

Fondamenti di Informatica  
per la Sicurezza  
a.a. 2006/07

## Espressioni regolari

**Stefano Ferrari**



UNIVERSITÀ DEGLI STUDI DI MILANO  
DIPARTIMENTO DI TECNOLOGIE DELL'INFORMAZIONE

Stefano Ferrari ★ Università degli Studi di Milano

Fondamenti di Informatica per la Sicurezza ◇ Espressioni regolari ◇ a.a. 2006/07 - p. 1/36

## Espressioni regolari

Le **espressioni regolari** (ER) sono un metodo alternativo per descrivere i linguaggi regolari.

L'insieme delle ER descrive quindi la stessa classe di linguaggi delle grammatiche regolari e degli automi a stati finiti.

Esse presentano i seguenti vantaggi:

- offrono una notazione compatta;
- sono facilmente intelleggibili;
- offrono una descrizione di tipo costruttivo/operazionale.

Stefano Ferrari ★ Università degli Studi di Milano

Fondamenti di Informatica per la Sicurezza ◇ Espressioni regolari ◇ a.a. 2006/07 - p. 2/36

## Alfabeto di una espressione regolare

Una espressione regolare:

- è una stringa su un alfabeto  $\Sigma \cup \{\emptyset, +, *, (, )\}$ ;
- denota un insieme di stringhe di  $\Sigma^*$ .

Una espressione regolare è dunque composta di:

- simboli dell'alfabeto su cui definito il linguaggio da essa descritto,  $\Sigma$ ;
- un insieme di metasimboli usati per indicare le operazioni sui linguaggi,  $\{\emptyset, +, *, (, )\}$ .

## Sintassi e semantica delle ER

L'insieme delle stringhe denotate da una ER è definito ricorsivamente come:

espressione regolare	insieme descritto	note
$\emptyset$	$\emptyset$	
$\epsilon$	$\{\epsilon\}$	
$a$	$\{a\}$	$a \in \Sigma$
$(r + s)$	$R \cup S$	$r$ e $s$ sono ER che denotano gli insiemi $R$ e rispettivamente
$(rs)$	$RS$	
$(r^*)$	$R^*$	

## Esempio 1

L'ER  $(0 + 1)^*0$  indica le stringhe binarie pari.

Infatti:

- $0 + 1$  è l'insieme dei simboli binari;
- $(0 + 1)^*$  è una qualsiasi stringa binaria (eventualmente vuota);
- $(0 + 1)^*0$  è una qualsiasi stringa binaria terminata con 0.

## Esempio 2

L'ER  $(0 + 1)^*111(0 + 1)^*$  indica le stringhe binarie che contengono almeno tre simboli 1 consecutivi.

Infatti:

- $(0 + 1)^*$  è una qualsiasi stringa binaria;
- 111 è la stringa costituita da tre 1 consecutivi.

L'ER data descrive anche:

- le stringhe che iniziano per 111;
- le stringhe che terminano per 111;
- la stringa 111.

### Esempio 3

L'ER  $((0 + 1)^2)^*(0 + 1)$  indica le stringhe binarie di lunghezza dispari.

Infatti:

- $(0 + 1)^2$  è una qualsiasi stringa binaria di lunghezza 2;
- $((0 + 1)^2)^*$  è una qualsiasi stringa binaria di lunghezza pari;
- $((0 + 1)^2)^*(0 + 1)$  è una qualsiasi stringa binaria di lunghezza dispari.

L'ER data descrive anche le stringhe di lunghezza 1.

### Estensioni

La sintassi ER può essere arricchita:

ER	insieme descritto	note
$(r^+)$	$R^+ = R^* - \{\epsilon\}$	$r$ denota
$(r?)$	$R \cup \{\epsilon\}$	l'insieme $R$

Inoltre, a seconda delle situazioni d'uso:

- metasimboli per l'indicazione di  $\Sigma$  o di suoi sottoinsiemi;
- altri operatori (complemento, sottoespressioni);
- eliminazione di parentesi superflue (regole di precedenza, associatività di alcune operazioni).

## Uso delle espressioni regolari

Le ER vengono utilizzate in diversi ambienti:

- linguaggi di programmazione (Perl, PHP, Python, JavaScript, Java);
- shell (DOS, bash);
- programmi di utilità (grep, expr, sed, awk).

Esempi:

- lista di tutti i file eseguibili (in DOS shell):

```
C:> dir *.EXE
```

- lista degli accessi dal dominio 159.149 (con grep):

```
grep -E "^159\.149(\.[0-9]{1,3}\){2}" access.log
```

## Esercizio 1 (1)

Scrivere una espressione regolare (ER) per stringhe binarie che descriva tutte le stringhe che contengono almeno tre 1.

### Soluzione

L'ER richiesta può essere descritta discorsivamente come una stringa che ha tre simboli 1:

- intervallati da una qualsiasi stringa binaria;
- preceduti da una qualsiasi stringa binaria;
- seguiti da una qualsiasi stringa binaria.

**Esercizio 1 (2)**

Quindi, una ER che risponde ai requisiti è:

$$(0 + 1)^*1(0 + 1)^*1(0 + 1)^*1(0 + 1)^*$$

che può essere scritta più sinteticamente come:

$$((0 + 1)^*1)^3(0 + 1)^*$$

o, equivalentemente:

$$(0 + 1)^*(1(0 + 1)^*)^3$$

Tuttavia, tale descrizione è ridondante: alcuni simboli delle generiche stringhe binarie intercalanti potrebbero essere 1.

**Esercizio 1 (3)**

Quindi, la descrizione informale può diventare:

- una stringa che ha tre simboli 1;
- intervallati da una qualsiasi sequenza di 0;
- preceduti da una qualsiasi sequenza di 0;
- seguiti da una qualsiasi stringa binaria.

Usando la sintassi ER:

$$(0^*1)^3(0 + 1)^*$$

Con ragionamento analogo:  $(0 + 1)^*(10^*)^3$ .

## Esercizio 2 (1)

Scrivere una espressione regolare per stringhe di testo che descriva le date in formato GG/MM/AAAA.

### Soluzione

Per comodità, definiamo l'insieme delle cifre decimali:

$$D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

e l'espressione regolare corrispondente:

$$D = (0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9)$$

## Esercizio 2 (2)

L'espressione regolare cercata può ora essere espressa come:

$$D^2/D^2/D^4$$

**Nota:** Oltre alle date, l'ER data descrive anche stringhe che non hanno significato, se interpretate come date.

Per esempio, 00/00/0000 o 62/91/3456.

È possibile costruire una ER che descriva tutte e sole le date significative, ma non è semplice.

Tale ER dovrebbe incorporare tutte le regole per il numero di giorni per mese e per gli anni bisestili.

### Esercizio 3 (1)

Scrivere una espressione regolare per i numeri telefonici italiani di sei o otto cifre, che comprenda recapiti del tipo:

- 0039 0373 89 80 62;
- +39 0373 89 80 62;
- 02 50 33 00 62;

#### Soluzione

Per comodità, riutilizziamo l'insieme  $D$  e definiamo  $S = \{ " " \}$  e  $P = \{ "+" \}$ .

### Esercizio 3 (2)

Il prefisso internazionale può essere in due forme:

- $0^239$
- $P39$

Il prefisso urbano può essere costituito da una a tre cifre, precedute da 0:

- $0(D + D^2 + D^3)$

Infine, il numero può essere composto da tre o da quattro coppie di decimali, separate da uno spazio:

- $((D^2S)^2 + (D^2S)^3)D^2$



### Esercizio 3 (3)

Per unire il tutto bisogna tener conto che:

- il prefisso internazionale può non esserci;
- tra le varie parti del numero ci vanno gli spazi.

Quindi:

$$(0^239 + P39 + \epsilon)S0(D + D^2 + D^3)S((D^2S)^2 + (D^2S)^3)D^2$$

### Esercizio 4 (1)

Sia data l'espressione regolare su  $\Sigma = \{a, b, c\}$ :

$$E = (a^*b^2)^*(b + abc^*)^*$$

Quali fra le seguenti stringhe vengono descritte da  $E$ ?

a)  $aabc$

d)  $bac$

b)  $bb$

e)  $abbabbbabc$

c)  $abc$

## Esercizio 4 (2)

### Soluzione

Si può notare che  $E = E_1 E_2$ , dove  $E_1 = (a^* b^2)^*$  e  $E_2 = (b + abc^*)^*$ .

a)  $aabc$

$aabc$  non può essere descritta da  $E$  perché:

- la  $c$  finale potrebbe essere generata solo da  $E_2$ , ma può esserlo solo se preceduta da  $ab$ ;
- la sottostringa  $abc$  potrebbe essere descritta da  $E_2$ , ma in nessun modo la singola  $a$  potrebbe essere ricavata.

## Esercizio 4 (3)

b)  $bb$

L'espressione regolare  $bb$  può essere coinvolta nella catena di inclusioni:

$$bb = b^2 \subset a^* b^2 \subset (a^* b^2)^* \subset (a^* b^2)^* (b + abc^*)^*$$

c)  $abc$

L'espressione regolare  $abc$  può essere coinvolta nella catena di inclusioni:

$$abc \subset abc^* \subset b + abc^* \subset (b + abc^*)^* \subset (a^* b^2)^* (b + abc^*)^*$$

## Esercizio 4 (4)

d)  $bac$

$bac$  non può essere descritta da  $E$  perché la  $c$  finale potrebbe essere generata solo da  $E_2$ , ma può esserlo solo se preceduta da  $ab$ .

e)  $abbabbbabc$

L'espressione regolare  $abbabbbabc$  può essere coinvolta nella catena di inclusioni:

$$\begin{aligned} abbabbbabc &= (abb)(abb)(b)(abc) = \\ &= (ab^2)(ab^2)(b)(abc) = (ab^2)^2(b)(abc) \subset \\ &\subset (a^*b^2)^2(b)(abc^*) \subset (a^*b^2)^*(b + abc^*)^2 \subset \\ &\subset (a^*b^2)^*(b + abc^*)^* \end{aligned}$$

## Esercizio 5 (1)

Indicare una ER su  $\Sigma = \{a, b, c\}$  che descriva le seguenti stringhe:

•  $bbccaaa$  •  $bbcca$

•  $bbb$  •  $bbbaa$

ma non le seguenti:

•  $bbcbac$  •  $cbbaa$

•  $bbcab$  •  $abbcab$

## Esercizio 5 (2)

### Soluzione

Soluzioni banali (da non considerare):

- $\{bbccaaa, bbb, bbcca, bbbaa\}$
- $\Sigma^* - \{bbcbac, bbcabc, cbbaa, abbcab\}$

Si può notare che le stringhe da includere sono tutte composte da:

- una sequenza di  $b$ ,
- seguita eventualmente da una sequenza di  $c$ ,
- terminata da un'eventuale sequenza di  $a$ .

## Esercizio 5 (3)

Queste caratteristiche possono essere descritte dall'espressione regolare  $b^*c^*a^*$ :

- sequenza iniziale di  $b$  come prefisso:  $\underline{b^*}c^*a^*$ ;
- eventuale sequenza di  $c$ :  $b^*\underline{c^*}a^*$ ;
- ed eventuale sequenza di  $a$ :  $b^*c^*\underline{a^*}$ .

## Esercizio 5 (4)

Nessuna delle stringhe del secondo gruppo viene descritta da  $b^*c^*a^*$ :

- *bbcbac*: ha una *c* che inframezza la potenziale sequenza di *b*;
- *bbcab*: dopo la sequenza *bbca* iniziale (che verrebbe descritta dall'espressione regolare considerata) inizia una nuova sequenza *bc*;
- *cbbaa*: antepone *c* alla sequenza di *b*;
- *abbcab*: antepone *a* alla sequenza di *b* iniziale.

## Pattern matching

Le espressioni regolari (ER) sono lo strumento ideale per le operazioni di **pattern matching** (corrispondenza al modello) su informazioni testuali.

L'ER descrive un modello e i programmi cercano nei dati le stringhe che corrispondono a tale modello.

Esempi:

- analisi di log;
- sostituzione di stringhe di testo;
- ricerca in database.

## Motivazioni

Le espressioni regolari (ER) permettono di specificare un insieme di stringhe in modo:

- **compatto:**
  - ripetizioni, alternative;
- **simile all'oggetto descritto:**
  - l'alfabeto delle stringhe è parte dell'alfabeto delle ER;
- **costruttivo:**
  - combinazione di ER mediante operatori di stringhe.

## Dove si usano le espressioni regolari

Le ER sono comunemente usate in molti ambiti:

- **linguaggi di programmazione:**
  - C, Java, PHP, Python, Perl;
- **shell:**
  - DOS, bash;
- **file di configurazione:**
  - apache, procmail, majordomo;
- **programmi di utilità:**
  - sed, awk, grep;
- **programmi avanzati di elaborazione testi:**
  - vi, emacs.

## Espressioni regolari standard

Non esiste uno standard, ma molte convenzioni sono condivise.

IEEE POSIX 1003.2 definisce:

- basic regular expression (BRE);
- extended regular expression (ERE).

**Nota:** POSIX sta per Portable Operating System Interface.

Faremo riferimento principalmente alla notazione ERE.

## Metasimboli

In un file di testo possono essere utilizzati tutti i caratteri consentiti dalla codifica.

L'alfabeto delle ER è composto dall'alfabeto delle stringhe da specificare e da metasimboli.

Le ER sono specificate tramite caratteri di testo.

Questo comporta che alcuni simboli devono essere realizzati tramite una coppia di caratteri consecutivi: le **sequenze di escape**.

Esempio: `"\"` può significare `"`.

## Operazioni di base

Gli operatori di base delle ER si traducono:

- **potenza**: la ripetizione di una sottoespressione si indica con il numero di ripetizioni fra parentesi graffe;
  - esempio:  $\mathbf{ab}\{3\}$  corrisponde all'ER  $ab^3 = abbb$ ;
- **chiusura**: analogamente, la chiusura viene indicata con l'asterisco;
  - esempio:  $\mathbf{ab}^*$  corrisponde all'ER  $ab^*$ ;
- **sottoespressioni**: le sottoespressioni si delimitano con le parentesi tonde;
  - esempio:  $\mathbf{(ab)^*cd}$  corrisponde all'ER  $(ab)^*cd$ ;
- **unione**: sottoespressioni alternative vengono indicate con il simbolo "pipe";
  - esempio:  $\mathbf{a|b}$  corrisponde all'ER  $a + b$ .

## Estensioni

Le estensioni facilitano la specifica dei pattern:

- $\Sigma$ : un qualsiasi carattere di testo viene indicato tramite il punto;
  - esempio:  $\mathbf{.*}$  corrisponde all'ER  $\Sigma^*$ ;
- $\epsilon$ : la stringa nulla si indica come una sottoespressione vuota;
  - esempio:  $\mathbf{(a|())}$  corrisponde all'ER  $(a + \epsilon)$ ;
- **corrispondenze multiple**:
  - $\mathbf{a^+b}$  corrisponde all'insieme  $(a^* - \epsilon)b$ ;
  - $\mathbf{a?b}$  corrisponde all'ER  $(a + \epsilon)b$ ;
  - $\mathbf{a\{2, \}b}$  corrisponde all'insieme  $(a^* - a - \epsilon)b$ ;
  - $\mathbf{a\{2, m\}b}$  corrisponde all'insieme  $(a^2 + a^3 + a^4)b$ .



## Sottoinsiemi di caratteri

- **elenco di caratteri**: una sequenza di caratteri fra una coppia di parentesi quadre indica un insieme;
  - esempio: `[abd]` corrisponde all'insieme  $\{a, b, d\}$ ;
- **intervallo**: sfruttando l'ordinamento dei caratteri ASCII, si possono indicare intervalli di caratteri;
  - esempio: `[a-d]` corrisponde all'insieme  $\{a, b, c, d\}$ ;
- **complemento**: l'operatore di complemento insiemistico si indica ponendo l'accento circonflesso come primo elemento di un sottoinsieme di caratteri;
  - esempio: `[^abd]` corrisponde all'insieme  $\Sigma - \{a, b, d\}$ ;
- **classi**: esistono identificatori per i sottoinsiemi comunemente utilizzati;
  - esempio: `[:digit:]` corrisponde all'insieme  $\{0, \dots, 9\}$ .

## Ancoraggi

La posizione di una stringa nel testo può essere un fattore discriminante.

Esempio: l'IP del client viene posto all'inizio della riga di log.

I **metasimboli di ancoraggio** servono per indicare una posizione relativa all'interno del testo:

- `"^"` indica l'inizio della riga;
- `"$"` indica la fine della riga;
  - e `"\^"` e `"\$"` indicano, rispettivamente, i caratteri `"^"` e `"$"`.

## Riferimenti

I **riferimenti a precedenti sottoespressioni** (solo BRE) sono usati per poter usare la sottostringa che corrisponde al sottomodello.

Esempio:

- `(a*)b\1` corrisponde ad `aabaa`, ma non ad `aaba`.

I riferimenti vengono utilizzati soprattutto per le sostituzioni.

Esempio:

- cambiare il path a tutti i file di un file di configurazione;
  - `sed 's/olddir\/\(.*\)/newdir\/\1/g' config.txt`

## Esempi

Alcuni esempi di impiego delle ER:

- analisi degli accessi (grep):

```
grep -E "^159\.149(\.[:digit:]{1,3}){2}" access.log
```

- controllo (php):

```
eregi ("ozilla.[23]|MSIE.3)", $HTTP_USER_AGENT);
/* Returns true if client browser is
   Netscape 2, 3 or MSIE 3. */
```

- rewriting (apache):

```
RewriteRule ^/rim(/.*)" http://rim.dti.unimi.it/$1
```