

Linguistica Computazionale

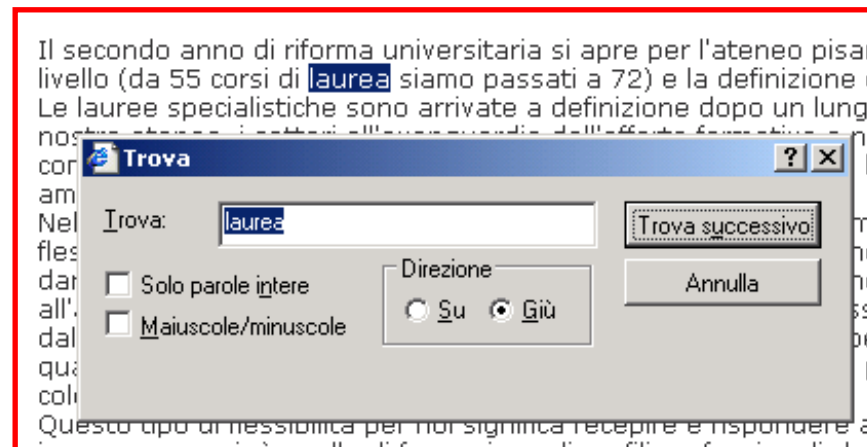
Salvatore Sorce
Dipartimento di Ingegneria
Chimica, Gestionale, Informatica e Meccanica

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/ /a/	“il <u>testo</u> del corpus” “il <u>ca</u> ne di Mario è nero”
/mark up/	“ <u>mark up</u> del testo” “markup del testo”
/Linguistica/	“la <u>Linguistica</u> 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 <u>s</u> intassi" "il <u>t</u> empo"
/[1234567890]/	qualsiasi cifra	" <u>2</u> parole"
/[Ll]inguistica/	'linguistica' o 'Linguistica'	"la <u>Linguistica</u> Computazionale" "la <u>linguistica</u> computazionale"

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

/[ast]/	il carattere 's' o 't'	"la <u>s</u> intassi" "il <u>t</u> ema"
/st/	la stringa 'st'	"la sintassi" "il tema" "la <u>s</u> trada"
/[123]/	il carattere '1' o '2' o '3'	"7 <u>1</u> 5.478"
/123/	la stringa di caratteri '123'	"715.478" "674. <u>1</u> 23"

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"
<code>/[a-z]/</code>	qualsiasi lettera minuscola	<u>"la sintassi"</u> <u>"il Tempo"</u>
<code>/[0-9]/</code>	qualsiasi cifra	<u>"2 parole"</u>
<code>/[a-zA-Z]/</code>	qualsiasi lettera minuscola o maiuscola	<u>"la Linguistica"</u> <u>"la linguistica"</u>

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"
<code>/[^2]/</code>	Qualsiasi carattere diverso da 2	" <u>il</u> 25%"
<code>/[^a-z]/</code>	qualsiasi carattere diverso da una lettera minuscola	"la <u>S</u> intassi" "il tempo"
<code>/[^st]/</code>	qualsiasi carattere che non sia né 's' né 't'	" <u>2</u> parole" "ssssss"

ATTENZIONE!

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

`/[2^]/` il carattere '2' o '^' "35"

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 20110523

- Alcune utili abbreviazioni per classi di caratteri

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

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 <u>cane</u> abbaia” “il <u>gatto</u> 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)	“ <u>brio</u> ” “ <u>bario</u> ” “berio”
/tokens?/	l’ultimo carattere ‘s’ è opzionale	“ <u>token</u> ” “ <u>tokens</u> ” “tokened”

Moltiplicatori

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

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”
<code>/a{2,3}b/</code>	la stringa formata da almeno 2 ‘a’ e al massimo da 3 ‘a’ seguita da una ‘b’	<u>“aab”</u> “aaab” “ab” “aaaab”
<code>/a{2}b/</code>	la stringa formata da esattamente 2 ‘a’ e una b	<u>“aab”</u> “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"
<code>/cane\$/</code>	la stringa 'cane' quando compare alla fine di una linea	"... <u>cane</u> " "il cane di Mario"
<code>/^La/</code>	la stringa 'La' quando compare all'inizio di una linea	" <u>La</u> macchina era guasta" "il treno per La Spezia"
<code>/^La Spezia\$/</code>	una riga che contiene solo la stringa "La Spezia"	" <u>La Spezia</u> " "...a La Spezia per lavoro ..."

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""

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]+\b[;:.,?!\\.]`