Recherche textuelle

Approche naïve

Dringing

Implémentatio

Approche plus

Royer Moore

Recherche à l'envers

Décalages par sauts

Algorithme de Boyer-Moor (simplifié - version Horspool)

Complexité

Recherche textuelle

Christophe Viroulaud

Terminale - NSI

Problématique

Approche naïve

Princip

Implémentation

efficace :

Boyer-Moore

Recherche à l'envers

Décalages par sauts

Prétraitement du moti

Algorithme de Boyer-Moore (simplifié - version

Complexité

$1. \ \mathsf{Probl\'ematique}$

- 2. Approche naïve
- Approche plus efficace : Boyer-Moore

Problématique

Recherche textuelle

Problématique

Approche naïve

Implémentation

efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Complexité

Une fonctionnalité intégrée dans tous les logiciels de traitements de texte.

Des applications multiples

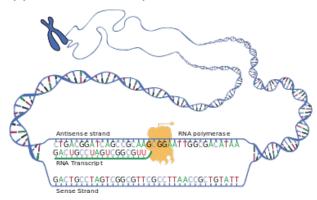


FIGURE 1 – Séquence ADN

- ► 4 bases nucléiques : Adénine, Cytosine, Guanine, Thymine,
- ADN humain : 3 milliards de bases répartis sur 23 paires de chromosomes.

Recherche textuelle

Problématique

Approche naïve

Approche plus efficace :
Boyer-Moore

Recherche à l'envers

Décalages par sauts Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Recherche textuelle

Problématique

Approche naïve

1 1/ . . .

Approche plus efficace :

Boyer-Moore

Décalages par caute

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Complexité

Comment effectuer une recherche textuelle efficace?

- 2. Approche naïve

- observer une fenêtre du texte.
- dans cette fenêtre, comparer chaque lettre du motif recherché au texte,
- décaler la fenêtre d'un cran dès qu'il n'y a pas de correspondance.

				3							
texte	а	С	g	а	t	С	С	а	t	g	а
motif	С	а	t								
	0	1	2								

Problematique

Approche naïve

Principe

mplémentation

efficace :

Oyer-IVIOOre

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Complexite

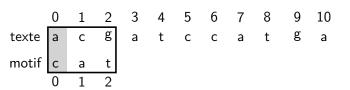


FIGURE 2 – Première comparaison : pas de correspondance

Décalage de la fenêtre

Recherche textuelle

roblématique

Approche naïve

Principe

Approche plus efficace :

Boyer-Ividore

Recherche à l'envers

Decarages par sauts

Algorithme de Boyer-Moor

(simplifié - version Horspool)

0 1 2 3 4 5 6 7 8 9 10 texte a c g a t c c a t g a motif 0 1 2

FIGURE 3 – Première comparaison : correspondance

Recherche textuelle

Problématique

Approche naïve

Principe

piementation

efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

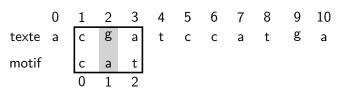


FIGURE 4 – Deuxième comparaison : pas de correspondance

Décalage de la fenêtre

Recherche textuelle

Problématique

Approche naïve

Principe

............

Approche plus

Boyer-Moore

Recherche à l'envers

Décalages par sauts

Algorithme de Boyer-Moore

(simplifié - version Horspool)

Implémentation

- 2. Approche naïve
- 2.2 Implémentation

Activité 1:

- Écrire la fonction recherche_naive(texte: str, motif: str) → int qui renvoie la position du motif dans le texte ou -1 s'il n'est pas présent.
- 2. Estimer la complexité temporelle de cet algorithme dans le pire des cas : le motif n'est pas présent dans le texte.

Problématiqu

Principe

Implémentation

efficace :
Boyer-Moore

Recherche à l'envers

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

```
def recherche_naive(texte: str, motif: str) -> int:
 1
 3
        renvoie la position du motif dans le texte
        −1 s'il n'est pas présent
 4
       # dernière position = taille(texte) - taille(motif)
 6
       for i in range(len(texte)—len(motif)+1):
          i = 0
           while (i < len(motif)) and (motif[i] == texte[i+i]):
 9
10
              i += 1
           if j == len(motif): # correspondance sur toute la fenê
11
        tre
12
              return i
       return -1
13
```

Code 1 – Approche naïve

Problématique

Approche naive

Implémentation

Approche plus efficace : Bover-Moore

Recherche à l'envers

Prétraitement du motif

Algorithme de Boyer-Moore simplifié - version Horspool)

Imaginons le cas :

- On vérifie toute la fenêtre à chaque fois.
- ▶ À chaque **non correspondance** la fenêtre avance de 1.
- La complexité dépend de la taille de la fenêtre et de celle du motif.

Problématique

Principe

Implémentation

efficace :
Bover-Moore

Décalages par caute

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Approche plus efficace :

Boyer-Moore

- 3. Approche plus efficace : Boyer-Moore

- 3.4 Algorithme de Boyer-Moore (simplifié version

Boyer-Moore

1970 : algorithme de Knuth-Morris-Pratt. O(T + M)

Recherche textuelle

Problématique

Approche naïve

Principe

Approche plus efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore simplifié - version Horspool)

Boyer-Moore

Recherche textuelle

Problématique

Approche naïve

l---li----

Approche plus efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version Horspool)

Complexité

▶ 1970 : algorithme de Knuth-Morris-Pratt. O(T + M)

▶ 1977 : algorithme de Boyer-Moore.

Problématiqu

Principe

Approche plus efficace : Boyer-Moore

Recherche à l'envers Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Complevité

- ▶ 1970 : algorithme de Knuth-Morris-Pratt. O(T + M)
- ▶ 1977 : algorithme de Boyer-Moore.
- ▶ 1980 : Horspool propose une version simplifiée de l'algorithme de Boyer-Moore. O(T)

La première idée de cet algorithme est de commencer la

recherche en partant de la fin du motif.

FIGURE 5 – Première comparaison : pas de correspondance

Problématiqu

pproche naïv

plémentation

efficace :

Recherche à l'envers

Décalages par sauts

Algorithme de Boyer-Moore (simplifié - version

`omnlovitá

Recherche textuelle

Problematique

Approche naïve

Implémentatio

Approche plus efficace :

Recherche à l'envers

Décalages par sauts

Algorithme de Boyer-Moor (simplifié - version Horspool)

Complexité

Pour l'instant cette approche ne semble par apporter d'amélioration par rapport à l'algorithme précédent.

Recherche textuelle

Problèmatique

Approche naïve

Princip

Implémentation

efficace :

Deskarsky à Passars

Décalages par sauts

Prétraitement du motif

(simplifié - version Horspool)

- 3. Approche plus efficace : Boyer-Moore
- 3.1 Recherche à l'envers
- 3.2 Décalages par sauts
- 3.3 Prétraitement du motif
- Algorithme de Boyer-Moore (simplifié version Horspool)
- 3.5 Complexité

Approche naïve

plémentation

efficace : Boyer-Moore

Décalages par sauts

Prétraitement du me

Algorithme de Boyer-Moor (simplifié - version

omplexité

Le motif ne contient pas la lettre ${\bf g}$ (la dernière lettre de la fenêtre).

										9	
texte	а	C	ℊg	а	t	С	С	а	t	g	а
texte motif	c [^]	a [*]	t								
	0	1	2								

FIGURE 6 - Comparaisons inutiles

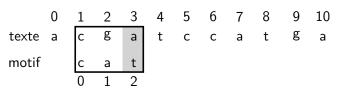


FIGURE 7 – Comparaison inutile

Recherche textuelle

Problématique

Approche naïve

Implémentation

Approche plue efficace :

-

Décalages par sauts

Algorithme de Boyer-Moore (simplifié - version

0 1 2 3 4 5 6 7 8 9 10 texte a c g a t c c a t g a motif 0 1 2

FIGURE 8 – Comparaison inutile

Recherche textuelle

Problématique

pproche naïv

Implémentation

Approche plu

Boyer-Moore

nccincione a remers

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

On peut donc directement décaler le motif à l'indice 3 du texte (figure 9).

	0	1	2	3	4	5	6	7	8	9	10
texte	a	С	g	а	t	С	С	a	t	g	а
motif					a						
				0	1	2					

FIGURE 9 – Décalage par saut

Problématique

rincipe

Approche pl

Boyer-Moore

Décalages par sauts

Algorithme de Boyer-Moore

On n'observe pas de correspondance par contre la lettre **c** existe dans le motif. On va donc le décaler pour les faire coïncider.

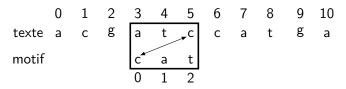


FIGURE 10 – Nouvelle situation

Problématique

Approche na

Frincipe

Approche efficace :

Boyer-Ividore

Décalages par sauts

Prétraitement du moti

Algorithme de Boyer-Moor (simplifié - version Horspool)

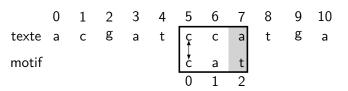


FIGURE 11 – Décalage par saut

Recherche textuelle

Problématique

pproche naiv

Implémentation

pproche plu

Boyer-Moore

- . .

Décalages par sauts

Algorithme de Boyer-Moore

Algorithme de Boyer-Moore (simplifié - version Horspool)

À retenir

On décale la position de recherche dans le texte en fonction de la dernière lettre de la fenêtre.

Recherche textuelle

Problématique

Approche naïve

Implémentation

Approche plus

Boyer-Moore

Décalages par sauts

-----g--- p--- ------

Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version Horspool)

Complevité

Recherche textuelle

Prétraitement du motif

- 3. Approche plus efficace : Boyer-Moore

- 3.3 Prétraitement du motif
- 3.4 Algorithme de Boyer-Moore (simplifié version

Implémentatio

efficace : Bover-Moo

Recherche à l'envers

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moo (simplifié - version Horspool)

Complexit

Pour pouvoir décaler par saut, il faut connaître la dernière position de chaque lettre dans le motif. Le prétraitement consiste à calculer le décalage à appliquer pour amener chaque caractère du motif à la place du dernier caractère.

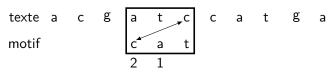


FIGURE 12 – Calculs des décalages

Remarque

On ne regarde pas la dernière position de la clé (la lettre t ici). Sinon la distance associée serait nulle et on resterait sur place après l'avoir lue dans le texte.

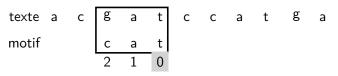


FIGURE 13 – Sauf la dernière lettre

Recherche textuelle

Problématique

opproche naive

plémentation

efficace :

Recherche à l'envers

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version Horspool)

Autre exemple

texte	а	С	g	а	t	С	С	a	t	g	а
motif	t	а	С	а	t						
	4		2	1							

FIGURE 14 - Répétition dans le motif

Recherche textuelle

Problematique

Approche na

Tillcipe

pproche plu

Bover-Moon

Recherche à l'envers

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Comployitá

Recherche textuelle

Approche Principe

Approche plu

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Complexité

Activité 2 : Écrire la fonction pretraitement_decalages(motif: str) \rightarrow dict qui associe chaque lettre du motif (sauf la dernière) à son décalage.

11

return decalages

```
def pretraitement decalages(motif: str) -> dict:
1
3
       renvoie le dictionnaire des décalages à appliquer
       pour chaque lettre du motif (sauf dernière)
4
5
       decalages = dict()
6
       # on s'arrête à l'avant dernière lettre du motif
       for i in range(len(motif)-1):
8
9
          \# len(motif)-1 est la position de la dernière
        lettre
          decalages[motif[i]] = len(motif)-1-i
10
```

Approche naïve

Principe

Approche plu

Boyer-Moore

Décalages par saute

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

- 3. Approche plus efficace : Boyer-Moore

- 3.4 Algorithme de Boyer-Moore (simplifié version Horspool)

Algorithme de Boyer-Moore (simplifié - version Horspool)

Algorithme de Boyer-Moore

L'algorithme de Boyer-Moore s'écrit alors :

```
Créer le tableau des décalages
Tant qu'on n'est pas à la fin du texte

Comparer le motif à la position du texte
Si le motif est présent
Renvoyer la position
Sinon
Décaler la fenêtre
Renvoyer —1 si le motif n'est pas présent
```

Code 2 – Algorithme de Boyer-Moore (version Horspool)

Recherche textuelle

Problématique

Approche naïve

Principe

Approche plu

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Complevité

Activité 3:

- Écrire la fonction compare(texte: str, position: int, motif: str) → bool qui renvoie *True* si le motif est présent à la position i du texte.
- Écrire la fonction decalage_fenetre(decalages: dict, taille: int, lettre: str) → int qui renvoie le décalage à appliquer pour faire coïncider le motif à la dernière lettre de la fenêtre. Si la lettre n'est pas présente, la taille du motif est renvoyée.
- 3. Écrire alors la fonction boyer_moore(texte: str, motif: str) → int qui renvoie la position du motif dans le texte et -1 sinon.

Problémation

Approche naï

plémentation

Approche plus efficace :
Boyer-Moore

Recherche à l'envers

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

2

4

5 6 7

q

10

11

12

13

14

15

16 17

18

```
def compare(texte: str, position: int, motif: str) -> bool:
   compare le morceau du texte
   (en partant de position + taille(motif))
   avec le motif
   Returns:
      bool: True si on a trouvé le motif
   # position de la dernière lettre de la fenêtre
   en\_cours = position + len(motif) - 1
   # parcours de la fenêtre à l'envers
   for i in range(len(motif)-1, -1, -1):
      if not(texte[en_cours] == motif[i]):
         return False
      else:
         en cours -=1
   return True
```

Problématique

Approche naïve

Implémentation

Approche plue

Recherche à l'envers

Prétraitement du motif

Algorithme de Boyer-Moore
(simplifié - version

Horspool)

```
def decalage_fenetre(decalages: dict, taille: int, lettre: str) ->
1
        int:
       77 77 77
       renvoie la valeur du décalage à appliquer.
       si la lettre n'est pas dans le tableau
       c'est la taille du motif qui est appliqué
6
       Args:
           decalages (dict): dico des décalages
           taille (int): taille du motif (= décalage max)
                                                                                      Algorithme de Boyer-Moore
                                                                                      (simplifié - version
           lettre (str): dernière lettre de la fenêtre
                                                                                      Horspool)
       Returns:
.2
           int: décalage à appliquer
       77 77 77
       for cle, val in decalages.items():
           if cle == lettre:
.6
               return val
       # si la lettre n'est pas dans le dico (= le motif)
       return taille
```

Recherche

textuelle

37 / 49

```
def decalage fenetre2(decalages: dict, taille: int, lettre: str) ->
       int:
       # la méthode get renvoie une valeur par défaut si elle ne
       trouve pas la clé
       return decalages.get(lettre, taille)
5
   def decalage_fenetre3(decalages: dict, taille: int, lettre: str) ->
6
       int:
      try:
          res = decalages[lettre]
      except KeyError:
          res = taille
       return res
```

oproche naive incipe

Implémentation

fficace :

oyer-Moore

echerche a l'envers Décalages par sauts

Algorithme de Boyer-Moore (simplifié - version Horspool)

```
def boyer_moore(texte: str, motif: str) -> int:
 1
 2
 3
        Returns:
           int: la position du motif dans le texte, -1 sinon.
 4
        77 77 77
 5
       decalages = pretraitement_decalages(motif)
 6
        i = 0
        while i \le len(texte) - len(motif):
 8
           # si on trouve le motif
 9
           if compare(texte, i, motif):
10
              return i
11
           else:
12
13
              # sinon on décale (en fonction de la dernière
        lettre de la fenêtre)
              decale = decalage fenetre(decalages,
14
                                     len(motif),
15
                                     texte[i+len(motif)-1])
16
              i += decale
17
        # si on sort de la boucle, on n'a rien trouvé
18
       return -1
19
```

Recherche

Problematiqu

Approche naïve

plémentation

Approche plu efficace : Bover-Moore

Recherche à l'envers

Algorithme de Boyer-Moore (simplifié - version Horspool)

°omnlevité

Recherche textuelle

1 Problématique

2. Approche naïve

- 3. Approche plus efficace: Boyer-Moore
- 3.1 Recherche à l'envers
- 3.2 Décalages par sauts
- 3.3 Prétraitement du motif
- Algorithme de Boyer-Moore (simplifié version Horspool)
- 3.5 Complexité

Problématique

Approche naïve

Principe

Implémentation

efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version

Intuitivement l'algorithme semble plus rapide que la version naïve car il ne teste pas toutes les lettres du texte.

aaabaaabaaab

FIGURE 15 – Un cas représentatif

Problématiqu

pproche nai rincipe

Approche plu

Boyer-Moore

Décalages par sauts

Algorithme de Boyer-Moore (simplifié - version

Recherche textuelle

Problématique

rincipe

nplémentation

efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version

Complexité

a a a b a a a a b a a a a b c c c c c

a a a a b a a a a b a a a a b c c c c c

FIGURE 16 – Algorithme naïf

aaabaabaaabaaab

FIGURE 17 – Algorithme de Boyer-Moore

Recherche textuelle

Problématique

pproche naiv

mplémentation

Approche plu efficace :

Boyer-Moore

Décalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Activité 4 : Compter le nombre d'itérations de la recherche avec l'algorithme naïf puis celui de Boyer-Moore.

Recherche textuelle

Problèmatiqu

Approche naïve

Implémentation

Approche plu

Boyer-Moore

Recherche à l'envers

Decarages par sauts

Algorithme de Boyer-Moore

Horspool)

Correction

Recherche textuelle

Problèmatique

Approche naïve

1 1/ 1/

Approche plu

etticace : Bover-Moore

Recherche à l'envers

Décalages par sauts Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version

Complexité

► Algorithme naïf : 10 décalages,

Algorithme de Boyer-Moore : 3 décalages.

Remarques

- ▶ Dans le meilleur des cas, la complexité temporelle de l'algorithme est O(T/M) où T est la taille du texte et M celle du motif.
- ▶ Plus le motif est long plus l'algorithme est rapide.

Recherche textuelle

Problèmatiqi

pprocne nar Principe

nplémentation

efficace : Boyer-Moore

Recherche à l'envers

Decalages par sauts

Prétraitement du motif

Algorithme de Boyer-Moore (simplifié - version Horspool)

Recherche textuelle

Problématique

Approche naïve

.

Approche plus

Boyer-Moore

Recherche à l'envers

Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version

Complexité

Le prétraitement a un coût (temporel et spatial) mais qui est grandement compensé.

Cas critique

texte t a a a a a a a a a a a motif a a a a a a a a a $\label{eq:figure 18-Cas} \text{Figure 18-Cas critique}$

Recherche textuelle

Problématique

Approche naïve

I---I/----

Approche p

etficace : Boyer-Moore

echerche à l'envers

Décalages par sauts Prétraitement du motif

Algorithme de Boyer-Moor (simplifié - version

Code complet

Recherche textuelle

Problématique

Approche naïve

I micipe

Approche pl

efficace :

Recherche à l'envers

Décalages par sauts

Algorithme de Boyer-Moore

Algorithme de Boyer-Moore simplifié - version Horspool)

Complexité

Les programmes du cours sont téléchargeables ici.