Linguaggi Formali e Compilatori
(Formal Languages and Compilers)

prof. S. Crespi Reghizzi, prof.ssa L. Sbattella
(prof. Luca Breveglieri)
Prova scritta - Martedì 7 marzo 2006 - Parte I: Teoria
CON SOLUZIONI

NOME & COGNOME:

MATRICOLA: FIRMA:

ISTRUZIONI - LEGGERE CON ATTENZIONE:

- L'esame si compone di due parti:
  - I (80%) Teoria:
    1. espressioni regolari e automi finiti
    2. grammatiche e automi a pila
    3. analisi sintattica e parsificatori
    4. traduzione e analisi semantica
  - II (20%) Esercitazioni Flex e Bison
- Per superare l'esame l'allievo deve avere sostenuto con successo entrambe le parti (I e II), unitamente in un solo appello oppure separatamente ma entro quattro appelli. Esempio: se si sostiene con successo la parte II nell'appello di settembre, la parte I va superata entro (e non oltre) l'appello di luglio; altrimenti il voto della parte II scade.
- Consegnando una parte (qualunque sia l'esito della correzione), il voto precedente della stessa è annullato. Ritirandosi senza consegnare una parte, l'eventuale voto precedente della stessa resta valido (entro quattro appelli, vedi sopra).
- Per superare la parte I (teoria) occorre dimostrare di possedere sufficiente conoscenza di tutte le quattro sezioni (1-4) in cui si divide il tema d'esame.
- È permesso consultare libri e appunti personali.
- Per scrivere si utilizzi lo spazio libero e se occorre anche il tergo del foglio; in fondo a ogni sezione (1-4) c'è un foglio bianco addizionale.
- Tempo: Parte I (teoria): 2h.30m - Parte II (esercitazioni): 30m

1 Espressioni regolari e automi finiti 20%

1. Sia data l'espressione regolare R seguente, di alfabeto $\Sigma = \{a, b, c\}$:

$$R = \ (\ ab \ \mid \ ac\ )^*$$

(a) Si dica quali delle uguaglianze seguenti siano valide e quali no (marcare a lato), dando (sotto, non a lato) un controesempio per ciascuna uguaglianza non valida:

#	uguaglianza è valida ?	sì	no
1	$R = \overline{(b \mid c)\,\Sigma^*} \ \mid \ \overline{\Sigma^* a} \ \mid \ \overline{\Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*} \ \mid \ \overline{\Sigma\,(\Sigma^2)^*}$		
2	$R = \overline{\left(\ (b \mid c)\,\Sigma^* \ \mid \ \Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^* \ \mid \ \Sigma\,(\Sigma^2)^*\ \right)}$		
3	$R = \overline{\left(\ (b \mid c)\,\Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^* \ \mid \ \Sigma\,(\Sigma^2)^*\ \right)}$		
4	$R = \overline{(\ (b \mid c)\,\Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*\ )}$		

(b) Si ricavi un automa indeterministico equivalente all'espressione regolare $S = a^+\ R = a^+\ (\ ab\ \mid\ ac\ )^*$, cercando di contenerne il numero di stati.

(c) (facoltativo) Si ricavi l'automa deterministico minimo equivalente a S.

Soluzione

(a) Ecco la risposta complessiva:

#	uguaglianza è valida ?	sì	no
1	$R = \overline{(b \mid c)\,\Sigma^*} \ \mid \ \overline{\Sigma^* a} \ \mid \ \overline{\Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*} \ \mid \ \overline{\Sigma\,(\Sigma^2)^*}$		×
2	$R = \overline{\left(\ (b \mid c)\,\Sigma^* \ \mid \ \Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^* \ \mid \ \Sigma\,(\Sigma^2)^*\ \right)}$	×	
3	$R = \overline{\left(\ (b \mid c)\,\Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^* \ \mid \ \Sigma\,(\Sigma^2)^*\ \right)}$	×	
4	$R = \overline{(\ (b \mid c)\,\Sigma^* a \ \mid \ \Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*\ )}$		×

Ed ecco qua il ragionamento. I componenti delle espressioni a destra del segno di uguaglianza sono più o meno di tipo locale, cioè danno vincoli di inizio e fine della stringa, e di adicenza tra lettere, più un vincolo di parità sulla lunghezza. In particolare tali vincoli sono i seguenti:

i. $(b \mid c)\,\Sigma^*$ sono le stringhe di lunghezza ≥ 1 inizianti con b o c

ii. $\Sigma^* a$ sono le stringhe di lunghezza ≥ 1 terminanti con a

iii. $\Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*$ sono le stringhe di lunghezza ≥ 2 contenenti (almeno) una coppia di lettere adiacenti di tipo aa, bc, cc, cb o bb

iv. $\Sigma\left(\Sigma^2\right)^*$ sono le stringhe di lunghezza dispari 1, 3, 5, ...

Siccome tali vincoli compaiono anche in forma separatamente complementata, è bene rivederli pure così (si ricordi che $\overline{\Delta} = \Sigma^* - \Delta$):

i. $\overline{(b \mid c)\,\Sigma^*}$ sono le stringhe di lunghezza ≥ 1 inizianti con a, più ε[1]

ii. $\overline{\Sigma^* a}$ sono le stringhe di lunghezza ≥ 1 terminanti con b o c, più ε[2]

iii. $\overline{\Sigma^*\,(aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^*}$ sono le stringhe di lunghezza ≥ 2 contenenti solo coppie di lettere adiacenti di tipo[3]:

$$\Sigma^2 - \{aa, bc, cc, cb, bb\} = \{ab, ac, ba, ca\}$$

più le stringhe a, b e c, ed ε (ossia quelle di lunghezza < 2)

iv. $\overline{\Sigma\,(\Sigma^2)^*}$ sono le stringhe di lunghezza pari 2, 4, ..., più ε

Conviene dunque vedere anche l'espressione regolare R sotto tale luce. Esaminando le stringhe generate R si vede subito come quelle più brevi siano ε, ab, ac, $abab$, $abac$, $acab$, $acac$, e come in generale abbiano l'aspetto seguente:

$$L(R) \ni a\ b\ a\ b\ \ldots\ a\ b\ a\ c\ a\ c\ \ldots\ a\ c\ a\ b\ \ldots\ a\ b$$

$$L(R) \ni a\ c\ a\ c\ \ldots\ a\ c\ a\ b\ a\ b\ \ldots\ a\ b\ a\ c\ \ldots\ a\ c$$

L'intuizione suggerisce immediatamente che le stringhe di R siano caratterizzate dagli aspetti seguenti:

i. iniziano con la lettera a

ii. terminano con la lettera b o c

iii. contengono solamente coppie di lettere adiacenti del tipo ab, ac, ba e ca

iv. sono di lunghezza pari: 2, 4, ecc, compresa ε (lunghezza 0, pari, se si vuole)

Si nota subito come gli aspetti di R siano i complementi dei vincoli elencati prima. Va inoltre osservato che il complemento dell'unione di insiemi è equivalente all'intersezione dei complementi di tali insiemi (De Morgan). Tenendo conto di ciò, si vede che valgono solo le uguaglianze (2) e (3):

- la (2) vale perché a destra è l'unione complementata degli aspetti di R e dunque è l'intersezione dei complementi dei vincoli (De Morgan); la parte destra comprende i vincoli di inizio (né b né c) e di fine (non a), le adiacenze vietate, le quali sono aa, bc, cb, cc e bb come visto prima, e il vincolo di lunghezza (no stringhe di lunghezza dispari)
- la (3) (a destra anch'essa è strutturata come unione complementata) vale giacché pur accoppiando i vincoli di inizio e fine (si vietano le stringhe che, iniziando con b o c, finiscono con a, e viceversa), e dunque restringendone la portata rispetto a quanto si vede nella (2), non riesce comunque, stanti le adiacenze permesse ab, ac, ba e ca, a formare stringhe di lunghezza pari le quali, pur iniziando con b (o con c), non finiscano con a (tipo la stringa $baba$, che ha adiacenze ammissibili ma $\notin L(R)$), perché si deve alternare tra b (o c) e a ... o che finendo con a non inizino con b (o con c)

[1] Infatti ε è una stringa *non* di lunghezza ≥ 1 e *non* iniziante né con b né con c.

[2] Infatti ε è una stringa *non* di lunghezza ≥ 1 e *non* iniziante con a.

[3] Ovvero, all'insieme di tutte le coppie di lettere adiacenti possibili, Σ^2, si tolgono le coppie che non devono figurare (adiacenze vietate), ottenendo così quelle che sole possono figurare (adiacenze permesse).

Invece, l'uguaglianza (1) è *insensata*, perché per esempio a destra consente *tutte* le stringhe di lunghezza pari, cosa che R non fà (essa è l'unione dei complementi dei vincoli, e confrontandola con la (2) dove invece di fatto sono intersecati, se ne comprende tutta la radicale assurdità, come dovrebbe saltare immediatamente all'occhio), e per controesempio basta dare una delle adiacenze vietate, come aa. E la (4) (essa è affine alla (2) e alla (3)) non vale perché, indipendentemente da ogni altra osservazione, a destra non ha vincolo di parità, e dunque ammette le stringhe, non generate da R, di lunghezza uno (controesempi: a, b e c), oltre a stringhe di lunghezza dispari ≥ 3 e dunque $\notin L(R)$. Ciò chiude la questione.

Si lasciano altre tre relazioni di uguaglianza con risposta, da esaminare:

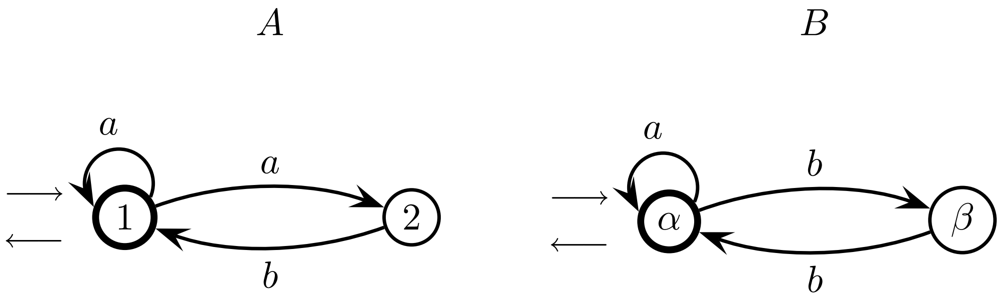
#	uguaglianza è valida ?	sì	no
5	$R = \overline{\left((b \mid c)\,\Sigma^* \;\mid\; \Sigma^* a \;\mid\; \Sigma^* (aa \mid bc \mid cc \mid cb \mid bb)\,\Sigma^* \right)}$	×	
6	$R = \overline{(b \mid c)\,\Sigma^*} \;\mid\; \overline{\Sigma^* a} \;\mid\; \overline{\Sigma^* (aa \mid bc \mid cc \mid cb)\,\Sigma^*} \;\mid\; \overline{\Sigma\,(\Sigma^2)^*}$		×
7	$R = \overline{\Big((b \mid c)\,\Sigma^* a \;\mid\; \Sigma^* (aa \mid bc \mid cc \mid cb \mid bb \mid ba)\,\Sigma^* \;\mid\; \Sigma\,(\Sigma^2)^* \Big)}$		×

Il lettore provi a giustificare le risposte servendosi degli stessi concetti di prima.

(b)

(c)

2. Sono dati i due automi riconoscitori seguenti A e B, a stati finiti, di alfabeto $\{a, b\}$:



Essi riconoscono i linguaggi regolari $L_A = L(A)$ e $L_B = L(B)$, rispettivamente.

Si svolgano i punti seguenti:

(a) Si costruisca l'automa che riconosce il linguaggio intersezione $L_\cap = L_A \cap L_B$.

(b) Si costruisca l'automa che riconosce il linguaggio unione $L_\cup = L_A \cup L_B$.

(c) Si costruisca l'automa che riconosce il linguaggio unione disgiunta $L_\oplus = L_A \oplus L_B$.

Nota: gli automi costruiti possono essere deterministici o indeterministici, a scelta; si proceda nel modo che si preferisce.

Soluzione

Se si adatta la costruzione del prodotto, si fanno tutte e tre con poco sforzo; per l'ultima occorre prima determinizzare A (in $\oplus$ è implicito un complemento) e anche mettere in evidenza lo stato di errore.

Per le prime due domande, ci sono anche facili risposte quasi intuitive: $L_A = (a \mid ab)^*$ e $L_B = (a \mid b^2)^*$, dunque $L_\cup = (a \mid ab)^* \cup (a \mid b^2)^*$, e l'automa corrispondente è quasi immediato (non-det.); mentre $L_\cap = a^*$, perché in L_A la lettera b è sempre isolata ma in L_B sempre accoppiata come bb, e l'automa corrispondente è immediato (anche det.). La terza domanda richiede la costruzione del prodotto e, prima di questa, la determinizzazione di A e l'esposizione dello stato di errore sia A sia in B; ho verificato che determinizzando A gli stati restano due (più uno di errore); l'automa prodotto ha pertanto al massimo $3 \times 3 = 9$ stati (o meno, magari si pulisce), dunque è gestibile ... Si osservi che, comunque, $L_\oplus = (a \mid ab)^* ab (a \mid ab)^* \cup (a \mid b^2)^* b^2 (a \mid b^2)^*$, perché la parte comune a^* va tolta, mentre le stringhe che contengono almeno una lettera b sono sempre disgiunte; dunque ci si arriva anche per via intuitiva, e l'automa corrispondente (non-det.) è abbastanza semplice. Oppure, si può fare con De Morgan. Ci sono pertanto diverse opzioni, sceglieranno quella che preferiscono.

2 Grammatiche libere e automi a pila 20%

1. Si consideri il linguaggio libero L seguente, di alfabeto $\Sigma = \{a, b\}$:

$$L = \left\{ a^h \, b^k \, a^h \, b^k \mid \; (h \leq 1 \land k \geq 0) \lor (h \geq 0 \land k \leq 1) \right\}$$

Esempi: ε $\quad aa \quad bb \quad abab \quad abbabb \quad aabaab$

Controesempi: $\quad aabab \quad aabbaabb$

Si svolgano i punti seguenti:

(a) Si progetti una grammatica libera G qualunque, che generi il linguaggio L.

(b) Se G è ambigua, se ne dia una forma G' equivalente non ambigua.

Soluzione

Facile, ma c'è ambiguità di unione, comunque si risolve ...

2. Si progetti la grammatica EBNF non ambigua che modella il linguaggio, semplificato, della teoria elementare degli insiemi. Sono presenti i concetti seguenti:

   - i nomi di elementi sono lettere alfabetiche minuscole, da a a z
   - i nomi di insiemi sono lettere alfabetiche maiuscole con indice intero positivo non nullo, p. es. $A1$, $B33$, ecc; $\emptyset$ è l'insieme vuoto
   - una collezione (anche vuota) di elementi separati da ',' e racchiusa tra parentesi graffe '{' e '}' è un insieme; un singoletto può non avere graffe
   - tra insiemi sono consentite le operazioni insiemistiche di unione '$\cup$', intersezione '$\cap$' e complemento '$\neg$'; complemento precede intersezione che precede unione
   - sono ammesse sotto-espressioni parentetizzate mediante '(' e ')'
   - le frasi del linguaggio consistono in liste (non vuote) di dichiarazioni del tipo:

     nome di insieme '=' espressione insiemistica ';'

     dove l'espressione è costruita con le operazioni insiemistiche consentite, a partire da altri nomi di insiemi o da collezioni di elementi.

   Esempio di frase:

$$
\begin{aligned}
A1 &= a; \\
B2 &= \{a, b\}\,; \\
C4 &= A1 \ \cup \ c; \\
D3 &= \neg \ B2 \ \cup \ \{d, e\}\,; \\
E32 &= C4 \ \cap \ (D3 \ \cup \{a\})\,;
\end{aligned}
$$

   Si scriva la grammatica G in questione (in forma EBNF non ambigua). Quali aspetti semantici non sono esprimibili sintatticamente ?

Soluzione

Abbastanza ovvia; notare che gli insiemi sono finiti, ma ce ne possono essere infiniti, per via degli infiniti nomi possibili; se ci pensano, generando le liste di elementi possono evitare le ripetizioni (non così con i nomi di insiemi).

3 Analisi sintattica e parsificatori 20%

1. È data la grammatica estesa (EBNF) G seguente:

$$S \rightarrow A^* \, b^* \, C \qquad\qquad C \rightarrow b \, C \, d$$

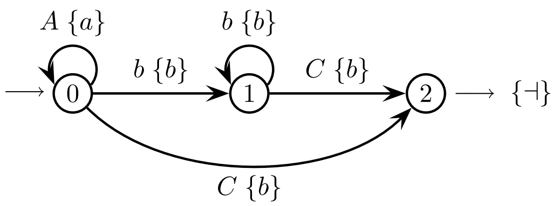
$$A \rightarrow a \, A \, b \qquad\qquad C \rightarrow b \, d^+$$

$$A \rightarrow a \, b$$

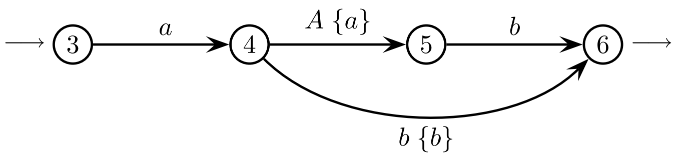
Si svolgano i punti seguenti:

(a) Si disegni la rete delle macchine ricorsive (automi) equivalente a G.
(b) Si calcolino gli insiemi guida e si verifichi che la rete delle macchine non è $LL(1)$.
(c) Si studi come modificare la grammatica G per renderla $LL(1)$.

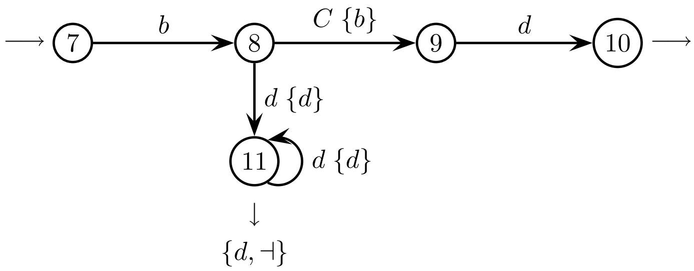
Soluzione



$S \rightarrow A^* \, b^* \, C$



$A \rightarrow a \, A \, b \mid a \, b$



$C \rightarrow b \, C \, d \mid b \, d^+$

Gli stati 1 e 11 violano la condizione $LL(1)$.

Per lo stato 11, si potrebbe eliminare l'indeterminismo, sostituendo all'ultima regola le due regole seguenti:

$$C \rightarrow D \, d^*$$

$$D \to b\ D\ d \ \mid\ b\ d$$

Ora nella regola di C l'autoanello d^* ha d come insieme guida, mentre la freccia dello stato finale ha $\dashv$ come insieme guida. La regola di D è chiaramente $LL(1)$.

Ma per togliere il problema in 1, si deve fare un ragionamento sul linguaggio

$$\{A^*\ b^*\ b^n\ d^n\ d^* \ \mid \quad m, n \geq 1\} \qquad L(A) = a^h b^h \quad h \geq 1$$

che può essere scritto come

$$\left\{A^*\ b^+\ d^+ \ \mid \quad m \geq 1\right\}$$

di qui si scrive la grammatica equivalente

$$S \to A^*\ b^+\ d^+ \qquad A \to a\ A\ b \ \mid\ a\ b$$

che risulta facilmente $LL(1)$.

2. È data la grammatica G seguente:

$$S \rightarrow S\,a\,S \qquad\qquad S \rightarrow A$$

$$A \rightarrow a\,A\,b \qquad\qquad A \rightarrow \varepsilon$$

Si svolgano i seguenti punti:

(a) Per G si costruisca l'automa pilota $LR(1)$ (cioè il riconoscitore dei prefissi).

(b) Si discuta se la grammatica G sia $LR(1)$ e $LALR(1)$.

Soluzione

Da fare ... (G è ambigua, dunque non è $LR(1)$).

4 Traduzione e analisi semantica 20%

1. Si consideri lo schema di traduzione sintattica τ seguente:

gramm. sorgente			*gramm. pozzo*		
S	$\rightarrow$	$a\ S\ b$	S	$\rightarrow$	$S\ b$
S	$\rightarrow$	$a\ S\ c$	S	$\rightarrow$	$S\ c$
S	$\rightarrow$	X	S	$\rightarrow$	X
X	$\rightarrow$	$b\ X\ a$	X	$\rightarrow$	$X\ c$
X	$\rightarrow$	$c\ X\ a$	X	$\rightarrow$	$X\ b$
X	$\rightarrow$	$b\ a$	X	$\rightarrow$	c
X	$\rightarrow$	$c\ a$	X	$\rightarrow$	b

Si svolgano i punti seguenti:

(a) Si calcoli la traduzione della stringa $a\,a\,b\,c\,a\,a\,c\,b$, e si completi la definizione della relazione di traduzione

$$\tau = \{(x, y) \mid \ x \in \{a, b, c\}^* \wedge y \in \{b, c\}^* \wedge \text{pred}\}$$

dando la definzione del predicato 'pred'.

(b) Si discuta se la traduzione, definita dallo schema, sia a un solo valore, e se sia invertibile.

(c) (facoltativo) Si costruisca un automa *riconoscitore*, il più semplice possibile, preferibilmente deterministico, che riconosca il linguaggio *sorgente* dello schema.

Soluzione

Da fare ... comunque, per il punto (a): $\tau(aabcaacb) = bccb$; se $x = a^n x_1 a^m x_2$, con $n \geq 0$, $m \geq 1$, $x_1, x_2 \in (b \mid c)^*$ e $|x_1| = m$, $|x_2| = n$, allora $\tau(x) = y = \pi(x_1^R)x_2$, dove π è la proiezione che scambia b con c e ^{R} la riflessione (p. es. da prima $x_1 = bc$, dunque $\pi(x_1^R) = \pi((bc)^R) = \pi(cb) = bc$ e $x_2 = cb$); il predicato si scrive di conseguenza. Per (b), è a un solo valore ma non invertibile. Per (c), è un automa a pila, il linguaggio sorgente è $a^n(b \mid c)^m a^m (b \mid c)^n$, $n \geq 0$, $m \geq 1$ (vedi prima), non è difficile anche se un po' noioso.

2. Una base dati contiene una relazione con tre attributi:

age	*name*	*wages*
23	mary	1200, 50
45	bob	1150, 00
19	susy	850, 00
...	...	...

Un esempio di interrogazione (in stile SQL) è il seguente:

$$\underline{\text{select}}\ \ name\ \ \underline{\text{from}}\ \ (\ 45\ \ \text{bob}\ \ 1150,00\ )\ \ (\ 23\ \ \text{mary}\ \ 1200,50\ )$$
$$(\ 19\ \ \text{susy}\ \ 850,00\ )\ \ \underline{\text{where}}\ \ age\ >\ 20$$

Essa produce come risultato una *lista* con i valori selezionati del campo indicato:

bob   mary

Il supporto sintattico è il seguente:

$$\begin{array}{rcl}
S & \rightarrow & \text{‘}\underline{\text{select}}\text{’}\ F\ \text{‘}\underline{\text{from}}\text{’}\ R\ \text{‘}\underline{\text{where}}\text{’}\ P \\
F & \rightarrow & \text{‘}age\text{’}\ \mid\ \text{‘}name\text{’}\ \mid\ \text{‘}wages\text{’} \\
R & \rightarrow & \text{‘(’}\ A\ N\ W\ \text{‘)’}\ R\ \mid\ \text{‘(’}\ A\ N\ W\ \text{‘)’} \\
P & \rightarrow & \text{‘}age\text{’}\ \text{‘>’}\ A \\
A & \rightarrow & \ldots \qquad \text{- - l’età è un intero} \\
N & \rightarrow & \ldots \qquad \text{- - il nome è una stringa} \\
W & \rightarrow & \ldots \qquad \text{- - il salario è un numero reale}
\end{array}$$

Si chiede di progettare una grammatica con attributi per calcolare il risultato della selezione, come attributo della radice dell’albero. La soluzione deve evitare di copiare inutilmente i valori che non fanno parte del risultato.

Ecco i punti da svolgere:

(a) Elencare gli attributi, con il rispettivo tipo e significato.
(b) Scrivere le funzioni semantiche che calcolano gli attributi (alle pagine successive sono già pronti gli schemi da compilare).
(c) Disegnare i grafi delle dipendenze funzionali tra attributi, per ciascuna produzione separatamente.
(d) Stabilire se la grammatica sia di tipo a una sola scansione.
(e) Stabilire se la grammatica sia di tipo L.

Soluzione

Da fare ...

sintassi	*funzioni semantiche*
$S \rightarrow$ ‘select’ F ‘from’ R ‘where’ P	
$R \rightarrow$ ‘(’ A N W ‘)’ R	
$R \rightarrow$ ‘(’ A N W ‘)’	

sintassi	*funzioni semantiche*
$F \to$ '*age*'	
$F \to$ '*name*'	
$F \to$ '*wages*'	
$P \to$ '*age*' $> A$	
$A \to \dots$	
$N \to \dots$	
$W \to \dots$	