Linguistica Computazionale

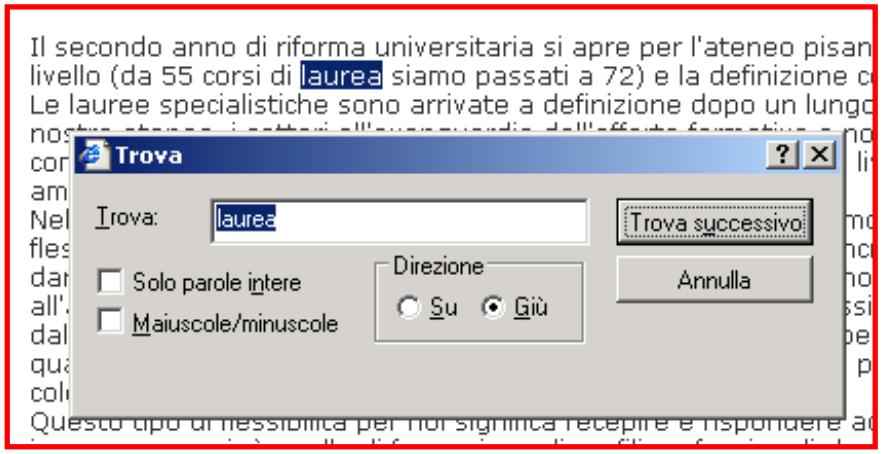
Salvatore Sorce
Dipartimento dell'Innovazione Industriale e Digitale
Ingegneria Chimica | Gestionale | Informatica | Meccanica

Ludici Adattati da Alessandro Lenci
Dipartimento di Linguistica "T. Bolelli"

Espressioni Regolari

Cercare, ricercare …

- Cercare una parola in un testo è semplice:



ma … come fare per ricerche più complesse?

- le parole che terminano in ‘ato’
- tutte e sole le sequenze di numeri che formano una data
- le frasi che incominciano con la parola ‘il’ e terminano con la parola ‘spesso’
- le linee di testo che non iniziano con una lettera maiuscola
- le parole di almeno 4 lettere e non più di 10, la cui seconda sia una ‘a’

Espressioni Regolari

- Linguaggio standard per caratterizzare stringhe di testo (regular expressions, regex, re)
  - definite da Kleene nel 1956
- Strumento ideale per
  - ricercare testo
  - sostituire testo
- Molti programmi supportano le RE:
  - "Trova e Sostituisci" in Word
  - *grep* in Unix
  - *Emacs* e altri editors di testo
- Perl è un linguaggio di programmazione che permette un trattamento estremamente avanzato e duttile delle RE

RE e Pattern Matching

- Il pattern matching è la forma più elementare di elaborazione di un testo:
  - dato un testo *T* vengono cercate le stringhe in *T* che corrispondono ad un pattern *p*
  - un pattern è uno schema di stringhe, ovvero definisce un insieme di stringhe di testo che soddisfano particolari criteri
    - "le parole che iniziano con la lettera maiuscola"
    - "le stringhe di numeri la cui seconda cifra è 2"
    - "le linee di testo che terminano con un punto esclamativo"
  - le RE sono il linguaggio standard per specificare pattern di testo da ricercare
- Stringa di testo
  - qualsiasi sequenza di caratteri alfanumerici
    - lettere, numeri, spazi, punteggiatura, caratteri speciali, ecc.
- Attenzione!!!
  - per il pattern matching, anche gli spazi, tabulazioni, ecc. contano come caratteri

RE e Pattern Matching

- In Perl una RE è un'espressione della forma /<pattern>/
- Uso delle espressioni regolari in Perl
  - Si definisce un pattern tramite una RE
  - La RE viene verificata su un testo e produce come risultato un valore booleano (true-false):
    - true = il testo contiene una stringa che corrisponde (match) al pattern
    - false = il testo non contiene una stringa che corrisponda al pattern
- Altri possibili output
  - documenti in cui viene trovata la stringa(stringhe) corrispondente(i) al pattern
  - linee di testo che contengono il pattern (es. *grep*)

Materiale on line sulle RE

- Add-on per verificare le RE su brevi stringhe di testo
  - Regular Expression (per Firefox)

- Software per le RE
  - Expresso 2.1 (http://www.ultrapico.com/Expresso.htm)
  - Visual REGEXP (http://laurent.riesterer.free.fr/regexp/)
  - RegexBuddy
- Tutorial
  - http://www.regular-expressions.info

Caratteri e sequenze di caratteri

- Un qualsiasi carattere o sequenza di caratteri (lettere, numeri, punteggiatura, spazi, ritorno-a-capo, caratteri speciali) è una RE
- le RE sono "case sensitive"

RE	Esempi di "matching"
/testo/	"il testo del corpus"
/a/	"il cane di Mario è nero"
/mark up/	"mark up del testo"
	"markup del testo"
/Linguistica/	"la Linguistica Computazionale"
	"la linguistica computazionale"

Classe di caratteri

- Un insieme di caratteri tra parentesi quadre è una RE che definisce una classe di caratteri disgiunti

RE	Definizione	Esempi di "matching"
/[st]/	il carattere 's' o il carattere 't'	"la sintassi" "il tempo"
/[1234567890]/	qualsiasi cifra	"2 parole"
/[Ll]inguistica/	'linguistica' o 'Linguistica'	"la Linguistica Computazionale" "la linguistica computazionale"

ATTENZIONE!!! - Una classe di caratteri corrisponde sempre a un solo carattere

/[ast]/ il carattere 'a' o 's' o 't' "la sintassi" "il tema"

/st/ la stringa 'st' "la sintassi" "il tema"

/[123]/ il carattere '1' o '2' o '3' "715.478"

/123/ la stringa di caratteri '123' "715.478" "674.123"

Classe di caratteri

- Dentro una classe di caratteri è possibile specificare un intervallo di caratteri in una scala usando '-':
  - /[2-5]/ il carattere 2 o 3 o 4 o 5

RE	Definizione	Esempi di "matching"
/[a-z]/	qualsiasi lettera minuscola	"la sintassi" "il tempo"
/[0-9]/	qualsiasi cifra	"2 parole"
/[a-zA-Z]/	qualsiasi lettera minuscola o maiuscola	"la Linguistica" "la linguistica"

Sono solo abbreviazioni:

/[2-5]/ è equivalente a /[2345]/

/[a-z]/ è equivalente a /[abcdefghijklmnopqrstuvwxyz]/

/[a-zA-Z0-9]/ qualsiasi carattere alfanumerico

Classe di caratteri

- Dentro una classe di caratteri è possibile specificare che un pattern non deve contenere un certo carattere usando il segno '^':
  - /[^2]/ qualsiasi carattere diverso da 2

RE	Definizione	Esempi di "matching"
/[^2]/	Qualsiasi carattere diverso da 2	"il 25%"
/[^a-z]/	qualsiasi carattere diverso da una lettera minuscola	"la Sintassi" "il tempo"
/[^st]/	qualsiasi carattere che non sia né 's' né 't'	"2 parole" "ssssss"

ATTENZIONE!

'^' ha valore negativo solo quando compare subito dopo la '['

/[2^]/ il carattere '2' o '^' "3^5"

ESEMPIO

Scrivere una RE che includa tutte le vocali

/[aeiouAEIOU]/

/[aAeEiIoOuU]/

Scrivere una RE che includa tutte le consonanti

/[^aeiouAEIOU]

Scegliere i caratteri alfabetici e poi escludere le vocali

Classe di caratteri

- Alcune utili abbreviazioni per classi di caratteri

RE	Classe di caratteri equivalente
/\d/	/[0-9]/
/\w/	/[a-zA-Z0-9_]/
/\s/	/[\t\n]/
/\D/	/[^0-9]
/\W/	/[^a-zA-Z0-9_]/
/\S/	/[^ \t\n]/

Caratteri particolari:

\t tabulazione

\n a capo

Alternativa

- L'operatore "|" esprime la disgiunzione tra due RE (operatore di alternativa)

RE	Definizione	Esempi di "matching"
/cane|gatto/	la stringa "cane" oppure la stringa "gatto"	"il cane abbaia" "il gatto miagola"

ATTENZIONE!

/[..]/ esprime solo la disgiunzione tra caratteri singoli

/[abc]/ il carattere 'a' o 'b' o 'c'

La disgiunzione tra stringhe deve essere espressa con l'operatore di alternativa

/ab|c/ la stringa 'ab' o il carattere 'c'

Moltiplicatori

- I seguenti simboli sono usati in una RE per specificare quante volte deve comparire il carattere che li precede immediatamente:
  - /<carattere>?/ “il carattere precedente è opzionale (occorre 0 o 1 volta)”
  - /<carattere>*/ “il carattere precedente occorre 0 o n volte” (Kleene Star)
  - /<carattere>+/ “il carattere precedente occorre 1 o n volte”

RE	Definizione	Esempi di “matching”
/ba?rio/	la stringa ‘brio’ o ‘bario’ (la a è opzionale)	“brio” “bario” “berio”
/tokens?/	l’ultimo carattere ‘s’ è opzionale	“token” “tokens” “tokened”

Moltiplicatori

RE	Definizione	Esempi di “matching”
/ba*/	il carattere ‘b’ seguito da *0* o *n* ‘a’	“b” “ba” “baa” “baaa” “baaaa” “baaaaa” …
/[0-9]*/	un numero infinitamente lungo, composto da *0* a *n* cifre	“2” “43” “534” “3546” “3830” “87474” “la repubblica”
/[0-9][0-9]*/	un numero infinitamente lungo che deve contenere almeno una cifra	“2” “43” “534” “3546” “3830” “87474” “la repubblica”
/[0-9]+/	un numero infinitamente lungo che deve contenere almeno una cifra	“2” “43” “534” “3546” “3830” “87474”
/ba+/	il carattere ‘b’ seguito da *1* o *n* ‘a’	“ba” “baa” “baaa” “baaaa” “baaaaa” …

Moltiplicatori

- Moltiplicatori avanzati:
  - /<carattere>{n,m}/ “il <carattere> deve occorrere almeno *n* volte e al massimo *m* volte
  - /<carattere>{n,}/ “il <carattere> deve comparire almeno *n* volte
  - /carattere>{n}/ “il <carattere> deve comparire esattamente *n* volte

RE	Definizione	Esempi di “matching”
/a{2,3}b/	la stringa formata da almeno 2 ‘a’ e al massimo da 3 ‘a’ seguita da una ‘b’	“aab” “aaab” “ab” “aaaab”
/a{2}b/	la stringa formata da esattamente 2 ‘a’ e una b	“aab” “ab” “aaab”

Ancore

- Le ancore sono caratteri speciali che specificano dove deve comparire il pattern di testo da cercare
  - /^<pattern>/ il <pattern> deve comparire all'inizio di una linea
  - /<pattern>$/ il <pattern> deve comparire alla fine di una linea
  - /\b<pattern>/ il <pattern> deve comparire all'inizio di una parola
  - /<pattern>\b/ il <pattern> deve comparire alla fine di una parola

RE	Definizione	Esempi di "matching"
/cane$/	la stringa 'cane' quando compare alla fine di una linea	"…cane¶" "il cane di Mario"
/^La/	la stringa 'La' quando compare all'inizio di una linea	"¶La macchina era guasta" "il treno per La Spezia"
/^La Spezia$/	una riga che contiene solo la stringa "La Spezia"	"¶La Spezia¶" "…a La Spezia per lavoro …"

Ancore (2)

- "\b" è un'ancora che indica il confine di una parola ("\B" indica ogni punto non confine di parola)
  - Il confine di una parola è un punto che ha da una parte un carattere di classe \w e dall'altra o un carattere di classe \W o l'inizio (fine) riga
  - **ATTENZIONE**: I caratteri accentati (à, è, è, ì, ò, ù) fanno parte della classe \W (così come lo spazio e gli altri segni di punteggiatura)

RE	Definizione	Esempi di "matching"
/\bcane\b/	la stringa 'cane' deve avere a destra e a sinistra un confine di parola	"il **cane** è …" "il canestro" "le americane sono"
/\Bcane\b/	la stringa 'cane' deve avere a destra (ma non a sinistra) un confine di parola	"il cane è …" "il canestro" "le ameri**cane** sono"
\bè\b **\Bè\B**	Trovare 'è' come copula	"Pinocchio **è** stanco" "Pinocchio è stanco"

Wildcard

- La RE /./ corrisponde a qualsiasi carattere (eccetto il ritorno-a-capo)
- /./ /.*/

RE	Definizione	Esempi di "matching"
/b.s/	qualsiasi stringa di tre caratteri che inizia con 'b' e termina con 's'	"bas" "bbs" "b3s" "b!s" "b s" "b,s" … "baas"
/b.*s/	qualsiasi stringa che inizia per b e termina per s	"bs" "bas" "bbs" "b3s" "b!s" "b s" "b,s" "baas" "bisogna prendere l'autobus"
/.*/	qualsiasi stringa (compresa quella vuota)	*l'intero corpus, anche se vuoto*

Raggruppamento e memoria

- Le parentesi tonde servono per raggruppare stringhe di caratteri da moltiplicare:

RE	Definizione	Esempi di "matching"
/(ab)+/	una o più stringhe "ab"	"ab" "abab" "ababab"
/ab+/	una "a" seguito da una o più "b"	"ab" "abb" "abbb"

Le parentesi tonde memorizzano la stringa di testo corrispondente al contenuto delle parentesi:

- la stringa viene memorizzata in una variabile temporanea
- Il contenuto della variabile può essere richiamato con \<numero>
- 1 = contenuto della prima coppia di parentesi; 2 = contenuto della seconda coppia di patentesi, ecc.

/(le|gli)(il|lo|la)+\1/

/([bcdfghjklmnpqrstvwxyz])+\1/

Raggruppamento e memoria

RE	Definizione	Esempi di "matching"
/(ab)+\1/	la variabile "\1" corrisponde a qualunque stringa abbia fatto matching con il contenuto delle parentesi	"abab" "abababab" "ababababababab"
/(a)+(b)+\1\2/	la variabile "\1" corrisponde a qualsiasi stringa abbia fatto matching con il contenuto della prima coppia di parentesi; la variabile "\2" idem, ma rispetto alla seconda coppia di parentesi	"abab" "abbabb" "aabaab" "abbaab"
/p(.)o p\1o/	la variabile "\1" corrisponde a qualunque stringa abbia fatto matching con il contenuto delle parentesi	"pio pio" "pao pao" "pro pro", ecc. "pio pao" "pao peo"
/p.o p.o/	la wildcard "." può essere sostituita da qualsiasi carattere	"pio pio" "pio pao" "pro pso", "pao pio", ecc.
/(a)(b)+\1\2/ /a(b)+a\1/		"abab" "abbabb" "aabaab" "abbaab"

Priorità

Ordine	Tipo	Esempi di “matching”
1	parentesi	( )
2	moltiplicatori	? + * {m,n} {m,} {n} ?? +? *?
3	Sequenza e ancore	“cane” ^ $ \b \B
4	alternativa	|

Caratteri speciali

- Alcuni caratteri hanno un significato speciale nel linguaggio delle RE
  - [ ] ? * . ( ) + - / { }
- Se questi caratteri fanno parte del pattern di testo da cercare, devono comparire in una RE con davanti il carattere '\' (carattere di escape)
  - /[a/ ERRORE! '[' è interpretato come classe di carattere e manca la parentesi ']'
  - /\[a/ la stringa "[a"
  - /a./ qualsiasi stringa di due caratteri che inizia con 'a' "ab" "au" as" "a1" "a?" …
  - /a\./ la stringa di testo "a."
  - /cane?/ le stringhe "cane" e "can"
  - /cane\?/ la stringa "cane?"

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le vocali minuscole o maiuscole“

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le vocali minuscole o maiuscole“

  Sol.: /[AaEeIiOoUu]/

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le parole che contengono la stringa “re””

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le occorrenze della stringa “re””

Sol.: /re/

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le parole che finiscono con la stringa “re” ”

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - “tutte le parole che finiscono con la stringa “re” ”

Sol.: /re\b/

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - "le parole che contengono "tar" o "tr"”

Sol.: /ta?r/

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - "le parole che iniziano per "tar" o per "tr""

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - "le parole che iniziano per "tar" o per "tr""

Sol.: /\bta?r/

/\btar|tr/

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - ”sequenze di numeri"

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - ”sequenze di numeri”

Sol.: /\d*/

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - \b(il | l.)\b
  - \b(il | l.)\b\s+
  - \b(il | l.)\b\w+
  - \b(il | l.)\b\s+\w+
  - \bun.?\b\s+\w+

Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare le stanze.

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - **\b(il | l.)\b**
  - \b(il | l.)\b\s+
  - \b(il | l.)\b\w+
  - \b(il | l.)\b\s+\w+
  - \bun.?\b\s+\w+

*Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere **il** fuoco e per riscaldare **le** stanze.*

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - \b(il | l.)\b
  - **\b(il | l.)\b\s+**
  - \b(il | l.)\b\w+
  - \b(il | l.)\b\s+\w+
  - \bun.?\b\s+\w+

Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare ***le*** *stanze.*

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - \b(il | l.)\b
  - \b(il | l.)\b\s+
  - **\b(il | l.)\b\w+**
  - \b(il | l.)\b\s+\w+
  - \bun.?\b\s+\w+

Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere ***il fuoco*** *e per riscaldare le stanze.*

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - \b(il | l.)\b
  - \b(il | l.)\b\s+
  - \b(il | l.)\b\w+
  - **\b(il | l.)\b\s+\w+**
  - \bun.?\b\s+\w+

Non era un legno di lusso, ma un semplice pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare ***le stanze****.*

Esercizi

- Elencare le stringhe corrispondenti alle seguenti espressioni regolari
  - \b(il | l.)\b
  - \b(il | l.)\b\s+
  - \b(il | l.)\b\w+
  - \b(il | l.)\b\s+\w+
  - **\bun.?\b\s+\w+**

*Non era **un legno** di lusso, ma **un semplice** pezzo da catasta, di quelli che d'inverno si mettono nelle stufe e nei caminetti per accendere il fuoco e per riscaldare le stanze.*

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - le parole che terminano con un segno di punteggiatura (es. "cane," "finito;" ecc.)

Esercizi

- Formalizzare con le espressioni regolari i patterns per trovare le seguenti stringhe (NB: parola = sequenza di caratteri separati da spazi)
  - le parole che terminano con un segno di punteggiatura (es. "cane," "finito;" ecc.)

Sol.: /[a-zA-z]+[;:,\?!\.]/