Structure d'ARN

20 novembre 2023 09:04

Entrée : séquence S d'ARN $\Sigma = \{A, C, G, U\}$
P = paires de nucléotides appariables (liste)

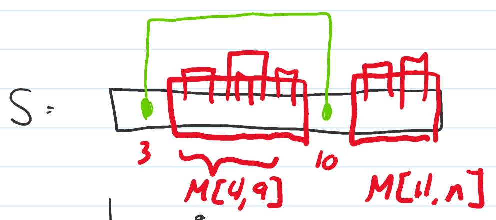
Sortie : liste A de paires de nucléotides, appelés appariements, tels que pour tous $(s_i, t_i), (s_j, t_j) \in A$ distincts, on a

$$s_i < s_j < t_i \Rightarrow s_i < t_j < t_i$$

ex :

$S = a\ c\ c\ u\ g\ a\ a\ g$ $\quad P = \{\{a, u\}, \{c, g\}\}$

Algo de Nussinov (1978)



- Par prog. dynamique

$M[i,j]$ = # d'appariements maximum possible dans $S[i,j]$

- Cas de base

$M[i,i] = 0 \quad \forall i = 1, \dots, n \quad n = |S|$

- Récurrence : $\forall i < j$

$$M[i,j] = \max \begin{cases} M[i+1,j] \\ \max\limits_{i<k\leq j} \left(M[i+1,k-1] + M[k+1,j] + \delta_{i,k}\right) \end{cases}$$

Cas possibles pour sol. opt. sur $S[i,j]$

① i est non-apparié, il suffit d'optimiser sur $S[i+1,j]$ $\quad [\cancel{i}\ \underbrace{i+1\ i+2\ \dots\ j}_{M[i+1,j]}]$

$\rightarrow$ cas ① $\Rightarrow M[i+1,j]$

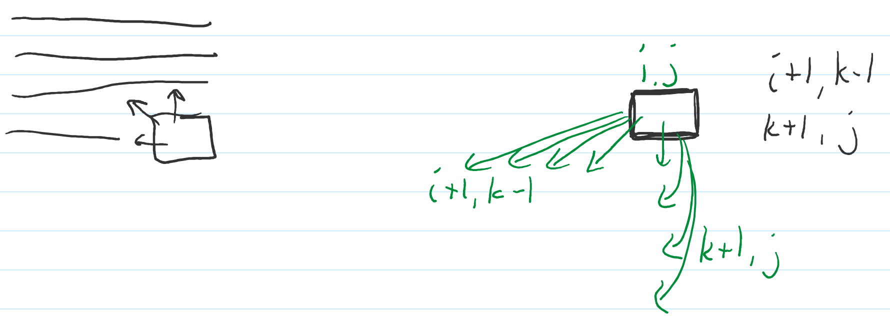
② i est apparié avec un certain $S[k]$, où $i < k \leq j$ $\quad [i\ \underbrace{i+1\ \dots}_{M[i+1,k-1]}\ k\ \underbrace{\dots\ j-1\ j}_{M[k+1,j]}]$

On ne connait pas ce k, alors on les essaie tous

Cas ② $\rightarrow \max\limits_{i<k\leq j} \left(M[i+1,k-1] + M[k+1,j] + \delta_{i,k}\right)$

où $\delta_{i,k} = \begin{cases} 1 & \text{si } \{S[i], S[k]\} \in P \\ -\infty & \text{sinon} \end{cases}$

align. global $\qquad$ Nussinov



$S = a\ c\ a\ g\ u\ c\ g$

1 2 3 4 5 6 7

S = acagucg

i \ j	1 a	2 c	3 a	4 g	5 u	6 c	7 g
1 a	0	0	1	1	2	1	3
2 c		0	0	1	1	1	1
3 a			0	0	1	1	1
4 g				0	0	1	1
5 u					0	0	1
6 c						0	1
7 g							0

$i=6 \quad j=7$

$i \quad j=k$

c $\quad$ g

$M[i+1, j-1] + M[j+1, j] + 1$

On calcule $M[i,j]$ récursivement

```
global M  (init None partout)
nussinov(S, i, j)
    si i ≥ j, return 0
    si M[i,j] ≠ None, return M[i,j]
    m1 = nussinov(i+1, j)        // M[i+1, j]
    m2 = -∞
    pour k = i+1, ..., j                          O(n) tours
        m2 = max(m2, nussinov(i+1, k-1) +
                     nussinov(k+1, j) + δ_{i,k})
    M[i,j] = max(m1, m2)
    return M[i,j]
```

Appel initial avec nussinov(1, n) pour calculer $M[1,n]$

Complexité : # d'entrées $M[i,j]$ × temps par entrée

$$O(n^2) \quad \times \quad O(n)$$

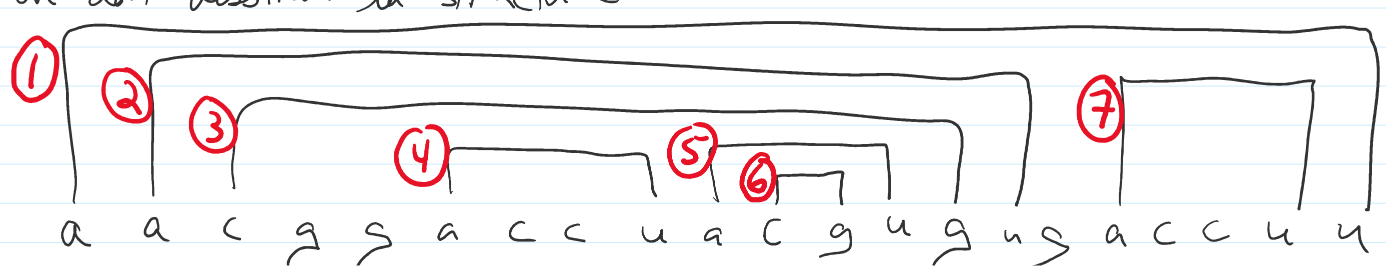
~n choix pour i
~n choix pour j

$$\rightarrow O(n^2 \cdot n) = O(n^3)$$

exo : reconstruire une liste concrète à partir de M.

Comment dessiner une structure 2D d'appariements

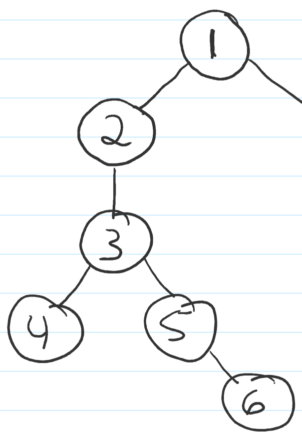
↳ on nous donne les appariements A
on doit dessiner la structure

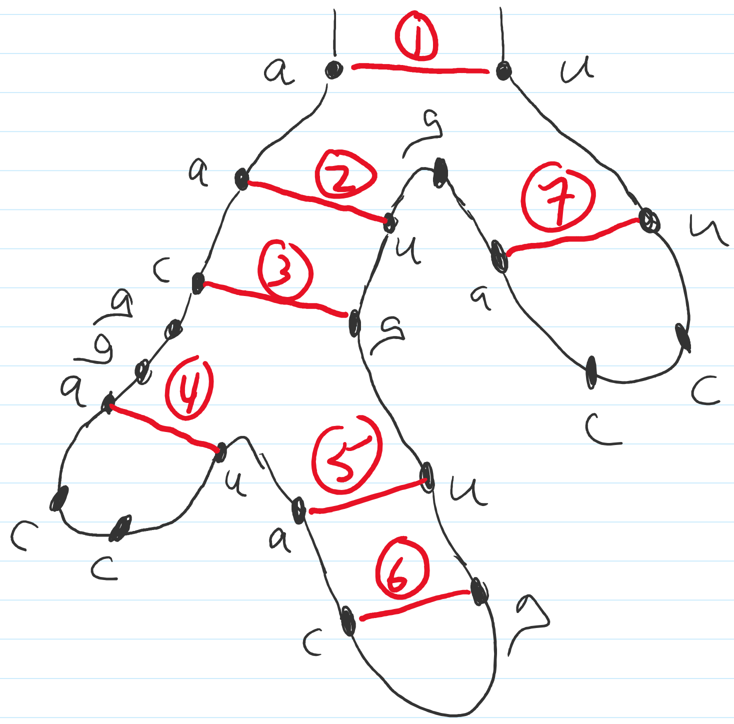


a a c g g a c c u a c g u g u g a c c u u

Les conditions de non-croisement permettent de construire un arbre d'appariements.

Noeuds = appar. A_1 parent de A_2 si A_2 est "emboîté" dans A_1



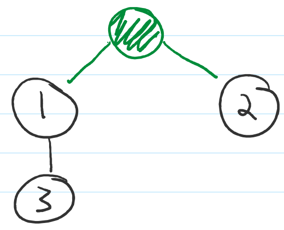


À savoir : comment construire l'arbre

- Note : il se peut qu'il y ait plusieurs racines.

ex : 1 3 2

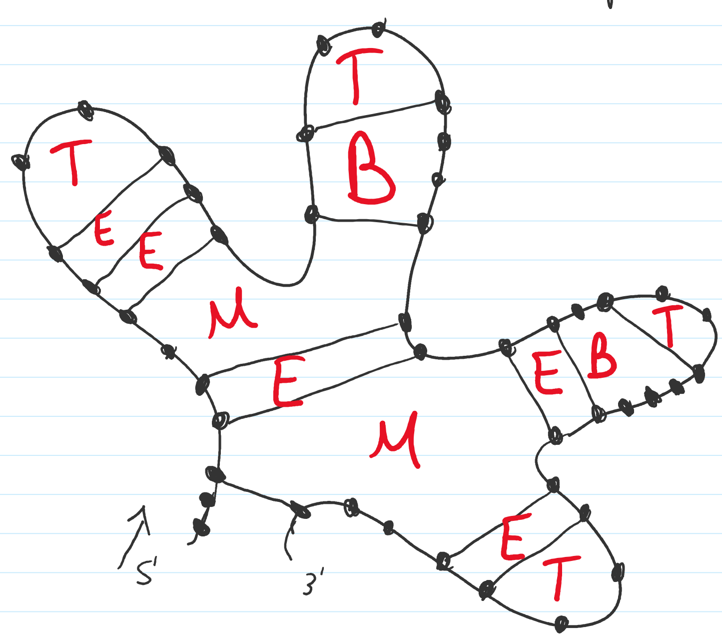
a g c u c c c g c



Dans ce cas, on met une racine "virtuelle"

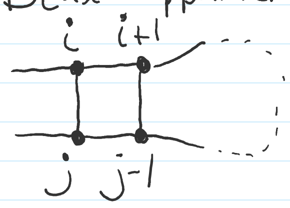
Prédictions de sous-structures

- Sous-structures possibles sur un ARN (sans pseudo-noeuds / croisements)



① Empilement E

Deux appariements consécutifs



Appariements (i, j) (i+1, j-1)

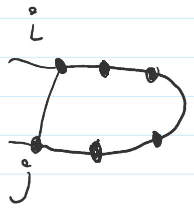
② Boucle interne B

Deux appariements consécutifs, mais avec des nucl. non-appariés sur le chemin.

③ Tige-boucle T

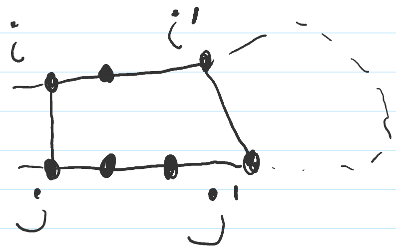
Appariement avec aucun autre

Appariement avec aucun autre entre



(i,j) avec aucun (i',j') tel que $i<i'<j'<j$

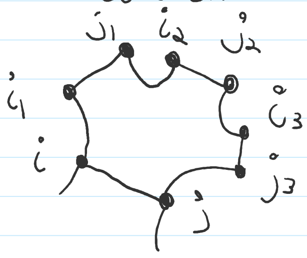
mais avec des nucl. non-appariés sur le chemin.



Appariements (i,j) (i',j') avec $i<i'<j'<j$ et $\{i',j'\} \neq \{i+1, j-1\}$

④ Multi-boucle $\mathcal{M}$

Sous-structure de branchement



Suite d'appariements (i,j) (i_1,j_1) $(i_2,j_2), \ldots, (i_k,j_k)$

telle que $i < i_1 < j_1 < i_2 < j_2 < \ldots < i_k < j_k < j$

Entrée: séquence d'ARN S,

fonction e de coûts des sous-structures

<0 $\quad eE(i,j)$ = coût d'empilement à (i,j)

>0 $\quad eB(i,j,i',j')$ = coût d'une boucle

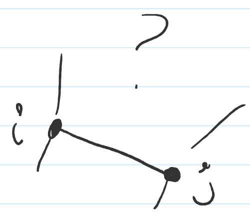
>0 $\quad eT(i,j)$ = coût de tige-boucle

>0 $\quad eM(i,j,i_1,j_1,\ldots,i_k,j_k)$ = coût de multi-boucle

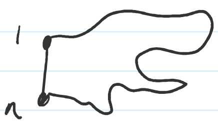
Sortie: liste d'appariements dont la structure 2D minimise la somme des coûts des sous-structures

Par prog. dynamique

$M[i,j]$ = coût min. en énergie pour $S[i..j]$ avec la contrainte que (i,j) soient appariées

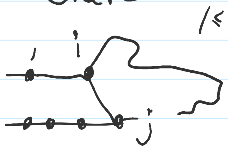


On cherche $M[1,n]$?



Non

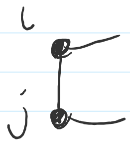
On cherche $\min\limits_{1\leq i<j\leq n} M[i,j]$



Cas de base: $i=j$

$M[i,i] = \infty$

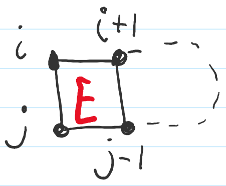
Récurrence: $i<j$



On liste les cas possibles, on prend le min

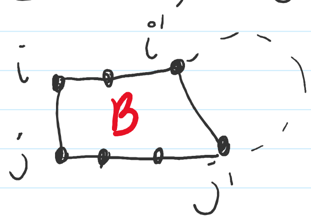
Cas ①: E, i,j sont le départ d'un empilement

Cas (1): E, i, j sont le départ d'un empilement



coût : $eE(i,j) + M[i+1, j-1]$ $O(1)$

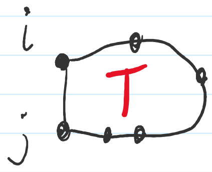
Cas (2) B, i, j sont le départ d'une boucle



On prend i', j' optimal qui ferme la boucle

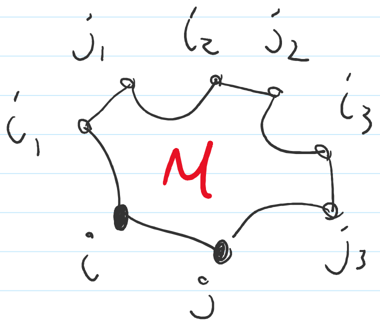
coût : $\min_{i<i'<j'<j} \left(eB(i,j,i',j') + M[i',j'] \right)$ $O(n^2)$

Cas (3) T, i, j = tige-boucle



coût : $eT(i,j)$ $O(1)$

Cas (4) M, i, j sont le début d'une multi-boucle



On essaie toutes les façons d'avoir cette multi-boucle

$$\min_{i<\underline{i_1<j_1}<\underline{i_2<j_2}<\dots<\underline{i_k<j_k}<j} \left(eM(i,j,i_1,j_1,\dots,i_k,j_k) + \sum_{l=1}^{k} M[i_l, j_l] \right)$$

On prend $M[i,j] = \min \begin{cases} (1) \\ (2) \\ (3) \\ (4) \end{cases}$

Complexité : horrible
dominée par (4)
↳ $O(n^{2k})$ avec k = nb max de branchements permis

Souvent, on prend $k = 2, 3$

Nombre de cas à énumérer pour un k donné
$= O(n^{2k})$

ex : $k=2 \rightarrow \underset{i_1}{n} \cdot \underset{j_1}{n} \cdot \underset{i_2}{n} \cdot \underset{j_2}{n} = n^4$

ex : $k=3 \rightarrow \underset{i_1}{n} \cdot \underset{j_1}{n} \cdot \underset{i_2}{n} \cdot \underset{j_2}{n} \cdot \underset{i_3}{n} \cdot \underset{j_3}{n} = n^6$

$\rightarrow \sum_{k=1}^{n/2} n^{2k}$ cas à évaluer