Mise en œuvre de la recherche de motif par l'algorithme naïf

Julien Reichert

Contexte

- Facteur d'une chaîne de caractères `s` : chaîne de caractères `m` dont tous les caractères apparaissent dans `s` de manière consécutive. Exemple : `"MOT"` dans `"ESCAMOTER"`.
- Différent de la notion de sous-mot où les caractères ne sont pas forcément consécutifs, comme `"MOT"` dans `"MONTER"`.
- En général, les chaînes sont alphabétiques avec espaces et ponctuation.
- Recherche de motif : problème ayant de nombreuses applications.
- Instance de ce problème : deux chaînes (a priori la première est très longue). Déterminer si la deuxième, **nommée « motif »**, est un facteur de la première, **nommée « texte »**.

Recherche de motif, variantes

En plus du problème standard (« déterminer si »), extensions possibles :

- Premier indice où le motif est trouvé.
- Nombre d'indices où le motif est trouvé.
- Liste des indices où le motif est trouvé.

Nous étudierons plutôt la première extension.

Recherche de motif

Trois algorithmes étudiés :

- Algorithme naïf : balayer le texte en testant toutes les possibilités de démarrage du motif en tant que facteur.
- Algorithme de Boyer-Moore : étudier le motif pour déterminer des indices où il ne peut pas démarrer dans le texte.
- Algorithme de Rabin-Karp : précéder la comparaison des caractères entre un facteur du texte démarrant à un certain indice et le motif par un rapide test d'égalité d'une empreinte ($\rightarrow$ hachage).

Pas au programme : Knuth-Morris-Pratt ($\rightarrow$ DM facultatif).

Algorithme naïf

On note `s` le texte, n sa taille, `m` le motif et k sa taille.

- Version peut-être plus intuitive mais utilisant un espace $\mathcal{O}(k)$: extraire à chaque fois un facteur de taille k de `s` en vue d'un test d'égalité.
- Version demandant de réfléchir aux indices : imbrication d'une boucle inconditionnelle sur les indices possibles (calculer le seuil !) et d'une boucle conditionnelle pour comparer les caractères (arrêter si c'est perdu !).
- Dans les deux cas la complexité en temps sera un $\mathcal{O}(nk)$.
- Pire des cas atteint par exemple en cherchant `"aa...ab"` dans `"aaa...aa"` (rappel : très long texte).

5 minutes de réflexion, puis passage au tableau

(Langage au choix !)

L'algorithme

Soient `s = "CHERCHEZ CHEZ CHER"` et `m = "CHEZ"`.

On propose le pseudo-code suivant pour la dernière variante :

```
fonction liste_motifs(s, m) {
  reponse <- []; n <- taille(s); k <- taille(m);
  pour i de 0 à n - k { // n - k inclus, vide si n < k
  j <- 0;
    tant que j < k et s[i+j] = m[j] {
      incrémenter j;
    }
    si j = k alors ajouter i dans reponse;
  }
  renvoyer reponse;
}
```

Mise en œuvre

```
s = "|C|HERCHEZ CHEZ CHER"

m = "|C|HEZ"

reponse = []

i = 0

j = 0
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "C|H|ERCHEZ CHEZ CHER"

m = "C|H|EZ"

reponse = []

i = 0

j = 1
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CH|E|RCHEZ CHEZ CHER"

m = "CH|E|Z"

reponse = []

i = 0

j = 2
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHE|R|CHEZ CHEZ CHER"

m = "CHE|Z|"

reponse = []

i = 0

j = 3
```

Échec de la correspondance, sortie de la boucle. 0 non incorporé.

Mise en œuvre

```
s = "C|H|ERCHEZ CHEZ CHER"

m = "|C|HEZ"

reponse = []

i = 1

j = 0
```

Échec de la correspondance, sortie de la boucle. 1 non incorporé.

Mise en œuvre

```
s = "CH|E|RCHEZ CHEZ CHER"

m = "|C|HEZ"

reponse = []

i = 2

j = 0
```

Échec de la correspondance, sortie de la boucle. 2 non incorporé.

Mise en œuvre

```
s = "CHE|R|CHEZ CHEZ CHER"

m = "|C|HEZ"

reponse = []

i = 3

j = 0
```

Échec de la correspondance, sortie de la boucle. 3 non incorporé.

Mise en œuvre

```
s = "CHER|C|HEZ CHEZ CHER"

m = "|C|HEZ"

reponse = []

i = 4

j = 0
```

Correspondance : on reste dans la boucle.

Mise en œuvre

`s = "CHERC|H|HEZ CHEZ CHER"`

`m = "C|H|EZ"`

`reponse = []`

`i = 4`

`j = 1`

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCH|E|Z CHEZ CHER"

m = "CH|E|Z"

reponse = []

i = 4

j = 2
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHE|Z| CHEZ CHER"

m = "CHE|Z|"

reponse = [4]

i = 4

j = 3
```

Correspondance puis sortie de la boucle. 4 incorporé.

Mise en œuvre

```
s = "CHERC|H|EZ CHEZ CHER"

m = "|C|HEZ"

reponse = [4]

i = 5

j = 0
```

Échec de la correspondance, sortie de la boucle. 5 non incorporé.

Mise en œuvre

```
s = "CHERCH|E|Z CHEZ CHER"

m = "|C|HEZ"

reponse = [4]

i = 6

j = 0
```

Échec de la correspondance, sortie de la boucle. 6 non incorporé.

Mise en œuvre

`s = "CHERCHE|Z| CHEZ CHER"`

`m = "|C|HEZ"`

`reponse = [4]`

`i = 7`

`j = 0`

Échec de la correspondance, sortie de la boucle. 7 non incorporé.

Mise en œuvre

```
s = "CHERCHEZ|_|CHEZ CHER"

m = "|C|HEZ"

reponse = [4]

i = 8

j = 0
```

Échec de la correspondance, sortie de la boucle. 8 non incorporé.

Mise en œuvre

```
s = "CHERCHEZ |C|HEZ CHER"

m = "|C|HEZ"

reponse = [4]

i = 9

j = 0
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ C|H|EZ CHER"

m = "C|H|EZ"

reponse = [4]

i = 9

j = 1
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ CH|E|Z CHER"

m = "CH|E|Z"

reponse = [4]

i = 9

j = 2
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ CHE|Z| CHER"

m = "CHE|Z|"

reponse = [4, 9]

i = 9

j = 3
```

Correspondance puis sortie de la boucle. 9 incorporé.

Puis quatre échecs immédiats qu'il est inutile de détailler...

Mise en œuvre

```
s = "CHERCHEZ CHEZ |C|HER"

m = "|C|HEZ"

reponse = [4, 9]

i = 14

j = 0
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ CHEZ C|H|ER"

m = "C|H|EZ"

reponse = [4, 9]

i = 14

j = 1
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ CHEZ CH|E|R"

m = "CH|E|Z"

reponse = [4, 9]

i = 14

j = 2
```

Correspondance : on reste dans la boucle.

Mise en œuvre

```
s = "CHERCHEZ CHEZ CHE|R|"

m = "CHE|Z|"

reponse = [4, 9]

i = 14

j = 3
```

Échec de la correspondance, sortie de la boucle. 14 non incorporé.

Mise en œuvre

On quitte la boucle inconditionnelle car on avait (non écrits pour ne pas encombrer) `n = 18` et `k = 4` d'où les quinze tours de boucle effectués pour `i` allant de `0` à `14`.

Fin de l'algorithme, les indices de départ de la chaîne `"CHEZ"` en tant que facteur de la chaîne `"CHERCHEZ CHEZ CHER"` sont `4` et `9`.

Nombre de comparaisons de caractères effectuées : vingt-sept.