KNUTH-MORRIS-PRATT

Antoine DEQUAY

21 septembre 2022

Notes

- Prof : .
- Leçon : 907, 921, 926.
- Références :
 - CORMEN.

On cherche à trouver toutes les occurrences d'un motif P de taille m dans un texte T de taille n .

Pour cela, on commence par un pré-traitement sur P : on calcul le tableau auxiliaire π (dont l'indiciage commence à 1) qui stocke en case i la longueur du plus grand suffixe propre du sous-mot de P qui se termine à l'indice i et qui est un préfixe propre de P (C'est la longueur du plus grand bord du sous-mot considéré).

Exemple 1

i	1	2	3	4	5	6
P	a	b	c	a	b	d
π	0	0	0	1	2	0

Pour cela, on utilise l'algorithme suivant

Algorithm 1: CALCUL- $\pi(P)$

```

 $m = \text{longueur}(P);$ 
 $\pi = \text{tableau de taille } m;$ 
 $\pi[1] = 0;$ 
 $k = 0;$ 
for  $q \leftarrow 2$  to  $m$  do
  while  $k > 0$  et  $P[k + 1] \neq P[q]$  do
     $k = \pi[k]$ 
  if  $P[k + 1] == P[q]$  then
     $k = k + 1$ 
     $\pi[q] = k$ 
return  $\pi$ .
```

Calculons sa complexité temporelle par analyse amortie via la méthode d'agrégat : on cherche à majorer le nombre d'itérations de la boucle while. On remarque que :

- k est initialisé à 0 et augmente d'au plus une unité par itération de la boucle for, donc vaut au plus $m - 1$,
- On a l'invariant de boucle $k < q$ grâce à la remarque précédente, donc $\pi[q] < q$ pour tout q , donc la boucle while ne peut que faire diminuer k ,
- $k > 0$, par définition de la boucle while.

Comme k ne peut être diminué qu'au sein de la boucle while, on ne peut entrer dans celle-ci qu'au plus $m - 1$ fois.

On a donc une complexité en $O(m)$!

Algorithm 2: KMP(P, T)

```

n = longueur(T);
m = longueur(P);
π = Calcul-π(P);
q = 0;
for i ← 1 to n do
  while q > 0 et P[q + 1] ≠ T[i] do
    q = π[q]
  if P[q + 1] == T[i] then
    q = q + 1
  if q == m then
    Afficher "le motif apparaît en position" i - m;
    q = π[q]

```

Exemple 2 k sert de pointeur sur P :

i	1	2	3	4	5	6	7	8	9
T	a	b	c	a	b	c	a	b	d
		↑	↑	↑	↑	↑	×		
P	a	b	c	a	b	d			
k	1	2	3	4	5				
k					2				
P				a	b	c	a	b	d
k				2	3	4	5	6	

Le même raisonnement que tout à l'heure permet directement de conclure qu'on a une complexité temporelle en $O(n)$.

C'est donc une amélioration de l'algorithme de MORRIS-PRATT! On a une complexité spatiale en $O(m)$ et temporelle en $O(m + n)$, alors que MP se fait en $O(m|\Sigma|)$ en espace!