Colloque U.P.S.

Algorithmique et programmation

lundi 3 mai 1999

Des carrés en tous sens

(To be or not to be squarefree)

Bruno Petazzoni

Lycée M. Berthelot – 94100 Saint-Maur

bpetazzoni@ac-creteil.fr

Des carrés en tous sens

- Genèse de cet exposé
- Vocabulaire
- Le motif α^2
- Quelques résultats
- Notre problème
- Quatre solutions
- Carrés abéliens
- Algorithmique du génome

Genèse de cet exposé

Objectif : concevoir un sujet de DS d'informatique, dans le style CCM ou CCP (trois petits problèmes)

Idée marrante : relier ces problèmes par un fil conducteur, en l'occurrence le mot *carré*

- Problème 1 : racine carrée d'un langage
- Problème 2 : circuits combinatoires pour calculer le carré d'un nombre
- Problème 3 : détection de facteurs carrés dans une chaîne de caractères

Je n'ai pas eu le temps de réaliser la troisième partie pour la date prévue du DS. Voici quelques éléments réunis depuis. Le titre de l'exposé n'est relié au contenu que par le mot carré.

Vocabulaire

$\Sigma = \{a, b, c, \ldots\}$ est un alphabet ; nous dirons que ses lettres sont des *constantes*.

Soit $\Delta = \{\alpha, \beta, \gamma, \ldots\}$ un deuxième alphabet, de *variables* ; un *motif sans constantes* est un mot sur l'alphabet Δ. Nous dirons que $u \in \Sigma^*$ contient le motif p s'il existe un morphisme $\varphi : \Delta^* \mapsto \Sigma^*$ non-effaçant ($\varphi(\alpha) \neq \varepsilon$ pour toute variable α) tel que $\varphi(p)$ soit facteur de u ; dans le cas contraire, on dira que u *évite* le motif p.

Un motif p est *évitable* s'il existe une infinité de mots qui l'évitent ; ou, ce qui revient au même, s'il existe un mot infini qui l'évite. Sinon, p est *inévitable*

Le motif α^2

Un *carré* est un mot de la forme u^2, où u est un mot non vide ; exemple : *abcabc*.

Dire qu'un mot est sans facteur carré revient à dire qu'il évite le motif α^2.

Sur un alphabet à deux lettres, le motif α^2 est inévitable ; en revanche, il est évitable sur un alphabet à trois lettres (Thue 1912).

La solution minimale du jeu des tours de Hanoï, considérée comme mot sur un alphabet de six lettres, évite le motif α^2 (Allouche, Astoorian, Randall et Shallit, 1994).

Un *mot de Langford* est un mot sur l'alphabet $\mathbb{N}^*$ tel que deux occurrences consécutives d'une même lettre n soient toujours séparées par n lettres exactement. Les mots de Langford évitent le motif α^2 (Berstel).

Quelques résultats

Il est immédiat que l'ensemble des mots qui évitent le motif α^2 n'est pas algébrique (appliquer un lemme de pompage).

Son complémentaire ne l'est pas non plus (voir [9] ou [17]).

Un lemme utile [8] : soient u^2, v^2 et w^2 trois préfixes d'un même mot. On suppose u primitif et $|u| < |v| < |w|$. Alors $|u| + |v| \leqslant |w|$.

Exemple :

$$\mathsf{0100101001001010}$$

Compter les carrés distincts

Notons $g(n)$ la longueur maximale d'un mot sur l'alphabet binaire $\Sigma = \{0{,}1\}$ contenant au plus n facteurs carrés distincts.

On a clairement $g(0) = 3$; on peut vérifier que $g(1) = 7$ et $g(2) = 18$ (bon exercice de programmation).

Fraenkel et Simpson [10] ont exhibé un mot infini ne contenant pas plus de trois facteurs carrés distincts ; donc $g(n) = \infty$ pour $n \geqslant 3$.

Nombre de mots sans carrés

Notons $C(n)$ le nombre de mots sans carrés de longueur n, sur l'alphabet $\{a, b, c\}$. On a clairement $C(m+n) \leqslant C(m)C(n)$; on vérifie manuellement que $C(1) = 3$, $C(2) = 6$ et $C(3) = 12$, dont on déduit $C(n) \leqslant kr^n$ avec $r = \sqrt[3]{12} \approx 2.29$ et $k = \max(3/r, 6/r^2) \approx 1.31$.

En examinant les valeurs de $C(n)$ pour $n \leqslant 22$, Brandenburg a établi la majoration $C(n) \leqslant 6{c_2}^n$ avec $c_2 \leqslant 1.38$ ainsi que la minoration $C(n) \geqslant 6{c_1}^n$ avec $c_1 \geqslant 1.032$ (avec un argument un peu plus subtil, cf. [4]) .

Pour ce qui concerne les mots infinis, Kakutani a montré qu'il existait une quantité non dénombrable de mots infinis sans carrés sur l'alphabet $\{a, b, c\}$, deux quelconques d'entre eux étant asymptotiquement distincts.

Notre problème

Bean, Ehrenfeucht et McNulty ont eu besoin en 1979 de vérifier que 36 mots, de longueur comprise entre 815 et 857, évitaient le motif α^2 ; pour ce faire, ils ont utilisé un PDP 11/45 qui a tourné pendant neuf heures [1].

Notre objectif : décrire un algorithme (aussi efficace que possible) décidant si un mot s évite le motif α^2, c'est-à-dire ne contient aucun facteur carré.

Extension : décrire un algorithme qui détermine tous les facteurs carrés du mot s.

Dans toute la suite, n est la longueur de s.

Solution naïve

Pour chaque couple (i, ℓ) de naturels non nuls vérifiant $i + 2\ell \leqslant n + 1$, comparer $s[i..i + \ell - 1]$ et $s[i + \ell..i + 2\ell - 1]$; en cas d'égalité, s admet le facteur carré $s[i..i + 2l - 1]$.

Le coût est clairement $\mathcal{O}(n^3)$.

Une estimation rapide montre que Bean, Ehrenfeucht et McNulty ont vraisemblablement utilisé cet algorithme.

Solution quadratique (KMP)

Un *bord* d'un mot u est un préfixe propre de u qui est en même temps un suffixe de u.
Exemple : *ababa* possède les bords ε, a et aba ; ce dernier est le *bord maximal*.

Notons $n = |u|$ et $\beta(u, i)$ la longueur du bord maximal de $u[1..i]$ pour $1 \leqslant i \leqslant n$. On peut calculer la suite des $\beta(u, i)$ pour un coût $\mathcal{O}(n)$.

u possède un préfixe carré ssi il existe un indice i tel que $\beta(u, i) \geqslant i/2$. On peut donc décider en temps $\mathcal{O}(n)$ si u possède un préfixe carré.

En appliquant ceci aux suffixes de s, on peut décider en temps $\mathcal{O}(n^2)$ si s possède un facteur carré

À noter : KMP avait été publié officiellement [13] deux ans avant [1], et avait fait l'objet d'un rapport technique dès 1970.

Digression

Quelques exercices utilisant KMP :

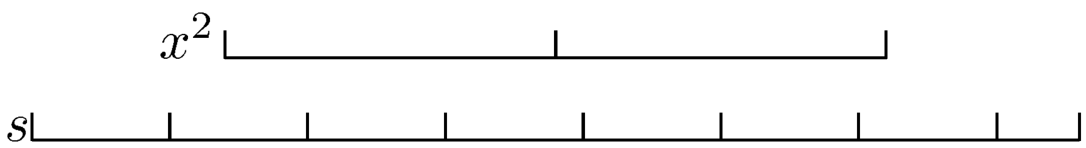
- décrire un algorithme décidant si un mot de longueur n est primitif, pour un coût $\mathcal{O}(n)$
- décrire un algorithme déterminant tous les préfixes palindromes d'un mot de longueur n, pour un coût $\mathcal{O}(n)$
- on dit que deux mots u et v sont *conjugués* s'il existe des mots x et y tels que $u = xy$ et $v = yx$; décrire un algorithme décidant, pour un coût $\mathcal{O}(n)$, si deux mots de même longueur n sont conjugués l'un de l'autre

Solution de coût $\mathcal{O}(n \ln n)$

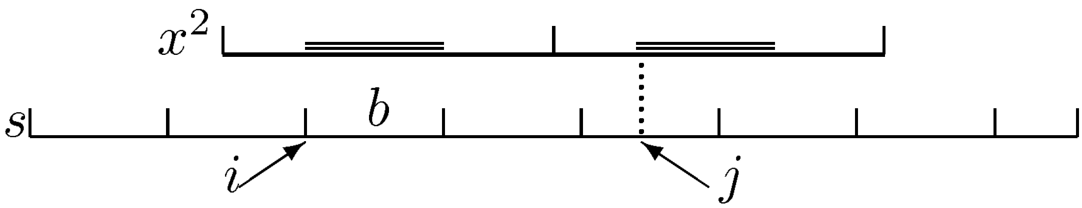
Main et Lorentz [14] ont proposé une méthode de coût $\mathcal{O}(n \ln n)$; elle s'appuie sur le principe *diviser pour régner.*

Soit $\ell \geqslant 1$. Supposons que s ne possède aucun facteur carré x^2 avec $|x| < 2\ell - 1$. Décomposons s en $\lfloor |s|/\ell \rfloor$ *blocs* consécutifs de longueur ℓ.

Observons ce qui se passe si s possède un facteur carré x^2 avec $2\ell - 1 \leqslant |x| < 4\ell - 2$.



Comme $|x| \geqslant 2\ell - 1$, l'un des blocs (noté b) est entièrement inclus dans la première occurrence de x. Notons i la position de ce bloc : $b = s[i..i + \ell - 1]$.



Ce bloc b a nécessairement une deuxième occurrence, dans le facteur

$$f = s[i + 2\ell - 1..i + 4\ell - 2]$$

L'hypothèse faite sur ℓ nous assure que f, qui est de longueur 2ℓ, contient *au plus* deux occurrences de b : s'il y en avait trois, deux d'entre elles se recouvriraient partiellement, si bien que la deuxième occurrence de x contiendrait un carré y^2 avec $|y| < 2\ell - 1$.

Tout est en place pour décrire l'algorithme.

Si l'on ne trouve pas de deuxième occurrence de b, on passe au bloc suivant.

Sinon, pour chaque position j d'une deuxième occurrence de b :

- on détermine la longueur maximale g d'un suffixe commun à $s[1..i-1]$ et $s[1..j-1]$, en n'examinant toutefois que les $3\ell - 2$ caractères les plus à droite
- on détermine la longueur maximale d d'un préfixe commun à $s[i+\ell..n]$ et $s[j+\ell..n]$, en n'examinant toutefois que les $3\ell - 2$ caractères les plus à gauche
- si $g + d \geqslant j - i$, on a trouvé un facteur carré : l'algorithme se termine

Lorsque l'on a examiné tous les blocs de longueur ℓ, on recommence en multipliant ℓ par 2, sauf bien entendu si $n < 4\ell - 2$, auquel cas l'algorithme se termine en décidant que s est sans facteur carré.

Le démarrage de l'algorithme se fait avec $\ell = 1$: l'hypothèse est alors satisfaite sans frais.

L'examen de chaque bloc b, et des éventuelles occurrences de b, a un coût $\mathcal{O}(\ell)$; comme le nombre de blocs est majoré par n/ℓ, le coût associé à chaque valeur de ℓ est $\mathcal{O}(n)$.

Comme ℓ prend pour valeurs successives des puissances de deux inférieures à n, le nombre de balayages est dominé par $\ln n$, si bien que le coût total de l'algorithme est un $\mathcal{O}(n \ln n)$.

Solution de coût $\mathcal{O}(n)$

En fait, il n'existe pas d'algorithme de coût $\mathcal{O}(n)$! Quelle déception…

Ceci dit, si on fixe l'alphabet (hypothèse assez peu contraignante), on peut obtenir un coût *linéaire* ; voir [15], [6] et [7].

Prouvons que, en l'absence d'hypothèse sur l'alphabet, le coût d'un algorithme de détection de facteur carré est un $\Omega(n \ln n)$.

Notons s le mot, de longueur n, dont on veut savoir s'il possède un facteur carré. Un *candidat* est un facteur de s, susceptible d'être un carré ; il est donc de longueur paire.

Soit $k \in [\![1, n/2]\!]$; une comparaison entre s_i et s_{i+k} permet, en cas de différence, d'éliminer k candidats de longueur $2k$. Comme il en existe $n - 2k + 1$, il faudra $\frac{n-2k+1}{2k}$ comparaisons pour éliminer tous les candidats de longueur $2k$.

Pour assurer que s n'a aucun facteur carré, le nombre de comparaisons requis sera (dans le pire des cas) :

$$\sum_{1 \leqslant k \leqslant \lfloor n/2 \rfloor} \frac{n-2k+1}{2k} = \frac{n+1}{2} \sum_{1 \leqslant k \leqslant n/2} \frac{1}{k} - \left\lfloor \frac{n}{2} \right\rfloor$$

Dans cette somme, le premier terme équivaut à $\frac{n \ln n}{2}$ et le deuxième à $\frac{n}{2}$.

On se convaincra que ceci ne vaut que si l'alphabet est infini (imaginer que le mot s est diabolique, et peut changer certaines de ses lettres avant de répondre à vos questions).

Combien de carrés?

Le nombre de facteurs carrés d'un mot de longueur n est au plus quadratique (cas de a^n).

Le lemme de [8] permet d'établir deux résultats :

- un mot de longueur n comporte $\mathcal{O}(n \ln n)$ facteurs carrés *primitifs* (x^2 avec x primitif)
- un mot de longueur n comporte au plus $2n - 2$ facteurs carrés *distincts*

Les mots de Fibonacci réalisent en un certain sens le nombre maximal de facteurs carrés primitifs : f_n, de longueur F_n, en comporte $\frac{2}{5}(3 - \phi)nF_n + \mathcal{O}(F_n)$; voir [11]

Carrés abéliens

Un *carré abélien* est un mot de la forme $\alpha\hat{\alpha}$, où $\hat{\alpha}$ se déduit de α par permutation de ses lettres ; exemple : *abcbcbba*. Nous conviendrons que $\alpha\hat{\alpha}$ est un motif.

On peut éviter le motif $\alpha\hat{\alpha}$ sur un alphabet à cinq lettres [16] et même sur un alphabet à quatre lettres [12].

Montrer que le motif $\alpha\hat{\alpha}$ est inévitable sur un alphabet à trois lettres.

Proposer un algorithme efficace pour décider si un mot évite le motif $\alpha\hat{\alpha}$.

Écrire un programme exhibant un mot de longueur n sur un alphabet à quatre lettres, évitant le motif $\alpha\hat{\alpha}$.

Déterminer les plus longs mots sur un alphabet à trois lettres, évitant le motif $\alpha\hat{\alpha}$.

Algorithmique du génome

Les biologistes s'intéressent aux répétitions dans le génome, qu'ils appelent des *satellites*. Il s'agit de séquences de bases répétées (deux fois ou plus).

Le nombre de répétitions peut varier d'un individu à l'autre (vestiges d'un processus évolutif, ou réserve permettant une évolution ultérieure).

Applications : empreintes génétiques ; diagnostic de certaines maladies.

Autres régularités intéressantes : les *palindromes complémentés*, par exemple ACGGCCGT (sites de restriction).

Autres ressources

On peut réaliser une partition de $\mathbb{Q}$ ou de $\mathbb{R}$ en trois classes, de telle façon que les intervalles $[x, x+a[$ et $[x+a, x+2a[$ ne soient pas identiques (à la translation près) et ce quels que soient x et a (cf. [1]).

Le chapitre 8 du (très onéreux) livre *Text algorithms*, de Crochemore et Rytter, s'intéresse aux facteurs carrés et aux palindromes.

Kolpakov et Kucherov ont donné récemment un algorithme de coût $\mathcal{O}(n+q)$ pour trouver les q facteurs carrés d'un mot de longueur n.

À signaler : parution prochaine chez Vuibert d'un livre consacré à l'algorithmique du texte, cosigné par Crochemore et Hancart.

Références

[1] D. R. Bean, A. Ehrenfeucht, and G. F. McNulty. Avoidable patterns in strings of symbols. *Pacific J. of Math.*, 85(2) : 261–294, 1979.

[2] J. Berstel. Some recent results on squarefree words. In *LNCS*, volume 166, pages 17–25, 1984.

[3] J. Berstel. Axel thue's work on repetitions in words. In *Proc. 4th FPSAC, Montreal*, 1992.

[4] F.-J. Brandenburg. Uniformly growing k-th power-free homomorphisms. *TCS*, 23 : 69–82, 1983.

[5] J. Cassaigne. *Motifs évitables et régularités dans les mots*. PhD thesis, Université Paris VI, 1994.

[6] M. Crochemore. Recherche linéaire d'un carré dans un mot. *C.R. Acad. Sci. Paris*, 196(1) : 781–784, 1983.

[7] M. Crochemore. Transducers and repetitions. *TCS*, 45 : 63–86, 1986.

[8] M. Crochemore and W. Rytter. Squares, cubes, and time-space efficient string searching. *Algorithmica*, 13 : 405–425, 1995.

[9] A. Ehrenfeucht and G. Rozenberg. On the separating power of e0l systems. *RAIRO Informatique Théorique*, 17 : 13–22, 1983.

[10] A. S. Fraenkel and R. J. Simpson. How many squares must a binary sequence contain ? *EJC*, 2, 1995.

[11] A. S. Fraenkel and R. J. Simpson. How many squares can a string contain ? *J. Combinatorial Theory (A)*, 82 : 112–120, 1998.

[12] V. Keränen. Abelian squares are avoidable on 4 letters. In *LNCS*, volume 623, pages 41–52, 1992.

[13] D.E. Knuth, J. H. Morris, and V. R. Pratt. Fast pattern-matching in strings. *SIAM J. Computing*, 6 : 323–350, 1977.

[14] M. G. Mainz and R. J. Lorentz. Linear time recognition of squarefree strings. In *Proc. NATO Advanced Workshop on Combinatorial Algorithms on Words*, pages 271–278, 1984.

[15] M. G. Mainz and R. J. Lorentz. An $O(n \lg n)$ algorithm for finding all repetitions in a string. *J. of Algorithms*, 5 : 422–432, 1984.

[16] P.A.B. Pleasants. Non-repetitive sequences. *Proc. Cambridge Phil. Soc.*, 68 : 267–274, 1970.

[17] R. Ross and K. Winklmann. Repetitive strings are not context-free. *RAIRO Informatique Théorique*, 16 : 191–199, 1982.