26

Performances d'algorithmes de recherche de motifs sous divers modèles probabilistes

Mireille Régnier
INRIA, Rocquencourt

[résumé par Abalo Baya]

Le problème de recherche de motifs consiste à trouver toutes les occurrences d'un motif p dans un texte t, ou la première occurrence, p et t étant des mots sur un alphabet $\mathcal{A}$ à q éléments. La complexité des algorithmes de résolution d'un tel problème est habituellement, et ce sera le cas ici, le nombre de comparaisons de caractères entre le texte et le motif. Dans cet exposé l'auteur présente les résultats qu'il a établis sur la complexité en moyenne des algorithmes de Morris-Pratt [3] et de Boyer-Moore [1]. On considère un modèle de probabilité stationnaire, ce qui généralise les résultats déjà connus sous le modèle uniforme.

1 Algorithme de Morris-Pratt

Cette section décrit l'algorithme de Morris-Pratt (MP) et la plus utilisée de ses variantes, i.e. l'algorithme de Knuth-Morris-Pratt (KMP). Ces deux algorithmes lisent le texte t de gauche à droite (la lecture de droite à gauche étant interdite) dans le but de trouver le motif p recherché. La comparaison à réaliser à un instant donné est déterminée par un pointeur de texte PT et un pointeur de motif PM : Si $PT = i$ et $PM = j$, alors le $i^{\text{ème}}$ caractère du texte est comparé au $j^{\text{ème}}$ caractère du motif. Après une comparaison, ces pointeurs sont mis à jour en fonction du résultat de celle-ci. Si les deux caractères sont identiques (on dit qu'il y a *match*), alors on incrémente chaque pointeur d'une unité (sauf dans le cas où le motif est trouvé). En revanche, si ces deux caractères sont distincts (il y a *mismatch*) ou si p est trouvé, la nouvelle comparaison est déterminée par le plus grand bord $p"$ du préfixe p' de p déjà trouvé, i.e. le plus grand sous-mot $p"$ tel que $p"$ soit à la fois un préfixe et un suffixe stricts de p'. On définit la fonction SUIVANT pour tout j par : p' étant le préfixe de longueur j de p, $p"$ son plus grand bord, de longueur $k-1$, SUIVANT(j)=k. Cela veut dire qu'après un *mismatch* ou une occurrence de p, $PM = k$. Prenons par exemple $p = 01201201345$ et $t = 030120120110120120345$. Après le premier *match*, PM et PT sont incrémentés d'une unité. Le *mismatch* entre $t[2] = 3$ et $p[2] = 1$ implique que $PM = 1$ et le *mismatch* suivant ($t[2] \neq p[1]$) implique que $PM = 1$ et $PT = 3$. On assiste alors à 8 *matches* suivis d'un *mismatch*, $t[11] \neq t[9]$. Le plus grand bord de $p' = 01201201$ est 01201 et PM prend la valeur 6. Comme $p[6] \neq t[11]$ et 01 est le plus grand bord de 01201, PM devient 3. Finalement $p[3] \neq t[11]$ implique $PM = 1$ et un *mismatch*. On incrémente PT d'une unité et on trouve ensuite le motif.

On remarquera que cet algorithme ne tient pas compte de l'information sur le caractère du texte où a lieu un *mismatch*. En fait, quand SUIVANT[j]=k, la comparaison qui suit, $t[i]?p[k]$, donne lieu à un *mismatch* lorsque $p[j] = p[k]$. Ainsi, SUIVANT[j] peut être redéfini comme étant le plus grand k tel que $p[1]...p[k-1] = p[j+1-k]...p[j-1]$ et $p[j] \neq p[k]$ et c'est précisément l'option choisie dans l'algorithme de Knuth-Morris-Pratt.

2 Algorithme de Boyer-Moore-Horspool

Les algorithmes du type Boyer-Moore lisent le texte de droite à gauche. Comme pour Morris-Pratt, le motif est décalé vers la droite lorsqu'il y a *mismatch* ou lorsque le motif a été trouvé. Ce décalage se fait suivant un *match* heuristique ou une occurrence heuristique. On s'intéresse ici à une simplification due à Horspool et basée sur l'occurrence heuristique. Si m désigne la longueur du motif p, alors la table de décalage est formellement définie par

$$d(x) = \min\{s : s = m \text{ ou } (1 \leq s < m \text{ et } p[m-s] = x)\}.$$

Intuitivement, si x, en position i sur le texte, est un caractère sur lequel il y a un *mismatch*, on fait glisser le motif p vers la droite jusqu'à ce que le caractère x de p (s'il y en a) le plus à droite vienne coïncider sur la position i du texte. Les positions du texte alignées avec le dernier caractère de p après un décalage sont appelées *têtes*.

3 Modèle probabiliste

On donne ici les notations, définitions et hypothèses nécessaires à la recherche d'un motif donné dans un texte aléatoire.

Définition 1 *Soit p un motif de longueur $|p| = m$. On note $C_n(p)$, le nombre moyen de comparaisons faites par un algorithme de recherche de motifs sur un texte aléatoire de longueur n et dont les caractères ont une distribution donnée. $C_n(m)$ désigne la valeur moyenne de $C_n(p)$ pour une distribution des caractères du motif donnée et lorsque p parcourt l'ensemble des motifs de longueur m.*

Définition 2 *On désigne par $\{p_a\}_{a\in A}$ (resp. $\{q_a\}_{a\in A}$), la distribution des caractères dans un motif (resp. dans un texte), c'est-à-dire la probabilité pour que toute position dans le motif ou dans le texte soit occupée par un caractère a donné est p_a (resp. q_a). $(\{p_a\},\{q_a\})_{a\in A}$ est dite distribution motif-texte.*

Notation. Pour $i \geq 1$, $\sigma_i = \sum_{a\in A}(p_a q_a)^i$, et pour $1 \leq i < j$, $s_{j,i} = \sum_{a\in A} p_a^j q_a^i$.

4 Performances des algorithmes KMP et MP

Avant d'énoncer les résultats obtenus, on pose la définition suivante :

Définition 3 *Soit F un polynôme ou une série entière sur l'ensemble $\{\sigma_i\}$. La troncature F_m est le polynôme déduit de F en ne considérant que les monômes $\prod \sigma_{\alpha_i}^{\beta_i}$ tels que $\sum_i \beta_i\alpha_i \leq m$.*

Théorème 1 *On se donne une distribution motif-texte $(\{p_a\},\{q_a\})_{a\in A}$ et un motif de longueur m. Alors le coût amorti, i.e. le nombre moyen de comparaisons effectuées par caractère de texte, $\frac{C_n^{MP}(m)}{n}$, tend vers une constante L_m^{MP} quand $n \to \infty$. Et on a:*

$$\begin{aligned} L_m^{MP} &\sim 1 + (\sigma_1 - \sigma_1^m) - F_m(\{\sigma_i\}), \\ \text{où} \qquad F(\{\sigma_i\}) &= \sum_{l\geq 1}\left(\frac{(\sigma_1 - 1)\sigma_{l+1}}{(1-\sigma_{l+1})(1-\sigma_l)} - \frac{\sigma_{l+2}}{1-\sigma_{l+1}}\right. \end{aligned}$$

Le terme d'erreur est $o(\sigma_1^5\sigma_3)$ et on a même l'égalité pour $m < 5$.

Corollaire 1 *Pour m assez grand, on a $L_m^{MP} \sim 1 + \sigma_1 - \sigma_1\sigma_2$. De plus,*

$$L_4^{MP} = 1 + \sigma_1 - \sigma_1\sigma_2 + \sigma_3 - \sigma_1^2\sigma_2 + \sigma_2^2 - \sigma_1^4 - \sigma_1\sigma_3 + \sigma_4$$

$$L_3^{MP} = 1 + \sigma_1 - \sigma_1^3 - \sigma_1\sigma_2 + \sigma_3 \qquad L_2^{MP} = 1 + \sigma_1 - \sigma_1^2.$$

Théorème 2 *On fait les mêmes hypothèses que dans le théorème précédent. Alors, le nombre moyen de comparaisons L_m^{KMP} faites par l'algorithme KMP satisfait $C_n^{KMP} \leq L_m^{MP}$ et*

$$|L_m^{KMP} - L_m^{MP}| \leq [F(\{p_iq_i\})[(s_{2,1} - \sigma_2) + F(\{p_i^2q_i^2\})\sigma_2(s_{2,1} - \sigma_2)]]_m$$

où F est la série définie dans le théorème précédent.

Théorème 3 *Pour le modèle uniforme, L_m^{MP} vérifie*

$$L_m^{MP} \sim 1 + \frac{1}{q} - \frac{1}{q^m} + \frac{q-1}{q^2} G_{m-1}(\frac{1}{q^2})$$

où

$$G(x) = \frac{x}{1-x} \sum_l P(x^l).$$

et où P est la série génératrice (connue) des mots primitifs (i.e. les mots qui ne sont puissance d'aucun autre mot). Pour tout $m > 6$, l'approximation est d'ordre $1/q^{11}$ et c'est un coût exact pour les petites valeurs de m.

5 Performances de l'algorithme de Boyer-Moore

L'analyse des performances pour l'algorithme de Boyer-Moore se fait en deux étapes : évaluation du nombre de "têtes", puis calcul du nombre de comparaisons.

Théorème 4 *Pour un motif p de longueur m, H_p^l désigne la probabilité pour que le $l^{\text{ème}}$ caractère soit une tête. Alors H_p^l converge vers une probabilité H_p^∞, avec*

$$H_p^\infty = \frac{1}{E_p[\text{Shift}]} \equiv \frac{1}{\sum_{a\in A} q_a s_p(a)}$$

où $s_p(a)$ est le décalage calculé lorsque a est une tête dans le texte et où $E_p[\text{Shift}]$ est le décalage moyen sur l'ensemble des q valeurs de l'alphabet.

Notation. Soit $H(q,m) = E(H_p^\infty)$ la probabilité de trouver une tête en une position donnée du texte lorsque p parcourt tous les motifs de longueur m sur un alphabet de cardinal q. On note $Ph(q)$, la limite de $H(q,m), m \longmapsto \infty$.

Remarque. M. Régnier et R. Baeza-Yates ont établi la valeur exacte de $Ph(q)$ pour les petites valeurs de q et une estimation pour les grandes valeurs. On donne ici les coûts moyens de l'algorithme de Boyer-Moore à l'aide de $H(q,m)$ et de $Ph(q)$.

Théorème 5 *Soit $C_n(m)$ le nombre moyen de comparaisons texte-motif pour les textes aléatoires de longueur n et les motifs de longueur m. Pour une distribution de texte pas trop irrégulière, on a :*

$$L_m^{BM} = \frac{C_n(m)}{n} = H(q,m)(1+\sigma_1+2\sigma_1^2),$$

ou pour de grands motifs, $L_m^{BM} \sim Ph(q)(1+\sigma_1+2\sigma_1^2)$ et l'approximation est majorée par $\sigma_1^2(1/q(q+1)min\{q_a : a \in A\})^2$.

Remarque. Tous les résultats ci-dessus ont été finalement étendus pour des dépendances markoviennes entre les données dans [6].

Références

[1] R. Boyer and J. S. Moore. A fast string searching algorithm. *Communications of the ACM*, 20:762–772, 1977.

[2] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: in Pascal and C.* Addison–Wesley, second edition, 1991.

[3] D. E. Knuth, J. Morris, and V. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6:323–350, 1977.

[4] M. Régnier. Knuth-Morris-Pratt algorithm: an analysis. In *MFCS'89*, volume 379 of *Lecture Notes in Computer Science*, pages 431–444. Springer-Verlag, 1989. Proc. Mathematical Foundations of Computer Science 89, Porubka, Poland.

[5] M. Régnier. Performance of String Searching Algorithms under Various Probabilistic Models. Rapport de recherche 1565, Institut National de Recherche en Informatique et en Automatique, 1991.

[6] M. Régnier. A language approach to string searching evaluation. In *Proceedings of Combinatorial Pattern Matching '92, Tucson, Arizona*, pages 15–26. Springer-Verlag, 1992.

[7] R. Sedgewick. *Algorithms.* Addison-Wesley, Reading, Mass., second edition, 1988.