Vers l'algorithme de Makanine

Sujet proposé par Jean-Jacques Lévy

jean-jacques.levy@inria.fr

1 Préambule

Un des algorithmes les plus ardus de l'informatique a été proposé en 1983 par Gennadi Makanine pour résoudre le problème du mot. Une excellente présentation de cet algorithme peut être trouvée dans le livre collectif *Algebraic Combinatorics on Words* [1] (chapitre 12 écrit par V. Diekert, accessible sur la page web de J. Berstel `http://www-igm.univ-mlv.fr/~berstel/Lothaire/ChapitresACW/C12.ps`). Ce projet informatique vise à traiter deux cas particuliers de ce problème ; les élèves super-motivés peuvent également comprendre le cas général sans essayer de le programmer, ce qui déborderait des limites raisonnables en temps pour un projet informatique.

Le problème du mot consiste à résoudre des équations sur les semi-groupes libres (autrement dit sur les chaînes de caractères). Par exemple, l'équation

$$abxcy = ycxba \tag{1}$$

admet la solution $x = a$ et $y = aba$ pour x et y. De même, l'équation

$$xbax^3abx^4abx = abx^5bax^4ba \tag{2}$$

admet $x = aba$ comme solution. Enfin l'équation

$$abxaxc = yxc \tag{3}$$

admet toutes les solutions de la forme $y = abxa$ où x est un mot quelconque.

Nous considérerons les deux cas particuliers expliqués dans les toutes premières pages du chapitre 12 de [1] : le cas quadratique quand il n'y a pas plus que deux occurrences de chaque variable (comme dans la première équation), et le cas où il n'y a qu'une seule variable (comme dans la deuxième équation).

Enfin, on peut également montrer que si on sait résoudre une seule équation, on sait résoudre tout système d'équations sur les mots.

2 Détail du sujet

On dispose d'un alphabet $A = \{a, b, c, \ldots\}$ et d'un ensemble de variables $\Omega = \{x, y, z, \ldots\}$ ($A \cap \Omega = \emptyset$). On se donne une équation $L = R$, avec $L, R \in (\Omega \cup A)^*$. Une substitution σ est une fonction de Ω dans A^* étendue de manière triviale (par homomorphisme) en fonction de $(\Omega \cup A)^*$ dans A^*.

Ainsi pour l'équation

$$xauzau = yzbxaaby \tag{4}$$

on a $L = xauzau$ et $R = yzbxaaby$. Si on considère la substitution σ

$$\sigma(x) = abb, \quad \sigma(y) = ab, \quad \sigma(z) = ba, \quad \sigma(u) = bab$$

on vérifie que σ est une solution de (4), puisque

$$\sigma(xauzau) = \sigma(yzbxaaby) = abbababbaabab$$

Une *solution singulière* d'une équation $L = R$ est une solution telle que $\sigma(x) = \epsilon$ pour une variable x (c'est donc une solution où une des variables vaut le mot vide).

Enfin, un *facteur* du mot $w \in (\Omega \cup A)^*$, est un mot v tel que $w = w_1 v w_2$.

2.1 Le cas quadratique

Dans ce cas, on suppose qu'il y a au plus 2 occurrences de chaque variable dans le mot LR comme dans les équations (1) et (4). Alors, il existe un simple algorithme non-déterministe (donc déterministe avec *backtracking*).

On commence par essayer si une solution singulière existe en remplaçant tous les sous-ensembles de variables par le mot vide. Sinon, une solution non singulière doit exister. On raisonne alors par cas sur les facteurs gauches de L et de R :

- Soit $L = aL'$ et $R = bR'$. Dans ce cas, on doit avoir $a = b$ et on recommence avec l'équation $L' = R'$. Si $a \neq b$, il n'existe pas de solution.
- Soit $L = xL'$ et $R = aR'$. Dans ce cas, on sait que la solution σ est de la forme $\sigma(x) = aw$ où $w \in A^*$. On peut donc appliquer la substitution $\sigma(x) = ax$ à L et R, ce qui donne une nouvelle équation $ax\sigma(L') = a\sigma(R')$, c'est-à-dire $x\sigma(L') = \sigma(R')$. Comme $L = R$ est une équation quadratique, la taille de la nouvelle équation n'est pas plus grande. On peut donc recommencer avec elle, si on ne l'a pas déjà rencontrée.
- Soit $L = aL'$ et $R = xR'$. Cas symétrique du précédent.
- Soit $L = xL'$ et $R = yR'$. Alors on a deux sous-cas symétriques. Soit x est un préfixe de y, soit y est un préfixe de x (lemme de Levy sans accent). Dans le premier sous-cas, on peut considérer la substitution σ telle que $\sigma(y) = xy$ et considérer la nouvelle équation $\sigma(L') = y\sigma(R')$, dont la taille n'est encore pas plus grande que celle de $L = R$ (à cause de la condition quadratique).

Pour implémenter cet algorithme, on peut considérer le graphe dont les sommets sont les équations de taille inférieure ou égale à l'équation de départ sur un même ensemble de variables Ω. Un arc existe d'une solution a une autre, quand on passe par une des substitutions expliquées dans les 4 cas précédents, ou par une solution singulière. On parcourt ce graphe jusqu'à tomber sur une équation triviale de la forme $L = L$ avec $L \in A^*$. Pour obtenir la solution, il ne reste plus qu'à noter les substitutions élémentaires sur le chemin entre l'équation de départ et une équation triviale.

2.2 Le cas d'une seule variable

On pose donc $\Omega = \{x\}$. Il faut faire quelques définitions avant de décrire l'algorithme.

Un mot *primitif* p est un mot qui ne peut se mettre sous la forme $p = r^\alpha$ ($\alpha > 1$, $r \in A^+$, $p \neq \epsilon$). Un mot primitif ne peut donc être la répétition d'un même mot.

Si p est un mot primitif, on peut montrer qu'il existe une décomposition unique de tout mot de $w \in A^*$ par rapport à p. C'est sa *p-forme normale* définie comme la plus courte suite (k minimum)

$$(u_0, \alpha_1, u_1, \cdots, \alpha_k, u_k)$$

telle que :

$$\begin{array}{rl}
(i) & w = u_0 p^{\alpha_1} u_1 \cdots p^{\alpha_k} u_k \quad (0 \le \alpha_i,\ 1 \le i \le k)\\
(ii) & k = 0 \text{ si } p^2 \text{ n'est pas un facteur de } w\\
(iii) & \text{si } k > 1, \text{ alors}\\
& \qquad u_0 \in A^*p - A^*p^2A^*\\
& \qquad u_i \in (A^*p \cap pA^*) - A^*p^2 \quad \text{pour } 0 < i < k\\
& \qquad u_k \in pA^* - A^*p^2A^*
\end{array}$$

La décomposition s'effectue autour de tous les facteurs p^2 dans w. Ainsi pour $p = aba$ et $w = ab(aba)^5ba(aba)^4ba$, la décomposition de w est

$$(ababa, 3, ababa, 3, ababa)$$

et sa p-forme normale est $w = ababa(aba)^3ababa(aba)^3ababa$.

Pour résoudre une équation à une seule inconnue, on regarde d'abord si une solution singulière est possible. On tente donc $\sigma(x) = \epsilon$. S'il n'y a pas de solution singulière, on cherche un ensemble de solutions de la forme $\sigma(x) = p^\alpha r$ ($\alpha \ge 0$) donnée par un nombre fini de paires (p, r) où p est primitif, et r est un préfixe strict de p ($p = rp'$, $p' \ne \epsilon$).

Pour rechercher de telles solutions, on teste d'abord $\sigma(x) = r$, et $\sigma(x) = pr$ comme solutions. Si cela ne marche pas, il ne reste que les solutions de la forme $\sigma(x) = p^\alpha r$ ($\alpha \ge 2$). On remplace toutes les occurrences de x par $p^\alpha r$ dans l'équation $L = R$, et on décompose L et R en p-forme normale. Ce qui donne

$$\begin{array}{l}
L = u_0 p^{m_1\alpha+n_1} u_1 \cdots p^{m_k\alpha+n_k} u_k\\
R = v_0 p^{m'_1\alpha+n_1} v_1 \cdots p^{m'_k\alpha+n_k} v_\ell
\end{array}$$

Il ne reste plus qu'à vérifier $k = \ell$, $u_i = v_i$ ($0 \le i \le k$) et résoudre les équations en nombres entiers

$$(m_i - m'_i)\alpha = n'_i - n_i$$

Il ne reste plus qu'à trouver la base de paires (p, r) sur laquelle s'appuie le paragraphe précédent. Alors, on raisonne par cas sur la forme de l'équation $L = R$, comme pour les équations quadratiques. Le seul cas intéressant est quand $L = uxL'$ et $R = xvR'$ ($u \in A^+$) (ou son cas symétrique). La solution σ doit donc vérifier $ux = xw$ pour un certain $w \in A^*$. Soit p tel que $u = p^e$ pour p primitif et $e \ge 0$. Alors on montre facilement qu'on doit avoir $\sigma(x) = p^\alpha r$ pour des α et r tels que $\alpha \ge 0$ et r préfixe strict de p. Comme base de paires (p, r), on peut donc prendre le p primitif unique défini par $u = p^e$ et r préfixe quelconque strict de p.

La programmation du cas où il n'y a qu'une seule variable se fait donc par simple itération, après avoir calculé p en fonction de u et les préfixes r de p. On a intérêt à choisir une bonne représentation des chaînes de caractères pour effectuer rapidement les différentes solutions.

3 Travail demandé

On pourra faire ce projet à plusieurs niveaux. D'abord, on ne peut faire qu'un des deux cas particuliers. On peut aussi faire les deux. Enfin, il est possible de continuer dans la compréhension de l'algorithme de Makanine.

Références

[1] M. Lothaire "Algebraic Combinatorics on Words", Cambridge University Press, 2002. Sur le web en `http://www-igm.univ-mlv.fr/~berstel/Lothaire`

[2] Yu. Matiyasevich "A connection between systems of word and length equations and Hilbert's Tenth Problem", Zap. Nauchn. Sem. Leningrad. Otdel. Mat. Inst. Steklov (LOMI), 8, 132-144.

[3] J. Jaffar. "Minimal and Complete Word Unification", Journal of the Association for Computing Machinery, Vol. 37, 1, January 1990, pp. 47-85.

[4] M. E. Stickel. "A Unification Algorithm for Associative-Commutative Functions", Journal of the Association for Computing Machinery, Vol. 28, 3, July 1981, pp. 423-434.

[5] F. Fages, G. Huet "Complete sets of unifiers and matchers in equational theories", Theoretical Computer Science, Vol 43, 1986, pp. 189–200.

[6] A. J. Robinson "On the mechanization of the theory of equations", Bull. Res. Council Israel, 9F, 1960, pp. 47-70.