Recherche des structures secondaires d'une chaîne d'ADN

Projet 2I003

LI Mengda, GE Zhichun

1er décembre 2017

1 Exercice 1

1.1

Comme un nucléotide ne peut former une paire avec lui même, donc

$$\forall i \in \{1, ..., n\}, \quad S_{i,i} = \{\} \text{ et } E_{i,i} = 0$$

1.2

1.2.1

Si ni i, ni j ne sont couplés dans $S_{i,j}$, alors

$$S_{i,j} = S_{i+1,j-1} \quad E_{i,j} = E_{i+1,j-1}$$

1.2.2

Si j n'est pas couplée dans $S_{i,j}$, alors

$$S_{i,j} = S_{i,j-1} \quad E_{i,j} = E_{i,j-1}$$

1.2.3

Si $(i, j) \in S_{i,j}$, alors

$$S_{i,j} = S_{i+1,j-1} \cup (i, j) \quad E_{i,j} = E_{i+1,j-1} + 1$$

1.2.4

Si $(k, j) \in S_{i,j}$ avec $k \in \{i + 1, ..., j - 1\}$, alors

$$S_{i,j} = S_{i,k-1} \cup S_{k,j} \quad E_{i,j} = E_{i,k-1} + E_{k,j}$$

car, par hypothèse, il n'existe pas de nœud : lorsqu'on coupe en k, on ne coupe pas de couple dans ce cas là, il n'y a donc pas de perte. Sinon, on a $S_{i,j} \subset S_{i,k-1} \cup S_{k,j}$ dans tous les cas.

1.3

1.3.1

Par la question 1.2, résumons en distinguant tous les cas possibles :

j
- couplé
  - couplé avec i — cas 3
  - couplé avec k — cas 4
- non couplé (cas 2)
  - i non couplé — cas 1
  - i couplé (en fait il n'est pas important) — cas 2

1. Soit ni i, ni j ne sont couplés dans $S_{i,j}$, alors $E_{i,j} = E_{i+1,j-1}$,
   et on a toujours $E_{i,j} \geq E_{i+1,j-1}$ dans tous les cas.
2. Soit j n'est pas couplée dans $S_{i,j}$, alors $E_{i,j} = E_{i,j-1}$,
   et on a toujours $E_{i,j} \geq E_{i,j-1}$ dans tous les cas.
3. Soit j est couplé avec i, c'est à dire $(i, j) \in S_{i,j}$, alors $E_{i,j} = E_{i+1,j-1} + 1$
4. Soit j est couplé avec un $k \in \{i + 1, ..., j - 1\}$, alors $E_{i,j} = E_{i,k-1} + E_{k,j}$,
   sinon $E_{i,j} \geq E_{i,k-1} + E_{k,j}$ car $S_{i,j} \supset S_{i,k-1} \cup S_{k,j} \quad \forall k \in \{i + 1, ..., j - 1\}$

Soit la fonction e :$\{1, ..., n\} \times \{1, ..., n\} \to \{0, 1\}$ qui vaut 1 ssi i et j peuvent être couplés.

Dans le cas 1 et cas 3, $E_{i,j} = E_{i+1,j-1} + e(i, j)$ et on a $E_{i,j} \geq E_{i+1,j-1} + e(i, j)$ dans tous les cas.

— si j est couplé avec i, $e(i, j) = 1$, $E_{i,j} = E_{i+1,j-1} + 1 = E_{i+1,j-1} + e(i, j)$,

— sinon $e(i, j) = 0$, on a toujours $E_{i,j} \geq E_{i+1,j-1} = E_{i+1,j-1} + e(i, j)$ par le 1.

Dans le cas 2, $E_{i,j} = E_{i,j-1}$ et on a $E_{i,j} \geq E_{i,j-1}$ dans tous les cas. car $S_{i,j} \supset S_{i,j-1}$

Par le cas 4, $\forall k \in \{i + 1, ..., j - 1\}$ $E_{i,j} \geq E_{i,k-1}+E_{k,j}$. Donc $E_{i,j} \geq \max\limits_{i<k<j} E_{i,k-1} + E_{k,j}$ dans tous les cas.

Et s'il existe $k_0 \in \{i + 1, ..., j - 1\}$ tel que $(k_0, j) \in S_{i,j}$, $E_{i,j} = E_{i,k_0-1} + E_{k_0,j}$
Dans ce cas-là, le max est atteint :
$E_{i,k_0-1} + E_{k_0,j} \geq \max\limits_{i<k<j} E_{i,k-1} + E_{k,j}$ et $\max\limits_{i<k<j} E_{i,k-1} + E_{k,j} \geq E_{i,k_0-1} + E_{k_0,j}$
D'où $E_{i,j} = E_{i,k_0-1} + E_{k_0,j} = \max\limits_{i<k<j} E_{i,k-1} + E_{k,j}$ dans le cas 4

En conclusion : $E_{i,j}$ est un majorant de l'ensemble $\{E_{i+1,j-1} + e(i, j),\, E_{i,j-1},\, \max\limits_{i<k<j} E_{i,k-1}\}$, et ce majorant est atteint : il y a toujours un cas où la condition d'égalité est vrai. Donc

$$E_{i,j} = \max\{E_{i+1,j-1} + e(i, j),\, E_{i,j-1},\, \max_{i<k<j} E_{i,k-1} + E_{k,j}\}$$

...

1.3.2

Si k = j, $E_{k,j} = E_{j,j} = 0$ par 1.1, donc $E_{i,j-1} = (E_{i,j-1} + E_{j,j}) \in \{E_{i,k-1} + E_{k,j} \mid i < k \leq j\}$, et

$$E_{i,j-1} \leq \max\{E_{i,k-1} + E_{k,j} \mid i < k \leq j\}$$

On en déduit que $E_{i,j} = \max\{E_{i+1,j-1} + e(i, j),\, \max\limits_{i<k\leq j} E_{i,k-1} + E_{k,j}\}$

1.4

```
def tailleMaxRec(a: str, i: int, j: int) -> int:
    n= len(a)
    assert 1 <= i <= n and 1 <= j <= n, 'impossible de tronquer'

    if a == '' or i >= j:
        return 0

    def couple(i: int, j:int) -> bool:
        return {a[i-1], a[j-1]} in [ {'A','T'}, {'C','G'} ]

    def e(i: int, j: int) -> {0,1}:
        if couple(i,j):
            return 1
        else:
            return 0

    return max(tailleMaxRec(a, i+1, j-1) + e(i,j),
                max([tailleMaxRec(a, i, k-1) + tailleMaxRec(a, k, j)
                    for k in range(i+1, j+1)]) )
```

1.5

Cette fonction se termine car chaque fois on appelle récursivement sur une sous-partie de séquence [i+1, j-1], le début de sous-séquence i est strictement croissant et la fin de sous-séquence j est stricetement décroissante. Dans $\mathbb{N}$, l'ordre est bien fondé donc i, j se rencontre en certaine récursion. Lorsque i, j se rencontre, c'est à dire $i \leq j$, la fonction retourne une valeure et se termine en dépillant récursivement.

Cette fonction calcule exactement $\max\{E_{i+1,j-1} + e(i,j), \max_{i<k<j} E_{i,k-1} + E_{k,j}\}$, elle est valide selon la question 1.3.

1.6

1.6.1

Pour u_0, $(p = 0) \Rightarrow (i = j)$. Donc l'appel de fonction retourne simplement 0, $u_0 = 1$

Pour u_1, $(p = 1) \Rightarrow (i + 1 = j)$, donc cet appel retourne

```
max(tailleMaxRec(a, i+1, j-1) + e(i,j),
                max([tailleMaxRec(a, i, k-1) + tailleMaxRec(a, k, j)
                    for k in range(i+1, j+1)]) )
```

Vu que i + 1 = j, tailleMaxRec(a, i+1, j-1) retourne simplement 0 car $i + 1 = j \geq j - 1$.
Analysons la dernière itérable

```
[tailleMaxRec(a, i, k-1) + tailleMaxRec(a, k, j) for k in range(i+1, j+1)]
```

range(j ,j+1) ne contient que j, donc cette liste n'a qu'un élément, cela vaut dire qu'on appelle deux fois tailleMaxRec. En sommant, $u_1 = 3$

1.6.2

Soit $p \leq 2$, pour u_p

- On appelle d'abord une fois la fonction elle-même : tailleMaxRec(a, i, j), on a 1 appel.
- Et puis on appelle tailleMaxRec (a, i+1, j−1) ,
  $(j-1)-(i+1) = (j-i)-2 = 0$, on a donc u_0 appel(s).
- Enfin, pour

```
tailleMaxRec(a,i,k-1) + tailleMaxRec(a,k,j) for k in range(i+1, j+1)
```

range(i+1, j+1) équivaut à $[\![i+1, i+p]\!]$, car p = j - i, j = i + p
Le nombre d'appel de tailleMaxRec(a, i, k−1) équivaut u_{k-1-i},
Le nombre d'appel de tailleMaxRec(a, k, j) équivaut u_{p+i-k}
On a donc $\sum_{k=i+1}^{i+p} u_{k-1-i} + \sum_{k=i+1}^{i+p} u_{p+i-k}$ d'appels.
Pour la première, en éffectuant un changement de variable q = k- (i+1), on a bien $\sum_{q=0}^{p-1} u_q$
Pour la deuxième, en éffectuant un changement de variable q = p + i - k, on a $\sum_{q=p-1}^{0} u_q$ qui vaut en effet $\sum_{q=0}^{p-1} u_q$.
En sommant les deux sommes, on a bien $2\sum_{i=0}^{p-1} u_i$

En conclusion, $u_p = u_0 + 1 + 2\sum_{i=0}^{p-1} u_i$