Algorithme de Karp-Rabin

Université Joseph Fourier GRENOBLE
UFR IM²AG

Concepts : Sous-chaîne de caractère
Méthodes : Fonction de hachage

Présentation

L'objectif est de rechercher une sous-chaîne de caractères M de longueur m appelée motif dans une chaîne de caractères C de longueur n. Un algorithme naïf consiste à comparer successivement le motif à toutes les sous-chaînes de C

On veut rechercher un motif M de longueur m dans un texte T de longueur n. Les textes sont exprimés dans un alphabet de K lettres. Le texte et le motif seront donnés dans deux tableaux, ces tableaux seront indicés à partir de 1.

Question 1 : Test d'égalité de mots

Écrire un algorithme qui teste l'égalité de deux mots. Calculer le coût de votre algorithme en nombre de comparaisons de caractères.

Afin de rechercher plus efficacement un motif, on dispose d'une fonction de hachage H sur les mots. Si A est un tableau de lettres $h(A, i, j)$ représente la valeur de hachage du sous-mot du tableau A défini de l'indice i à j inclus. On suppose que la fonction h prend des valeurs entières avec $0 \leqslant h(A, i, j) \leqslant H-1$.

```
SOUS-CHAINE(x,n)
  Données : T texte de taille n dans lequel le motif M de taille m est recherché (tableaux de
            caractères)
  Résultat : Les positions du motif M s'il est dans le texte

  hmotif=h (M[1..m])
  for i = 1 to n − m do
      hcourant = h (T[i..i+m-1])
      if (courant == hmotif) Commentaire 1
          if égalité (T[i..i+m-1],M[1..m]) Commentaire 2
              write i
```

Algorithme 1 : Algorithme de Karp-Rabin

Question 2 : Commentaires

Écrire les commentaires 1 et 2 et calculer le coût maximal de cet algorithme en nombre de comparaisons de caractères.

Question 3 : Propriétés de h

Quelles sont les propriétés que h doit vérifier pour que cet algorithme soit efficace en moyenne ? Calculer dans ce cas le coût moyen de cet algorithme.

Algorithme de Karp-Rabin

On suppose que chaque caractère est codé par un entier et que la fonction h s'écrit pour un mot $A = (a_1, \cdots, a_m)$

$$h(A[1..m]) = (a_1 + a_2 + \cdots + a_m) \text{ modulo } p,$$

avec p un entier donné.

Question 4 : h additive

Expliquer le rôle de l'entier p. Comment calculer $h(T[i..i+m-1])$ en fonction de $h(T[i-1..i+m-2])$, en déduire que l'algorithme ne nécessite que deux consultations du tableau T à chaque itération.

On se donne l'alphabet $\{A, B, C, D\}$ avec le codage standard $\{0, 1, 2, 3\}$. Soit

$$T = ABCADBBACABAC,\ M = CAB.$$

Question 5 : Exemple

Dérouler l'algorithme pour $p = 9$ et évaluer le nombre de comparaisons de caractères effectuées. Quel inconvénient voyez-vous à cette fonction h ?

On considère maintenant une nouvelle fonction h_1 construite sur le mot A par

$$h_1(A[1..m]) = (a_1.d^{m-1} + a_2.d^{m-2} + \cdots + a_{m-1}.d^1 + a_m.d^0) \text{ modulo } p,$$

avec d un entier donné (base).

Question 6 : h polynomiale

Expliquer le rôle de l'entier d. Comment calculer $h(T[i..i+m-1])$ en fonction de $h(T[i-1..i+m-2])$, en déduire que la ligne 13 de l'algorithme ne nécessite que deux consultations du tableau T.

Question 7 : Exemple

Dérouler l'algorithme sur le même exemple ci-dessus pour $p = 9$ et $d = 4$. Évaluer le nombre de comparaisons de caractères effectuées. Commenter votre résultat.