TD8 : Recherche de motifs

1. Mise en jambes

Soient $x = x_1 \dots x_m$ et $t = t_1 \dots t_n$ deux séquences de caractères, avec $n > m$. Donner un algorithme naïf permettant de déterminer si le motif x est un facteur du texte t et donner sa complexité dans le pire des cas.

2. Algorithme de Morris et Pratt

L'algorithme naïf est lent car il va faire de nombreuses comparaisons inutiles ! Supposons que l'échec se soit produit à la $i-$ème lettre du motif x, on a pour un entier k :

$$x_1 \dots x_{i-1} = t_{k+1} \dots t_{k+i-1} \text{ et } x_i \neq t_{k+i}$$

Si l'on reprend la recherche ℓ pas plus loin, avec $\ell < i$, on compare donc $t_{k+\ell+1} \dots$ avec $x_1 \dots$. En d'autres termes, on compare un suffixe de $x_1 \dots x_{i-1}$ avec x lui même. De là naît l'idée d'un prétraitement sur le motif à rechercher.

Définition 1 Pour un mot u, un *bord* de u est un mot v distinct de u tel que v soit préfixe et suffixe de u. Le *bord maximal* de u est le bord le plus long de u et est noté $\text{Bord}(x)$.

2.1. Quels sont les bords de *abacaba* ? Et son bord maximal ?

Le prétraitement consiste à associer à chaque préfixe v d'un motif x son bord maximal et la valeur $\beta_x(|v|)$ définie par :

- $\beta_x(0) = -1$
- $\beta_x(i) = |\text{Bord}(x_1 \dots x_i)|$

2.2. Effectuer ce prétraitement sur le motif $x = abacabac$.

2.3. Pour un mot x et une lettre a, expliciter l'ensemble des bords de x en fonction de $Bord(x)$ puis $Bord(xa)$ en fonction de $Bord(x)$.

2.4. On suppose qu'on a effectué le prétraitement sur le motif x. Utiliser ce prétraitement pour obtenir un algorithme de recherche du motif x, de complexité linéaire. Évaluer précisément le nombre de comparaisons de caractères dans le pire des cas.

3. Algorithme de Knuth, Morris et Pratt

Le décalage proposé dans l'algorithme de Morris et Pratt est utile si la nouvelle lettre t_{k+i} coïncide avec la lettre de x à laquelle elle va être comparée, c'est-à-dire si $x_{\beta_x(i-1)+1} = t_{k+i}$.

3.1. Pourquoi n'est-il pas possible de transcrire cette condition dans un prétraitement ?

On va remplacer cette condition par une condition plus faible en exigeant de ne pas se retrouver dans la même situation que précédemment. On exige donc : $x_i \neq x_{\beta_x(i-1)+1}$.

3.2. Expliquer pourquoi ceci est une condition nécessaire.

Définition 2 Soit $x = x_1 \dots x_m$. Deux préfixes u et v de x sont *disjoints* si $x_{1+|u|} \neq x_{1+|v|}$ ou si l'un des deux mots est x tout entier. Le préfixe u est un *bord disjoint* de v dans x si u est un bord de v et u et v sont des préfixes disjoints de x. Le plus long bord disjoint de v dans x est noté $\mathrm{DBord}_x(v)$.

3.3. Soit $x = abcababcac$. Donner deux préfixes u et v de x tel que u soit un bord disjoint de v dans x. Donner deux préfixes y et z de x non disjoints.

On définit la fonction $\gamma_x : \{0, \dots m\} \to \{-1, \dots, m-1\}$ telle que :

- $\gamma_x(0) = -1$
- $\gamma_x(i) = \begin{cases} |\mathrm{DBord}_x(x_1 \dots x_i)| & \text{si } x_1 \dots x_i \text{ admet un bord disjoint dans } x \\ -1 & \text{sinon.} \end{cases}$

3.4. Que vaut γ_x pour $x = abcababcac$?

3.5. Soit $x = x_1 \dots x_m$ un mot non vide.
Montrer que γ_x vérifie : $\gamma_x(j) = \begin{cases} \beta_x(j) & \text{si } j = m \text{ ou } x_{1+j} \neq x_{1+\beta_x(j)} \\ \gamma_x(\beta_x(j)) & \text{sinon.} \end{cases}$

3.6. Soit $x = x_1 \dots x_m$. Montrer que pour tout $0 \leq j < m$, on a $\beta_x(1+j) = 1 + \gamma_x^k(\beta_x(j))$, avec k le plus petit entier vérifiant l'une des deux affirmations suivantes :

- $1 + \gamma_x^k(\beta_x(j)) = 0$
- $1 + \gamma_x^k(\beta_x(j)) \neq 0$ et $x_{1+\gamma_x^k(\beta_x(j))} = x_{1+j}$

3.7. Écrire un squelette d'algorithme utilisant les résultats des questions précédentes, et l'appliquer sur $x = abacabac$ et $t = babacacabacaab$.