TD d'algorithmique numéro 13 - Recherche de motif
Madame Natacha Portier et Monsieur Daniel Hirschkoff
8 janvier 2000

Question 1 ⋆ Recherche de motif avec des automates finis ⋆

Un automate fini M est défini par la donnée d'un quintuplet $(Q, q_0, A, \Sigma, \delta)$:

- Q est un ensemble fini d'*états*, parmi lesquels on distingue $q_0 \in Q$, l'*état initial*, et $A \subseteq Q$, l'ensemble des états terminaux ;
- Σ est un *alphabet* fini ;
- $\delta : Q \times \Sigma \to Q$ est la *fonction de transition* de M.

On le représente graphiquement par son graphe de Moore : les sommets sont les états, les arêtes sont étiquetées par des lettres et représentent la fonction de transition.

Question 1.1 ⋆ Langage reconnu par un automate ⋆

À tout automate fini est associée une fonction $\phi : \Sigma^* \to Q$, appelée *fonction d'état final* qui associe à une chaîne w l'état dans lequel M se trouve après avoir lu celle-ci. On dit qu'un automate *reconnaît* un mot w si $\phi(w)$ est un état final.

Donner un automate sur l'alphabet $\{0; 1\}$ qui reconnaisse les mots se terminant par un nombre pair de 1.

Question 1.2 ⋆ Fonction suffixe ⋆

On veut trouver toutes les occurrences d'une chaîne de caractères, appelée aussi motif, dans un texte. C'est une fonctionnalité qui existe par exemple dans un éditeur de textes, ou dans la commande `egrep`. On va "programmer" un automate fini pour résoudre le problème de la recherche d'un motif $P[1 \dots m]$ dans un texte. Pour ce faire, on aura besoin de la *fonction suffixe* σ associée à P. Pour tout entier k compris entre 1 et m, le mot P_k est constitué des k premières lettres de P c'est-à-dire $P_k = P[1]...P[k]$. Par convention P_0 est le mot vide. On définit alors $\sigma(x)$ comme le plus grand k tel que P_k soit un suffixe de x :

$$\sigma(x) \ = \ \max\{k : P_k \sqsupset x\}$$

(la notation $P_k \sqsupset x$ signifie que P_k est un suffixe de x).

1. Pourquoi la fonction σ est-elle bien définie ?
   Quelle relation peut-on établir entre $\sigma(x)$ et $\sigma(y)$, pour $x \sqsupset y$?
2. Montrer que pour toute chaîne x et pour tout caractère a, on a $\sigma(xa) \leq \sigma(x) + 1$.
3. Montrer que si $q = \sigma(x)$, alors $\sigma(xa) = \sigma(P_q a)$.

Question 1.3 ⋆ Construction de l'automate ⋆

Décrire un automate pouvant être utilisé pour reconnaître un motif donné $P[1] \dots P[m]$: nombre d'états, fonction de transition.

Construire l'automate pour $m = 3$, $\Sigma = \{a, b\}$ et $P = aba$. Le faire tourner sur le mot $M = abbababaaaba$ et retenir les indices i pour lesquels la lecture de M_i fait arriver dans un état final.

Question 1.4 ⋆ Fonction d'état final ⋆

Si Σ est un alphabet, c'est-à-dire un ensemble fini de lettres, on note Σ^* l'ensemble des mots sur Σ, i.e. l'ensemble des suites finies de Σ. La suite de longueur nulle est le mot vide.

Montrer que si ϕ est la fonction d'état final (cf question 1.1) associée à l'automate défini ci-dessus, automate que l'on fait tourner sur un texte $T[1\ldots n]$, alors, pour $i = 0, 1, \ldots, n$:

$$\phi(T_i) = \sigma(T_i)$$

Question 1.5 ⋆ Complexité ⋆

Écrire deux procédures pour le calcul de la fonction de transition de l'automate et pour la recherche de motif à l'aide d'un automate. Proposer une valeur pour la complexité résultante de cette approche.

Question 2 ⋆ Knuth Morris Pratt ⋆

On veut éviter le calcul de la fonction de transition de l'automate fini, de manière à améliorer l'efficacité de la recherche de motif.

On part de l'observation suivante: lorsque l'on cherche le motif *abacd* dans le texte *dfababacabacd*, au moment où l'on se rend compte que le décalage "3" ne convient pas (c'est à dire au moment d'examiner le quatrième caractère du motif, cf Figure 1), plutôt que de reprendre l'inspection avec le décalage suivant, on exploite la connaissance que l'on a du motif pour passer directement à un décalage plus "plausible".

d	f	a	b	a	b	a	c	a	b	a	c	d

a	b	a	c	d	$\rightsquigarrow$

Figure 1: Recherche de motif

Par exemple, sur la Figure 1, comme le deuxième caractère du motif est différent du premier, on peut sauter directement le décalage "4", car on sait qu'il ne conviendra pas.

Fonction préfixe En se basant sur les observations qui viennent d'être faites, proposer une manière d'associer à tout motif P de longueur m une fonction π (que l'on appellera la *fonction préfixe*), de $\{1; 2; \ldots; m\}$ dans $\{0; 1; \ldots; m-1\}$, qui "implémente" le saut de décalage esquissé ci-dessus. L'idée sera d'utiliser cette fonction dans une version modifiée de l'algorithme qui exécute l'automate pour la recherche de motif, de manière à en améliorer les performances (la fonction π pouvant être précalculée, avant que la recherche de motif ne commence).

Question 2.1 ⋆ Définition ⋆

Écrire une procédure qui calcule la fonction π.

Question 2.2 ⋆ Correction ⋆

On pose $\pi^*[q] = \{q, \pi[q], \pi^2[q], \ldots, \pi^t[q]\}$, la séquence s'arrêtant lorsque l'on atteint $\pi^t[q] = 0$.

1. Montrer que pour $q = 1, 2, \ldots, m$:

$$\pi^*[q] \;=\; \{k : P_k \sqsupset P_q\}\,.$$

2. Montrer que pour $q = 1, 2, \ldots, m$, si $\pi[q] > 0$ alors $\pi[q] - 1 \in \pi^*[q-1]$.
   Pour $q = 2, 3, \ldots, m$ on définit le sous ensemble $E_{q-1} \subseteq \pi^*[q-1]$ par
$$E_{q-1} = \{k \mid k \in \pi^*[q-1] \text{ et } P[k+1] = P[q]\}$$
3. Montrer que pour $q = 2, 3, \ldots, m$:
$$\pi[q] = \begin{cases} 0 & \text{si } E_{q-1} = \emptyset \\ 1 + \max\{k \in E_{q-1}\} & \text{si } E_{q-1} \neq \emptyset \end{cases}$$
4. Utiliser les raisonnements que l'on vient d'effectuer pour montrer la correction de la procédure calculant π.

Question 2.3 ⋆ Complexité ⋆

Comment évaluer la complexité de la procédure qui calcule π?

Question 2.4 ⋆ Reconnaissance de motifs ⋆

Écrire l'algorithme de recherche de motifs modifié pour utiliser les informations exploitées dans la définition π. En évaluer la complexité.

Question 2.5 ⋆ Correction du tout ⋆

Comment se justifie la correction de l'algorithme résultant?

Question 3 ⋆ Rabin-Karp ⋆

On s'intéresse maintenant à une autre approche pour la recherche de motif, en travaillant avec l'alphabet $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, et en pensant aux mots de l'alphabet comme à des nombres écrits en base dix.

On cherche un motif $P[1 \ldots m]$ (tableau de m chiffres), auquel correspond une valeur décimale p dans un texte $T[1 \ldots n]$ (tableau de n chiffres) ; on note t_s la valeur décimale correspondant à la sous-chaîne $T[s+1 \ldots s+m]$ de longueur m . Lorsque $t_s = p$, on dira que *s est un décalage valide.*

Question 3.1

Comment calculer p et t_0 en $O(m)$?

Comment calculer les valeurs restantes $t_1, t_2, \ldots, t_{n-m}$ en $O(n-m)$?

À supposer que l'on fait abstraction de la possibilité pour p et les t_i d'être des nombres très grands, évaluer la complexité de cette méthode pour résoudre le problème de la recherche de motif.

Question 3.2

Seulement il n'est pas raisonnable de faire abstraction de la taille des nombres manipulés. On se restreint à raisonner sur un espace de mots réduit en travaillant modulo un certain entier q. On ne compare les mots que s'ils sont égaux modulo q. En pratique, lorsque l'on veut travailler avec un alphabet $d-$aire $\Sigma = \{0, 1, \ldots, d-1\}$, on choisit q de telle manière que dq tienne dans un mot d'ordinateur (ici d est égal à dix).

Bien entendu, puisque l'on travaille modulo q, l'exploration du texte que l'on fait ne fournira une réponse fiable pour un décalage donné que si la réponse est "NON". L'idée de la procédure de Rabin-Karp est donc de parcourir le texte en faisant la recherche de motif "modulo q", et de ne vérifier explicitement la correspondance entre motifs qu'en cas de réponse positive.

Écrire l'algorithme.

Quel est son temps d'exécution dans le pire des cas ?

Donner une configuration pour laquelle les vérifications prennent effectivement un temps correspondant à l'estimation ci-dessus.

Dans la pratique on s'attend à ce que ce genre de cas soit peu fréquent.