**Sorbonne Université**

**LU3IN003 (2019/2020)**

**Mini Projet : Alignement de séquences**

**CAI Eddy & MOUKOURI Steve**

# Alignement de deux mots

# Question 1

Montrer que si ,) et ,) sont respectivement des alignements de (x, y) et (u, v), alors ) est un alignement de (x , y ). Justifiez votre réponse

Supposons que ,) et ,) sont respectivement des alignements de (x, y) et (u, v), c’est-à-dire :

Et

Et montrons que ) est un alignement de (x , y ), c’est-à-dire :

On a (d’après (i) et (i’))

On a (d’après (ii) et (ii’))

On a = = = (d’après (iii) et (iii’))

Soit k, c’est-à-dire avec m = et n =, montrons que ou .

On a :

Pour [1…m] on a : or d’après (iv) on a :

Pour [m+1…n] on a : or d’après (iv’) on a :

D’où le résultat attendu

## Question 2

Si x est un mot de longueur n et y est de longueur m, quelle est la longueur maximale d’un alignement de (x, y) ? **NB**: un maximum est un majorant atteint. Vous devez donc justifier par un exemple que la longueur que vous proposez est réalisée par au moins un alignement.

Soit x et y deux mots quelconques, on définit n = et m =, ce qui nous donne l’alignement maximale suivant :

La longueur maximale d’un alignement de (x, y) est donc .

# Méthode naïve par énumération

## Question 3

Etant donné xun mot de longueur n, combien y a-t-il de mots obtenus en ajoutant à x exactement k gaps ? Autrement dit combien y a-t-il de mots tels que et  ? On exprimera cette valeur sous forme d’un coefficient binomiale.

Etant donné xun mot de longueur n et en ajoutant à x exactement k gaps, on a :

Mots obtenus

## Question 4

On chercher maintenant à en déduire le nombre d’alignements possibles d’un couple de mots (x, y) de longueurs respectives n et m, en supposant que n ≥ m. Une fois ajoutés k gaps à x pour obtenir un mot , combien de gaps seront ajoutés à y ? Combien y a-t-il de façon d’insérer ces gaps dans y sachant qu’un gap du mot ainsi obtenu ne doit pas être placé à la même position qu’un gap de ? En déduire le nombre d’alignements possibles de (x, y). On ne demande pas de simplifier l’expression obtenue, mais vous calculerez sur machine, à l’aide de cette expression, le nombre d’alignements possibles pour et

Soit (x, y) deux mots de longueurs respectives n et m en supposant que n ≥ m. Une fois ajouté k gaps à x pour obtenir , il faut ajouter gaps à y pour obtenir . Il y a donc façon d’insérer ces gaps dans y avec .

De ce fait, on trouve :

Nombres d’alignement possibles

Exemple avec et :

## Question 5

Quel genre de complexité temporelle aurait un algorithme naïf qui consisterait à énumérer tous les alignements de deux mots en vue de trouver la distance d’édition entre ces deux mots ? En vue de trouver un alignement de coût minimal ?

En vue de trouver la distance d’édition entre deux mots et de trouver un alignement de coût minimal, il faut dans un premier temps énumérer tous les alignements de mots, puis dans un second temps calculer leurs coûts respectifs afin de trouver le coût minimal. La complexité dépend donc de l’algorithme d’énumération et ainsi de l’algorithme de tri. La complexité finale sera donc en .

## Question 6

Quelle complexité spatiale (ordre de grandeur de la place requise en mémoire) aurait un algorithme naïf qui consisterait à énumérer tous les alignements de deux mots en vue de trouver la distance d’édition entres ces deux mots ? en vue de trouver un alignement de coût minimal ?

Il faut des tableaux à une dimension pour stocker l’alignements de deux mots et un tableau à une dimension pour stocker les coûts de chaque alignement, la complexité spatiale est donc en

# Programmation dynamique

## Question 7

Soit un alignement de de longueur l, Si , que vaut ? Si , que vaut ? Si , que valent ? Justifiez rapidement.

Si , est une lettre dans

Si , est une lettre dans

Si , sont des lettres dans

## Question 8

En distinguant les trois cas envisagés à la question 7, exprimer C à partir de C. Aucune justification n’est attendue.

Si et , )

Si et , )

Si ,)

Avec

Avec

## Question 9

Pour , déduire des questions 7 et 8 l’expression de D (i, j) à partir des valeurs de D à des rangs plus petits, c’est-à-dire à partir des termes D (i’, j’) où i’≤ i, j’≤ j et (i’, j’) (i, j). Justifiez votre réponse.

D (i, j) = d (

Or, D (i, j) = d (

On a donc, si

## Question 10

Que vaut D (0, 0) ? Justifiez.

D (0, 0) = 0 car le seul alignement possible pour 2 mots (x, y) avec x = et y = est (. Vu qu’ils sont identique, (

## Question 11

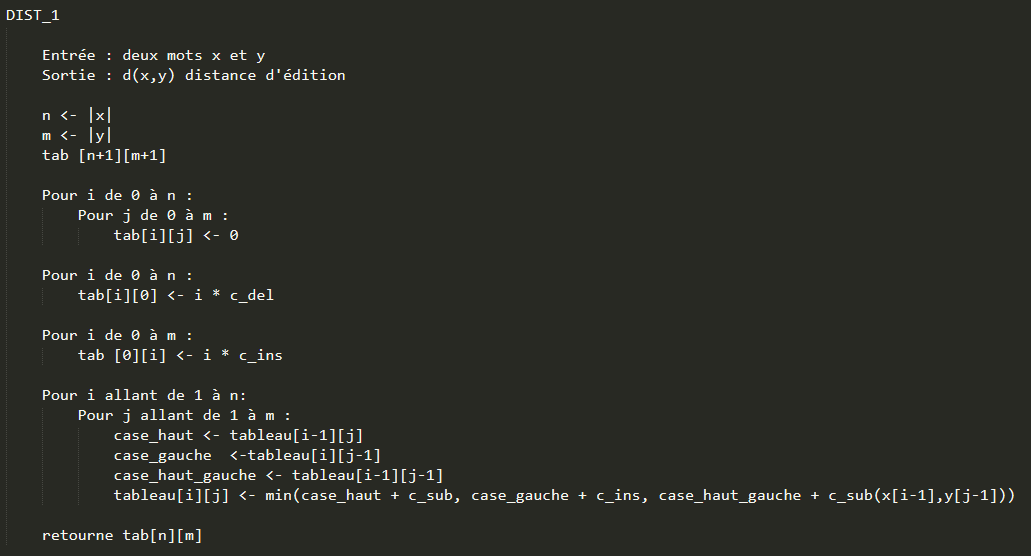
Que vaut D (0, j) pour ? Que vaut D (i, 0) pour ? Justifiez.

D (0, j) = 2 car les seuls alignements possibles pour 2 mots (x, y) avec x = est une insertion de gap pour chaque lettre de y. On a donc un coût de .

D (i, 0) = 2 car les seuls alignements possibles pour 2 mots (x, y) avec y = est une suppression de gap pour chaque lettre de x. On a donc un coût de .

## Question 12

En s’appuyant sur les réponses aux questions 9, 10 et 11, donner le pseudo-code d’un algorithme itératif nommé DIST\_1, qui prends en entrée deux mots, qui remplit un tableau à deux dimensions T avec toutes les valeurs de D pour finalement renvoyer la distance d’édition entre ces deux mots.



## Question 13

Quelle est la complexité spatiale de l’algorithme DIST\_1 ?

La complexité spatiale de l’algorithme DIST\_1 est

## Question 14

Quelle est la complexité temporelle de l’algorithme DIST\_1 ? Justifiez rapidement.

La complexité temporelle de l’algorithme DIST\_1 est . L’algorithme est constitué de 2 boucles qui font respectivement itérations et itérations, il est donc bien en .

## Question 15

Soit (i,. Montrer que :

Si j > 0 et = +, alors

Si i > 0 et = +, alors

Si , alors

Vous pouvez ne développer que l’un des trois cas au choix.

Supposons = +, avec j > 0

Montrons que :

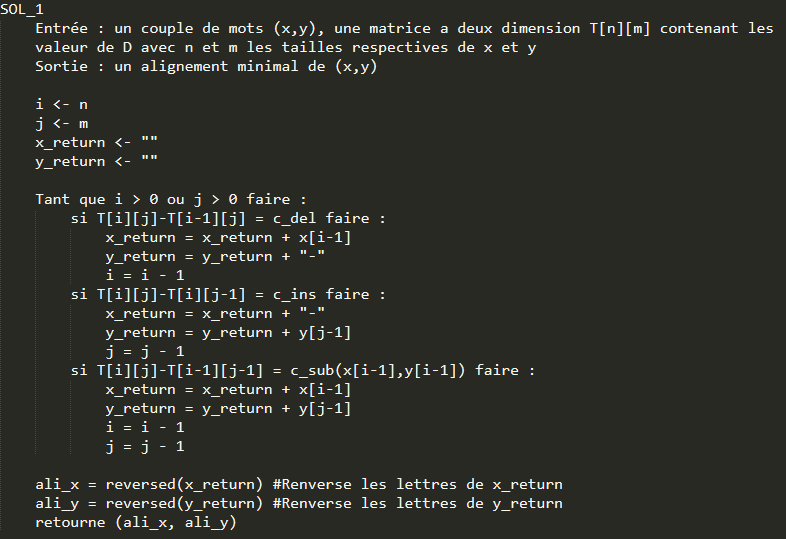
Soit on a :

Or,

par hypothèse

## Question 16

Donner le pseudo-code d’un algorithme itératif nommé SOL\_1, qui à partir d’un couple de mots (x, y) et d’un tableau T indexé par contenant les valeurs de D, renvoie un alignement minimal de (x, y).



## Question 17

En combinant les algorithmes DIST\_1 et SOL\_1 avec quelle complexité temporelle résout-on le problème ALI ?

En combinant les algorithmes DIST\_1 et SOL\_1, on obtient donc soit , car DIST\_1 est en et SOL\_1 est composé d’une seule boucle qui fait au plus itérations.

## Question 18

En combinant les algorithmes DIST\_1 et SOL\_1 avec quelle complexité spatiale résout-on le problème ALI ?

En combinant les algorithmes DIST\_1 et SOL\_1, on obtient une complexité mémoire soit , car on a besoin d’une matrice pour stocker les coûts de chaque alignement et de deux chaines de caractères de taille maximale pour renvoyer l’alignement optimal.

# Amélioration de la complexité spatiale du calcul de la distance

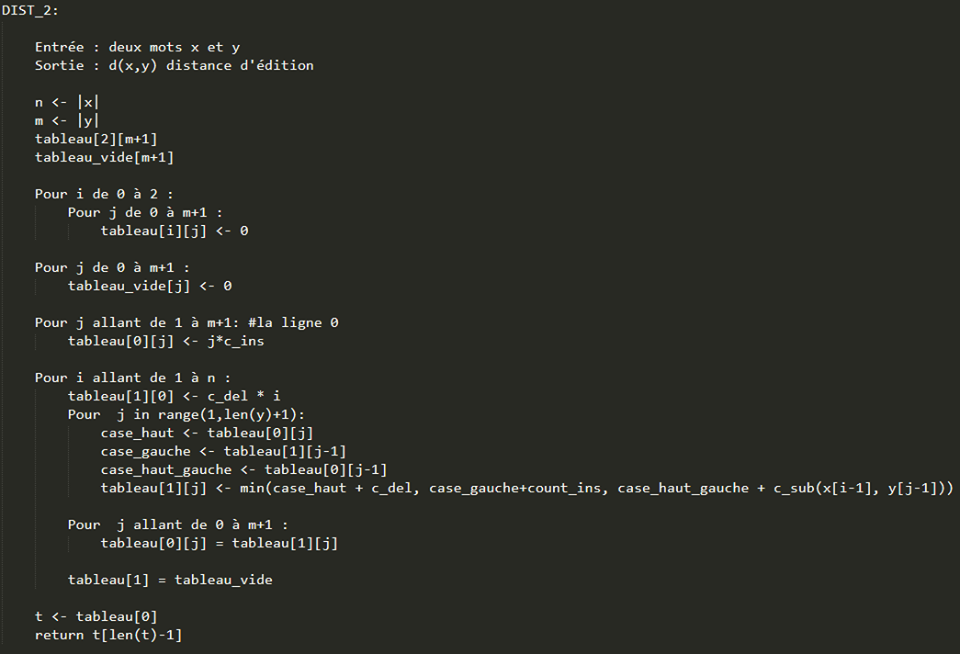
## Question 19

Expliquer pourquoi lors du remplissage de la ligne i > 0 du Tableau T dans l’algorithme DIST\_1, il suffirait d’avoir accès aux lignes et i du tableau (partiellement rempli pour cette dernière).

Pour compléter la case du tableau dans DIST\_1, il est nécessaire d’avoir accès aux 3 cases suivantes : , et . Ces cases correspondent aux 3 cas indiqué dans la question 15, ainsi il suffit donc d’avoir accès seulement aux lignes et du tableau pour le remplir.

## Question 20

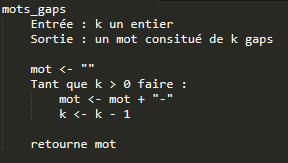
En utilisant la remarque de la question précédente, donner le pseudo-code d’un algorithme itératif DIST\_2, qui a la même spécification que DIST\_1, mais qui a une complexité linéaire (en ).



# Amélioration de la complexité spatiale du calcul d’un alignement optimal par la méthode « diviser pour régner »

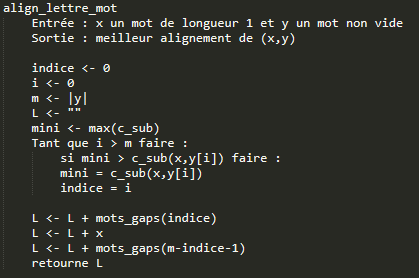
## Question 21

Donner le pseudo code d’une fonction mot\_gaps qui, étant donné un entier naturel k, renvoie le mot constitué de k gaps.



## Question 22

Donner le pseudo code d’une fonction align\_lettre\_mot qui, étant donné x un mot de longueur 1 et y un mot non vide de longueur quelconque, renvoie un meilleur alignement de (x, y).



## Question 23

On considère un exemple où est l’alphabet latin, constitué de 26 lettres majuscules, x = BALLON et y = ROND. On coupe x et y en leur milieux : x1 = BAL, x2 = LON, y1 = RO et y2 = ND.

On fixe , et

Donner un alignement optimal de et un alignement optimal de . Montrer que n’est pas un alignement optimal de (x, y). Pas besoin de justifier l’optimalité de et

Est un alignement optimal de avec :

: le coût est 13

est un alignement optimal de avec :

: le coût est 9

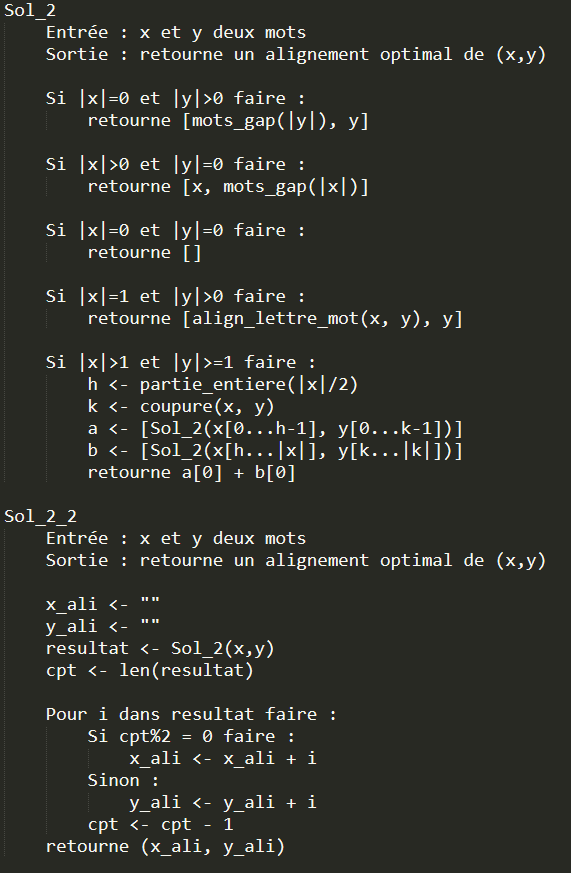
n’est pas un alignement optimal de (x, y) car il existe un meilleur alignement :

Etant donné que le coût de vaut la somme des coûts de et qui est égale à 13 + 9 = 21 et qu’on a 21 > 17, on peut donc conclure que n’est pas un alignement optimal de (x, y).

## Question 24

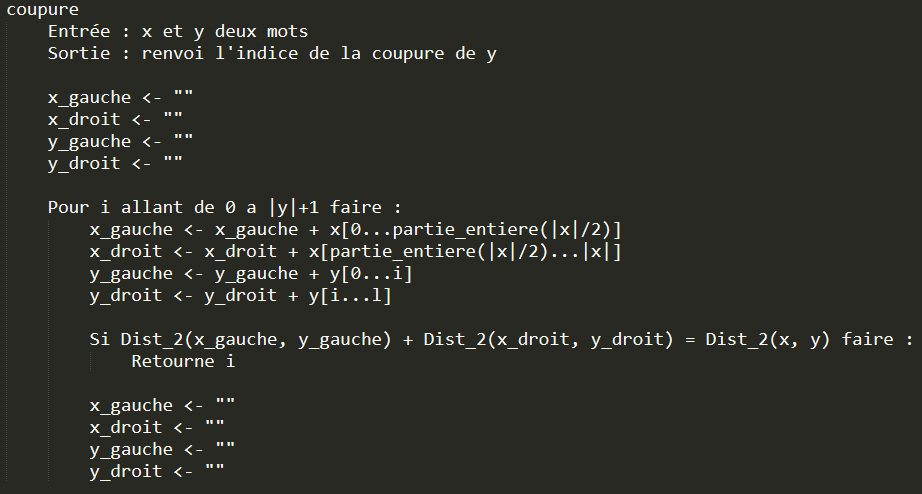
En supposant disposer de la fonction coupure, donner le pseudo code de l’algorithme récursif de type diviser pour régner, nommé SOL\_2, qui à partir d’un couple de mots (x, y) calcule un alignement minimal de (x, y).

Notre fonction Sol\_2 nous renvoie une liste contenant les lettres du meilleur alignement de x aux indices pairs et les lettres du meilleur alignement de y aux indices impairs. La fonction Sol\_2\_2 nous permet donc de recomposer et de renvoyer (, le meilleur alignement du couple (x, y).



## Question 25

Donner le pseudo-code d’une fonction coupure telle que décrite ci-dessus.



## Question 26

Quelle est la complexité spatiale de coupure ? Justifiez rapidement.

La complexité spatiale de coupure est en car on prend 4 sous-séquences à une dimension dans chaque boucle, et la complexité spatiale de DIST\_2 est .

## Question 27

Quelle est la complexité spatiale de SOL\_2 ? Justifiez rapidement.

La complexité spatiale de SOL\_2 est en car chaque appel récursif prend un espace de pour manipuler 2 séquences (gauche et droite) et seulement espace pour stocker le retour d’appel. Le nombre d’appel est inférieur à .

## Question 28

Quelle est la complexité temporelle de coupure ? Justifiez rapidement.

La complexité temporelle de coupure est en car on aura au maximum une boucle avec appels de DIST\_2. Sachant que la complexité de DIST\_2 est en , on a donc comme complexité :

## Question 29

A-t-on perdu en complexité temporelle en améliorant la complexité spatiale ? Comparez expérimentalement la complexité temporelle de SOL\_2 à celle de SOL\_1. On ne demande pas de preuve théorique.

et

Oui, on a amélioré la complexité spatiale en perdant en complexité temporelle.