Corso di Laurea in Sicurezza dei sistemi e delle reti informatiche

Fondamenti di informatica per la sicurezza

anno accademico 2003–2004

docente: Stefano Ferrari

Esempio di compitino — 13.01.2004 — SOLUZIONI

valutazioni **1** (4) ____ **2** (5) ____ **3** (4) ____ **4** (7) ____ **5** (7) ____ **6** (7) ____

Cognome _____

Nome _____

Matricola _____ **Firma** _____

Esercizio 1

Siano dati i linguaggi L_1 e L_2:

- $L_1 = \{are, ere, ire\}$
- $L_2 = \{rid, rad, rod\}$

Descrivere i linguaggi:

a) $L_3 = L_1 \cup L_2$

b) $L_4 = L_1 L_2$

c) $L_5 = {L_1}^4$

d) $L_6 = L_1 {L_2}^*$

e) $L_7 = (L_1 L_2)^*$

f) $L_8 = {L_1}^* {L_2}^*$

Per quegli insiemi di cui sia troppo lungo (o impossibile) dare una descrizione estensionale, elencare almeno tre elementi, indicando le caratteristiche degli elementi che li compongono.

Soluzione

a) $L_3 = L_1 \cup L_2 = \{are, ere, ire, rid, rad, rod\}$

b) $L_4 = L_1 L_2 = \{arerid, arerad, arerod, ererid, ererad, ererod, irerid, irerad, irerod\}$

c) $L_5 = {L_1}^4$
L'insieme L_5 è formato dalle stringhe che si ottengono concatenando quattrostringhe appartenenti a L_1. Poiché L_1 ha 3 elementi, possono essere formate $3^4 = 81$ stringhe come concatenazione di quattro elementi di L_1. L'insieme $\{areareareare, erereareireare, ireireareere\}$ è un sottoinsieme di L_5.

d) $L_6 = L_1 {L_2}^*$
L'insieme L_6 è formato da stringhe che hanno un elemento di L_1 come prefisso e una concatenzazione di un numero arbitrario (eventualmente nullo) di elementi di L_2 come suffisso. Poiché ${L_2}.^*$ è composto da infiniti elementi, anche L_6 avrà infiniti elementi. L'insieme $\{are, ereridrodrod, ireriadradrid\}$ è un sottoinsieme di L_5.

e) $L_7 = (L_1 L_2)^*$
L'insieme L_7 è equivalente a ${L_4}^*$. Pertanto, anche L_7 è composto da infiniti elementi. L'insieme $\{\epsilon, areraderid, ererodererodereradirerid\}$ è un sottoinsieme di L_7.

f) $L_8 = {L_1}^* {L_2}^*$
Gli elementi dell'insieme L_8 sono il risultato della concatenazione di una stringa di ${L_1}^*$ con una stringa di ${L_2}^*$. Poiché sia ${L_1}^*$ che ${L_2}^*$ hanno infiniti elementi, anche L_8 avrà cardinalità infinita. L'insieme $\{\epsilon, areere, rid, irerodrad\}$ è un sottoinsieme di L_8.

Esercizio 2

Sia data la seguente grammatica, $G = \langle T, V, P, S\rangle$, definita su $A = \{a, b, c\}$:

- insieme dei simboli terminali, T: $T = A$

- insieme dei metasimboli, V: $V = \{K, H\}$
- insieme delle regole di produzione, P: $P = \{S ::= K,\ K ::= a|Hb,\ H ::= c|Kb|Ha\}$

Quali fra le seguenti stringhe vengono generate da G?

a) cb

b) $cbbab$

c) $aaaaa$

d) aab

Riportare la successione di regole da applicare per la generazione di tali stringhe e le stringhe parziali ottenute.

Soluzione

a)

cb	
	S
$S ::= K$	K
$K ::= Hb$	Hb
$H ::= c$	cb

La stringa cb è generata da G: $cb \in L(G)$.

b)

$cbbab$	
	S
$S ::= K$	K
$K ::= Hb$	Hb
$H ::= Ha$	Hab
$H ::= Kb$	$Kbab$
$K ::= Hb$	$Hbbab$
$H ::= c$	$cbbab$

La stringa $cbbab$ è generata da G: $cbbab \in L(G)$.

c)

$aaaaa$	
	S
$S ::= K$	K
$K ::= a$	a

Non è possibile aggiungere altri simboli alla stringa generata, in quanto non ci sono altri metasimboli nella stringa parziale. La stringa $aaaaa$ non è quindi generata da G: $aaaaa \notin L(G)$.

d)

aab	
	S
$S ::= K$	K
$K ::= Hb$	Hb
$H ::= Ha$	Hab
$H ::= Ha$	$Haab$

Non è possibile ottenere la stringa aab senza metasimboli. La stringa aab non è quindi generata da G: $aab \notin L(G)$.

Esercizio 3

Sia dato il seguente automa a stati finiti, A, $A = \langle Q, \Sigma, \delta, q_0, F\rangle$:

- insieme degli stati, Q: $Q = \{q_0, q_1, q_2, q_3\}$
- alfabeto di input, Σ: $\Sigma = \{a, b, c, d, e\}$
- funzione di transizione δ:

	a	b	c	d	e
q_0	q_0	q_0	q_2	q_1	q_1
q_1	q_1	q_3	q_1	q_1	q_1
q_2	q_3	q_2	q_2	q_0	q_1
q_3	q_0	q_1	q_1	q_0	q_1

- stato iniziale, q_0
- insieme di stati finali, F: $F = \{q_1\}$

Indicare:

a) cinque stringhe accettate da A

b) cinque stringhe rifiutate da A

Soluzione

Per dare risposta ai quesiti di questo problema non è necessario tracciare la rappresentazione grafica dell'automa in questione, anche se aiuta.

Ricordiamo brevemente il funzionamento dell'automa a stati finiti. In ogni istante, l'automa si trova in uno degli stati dell'insieme Q e legge un simbolo di input. In seguito alla lettura del simbolo, la funzione di transizione, dato lo stato corrente ed il simbolo letto, determina lo stato nel quale l'automa si sposterà. Una stringa viene accettatta se, una volta letti tutti i simboli che la compongono, l'automa si trova in uno degli stati dell'insieme F.

Nel caso dell'automa in esame, quindi, verranno accettate tutte e sole le stringhe che portanto l'automa dallo stato q_0 allo stato q_1, indipendentemente dagli stati visitati durante la scansione della stringa.

Per esempio, le seguenti stringhe sono accettate:

- $bcabe$:

stato corrente	q_0	q_0	q_2	q_3	q_1	q_1
simbolo letto	b	c	a	b	e	

- *abddec*:

stato corrente	q_0	q_0	q_0	q_1	q_1	q_1
simbolo letto	a	b	d	d	e	c

- *cdbee*:

stato corrente	q_0	q_2	q_0	q_0	q_1
simbolo letto	c	d	b	e	e

- *aaacde*:

stato corrente	q_0	q_0	q_0	q_0	q_2	q_0
simbolo letto	a	a	a	c	d	e

- *deddc*:

stato corrente	q_0	q_1	q_1	q_1	q_1
simbolo letto	d	e	d	d	c

Le seguenti stringhe sono invece rifiutate:

- *cebacc*:

stato corrente	q_0	q_2	q_1	q_3	q_0	q_2
simbolo letto	c	e	b	a	c	c

- *baddab*:

stato corrente	q_0	q_0	q_0	q_1	q_1	q_1
simbolo letto	b	a	d	d	a	b

- *baaaaa*:

stato corrente	q_0	q_0	q_0	q_0	q_0	q_0
simbolo letto	b	a	a	a	a	a

- *dedab*:

stato corrente	q_0	q_1	q_1	q_1	q_1
simbolo letto	d	e	d	a	b

- *babcb*:

stato corrente	q_0	q_0	q_0	q_0	q_2
simbolo letto	b	a	b	c	b

Esercizio 4

Costruire un automa a stati finiti con le seguenti caratteristiche:

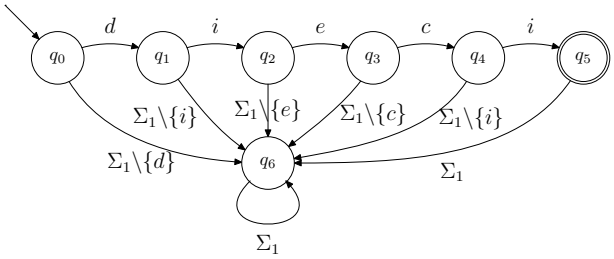
- ha come alfabeto di input l'insieme dei caratteri alfabetici maiuscoli e minuscoli
- accetta le stringhe che formano le parole *dieci*, *novemila* e *diecimila*, senza distinguere tra lettere maiuscole e minuscole (*Dieci* e *dIECimIla* sono accettate).

Soluzione

L'alfabeto di input dell'automa è $\Sigma = \{a, b, \ldots, z, A, B, \ldots, Z\}$.

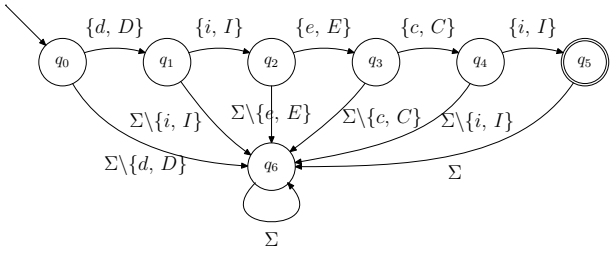
Bisogna notare che le parole che l'automa deve riconoscere sono tra loro correlate: *dieci* è una sottostringa di *diecimila* e *mila* è sottostringa sia di *diecimila* che di *novemila*. Ciò consente un approccio modulare nella costruzione dell'automa.

Cominciamo con il costruire un automa che accetti in input solo lettere minuscole: l'alfabeto sarà quindi $\Sigma_1 = \{a, b, \ldots, z\}$. Limitiamo inizialmente anche le parole accettate: l'automa A_1 dovrà rifiutare tutte le sequenze di simboli tranne la parola *dieci*. Un automa del genere è facilmente costruibile:



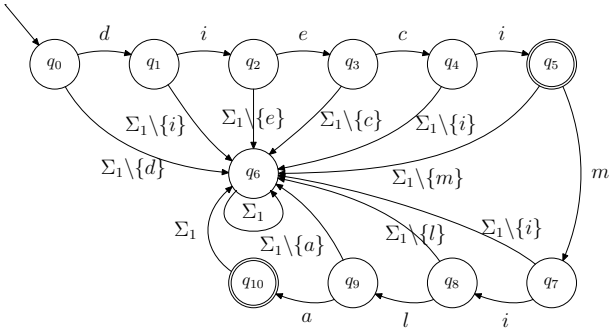
L'automa in figura usa i cinque stati $q1$–$q5$ per scandire la stringa *dieci*, costringendosi nello stato $q6$ (che una volta raggiunto non può più essere lasciato) se la stringa di input non è *dieci*.

Estendiamolo ora per far sì che accetti l'alfabeto di ingresso Σ e che non faccia differenza fra maiuscole e minuscole:



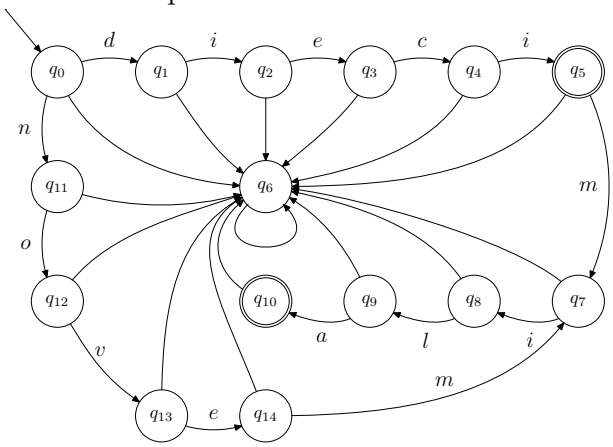
Come si può notare, la struttura ed il criterio di funzionamento sono esattamente gli stessi dell'automa precedente. Per semplificare la rappresentazione, quindi non considereremo le lettere maiuscole nel seguito.

È facile a questo punto costruire l'automa che accetti le parole *dieci* e *diecimila*. Basta estendere il primo automa in modo che scandisca la stringa *mila* dopo aver scandito la stringa *dieci*:



L'estensione per la stringa *novemila* è relativamente facile. Bisogna però notare che la stringa *nove* non deve essere accettata. Questo effetto può essere ottenuto aggiungendo gli stati necessari a scndire la stringa *nove* e concatenandoli con quelli della stringa *mila*, giaà presenti, avendo l'accortezza di non aggiungere altri stati

finali. Per maggiore chiarezza, non sono state etichettate le transizioni che portano allo stato q_6. Esse devono essere percorse per qualsiasi simbolo non presente sulle altre transizioni.



Esercizio 5

Sia data l'espressione regolare E, definita su $\Sigma = \{a, b, c\}$:

- $E = (a^*b^2)^*(b + abc^*)^*$

Quali fra le seguenti stringhe vengono descritte da E?

a) $aabc$

b) bb

c) abc

d) bac

e) $abbabbbabc$

Soluzione

L'insieme di stringhe denotato da un'espressione regolare è definito tramite una regola ricorsiva. In particolare, l'espressione regolare E descrive le stringhe che possono essere ottenute concatenando le stringhe descritte dalle espressioni $E_1 = (a^*b^2)^*$ e $E_2 = (b + abc^*)^*$. Le stringhe descritte da E_1 sono ottenute concatenando un numero non limitato (anche zero) di stringhe composte da una successione di a di qualsiasi lunghezza (anche zero) con due b. Le stringhe descritte da E_2 sono ottenute concatenando un numero non limitato (anche zero) di stringhe appartenenti all'insieme denotato da $E_3 = b + abc^*$. Qest'ultima espressione regolare denota l'insieme formato dalla stringa b e dalle stringhe composte concatenando la stringa ab con un numero qualsiasi (anche zero) di c. In altri termini:

- $\{\epsilon, bb, abb, bbabb, abbbb, aaabbbbaabb\} \subset L(E_1)$
- $\{\epsilon, b, abc, ababccc, abcbbabcc, bbbabccab\} \subset L(E_2)$

a) $s_1 = aabc$
Questa stringa non viene descritta da E ($aabc \notin L(E)$). Se il simbolo b di s_1 venisse descritto dalla sottoespressione E_1, infatti, dovrebbe trovarsi in coppia (b^2). Se b fosse descritto da E_2 potrebbe appartenere alla sottoespressione b, ma in tal caso non si potrebbe in alcun modo giustificare la c finale di s_1; se invece la b di s_1 fosse descritta dalla sottoespressione abc, non si potrebbe giustificare la a iniziale di s_1 (nessuna sottoespressione di E potrebbe descrivere la a singola).

b) bb
Questa stringa viene descritta da E ($bb \in L(E)$). Infatti $bb = (\epsilon b^2)^1\epsilon$, dove la prima ϵ viene descritta dalla sottoespressione a^* di E_1, $(\epsilon b^2)^1$ viene descritta da E_1 e la seconda ϵ viene descritta da E_2.

c) abc
Questa stringa viene descritta da E. Infatti $abc = \epsilon(abc^1)^1$, dove ϵ viene descritta da E_1 e $(abc^1)^1$ viene descritta da E_2.

d) bac
Questa stringa non viene descritta da E. La c finale infatti non può trovarsi non preceduta da una b in quanto il simbolo c in E è presente solo nella sottoespressione E_2 e lì è concatenate ad un simbolo b.

e) $abbabbbabc$
Questa stringa viene descritta da E. Essa può essere vista come la concatenazione delle stringhe $s_1 = abbabb$ e $s_2 = babc$ che sono descritte rispettivamente da $E1$ e E_2. Infatti $s_1 = abbabb = (a^1b^2)^2$ che è descritta da E_1, mentre s_2 riisulta dalla concatenazione di b con abc che sono entrambe descritte da E_2.

Esercizio 6

Indicare una espressione regolare definita su $\Sigma = \{a, b, c\}$ che descriva le seguenti stringhe:

- $aaababc$
- $ababc$

- *aaaac*
- *cababababc*

ma non le seguenti:

- *aaa*
- *aabbcc*
- *aaabac*
- *abacc*

Soluzione

Una descrizione banale (ma non accettabile in una prova scritta) è $aaababc + ababc + aaaac + cababababc$, in quanto questa espressione regolare include tutte e sole le stringhe del primo gruppo.

Per individuare un'espressione regolare accettabile bisogna individuare delle regolarità nel primo insieme di stringhe che non siano presenti nel secondo insieme.

Per esempio, si può notare che nel primo gruppo di stringhe il simbolo b è sempre preceduto da una a e che la stringa ab viene spesso ripetuta: l'espressione regolare $(ab)^*$ è quindi un buon candidato come nucleo della espressione regolare cercata. La situazione è quindi la seguente, dove le stringhe sottolineate sono quelle che vengono descritte dall'espressione regolare:

espressione	da includere	da escludere
$(ab)^*$	• $aa\underline{ab}abc$ • $\underline{ab}abc$ • $aaaac$ • $c\underline{ababab}c$	• aaa • $a\underline{ab}bcc$ • $aa\underline{ab}ac$ • $\underline{ab}acc$

Per ottenere l'espressione cercata, quindi dovremo estendere l'espressione nella prima colonna in modo da giungere alla situazione in cui le stringhe della seconda colonna siano tutte sottolineate e le stringhe della terza colonna siano non sottolineate.

Possiamo notare che tutte le stringhe da descrivere terminano con il simbolo c e che esso è preceduto da una stringa sottolineata (cioè già descritta dall'espressione regolare parziale attualmente in esame). Al contrario, tutte le stringhe da escludere dalla descrizione o non terminano per c, o hanno un'altra sottostringa che separa la c finale dalla parte sottolineata. Pertanto, concatenando l'espressione regolare c all'espressione parziale finora ottenuta, si perviene alla seguente situazione:

espressione	da includere	da escludere
$(ab)^*c$	• $aa\underline{ab}abc$ • $\underline{ab}abc$ • $aaaa\underline{c}$ • $c\underline{ababab}c$	• aaa • $aabbcc$ • $aaabac$ • $abacc$

Ora tutte le stringhe da escludere non possono essere descritte dall'espressione regolare fin qui trovata, che però non riesce ancora a descrivere tutte le sctringhe richeste. Bisogna quindi arricchire tale espressione, stanto attenti a non far rientrare in gioco le stringhe da escludere (e al momento escluse).

Si può notare che le sottostringhe non sottolineate sono composte o solo da a o solo da c. L'espressione regolare $a^* + c^*$ descrive queste stringhe, e perciò, concatenandola con l'espressione precedente, si giunge alla situazione:

espressione	da includere	da escludere
$(a^* + c^*)(ab)^*c$	• $\underline{aaababc}$ • $\underline{ababc}$ • $\underline{aaaac}$ • $\underline{cababababc}$	• aaa • $aabbcc$ • $aaabac$ • $abacc$

L'espressione regolare cercata può quindi essere $(a^* + c^*)(ab)^*c$.

Bisogna notare che tale espressione non è unica. Infatti, poiché (tra le altre) anche le espressioni $(a^2)^* + c$ e $(a + c)^*$ descrivono le sottostringhe sottolineate nell'ultimo passaggio, anche le espressioni $(a^2)^* + c)(ab)^*c$ e $(a+c)^*(ab)^*c$ saranno delle soluzioni accettabili.