   (b) Dimostrare che dato un NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, con $L(N) = L$ è possibile costruire un DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$, tale che $L(D) = L$.
   (c) Costruire il diagramma di transizione del DFA equivalente all'NFA dato.
   (d) Descrivere a parole il linguaggio accettato dall'automa (suggerimento: è più facile osservando il DFA).
4. Problema 4 [6 punti]
   (a) Dato il seguente DFA: D=($\{q_1, q_2\}$,$\{0,1\}$,δ, q_1, $\{q_2\}$), dove $\delta(q_1, 0) = q_1$, $\delta(q_1, 1) = q_2$, $\delta(q_2, 0) = q_2$, $\delta_N(q_2, 1) = q_1$. Calcolare l'espressione R^2_{12}.
   (b) Scrivere l'espressione regolare per il linguaggio su $\Sigma = \{0, 1, 2\}$ in cui le stringhe hanno somma dei numeri che le compongono dispari. Spiegare il ragionamento con cui si è costruita l'espressione.
5. Problema 5 [8 punti].
   Sia $L = \{w \in \{0,1\}^*$ tale che w ha esattamente una sola occorrenza della sottostringa 00$\}$
   (a) Disegnare il diagramma di transizione di un DFA che riconosce L.
   (b) Caratterizzare gli stati con una breve descrizione a parole.
   (c) Assumendo di aver già dimostrato che gli stati non accettanti sono caratterizzati dalle stringhe descritte al punto precedente, dimostrare per mutua induzione che:
   (1) se $w \in L$ allora $w \in L(A)$
   (2) se $w \in L(A)$ allora $w \in L$.
   In particolare svolgere entrambi i punti (1) e (2) per la base e l'ipotesi induttiva e dimostrare poi il passo induttivo di (1) se si ha matricola pari, oppure il passo induttivo di (2) se si ha matricola dispari.

Informatica Teorica	**II Compitino**
Pini-Pizzi	**24 Giugno 2013**

Cognome: ..... Nome: .....

n. Matricola: ..... Canale (Pini o Pizzi): .....

Esercizio 1	Esercizio 2	Esercizio 3	Esercizio 4	Esercizio 5	Voto Complessivo

Intestare ciascun foglio protocollo con: **COGNOME, Nome, n. Matricola, Canale**

1. Problema 1 [6 punti]
   (a) Definire gli elementi che costituiscono la settupla di un PDA.
   (b) Dare la definizione del linguaggio accettato da un PDA per stato finale.
2. Problema 2 [7 punti]
   (a) Dimostrare con il Pumping Lemma che il seguente linguaggio non è context-free $L = \{2^n 0^n 1^n, n \geq 1\}$.
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non context-free L_1 e di un linguaggio context-free L_2 può essere non context-free; (2) L'intersezione di un linguaggio non context-free e di un linguaggio finito è context-free.
3. Problema 3 [6 punti].
   (a) Data una grammatica in Forma Normale di Chomsky, quante derivazioni sono necessarie per ottenere una forma sentenziale di lunghezza 9 con 2 variabili e 7 terminali? Giustificare la risposta.
   (b) Dimostrare con un esempio che la seguente grammatica è ambigua (suggerimento: è sufficiente una stringa di lunghezza 4).
   $$S \rightarrow aSbS|bSaS|\epsilon$$
4. Problema 4 [8 punti]
   (a) Costruire un PDA che accetta per stato finale il linguaggio $L = \{a^n b^n c^i, n \geq 1, i \geq 1\}$ e spiegarne brevemente il funzionamento.
   (b) Descrivere una context free grammar che genera il linguaggio $L = \{a^i b^j c^k, j \neq k\}$ e descrivere per ogni variabile le stringhe generate.
5. Problema 5 [7 punti]
   (a) La Turing Machine $M = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_2\})$ ha le seguenti transizioni e nessun'altra:
   $$\delta(q_0, 0) = (q_1, 1, R) \quad \delta(q_1, 1) = (q_2, 0, L) \quad \delta(q_2, 1) = (q_0, 1, R)$$
   Specificare la sequenza di ID su input "0100".
   (b) Costruire una TM per il linguaggio $L = \{a^n b^{2n}, n \geq 1\}$ e spiegare brevemente a parole il ragionamento seguito nella costruzione. NB: Ci sono diverse soluzioni possibili, si può (ma non è necessario) assumere che a sinistra dell'input ci sia un Blank.

Informatica Teorica	**I Appello**
Pini-Pizzi	**24 Giugno 2013**

Cognome: ..... Nome: .....

n. Matricola: ..... Canale (Pini o Pizzi): .....

Esercizio 1	Esercizio 2	Esercizio 3	Esercizio 4	Esercizio 5	Voto Complessivo

Intestare ciascun foglio protocollo con: **COGNOME, Nome, n. Matricola, Canale**

1. Problema 1 [8 punti].
   Sia $L = \{w \in \{0,1\}^*$ tale che il simbolo 0 compare in w solo in blocchi di lunghezza dispari$\}$. Ad esempio, $0 \in L$, $1000110 \in L$, $100011001 \notin L$.
   (a) Disegnare il diagramma di transizione di un DFA che riconosce L con 5 stati (compresi eventuali stati trappola).
   (b) Caratterizzare brevemente a parole gli stati, cioè per ogni stato descrivere le stringhe che portano ad esso.
   (c) Assumendo di aver già dimostrato che gli stati non accettanti sono caratterizzati dalle stringhe descritte al punto precedente, si considerino i seguenti enunciati:
   (1) se $w \in L$ allora $w \in L(A)$
   (2) se $w \in L(A)$ allora $w \in L$.
   Dimostrare entrambi i punti (1) e (2) per la base e l'ipotesi induttiva e SOLO il passo induttivo di (1).
2. Problema 2 [7 punti]
   (a) Dimostrare con il Pumping Lemma che il seguente linguaggio non è context-free $L = \{2^n 0^n 1^n, n \geq 1\}$.
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non context-free L_1 e di un linguaggio context-free L_2 puo' essere non context-free; (2) L'intersezione di un linguaggio non context-free e di un linguaggio finito è context-free.
3. Problema 3 [4 punti]
   (a) Scrivere la formula di ricorsione R_{ij}^k per il calcolo di un'espressione regolare a partire da un DFA.
   (b) Spiegare cosa rappresentano le varie componenti della formula.
4. Problema 4 [8 punti]
   (a) Costruire un PDA che accetta per stato finale il linguaggio $L = \{a^n b^n c^i, n \geq 1, i \geq 1\}$ e spiegarne brevemente il funzionamento.
   (b) Descrivere una context free grammar che genera il linguaggio $L = \{a^i b^j c^k, j \neq k\}$ e descrivere per ogni variabile le stringhe generate.
5. Problema 5 [7 punti]
   (a) La Turing Machine $M = (\{q_0, q_1, q_2\}, \{0,1\}, \{0,1,B\}, \delta, q_0, B, \{q_2\})$ ha le seguenti transizioni e nessun'altra:

   $$\delta(q_0, 0) = (q_1, 1, R) \quad \delta(q_1, 1) = (q_2, 0, L) \quad \delta(q_2, 1) = (q_0, 1, R)$$

   Specificare la sequenza di ID su input "0100".
   (b) Costruire una TM per il linguaggio $L = \{a^n b^{2n}, n \geq 1\}$ e spiegare brevemente a parole il ragionamento seguito nella costruzione. NB: Ci sono diverse soluzioni possibili, si può (ma non è necessario) assumere che a sinistra dell'input ci sia un Blank.

Informatica Teorica	**II Appello**
Pini-Pizzi	**10 Luglio 2013**

Cognome: ..... Nome: .....

n. Matricola: ..... Canale (Pini o Pizzi): .....

Esercizio 1	Esercizio 2	Esercizio 3	Esercizio 4	Esercizio 5	Voto Complessivo

Intestare ciascun foglio protocollo con: **COGNOME, Nome, n. Matricola, Canale**

1. Problema 1 [8 punti].
   (a) Costruire il **diagramma** di transizione per un DFA con 3 stati (compresi gli stati trappola) che riconosce il linguaggio $L = \{w \in \{a,b\}^* : n_a(w) \bmod 3 > 1\}$ e caratterizzare in termini delle stringhe accettate i vari stati.
   (b) Si indichi se $L_1 \subset L_2$, $L_1 \supset L_2$ o $L_1 = L_2$ motivando brevemente la risposta.
   L_1: il linguaggio della CGF con produzioni $S \to 0S1|1S0|\epsilon$
   L_2: il linguaggio delle espressioni regolari $(0+1)^*$.
   (c) Sia A un DFA e $a \in \Sigma$ tale che $\delta(q,a) = q$. Dimostrare per induzione su n che $\forall n \geq 0$, $\hat{\delta}(q,a^n) = q$.
2. Problema 2 [6 punti].
   (a) Dare la definizione di stati equivalenti.
   (b) Costruire il **diagramma** di transizione dell'automa che minimizza l'automa seguente: $(\{q_0,q_1,q_2,q_3,q_4,q_5\}, \{0,1\}, \delta, q_0, \{q_3,q_5\})$ dove δ è descritta dalla seguente tabella di transizione:

	0	1
$\to q_0$	q_1	q_3
q_1	q_0	q_3
q_2	q_1	q_4
$^*q_3$	q_5	q_5
q_4	q_3	q_3
$^*q_5$	q_5	q_5

   Spiegare **sinteticamente** il procedimento seguito.
   (c) Scrivere l'espressione regolare accettata dall'automa $(\{q_0,q_1,q_2,q_3\}, \{a,b\}, \delta, q_0, \{q_0,q_1,q_3\})$ con $\delta(q_0,a) = q_1$, $\delta(q_0,b) = q_2$, $\delta(q_1,a) = q_1$, $\delta(q_2,b) = q_2$ e $\delta(q_2,a) = q_3$.
3. Problema 3 [6 punti]
   (a) Dimostrare con il Pumping Lemma che il seguente linguaggio non è context-free $L = \{a^i b^j c^k, k = \max\{i,j\},\ i,j,k \geq 0\}$
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta:
   (1) Il linguaggio $L = \{2^i 0^p 1^p, p \geq 1, i \geq 1\}$ è context free.
   (2) $L_1 \subseteq L_2$ e L_1 regolare $\Rightarrow L_2$ è regolare.
4. Problema 4 [8 punti]
   (a) Indicare i tipi di produzioni di una grammatica in Forma Normale di Chomsky.
   (b) Ridurre in CNF la grammatica con le seguenti produzioni:

$$S \to aB|bA;\ A \to bAA|aS|a;\ B \to aBB|BS|b$$

(c) Costruire un PDA che accetta per pila vuota il linguaggio $L = \{c^i a^n b^n, n \geq 1, i \geq 1\}$ e spiegarne brevemente il funzionamento.

5. Problema 5 [6 punti]

(a) Dare la definizione di linguaggio accettato da una TM M=(Q, Σ, Γ, δ, q_0, B, F).

(b) Siano L e $\bar{L}$ due linguaggi complementari. Dire quali fra le seguenti situazioni sono impossibili giustificando la risposta:

i. L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile ma non ricorsivo

ii. L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile

iii. L è ricorsivamente enumerabile, $\bar{L}$ è ricorsivamente enumerabile, $L \cap \bar{L}$ non è ricorsivo

Informatica Teorica	**III Appello**
Pini-Pizzi	**9 Settembre 2013**

Cognome: ..... Nome: .....

n. Matricola: ..... Canale (Pini o Pizzi): .....

Esercizio 1	Esercizio 2	Esercizio 3	Esercizio 4	Esercizio 5	Voto Complessivo

Intestare ciascun foglio protocollo con: **COGNOME, Nome, n. Matricola, Canale**

1. Problema 1 [8 punti].
   (a) Costruire il **diagramma** di transizione per un DFA A che riconosce il linguaggio $L = \{w \in \{0,1,2\}^* : s_w \bmod 2 = 1\}$, dove s_w è la somma dei valori corrispondenti ai caratteri di w.
   (b) Caratterizzare in termini delle stringhe accettate i vari stati.
   (c) Dimostrare per induzione che se $w \in L$ allora $w \in L(B)$, dove B è il DFA minimo che si ottiene dall'automa A trovato al punto (a).
2. Problema 2 [6 punti].
   (a) Se L e M sono linguaggi regolari, allora $L \cup M$ è regolare? Dimostrare la risposta.
   (b) Scrivere l'espressione regolare che rappresenta il linguaggio accettato dall'NFA sequente $Q = (\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, q_0, \{q_2\})$ dove $\delta(q_0, a) = \{q_0\}$, $\delta(q_0, b) = \{q_1\}$, $\delta(q_1, c) = \{q_1\}$, $\delta(q_1, a) = \{q_2\}$, $\delta(q_2, a) = \{q_2\}$, $\delta(q_2, b) = \{q_1\}$.
3. Problema 3 [6 punti]
   (a) Dimostrare che il linguaggio delle palindromi di lunghezza dispari su alfabeto {0,1} non è regolare.
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) Se L è non regolare e $L' \subseteq L$ allora L' non è regolare (2) Se L_1 è regolare e L_2 è un linguaggio qualsiasi, allora $L_1 \cap L_2$ è regolare?
4. Problema 4 [8 punti]
   (a) Indicare la grammatica che genera il linguaggio $L = \{w \in \{0,1,2\}^+ | w = x2x', \}, x, x' \in (0+1)^*, x' = x^R\}$. $x' = x^R$ significa che x' è la stringa di x rovesciata, per esempio se $x = 011$ allora $x' = 110$. Descrivere l'insieme delle stringhe accettate dalle variabili della grammatica proposta.
   (b) Costruire un PDA che accetta per pila vuota il linguaggio generato dalla grammatica proposta e descriverne brevemente **(max mezza pagina)** il funzionamento. La funzione di transizione del PDA dovrà essere rappresentata mediante il **diagramma** di transizione.
5. Problema 5 [6 punti]
   (a) Dare la definizione della funzione di transizione δ di una Turing Machine M=(Q, Σ, Γ, δ, q_0, B, F).
   (b) Costruire una Turing Machine (rappresentando la funzione di transizione mediante il **diagramma** di transizione) che accetta il linguaggio delle stringhe su $\{a, b, c\}$ tali che $\#a = \#b = \#c$ e spiegare brevemente a parole il ragionamento seguito nella costruzione.

Informatica Teorica	**IV Appello**
Pini-Pizzi	**4 Febbraio 2014**

Cognome: ..... Nome: .....

n. Matricola: ..... Canale (Pini o Pizzi): .....

Esercizio 1	Esercizio 2	Esercizio 3	Esercizio 4	Esercizio 5	Voto Complessivo

Intestare ciascun foglio protocollo con: **COGNOME, Nome, n. Matricola, Canale**

1. Problema 1 [8 punti].
   Sia $L = \{w | w \in \{0,1\}^*, w[i] \text{ è dispari per ogni } i \text{ pari}, i > 0\}$.
   (a) Disegnare il diagramma di transizione del DFA **minimo** che riconosce L.
   (b) Caratterizzare brevemente a parole gli stati, cioè per ogni stato descrivere le stringhe che portano ad esso.
   (c) Assumendo di aver già dimostrato che gli stati non accettanti sono caratterizzati dalle stringhe descritte al punto precedente, si considerino i seguenti enunciati:
   (1) se $w \in L$ allora $w \in L(A)$
   (2) se $w \in L(A)$ allora $w \in L$.
   Dimostrare entrambi i punti (1) e (2) per la base e l'ipotesi induttiva e SOLO il passo induttivo di (1).

2. Problema 2 [6 punti]
   (a) Dimostrare con il Pumping Lemma che il seguente linguaggio non è regolare: $L = \{a^n b^m c^{n-m}\}$.
   (b) Commentare la veridicità delle seguenti affermazioni, giustificando la risposta: (1) L'intersezione di un linguaggio non context-free L_1 e di un linguaggio regolare L_2 puo' essere non context-free; (2) L'intersezione di un linguaggio non context-free e di un linguaggio finito è context-free.

3. Problema 3 [6 punti]
   (a) Dimostrare che se L e M sono linguaggi regolari, allora anche i linguaggi $L \cup M$ e $L - M$ sono regolari.
   (b) Scrivere l'espressione regolare per il linguaggio $L = \{$stringhe di a e b che iniziano per b e finiscono con $aaa\}$. Spiegare il ragionamento con cui si è costruita l'espressione.

4. Problema 4 [8 punti]
   (a) Costruire un PDA che accetta per stato finale il linguaggio $L = \{a^n b^m c^{n-m}, n > m \geq 0\}$ e spiegarne brevemente il funzionamento. Non costruire il PDA a partire dalla grammatica che genera L.
   (b) Descrivere una context free grammar che genera il linguaggio $L = \{a^n b^n c^{n+1}, n \geq 0\}$ e descrivere per ogni variabile le stringhe generate.

5. Problema 5 [8 punti]
   (a) Dare le definizioni di linguaggio ricorsivo e ricorsivamente enumerabile.
   (b) Se L e $\bar{L}$ (il complementare di L) sono ricorsivamente enumerabili allora L che tipo di linguaggio è? Dimostrare formalmente la risposta.
   (c) Disegnare il diagramma di transizione di una TM che accetta il linguaggio $L = \{a^n b^{2k} a^n, b, k \geq 0\}$. e spiegare brevemente a parole il ragionamento seguito nella costruzione.