UNIVERSITÀ
DEGLI STUDI
DI TORINO

Nozioni di base

a.a. 2018-2019

Riferimenti bibliografici

[1] **Automi, Linguaggi e Calcolabilità**, J. E. Hopcroft, R. Motwani, J. D. Ullman

[2] **Compilatori:** principi, tecniche e strumenti, A.V. Aho, M. S. Lam, R. Sethi, J.D. Ullman

Analisi lessicale [2, cap. 3]

- 3.1 Ruolo dell'analizzatore lessicale
  - 3.1.1 Analisi lessicale e parsing
  - 3.1.2 Token, pattern e lessemi
  - 3.1.3 Attributi per i token

Automi: metodo e follia [1, cap. 1]

- 1.5 I concetti centrali della teoria degli automi.
  - 1.5.1 Alfabeto.
  - 1.5.2 Stringa.
  - 1.5.3 Linguaggio.
  - 1.5.4 Problema (leggere).

Analisi lessicale [2, cap. 3]

- 3.3 Descrizione dei token
  - 3.3.1 Stringhe e linguaggi
  - 3.3.2 Operazioni sui linguaggi

Analisi lessicale

L'analizzatore lessicale o scanner:

- legge il testo sorgente, una sequenza di caratteri, e lo segmenta in sottosequenze ciascuna delle quali corrisponde ad una unità logica (sequenza di caratteri che ha un significato); le sottosequenze così individuate sono i *lessemi*.

- quando riconosce un lessema crea una coppia, chiamata *token*, che contiene le informazioni necessarie per le fasi successive.

trasforma la stringa in input in un flusso di token
da sottoporre all'analizzatore sintattico

- Il *token* è costituito da un nome e un attributo:
  <nome-token, valore-attributo>
  - *nome-token*: simbolo associato al token
  - *valore-attributo*: informazione sul lessema

Analisi lessicale

- *nome-token*: simbolo associato al token
- *valore-attributo*: informazione sul lessema

Esempio: **posizione = iniziale + velocità * 60**

lessema **posizione**; token <**id**,1>: la riga 1 della tabella dei simboli contiene le informazioni dell'identificatore **posizione**

lessema **=**; token <**=**> (nessun valore attributo)

lessema **iniziale**; token <**id**,2>, la riga 2 contiene le informazioni dell'identificatore **iniziale**

lessema **+**; token <+>

. . . . . .

lessema **60**; token <60> (<**num**,4>)

Rappresentazione finale: <**id**,1> <=> <**id**,2> <+> <**id**,3> <*****> <60>

Analisi lessicale

- *Token*: coppia <nome-token, valore-attributo>
- *nome-token*: simbolo astratto che rappresenta un'unità lessicale (una parola chiave, un identificatore, ecc.)
- *Pattern*: descrizione della forma che i lessemi di un'unità lessicale possono avere.

  Esempio: se il token è una parola chiave, il pattern è la sequenza di caratteri che formano la parola chiave. Per gli identificatori, il pattern descrive sequenze di caratteri, che sono tutte identificatori.
- *Lessema*: sequenza di caratteri del programma sorgente che rispetta il pattern di un token.

  Esempi:

  nome-token **if**, pattern: caratteri i seguito da f, lessema: **if**

  nome-token **id**, pattern: una lettera seguita da lettere e cifre

  esempi di lessemi: **posizione, iniziale**

  nome-token **num**, pattern: una cifra oppure una cifra diversa da 0 seguita da altre cifre

  esempi di lessemi: **3578, 0, 62**

Casi tipici di token

1. Un token per ogni parola chiave
2. Token per gli operatori
3. Token per gli identificatori
4. Token per le costanti (numeri, letterali, ecc.)
5. Token per altri simboli (parentesi, virgola, ecc.)

Nella coppia <nome-token, valore-attributo>,
«nome-token» specifica l'unità lessicale,
«valore-attributo» descrive il particolare lessema istanza dell'unità lessicale.

Un esempio importante è il token **id**, al quale devono essere associate tante informazioni; spesso viene utilizzato come valore-attributo il puntatore alla tabella dei simboli, dove vengono memorizzati lessema, tipo, indicazione della prima occorrenza nel programma, …

Esempio

Comando Fortran E = m*c**2

Token generati:

1. <id, puntatore ad E nella tabella dei simboli>
2. <assign-op>
3. <id, puntatore a ‘m’ nella tabella dei simboli>
4. <mult-op>
5. <id, puntatore a ‘c’ nella tabella dei simboli>
6. <exp-op>
7. <number, valore intero 2>

Analisi lessicale

Cosa deve fare l'analizzatore lessicale:

1. Rimuovere spazi, tabulazioni, a capo, . . .

2. Raccogliere i digit in numeri, in modo che nella parsificazione i numeri siano trattati come elementi atomici: 3 + 25 – 2

<**num**, 3> <+> < **num**, 25> <-> < **num**, 2>

oppure

<**num**, 1> <+> < **num**, 2> <-> < **num**, 3>

3. Riconoscere parole chiave e identificatori:
count = count + inc

<**id**, 3> <=> <**id**, 3> <+> <**id**, 4>

Analisi lessicale

Per l'analisi sintattica non è importante conoscere i lessemi associati ad ogni identificatore, ma lo è per la traduzione.

Le parole chiave di solito soddisfano le regole per la costruzione degli identificatori, quindi serve un meccanismo per decidere quando un lessema forma una parola chiave o un identificatore.
È più facile se le parole chiave sono riservate: un lessema è un **id** solo se non è una keyword.

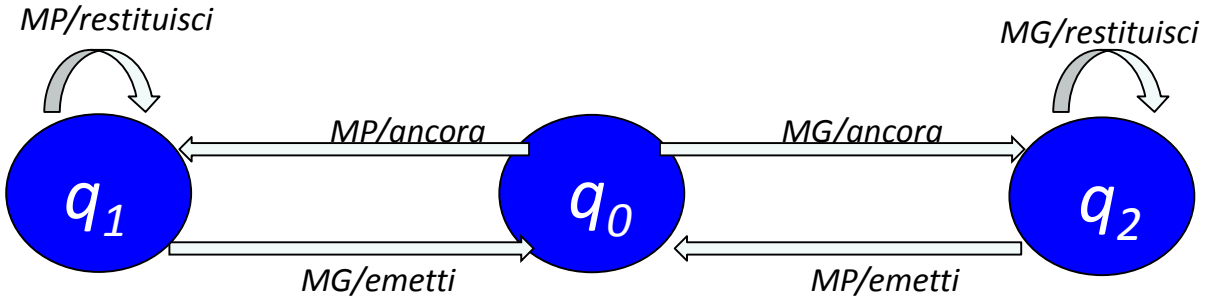
Il *riconoscimento dei lessemi* viene fatto da una "macchina", chiamata automa a stati finiti o automa finito.

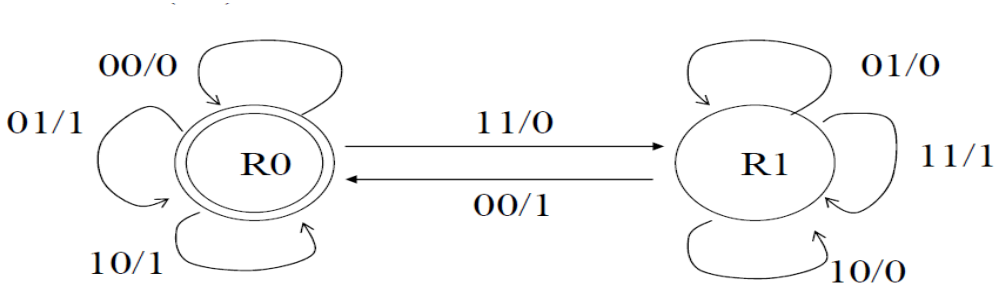
Un automa a stati finiti può essere rappresentato con un diagramma di transizione (come il Sommatore sequenziale o il Lacth SR):

Diagrammi di transizione

Un **distributore automatico** di biglietti

- accetta solo monete grandi (*MG*) e monete piccole (*MP*)
- un biglietto viene emesso quando vengono ricevute una *MP* ed una *MG*
- l'ordine di immissione delle monete non è rilevante
- è necessario sempre introdurre *MP* e *MG*
- *vengono restituite le monete che "non servono"*





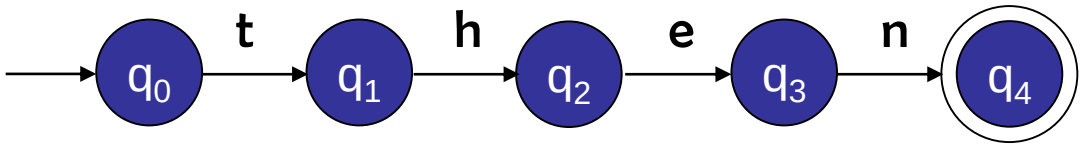
Sommatore sequenziale

Diagrammi di transizione e automi finiti

Alcune differenze:

- Gli automi finiti sono *riconoscitori*: per ogni input rispondono solo “si” o “no”
- Hanno uno stato iniziale e uno o più stati di “riconoscimento”
- Possono essere deterministici o non deterministici

Esempio: automa a stati finiti che riconosce la stringa **then**



Gli automi a stati finiti sono usati come modello per:

- Analizzatore lessicale di un compilatore
- Software per la progettazione di circuiti digitali.
- Ricerca di parole chiave in un file o sul web
- Software per verificare sistemi a stati finiti, come protocolli di comunicazione.

Nozioni di base

- *Alfabeto*: Insieme finito e non vuoto di simboli

  Esempi: - {0,1} alfabeto binario

  - {a, b, c, … , z} insieme di tutte le lettere minuscole

  - Insieme di tutti i caratteri ASCII

- *Stringa*: Sequenza finita di simboli da un alfabeto Σ, e.g. 0011001
- *Stringa vuota*: La stringa con zero occorrenze di simboli di Σ

  La stringa vuota è denotata da ε

Nozioni di base

- *Lunghezza di una stringa*: Numero di simboli nella stringa.

  |w| denota la lunghezza della stringa w

  |0110| = 4;

  |ε| = 0

- *Potenze di un alfabeto*: Σ^k = insieme delle stringhe di lunghezza k con simboli da Σ

  Esempio: $\Sigma = \{0, 1\}$

  $\Sigma^1 = \{0, 1\}$

  $\Sigma^2 = \{00, 01, 10, 11\}$

  $\Sigma^0 = \{\varepsilon\}$

Domanda: Quante stringhe ci sono in Σ^3 ?

Nozioni di base

- L'insieme di tutte le stringhe su Σ è denotato da Σ^*

  $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$

- Definiamo anche:

  $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$

  Nota: $\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$

- *Concatenazione*: Se x e y sono stringhe, allora xy è la stringa ottenuta scrivendo una copia di y immediatamente dopo una copia di x

  $x = a_1 a_2 \ldots a_i \qquad y = b_1 b_2 \ldots b_j$

  $xy = a_1 a_2 \ldots a_i b_1 b_2 \ldots b_j$

  Esempio: x = 01101, y = 110, xy = 01101110

  Nota: Per ogni stringa x: $x\,\varepsilon = \varepsilon\, x = x$

Nozioni di base

- la stringa y è una *sottostringa* della stringa x se esistono delle stringhe u e v tali che x = uyv

- la stringa y è un *prefisso* della stringa x se esiste una stringa v tale che x = yv
- N.B. un prefisso è una sottostringa in cui u = ε

- la stringa y è un *suffisso* della stringa x se esiste una stringa u tale che x = uy
- N.B. un suffisso è una sottostringa in cui v = ε

- una sottostringa (prefisso, suffisso) di una stringa è *propria* se non coincide con la stringa vuota o con la stringa stessa.

Linguaggi

- Definizione: Se Σ è un alfabeto, e $L \subseteq \Sigma^*$, allora L è un *linguaggio* su Σ

Nota: Un alfabeto può sempre essere visto come linguaggio.

Esempi di linguaggi:

- L'insieme delle parole italiane
- L'insieme dei programmi C sintatticamente corretti
- L'insieme delle stringhe che consistono di *n* zero seguiti da *n* uno, per $n \geq 0$: $\{\varepsilon, 01, 0011, 000111, \ldots\}$
- L'insieme delle stringhe con un numero uguale di zero e di uno:

  $\{\varepsilon, 01, 10, 0011, 0101, 1001, \ldots\}$
- LP = insieme dei numeri binari il cui valore è primo: {10, 11, 101, 111, 1011, . . .}
- Il linguaggio vuoto Φ
- Il linguaggio $\{\varepsilon\}$ consiste della stringa vuota

Nota: $\Phi \neq \{\varepsilon\}$

Descrittori di insiemi

Un linguaggio può essere definito mediante un descrittore di insiemi:

{w | enunciato su w}

Questa espressione va letta come “l’insieme delle *parole* w tali che vale l’enunciato su w scritto a destra della barra verticale”.

Alcuni esempi:

- {w | w consiste di un numero uguale di 0 e di 1}
- {w | w è un intero binario primo}
- {w | w è un programma C sintatticamente corretto}

Descrittori di insiemi

Spesso w viene sostituito da un'espressione con parametri e si descrivono le stringhe del linguaggio enunciando le condizioni sui parametri.

- $\{0^n1^n \mid n \geq 1\}$

  Si legge: Insieme delle stringhe formate da 0 ripetuto n volte seguito da 1 ripetuto lo stesso numero n di volte, con n maggiore o uguale a 1;

  Questo linguaggio è formato dalle parole {01, 0011, 000111, . . .}

- $\{0^n1^m \mid 0 \leq n \leq m\}$

  Questo linguaggio è formato da stringhe con un certo numero di 0 (eventualmente nessuno) seguiti da almeno altrettanti 1.

Operazioni sui linguaggi

- *Unione*:

$$L \cup M = \{w \mid w \in L \text{ o } w \in M\}$$

- *Intersezione*:

$$L \cap M = \{w \mid w \in L \text{ e } w \in M\}$$

- *Concatenazione*:

$$L.M = \{w \mid w = xy,\ x \in L,\ y \in M\}$$

- *Potenze*:

$$L^0 = \{\varepsilon\}, \quad L^1 = L, \quad L^{k+1} = L.L^k = L^k.L$$

- *Chiusura di Kleene*:

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

- *Chiusura positiva di Kleene*:

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Operazioni sui linguaggi: esempi

Sia L = {A, B, . . . , Z, a, b, . . . , z} e D = {0,1,.. .9}.

L e D sono sia alfabeti che linguaggi.

Altri linguaggi costruiti da L and D:

1. $L \cup D$ insieme di lettere e cifre – cioè il linguaggio con 62 stringhe di lunghezza uno, ognuna delle quali è una lettera o una cifra.

2. LD è l'insieme di 520 stringhe di lunghezza 2, ognuna delle quali formata da una lettera seguita da una cifra.

3. L^4 è l'insieme delle stringhe di 4 lettere.

4. L^* è l'insieme di tutte le stringhe di lettere di lunghezza finita, inclusa la stringa vuota.

5. $L(L \cup D)^*$ è l'insieme delle stringhe di lettere e cifre che cominciano con una lettera.

6. D^+ è l'insieme delle stringhe formate da una o più cifre decimali.

Proprietà delle operazioni sui linguaggi

- $L \cup M = M \cup L$

L'unione è *commutativa.*

- $(L \cup M) \cup N = L \cup (M \cup N)$

L'unione è *associativa.*

- $(L.M).N = L.(M.N)$

La concatenazione è *associativa.*

- $\Phi \cup L = L \cup \Phi = L$

Φ è *l'elemento neutro* per l'unione.

- $\{\varepsilon\}.L = L.\{\varepsilon\} = L$

$\{\varepsilon\}$ è *l'elemento neutro* per la concatenazione.

- $\Phi.L = L.\Phi = \Phi$

Φ è *l'elemento zero* per la concatenazione.

- $L.(M \cup N) = L.M \cup L.N$

La concatenazione è *distributiva a sinistra* sull'unione.

- $(L \cup M).N = L.N \cup M.N$

La concatenazione è *distributiva a destra* sull'unione.

- $L \cup L = L$

L'unione è *idempotente.*

- $(L^*)^* = L^*$

La chiusura è *idempotente*.

- $\Phi^* = \{\varepsilon\}$
- $\{\varepsilon\}^* = \{\varepsilon\}$
- $L.L^* = L^*.L = L^+$
- $L^* = L^+ \cup \{\varepsilon\}$

Esercizi

Applicare le operazioni di concatenazione, unione, intersezione e potenza ai seguenti linguaggi:

- $\{a^n b^n \mid n \geq 0\}$
- $\{a^n b^m c^p \mid n = m + p \quad e \quad n, m, p \geq 0\}$
- $\{(ab)^n (cd)^n \mid n \geq 0\}$
- $\{a^n b^m c^m d^n \mid n, m \geq 0\}$
- $\{b^m a^{2k} b^m a^n \mid n \geq 0 \quad e \quad m, k \geq 0\}$
- $\{a^n b^m c^p d^q \mid n + m = p + q \quad e \quad n, m, p, q \geq 0\}$