TD3 : Palindromes et fonction témoin

1 Recherche de palindromes pairs

Si σ est un mot, $\widetilde{\sigma}$ désigne le *miroir* de σ, défini inductivement par $\widetilde{\varepsilon} = \varepsilon$ et $\widetilde{a\sigma} = \widetilde{\sigma}a$ avec a une lettre. Un *palindrome pair* est un mot de la forme $\sigma\widetilde{\sigma}$. Le but de cette partie est de rechercher dans un texte T de longueur n un préfixe non vide qui soit un palindrome pair. Dans la suite, un *prefpal* désignera un tel préfixe (non vide et palindrome pair).

Question 1 : Proposer un algorithme naïf de recherche d'un prefpal de T et donner sa complexité en fonction de n.

Question 2 : Proposer un algorithme de recherche du plus grand prefpal de T, puis du plus petit prefpal de T, qui ait une complexité linéaire. *Indication :* on vous conseille, à partir de T et de $\widetilde{T}$, de former un mot de longueur $2n+1$ et d'utiliser la fonction π du cours.

Soit $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ une position du texte. On note $pal(i)$ le plus grand palindrome pair *centré* sur i et contenu dans T, c'est-à-dire $pal(i) = T[i-r+1, i+r]$ avec :

$$r = \max(j \mid T[i+1, i+j] = \widetilde{T[i-j+1, i]})$$

On désigne cet entier r (qui peut être nul) par $ray(i)$.

Question 3 : Caractériser le plus petit prefpal de T à l'aide de la fonction ray.

Question 4 : Soit $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ et $1 \leq k \leq ray(i)$. Démontrer que si $ray(i-k) \neq ray(i) - k$ alors $ray(i+k) = \min(ray(i-k), ray(i) - k)$.

Question 5 : Soit $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ et $1 \leq k \leq ray(i)$. Démontrer que si $ray(i-k) = ray(i) - k$ alors on peut calculer $ray(i+k)$ en effectuant des comparaisons de caractères de texte avec successivement $T[i + ray(i) + 1]$, $T[i + ray(i) + 2]$, etc.

Question 6 : Déduire des deux questions précédentes un algorithme de recherche du plus petit prefpal de T de complexité linéaire et tel que, si ce préfixe existe et qu'il est de taille s, alors la recherche s'effectue en $O(s)$.

Question 7 : On mot σ est un palindrome si $\tilde{\sigma} = \sigma$ (ainsi, la notion de palindrome pair correspond aux palindromes de longueur paire). Expliquer comment ré-utiliser l'algorithme précédent pour rechercher le plus petit préfixe qui est un palindrome (au lieu de seulement le plus petit palindrome pair).

2 Calcul d'une fonction témoin

Soit P un motif de longueur m. Une fonction $tem : \{1, \ldots, m-1\} \mapsto \{0, \ldots, m-1\}$ est dite fonction *témoin* si elle vérifie la spécification suivante :

- $tem(i) \neq 0$ ssi il existe (au moins) un $1 \leq k \leq m-i$ tel que $P[i+k] \neq P[k]$
- si $tem(i) \neq 0$ alors $P[i + tem(i)] \neq P[tem(i)]$

Généralement il y a plusieurs fonctions témoins pour un motif (pour tout i, $tem(i)$ est égal à l'un des k vérifiant la première propriété).

Question 1 : Proposer un algorithme en $O(m)$ de calcul d'une fonction témoin. *Indication :* on vous conseille d'utiliser la fonction *Pref* du cours.

On se donne maintenant un texte T de longueur n. Deux positions du texte $1 \leq i < j \leq n-m+1$ sont dites *incohérentes* si $j - i < m$ et $tem(j-i) \neq 0$.

Question 2 : Soient i et j deux positions incohérentes. Montrer que, **quel que soit** T, on a soit $T[i, i+m-1] \neq P$, soit $T[j, j+m-1] \neq P$ (les deux inégalités peuvent être vérifiées).

Question 3 : Démontrer que si $i < j$ sont deux positions cohérentes et $j < k$ sont deux positions cohérentes, alors i et k sont deux positions cohérentes.

On se propose de calculer un ensemble de positions de T toutes cohérentes entre elles, qui soit un sur-ensemble des positions d'où démarre le motif. À cette fin, on gère une pile initialement vide et on parcourt les positions de 1 à $n-m+1$. À chaque itération, on effectue les opérations suivantes :

- On empile la position courante.
- Tant que la pile a au moins deux éléments et que les deux positions au sommet $i < j$ sont incohérentes, si $T[j-1+tem(j-i)] \neq P[tem(j-i)]$, alors on supprime j de la pile, sinon on supprime i.

L'ensemble recherché *Possible* est l'ensemble des positions dans la pile à l'issue de l'algorithme.

Question 4 : Montrer que cet algorithme vérifie sa spécification et qu'il opère en $O(n)$.

À partir de *Possible*, on construit un texte T' de longueur n sur l'alphabet $\{0, 1\}$ en parcourant le texte T à l'aide d'un indice i variant de n à 1. On maintient simultanément un entier j, égal à la plus grande position k de *Possible* telle que $k \leq i$ ($j = 0$ s'il n'existe pas une telle position).

- Si $j = 0$ ou $i - j \geq m$ ou $T[i] \neq P[i-j+1]$ alors $T'[i] = 0$.
- Sinon $T'[i] = 1$.

Question 5 : Montrer que cet algorithme opère en $O(n)$ et que $T'[i, i+m-1] = 1^m$ si et seulement si $T[i, i+m-1] = P$.

Question 6 : Proposer un algorithme qui opère en temps $O(n)$ en maintenant un unique compteur pour trouver les positions i de T' telles que $T'[i, i+m-1] = 1^m$.