Approfondimento sul Pattern Matching 2D

Liva Giovanni

Università di Udine

31 maggio 2014

Argomenti

1. Lavori Precedendi
    - Exact vs Approximate
    - Approximate Dictionary Matching
    - Baker and Bird
    - Zhu and Takaoka
    - Baeza-Yates and Regnier
    - Strutture dati utili
2. 2D con FA
3. 2D con PDA

Definizioni di Base

Dimensioni -¿ (m*m') Testo 2D; m Testo 1D
(n*n') Pattern 2D -¿ n Pattern 1D
k errori ammessi
D_H = hamming
P = pattern
T = testo

Exact vs Approximate

Il pattern matching esatto prevede di cercare l'occorrenza di una stringa all'interno di un testo. L'uso di automi per questo approccio è abbastanza naturale come si vede dal seguente algoritmo:

Algorithm 1 Creazione FA :: Pattern - Matching Esatto

```
$\delta(q_0, a) = \{q_0\}, \forall a \in A$
$\delta(q_0, p_1) = \{q_0, q_1\}$
for $i = 1$ to $m - 1$ do
    $\delta(q_i, p_{i+1}) = \{q_{i+1}\}$
end for
```

L'automa risultante ha $m + 1$ stati. Ogni stato q_i indica che si è letto il prefisso del pattern fino all'*i*-esimo carattere

Exact vs Approximate

Il pattern matching approsimato si basa sulla distanza di Hamming che permette di quantificare il numero di errori ammessi. La costruzione di tale automa prevede l'uso di $k+1$ copie di automi per il Pattern Matchin esatto, $M_0, \ldots, M_k$.
L'idea è quella che ogni M_i rappresenta il pattern accettato con i errori. I vari M_i sono collegati con una transizione dallo stato q_j allo stato q_{j+1} che corrisponde all'azione di *sostituzione* nel calcolo della distanza di Hamming, etichettata con il simbolo $\overline{p}_{j+1}$ corrispondente al carattere complementare in posizione $j+1$ in P.

Exact vs Approximate

Theorem

L'automa per il Pattern Matching approsimato ha $(k+1)(m+1-k/2)$ *stati*

Approximate Dictionary Matching

Definition

Sia π un dizionario di s pattern, $\pi = \{p_1, \ldots, p_s\}$
Sia $m = min\{|p_1|, \ldots, |p_s|\}$
Sia k il numero di errori ammessi per ogni pattern, $k < m$
L'automa $\mathcal{A}$ per l'approximate dictionary matching riconosce il linguaggio
$L(\mathcal{A}) = \bigcup_{i=1}^{s} \{uv | u, v \in A^*, D_H(p_i, v) \leq k, p_i \in \pi\}$

Algorithm 2 Creazione FA :: Approximate Dictionary Matching

```
for $i = 1$ to $s$ do
    Costruisci $M_i$ con la tecnica per il pattern matching approssimato
end for
Costruisci lo stato iniziale $q_0$
for $i = 1$ to $s$ do
    Aggiungi una transizione da $q_0$ a $q_0^i$ e da $q_0$ a $q_1^i$ etichettata come la transizione $q_0^i \rightarrow q_1^i$
end for
```

Baker and Bird

Definition

Sia π il dizionario ottenuto da PA, $\pi = \{p_i | p_i$ è la i-esima colonna di PA$\}$
Sia $\mathcal{A}$ l'automa ottenuto da PA con l'algoritmo di $Aho - Corasick$
Sia TA' il *textarray* ottenuto lanciando $\mathcal{A}$ su ogni colonna di TA e salvando lo stato ad ogni iterazione dell'automa

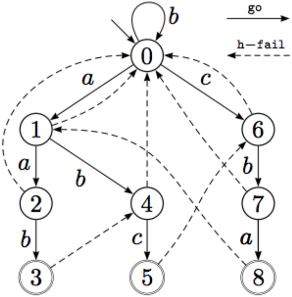
- Linearizzare TA' ottenendo T
- Linearizzare PA ottenendo P
- Usare KMP su P e T
- La complessità finale è $\mathcal{O}(mm' + nn')$

Esempio

$$PA = \begin{array}{|c|c|c|}\hline a & c & a \\ \hline b & b & a \\ \hline c & a & b \\ \hline \end{array},\ TA = \begin{array}{|c|c|c|c|c|c|c|}\hline b & b & a & b & b & a & b \\ \hline a & a & c & a & c & b & a \\ \hline b & b & b & a & c & a & c \\ \hline a & c & a & b & b & a & b \\ \hline c & a & a & c & a & b & a \\ \hline b & b & b & b & a & c & c \\ \hline a & c & c & a & b & a & b \\ \hline \end{array},\ |PA| = (3 \times 3),\ |TA| = (7 \times 7).$$

Esempio

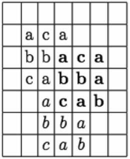
$$PA = \begin{array}{|c|c|c|} \hline a & c & a \\ b & b & a \\ c & a & b \\ \hline \multicolumn{1}{c}{5} & \multicolumn{1}{c}{8} & \multicolumn{1}{c}{3} \end{array}$$



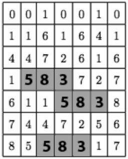
Esempio

$$TA=\begin{array}{|c|c|c|c|c|c|c|}
\hline
b & b & a & b & b & a & b \\
a & a & c & a & c & b & a \\
b & b & b & a & c & a & c \\
a & c & a & b & b & a & b \\
c & a & a & c & a & b & a \\
b & b & b & b & a & c & c \\
a & c & c & a & b & a & b \\
\hline
\end{array}
\qquad
TA'=\begin{array}{|c|c|c|c|c|c|c|}
\hline
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
1 & 1 & 6 & 1 & 6 & 4 & 1 \\
4 & 4 & 7 & 2 & 6 & 1 & 6 \\
1 & 5 & 8 & 3 & 7 & 2 & 7 \\
6 & 1 & 1 & 5 & 8 & 3 & 8 \\
7 & 4 & 4 & 7 & 2 & 5 & 6 \\
8 & 5 & 5 & 8 & 3 & 1 & 7 \\
\hline
\end{array}$$

Esempio



	a	c	a			
	b	b	**a**	**c**	**a**	
	c	a	**b**	**b**	**a**	
		a	**c**	**a**	**b**	
		b	*b*	*a*		
		c	*a*	*b*		



0	0	1	0	0	1	0
1	1	6	1	6	4	1
4	4	7	2	6	1	6
1	**5**	**8**	**3**	7	2	7
6	1	1	**5**	**8**	**3**	8
7	4	4	7	2	5	6
8	5	**5**	**8**	**3**	1	7