ALGORITHMES DE RECHERCHE DE SOUS-CHAINES
ALGORITHME DE BOYER-MOORE

INTRODUCTION

En algorithmique du texte, un algorithme de recherche de sous-chaînes est un type d'algorithme de recherche qui a pour objectif de trouver une chaîne de caractères (le motif P) à l'intérieur d'une autre chaîne (le texte T).

Il existe de nombreux algorithmes de recherche de sous-chaînes. En pratique, pour une plus grande efficacité, les bibliothèques de programme combinent de multiples algorithmes. Par exemple stringlib/fastsearch en Python combine quatre algorithmes.

Algorithmes de recherche de sous-chaînes : Algorithme **naïf** - Algorithme de **Knuth-Morris-Pratt** - Algorithme de **Boyer-Moore** - Algorithme de **Boyer-Moore-Horspool** - Algorithme de **Raita** - Algorithme de **Baeza-Yates-Gonnet** - Algorithme **Z**

L'algorithme de Boyer-Moore est considéré comme l'algorithme de recherche de sous-chaînes le plus efficace dans les applications usuelles.

RECHERCHES MULTIPLES

Les algorithmes de **recherches multiples** permettent la **détection de plagiat** ou la **comparaison d'un fichier suspect à des fragments de virus**.

Algorithmes de recherches multiples : Algorithme de **Rabin-Karp** - Algorithme d'**Aho-Corasick**.

APPLICATIONS EN BIO-INFORMATIQUE

Les algorithmes de recherche de sous-chaînes peuvent être utilisés pour interpréter le texte des génomes. Un nombre croissant de séquences de génomes ou ADN complémentaire sont accessibles à tous les chercheurs. L'algorithmique du texte et ses applications est donc un passage obligé pour Les étudiants en biologie qui se destinent à la recherche.

ALGORITHME NAÏF / FORCE BRUTE

L'idée est de réaliser une comparaison caractère par caractère de la chaîne initiale et de la chaîne recherchée.

On parcourt les caractères de la chaîne initiale tant qu'ils sont différents du premier caractère de la chaîne à trouver. Dès qu'on trouve un caractère identique, on parcourt les caractères suivants tant qu'ils correspondent. Si un caractère diffère alors qu'on n'a pas atteint la fin de la chaîne recherchée, alors on reprend la recherche du premier caractère identique, à partir du caractère suivant dans la chaîne initiale.

Si tous les caractères correspondent, on retourne la position du premier caractère de la chaîne trouvée dans la chaîne initiale.

ALGORITHME DE BOYER-MOORE

L'algorithme de Boyer-Moore est un algorithme de recherche de sous-chaîne particulièrement efficace. Il a été développé en 1977 par Robert S. Boyer et J Strother Moore, à l'université du Texas.

L'algorithme de Boyer-Moore effectue la vérification à l'envers. Par exemple, s'il commence la recherche de la sous-chaîne OZANAM au début d'un texte, il vérifie d'abord la 6e position en regardant si elle contient un M. Ensuite, s'il a trouvé un M, il vérifie la huitième position pour regarder si elle contient le A, et ainsi de suite jusqu'à ce qu'il ait vérifié toutes les lettres.

Ainsi si le M n'est pas trouvé au départ, et que le caractère n'est pas contenu dans OZANAM, alors l'algorithme n'a pas besoin de tester les 5 premiers caractères.

Paradoxalement, plus la sous-chaîne est longue, et plus l'algorithme de Boyer-Moore est efficace pour la trouver.

Exemple : recherche du motif `OZANAM` dans la chaine `NOTRE DAME OZANAM A MACON`

```
OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

On compare le **6**e caractère. Le caractère E n'est pas dans le motif. On décale de **6** caractères.

```
      OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

Le caractère A est dans le motif. On décale le motif de **1** place vers la droite pour faire correspondre le A de la chaine et le A le plus à droite dans le motif.

```
       OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

M et I ne correspondent pas. Le caractère I n'est pas dans le motif. On décale de **6** caractères.

```
             OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

Le caractère T n'est pas dans le motif. On décale de **6** caractères.

```
                    OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

M et M correspondent, A et A correspondent, mais N et D ne correspondent pas. Le caractère D n'est pas dans le motif. On décale de **4** caractères.

```
                        OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

Le caractère Z est dans le motif. On décale de **4** caractères.

```
                            OZANAM
CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON
```

Tous les caractères correspondent. Le motif a été trouvé.

On remarque que pour pouvoir effectuer les décalages appropriés, l'algorithme doit au préalable déterminer les positions respectives des caractères dans le motif. Ces informations sont enregistrées dans une table dite « **table des sauts** ».

IMPLEMENTATION DE L'ALGORITHME DE BOYER-MOORE EN PYTHON

```
def creer_table_sauts(t):
    """Retourne un dictionnaire avec pour clé la lettre
    et pour valeur le   décalage
    t : chaine de caractères
    """
    n = len(t)
    dico = {}
    # La dernière lettre n'est pas nécessaire
    for i, lettre in enumerate(t[:-1]):
        dico[lettre] = n - i - 1
    return dico

def boyer_moore(txt, motif):
    """Recherche la présence d'un mot dans un texte avec l'algorithme
    de boyer-moore
    txt: le texte dans lequel on fait la recherche
    motif: le texte recherché
    la fonction retourne True si le motif est trouvé, False sinon.
    """
    N = len(txt)
    n = len(motif)

    # création d'un dictionnaire de décalages
    dico_sauts = creer_table_sauts(motif)

    # on commence à la fin du mot
    i = n - 1
    while i < N:
        lettre = txt[i]
        if lettre == motif[-1]:
            if txt[i-n+1:i+1] == motif:
                return True
        if lettre in dico_sauts.keys():
            i += dico_sauts[lettre]
        else:
            i += n
    return False


#----- TEST -----
texte= 'CENTRE SCOLAIRE NOTRE DAME OZANAM A MACON'
seq=' OZANAM '
print(boyer_moore(texte,seq))
```