

Recherche approchée de motifs courts

Recherche exacte

On recherche toutes les occurrences d'un mot x de longueur m dans un texte y de longueur n .

On considère $n+1$ vecteurs de m bits : R_{-1}^0 ,
 R_0^0, \dots, R_{n-1}^0 .

Le vecteur R_j^0 correspond au traitement de la lettre $y[j]$: il contient les informations sur tous les préfixes de x qui se terminent à la position j sur y :

$$R_j^0[i] = \begin{cases} 0 & \text{si } x[0..i] = y[j-i..j] \\ 1 & \text{sinon} \end{cases}$$

pour $0 \leq i \leq m-1$.

$x = \text{AATAA}$

$y = \text{CAAATAAG}$

$j = 6$

CAAATAAG

A 0

AA 0

AAT 1

AATA 1

AATAA 0

R_6^0

Lorsque $R_j^0[m-1] = 0$ cela signifie que x apparaît dans y à la position (droite) j .

On initialise $R_{-1}^0[i]$ à 1 pour $0 \leq i \leq m-1$.

Pour $0 \leq j \leq n-1$, le vecteur R_j^0 s'exprime au moyen du vecteur R_{j-1}^0 comme suit :

$$R_j^0[i] = \begin{cases} 0 & \text{si } R_{j-1}^0[i-1] = 0 \text{ et } x[i] = y[j] \\ 1 & \text{sinon} \end{cases}$$

pour $1 \leq i \leq m-1$ et

$$R_j^0[0] = \begin{cases} 0 & \text{si } x[0] = y[j] \\ 1 & \text{sinon} \end{cases}$$

Pour cela on note, pour $a \in A$, S_a le vecteur de m bits défini par

$$S_a = \begin{cases} 0 & \text{si } x[i] = a \\ 1 & \text{sinon} \end{cases}$$

Le vecteur S_a est le vecteur caractéristique des positions de la lettre a sur le mot x .

Lemme 1

Pour $0 \leq j \leq n-1$, le calcul de R_j^0 se réduit à deux opérations logiques, un décalage et une disjonction :

$$R_j^0 = (1 \dashv R_{j-1}^0) \vee S_{y[j]} .$$

Preuve

$R_j^0[i]$ pour $0 \leq i \leq m-1$ signifie que $x[0..i]$ est un suffixe de $y[0..j]$ donc

- $x[0..i-1]$ doit être un suffixe de $y[0..j-1]$ donc $R_{j-1}^0[i-1] = 0$;
- $x[i] = y[j]$ donc $S_{y[j]}[i] = 0$

et $R_j^0[0] = 0$ lorsque $S_{y[j]}[0] = 0$. □

Exemple

$x = \text{AATAA}, m = 5$

$y = \text{CAAATAATAGAA}, n = 12$

i	$x[i]$	S_A	S_C	S_G	S_T
0	A	0	1	1	1
1	A	0	1	1	1
2	T	1	1	1	0
3	A	0	1	1	1
4	A	0	1	1	1

j	-1	0			1			2			3			4		5	6	7	8	9	10	11	
$y[j]$		C			A			A			A			T		A	A	T	A	G	A	A	
$R_j^0[0]$	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0
$R_j^0[1]$	1	1	1	1	1	0	1	0	0	0	0	0	0	0	1	1	1	0	1	1	1	1	0
$R_j^0[2]$	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0	1	1	0	1	1	1	1
$R_j^0[3]$	1	1	1	1	1	0	1	1	0	1	1	0	1	1	1	1	0	1	1	0	1	1	1
$R_j^0[4]$	1	1	1	1	1	0	1	1	0	1	1	0	1	1	1	1	1	0	1	1	1	1	1

algo SHIFT-OR-EXACT(x, m, y, n)

pour chaque lettre $a \in A$ faire

$$S_a \leftarrow 1^m$$

pour $i \leftarrow 0$ à $m-1$ faire

$$S_{x[i]} \leftarrow 0$$

$$R^0 \leftarrow 1^m$$

pour $j \leftarrow 0$ à $n-1$ faire

$$R^0 \leftarrow (1 \dashv R^0) \vee S_{y[j]}$$

si $R^0[m-1] = 0$ alors

signaler une occurrence de x

Si $m \leq$ longueur d'un mot machine alors les vecteurs S_a et R^0 peuvent être représentés par des entiers.

Proposition 2

Lorsque la longueur du mot x est inférieure au nombre de bits d'un mot machine alors la phase de prétraitement de l'algorithme **SHIFT-OR-EXACT**(x, m, y, n) s'exécute en temps $\Theta(\text{card } A)$ dans un espace $\Theta(\text{card } A)$. La phase de recherche s'exécute en temps $\Theta(n)$.

Recherche avec une inégalité

On considère les vecteurs R_j^0 pour $-1 \leq j \leq n-1$ comme précédemment.

On introduit les vecteurs R_j^1 pour $0 \leq j \leq n-1$ comme suit :

$$R_j^1[i] = \begin{cases} 0 & \text{si } Ham(x[0..i], y[j-i..j]) \leq 1 \\ 1 & \text{sinon} \end{cases}$$

et $R_{-1}^1 = 01^{m-1}$.

Lemme 3

Pour $0 \leq j \leq n-1$, les vecteurs R_j^1 vérifient la relation

$$R_j^1 = ((1 \dashv R_{j-1}^1) \vee S_{y[j]}) \wedge (1 \dashv R_{j-1}^0).$$

Preuve

Trois cas :

1. $x[0..i-1] = y[j-i..j-1]$ ($R_{j-1}^0[i-1] = 0$) alors substituer $x[i]$ à $y[j]$ crée une occurrence avec au plus une inégalité de $x[0..i]$ comme suffixe de $y[0..j]$. D'où $R_j^1[i] = 0$ lorsque $R_{j-1}^0[i-1] = 0$.

2. $x[0..i-1]$ apparaît en suffixe de $y[0..j-1]$ avec au plus une inégalité ($R_{j-1}^1[i-1] = 0$). Si $x[i] = y[j]$ alors $x[0..i]$ apparaît en suffixe de $y[0..j]$ avec au plus une inégalité. D'où $R_j^1[i] = 0$ lorsque $R_{j-1}^1[i-1] = 0$ et $x[i] = y[j]$.

3. ni les conditions du cas ni celles du cas 2 ne sont vérifiées. D'où $R_j^1[i] = 1$ dans ce cas. \square

Pour la recherche avec au plus k inégalités on utilise $k+1$ vecteurs R^0, R^1, \dots, R^k .

Les valeurs de R^0 sont calculées comme dans l'algorithme **SHIFT-OR-EXACT**.

Les valeurs des vecteurs R^1, R^2, \dots, R^k sont calculées conformément au lemme 3.

Lorsque $R_j^k[m-1] = 0$ cela signifie que x apparaît dans y à la position (droite) j avec au plus k inégalités.

```

algo SHIFT-OR-INEG( $x, m, y, n, k$ )
  pour chaque lettre  $a \in A$  faire
     $S_a \leftarrow 1^m$ 
  pour  $i \leftarrow 0$  à  $m-1$  faire
     $S_{x[i]} \leftarrow 0$ 
   $R^0 \leftarrow 1^m$ 
  pour  $\ell \leftarrow 1$  à  $k$  faire
     $R^\ell \leftarrow (1 \dashv R^{\ell-1})$ 
  pour  $j \leftarrow 0$  à  $n-1$  faire
     $T \leftarrow R^0$ 
     $R^0 \leftarrow (1 \dashv R^0) \vee S_{y[j]}$ 
    pour  $\ell \leftarrow 1$  à  $k$  faire
       $T' \leftarrow R^\ell$ 
       $R^\ell \leftarrow ((1 \dashv R^\ell) \vee S_{y[j]}) \wedge (1 \dashv T)$ 
       $T \leftarrow T'$ 
    si  $R^k[m-1] = 0$  alors
      signaler une occurrence de  $x$ 

```

j	-1	0				1				2			
$y[j]$		C				A				A			
$R_y[0]$	0	0	1	0	0	0	0	0	0	0	0	0	0
$R_y[1]$	1	0	1	1	1	0	0	1	0	0	0	0	0
$R_y[2]$	1	1	1	1	1	1	1	1	1	0	1	1	1
$R_y[3]$	1	1	1	1	1	1	0	1	1	1	0	1	1
$R_y[4]$	1	1	1	1	1	1	0	1	1	1	0	1	1

Recherche avec une insertion

On adapte les vecteurs R_j^1 au problème.

Lemme 4

Pour $0 \leq j \leq n-1$, les vecteurs R_j^1 , correspondant à la recherche approchée avec une insertion, vérifient la relation

$$R_j^1 = ((1 \dashv R_{j-1}^1) \vee S_{y[j]}) \wedge R_{j-1}^0.$$

Preuve

Trois cas :

1. $x[0..i] = y[j-i-1..j-1]$ ($R_{j-1}^0[i] = 0$) alors insérer $y[j]$ crée une occurrence avec au plus une insertion de $x[0..i]$ comme suffixe de $y[0..j]$.
D'où $R_j^1[i] = 0$ lorsque $R_{j-1}^0[i] = 0$.

2. $x[0..i-1]$ apparaît en suffixe de $y[0..j-1]$ avec au plus une insertion ($R_{j-1}^1[i-1] = 0$). Si $x[i] = y[j]$ alors $x[0..i]$ apparaît en suffixe de $y[0..j]$ avec au plus une insertion. D'où $R_j^1[i] = 0$ lorsque $R_{j-1}^1[i-1] = 0$ et $x[i] = y[j]$.

3. ni les conditions du cas ni celles du cas 2 ne sont vérifiées. D'où $R_j^1[i] = 1$ dans ce cas. \square

Recherche avec une suppression

On adapte les vecteurs R_j^1 au problème.

Lemme 5

Pour $0 \leq j \leq n-1$, les vecteurs R_j^1 , correspondant à la recherche approchée avec une suppression, vérifient la relation

$$R_j^1 = ((1 \dashv R_{j-1}^1) \vee S_{y[j]}) \wedge (1 \dashv R_j^0).$$

Preuve

Trois cas :

1. $x[0..i-1] = y[j-i+1..j]$ ($R_j^0[i-1] = 0$) alors supprimer $x[i]$ crée une occurrence avec au plus une insertion de $x[0..i]$ comme suffixe de $y[0..j]$. D'où $R_j^1[i] = 0$ lorsque $R_j^0[i-1] = 0$.

2. $x[0..i-1]$ apparaît en suffixe de $y[0..j-1]$ avec au plus une suppression ($R_{j-1}^1[i-1] = 0$). Si $x[i] = y[j]$ alors $x[0..i]$ apparaît en suffixe de $y[0..j]$ avec au plus une suppression. D'où $R_j^1[i] = 0$ lorsque $R_{j-1}^1[i-1] = 0$ et $x[i] = y[j]$.

Recherche avec k différences

Pour $1 \leq \ell \leq k$ et $0 \leq j \leq n-1$

$$\begin{aligned} R_j^\ell &= ((1 \dashv R_{j-1}^\ell) \vee S_{y[j]}) \wedge \\ &\quad (1 \dashv R_j^{\ell-1}) \wedge \\ &\quad (1 \dashv R_{j-1}^{\ell-1}) \wedge \\ &\quad R_{j-1}^{\ell-1} \\ &= ((1 \dashv R_{j-1}^\ell) \vee S_{y[j]}) \wedge \\ &\quad (1 \dashv (R_{j-1}^{\ell-1} \wedge R_j^{\ell-1})) \wedge \\ &\quad R_{j-1}^{\ell-1} \end{aligned}$$

```

algo SHIFT-OR-DIFF( $x, m, y, n, k$ )
  pour chaque lettre  $a \in A$  faire
     $S_a \leftarrow 1^m$ 
  pour  $i \leftarrow 0$  à  $m-1$  faire
     $S_{x[i]} \leftarrow 0$ 
   $R^0 \leftarrow 1^m$ 
  pour  $\ell \leftarrow 1$  à  $k$  faire
     $R^\ell \leftarrow (1 \dashv R^{\ell-1})$ 
  pour  $j \leftarrow 0$  à  $n-1$  faire
     $T \leftarrow R^0$ 
     $R^0 \leftarrow (1 \dashv R^0) \vee S_{y[j]}$ 
    pour  $\ell \leftarrow 1$  à  $k$  faire
       $T' \leftarrow R^\ell$ 
       $R^\ell \leftarrow ((1 \dashv R^\ell) \vee S_{y[j]}) \wedge (1 \dashv (T \wedge R^{\ell-1})) \wedge T$ 
       $T \leftarrow T'$ 
    si  $R^k[m-1] = 0$  alors
      signaler une occurrence de  $x$ 

```

Théorème 6

Lorsque la longueur du mot x est inférieure au nombre de bits d'un mot machine alors la phase de prétraitement de l'algorithme **SHIFT-OR-DIFF**(x, m, y, n, k) s'exécute en temps $\Theta(k + \text{card } A)$ dans un espace $\Theta(k + \text{card } A)$. La phase de recherche s'exécute en temps $\Theta(kn)$.

Références

- Wu & Manber
Fast text searching allowing errors
Commun. ACM, 35(10):83-91, 1992.
- Baeza-Yates & Gonnet
A new approach to text searching
Commun. ACM, 35(10):74-82, 1992.