Linguaggi formali e compilazione

Corso di Laurea in Informatica

A.A. 2015/2016

Linguaggi formali e compilazione

Nozioni introduttive

- Informazioni generali
- Alfabeti e linguaggi
- Notazioni e convenzioni utilizzate
- Primi programmi riconoscitori in Python

Informazioni generali sul corso

- Sito web: http://algogroup.unimore.it/people/mauro/dida/2015-2016_LFC/
- Ricevimento: si consulti la pagina web all'indirizzo http://algogroup.unimore.it/people/mauro/dida/orario_ricevimento.shtml)
- Materiale didattico: si veda il sito web
- Modalità d'esame: scritto (orale solo su richiesta)
- CFU assegnati: 6 (circa 150 ore di lavoro)

Obiettivi formativi

Sviluppo di conoscenze e competenze pratiche su:

- Linguaggi formali e automi
- Linguaggi formali nell'uso quotidiano (strumenti automatici come *grep*, *sed*, *AWK*, *Lex* e *YACC*)
- Parsing di linguaggi context-free
- Cenni sulla struttura dei compilatori

Linguaggi formali in Informatica

- Linguaggi di programmazione: `C/C++`, `Java`, `Python`, ...
- Linguaggi di marcatura: `HTML`, `XML`, `LaTeX`
- Linguaggi di interrogazione: `SQL` (per DB relazionali), `SPARQL` (per dati rappresentati mediante il modello *RDF*), `GraphLog` (per *graph database*)
- Linguaggi di configurazione: `sendmail`, `apache`, `iptables` (nella directory `/etc` di un sistema Linux gli esempi si sprecano).
- Linguaggi per la descrizione (e visualizzazione) di strutture matematico/scientifiche: grafi, molecole, proteine, allineamenti di sequenze genomiche, ...

Ma che cosa è un linguaggio?

- "Il sistema di parole e segni che le persone usano per comunicarsi pensieri e sentimenti" (Merriam-Webster)
- Nozione inadeguata per lo sviluppo di una teoria matematica ed algoritmi di manipolazione dei linguaggi.
- Definizione insiemistica: *un linguaggio è un insieme di sequenze di caratteri (stringhe) su un dato alfabeto che riusciamo a caratterizzare matematicamente*
- Come vedremo, ci sono differenti metodi per "caratterizzare matematicamente" un insieme di stringhe.

Andiamo per ordine: alfabeti e stringhe

- Un *alfabeto* è un insieme finito di simboli, detti anche *caratteri*
- Esempi di alfabeti importanti:
  - I set di caratteri ASCII e UNICODE;
  - $\mathcal{B} = \{\mathtt{0}, \mathtt{1}\}$, l'alfabeto binario;
  - $\mathcal{D} = \{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$, l'alfabeto del DNA.
- Una *stringa* su un dato alfabeto è una sequenza di caratteri giustapposti
- Si noti che una stringa formata da un solo carattere è un oggetto diverso dal carattere stesso.
- Una stringa speciale è quella formata da zero caratteri, detta stringa vuota e indicata con ϵ

Operazioni e nozioni sulle stringhe

- Concatenazione (associativa ma non commutativa): $X = \texttt{reggio}$, $Y = \texttt{emilia}$, Z=XY=$\texttt{reggioemilia}$
- Potenza di una stringa X

$$X^k = \begin{cases} \epsilon & \text{se } k = 0 \\ X^{k-1}X & \text{se } k > 0 \end{cases}$$

- Prefissi, suffissi e sottostringhe (proprie e improprie)
- Riflessione di una stringa: Se $X = \texttt{Roma}$, allora $X^R = \texttt{amoR}$.
- Lunghezza di una stringa: $|\texttt{modena}| = 6$.

Specifica di linguaggi

- Un *linguaggio* su un dato alfabeto Σ è un insieme di stringhe su Σ, "caratterizzate in qualche modo".
- Se le stringhe che compongono il linguaggio sono in numero finito, una possibilità consiste nella loro elencazione.
- Esempio: il linguaggio

$$L_1 = \{00, 01, 10, 11\}$$

  definito su $\mathcal{B}$.
- Siamo naturalmente interessati a descrizioni "finite" di linguaggi infiniti, presupposto per poterli trattare

Specifica di linguaggi

- Se il linguaggio è infinito può essere comunque possibile descriverlo con quantità finita di informazione.
- Esempio: il linguaggio L_2 costituito da tutte le stringhe su $\mathcal{B}$ che terminano con il carattere 0.
- Matematicamente:

$$L_2 = \{X \in \mathcal{B}^* | X = Y0, Y \in \mathcal{B}^*\}$$

- In questo modo si descrive la "struttura" delle stringhe:
  - $L_1 = \{X \in \mathcal{B}^* : |X| = 2\}$;
  - $L_3 = \{X \in \mathcal{B}^* : |X| \geq 3\}$;
  - $L_4 = \{X \in \mathcal{B}^* : \exists k \geq 0 \,\text{t.c.}\, X = 01^k0\}$;
  - $L_5 = \{X \in \mathcal{B}^* : X = X^R\}$.

Specifica di linguaggi

- Per "scopi informatici", molto più interessanti sono altri due modi di caratterizzare i linguaggi.
- Caratterizzazione *algoritmica* (o *riconoscitiva*).
- Ogni algoritmo di decisione, tale cioè che, data in input una stringa su una dato alfabeto, risponda sempre `yes` o `no` (oppure `True` o `False`) può essere utilizzato per definire un linguaggio.
- A algoritmo di decisione, Σ alfabeto generico:

$$\mathcal{L}_A = \{X \in \Sigma^* | A(X) = \texttt{True}\}$$

- Il linguaggio `C++` è l'insieme delle stringhe sull'alfabeto ASCII (programmi) per cui il compilatore `C++` (l'algoritmo) non produce errore.

Specifica di linguaggi

- La seconda tecnica importante in ambito informatico per descrivere un linguaggio è quella *generativa*.
- Con questa tecnica si danno "regole" mediante le quali è possibile generare tutte e sole le stringhe del linguaggio che si vuole specificare.
- I due formalismi più importanti in ambito informatico sono le *espressioni regolari* e le *grammatiche context-free*.

Operazioni con i linguaggi

- Poiché i linguaggi sono insiemi, su di essi sono definite tutte le operazioni insiemistiche: unione, intersezione, differenza, ecc.
- Due linguaggi L' ed L'' si possono poi *concatenare*:

$$L = L'L'' = \{X \in \Sigma^* : \exists Y \in L', \exists Z \in L'' \,\text{t.c.}\, X = YZ\}$$

  In altre parole, L è costituito da stringhe che sono la concatenazione, in tutti i modi possibili, di una stringa di L' e di una stringa di L''.
- Più in generale, la seguente ricorrenza permette di definire, per ogni $m \geq 0$, la *potenza n-esima* di un linguaggio L:

$$\begin{aligned} L^0 &= \{\epsilon\} \\ L^n &= L^{n-1}L,\ n > 0. \end{aligned}$$

Operazioni con i linguaggi (continua)

- La *chiusura* (*riflessiva*) di L è il linguaggio

$$L^* = \bigcup_{n=0}^{\infty} L^n = L^0 \cup L^1 \cup L^2 \cup \dots$$

- Ad esempio:

$$\begin{aligned} \mathcal{B}^* &= \bigcup_{n=0}^{\infty} \{0,1\}^n \\ &= \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \dots \\ &= \{\epsilon\} \cup \{0,1\} \cup \{00,01,10,11\} \dots \end{aligned}$$

  e dunque $\mathcal{B}^*$ è l'insieme di tutte le possibili stringhe binarie.

- In generale, Σ^* é l'insieme di tutte le possibili stringhe su un alfabeto Σ.

Operazioni con i linguaggi (continua)

- La *chiusura* (non riflessiva) di L è definita come $L^+ = LL^*$, ovvero:

$$L^+ = \bigcup_{n=1}^{\infty} L^n = L^1 \cup L^2 \cup \ldots$$

- La *riflessione* di un linguaggio L su un alfabeto Σ è il linguaggio:

$$L^R = \{X \in \Sigma^* : \exists Y \in L \,\text{t.c.}\, X = Y^R\}.$$

- Il numero di stringhe di un linguaggio finito verrà indicato con la notazione $|L|$. Se L è infinito risulta $|L| = \mathbf{N}$, con ciò intendendo che L ha la stessa cardinalità dell'insieme dei numeri naturali.

Notazioni

- Quando si vorrà indicare un generico alfabeto si utilizzerà la lettera greca Σ
- I simboli di un alfabeto verranno scriti usando il font `typewriter`
- Per indicare generici caratteri di un alfabeto si useranno le lettere finali dell'alfabeto latino (x, y e z)
- Le stringhe, cioè sequenze di caratteri su un dato alfabeto, saranno spesso racchiuse fra coppie di apici, soprattutto qualora possano sorgere dubbi su dove la stringa stessa *inizia e finisce*
- In particolare, una stringa formata da un solo carattere, sarà sempre racchiusa fra apici

Notazioni (2)

- Per indicare stringhe generiche si utilizzeranno (a seconda dei casi) sia le ultime lettere dell'alfabeto latino (X, Y, Z), in maiuscolo, sia le prime lettere dell'alfabeto greco (α, β, γ), in minuscolo.
- Il numero di caratteri che compongono una stringa X è detto lunghezza di X.
- La lunghezza di una stringa X si indica spesso con $|X|$.

Notazioni (3)

- Se $|X| = n$, i singoli caratteri della stringa verranno individuati da X_i, $i = 1, \ldots, n$.
- Nel linguaggi di programmazione, si usa spesso la stessa notazione utilizzata per gli array: $X[i]$.
- La notazione $X_{i..j}$ verrà utilizzata per indicare la sottostringa di X formata dai caratteri dall'i-esimo al j-esimo.

Riconoscimento di semplici linguaggi

- Il linguaggio L_1 (già visto) delle stringhe binarie di lunghezza 2 (è un linguaggio finito).
- Il linguaggio L_6 delle stringhe sull'alfabeto $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ la cui penultima lettera è una a (due versioni caratterizzate da un diverso "consumo" di memoria RAM).
- Il linguaggio L_{parity} delle stringhe sull'alfabeto $\mathcal{B}$ che includono un numero pari di 1 (due versioni caratterizzate da un diverso "consumo" di memoria RAM).

Programma Python per L_1

```
#!/usr/bin/env python
'''
Semplice programma Python per riconoscere le
stringhe sull'alfabeto {0,1} di lunghezza 2
'''

import re, sys

language = ['00','01','10','11']

def main():
    if len(sys.argv) < 2:
        print "Usage: %s stringa_di_input"\
            %(re.sub('^.*/','',sys.argv[0]))
        sys.exit(1)
    input_string = sys.argv[1]

    if input_string in language:
        print "YES"
    else:
        print "NO"

if __name__ == '__main__': main()
```

Programma Python per L_6

```
#!/usr/bin/env python
'''
Semplice programma Python per riconoscere le stringhe
di {a,b,c}* il cui penultimo carattere e' una a
'''
import re, sys
alphabet = ['a','b','c']

def main():
    if len(sys.argv) < 2:
        print "Usage: %s stringa_di_input"\
            %(re.sub('^.*/','',sys.argv[0]))
        sys.exit(1)
    input_string = sys.argv[1]

    for c in input_string:
        if c not in alphabet:
            print "NO"
            sys.exit(1)

    if len(input_string)>1 and input_string[-2] == 'a':
        print "YES"
    else:
        print "NO"

if __name__ == '__main__': main()
```

Programma Python per L_{parity}

```
#!/usr/bin/env python
'''
Semplice programma Python per riconoscere le stringhe
di {0,1}* con un numero pari di 1
'''
import sys
alphabet = ['0','1']

def even(x):
    return x%2==0

def main():
    if len(sys.argv) < 2:
        print "Stringa di input mancante"
        sys.exit(1)

    input_string = sys.argv[1]

    count = 0
    for c in input_string:
        if c not in alphabet:
            print "NO"
            sys.exit(1)
        elif c == '1':
            count += 1

    if even(count):
        print "YES"
    else:
        print "NO"

if __name__ == '__main__': main()
```