Linguaggi Formali e Compilatori
Prof. Crespi Reghizzi
Soluzioni Prova scritta[1]
27/07/2004

NOTA: preferisco pubblicare ora queste soluzioni, prima dell'imminente appello d'esame, anche se non ho trovato il tempo per rileggerle.

1 Espressioni regolari e automi finiti 20%

1. Progetto di espr. regolare. Alfabeti $\Sigma_1 = \{a, b, c\}, \Sigma_2 = \{a, b\}$. Dati i linguaggi
   $R_1 = \{x \in (\Sigma_1)^* \mid x \text{ non contiene la sottostringa } aa\}$
   $R_2 = \{x \in (\Sigma_2)^* \mid x \text{ non contiene la sottostringa } aa\}$
   considerate i linguaggi

$$L' = R_1 R_2 \qquad L'' = R_2 R_1$$

   (a) Scrivete una frase di:

L'		L''		
$L' \setminus L''$		$L'' \setminus L'$		$L' \cap L''$

   (b) Scrivete l'espr. reg. di L' con i soli operatori di base $\{\cup, *, .\}$

   (c) Scrivete l'espr. reg. di L' usando anche l'operatore $\neg$.

Soluzione

Risulta

$$R_1 = \neg((a \mid b \mid c)^* aa (a \mid b \mid c)^*) = \Big((b \mid c) \mid a(b \mid c)\Big)^* (\varepsilon \mid a)$$

e analogamente

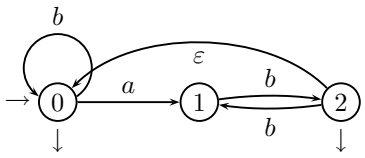
$$R_2 = \neg((a \mid b)^* aa (a \mid b)^*) = \Big(b \mid ab\Big)^* (\varepsilon \mid a)$$

[1] Tempo 2 ore 30'. Libri e appunti personali possono essere consultati. Per superare la prova l'allievo deve dimostrare la conoscenza di tutte e 5 le parti.

da cui $L' = R_1 R_2$ e $L' = R_1 R_2$, dove per R_1 e R_2 si può usare una delle due formule precedenti.

L'	ε	L''	ε		
$L' \setminus L''$	caa	$L'' \setminus L'$	aac	$L' \cap L''$	ac

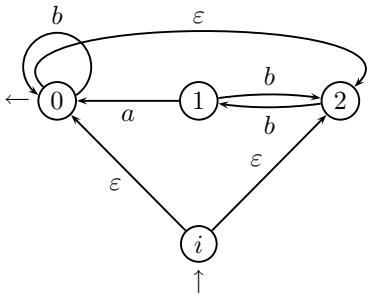
2. Dato l'automa A



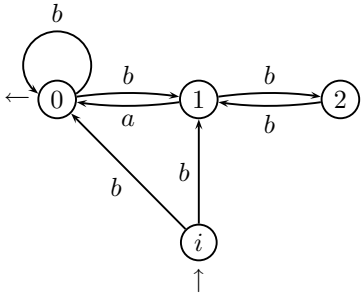
costruite, mostrando i passaggi, il riconoscitore deterministico minimo del ling. speculare $(L(A))^R$.

Soluzione

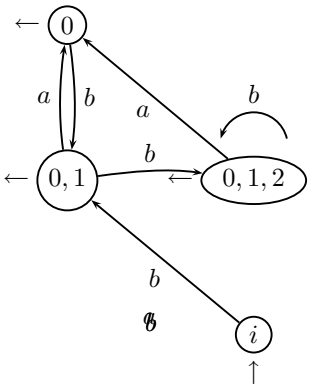
Riconoscitore del ling. riflesso:



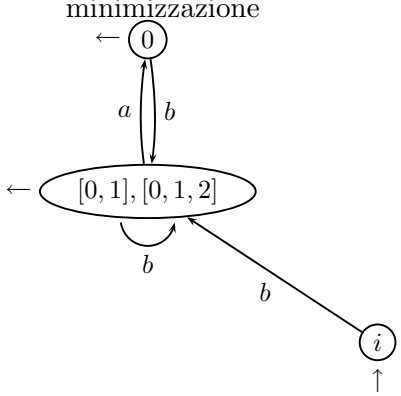
Eliminazione ε-archi :



e determinizzazione:



minimizzazione



2 Grammatiche 20%

1. Progettate una grammatica (consentita la forma EBNF) per il ling. di alfabeto $\{[,], a, b\}$ le cui frasi sono ben parentesizzate (b.p.) e inoltre contengono almeno una sottostringa b.p. in cui sono presenti lettere a ma non b.
   Esempi $[a], [[aaa][b]], [[[bb][a][a[aab]]]$
   Controesempi $[], [b], [bba]$
   Disegnare l'albero sint. di una frase rappresentativa.

   Soluzione

   Consideriamo i seguenti ling.:

   - L_1, stringhe b.p. prive di a e di b, ad es. $[[][]]$
   - L_2, stringhe b.p. prive di b con almeno una sosstostringa b.p contenente delle a
   - L_3, stringhe b.p. contenenti a o b o anche nulla.

   $A_1 \rightarrow {A_1}^+ \mid [A_1] \mid [\,]$
   $A_2 \rightarrow [a^+] \mid [(A_1 \mid A_2)^* A_2 (A_1 \mid A_2)^*]$
   $A_3 \rightarrow [(a \mid b \mid A_3)^*]$
   Il ling. richiesto è definito da:
   $S \rightarrow [{A_3}^* S {A_3}^*] \mid A_2$

2. Progettate la grammatica di un minilinguaggio di programmazione contenente

   - dichiarazioni di variabili.
     Le var. possono essere del tipo *int* o *record*.
     Le var. possono essere inizializzate con una costante o con un record di costanti.
   - istruzioni di assegnamento, con nella parte destra espr. aritmetiche semplici, senza parentesi
   - le dichiarazioni precedono gli assegnamenti
   - il separatore è il punto e virgola
   - nella grammatica un identificatore sarà denotato dal terminale v, una cost. dal terminale c
   - per quanto non precisato siete liberi di scegliere.

   Esempio:
   $Z : int := 8$; $A : int$; $B : record(C, D : int) := (3, 7)$; $E : record(F : int, G : record(H, I : int))$; $A := B.C + 2$; $E.G.H := 1$

   (a) Scrivere la grammatica (consentita la forma EBNF) non ambigua
   (b) Disegnate il diagrammia sint. di una regola della grammatica
   (c) Disegnate un albero sintattico sufficientemente rappresentativo.

Soluzione

$S \to D(;D)^* I(;I)^*$
$D \to v(,v)^* :$ int $[:=$ const $(,$ const $)^*]$
$D \to v(,v)^* : R[:= C]$
$R \to$ record $'('F(,F)^*')'$
$F \to v :$ int
$F \to R$
$C \to' ('(C \mid$ const $)(,(C \mid$ const $))^*')'$
$I \to N := [-]A((+ \mid -)A)^*$
$N \to v(\bullet v)^*)$
$A \to N \mid$ const

Domanda relativa alle esercitazioni 20%

3 Grammatiche e analisi sintattica 20%

Data la grammatica:
$S \to a(Bc)^*d \qquad B \to \varepsilon \mid (aS)^+$

1. Verificate se essa è ELL(1) (o ELL(k))
2. Scrivete, se possibile, una procedura del parsificatore
3. Costruite una grammatica BNF, equivalente, verificando che sia adatta all'analisi LR(1).

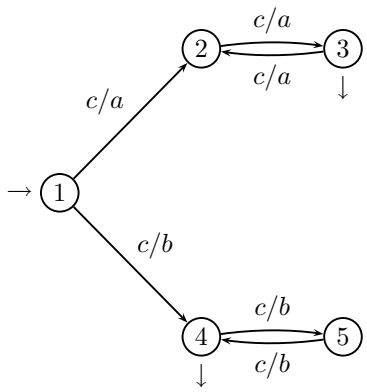
Soluzione

1. Verificate se essa è ELL(1) (o ELL(k))
   I due insiemi $ini(Bc) = \{a, c\}$ e $seg((Bc)^*) = \{d\}$ sono disgiunti.
   Inoltre per le alternative di B sono disgiunti i due insiemi $seg(B) = \{c\}$ e $ini((aS)^+) = \{a\}$.
   Infine per l'iterazione $(aS)^+$ sono disgiunti gli insiemi $ini(aS)$ e $seg(aS)^+) = seg(B) = \{c\}$.
   Pertanto la grammatica è ELL(1)
2. Scrivete, se possibile, una procedura del parsificatore
3. Costruite una grammatica BNF, equivalente, verificando che sia adatta all'analisi LR(1).
   La gramm. data equivale alla seguente
   $S \to a((aS)^*c)^*d$
   la quale equivale alla gramm. BNF G_2:
   $S \to aXd$
   $X \to \varepsilon$
   $X \to AcX$
   $A \to aSA$
   $A \to \varepsilon$

   E' facile verificare che G_2 è LL(1), quindi, a fortiori è anche LR(1). L'affermazione si può verificare costruendo il riconoscitore dei prefissi ascendenti.

4 Traduzione e semantica 20%

1. Dato il trasduttore finito:



Progettare il traduttore:

(a) Definite con un predicato la relazione di traduzione $\tau \subseteq (\{c\}^* \times \{a, b\}^*)$ calcolata dal trasduttore:

$$\tau = \{(x, y) \mid x \ldots \wedge y \ldots\}$$

(b) Scrivete uno schema di traduzione sintattica (senza attributi) per la stessa traduzione

(c) Progettate un automa trasduttore deterministico a pila per la stessa traduzione.

Soluzione

(a) Definite con un predicato la relazione di traduzione

$$\tau = \{(x, y) \mid x = c^n \wedge (y = a^n \text{ per } n \text{ pari}, y = b^n \text{ per } n \text{ dispari})\}$$

(b) Schema di traduzione sintattica

$S_1 \to cS_2$	$S_1 \to aS_2$
$S_1 \to cS_4$	$S_1 \to bS_4$
ecc.	ecc.

(c) Automa trasduttore deterministico a pila:
esso spinge nella pila le c lette, alternando tra i due stati q_0 e q_1.

Incontrando il terminatore $\dashv$, a seconda che si trovi nello stato q_0 o q_1, svuota la pila emettendo una a o una b per ogni simbolo della pila.

2. Dato il ling. di Dyck L_D definito dalla sintassi G:

1	$S \to (D)$
2	$D \to (D)D$
3	$D \to \varepsilon$

Per ogni frase $x \in L(G)$ è definito il predicato

$(\alpha(x) = true) \Leftrightarrow$ ogni coppia di parentesi (escluse le più interne) contiene lo stesso numero d

Esempi: $x_1 = (\overbrace{(\underbrace{()()}_{2\,coppie}\,)()}^{2\,coppie})$, $\qquad \alpha(x_1) = True$

$x_2 = (\overbrace{(\underbrace{()}_{1\,coppia}\,)()}^{2\,coppie})$, $\qquad \alpha(x_2) = False$

(a) Progettate una gramm. a attributi per calcolare il predicato α

(b) Disegnate un albero decorato con gli attributi

(c) Disegnate i grafi delle dipendenze funzionali e indicate queli algoritmi di valutazione si possono applicare.

Soluzione

(a) Progettate una gramm. a attributi per calcolare il predicato α
Usiamo tre attributi:

- n, sint, il numero di sottoespr. b. p., in via di calcolo
- m, ered., il numero definitivo di sottoespr. b. p.
- α, sint, il predicato

Funzioni per il calcolo di n:

1	
2	$n_0 \leftarrow 1 + n_2$
3	$n_0 \leftarrow 0$

Funzioni per il calcolo di m:

1		$\alpha_0 \leftarrow \alpha_1$
2	$m_1 \leftarrow m_0$	$\alpha_0 \leftarrow \alpha_1 \wedge \alpha_2 \wedge ((m_1 = n_1) \vee (n_1 = 0))$
	$m_2 \leftarrow m_0$	
3		$\alpha_0 \leftarrow T$

(b) Disegnate un albero decorato con gli attributi

(c) Disegnate i grafi delle dipendenze funzionali e indicate queli algoritmi di valutazione si possono applicare.
La grammatica non è a 1 scansione, quindi a fortiori neanche del tipo L, poiché nella regola 1 m_1, ereditato, dipende da n_1, sintetizzato.
Ma il calcolo di n può essere eseguito con una passata asc. (o disc.). I restanti attributi, m, α possono poi essere calcolati con una visita di tipo L.