

Projet d'Algorithmique II : Un problème de tomographie discrète

LU3IN003 : Groupe 3

PINHO FERNANDES Enzo - 21107465

DURBIN Deniz Ali - 21111116

2023

Novembre

Table des matières

1	Méthode incomplète de résolution	2
1.1	Première étape	2
1.2	Généralisation	5

1 Méthode incomplète de résolution

1.1 Première étape

Question 1 Si l'on a calculé tous les $T(j, l)$, comment savoir s'il est possible de colorier la ligne l_i entière avec la séquence entière ?

Il est possible de colorier la ligne l_i avec la séquence entière en vérifiant si $\forall j \in \{0, \dots, M-1\}, T(j, k)$ est défini comme vrai.

Question 2 Pour chacun des cas de base 1, 2a et 2b, indiquez si $T(j, l)$ prend la valeur vrai ou faux, éventuellement sous condition.

Pour commencer, formulons les règles générales dans une formule. Afin que $\forall j \in 1, \dots, M-1, \forall l \in 1, \dots, k, T(j, l)$ soit défini comme vrai, il faut que deux conditions soient remplies.

- Les l premiers blocs de la séquence sont placés dans les $j+1$ premières cases de la ligne.
- Il doit y avoir exactement $l-1$ cases blanches, dans les $j+1$ premières cases, qui serviront de séparateur de blocs.

Avec cela, nous pouvons constater que cette formule doit être respectée dans n'importe quel cas : $j+1 \geq (\sum_{i=1}^l s_i) + l-1$
Isolons s_l de la somme, et j du reste pour plus de maniabilité, et nous nous retrouvons avec la formule suivante :

$$j \geq (\sum_{i=1}^{l-1} s_i) + (s_l - 1) + (l - 1)$$

1. Cas $l = 0$ (pas de bloc), $j \in \{0, \dots, M-1\}$:

Il n'y a pas de bloc à placer dans la séquence, par conséquent nous pouvons colorier toutes les cases en blanc.

$T(j, l)$ est donc vrai $\forall j \in \{0, \dots, M-1\}$ si $l = 0$.

2. Cas $l \geq 0$ (au moins un bloc) :

a. $j < s_l - 1$:

Nous savons que $l \geq 1$ dans ce cas, par conséquent d'après la formule à respecter, nous aurions : $j \geq (\sum_{i=1}^{l-1} s_i) + (s_l - 1) + (l - 1) \geq s_l - 1$

Nous avons une contradiction, étant donné que nous sommes censés avoir : $j < s_l - 1$.

$\forall l \geq 1, T(j, l)$ est faux si $j < s_l - 1$.

b. $j = s_l - 1$:

Afin de répondre, nous devons séparer deux sous-cas distincts :

— Cas $l = 1$: $j \geq (\sum_{i=1}^{l-1} s_i) + (s_l - 1) + (l - 1) = s_l - 1$

Toutes les conditions sont respectées.

$T(j, l)$ est vrai si $j = s_l - 1$ et $l = 1$.

- Cas $l > 1$: $j \geq (\sum_{i=1}^{l-1} s_i) + (s_l - 1) + (l - 1) > s_l - 1$
 Il y a contradiction entre $j = s_l - 1$ et $j > s_l - 1$.
 $T(j, l)$ est faux si $j = s_l - 1$ et $l > 1$.
-

Question 3 Exprimez une relation de récurrence permettant de calculer $T(j, l)$ dans le cas 2c en fonction de deux valeurs $T(j', l')$ avec $j' < j$ et $l' \leq l$.

Nous avons deux possibilités pour la case (i, j) , il suffit qu'une des deux soit vérifiée :

1. La case (i, j) est blanche :

La case est blanche, par conséquent il faut vérifier si nous pouvons placer le dernier bloc s_l à la case précédente $j - 1$, avec la même séquence. Il faut donc vérifier que $T(j - 1, l)$ soit vrai.

2. La case (i, j) est noire :

La case est noire, par conséquent, le bloc s_l occupe les s_l dernières cases. La case précédant le bloc s_l , si elle existe, sera coloriée en blanc. Il faut donc vérifier que $T(j - s_l - 1, l - 1)$ soit vrai.

La relation de récurrence est donc : $T(j, l) = T(j - 1, l)$ OU $T(j - s_l - 1, l - 1)$

Question 4 Codez l'algorithme, puis testez-le.

Afin d'optimiser le code, nous avons utilisé la programmation dynamique. Nous enregistrons chaque résultat d'appels récursifs dans une matrice.

Le code source est src/partie1/Q4_isColorable.py

Le fichier test est src/partie1/Q4_isColorableTest.py

```

1 def isColorable(j : int, l : int, s : list[int], memo : list[list[int]] =
  None) -> bool :
2     """
3     Renvoie vrai s'il est possible de colorier les j+1 premières cases
    (i,0), ..., (i,j) de la ligne l_i avec la sous-séquence (s_1, ..., s_l)
    des l premiers blocs de la ligne_i.
4
5     Précondition : j = 0, ..., M-1 ; l = 1, ..., k; s[i] > 0 pour tout
    i, memo = None.
6     """
7
8     # Si c'est la premier appel à la fonction, on crée une matrice afin de
    sauvegarder en mémoire les résultats des appels récursifs.
9     if memo == None :
10         memo = [[None] * (l+1) for _ in range(j+1)]
11
12     # On vérifie si on a déjà calculé T(j,l). Si c'est le cas, on renvoie
    directement sa valeur.
13     if memo[j][l] != None :
14         return memo[j][l]
15
16     # Cas (1)
17     if (l == 0) :
18         memo[j][l] = True
19
20     # Cas (2a)
21     elif (j < s[-1] - 1) :
22         memo[j][l] = False
23
24     # Cas (2b)
25     elif (j == s[-1] - 1) :
26         if (l == 1) :
27             memo[j][l] = True
28         else :
29             memo[j][l] = False
30
31     # Cas (2c)
32     else :
33         memo[j][l] = isColorable(j-1, l, s, memo) or isColorable(j-s[-1]-1,
    l-1, s, memo)
34
35     return memo[j][l]
36
37 # PS : On retourne la valeur exceptionnellement au cas 1 pour éviter l'
    erreur => IndexError : list index out of range, dans le cas où l = 0.

```

1.2 Généralisation

Question 5 *Modifiez chacun des cas de l'algorithme précédent afin qu'il prenne en compte les cases déjà coloriées.*