Combinatoire et **algorithmique** de l'ARN - Partie 2

Programmation dynamique

Alain Denise Yann Ponty

CNRS/LIX, École Polytechnique

AMIB, Inria Saclay

PIMS Vancouver
Simon Fraser University
Canada

http://tinyurl.com/EPIT14-Ponty http://goo.gl/Nic9kJ

http://www.lix.polytechnique.fr/~ponty/enseignement/2014-05-EPIT-RNA-Part2-Ponty.pdf

Résumé

1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique

Avant propos . . .

. . . ou *comment gagner 1 million$ (+garder votre job) en rendant la monnaie ! !*

Problème : Vous disposez de pièces de **1**, **20** et **50** centimes.
Comment rendre N sans surcharger inutilement les poches du client ?

Stratégie 1 : Commencer par les grosses pièces puis compléter.

$$
\begin{aligned}
21 &= ??\\
&=\\
&=\\
&=
\end{aligned}
$$

. . . ou *comment gagner 1 million$ (+garder votre job) en rendant la monnaie ! !*

Problème : Vous disposez de pièces de **1**, **20** et **50** centimes.
Comment rendre N sans surcharger inutilement les poches du client ?

Stratégie 1 : Commencer par les grosses pièces puis compléter.

21 = +

55 = ? ?

=

=

. . . ou *comment gagner 1 million$ (+garder votre job) en rendant la monnaie ! !*

Problème : Vous disposez de pièces de **1**, **20** et **50** centimes.
Comment rendre N sans surcharger inutilement les poches du client ?

Stratégie 1 : Commencer par les grosses pièces puis compléter.

21 = +

55 = + + + + +

60 = ? ?

=

. . . ou *comment gagner 1 million$ (+garder votre job) en rendant la monnaie ! !*

Problème : Vous disposez de pièces de **1**, **20** et **50** centimes.
Comment rendre N sans surcharger inutilement les poches du client ?

Stratégie 1 : Commencer par les grosses pièces puis compléter.

21 = +

55 =

60 = ??

=

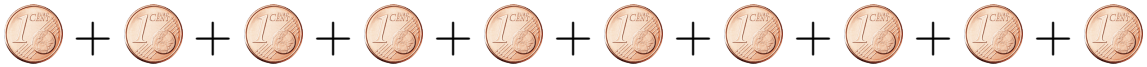
. . . ou *comment gagner 1 million$ (+garder votre job) en rendant la monnaie ! !*

Problème : Vous disposez de pièces de **1**, **20** et **50** centimes.
Comment rendre N sans surcharger inutilement les poches du client ?

Stratégie 1 : Commencer par les grosses pièces puis compléter.

21 = +

55 = +

60 = +  ??

= !

Problème non-résoluble (? !) en temps polynomial pour un jeux de pièces arbitraire ($\approx$ Pb. sac à dos NP-complet)

(Algo. polynomial $\Rightarrow$ P=NP $\Rightarrow$1M$)

Stratégie 2 : Récurrence sur nombre min. $\#MinPieces$ de pièces rendues :

$$\#MinPieces(N) = \text{Min}\begin{cases} \varnothing & \rightarrow & 0 \\ \text{ } & \rightarrow & 1 + \#MinPieces(N-1) \quad [\text{Si } N \geq 1] \\ \text{ } & \rightarrow & 1 + \#MinPieces(N-20) \quad [\text{Si } N \geq 20] \\ \text{ } & \rightarrow & 1 + \#MinPieces(N-50) \quad [\text{Si } N \geq 50] \end{cases}$$

Parcours exhaustif (redondant) de l'arbre des possibles $\rightarrow \Theta(\#\text{Pièces}^N)$

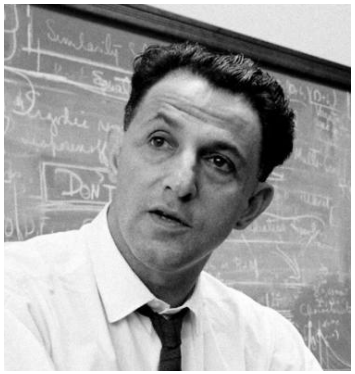
Programmation dynamique

Pour tout $N' \in 0, 1, 2 \rightarrow N$,

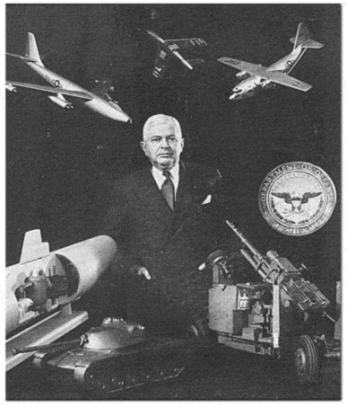
- Calculer $\#MinPieces(N')$ $\rightarrow \mathcal{O}(1)$
- Mémoriser le résultat

Reconstruire les pièces rendues en retraçant les termes contribuant au Min.
$\Rightarrow$ Algorithme en $\Theta(N \times \#MinPieces)$ [a]

a. Remarque : Algorithme correct pour tout jeux de pièces, mais on n'a pas gagné le million ($\not\Rightarrow$ P=NP), car N est potentiellement exponentiel sur son codage. La complexité de cet algorithme est donc exponentielle.



R. Bellman



C. E. Wilson

Programmation dynamique = Technique algorithmique de résolution des problèmes d'optimisation.

Principe : Partant d'une équation liant le score optimal d'un (sous-)problème à celui de certains de ses sous-problèmes,

- Calculer efficacement le score optimal du problème initial
- Reconstruire la (ou les) solution(s) associée(s) au score optimal

Technique robuste face à des altérations *locales* de la fonction objectif
$\Rightarrow$ Omniprésente en bioinformatique : Alignement, repliement...

Challenge

Trouver l'équation de programmation dynamique

Propriétés requises d'une équation de prog. dyn. =

1. Correction du score optimal hérité des sous-problèmes
2. Complétude dans le parcours de l'espace des solutions

Remarques : 2 peut être violé, par ex. pour élaguer des sous-optimaux.
2 + 3 $\Rightarrow$ Bijection : espace de recherche $\Leftrightarrow$ exécutions de la récurrence

Challenge

Trouver l'équation de programmation dynamique

Propriétés requises /souhaitables d'une équation de prog. dyn. =

1. Correction du score optimal hérité des sous-problèmes
2. Complétude dans le parcours de l'espace des solutions
3. Non-ambiguité de la décomposition

Remarques : 2 peut être violé, par ex. pour élaguer des sous-optimaux.
2 + 3 $\Rightarrow$ Bijection : espace de recherche $\Leftrightarrow$ exécutions de la récurrence

Challenge

Trouver l'équation de programmation dynamique

Propriétés requises /souhaitables d'une équation de prog. dyn. =

1. Correction du score optimal hérité des sous-problèmes
2. Complétude dans le parcours de l'espace des solutions
3. Non-ambiguité de la décomposition

Remarques : 2 peut être violé, par ex. pour élaguer des sous-optimaux.
2 + 3 $\Rightarrow$ Bijection : espace de recherche $\Leftrightarrow$ exécutions de la récurrence

Approche combinatoire peut aider :

- Grammaire/spécification $\rightsquigarrow$ Récurrences de comptage
  $\Rightarrow$ Complétude et non-ambiguité
- +Correction
  $\Rightarrow$ Équation de programmation dynamique

Alignement local de séquence (Algorithme Smith/Waterman)

Problème : Trouver l'alignement local

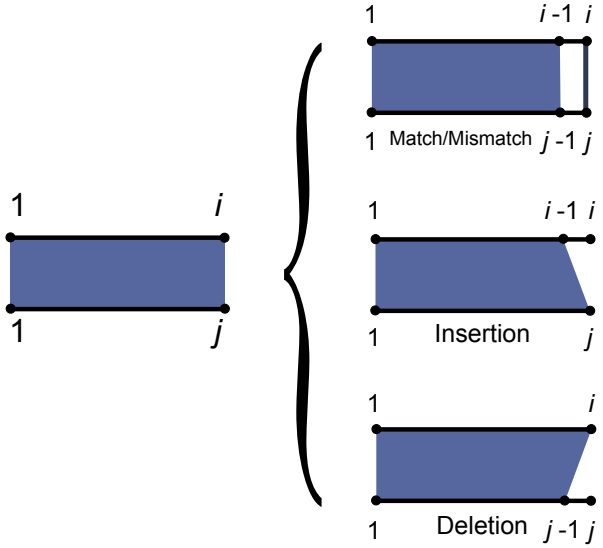
$$a = ((i_1, j_1), (i_2, j_2), \cdots, (i_k, j_k)), i_x < i_{x+1} \text{ et } j_x < j_{x+1},$$

de deux séquences s et t qui maximise la similarité

$$W(a) = \sum_{(i,j)\in a} m_{s_i,t_j} + \sum_{r\in[1,k-1]} (i_{r+1} - i_r - 1) \cdot p_i + (j_{r+1} - j_r - 1) \cdot p_d$$

Récurrence pour l'alignement optimal $W^\star_{i,j}$ des préfixes $[1, i]$ et $[1, j]$:

$$\begin{aligned} W^\star_{i,0} &= 0 \\ W^\star_{0,j} &= 0 \\ W^\star_{i,j} &= \max \begin{cases} W^\star_{i-1,j-1} + m_{s_i,t_j} \\ W^\star_{i-1,j} + p_i \\ W^\star_{i,j-1} + p_d \end{cases} \end{aligned}$$



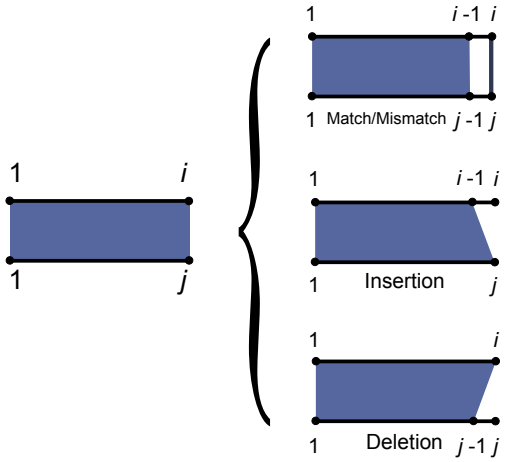
Exercices :

- Quelle complexité pour deux séquences de taille N et M ?
- Ce schéma est-il complet ? non-ambigu ? Comment le corriger ?

Complexité du calcul dépend de :

- Taille de l'espace des sous-problèmes
- Nombres de sous-problèmes considérés (#Termes décomposition)

$i : 1 \rightarrow N + 1 \Rightarrow \Theta(N)$
$j : 1 \rightarrow M + 1 \Rightarrow \Theta(M)$
Trois opérations en $\mathcal{O}(1)$ par sous-calcul
$\Rightarrow \Theta(M \cdot N)$ temps/mémoire



Ce schéma de programmation dynamique est ambigu, car on obtient l'alignement vide sur $\binom{M+N}{N}$ exécutions différentes :

A A A - - -
- - - B B B

$\Leftrightarrow$

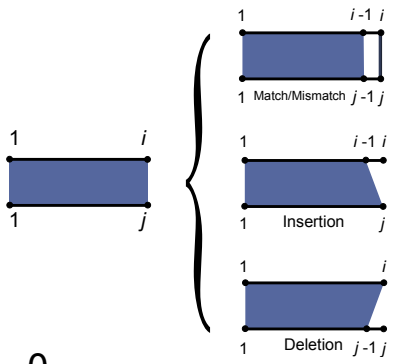
- - - A A A
B B B - - -

$\Leftrightarrow$

- A - A - A
B - B - B -

$\Leftrightarrow \quad \ldots$

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

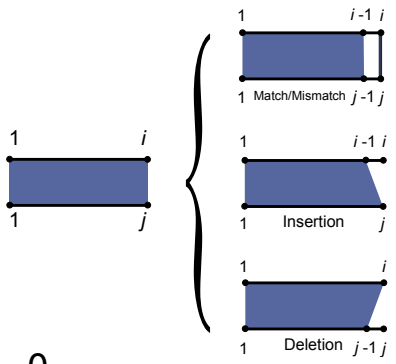


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0								
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

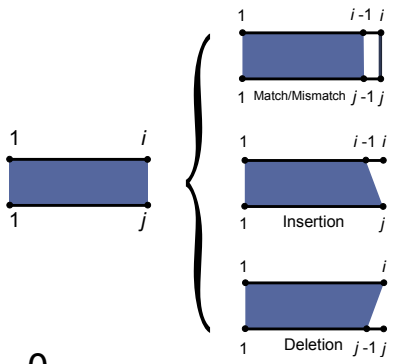


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2							
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



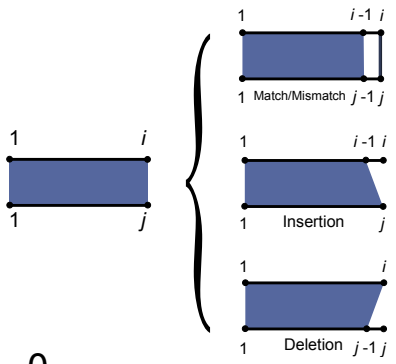
$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1						
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple complet

Exemple : Alignement local de séquences AGCACACA et ACACACTA
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

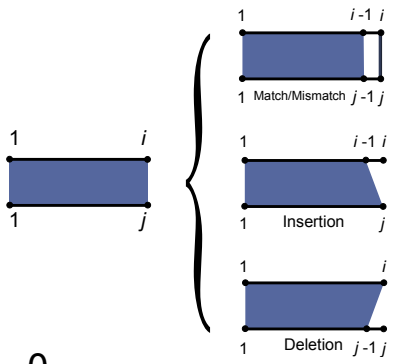


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2					
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



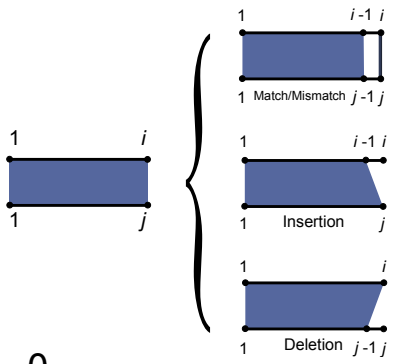
$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1				
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple complet

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



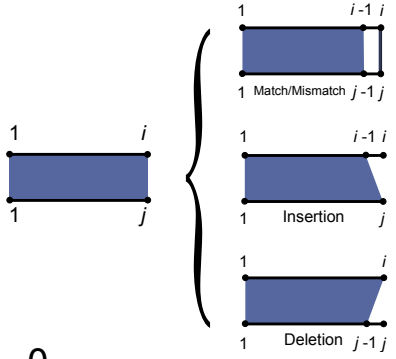
$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0								
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple complet

Exemple : Alignement local de séquences AGCACACA et ACACACTA
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

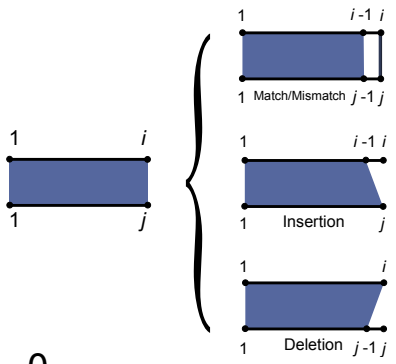


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0								
A	0								
C	0								
A	0								
C	0								
A	0								

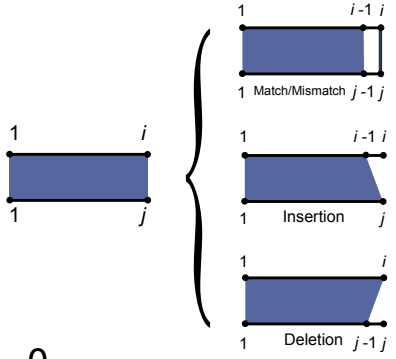
Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0								
C	0								
A	0								
C	0								
A	0								

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

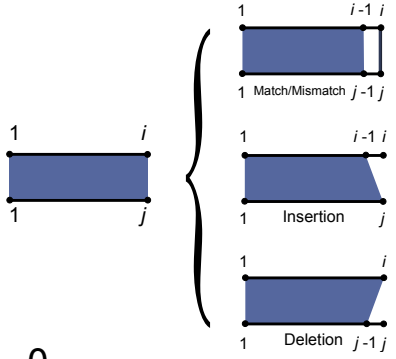


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0								
A	0								
C	0								
A	0								

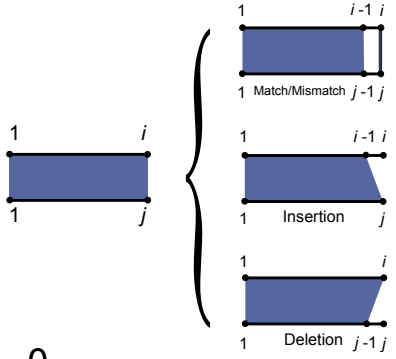
Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$

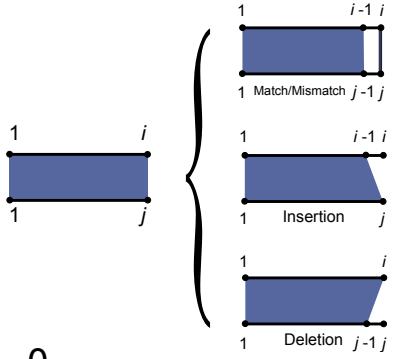


$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



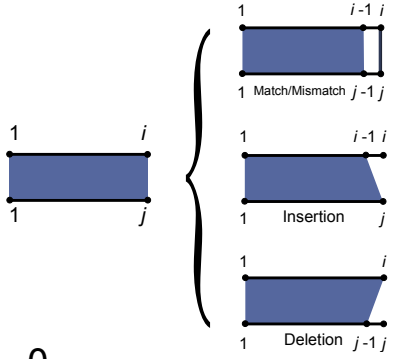
$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

Meilleur alignement

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



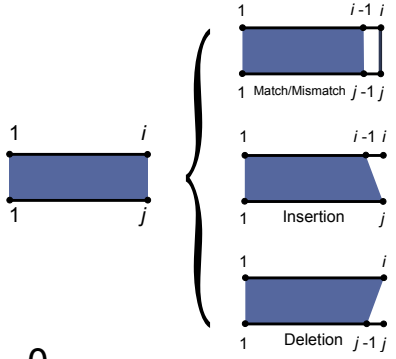
$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

Meilleur alignement

		A	C	A	C	A	C	T	A	
	0	0	0	0	0	0	0	0	0	
A	0	2	1	2	1	2	1	0	2	
G	0	1	1	1	1	1	1	0	1	
C	0	0	3	2	3	2	3	2	1	
A	0	2	2	5	4	5	4	3	4	
C	0	1	4	4	7	6	7	6	5	
A	0	2	3	6	6	9	8	7	8	
A C	0	1	4	5	8	8	11	10	9	
A	0	2	3	6	7	10	10	10	12	

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

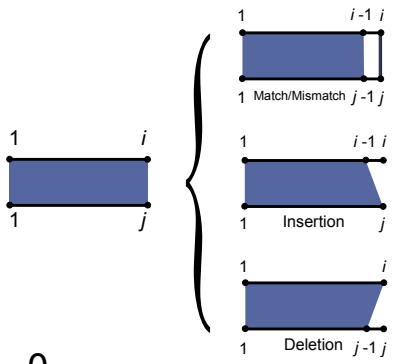
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

-	A
T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

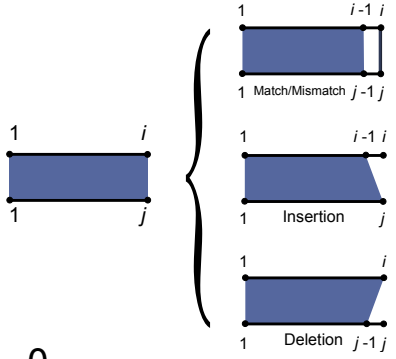
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

C	-	A
C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

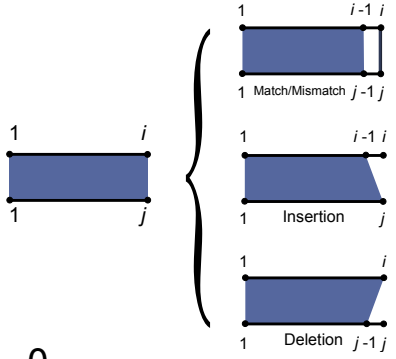
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

A	C	-	A
A	C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

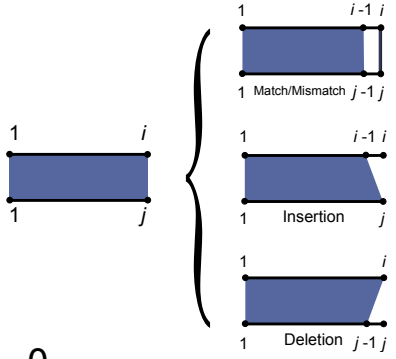
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

C	A	C	-	A
C	A	C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

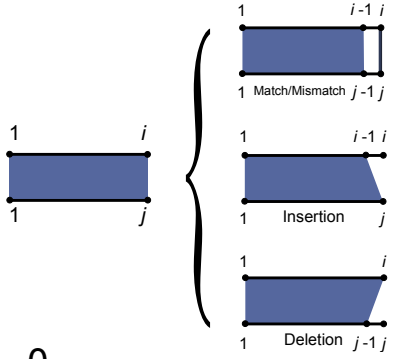
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

A	C	A	C	-	A
A	C	A	C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

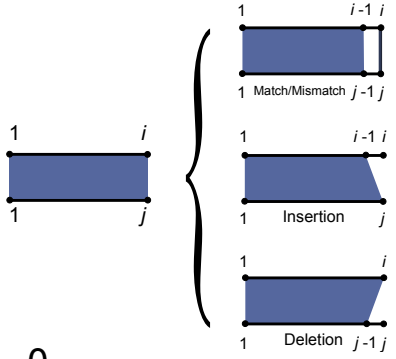
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

C	A	C	A	C	-	A
C	A	C	A	C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

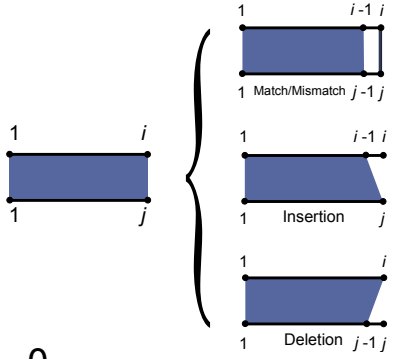
$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

G	C	A	C	A	C	-	A
-	C	A	C	A	C	T	A

Exemple : Alignement local de séquences `AGCACACA` et `ACACACTA`
Coûts : Match $m_{i,j} = +2$, Insertion/Déletion $p_i = p_j = -1$



$$W^\star(i,0) = 0$$
$$W^\star(0,j) = 0$$

$$W^\star(i,j) = \max \begin{cases} W^\star(i-1,j-1) + m_{s_i,t_j} \\ W^\star(i-1,j) + p_i \\ W^\star(i,j-1) + p_d \end{cases}$$

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

Meilleur alignement

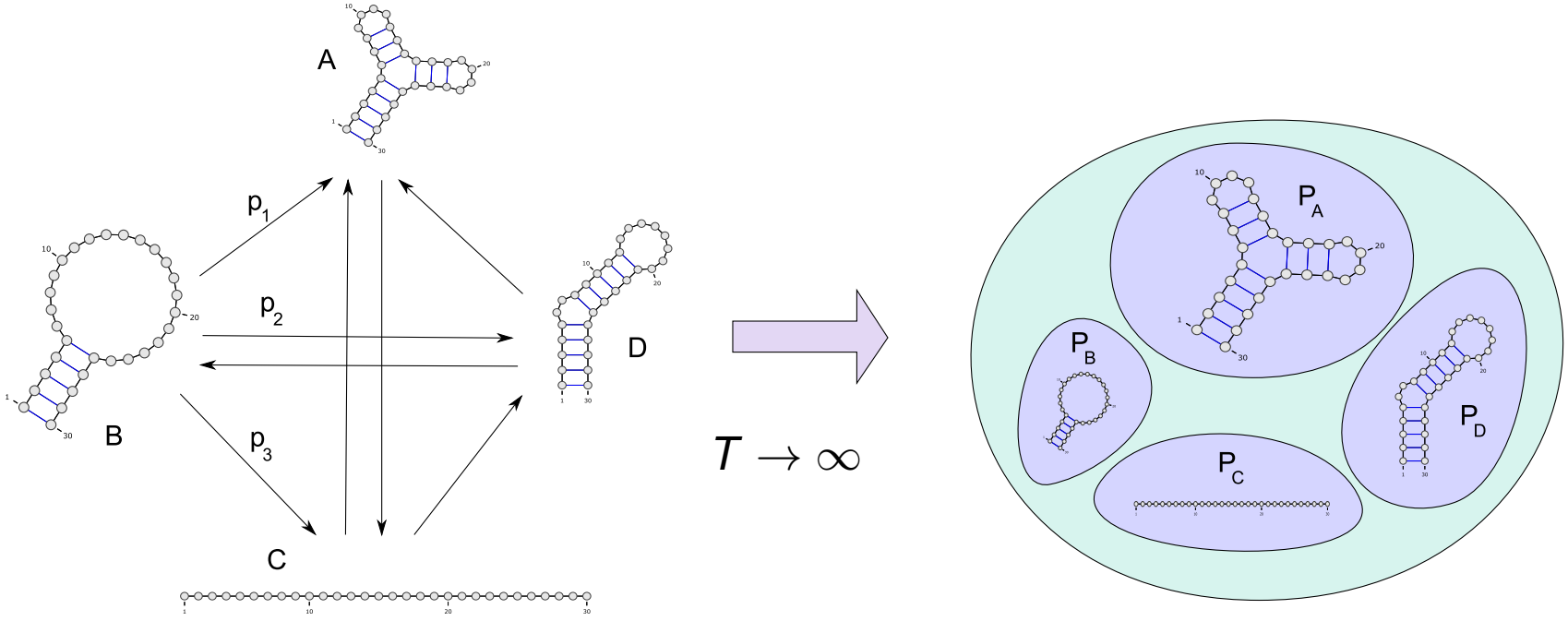
A	G	C	A	C	A	C	-	A
A	-	C	A	C	A	C	T	A

Résumé

1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique

Retour à l'ARN

A l'échelle nanoscopique, la structure de l'ARN *fluctue* (chaîne de Markov).



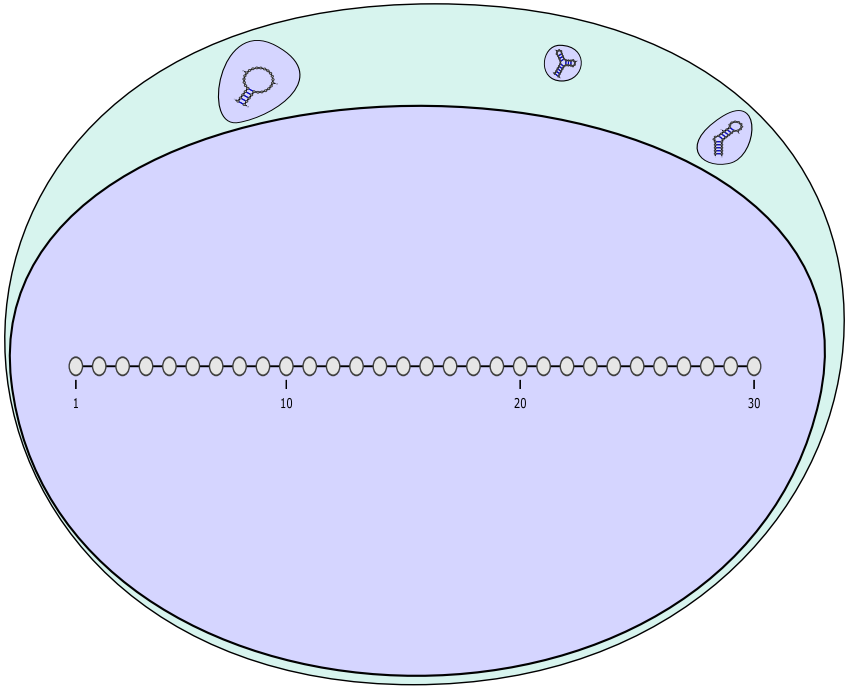
Convergence vers une distribution stationnaire de probabilité, l'équilibre de Boltzmann, où la probabilité est exponentiellement faible sur l'énergie libre.
Corollaire : La conformation initiale est sans d'importance.

Problèmes algorithmiques

Étant donné modèles pour l'ensemble des conformations et l'énergie libre.

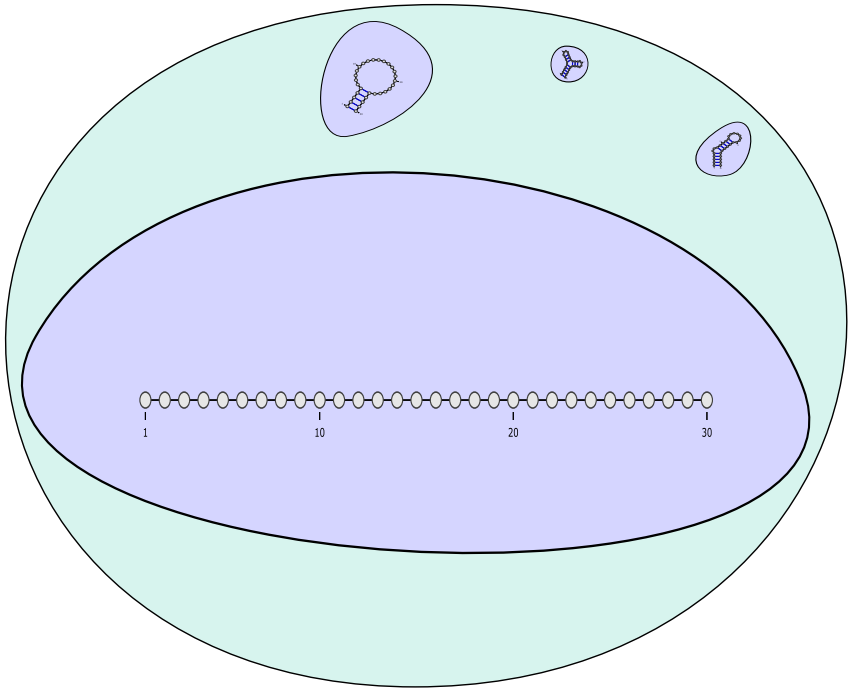
A. Déterminer la structure la plus probable (ou stable) à l'équilibre
B. Déterminer des propriétés moyennes de l'ensemble de Boltzmann

Transcription : ARN synthétisé sans appariement (Sauf exception)



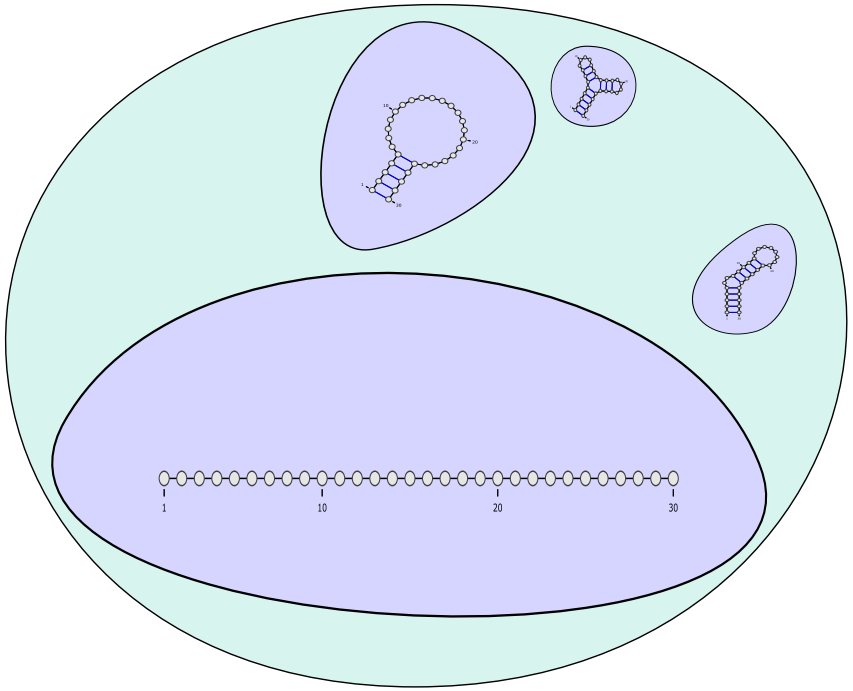
$T = 0$

Transcription : ARN synthétisé sans appariement (Sauf exception)



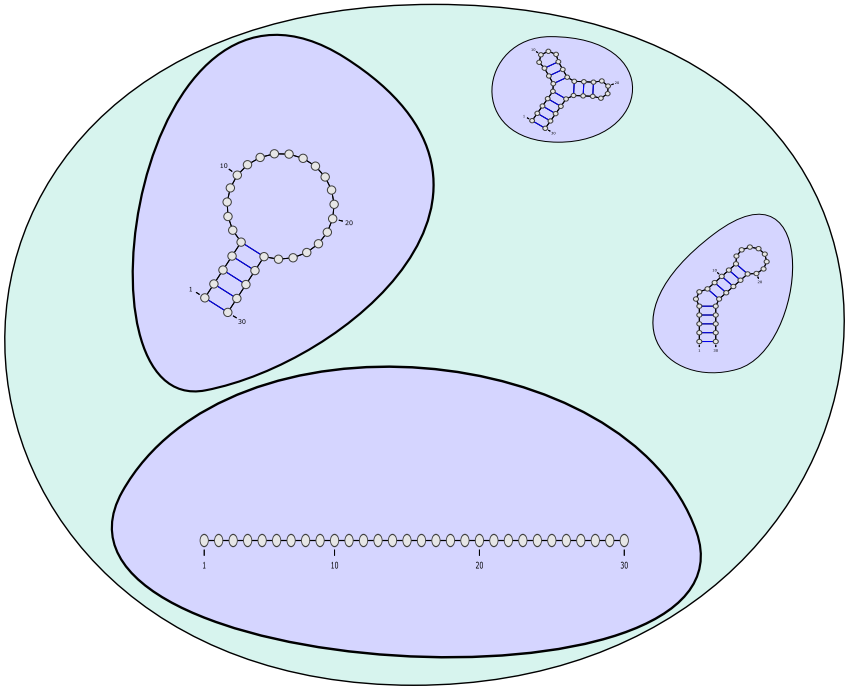
$T = 1h$

Transcription : ARN synthétisé sans appariement (Sauf exception)



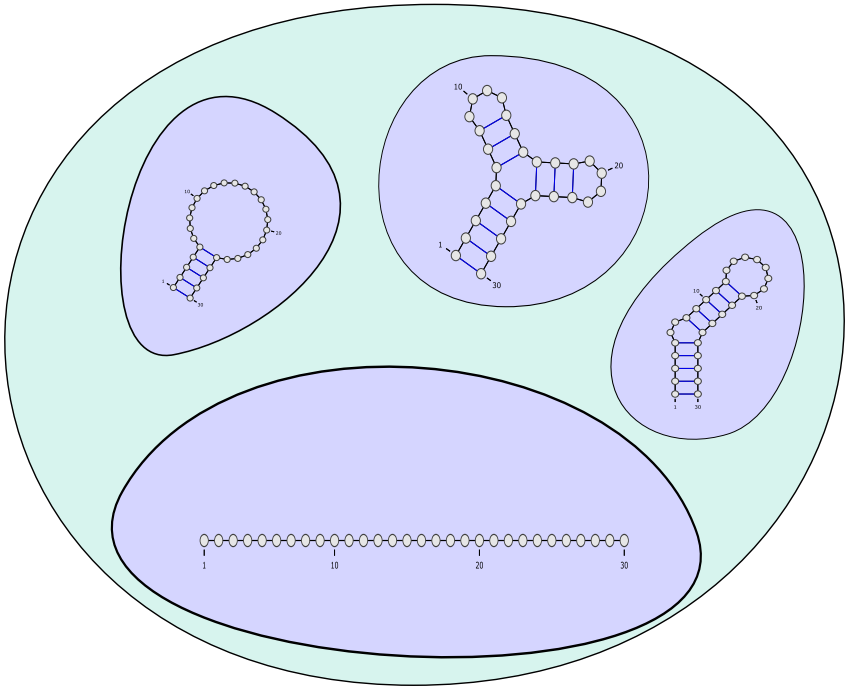
$$T = 2h$$

Transcription : ARN synthétisé sans appariement (Sauf exception)



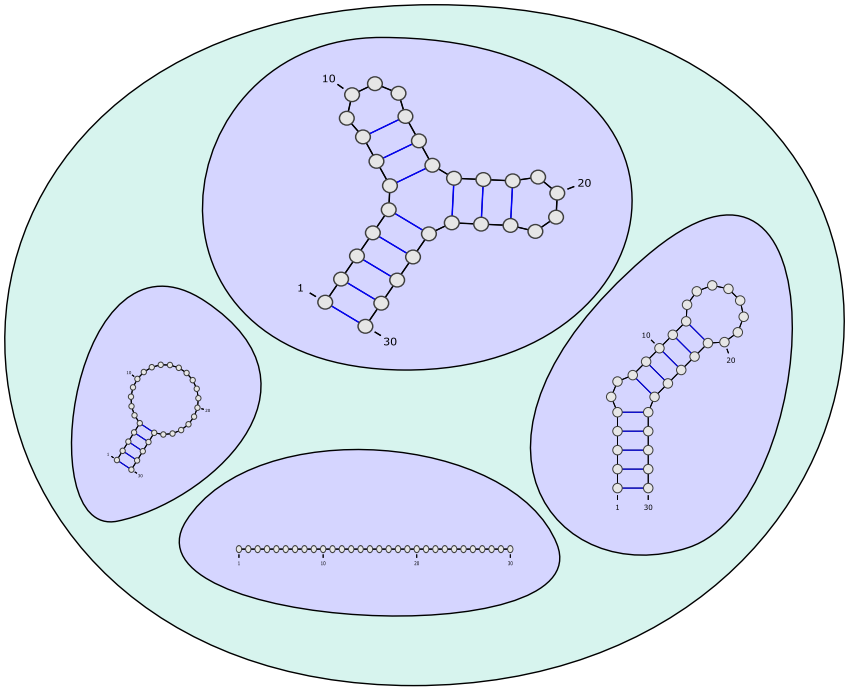
$$T = 5h$$

Transcription : ARN synthétisé sans appariement (Sauf exception)



$T = 10h$

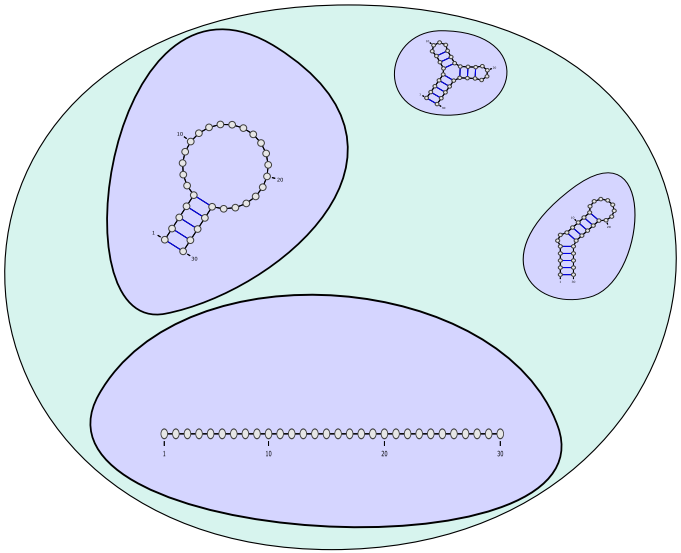
Transcription : ARN synthétisé sans appariement (Sauf exception)



$$T \to \infty$$

Mais majorité des ARNm dégradés avant 7h (Org. : Souris [SSN+09]).

Transcription : ARN synthétisé sans appariement (Sauf exception)



$T = 10h$

Mais majorité des ARNm dégradés avant 7h (Org. : Souris [SSN+09]).

Problèmes algorithmiques

A. Déterminer struct. la plus probable (= Energie libre min.) à l'équilibre
B. Déterminer des propriétés moyennes de l'ensemble de Boltzmann
C. Déterminer la structure la plus probable à temps T.
(c.f. H. Isambert par simulation, NP-complet en déterministe [MTSC11])

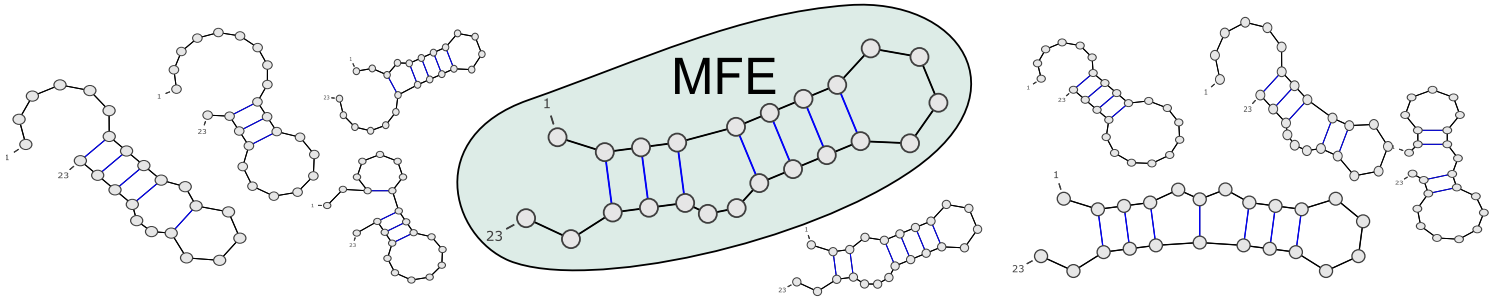
Résumé

1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique

Problème A : Déterminer la structure d'énergie minimale.

Repliement ab initio =

Trouver structure d'un ARN ω uniquement à partir de sa séquence.



- **Conformations** : Ensemble S_ω des structures secondaires compatibles avec la structure primaire ω (contrainte d'appariements).
- **Fonction d'énergie** Énergie libre associant une valeur numérique $E_{\omega,S}$ (KCal.mol^{-1}) à tout couple séquence/conformation (ω, S).
- **Structure native** : Conformation *fonctionnelle* de la molécule.

Modèle de Nussinov/Jacobson [NJ80]

Plus proche voisins simple :

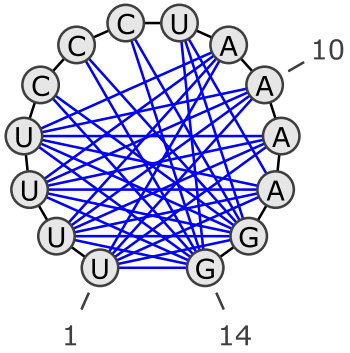
- Seuls les appariements contribuent à l'énergie
- Uniquement liaisons Watson/Crick (A/U,C/G) et Wobble (G/U)

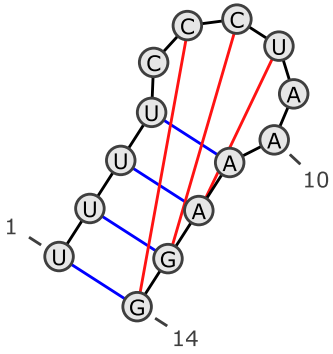
$$\Rightarrow E_{\omega,S} = -\#Paires(S)$$

Minimisation de l'énergie $\Leftrightarrow$ Maximisation du nombre de paires de bases.

Exemple :

UUUUCCCUAAAAGG

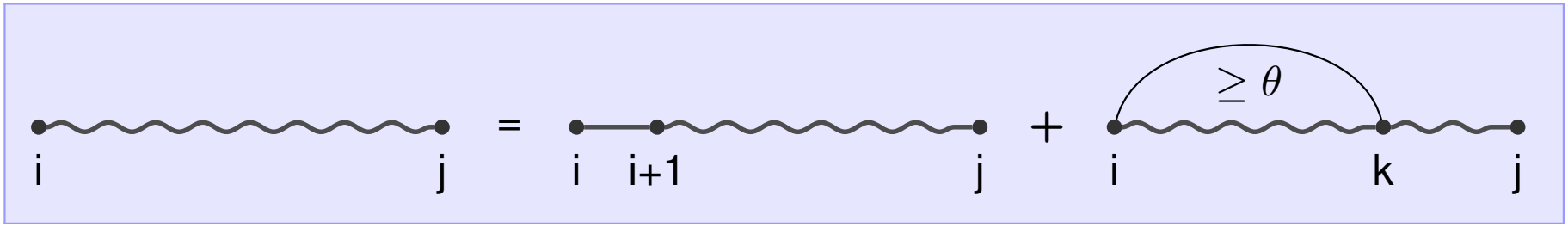




Variante : Pondérer les paires selon leur nombre de liaisons hydrogène

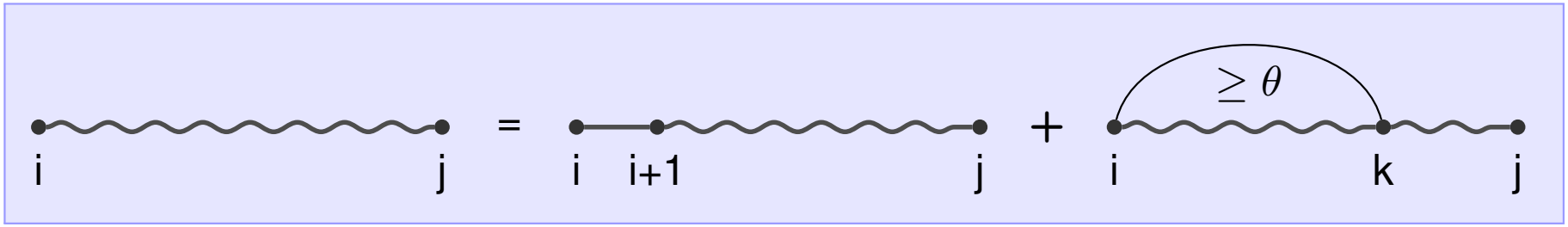
$\Delta G(G\equiv C) = -3$ $\Delta G(A=U) = -2$ $\Delta G(G-U) = -1$

Décomposition de Nussinov/Jacobson



$$
\begin{aligned}
N_{i,t} &= 0, \quad \forall t \in [i, i+\theta] \\
N_{i,j} &= \min \begin{cases} N_{i+1,j} & i \text{ non apparié} \\ \min\limits_{k=i+\theta+1}^{j} E_{i,k} + N_{i+1,k-1} + N_{k+1,j} & i \text{ apparié à } k \end{cases}
\end{aligned}
$$

où $E_{i,k}$ = Énergie d'appariement des positions i et k ($1/+\infty$ ou $\Delta G(s_i \equiv s_k)$)



$$
\begin{aligned}
N_{i,t} &= 0, \quad \forall t \in [i, i+\theta] \\
N_{i,j} &= \min \begin{cases} N_{i+1,j} & i \text{ non apparié} \\ \min\limits_{k=i+\theta+1}^{j} E_{i,k} + N_{i+1,k-1} + N_{k+1,j} & i \text{ apparié à } k \end{cases}
\end{aligned}
$$

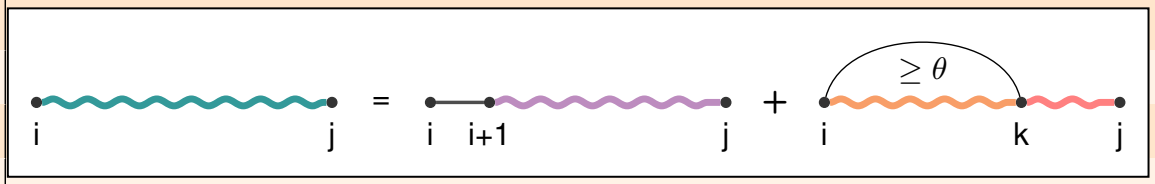
où $E_{i,k}$ = Énergie d'appariement des positions i et k ($1/+\infty$ ou $\Delta G(s_i \equiv s_k)$)

Exercice

- Correction ?
- Complétude ?
- Non-ambiguité ?

Nussinov/Jacobson

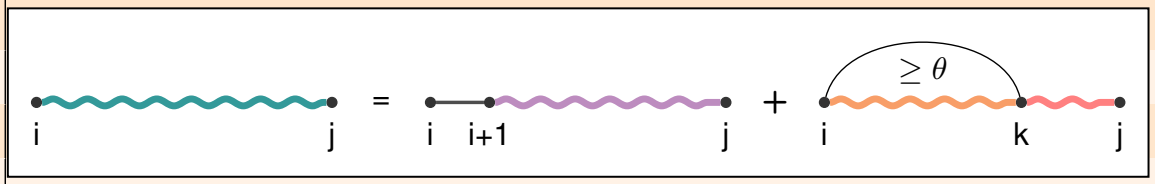
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

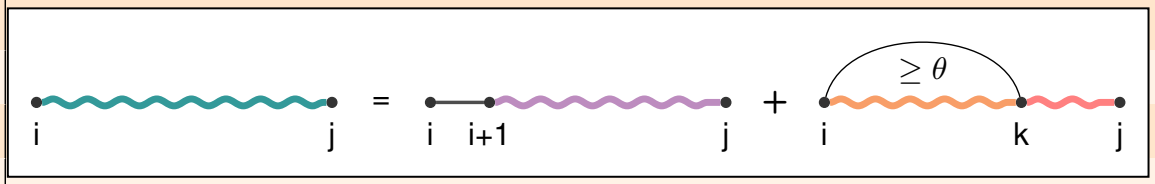
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

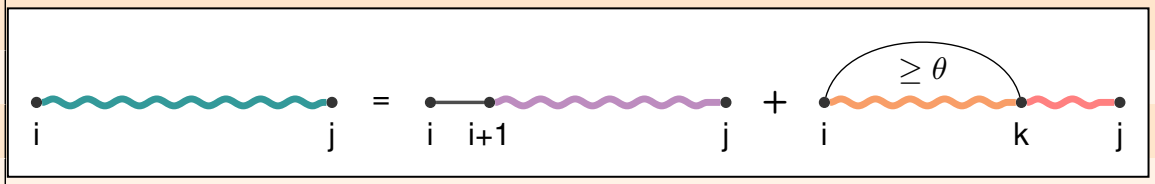
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

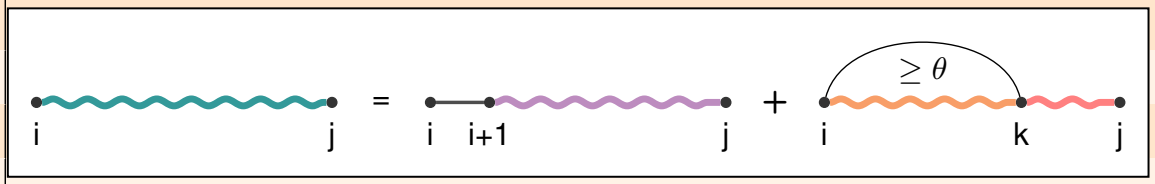
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

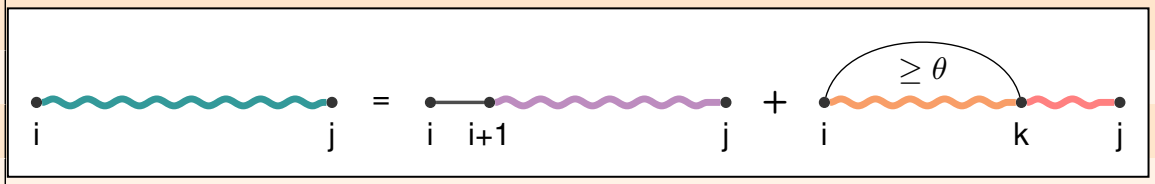
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

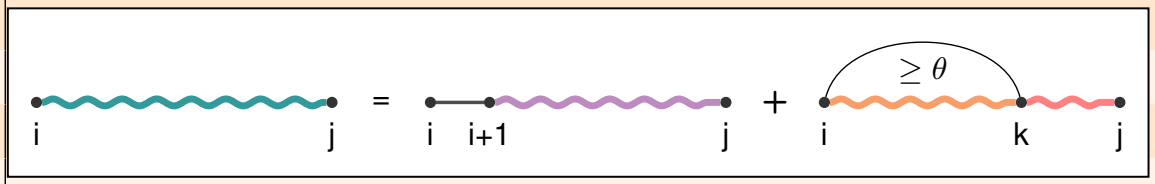
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

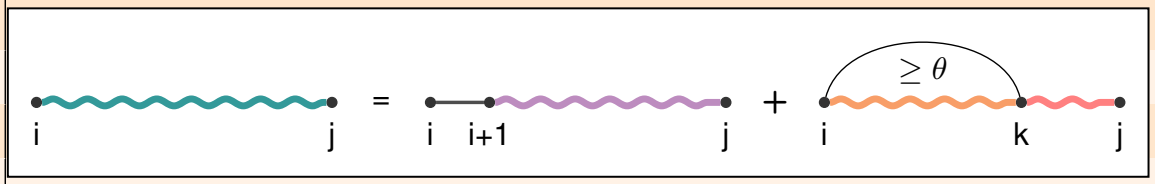
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

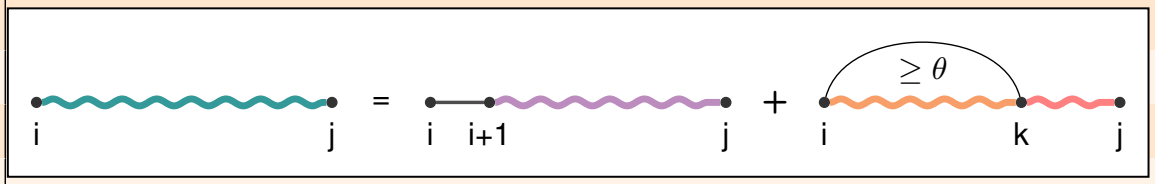
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

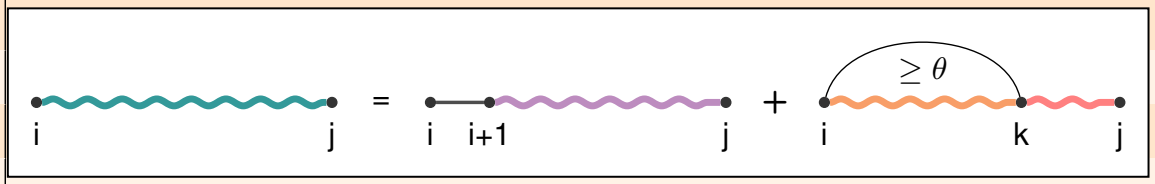
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

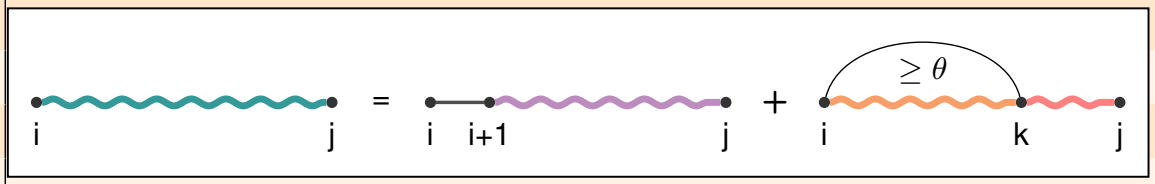
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

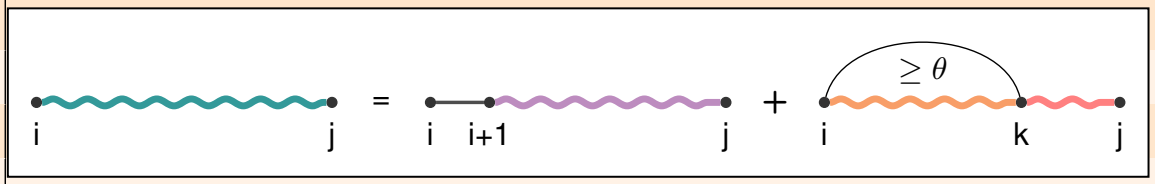
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

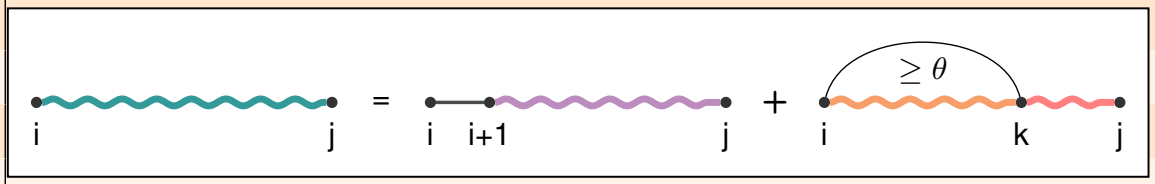
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

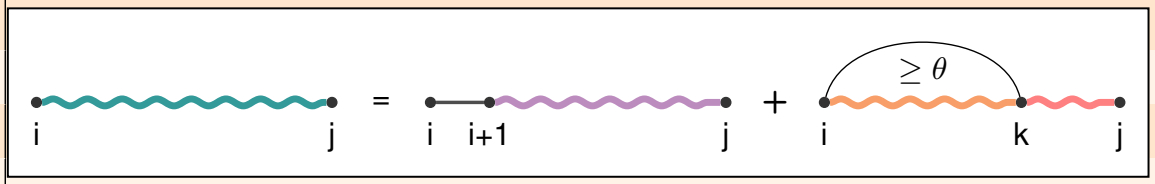
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

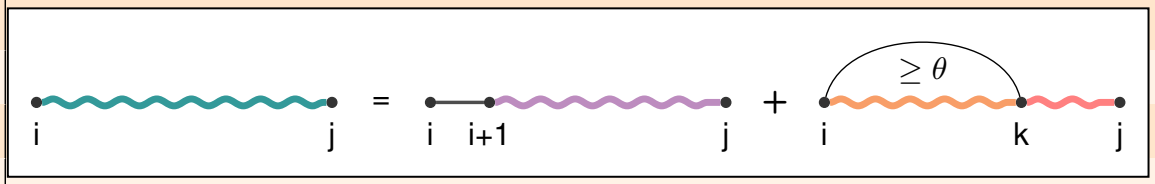
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	.	.	.	.	.	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

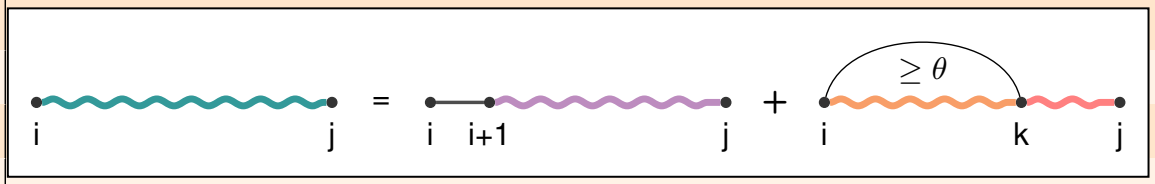
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	.	.	.	.	.	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

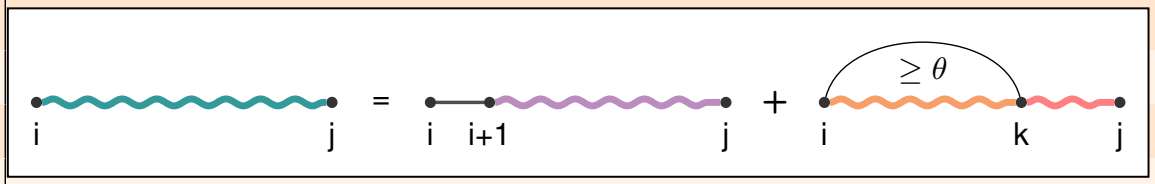
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	.	.	.	.	.	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

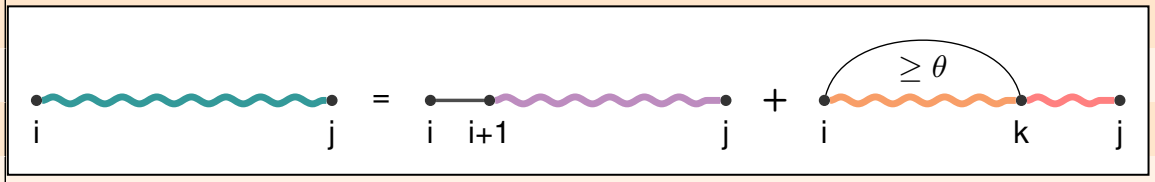
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

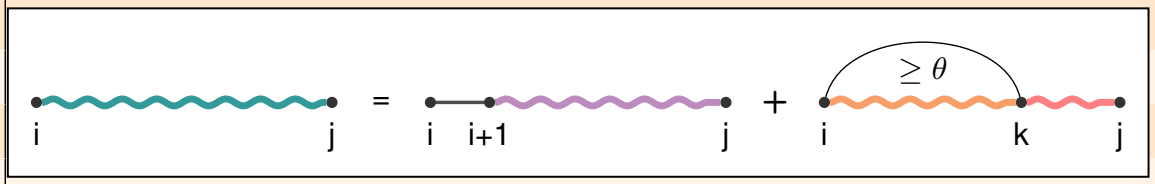
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

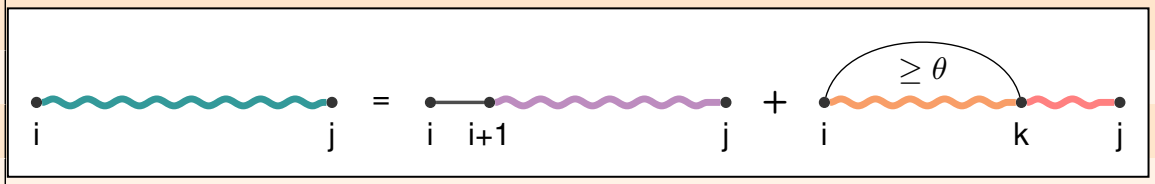
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

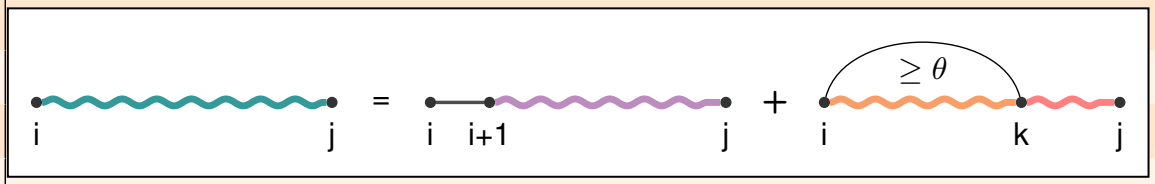
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

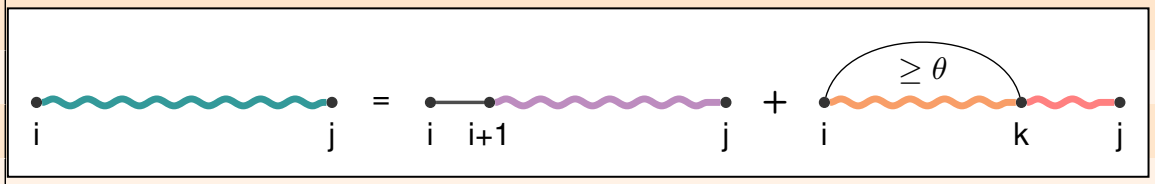
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k}$ = 1 (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

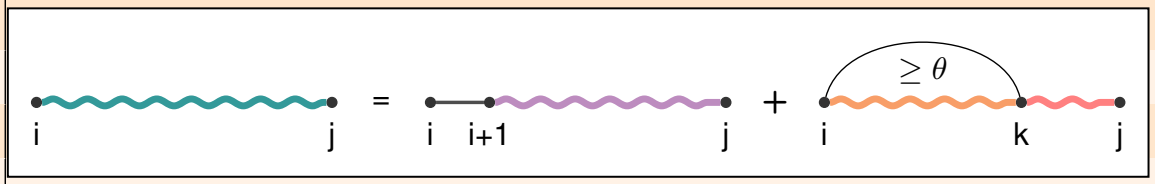
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k}$ = 1 (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

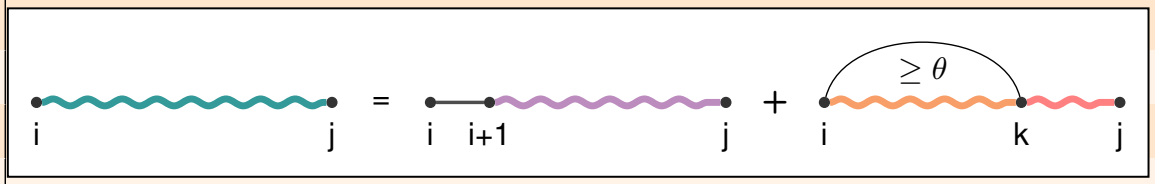
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	.	.	.	.	.	.	.	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

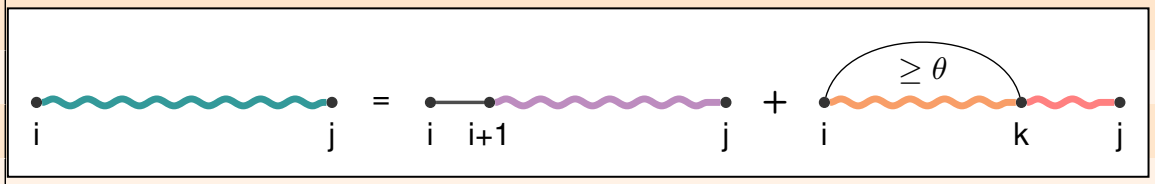
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	(	.	.	.	.	.	)	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

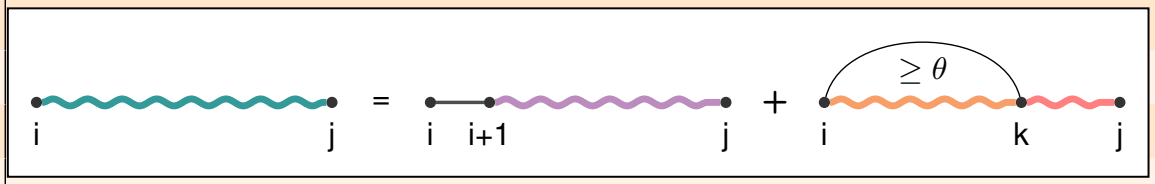
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	(	.	.	.	.	.	)	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

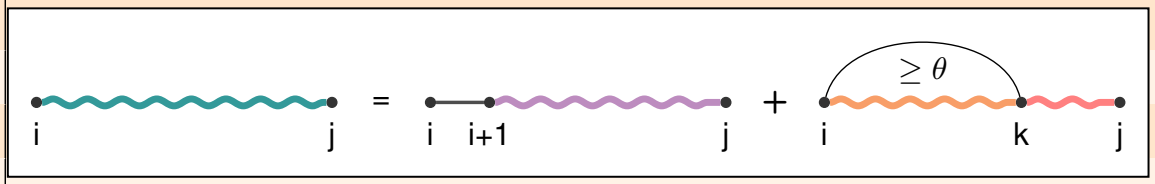
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	(	.	.	.	.	.	)	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k}$ = 1 (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

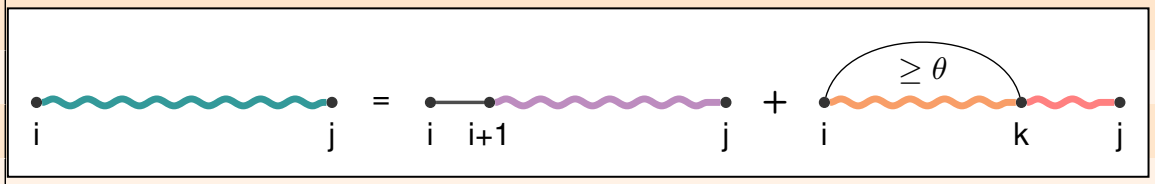
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	(	(	.	.	.	)	)	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Nussinov/Jacobson

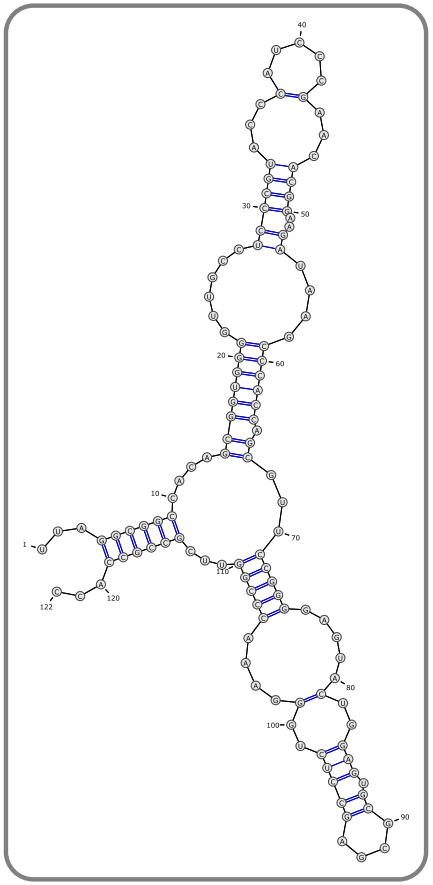
	C	G	G	A	U	A	C	U	U	C	U	U	A	G	A	C	G	A
	(	(	(	.	.	.	)	.	(	(	.	.	.	)	)	)	)	.
C	0	0	0	0	0	0	3	4	4	6	6	6	6	9	9	11	14	14
G		0	0	0	0	0	3	4	4	6	6	6	6	7	9	11	11	11
G			0	0	0	0	3	3	3	5	5	5	5	6	8	10	10	10
A				0	0	0	0	2	2	2	2	4	4	5	7	7	8	10
U					0	0	0	0	0	0	2	2	4	5	7	7	8	10
A						0	0	0	0	0	2	2	2	5	5	5	8	8
C							0	0	0	0	0	0	2	5	5	5	8	8
U								0	0	0	0	0	2	3	5	5	6	7
U									0	0	0	0	2	3	5	5	5	7
C										0	0	0	0	3	3	3	5	5
U											0	0	0	0	2	2	2	3
U												0	0	0	0	0	1	2
A													0	0	0	0	0	0
G														0	0	0	0	0
A															0	0	0	0
C																0	0	0
G																	0	0
A																		0



$E_{i,k} = 1$ (G−U), 2 (A=U) ou 3 (G≡C)

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :



Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

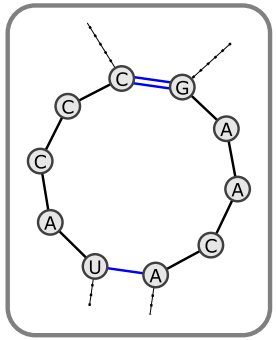
Déterminées expérimentalement
+ Interpolation pour grandes boucles.

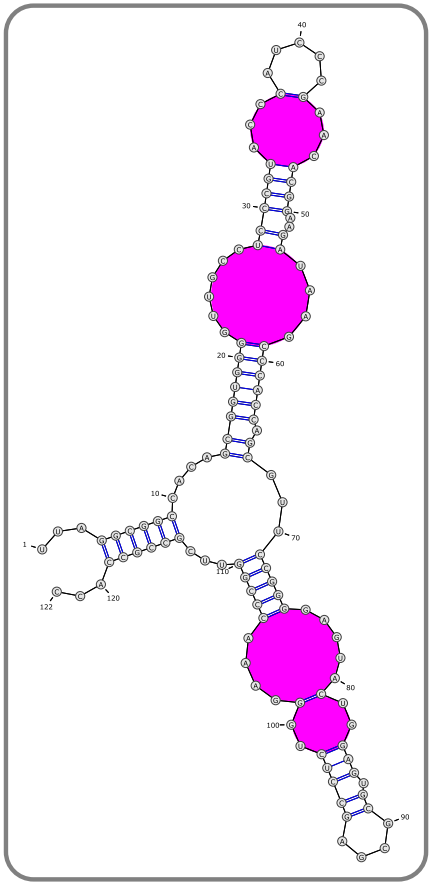
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :

- Boucles internes





Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

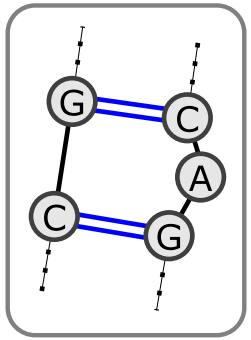
Déterminées expérimentalement
+ Interpolation pour grandes boucles.

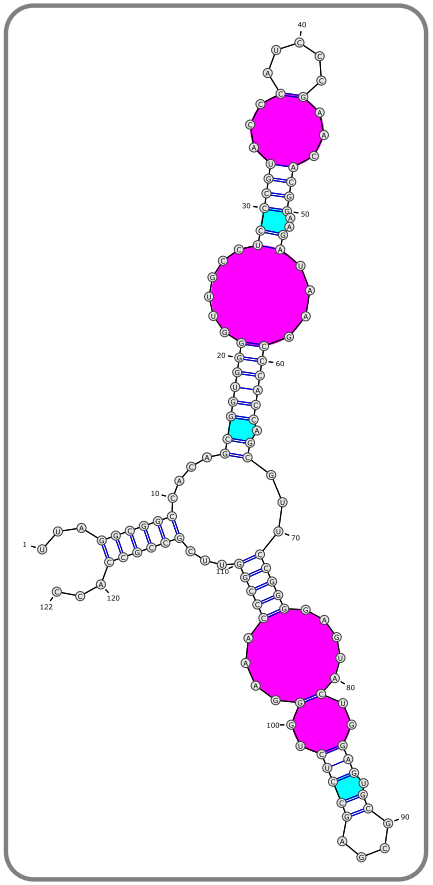
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :

- Boucles internes
- Renflements





Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

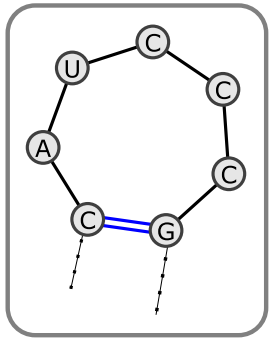
Déterminées expérimentalement
+ Interpolation pour grandes boucles.

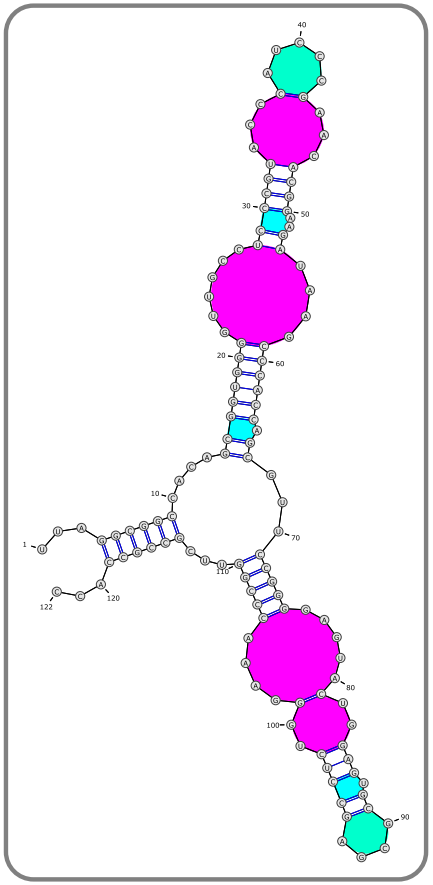
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :

- Boucles internes
- Renflements
- Boucles terminales





Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

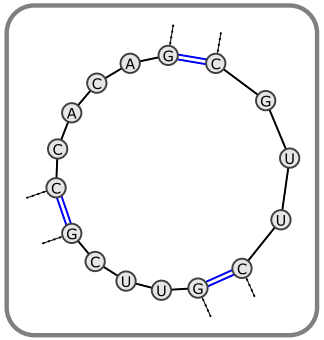
Déterminées expérimentalement
+ Interpolation pour grandes boucles.

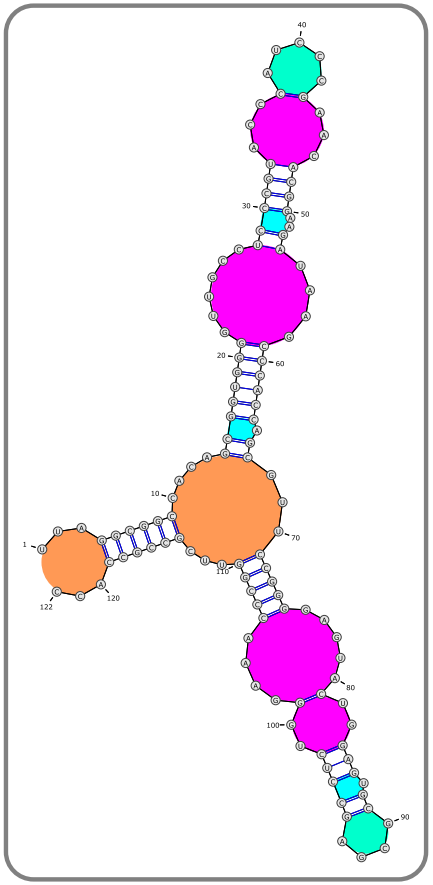
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :

- Boucles internes
- Renflements
- Boucles terminales
- Boucles multiples





Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

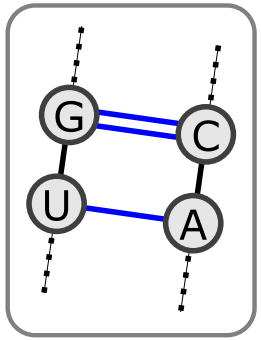
Déterminées expérimentalement
+ Interpolation pour grandes boucles.

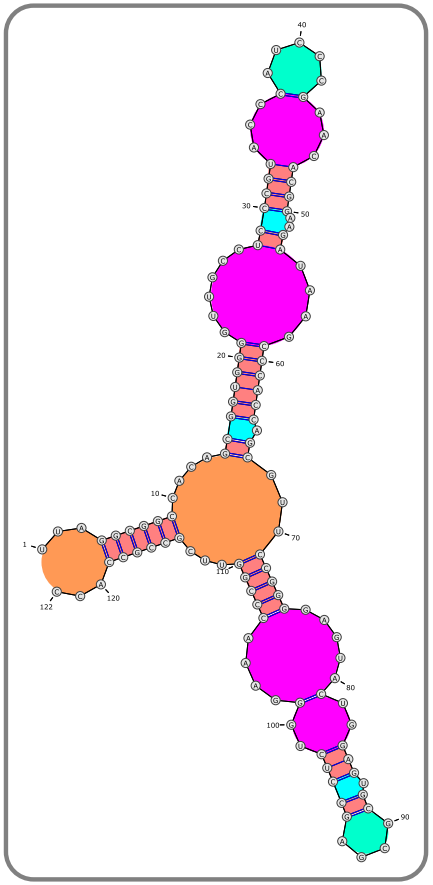
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner

Basée sur décomposition non-ambiguë en boucles de la structure 2^{aire} :

- Boucles internes
- Renflements
- Boucles terminales
- Boucles multiples
- Empilements



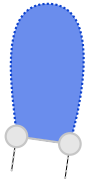


Énergies libres ΔG des boucles dépendent des bases, asymétrie, bases *libres* (dangle) . . .

Déterminées expérimentalement
+ Interpolation pour grandes boucles.

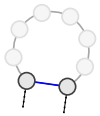
Meilleures prédictions grâce à la prise en compte des empilements.

Modèle de Turner : Décomposition

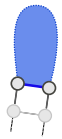


Tiges
boucles

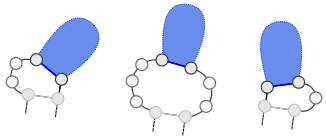
$=$



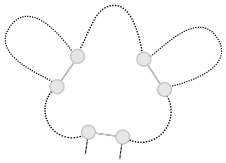
Boucle terminale



Empilement



Renflement/Boucle interne

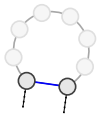


Boucle multiple
(Séq. $\geq$2 hélices)

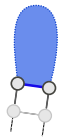
Modèle de Turner : Décomposition

Tiges boucles

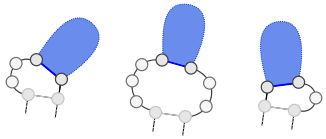
$=$



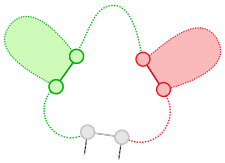
Boucle terminale



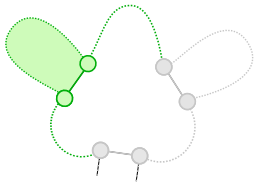
Empilement



Renflement/Boucle interne

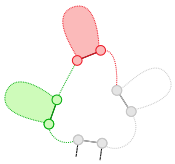


Boucle multiple
(Séq. $\geq$2 hélices)



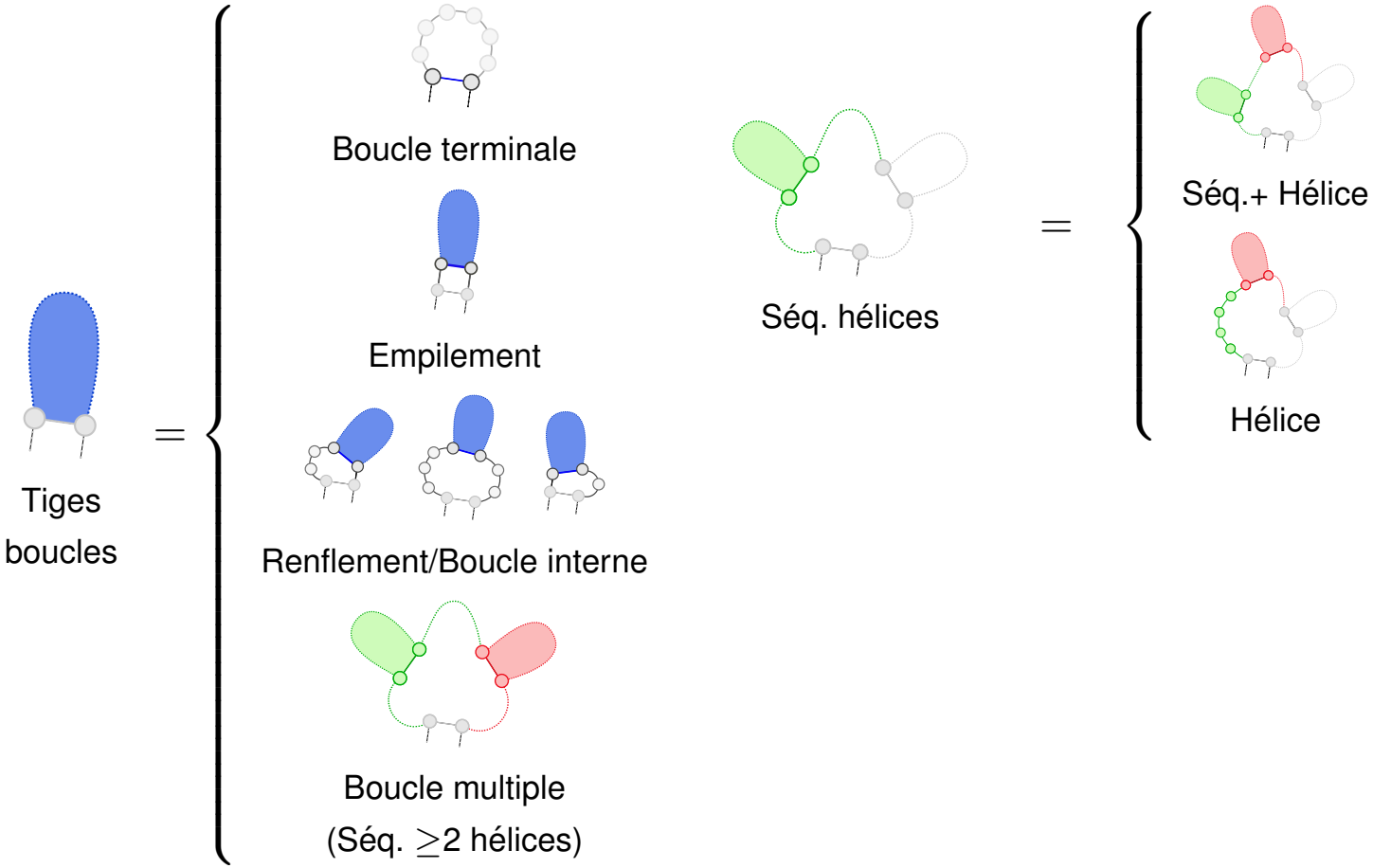
Séq. hélices

$=$

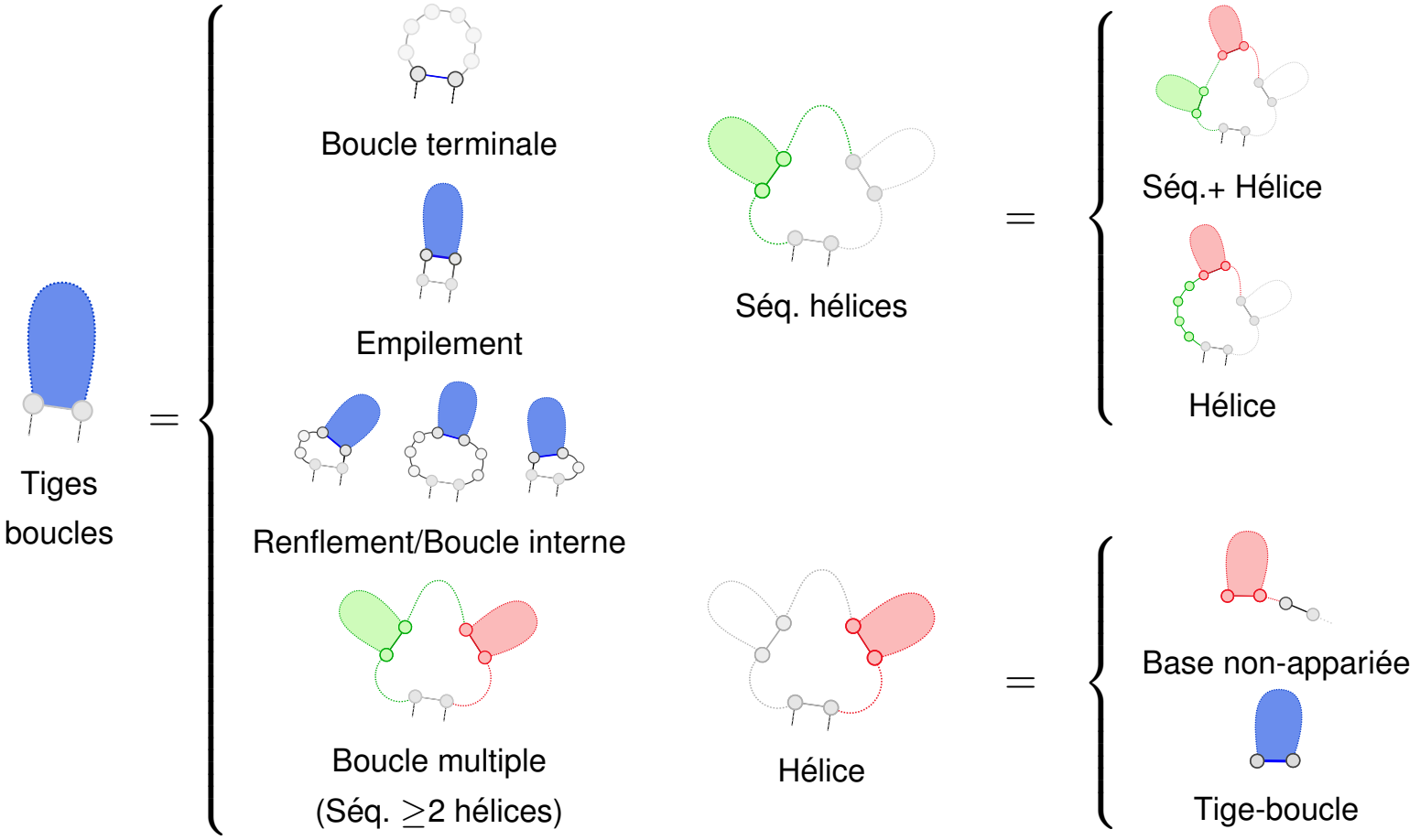


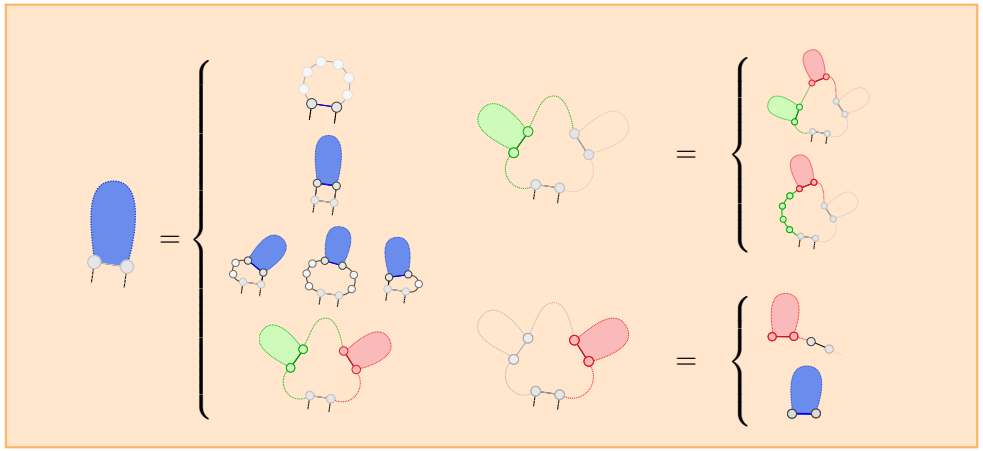
Séq.+ Hélice

Modèle de Turner : Décomposition

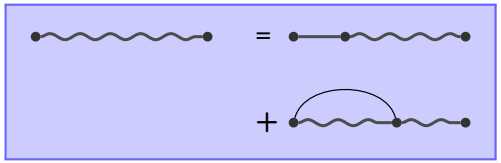


Modèle de Turner : Décomposition





$\overset{??}{\Longleftrightarrow}$

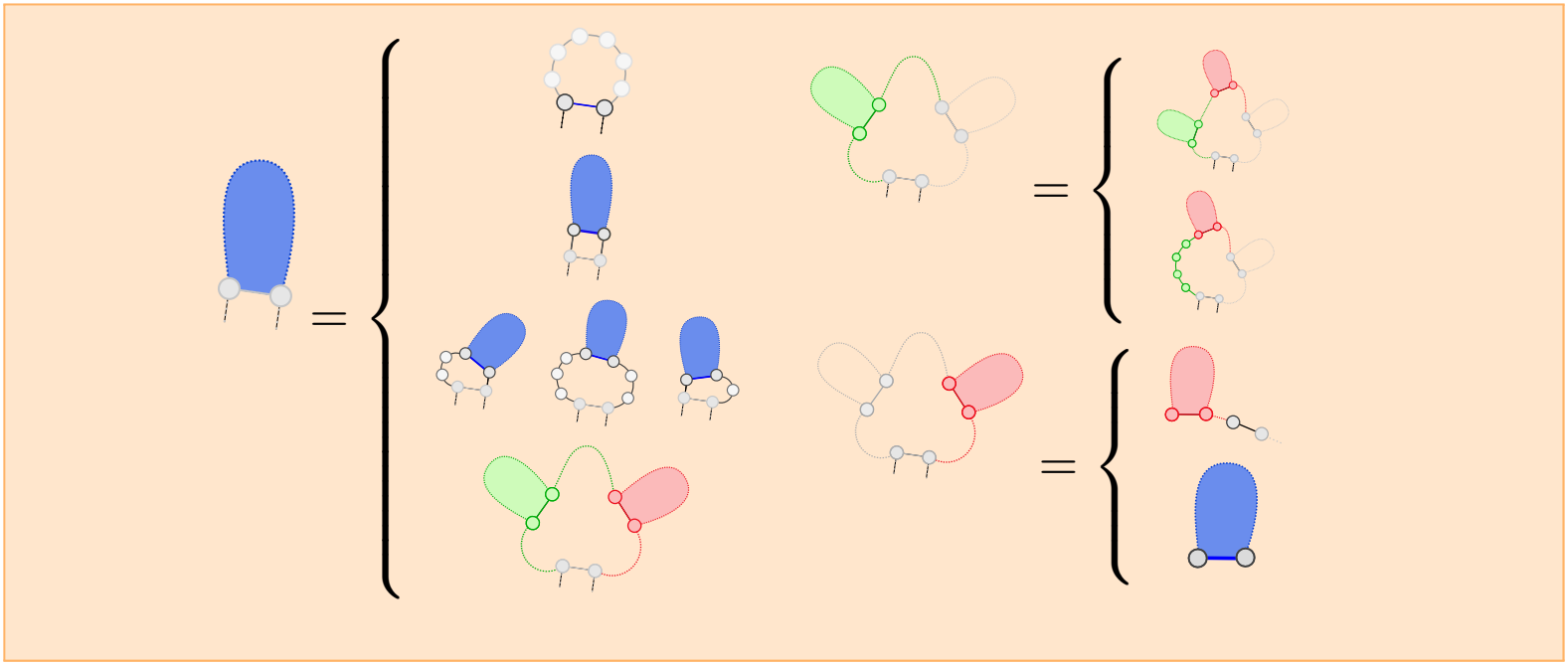


Décomposition non-ambiguë $\rightarrow$ Ok !

Complète ? Comparons les séries génératrices ...

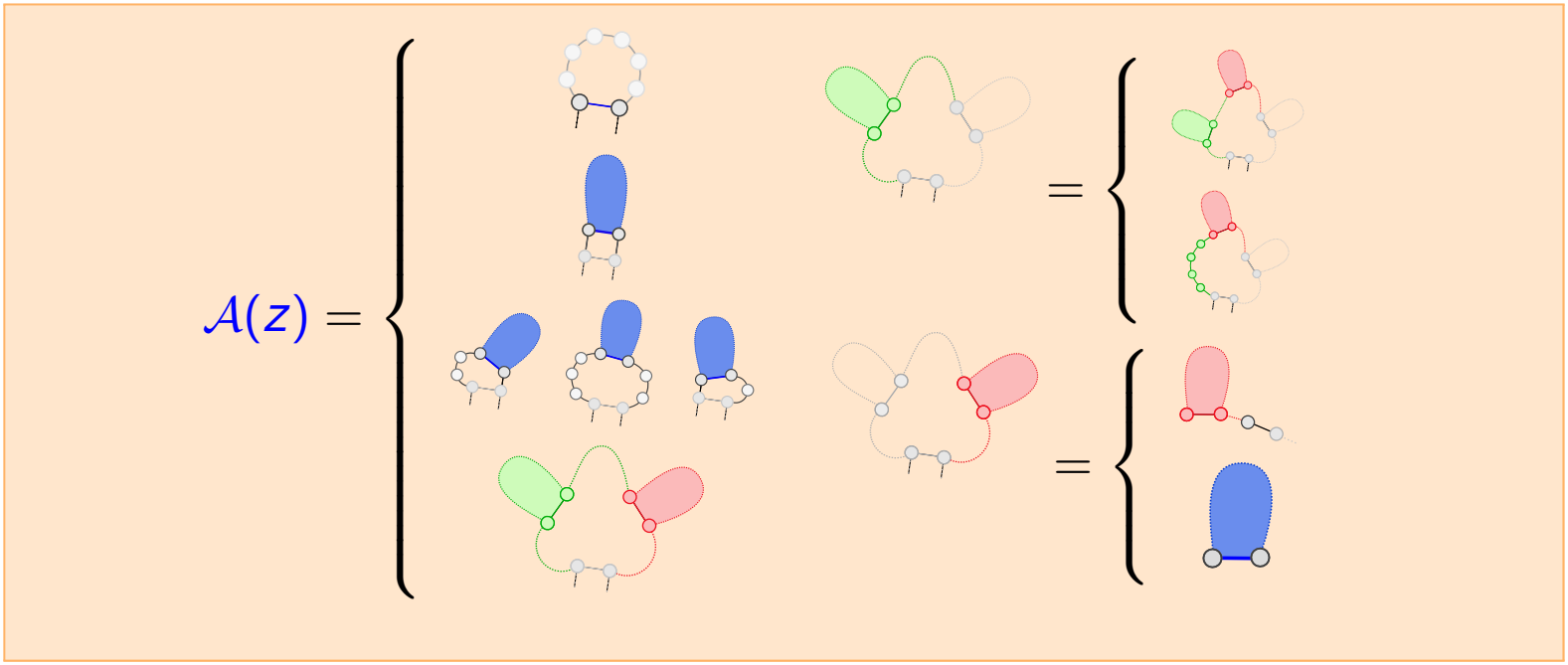
Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$



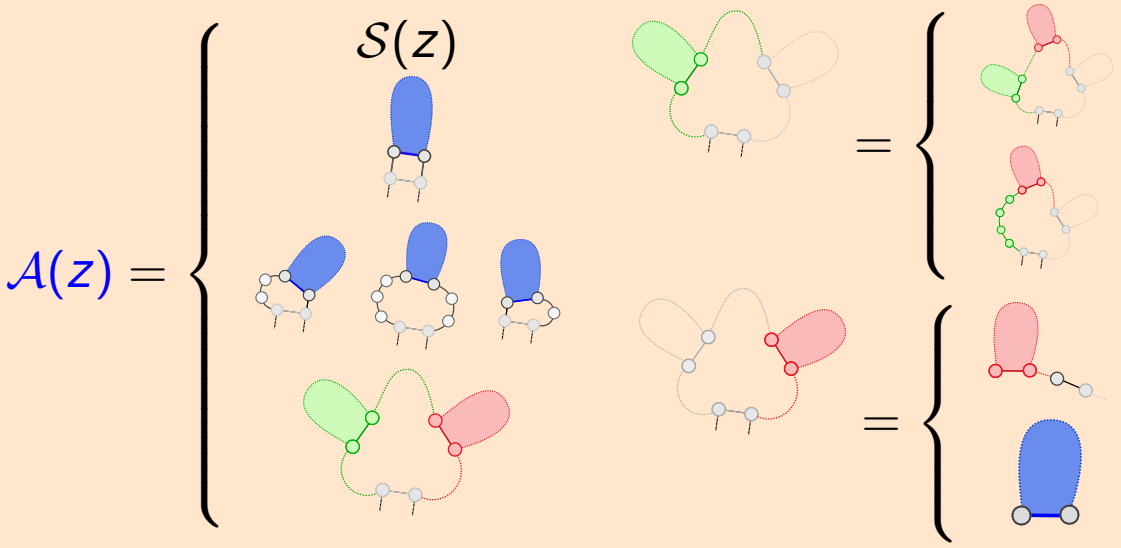
Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$



Rappel : Série génératrice des structures secondaires [Wat78]

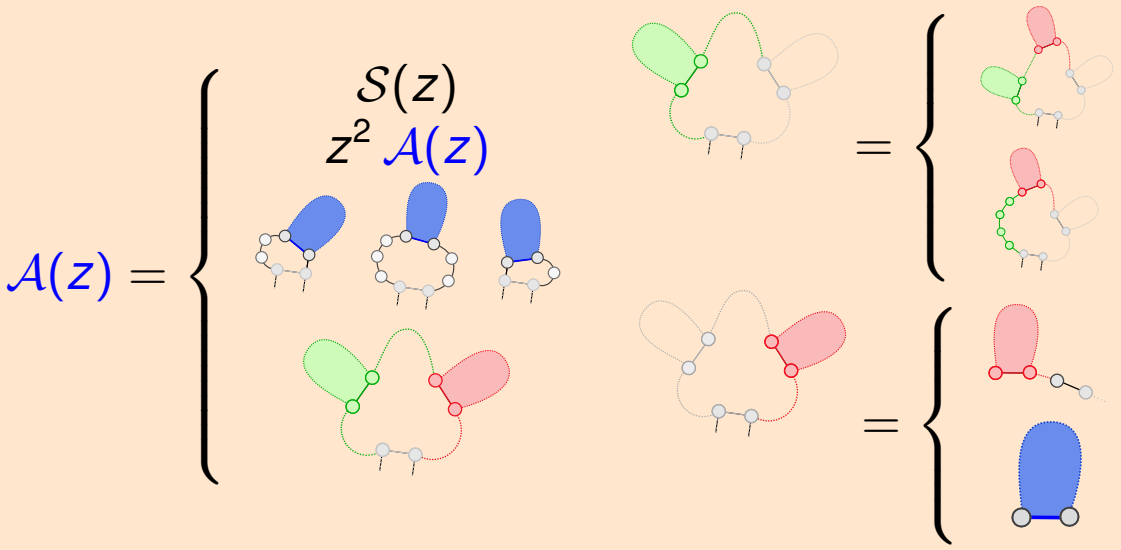
$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$



$$\mathcal{S}(z) = 1 + z\,\mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

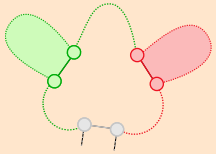


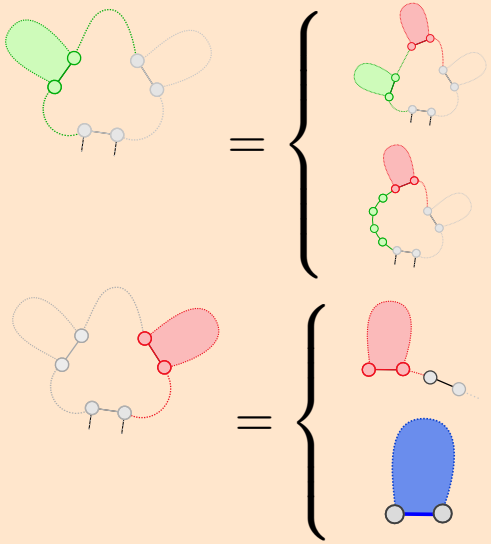
$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \end{cases}$$



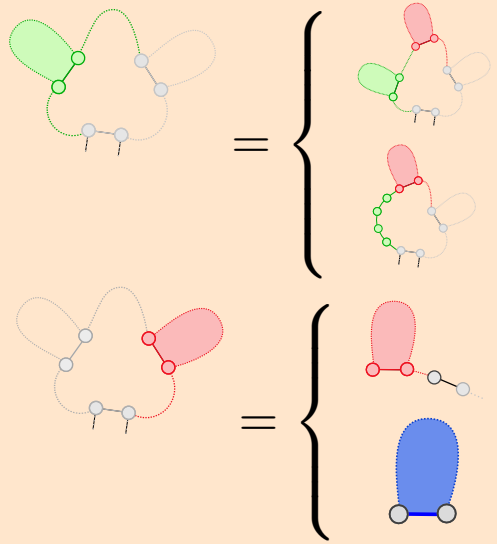


$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ \quad + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$



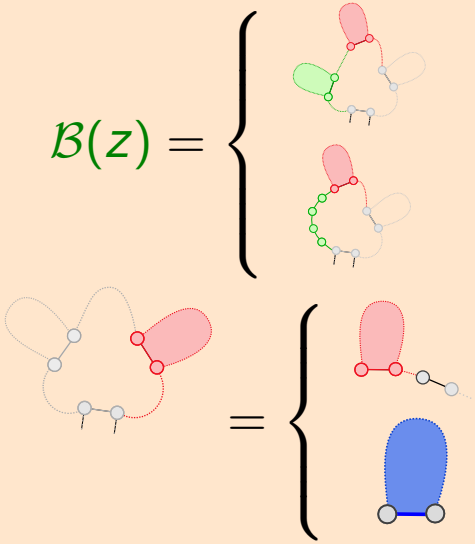
$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$

$\mathcal{B}(z) =$



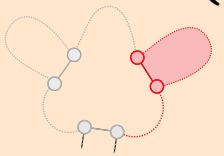
$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

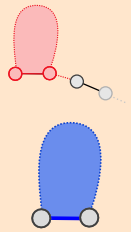
$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$

$\mathcal{B}(z) = \{ \mathcal{B}(z) \, \mathcal{C}(z)$



$= \{$



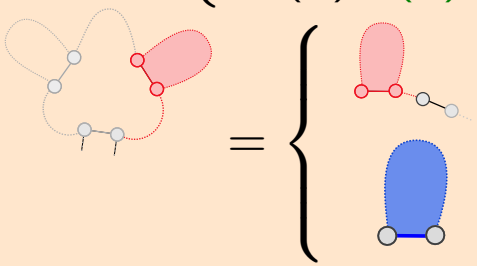
$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$

$$\mathcal{B}(z) = \begin{cases} \mathcal{B}(z) \, \mathcal{C}(z) \\ \mathcal{S}(z) \, \mathcal{B}(z) \end{cases}$$



$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

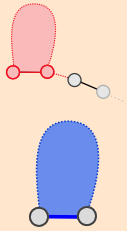
Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$

$$\mathcal{B}(z) = \begin{cases} \mathcal{B}(z) \, \mathcal{C}(z) \\ \mathcal{S}(z) \, \mathcal{B}(z) \end{cases}$$

$\mathcal{C}(z) =$



$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

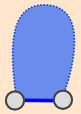
Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n \geq 0} s_n \, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2 \, \mathcal{A}(z) \\ z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) + z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) \, z \\ + z \, \mathcal{S}(z) \, z^2 \, \mathcal{A}(z) \, \mathcal{S}(z) z \\ \mathcal{B}(z) \, \mathcal{C}(z) \end{cases}$$

$$\mathcal{B}(z) = \begin{cases} \mathcal{B}(z) \, \mathcal{C}(z) \\ \mathcal{S}(z) \, \mathcal{B}(z) \end{cases}$$

$$\mathcal{C}(z) = \begin{cases} \mathcal{C}(z) \, z \\ \end{cases}$$



$$\mathcal{S}(z) = 1 + z \, \mathcal{S}(z)$$

Rappel : Série génératrice des structures secondaires [Wat78]

$$\mathcal{W}(z) := \sum_{n\geq 0} s_n\, z^n = \frac{1 - z + z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2}$$

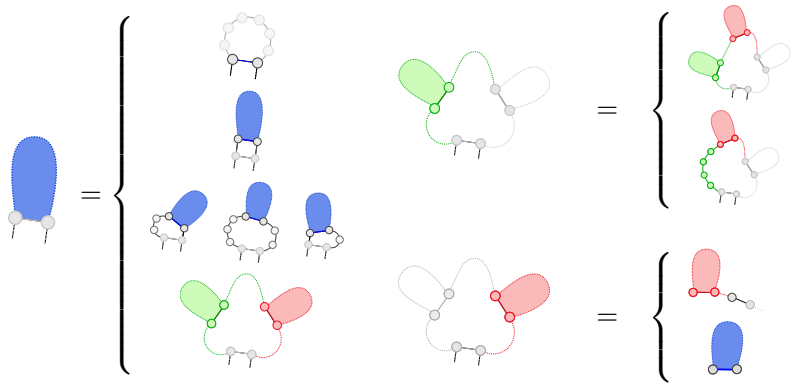
$$\mathcal{A}(z) = \begin{cases} \mathcal{S}(z) \\ z^2\,\mathcal{A}(z) \\ z\,\mathcal{S}(z)\,z^2\,\mathcal{A}(z) + z^2\,\mathcal{A}(z)\,\mathcal{S}(z)\,z \\ +z\,\mathcal{S}(z)\,z^2\,\mathcal{A}(z)\,\mathcal{S}(z)z \\ \mathcal{B}(z)\,\mathcal{C}(z) \end{cases} \qquad \mathcal{B}(z) = \begin{cases} \mathcal{B}(z)\,\mathcal{C}(z) \\ \mathcal{S}(z)\,\mathcal{B}(z) \end{cases} \qquad \mathcal{C}(z) = \begin{cases} \mathcal{C}(z)\,z \\ z^2\,\mathcal{A}(z) \end{cases}$$

$$\mathcal{S}(z) = 1 + z\,\mathcal{S}(z)$$

$$\begin{aligned} \mathcal{A}(z) &= \frac{1 - z - z^2 - \sqrt{1 - 2z - z^2 - 2z^3 + z^4}}{2z^2} \\ &= \mathcal{W}(z) - 1 \quad \text{(Oubli de la str. sec. de longueur 0)} \end{aligned}$$

$\Rightarrow$ Schéma complet, correct par construction $\Rightarrow$ Minimisation possible.

- $E_H(i,j)$: Energie de boucle terminale *fermée* par une paire (i,j)
- $E_{BI}(i,j)$: Energie de renflement ou boucle interne *fermée* par une paire (i,j)
- $E_S(i,j)$: Energie d'empilement $(i,j)/(i+1,j-1)$
- a,c,b : Pénalité de boucle multiple, hélice et non-appariées dans multiboucle.



Calcul des matrices

$$\mathcal{M}'_{i,j} = \min \begin{cases} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \mathrm{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \mathrm{Min}_k(\mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}) \end{cases}$$

$$\mathcal{M}_{i,j} = \mathrm{Min}_k \left\{ \min\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1_{k,j} \right\}$$

$$\mathcal{M}^1_{i,j} = \mathrm{Min}_k \left\{ b + \mathcal{M}^1_{i,j-1}, c + \mathcal{M}'_{i,j} \right\}$$

Remontée (Backtracking)

Reconstruction de la structure d'énergie minimale :

$$
\begin{aligned}
\boxed{\mathcal{M}'_{i,j}} &= \text{Min}\left\{\begin{array}{c} \boxed{E_H(i,j)} \\ \boxed{E_S(i,j) + \mathcal{M}'_{i+1,j-1}} \\ \boxed{\text{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'})} \\ \boxed{a + c + \text{Min}_k(\ \mathcal{M}_{i+1,k-1} + \mathcal{M}^1{}_{k,j-1}\ )} \end{array}\right\} \\
\mathcal{M}_{i,j} &= \text{Min}_k\left\{\min\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1{}_{k,j}\right\} \\
\mathcal{M}^1{}_{i,j} &= \text{Min}_k\left\{b + \mathcal{M}^1{}_{i,j-1}, c + \mathcal{M}'_{i,j}\right\}
\end{aligned}
$$

1. Avec une astuce pour les bulges/boucles internes ...

Reconstruction de la structure d'énergie minimale :

$$\mathcal{M}'_{i,j} = \text{Min}\left\{\begin{array}{l} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \text{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \text{Min}_k(\ \mathcal{M}_{i+1,k-1} + \mathcal{M}^1{}_{k,j-1}\ ) \end{array}\right\}$$

$$\mathcal{M}_{i,j} = \text{Min}_k\left\{\min\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1{}_{k,j}\right\}$$

$$\mathcal{M}^1{}_{i,j} = \text{Min}_k\left\{b + \mathcal{M}^1{}_{i,j-1}, c + \mathcal{M}'_{i,j}\right\}$$

1. Avec une astuce pour les bulges/boucles internes ...

Reconstruction de la structure d'énergie minimale :

$$\mathcal{M}'_{i,j} = \text{Min}\left\{ \begin{array}{l} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \text{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \text{Min}_k(\ \mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}\ ) \end{array} \right\}$$

???

$$\mathcal{M}_{i,j} = \text{Min}_k \left\{ \min\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1_{k,j} \right\}$$

$$\mathcal{M}^1_{i,j} = \text{Min}_k \left\{ b + \mathcal{M}^1_{i,j-1}, c + \mathcal{M}'_{i,j} \right\}$$

$\Theta(n)$ contributeurs potentiels au Min :
$\Rightarrow$ Complexité au pire en $\mathcal{O}(n^2)$ pour un backtrack naif.

1. Avec une astuce pour les bulges/boucles internes ...

Remontée (Backtracking)

Reconstruction de la structure d'énergie minimale :

$$\mathcal{M}'_{i,j} = \text{Min}\left\{\begin{array}{c} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \text{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \text{Min}_k(\ \mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}\ ) \end{array}\right\}$$

$$\mathcal{M}_{i,j} = \text{Min}_k\left\{\min\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1_{k,j}\right\}$$

$$\mathcal{M}^1_{i,j} = \text{Min}_k\left\{b + \mathcal{M}^1_{i,j-1}, c + \mathcal{M}'_{i,j}\right\}$$

$\Theta(n)$ contributeurs potentiels au Min :
$\Rightarrow$ Complexité au pire en $\mathcal{O}(n^2)$ pour un backtrack naif.
Garder les meilleurs contributeurs aux Min $\Rightarrow$ Backtrack en $\mathcal{O}(n)$

Complexités temps/mémoire en $\mathcal{O}(n^3)/\mathcal{O}(n^2)$ pour le précalcul [1]

1. Avec une astuce pour les bulges/boucles internes ...

Reconstruction de la structure d'énergie minimale :

$$\mathcal{M}'_{i,j} = \text{Min}\left\{\begin{array}{c} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \text{Min}_{i',j'}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \text{Min}_k(\mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}) \end{array}\right\}$$

$$\mathcal{M}_{i,j} = \text{Min}_k\left\{\min(\mathcal{M}_{i,k-1}, b(k-1)) + \mathcal{M}^1_{k,j}\right\}$$

$$\mathcal{M}^1_{i,j} = \text{Min}_k\left\{b + \mathcal{M}^1_{i,j-1}, c + \mathcal{M}'_{i,j}\right\}$$

$\Theta(n)$ contributeurs potentiels au Min :
$\Rightarrow$ Complexité au pire en $\mathcal{O}(n^2)$ pour un backtrack naif.
Garder les meilleurs contributeurs aux Min $\Rightarrow$ Backtrack en $\mathcal{O}(n)$

Complexités temps/mémoire en $\mathcal{O}(n^3)/\mathcal{O}(n^2)$ pour le précalcul [1]
$\Rightarrow$ `UnaFold` [MZ08] minimise l'énergie de Turner.

1. Avec une astuce pour les bulges/boucles internes ...

Résumé

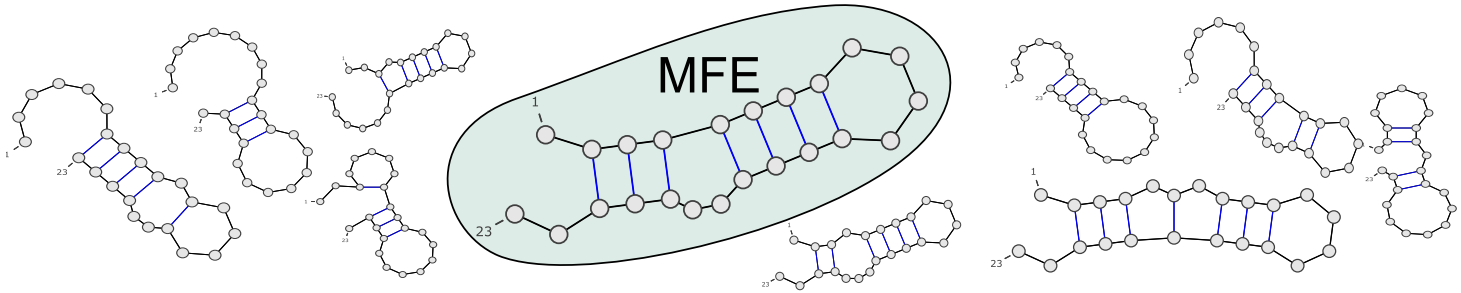
1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique

Ensemble canonique de Boltzmann

L'ARN *respire* $\Rightarrow$ Il n'existe pas UNE unique conformation native.

Nouveau paradigme

Les conformations d'un ARN coexistent dans une distribution de Boltzmann.



Conséquence : La probabilité de la MFE peut être négligeable.
$\Rightarrow$ Comprendre les modes d'actions de l'ARN exige de prendre en considération l'ensemble des structures.

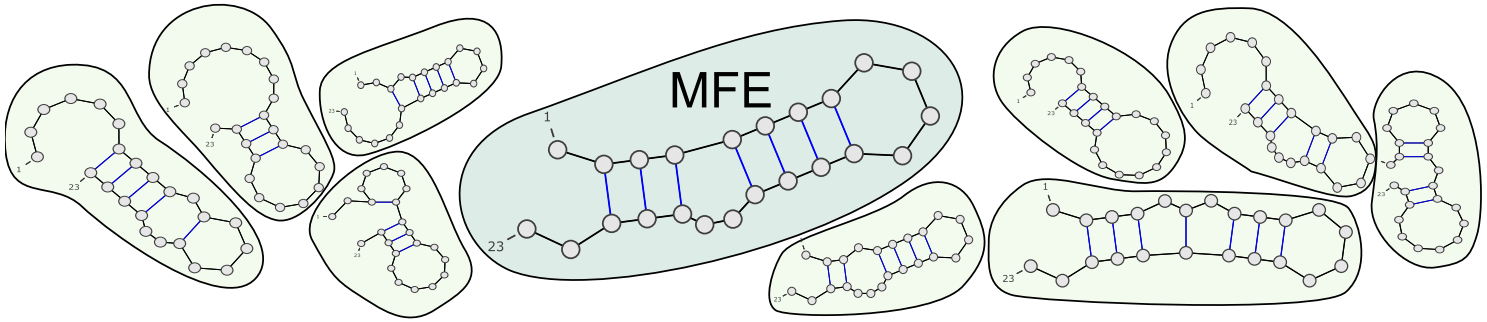
En particulier, des structures proches peuvent se *grouper* et devenir le candidat le plus crédible fonctionnellement.

Ensemble canonique de Boltzmann

L'ARN *respire* ⇒ Il n'existe pas UNE unique conformation native.

Nouveau paradigme

Les conformations d'un ARN coexistent dans une distribution de Boltzmann.



Conséquence : La probabilité de la MFE peut être négligeable.
⇒ Comprendre les modes d'actions de l'ARN exige de prendre en considération l'ensemble des structures.

En particulier, des structures proches peuvent se *grouper* et devenir le candidat le plus crédible fonctionnellement.

Distribution de Boltzmann : Définition

Notations : Séquence d'ARN ω

- $\mathcal{S}_\omega$ = ensemble des conformations compatibles avec ω.
- $E_{S,\omega}$ = énergie libre de S (kCal.mol^{-1})
- T = température (K)
- R = constante des gaz parfaits (1.986.10^{-3} kCal.K^{-1}.mol^{-1})

Définition (Distribution de Bolzmann)

Poids d'une structure S = facteur de Boltzmann : $\mathcal{B}_{S,\omega} = e^{\frac{-E_{S,\omega}}{RT}}$

Distribution renormalisée sur S_ω par la fonction de partition :

$$\mathcal{Z}_\omega = \sum_{S \in \mathcal{S}_\omega} e^{\frac{-E_{S,\omega}}{RT}}.$$

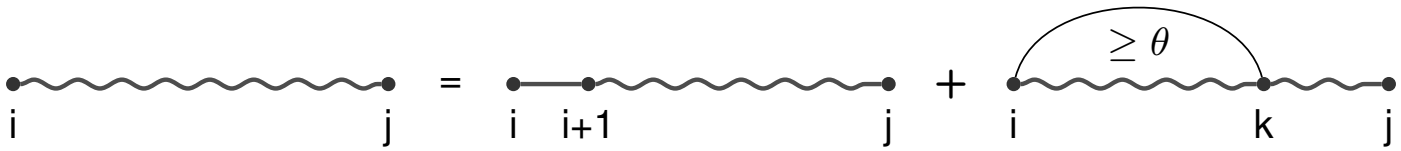
Probabilité de Boltzmann d'une structure S : $P_{S,\omega} = \frac{\mathcal{B}_{S,\omega}}{\mathcal{Z}_\omega}$.

Problème : Comment calculer efficacement la fonction de partition ?

Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$



Récurrence sur l'énergie minimale d'un repliement :

$$\begin{aligned} N_{i,t} &= 0, \quad \forall t \in [i, i+\theta] \\ N_{i,j} &= \min \begin{cases} N_{i+1,j} & (i \text{ non apparié}) \\ \min_{k=i+\theta+1}^{j} E_{i,k} + N_{i+1,k-1} + N_{k+1,j} & (i \text{ comp. avec } k) \end{cases} \end{aligned}$$

Récurrence de comptage des structures compatibles :

$$\begin{aligned} C_{i,t} &= 1, \quad \forall t \in [i, i+\theta] \\ C_{i,j} &= \sum \begin{cases} C_{i+1,j} & (i \text{ non apparié}) \\ \sum_{k=i+\theta+1}^{j} 1 \times C_{i+1,k-1} \times C_{k+1,j} & (i \text{ comp. avec } k) \end{cases} \end{aligned}$$

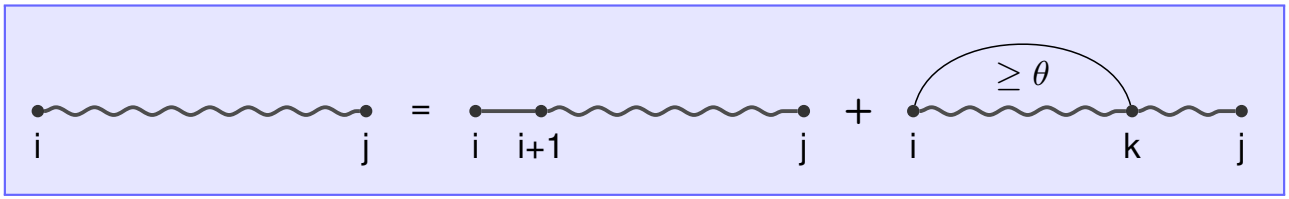
Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$

$$\textit{Fonction de partition} \rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$$

Fonction de partition = Comptage pondéré des structures compatibles



$$\begin{aligned} \mathcal{Z}_{i,t} &= 1, \quad \forall t \in [i, i+\theta] \\ \mathcal{Z}_{i,j} &= \sum \left\{ \sum_{k=i+\theta+1}^{j} 1 \times \mathcal{Z}_{i+1,k-1} \times \mathcal{Z}_{k+1,j} \right. \end{aligned}$$

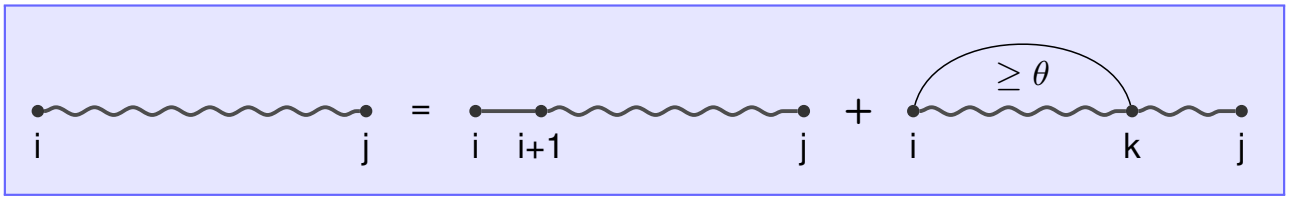
Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$

$$\textit{Fonction de partition} \rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$$

Fonction de partition = Comptage pondéré des structures compatibles



$$\mathcal{Z}_{i,t} = 1, \quad \forall t \in [i, i+\theta]$$

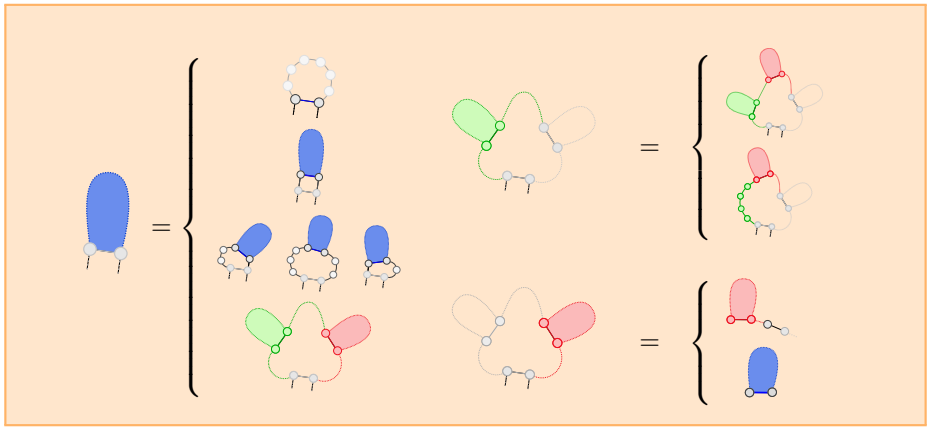
$$\mathcal{Z}_{i,j} = \sum \left\{ \sum_{k=i+\theta+1}^{j} e^{\frac{-E_{i,k}}{RT}} \times \mathcal{Z}_{i+1,k-1} \times \mathcal{Z}_{k+1,j} \right.$$

Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$

$$\textit{Fonction de partition} \rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$$



[McC90, MZ08]

$$\mathcal{M}'_{i,j} = \text{Min} \begin{cases} E_H(i,j) \\ E_S(i,j) + \mathcal{M}'_{i+1,j-1} \\ \text{Min}(E_{BI}(i,i',j',j) + \mathcal{M}'_{i',j'}) \\ a + c + \text{Min}\left(\mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}\right) \end{cases}$$

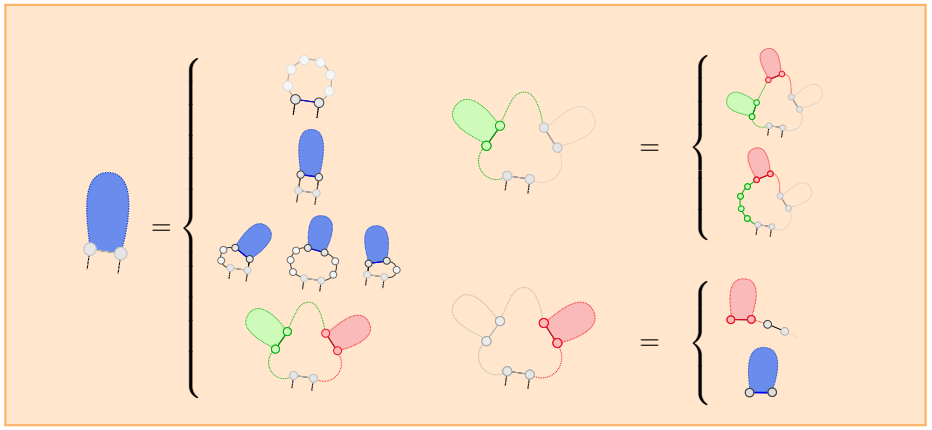
$$\mathcal{M}_{i,j} = \text{Min}\left\{\text{Min}\left(\mathcal{M}_{i,k-1}, b(k-1)\right) + \mathcal{M}^1_{k,j}\right\} \qquad \mathcal{M}^1_{i,j} = \text{Min}\left\{b + \mathcal{M}^1_{i,j-1}, c + \mathcal{M}'_{i,j}\right\}$$

Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$

$$\textit{Fonction de partition} \rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$$



[McC90, MZ08]

$$\mathcal{M}'_{i,j} = \text{Min} \begin{cases} e^{\frac{-E_H(i,j)}{RT}} \\ e^{\frac{-E_S(i,j)}{RT}} + \mathcal{M}'_{i+1,j-1} \\ \text{Min}\left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} + \mathcal{M}'_{i',j'}\right) \\ e^{\frac{-(a+c)}{RT}} + \text{Min}\left(\mathcal{M}_{i+1,k-1} + \mathcal{M}^1_{k,j-1}\right) \end{cases}$$

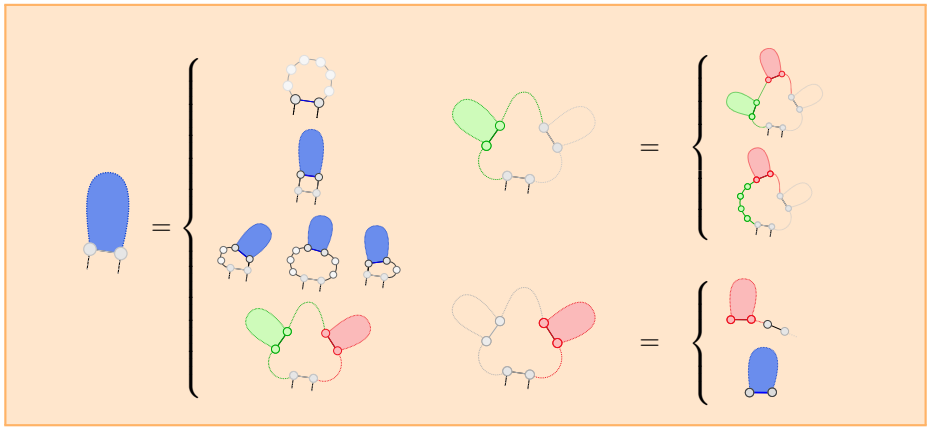
$$\mathcal{M}_{i,j} = \text{Min}\left\{\text{Min}\left(\mathcal{M}_{i,k-1}, e^{\frac{-b(k-1)}{RT}}\right) + \mathcal{M}^1_{k,j}\right\} \qquad \mathcal{M}^1_{i,j} = \text{Min}\left\{e^{\frac{-b}{RT}} + \mathcal{M}^1_{i,j-1}, e^{\frac{-c}{RT}} + \mathcal{M}'_{i,j}\right\}$$

Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

$$\textit{Optimisation} \rightarrow (\min / \max, +, E(\cdot), 0) \qquad \textit{Comptage} \rightarrow (+, \times, 1, 1)$$

$$\textit{Fonction de partition} \rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$$



[McC90, MZ08]

$$\mathcal{M}'_{i,j} = \text{Min} \begin{cases} e^{\frac{-E_H(i,j)}{RT}} \\ e^{\frac{-E_S(i,j)}{RT}} \mathcal{M}'_{i+1,j-1} \\ \text{Min}\left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{M}'_{i',j'}\right) \\ e^{\frac{-(a+c)}{RT}} \text{Min}\left(\mathcal{M}_{i+1,k-1} \mathcal{M}^1_{k,j-1}\right) \end{cases}$$

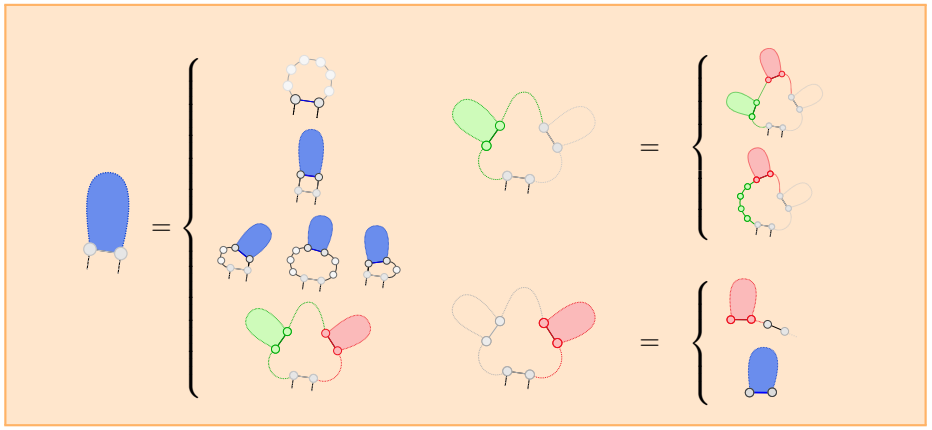
$$\mathcal{M}_{i,j} = \text{Min}\left\{\text{Min}\left(\mathcal{M}_{i,k-1}, e^{\frac{-b(k-1)}{RT}}\right) \mathcal{M}^1_{k,j}\right\} \qquad \mathcal{M}^1_{i,j} = \text{Min}\left\{e^{\frac{-b}{RT}} \mathcal{M}^1_{i,j-1}, e^{\frac{-c}{RT}} \mathcal{M}'_{i,j}\right\}$$

Théorème (Meta)

Décomposition complète et non-ambiguë, score défini additivement :

Optimisation $\rightarrow (\min / \max, +, E(\cdot), 0)$ *Comptage* $\rightarrow (+, \times, 1, 1)$

Fonction de partition $\rightarrow (+, \times, e^{-E(\cdot)/RT}, 1)$



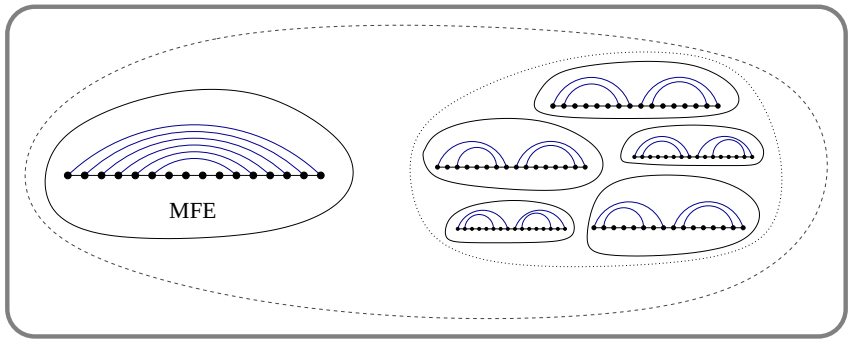
[McC90, MZ08]

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} \\ e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) \\ + \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) \\ + e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) \end{cases}$$

$$\mathcal{Z}(i,j) = \sum \left(\mathcal{Z}(i,k-1) + e^{\frac{-b(k-1)}{RT}} \right) \mathcal{Z}^1(k,j)$$

$$\mathcal{Z}^1(i,j) = e^{\frac{-b}{RT}} \mathcal{Z}^1(i,j-1) + e^{\frac{-c}{RT}} \mathcal{Z}'(i,j)$$

La MFE (struc. probabilité maximale) peut être écrasée par un ensemble $\mathcal{D}$ de sous-optimaux structurellement similaires.
$\Rightarrow$ Conformation fonctionnelle trouvée plus probablement dans $\mathcal{D}$.



Expérience : [DCL05]

- Engendrer des structures selon une distribution de Boltzmann
- Effectuer un clustering sur la structure
- Construire structure consensus dans le plus lourd cluster

$\Rightarrow$ Amélioration relative pour spécificité (+17.6%) et sensibilité (+21.74%, sauf Introns du groupe II)

Problème : Comment échantillonner la distribution de Boltzmann ?

Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}$, $\mathcal{Z}'$, $\mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

$$\mathcal{Z}'(i,j) = \boxed{???} \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

Algorithme (Reformulation SFold [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}$, $\mathcal{Z}'$, $\mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$

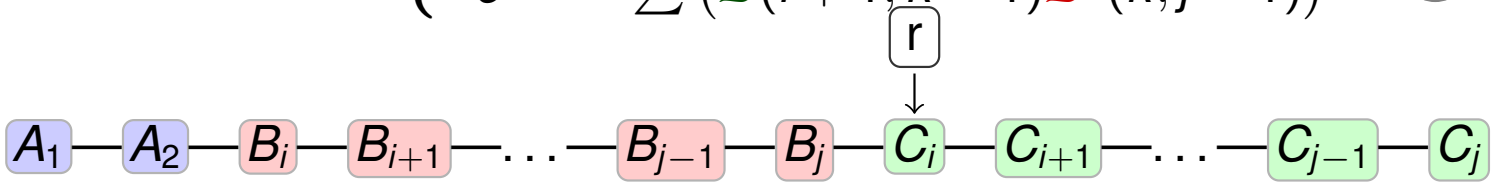
$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}, \mathcal{Z}', \mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

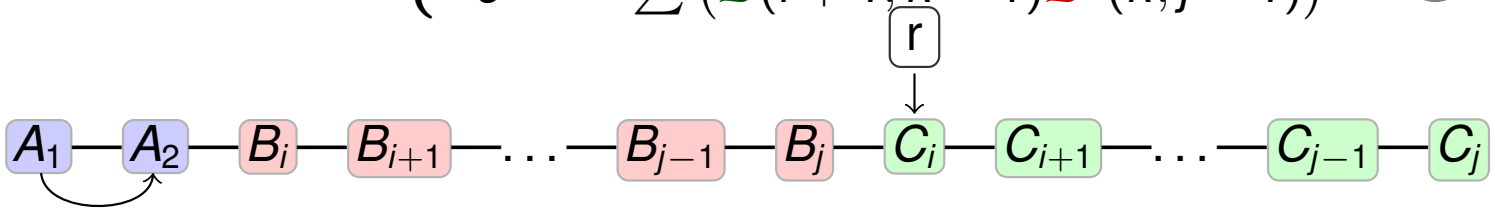


Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}, \mathcal{Z}', \mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

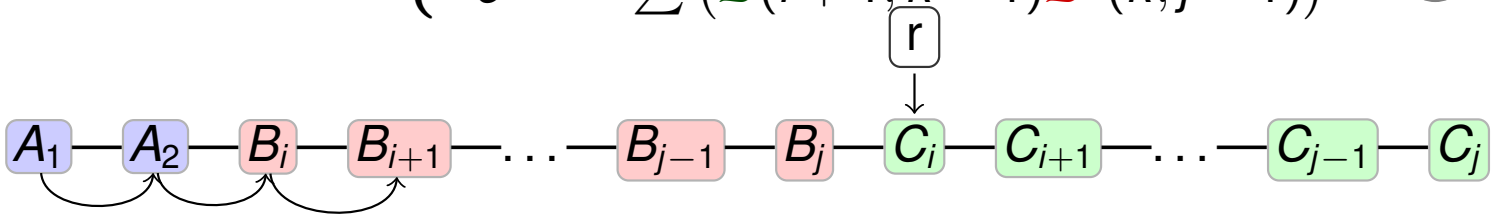


Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}$, $\mathcal{Z}'$, $\mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

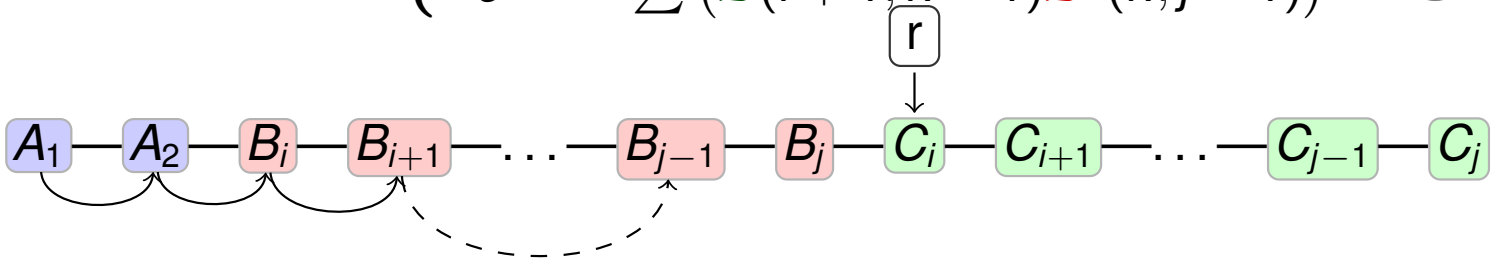


Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}$, $\mathcal{Z}'$, $\mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

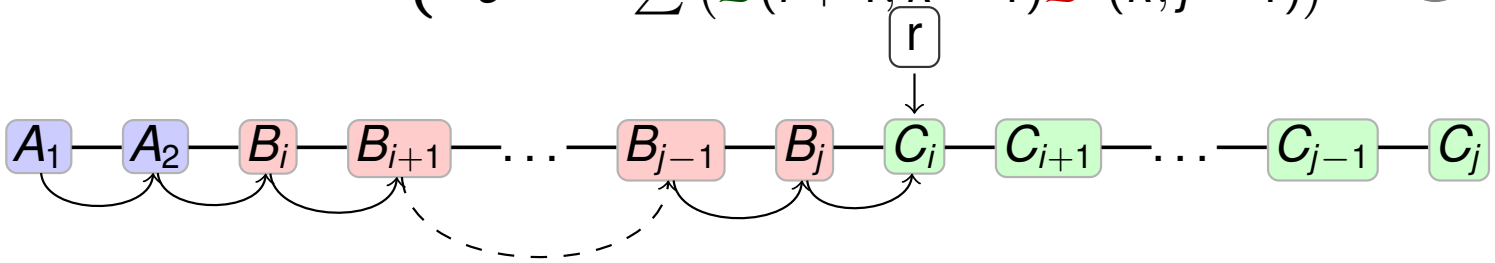


Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}, \mathcal{Z}', \mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$



Algorithme (Reformulation `SFold` [DL03])

Précalcul : Calculer les matrices ($\mathcal{Z}, \mathcal{Z}', \mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$
3. Rappels récursifs sur les termes issus de la contribution choisie.

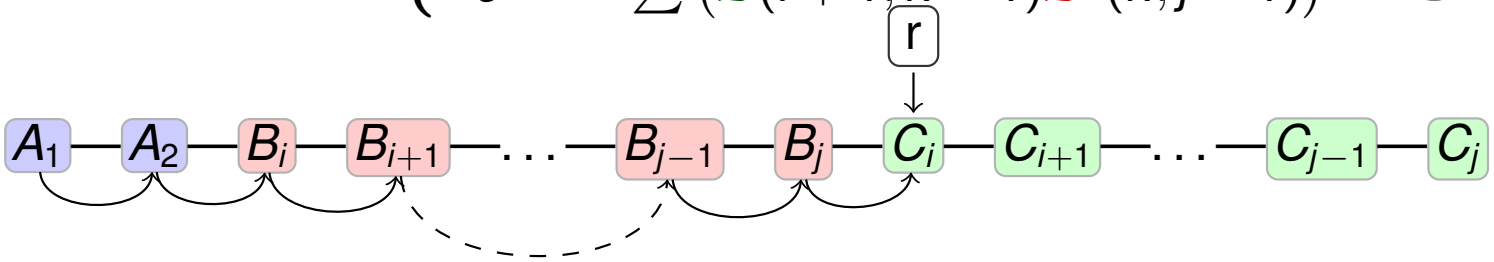
$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

Algorithme (Reformulation `SFold` [DL03]) : Complexité

Précalcul : Calculer les matrices ($\mathcal{Z}$, $\mathcal{Z}'$, $\mathcal{Z}^1$) de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$
3. Rappels récursifs sur les termes issus de la contribution choisie.

$$\mathcal{Z}'(i,j) = \sum ??? \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$

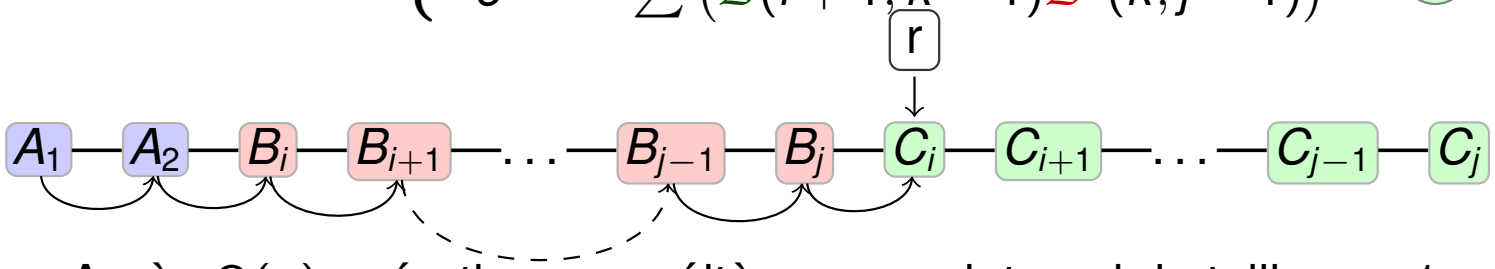


Algorithme (Reformulation `SFold` [DL03]) : Complexité

Précalcul : Calculer les matrices $(\mathcal{Z}, \mathcal{Z}', \mathcal{Z}^1)$ de la fonction de partition.
Génération (Remontée stochastique) :

1. Tirer un nombre aléatoire r uniformément sur $[0, \mathcal{Z}'(i,j)[$
2. Soustraire à r les contributions à $\mathcal{Z}'(i,j)$, jusqu'à ce que $r < 0$
3. Rappels récursifs sur les termes issus de la contribution choisie.

$$\mathcal{Z}'(i,j) = \sum \begin{cases} e^{\frac{-E_H(i,j)}{RT}} + e^{\frac{-E_S(i,j)}{RT}} \mathcal{Z}'(i+1,j-1) & \text{A} \\ \sum \left(e^{\frac{-E_{BI}(i,i',j',j)}{RT}} \mathcal{Z}'(i',j') \right) & \text{B} \\ e^{\frac{-(a+c)}{RT}} \sum \left(\mathcal{Z}(i+1,k-1) \mathcal{Z}^1(k,j-1) \right) & \text{C} \end{cases}$$



Après $\Theta(n)$ opérations, on réitère sur un interval de taille $n-1$
$\Rightarrow$ Complexité au pire en $\mathcal{O}(n^2 k)$ pour k échantillons

Remarque : Équivalent à une génération aléatoire pondérée [DPT10].
Complexité en moyenne en $\Theta(n\sqrt{n})$, mais
Parcours Boustrophedon [Pon08] $\rightarrow \mathcal{O}(n \log nk)$ au pire.

Résumé

1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique

Pourquoi ça marche ?

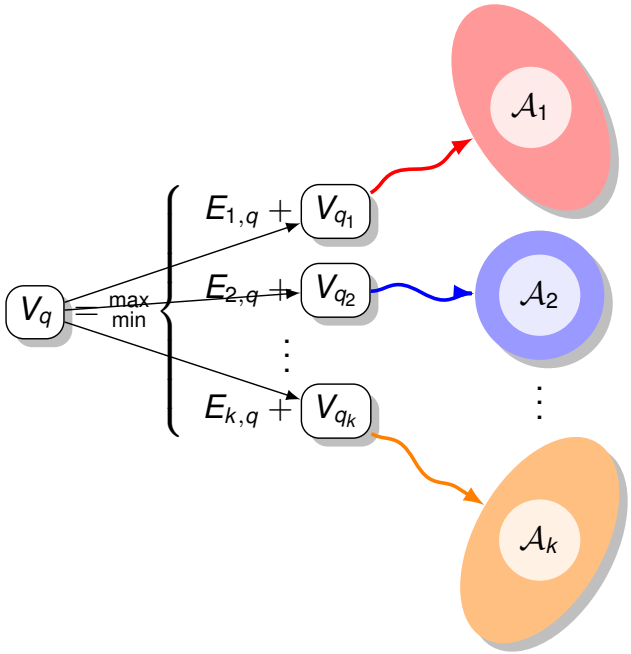
Equation de prog. dyn. = Générateur d'un (sous) espace de recherche
Complétude : Parcours intégral de l'espace de recherche
Non-ambiguité : Choix locaux $\rightarrow$ ensembles mutuellement disjoints

Correction du comptage

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \cdots \cup \mathcal{A}_k$$

$$+ \quad \mathcal{A}_i \cap \mathcal{A}_j = \varnothing, \forall i \neq j \text{ (non-ambiguité)}$$

$$\Rightarrow \quad |\mathcal{A}| = |\mathcal{A}_1| + |\mathcal{A}_2| + |\mathcal{A}_k|$$



Cas non linéaire : Argument similaire

$$A_1 \rightarrow A_{1,1} \times A_{1,2} \times \cdots$$

$$|A_1| \rightarrow |A_{1,1}| \times |A_{1,2}| \times \cdots$$

Pourquoi ça marche ?

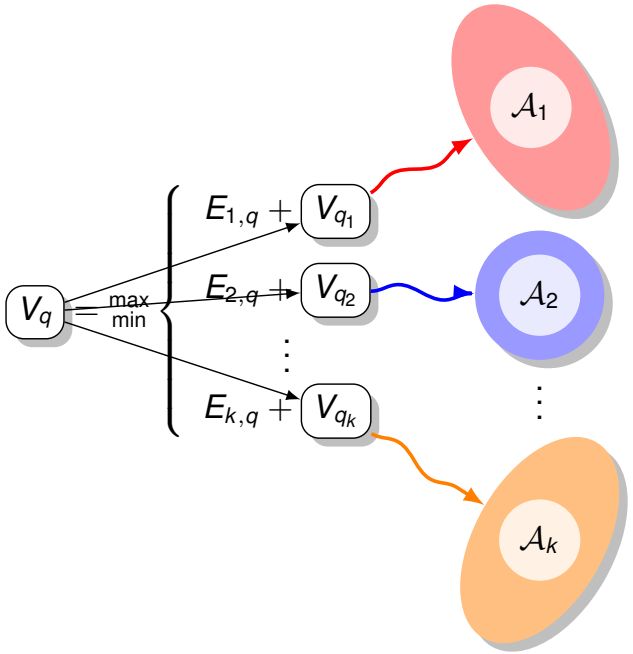
Equation de prog. dyn. = Générateur d'un (sous) espace de recherche
Complétude : Parcours intégral de l'espace de recherche
Non-ambiguité : Choix locaux $\rightarrow$ ensembles mutuellement disjoints

Correction du comptage

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \cdots \cup \mathcal{A}_k$$

$$+ \quad \mathcal{A}_i \cap \mathcal{A}_j = \varnothing, \forall i \neq j \text{ (non-ambiguité)}$$

$$\Rightarrow \quad |\mathcal{A}| = |\mathcal{A}_1| + |\mathcal{A}_2| + |\mathcal{A}_k|$$



Cas non linéaire : Argument similaire

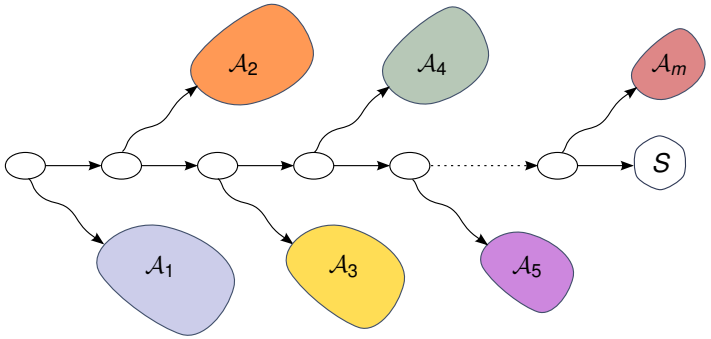
$$A_1 \rightarrow A_{1,1} \times A_{1,2} \times \cdots$$

$$|A_1| \rightarrow |A_{1,1}| \times |A_{1,2}| \times \cdots$$

Correction de l'échantillonnage

$(E_i := \cup_{k=i+1}^{m} \mathcal{A}_k)$

$$p_S = \frac{\mathcal{B}(E_1)}{\mathcal{B}(\mathcal{S}_w)} \cdot \frac{\mathcal{B}(E_2)}{\mathcal{B}(E_1)} \cdot \frac{\mathcal{B}(E_3)}{\mathcal{B}(E_2)} \cdots \frac{\mathcal{B}(\{S\})}{\mathcal{B}(E_m)}$$



Pourquoi ça marche ?

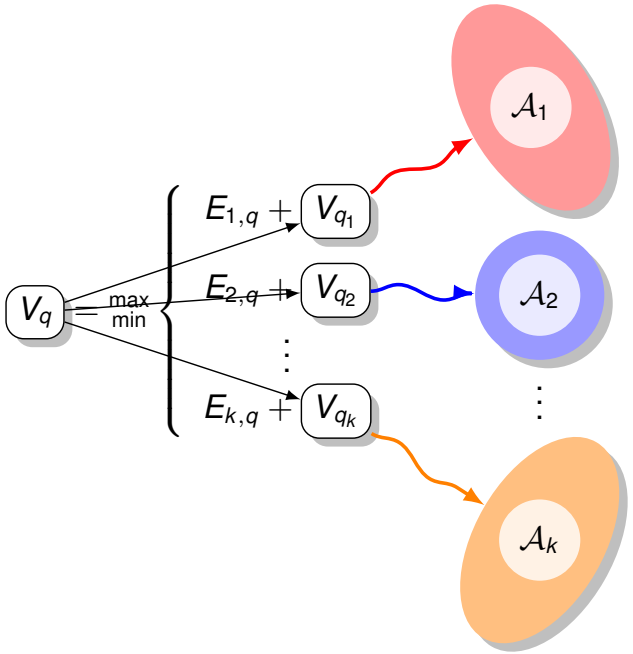
Equation de prog. dyn. = Générateur d'un (sous) espace de recherche
Complétude : Parcours intégral de l'espace de recherche
Non-ambiguité : Choix locaux $\rightarrow$ ensembles mutuellement disjoints

Correction du comptage

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \cdots \cup \mathcal{A}_k$$

$$+ \quad \mathcal{A}_i \cap \mathcal{A}_j = \varnothing, \forall i \neq j \text{ (non-ambiguité)}$$

$$\Rightarrow \quad |\mathcal{A}| = |\mathcal{A}_1| + |\mathcal{A}_2| + |\mathcal{A}_k|$$



Cas non linéaire : Argument similaire

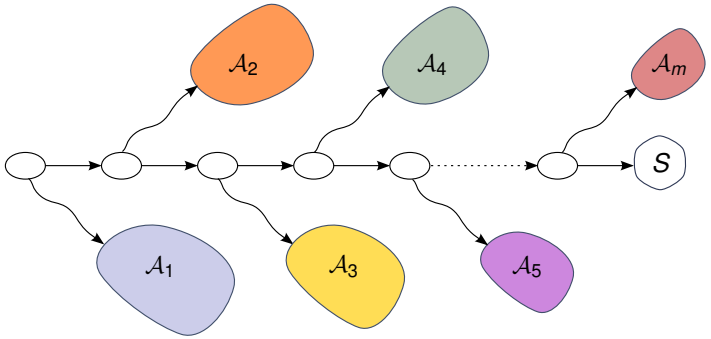
$$A_1 \rightarrow A_{1,1} \times A_{1,2} \times \cdots$$

$$|A_1| \rightarrow |A_{1,1}| \times |A_{1,2}| \times \cdots$$

Correction de l'échantillonnage

$(E_i := \cup_{k=i+1}^{m} \mathcal{A}_k)$

$$p_S = \frac{1}{\mathcal{B}(\mathcal{S}_w)} \cdot \frac{1}{1} \cdot \frac{1}{1} \cdots \frac{\mathcal{B}(\{S\})}{1}$$



Pourquoi ça marche ?

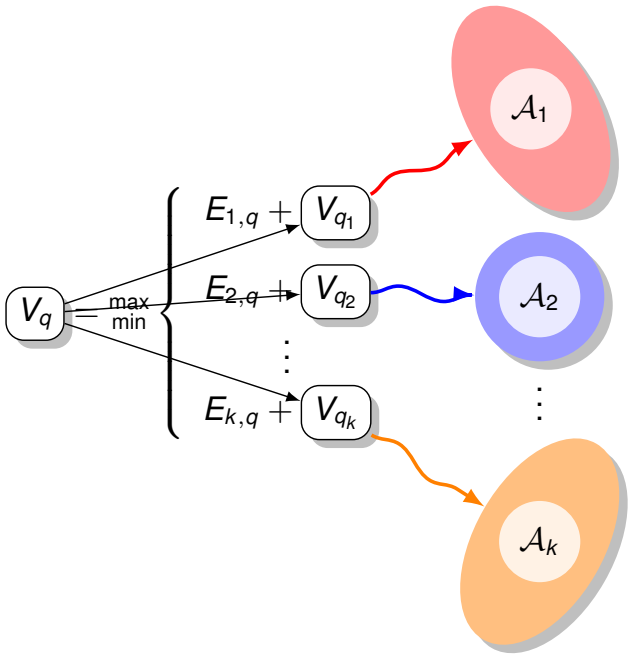
Equation de prog. dyn. = Générateur d'un (sous) espace de recherche
Complétude : Parcours intégral de l'espace de recherche
Non-ambiguité : Choix locaux $\rightarrow$ ensembles mutuellement disjoints

Correction du comptage

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \cdots \cup \mathcal{A}_k$$

$$+ \quad \mathcal{A}_i \cap \mathcal{A}_j = \varnothing, \forall i \neq j \text{ (non-ambiguité)}$$

$$\Rightarrow \quad |\mathcal{A}| = |\mathcal{A}_1| + |\mathcal{A}_2| + |\mathcal{A}_k|$$



Cas non linéaire : Argument similaire

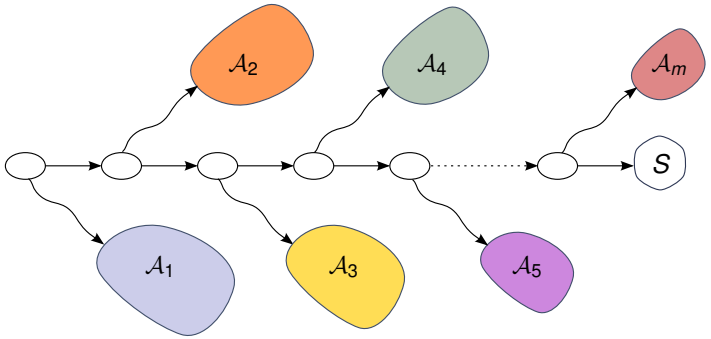
$$A_1 \rightarrow A_{1,1} \times A_{1,2} \times \cdots$$

$$|A_1| \rightarrow |A_{1,1}| \times |A_{1,2}| \times \cdots$$

Correction de l'échantillonnage

$(E_i := \cup_{k=i+1}^{m} \mathcal{A}_k)$

$$p_S = \frac{\mathcal{B}(\{S\})}{\mathcal{B}(\mathcal{S}_w)} = \frac{e^{-E_s/RT}}{\mathcal{Z}} = P_{S,\omega}$$



Formalisation de la programmation dynamique

The development of successful dynamic programming recurrences is a matter of experience, talent and luck – Anonyme, 2000

Tout algorithme de programmation dynamique définit :

1. Espace de recherche
2. Fonction objectif ou score
3. Décomposition

Entités conceptuelles non-indépendantes :

- Décomposition doit parcourir tout l'Espace de recherche de façon unique
- Décomposition doit permettre un calcul local de la Fonction objectif

Choix d'une bonne décomposition est crucial !

Compilation de ces trois éléments ⇒ Équation de programmation dynamique.
Question : Comment formaliser cette notion de compilation ?

Propositions :
Grammaires (Giegerich *et al* [SMJG13]) Hypergraphes (Roytberg *et al* [FR93])

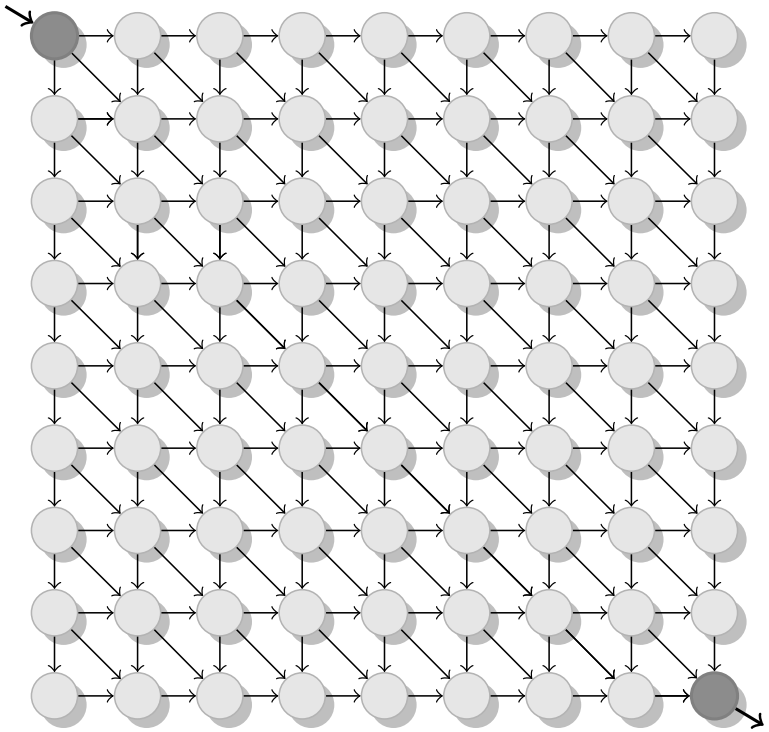
Needleman-Wunsch [NW70] (alignement global) :

Remontée = Chemin de score maximal dans la matrice

		A	C	A	C	A	C	T	A
	0	0	0	0	0	0	0	0	0
A	0	2	1	2	1	2	1	0	2
G	0	1	1	1	1	1	1	0	1
C	0	0	3	2	3	2	3	2	1
A	0	2	2	5	4	5	4	3	4
C	0	1	4	4	7	6	7	6	5
A	0	2	3	6	6	9	8	7	8
C	0	1	4	5	8	8	11	10	9
A	0	2	3	6	7	10	10	10	12

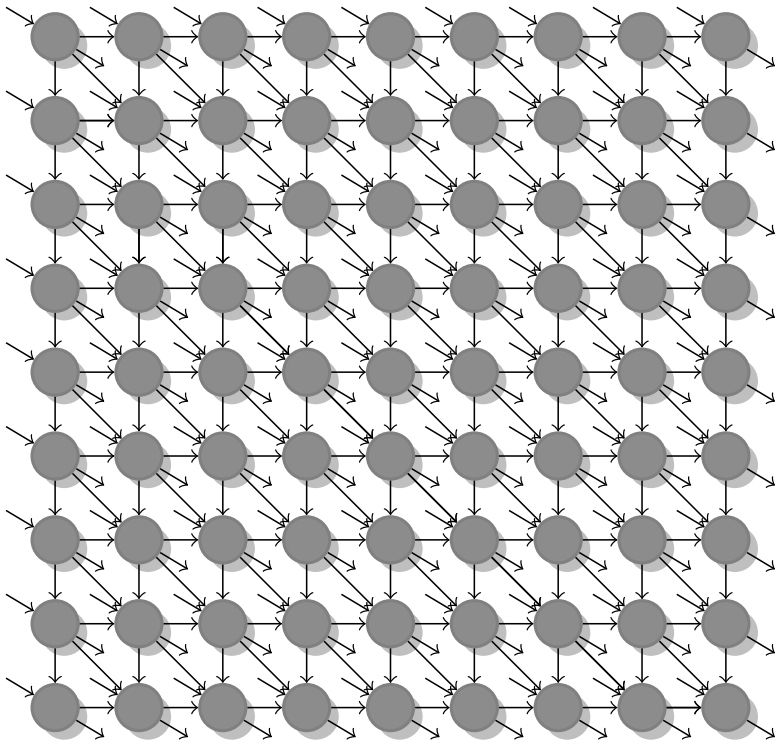
Needleman-Wunsch [NW70] (alignement global) :

Remontée = Chemin de score maximal dans la matrice



Smith-Waterman [SW81] (alignement global) :

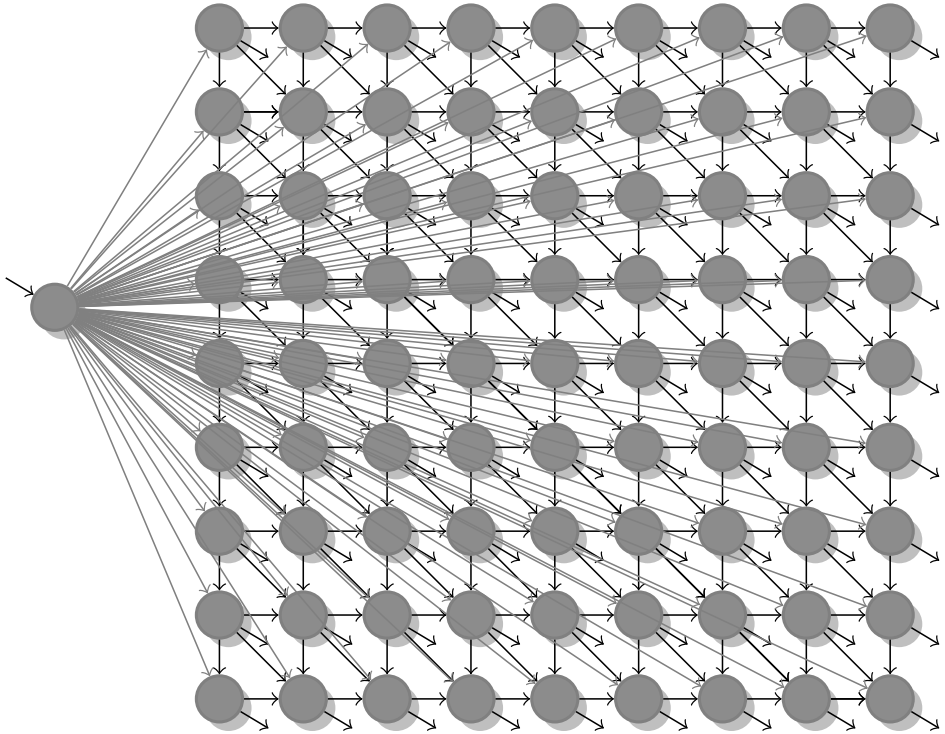
Remontée = Chemin de score maximal dans la matrice



Pourquoi les hypergraphes ?

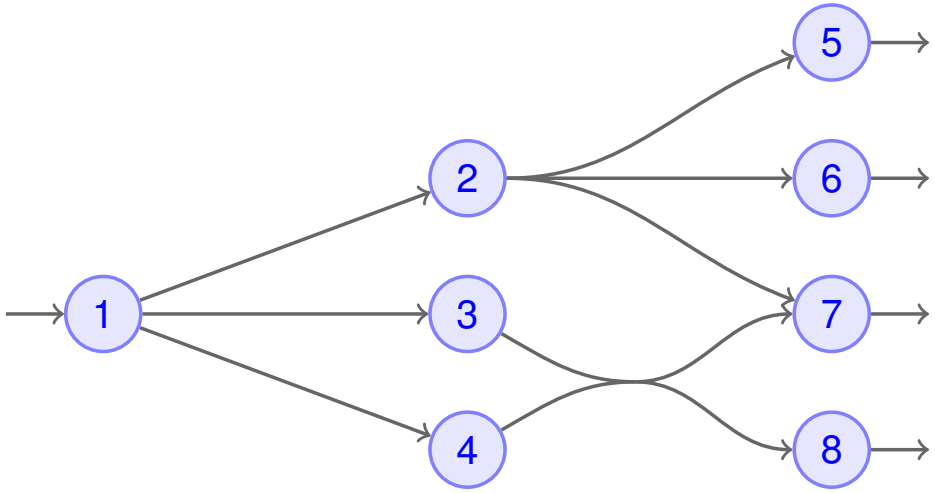
Smith-Waterman [SW81] (alignement global) :

Remontée = Chemin de score maximal dans la matrice



Mais : Analogie précaire en ARN car les remontées sont des arbres.
Il faudrait des graphes dont les chemins sont des arbres ? ? ?

→ Hypergraphes !

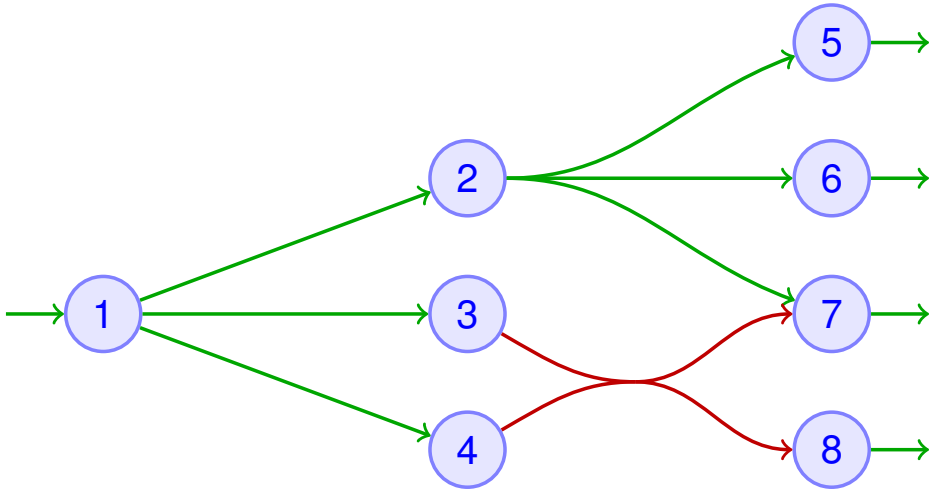


Les hypergaphes (orientés) généralisent les graphes (dirigés) grâce à des (hyper) arcs de degrés entrant/sortant arbitraires.

Définition (Hypergraphes)

Un hypergraphe $\mathcal{H}$ est un couple (V, E) tel que :

- $V =$ ensemble de sommet
- $E =$ ensemble d'hyperarcs $e = (t(e) \to h(e))$ tels que $t(e), h(e) \subset E$



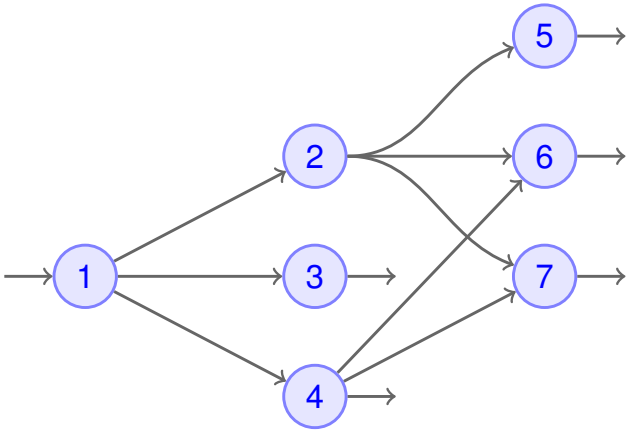
Les hypergaphes (orientés) généralisent les graphes (dirigés) grâce à des (hyper) arcs de degrés entrant/sortant arbitraires.

Définition (Hypergraphes)

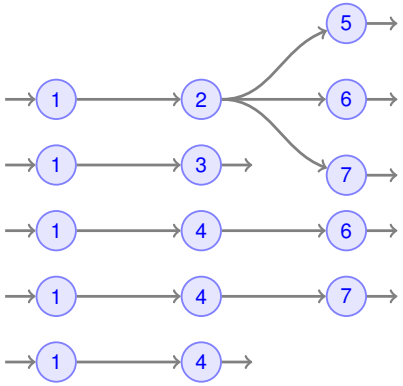
Un hypergraphe $\mathcal{H}$ est un couple (V, E) tel que :

- $V =$ ensemble de sommet
- $E =$ ensemble d'hyperarcs $e = (t(e) \to h(e))$ tels que $t(e), h(e) \subset E$

Les hypergraphes avant, ou F(orward)-graphes, sont des hypergraphes dont les arcs ont degré entrant exactement égal à 1.



F-graphe

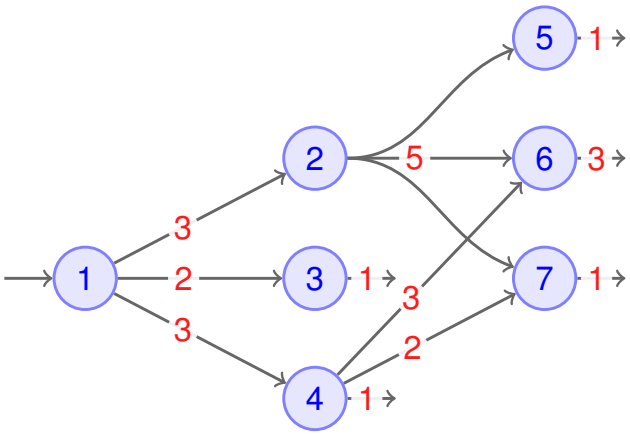


Tous les F-chemins partant du sommet 1

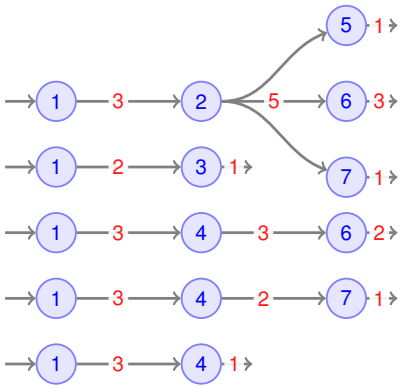
Définition (F-chemin)

Un F-chemin est un arbre de racine $s \in V$, et dont les fils sont des F-chemins construits à partir des sommets sortants d'un arc $e = (s \to \mathbf{t}) \in E$.

Remarque : Des arcs de degré sortant nul ($\mathbf{t} = \varnothing$) *ferment* cette définition.



F-graphe



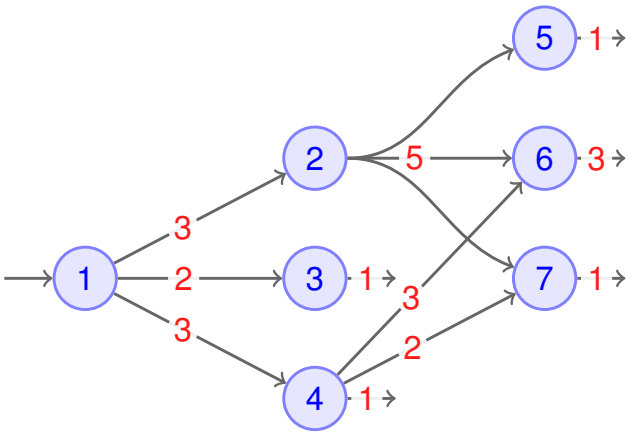
Tous les F-chemins partant du sommet 1

Définition (F-chemin)

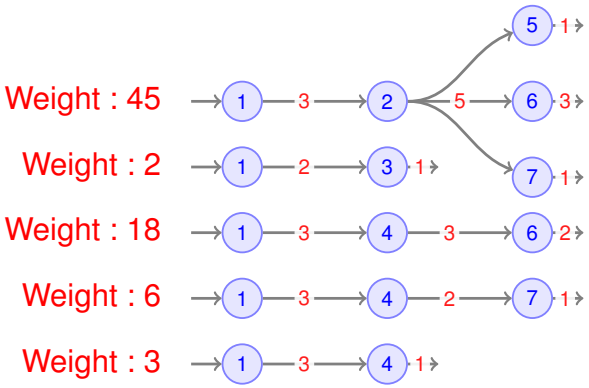
Un F-chemin est un arbre de racine $s \in V$, et dont les fils sont des F-chemins construits à partir des sommets sortants d'un arc $e = (s \to \mathbf{t}) \in E$.

Remarque : Des arcs de degré sortant nul ($\mathbf{t} = \varnothing$) *ferment* cette définition.

Indépendance : Un F-graphe est **indépendant** ssi chaque arc est emprunté au plus une fois par chaque chemin.



F-graphe



Tous les F-chemins partant du sommet 1

Définition (F-chemin)

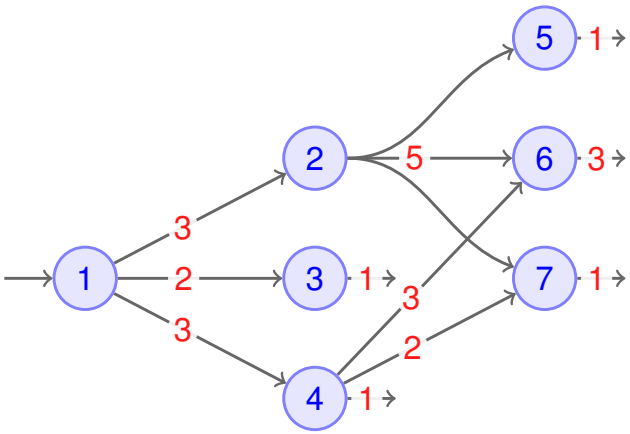
Un F-chemin est un arbre de racine $s \in V$, et dont les fils sont des F-chemins construits à partir des sommets sortants d'un arc $e = (s \to \mathbf{t}) \in E$.

Remarque : Des arcs de degré sortant nul ($\mathbf{t} = \varnothing$) *ferment* cette définition.

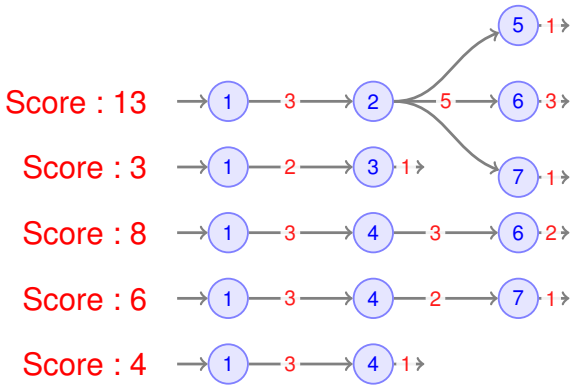
Indépendance : Un F-graphe est **indépendant** ssi chaque arc est emprunté au plus une fois par chaque chemin.

Valuation : Fonction $\pi : E \to \mathbb{R}$ donnant valeur réelle aux arcs $e \in E$.

- **Poids** d'un chemin = **Produit** des valeurs de ses arcs.
- **Score** d'un chemin = **Somme** des valeurs de ses arcs.



F-graphe



Tous les F-chemins partant du sommet 1

Définition (F-chemin)

Un F-chemin est un arbre de racine $s \in V$, et dont les fils sont des F-chemins construits à partir des sommets sortants d'un arc $e = (s \to \mathbf{t}) \in E$.

Remarque : Des arcs de degré sortant nul ($\mathbf{t} = \varnothing$) *ferment* cette définition.

Indépendance : Un F-graphe est **indépendant** ssi chaque arc est emprunté au plus une fois par chaque chemin.

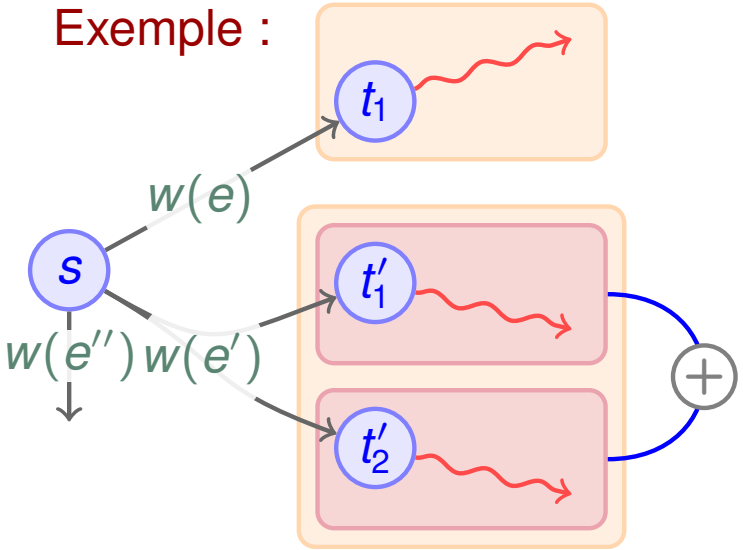
Valuation : Fonction $\pi : E \to \mathbb{R}$ donnant valeur réelle aux arcs $e \in E$.

- **Poids** d'un chemin = **Produit** des valeurs de ses arcs.
- **Score** d'un chemin = **Somme** des valeurs de ses arcs.

$\mathcal{H} = (s_0, V, E, \pi)$: F-graphe acyclique s_0 : Noeud initial π : Valuation

$$m_s = \min_{e=(s\to\mathbf{t})\in E} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$$

Exemple :

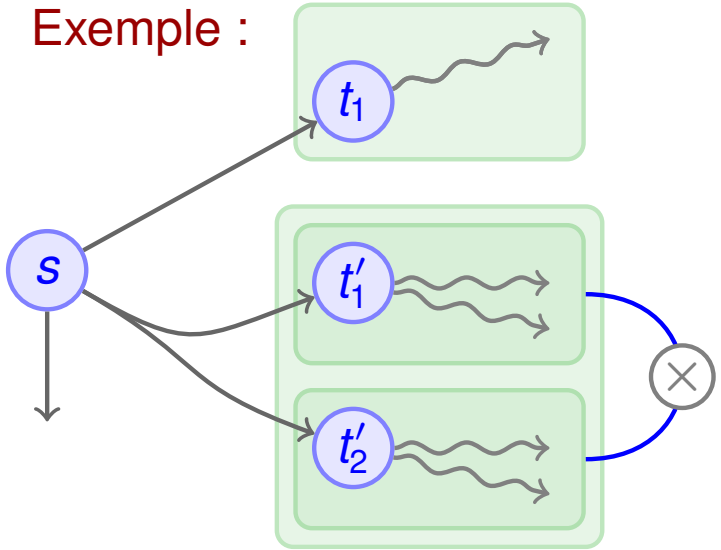


Problème	Récurrence	Temps/mémoire
Score min.	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$

$\mathcal{H} = (s_0, V, E, \pi)$: F-graphe acyclique s_0 : Noeud initial π : Valuation

$$n_s = \sum_{(s \to \mathbf{t}) \in E} \prod_{u \in \mathbf{t}} n_u$$

Exemple :

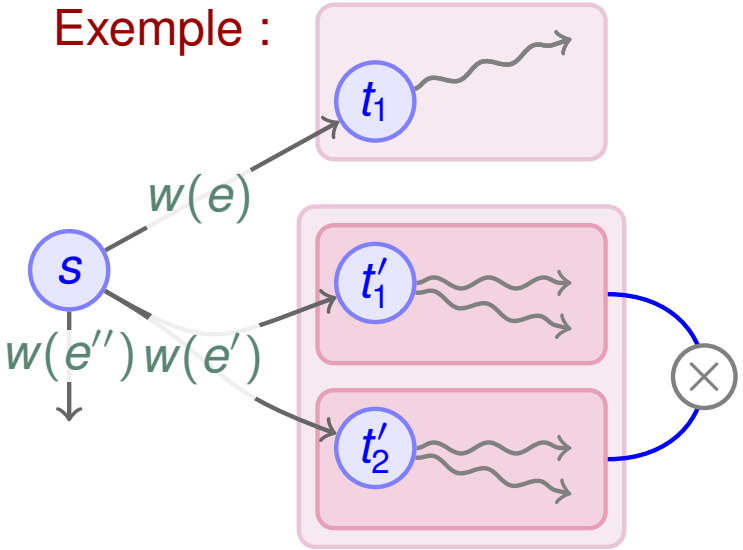


Problème	Récurrence	Temps/mémoire
Score min.	$m_s = \min_{e=(s \to \mathbf{t})} \left(w(e) + \sum_{u \in \mathbf{t}} m_u \right)$	$\Theta(\lvert E \rvert + \lvert V \rvert)/\Theta(\lvert V \rvert)$
#Chemins	$n_s = \sum_{(s \to \mathbf{t})} \prod_{u \in \mathbf{t}} n_u$	$\Theta(\lvert E \rvert + \lvert V \rvert)/\Theta(\lvert V \rvert)$

$\mathcal{H} = (s_0, V, E, \pi)$: F-graphe acyclique s_0 : Noeud initial π : Valuation

$$w_s = \sum_{e=(s \to \mathbf{h}(e)) \in E} w(e) \cdot \prod_{s' \in \mathbf{h}(e)} w_{s'}$$

Exemple :



Problème	Récurrence	Temps/mémoire
Score min.	$m_s = \min_{e=(s \to \mathbf{t})} \left(w(e) + \sum_{u \in \mathbf{t}} m_u \right)$	$\Theta(\lvert E \rvert + \lvert V \rvert)/\Theta(\lvert V \rvert)$
#Chemins	$n_s = \sum_{(s \to \mathbf{t})} \prod_{u \in \mathbf{t}} n_u$	$\Theta(\lvert E \rvert + \lvert V \rvert)/\Theta(\lvert V \rvert)$
Poids total	$w_s = \sum_{e=(s \to \mathbf{t})} w(e) \cdot \prod_{s' \in \mathbf{t}} w_{s'}$	$\Theta(\lvert E \rvert + \lvert V \rvert)/\Theta(\lvert V \rvert)$

Generic algorithms [PS11]

Problème	Récurrence	Temps/mémoire
Score min.	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
#Chemins	$n_s = \sum_{(s\to\mathbf{t})} \prod_{u\in\mathbf{t}} n_u$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
Poids total	$w_s = \sum_{e=(s\to\mathbf{t})} w(e) \cdot \prod_{s'\in\mathbf{t}} w_{s'}$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$

Supposons une distribution pondérée (Boltzmann) sur les F-paths $\mathcal{P}$:

$$\mathbb{P}(p) = \prod_{e\in p} \pi(e)/w_{s_0}$$

Problème	Algorithme	Temps/mémoire
Gén. aléatoire	Calculer w_s ; En partant de $s \leftarrow s_0$, choisir un arc $s \to (t_1, t_2, \ldots)$ avec proba. $\prod w_{t_i}/w_s$ et réitérer sur les t_i.	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$

Generic algorithms [PS11]

Problème	Récurrence	Temps/mémoire
Score min.	$m_s = \min_{e=(s\to\mathbf{t})} \left(w(e) + \sum_{u\in\mathbf{t}} m_u \right)$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
#Chemins	$n_s = \sum_{(s\to\mathbf{t})} \prod_{u\in\mathbf{t}} n_u$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
Poids total	$w_s = \sum_{e=(s\to\mathbf{t})} w(e) \cdot \prod_{s'\in\mathbf{t}} w_{s'}$	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$

Supposons une distribution pondérée (Boltzmann) sur les F-paths $\mathcal{P}$:

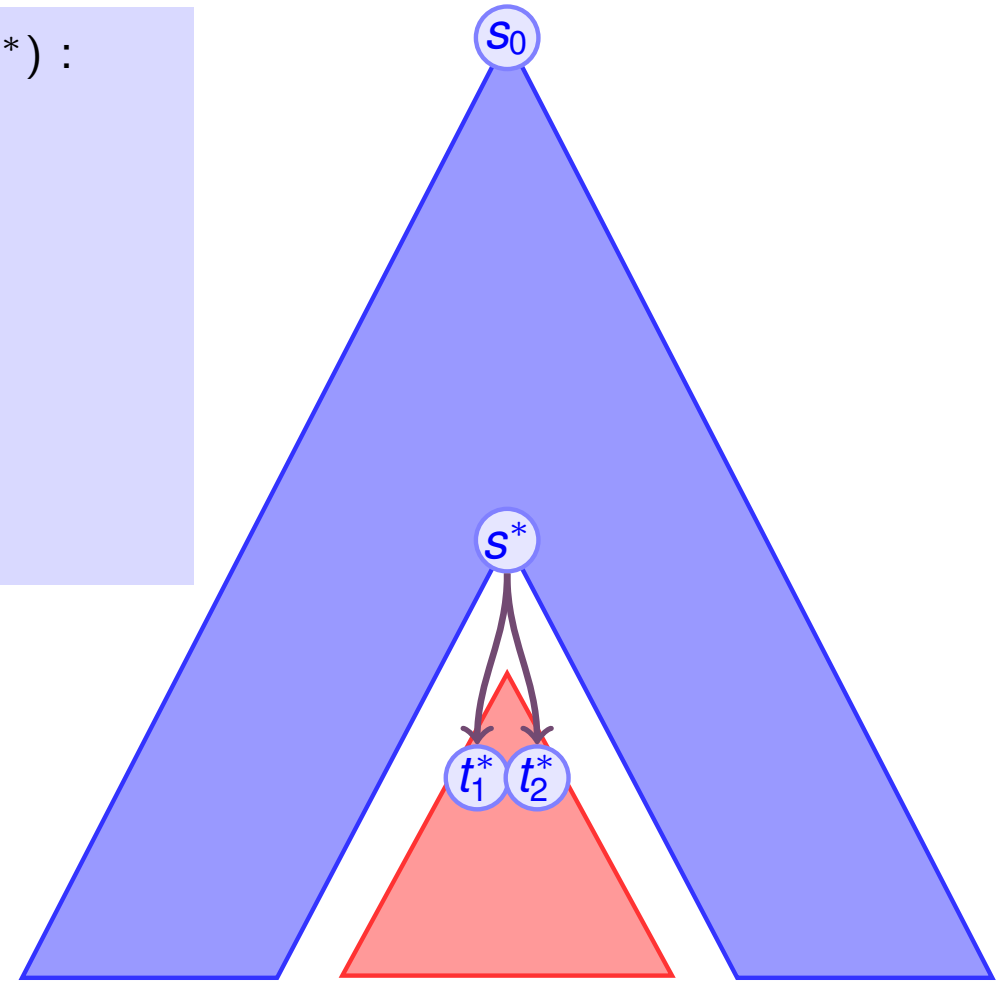
$$\mathbb{P}(p) = \prod_{e\in p} \pi(e)/w_{s_0}$$

Problème	Algorithme	Temps/mémoire
Gén. aléatoire	Calculer w_s ; En partant de $s \leftarrow s_0$, choisir un arc $s \to (t_1, t_2, \ldots)$ avec proba. $\prod w_{t_i}/w_s$ et réitérer sur les t_i.	$\Theta(\lvert E\rvert + \lvert V\rvert)/\Theta(\lvert V\rvert)$
Proba. arcs	$p_e = \dfrac{b_{t(e)} \cdot \prod_{s'\in\mathbf{h}(e)} w_{s'}}{w_{s_0}}$ $b_s = (1+) \sum_{s'\to(t_1\cdots s\cdots)} \pi(e') b_{s'} \prod_{t_i} w_{t_i}$	$\Theta(\lvert E\rvert + \lvert V\rvert + \sum \lvert\mathbf{h}(e)\rvert^2)$ $\Theta(\lvert V\rvert)$

Décomposition

F-chemins empruntant $e^* = (s^* \rightarrow \mathbf{t}^*)$:

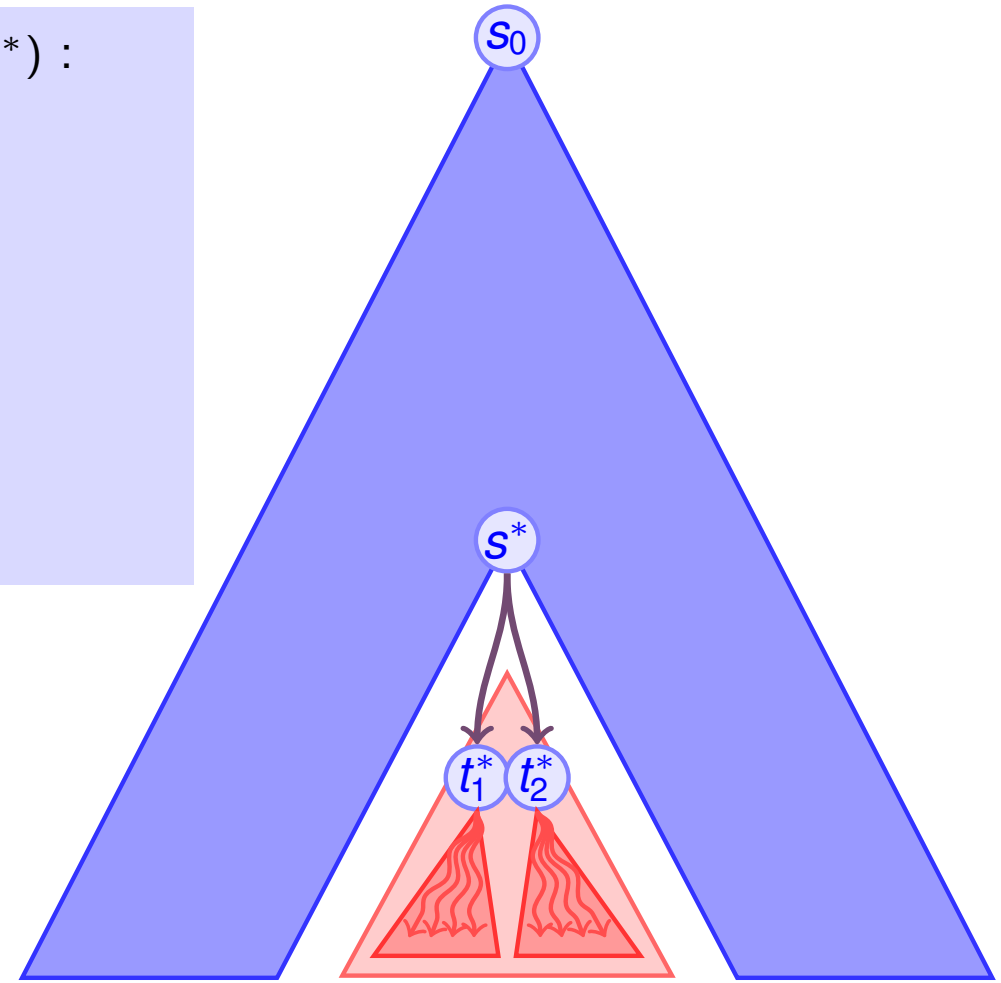
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$



Décomposition

F-chemins empruntant $e^* = (s^* \rightarrow \mathbf{t}^*)$:

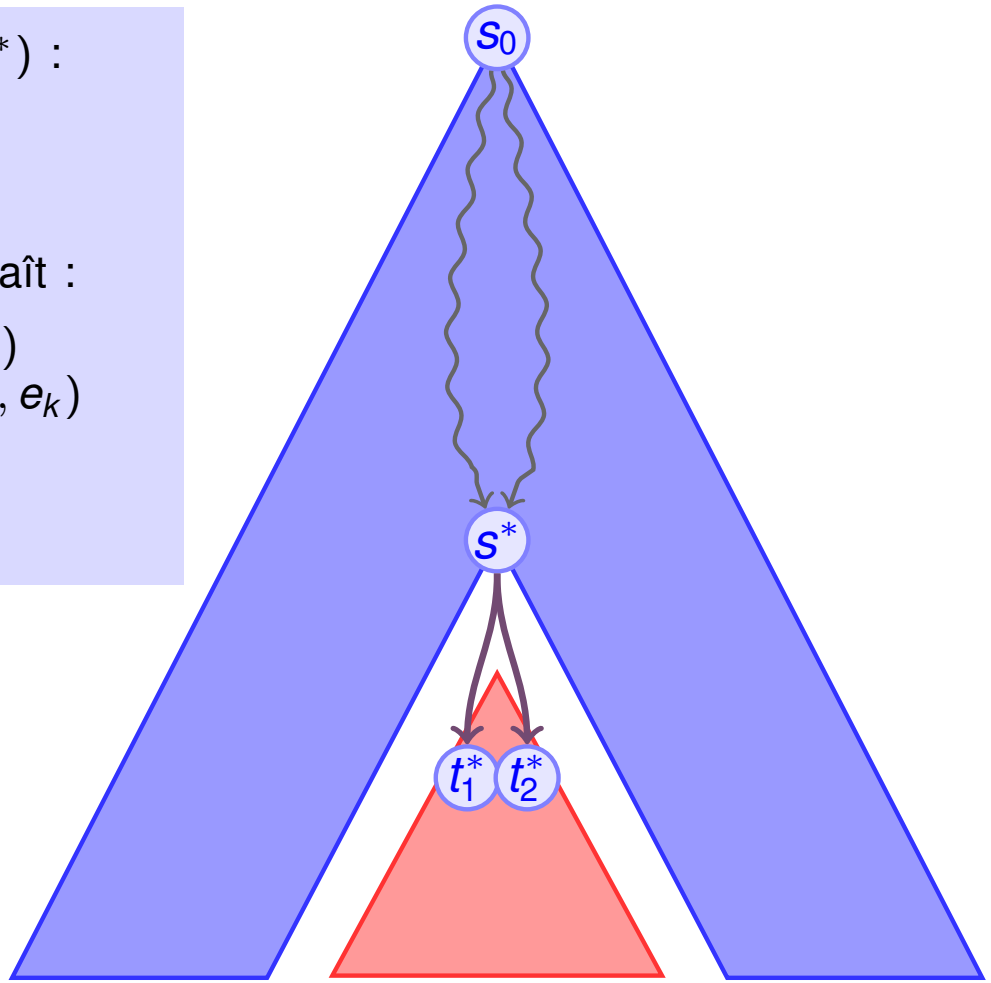
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$



Décomposition

F-chemins empruntant $e^* = (s^* \to \mathbf{t}^*)$:

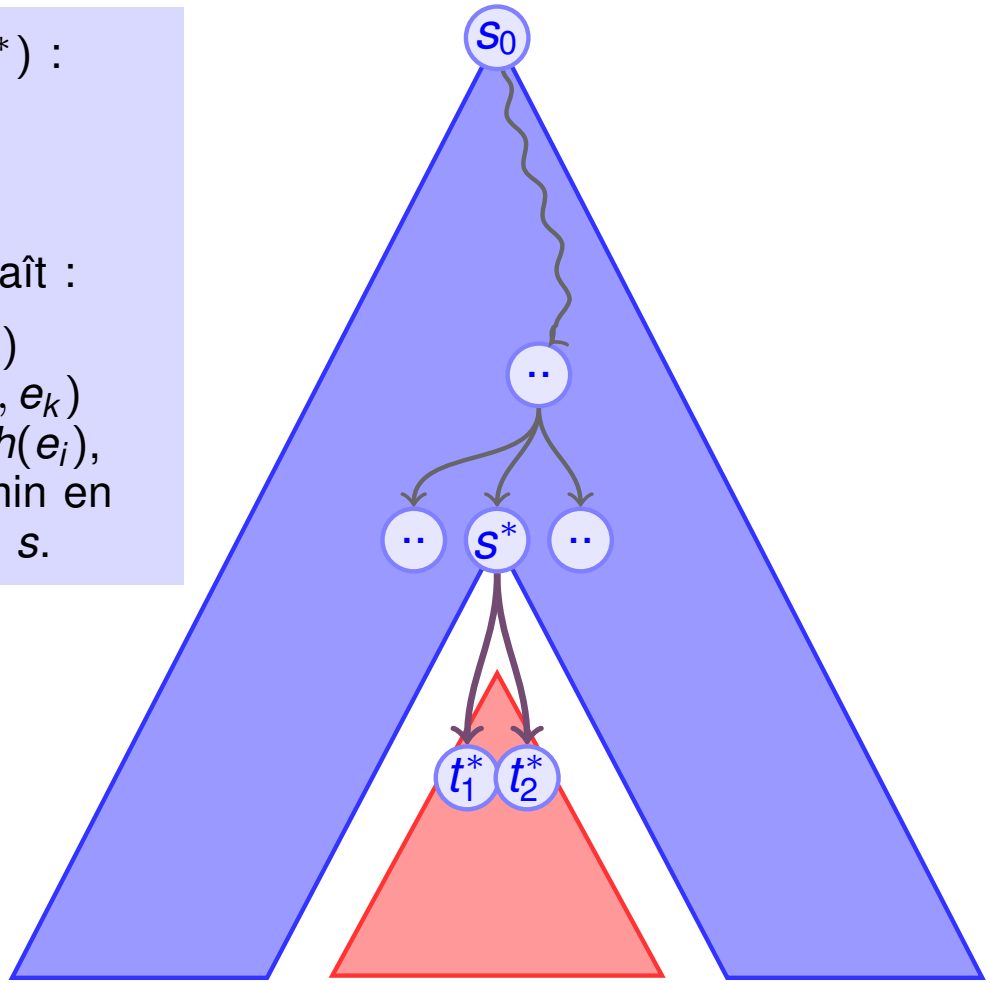
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$
- Arc distingué e^*
- Outside : Contexte où e^* apparaît :
  - Chemin $p = (s_0, s_1, \ldots, s^*)$ empruntant F-arcs $(e_1, \ldots, e_k)$



Décomposition

F-chemins empruntant $e^* = (s^* \to \mathbf{t}^*)$:

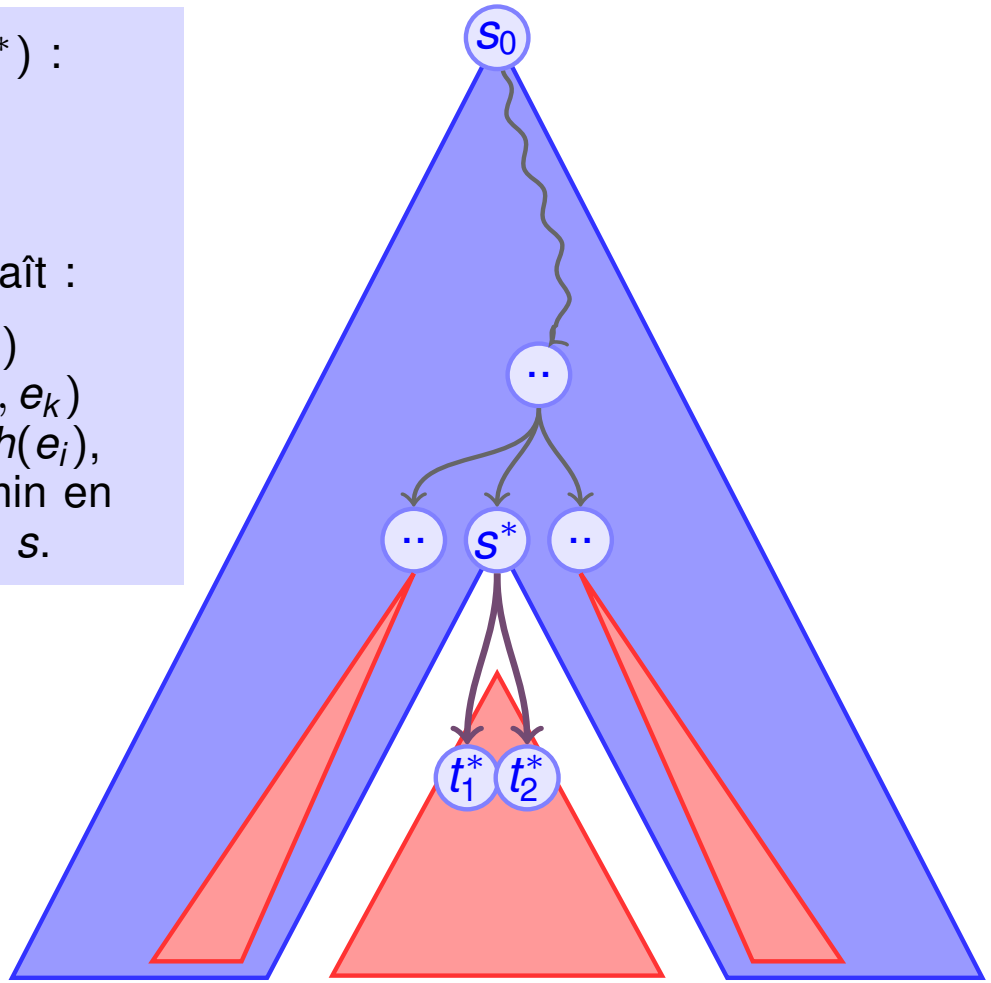
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$
- Arc distingué e^*
- Outside : Contexte où e^* apparaît :
  - Chemin $p = (s_0, s_1, \ldots, s^*)$ empruntant F-arcs $(e_1, \ldots, e_k)$
  - Pour chaque sommet $s \in h(e_i)$, $s \notin p$, compléter le F-chemin en considérant les cousins de s.



Décomposition

F-chemins empruntant $e^* = (s^* \to \mathbf{t}^*)$:

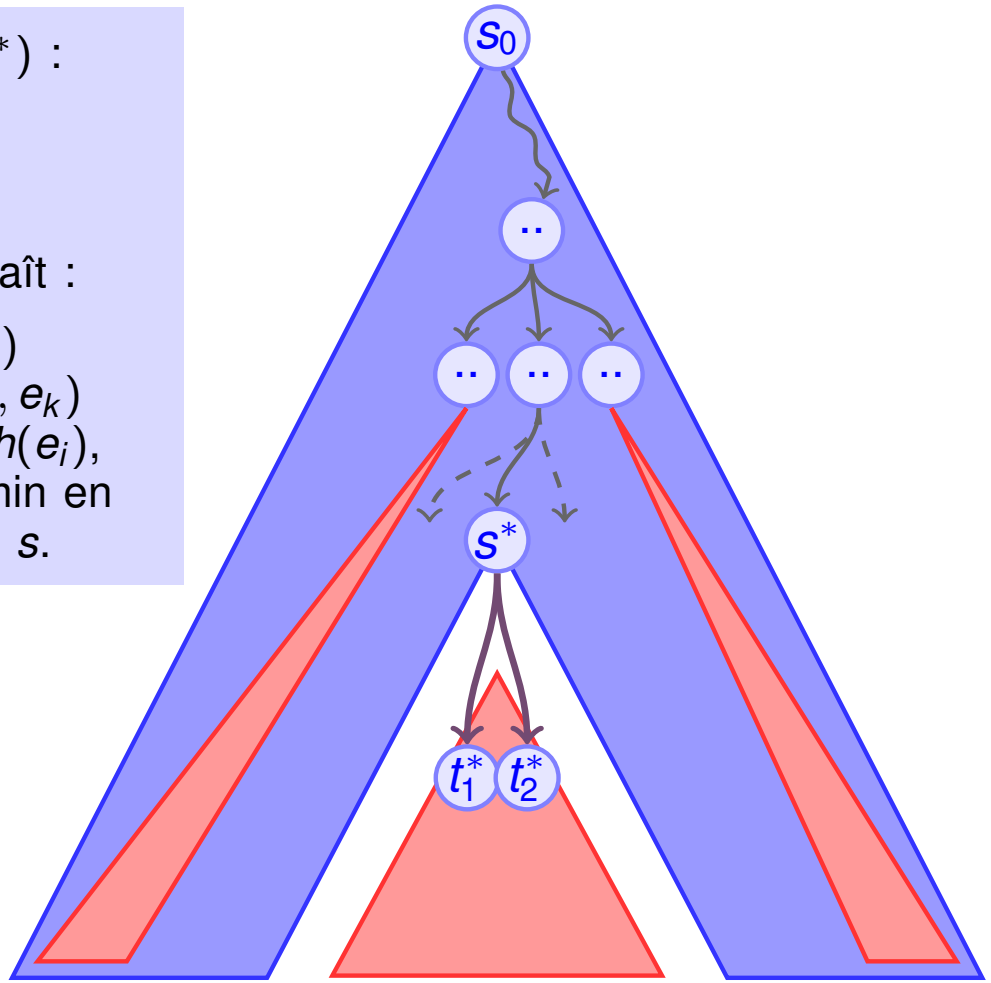
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$
- Arc distingué e^*
- Outside : Contexte où e^* apparaît :
  - Chemin $p = (s_0, s_1, \ldots, s^*)$ empruntant F-arcs $(e_1, \ldots, e_k)$
  - Pour chaque sommet $s \in h(e_i)$, $s \notin p$, compléter le F-chemin en considérant les cousins de s.



Décomposition

F-chemins empruntant $e^* = (s^* \to \mathbf{t}^*)$:

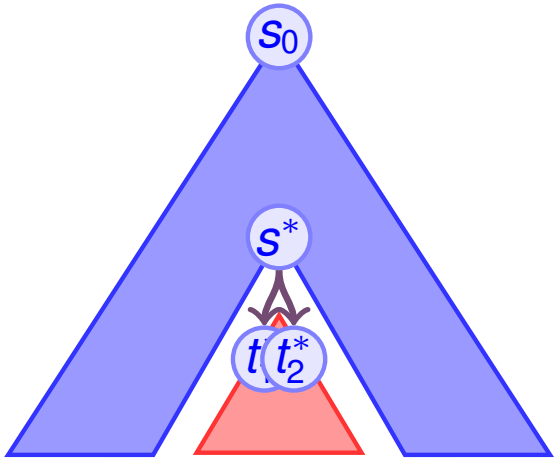
- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$
- Arc distingué e^*
- Outside : Contexte où e^* apparaît :
  - Chemin $p = (s_0, s_1, \ldots, s^*)$ empruntant F-arcs $(e_1, \ldots, e_k)$
  - Pour chaque sommet $s \in h(e_i)$, $s \notin p$, compléter le F-chemin en considérant les cousins de s.



Décomposition

F-chemins empruntant $e^* = (s^* \to \mathbf{t}^*)$:

- Inside : $|t^*|$-uplet issu de $\mathbf{t}^*$
- Arc distingué e^*
- Outside : Contexte où e^* apparaît :
  - Chemin $p = (s_0, s_1, \ldots, s^*)$ empruntant F-arcs $(e_1, \ldots, e_k)$
  - Pour chaque sommet $s \in h(e_i)$, $s \notin p$, compléter le F-chemin en considérant les cousins de s.



F-graphe acyclique indépendant $\Rightarrow$ Décomposition complète non-ambiguë

Récurrence sur la proba. totale p_{e^*} des F-chemins empruntant e^* :

$$p_{e^*} = \frac{b_{s^*} \cdot \pi(e) \cdot \prod_{s' \in t^*} w_{s'}}{w_{s_0}}$$

$$b_s = \mathbf{1}_{s=s_0} + \sum_{\substack{e'=(s' \to \mathbf{t}') \in E \\ \text{s. t. } s \in \mathbf{t}}} \pi(e') \cdot b_{s'} \cdot \prod_{\substack{s'' \in \mathbf{t}' \\ s'' \neq s}} w_{s''}, \quad \forall s \in V$$

Retour vers l'ARN

Soit une séquence d'ARN ω et une fonction d'énergie E, supposons :

- Un hypergraphe acyclique $\mathcal{H}$ t.q. F-chemins $\Leftrightarrow$ conformations (avec PK)
- Une fonction de valuation α : F-chemin $p \rightarrow$ énergie libre $E_{\omega,s}$ de s sur ω.

Retour vers l'ARN

Soit une séquence d'ARN ω et une fonction d'énergie E, supposons :

- Un hypergraphe acyclique $\mathcal{H}$ t.q. F-chemins $\Leftrightarrow$ conformations (avec PK)
- Une fonction de valuation α : F-chemin $p \rightarrow$ énergie libre $E_{\omega,s}$ de s sur ω.

Application		Algorithme Hypergraphe	Paramètres
Prédiction repliement	$\Leftrightarrow$	Score min.	$(\mathcal{H}, \alpha)$
Fonction de partition	$\Leftrightarrow$	Poids total	$(\mathcal{H}, e^{-\alpha/RT})$
Echantillonnage	$\Leftrightarrow$	Génération aléatoire	$(\mathcal{H}, e^{-\alpha/RT})$
Proba. paires de bases	$\Leftrightarrow$	Proba Arcs	$(\mathcal{H}, e^{-\alpha/RT})$

Retour vers l'ARN

Soit une séquence d'ARN ω et une fonction d'énergie E, supposons :

- Un hypergraphe acyclique $\mathcal{H}$ t.q. F-chemins $\Leftrightarrow$ conformations (avec PK)
- Une fonction de valuation α : F-chemin $p \rightarrow$ énergie libre $E_{\omega,s}$ de s sur ω.

Application		Algorithme Hypergraphe	Paramètres
Prédiction repliement	$\Leftrightarrow$	Score min.	$(\mathcal{H}, \alpha)$
Fonction de partition	$\Leftrightarrow$	Poids total	$(\mathcal{H}, e^{-\alpha/RT})$
Echantillonnage	$\Leftrightarrow$	Génération aléatoire	$(\mathcal{H}, e^{-\alpha/RT})$
Proba. paires de bases	$\Leftrightarrow$	Proba Arcs	$(\mathcal{H}, e^{-\alpha/RT})$

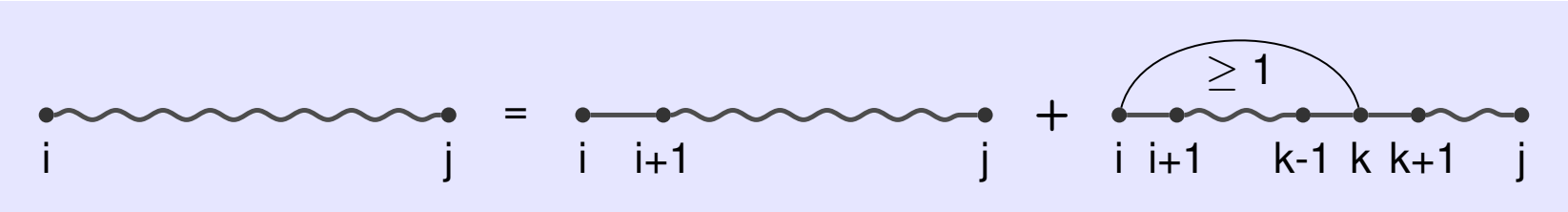
Message

Equations de programmation dynamique sont le produit d'une décomposition combinatoire ($\Rightarrow$ Famille d'hypergraphes).

Comment concevoir la bonne famille d'hypergraphes ?
À la main, mais la combinatoire peut aider ...

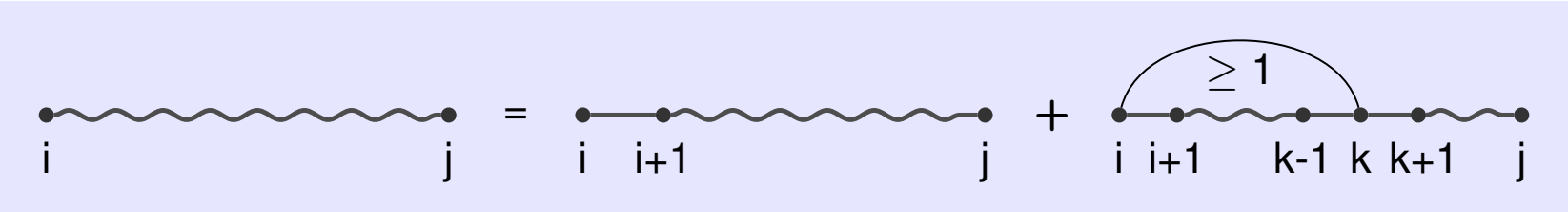
Exemple : Maximisation de paires de bases (Nussinov)

Décomposition :

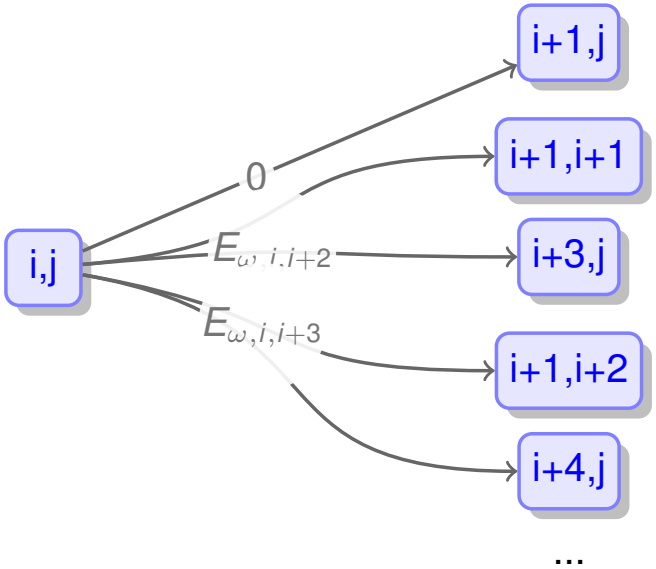


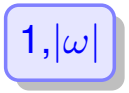
Exemple : Maximisation de paires de bases (Nussinov)

Décomposition :



En tant qu'hypergraphe : $\Theta(n^2)$ noeuds, $\Theta(n^3)$ arcs ($n = |\omega|$).





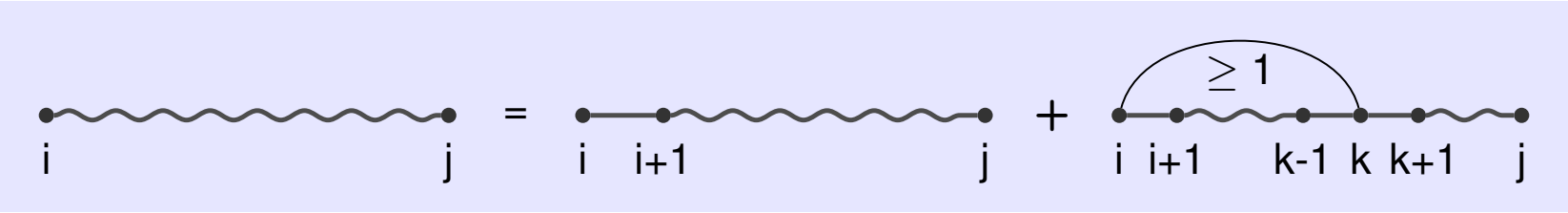
Noeud initial

Noeud terminal

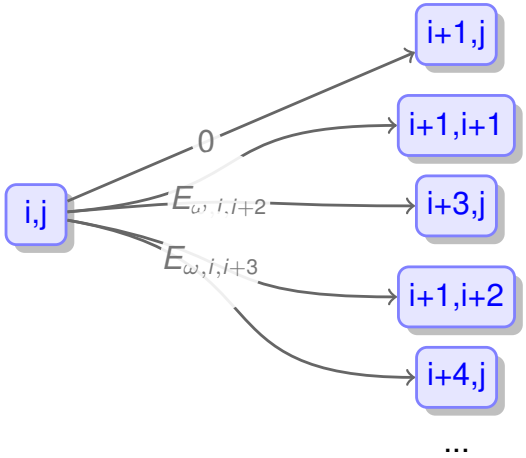
$$E_{\omega,i,j} = \begin{cases} -1 & \text{Si } \omega_i \text{ s'apparie avec } \omega_j \\ +\infty & \text{Sinon} \end{cases}$$

Fonction d'énergie

Décomposition :



En tant qu'hypergraphe : $\Theta(n^2)$ noeuds, $\Theta(n^3)$ arcs ($n = |\omega|$).



Noeud initial

Noeud terminal

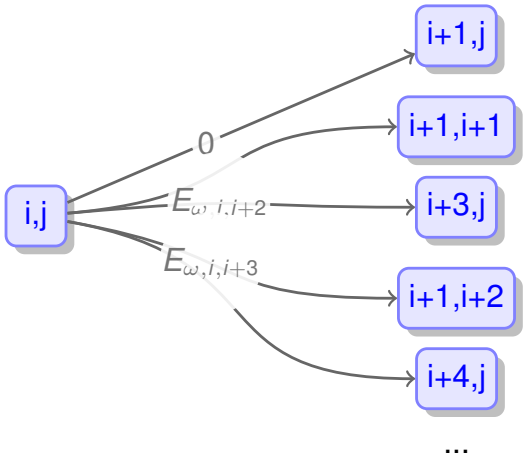
$$E_{\omega,i,j} = \begin{cases} -1 & \text{Si } \omega_i \text{ s'apparie avec } \omega_j \\ +\infty & \text{Sinon} \end{cases}$$

Fonction d'énergie

Remarque : Avant d'appliquer les algorithmes génériques, vérifier :

- F-chemins $\Leftrightarrow$ Structures secondaires $\rightarrow$ Séries génératrices
- Poids/Scores $\Leftrightarrow$ Energie libre

En tant qu'hypergraphe : $\Theta(n^2)$ noeuds, $\Theta(n^3)$ arcs ($n = |\omega|$).



Noeud initial

Noeud terminal

$$E_{\omega,i,j} = \begin{cases} -1 & \text{Si } \omega_i \text{ s'apparie avec } \omega_j \\ +\infty & \text{Sinon} \end{cases}$$

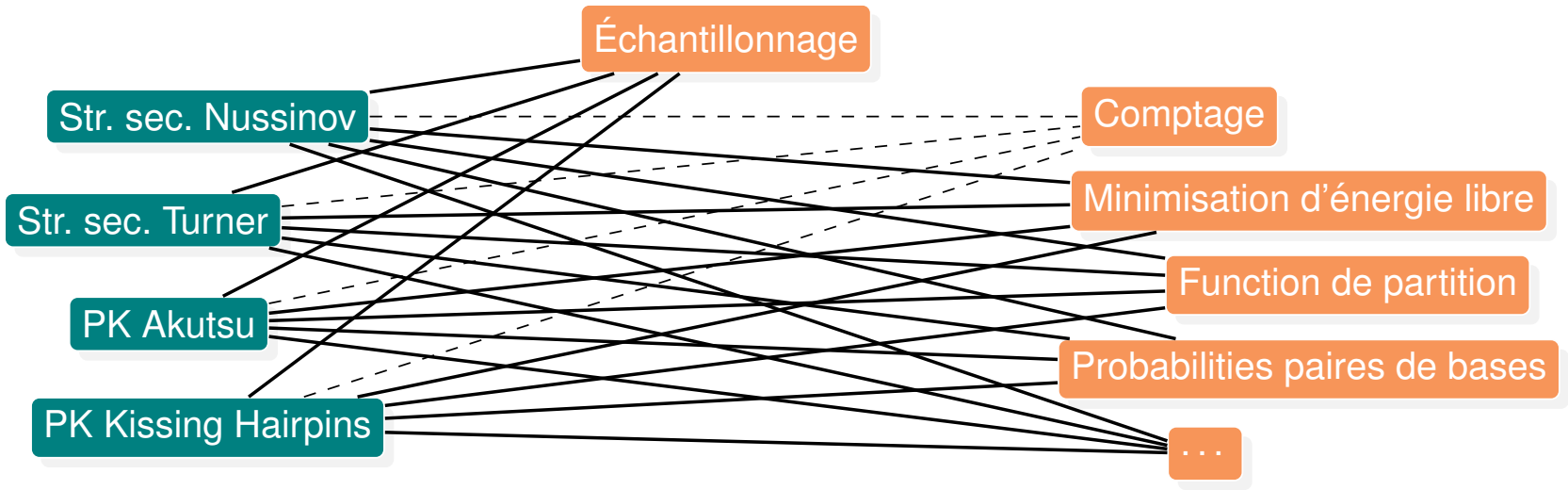
Fonction d'énergie

Application	Algorithme	Valuation	Temps/Mémoire
Minimisation d'énergie	Score minimal	E	$\mathcal{O}(n^3)/\mathcal{O}(n^2)$
Fonction de partition	Poids total	$e^{\frac{-E}{RT}}$	$\mathcal{O}(n^3)/\mathcal{O}(n^2)$
Proba. paires de bases	Proba. arcs	$e^{\frac{-E}{RT}}$	$\mathcal{O}(n^3)/\mathcal{O}(n^2)$
Échantillonnage (k str.)	Gén. aléatoire	$e^{\frac{-E}{RT}}$	$\mathcal{O}(n^3 + kn \log n)/\mathcal{O}(n^2)$

Faire tourner la presse à publis . . .

Message (cf ADP)

Les applications de la prog. dyn. peuvent (devraient) être détachées des équations, et être exprimées à un niveau abstrait (combinatoire).



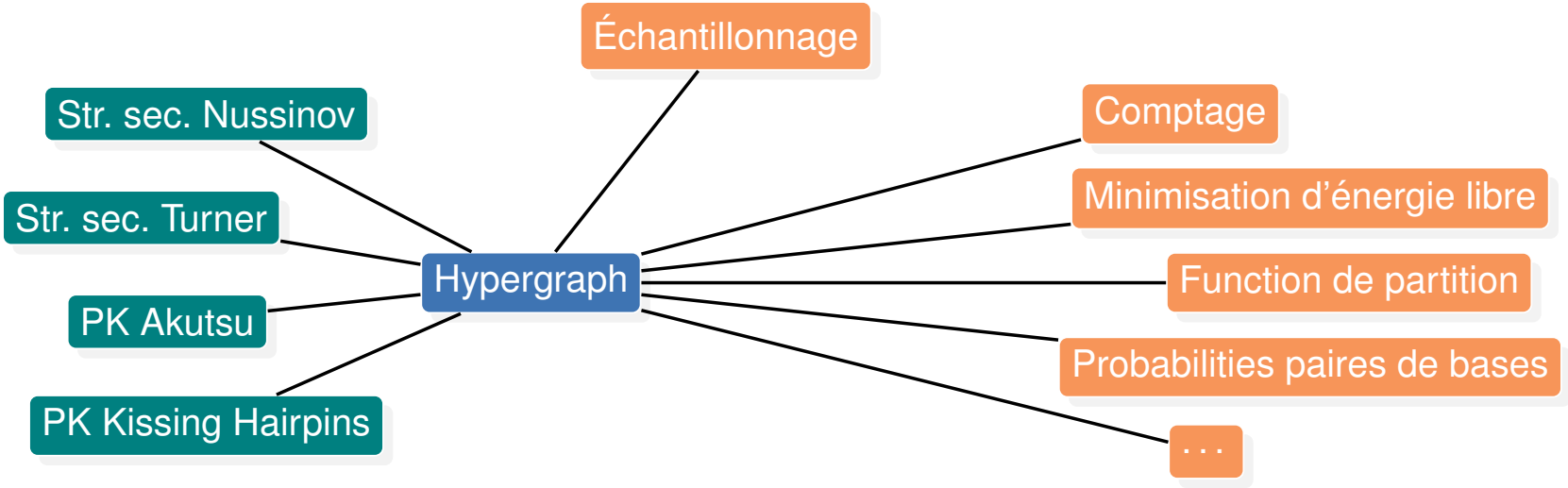
Credits [PS11] :

- Roytberg/Finkelstein : Prog. dyn. par hypergraphes en Bioinformatique ;
- L. Hwang : Prog. dyn. algébrique par hypergraphe ;
- R. Giegerich : Prog. dyn. algébrique par grammaires.

Faire tourner la presse à publis . . .

Message (cf ADP)

Les applications de la prog. dyn. peuvent (devraient) être détachées des équations, et être exprimées à un niveau abstrait (combinatoire).



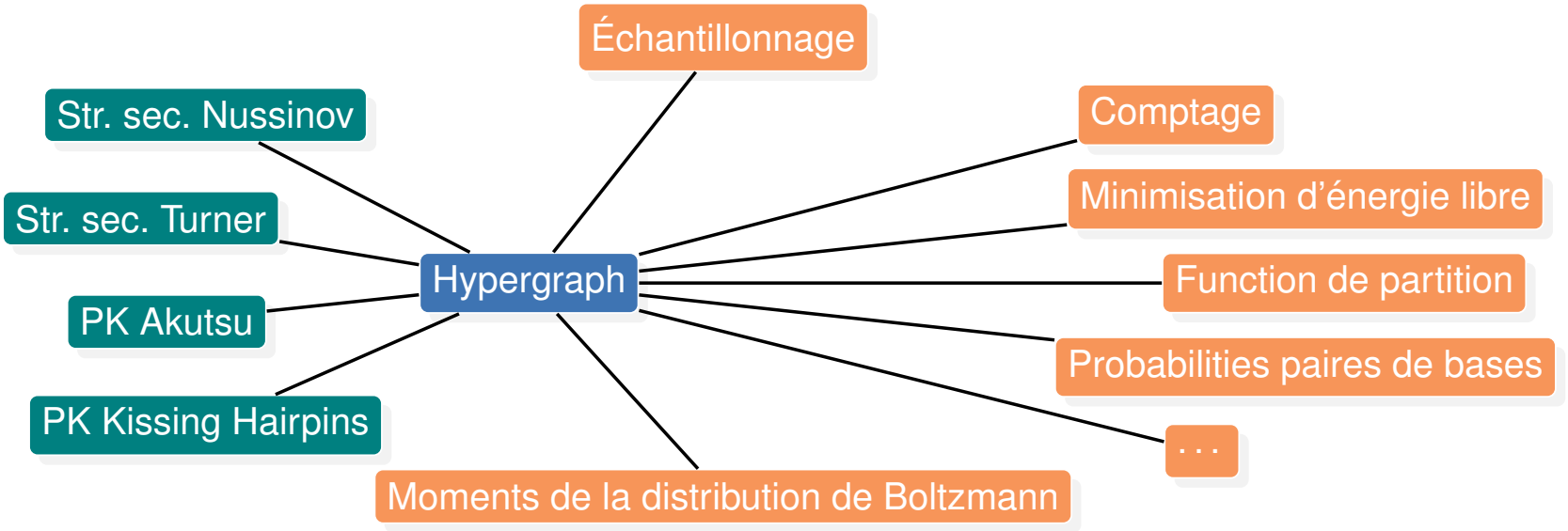
Credits [PS11] :

- Roytberg/Finkelstein : Prog. dyn. par hypergraphes en Bioinformatique ;
- L. Hwang : Prog. dyn. algébrique par hypergraphe ;
- R. Giegerich : Prog. dyn. algébrique par grammaires.

Faire tourner la presse à publis . . .

Message (cf ADP)

Les applications de la prog. dyn. peuvent (devraient) être détachées des équations, et être exprimées à un niveau abstrait (combinatoire).

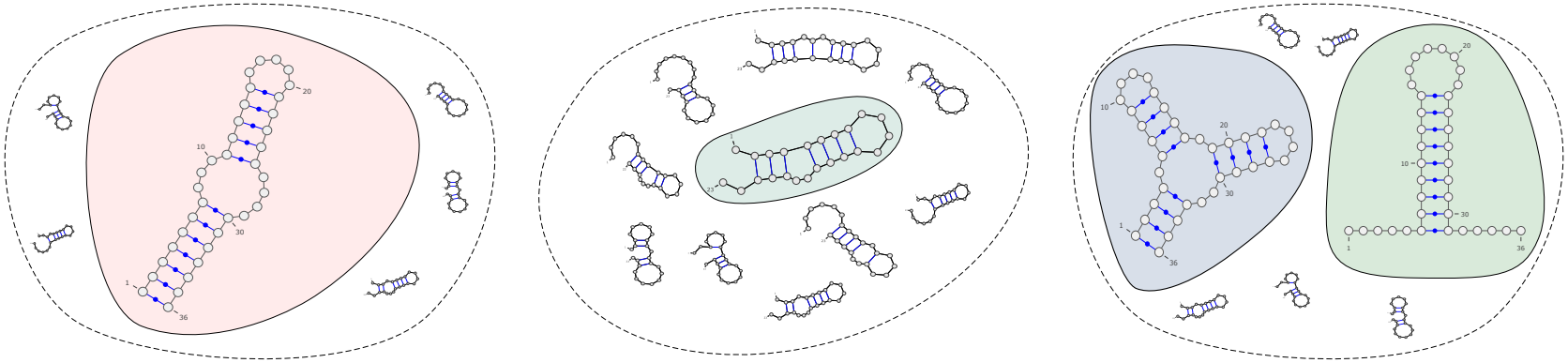


Credits [PS11] :

- Roytberg/Finkelstein : Prog. dyn. par hypergraphes en Bioinformatique ;
- L. Hwang : Prog. dyn. algébrique par hypergraphe ;
- R. Giegerich : Prog. dyn. algébrique par grammaires.

Résumé

1. Rappels de programmation dynamique
2. Repliement par minimisation d'énergie
   - Apparté thermodynamique
   - Modèle de Nussinov
   - Modèle de Turner
   - MFold/Unafold
3. Ensemble de Boltzmann
   - Définition
   - Calcul de la fonction de partition
   - Échantillonnage statistique
4. Programmation dynamique : Méthodes génériques
5. Algorithmique ensembliste avancée
   - Distribution exacte
   - Calcul des moments
6. Conclusion
   - Analyse syntaxique



Constat : Grande diversité des ensembles de Boltzmann pour des séquence différentes, associés à des rôles fonctionnels différents.

Comment extraire l'information fonctionnelle de l'ensemble de Boltzmann ?

Idée : Observer la distribution d'un paramètre structurel additif

- #Hélices ;
- #Paires de bases ;
- Energie libre ;
- ...

Distribution des paires de bases

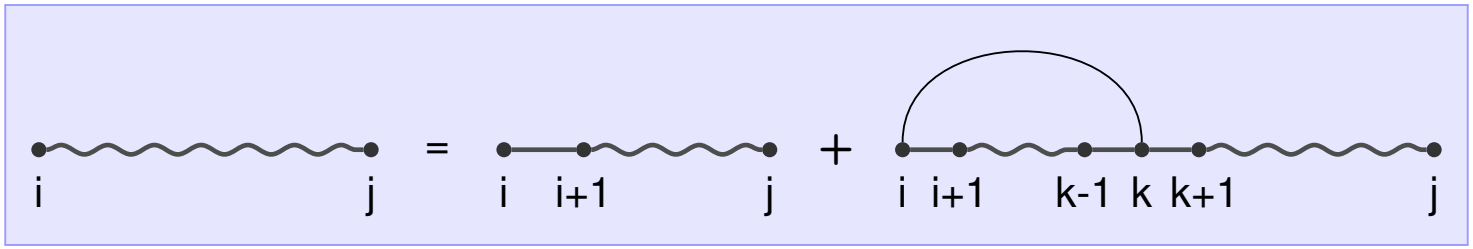
BP = Variable aléatoire #paires de bases dans une struct. sec., compatible avec ω, et engendrée selon la distribution uniforme (resp. Boltzmann).

Problème : Calculer la distribution de $BP \in [0, n]$, c'est à dire

$$\forall m \in [0, n] : \mathbb{P}(BP = m \mid \omega) \equiv \sum_{\substack{S \in \mathcal{S}_\omega \\ \#BP(S)=m}} \frac{1}{|\mathcal{S}_\omega|} = \frac{\boxed{|\{S \in \mathcal{S}_\omega \mid \#BP(S) = m\}|}}{|\mathcal{S}_\omega|}$$

$C^m_{i,j}$

Solution naïve : Adaptée du comptage



$$C_{i,t} = 1, \qquad \forall t \in [i, i+\theta]$$

$$C_{i,j} = \sum \begin{cases} C_{i+1,j} & (i \text{ non apparié}) \\ \displaystyle\sum_{k=i+\theta+1}^{j} 1 \times C_{i+1,k-1} \times C_{k+1,j} & (i \text{ comp. avec } k) \end{cases}$$

Distribution des paires de bases

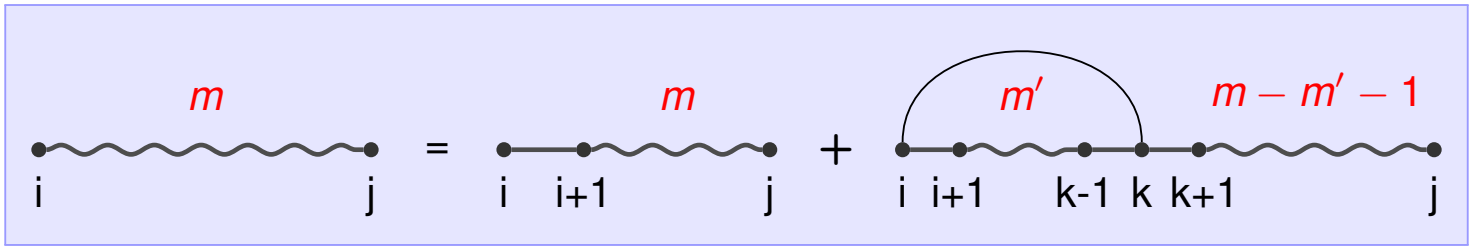
BP = Variable aléatoire #paires de bases dans une struct. sec., compatible avec ω, et engendrée selon la distribution uniforme (resp. Boltzmann).

Problème : Calculer la distribution de $BP \in [0, n]$, c'est à dire

$$\forall m \in [0, n] : \mathbb{P}(BP = m \mid \omega) \equiv \sum_{\substack{S \in \mathcal{S}_\omega \\ \#BP(S)=m}} \frac{1}{|\mathcal{S}_\omega|} = \frac{\boxed{|\{S \in \mathcal{S}_\omega \mid \#BP(S) = m\}|}}{|\mathcal{S}_\omega|}$$

$C^m_{i,j}$

Solution naïve : Adaptée du comptage $\rightarrow$ Temps : $\Theta(n^5)$, Mémoire : $\Theta(n^3)$



$$C^m_{i,t} = \mathbb{1}_{m=0}, \quad \forall t \in [i, i+\theta]$$

$$C^m_{i,j} = \sum \begin{cases} C^m_{i+1,j} & (i \text{ non apparié}) \\ \displaystyle\sum_{k=i+\theta+1}^{j} \sum_{m'=0}^{m-1} 1 \times C^{m'}_{i+1,k-1} \times C^{m-m'-1}_{k+1,j} & (i \text{ comp. avec } k) \end{cases}$$

Distribution des paires de bases

BP = Variable aléatoire #paires de bases dans une struct. sec., compatible avec ω, et engendrée selon la distribution uniforme (resp. Boltzmann).

Problème : Calculer la distribution de $BP \in [0, n]$, c'est à dire

$$\forall m \in [0, n] : \mathbb{P}(BP = m \mid \omega) \equiv \sum_{\substack{S \in \mathcal{S}_\omega \\ \#BP(S) = m}} \frac{1}{|\mathcal{S}_\omega|} = \frac{\boxed{|\{S \in \mathcal{S}_\omega \mid \#BP(S) = m\}|}}{|\mathcal{S}_\omega|}$$

$C_{i,j}^m$

Solution naïve : Adaptée du comptage $\rightarrow$ Temps : $\Theta(n^5)$, Mémoire : $\Theta(n^3)$

Par interpolation [WP11] :

- Considérer les polynomes $C_{i,j}(z) = \sum_{m=0}^{n} C_{i,j}^m \cdot z^m$
- Evaluation possible en $\Theta(n^3)/\Theta(n^2)$ pour $z \in \mathbb{R}^+$ donné, via :

$$C_{i,j}(z) = \sum \begin{cases} C_{i+1,j}(z) & (i \text{ non apparié}) \\ \sum_{k=i+\theta+1}^{j} z \cdot C_{i+1,k-1}(z) \cdot C_{k+1,j}(z) & (i \text{ comp. avec } k) \end{cases}$$

Distribution des paires de bases

BP = Variable aléatoire #paires de bases dans une struct. sec., compatible avec ω, et engendrée selon la distribution uniforme (resp. Boltzmann).

Problème : Calculer la distribution de $BP \in [0, n]$, c'est à dire

$$\forall m \in [0, n] : \mathbb{P}(BP = m \mid \omega) \equiv \sum_{\substack{S \in \mathcal{S}_\omega \\ \#BP(S) = m}} \frac{1}{|\mathcal{S}_\omega|} = \frac{\boxed{|\{S \in \mathcal{S}_\omega \mid \#BP(S) = m\}|}}{|\mathcal{S}_\omega|}$$

$C_{i,j}^m$

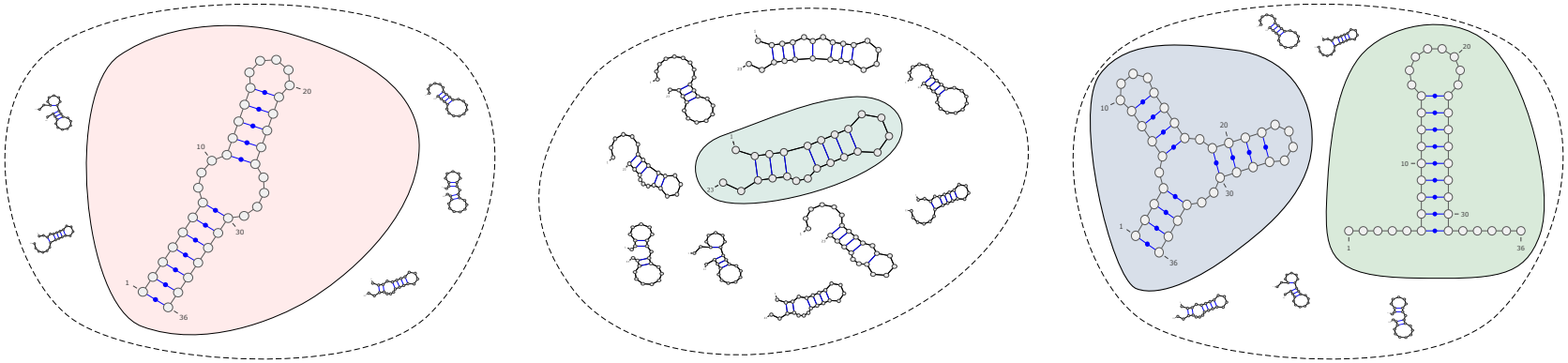
Solution naïve : Adaptée du comptage $\rightarrow$ Temps : $\Theta(n^5)$, Mémoire : $\Theta(n^3)$

Par interpolation [WP11] : $\rightarrow$ Temps : $\Theta(n^4)$, Mémoire : $\Theta(n^2)$

- Considérer les polynomes $C_{i,j}(z) = \sum_{m=0}^{n} C_{i,j}^m \cdot z^m$
- Evaluation possible en $\Theta(n^3)/\Theta(n^2)$ pour $z \in \mathbb{R}^+$ donné, via :

$$C_{i,j}(z) = \sum \begin{cases} C_{i+1,j}(z) & (i \text{ non apparié}) \\ \sum_{k=i+\theta+1}^{j} z \cdot C_{i+1,k-1}(z) \cdot C_{k+1,j}(z) & (i \text{ comp. avec } k) \end{cases}$$

- Evaluer le polynôme $C_{1,n}(z)$ en $n+1$ valeurs distinctes $(z_1, z_2, \cdots, z_{n+1})$
- Interpoler les coeff. de $C_{1,n}(z) \rightarrow C_{i,j}^m$ $\rightarrow$ FFT [SSD+12] : $\Theta(n \log(n))$



Constat : Grande diversité des ensembles de Boltzmann pour des séquence différentes, associés à des rôles fonctionnels différents.

Comment extraire l'information fonctionnelle de l'ensemble de Boltzmann ?

Idée : Observer la distribution d'un paramètre structurel additif

Mais : Calcul exact coûteux + Moyenne, écart-type . . . suffisent souvent
. . . Comment calculer efficacement les moments de la distribution ?

Extraction des moments

Problème : Étant donné un ARN ω calculer le 1er, ..., $p^{\text{ième}}$ moment de BP :

$$\mathbb{E}(BP^p \mid \omega) = \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S)^p = \frac{\boxed{\sum_{S \in \mathcal{S}_\omega} \#BP(S)^p}}{|\mathcal{S}_\omega|} \longrightarrow \dot{C}_{i,j}$$

Extraction des moments

Problème : Étant donné un ARN ω calculer le 1er, ..., $p^{\text{ième}}$ moment de BP :

$$\mathbb{E}(BP^p \mid \omega) = \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S)^p = \frac{\boxed{\sum_{S \in \mathcal{S}_\omega} \#BP(S)^p}}{|\mathcal{S}_\omega|}$$

$\dot{C}_{i,j}$

Pourquoi ?

- 1er moment : Nombre moyen de paires de bases $\mu := \mathbb{E}(BP)$
- 2ème moment : Permet calcul de variance/écart-type

$$\sigma = \sum_{S \in \mathcal{S}_\omega} p_S \cdot (\#BP(S) - \mu)^2 = \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S)^2 - 2 \cdot \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S) \cdot \mu + \mu^2 = \mathbb{E}(BP^2) - \mu^2$$

Extraction des moments

Problème : Étant donné un ARN ω calculer le 1er, . . . , $p^{\text{ième}}$ moment de BP :

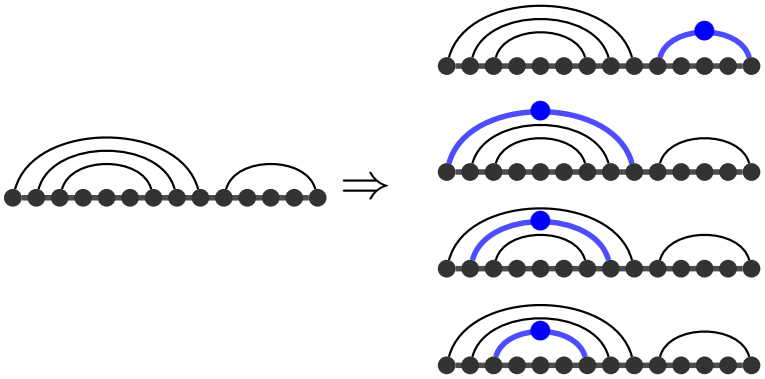
$$\mathbb{E}(BP^p \mid \omega) = \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S)^p = \frac{\boxed{\sum_{S \in \mathcal{S}_\omega} \#BP(S)^p}}{|\mathcal{S}_\omega|} \longrightarrow \dot{C}_{i,j}$$

Pourquoi ?

- 1er moment : Nombre moyen de paires de bases $\mu := \mathbb{E}(BP)$
- 2ème moment : Permet calcul de variance/écart-type

$$\sigma = \sum_{S \in \mathcal{S}_\omega} p_S \cdot (\#BP(S) - \mu)^2 = \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S)^2 - 2 \cdot \sum_{S \in \mathcal{S}_\omega} p_S \cdot \#BP(S) \cdot \mu + \mu^2 = \mathbb{E}(BP^2) - \mu^2$$

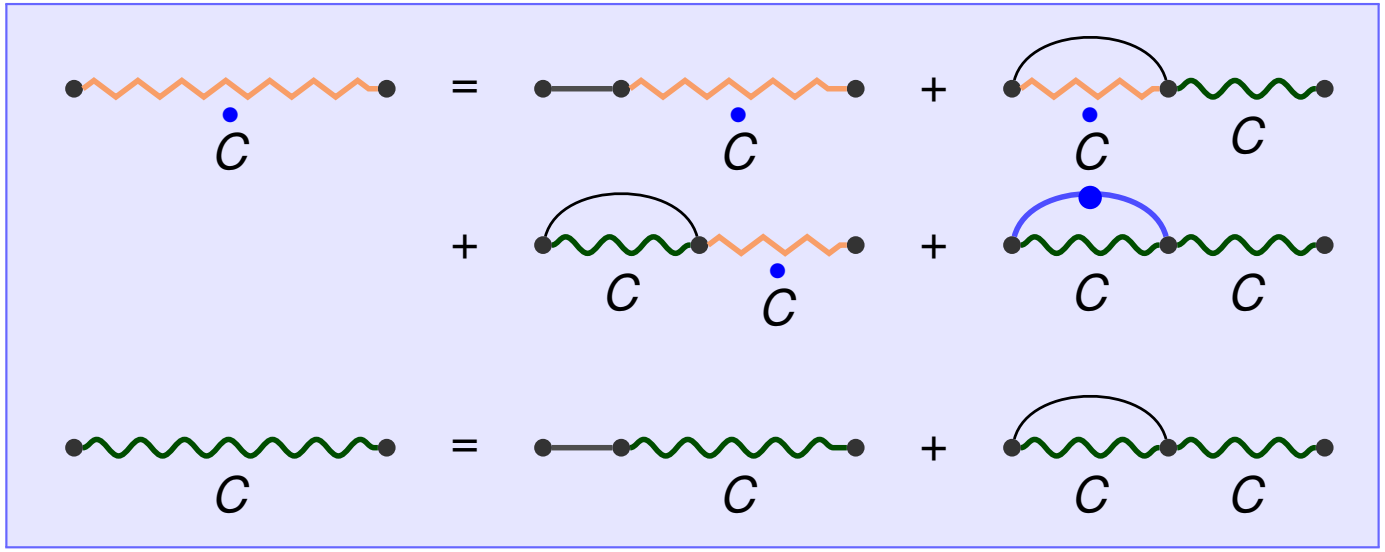
Idée [PS11] : Transformer/dupliquer la décomposition de façon à engendrer toute struct. sec. avec multiplicité égale à son nombre de paires de bases.



Compter dans cette décomposition

$\Leftrightarrow$ Calculer $\dot{C}_{i,j}$

Calcul du premier moment $\rightarrow$ Temps : $\Theta(n^3)$, Mémoire : $\Theta(n^2)$



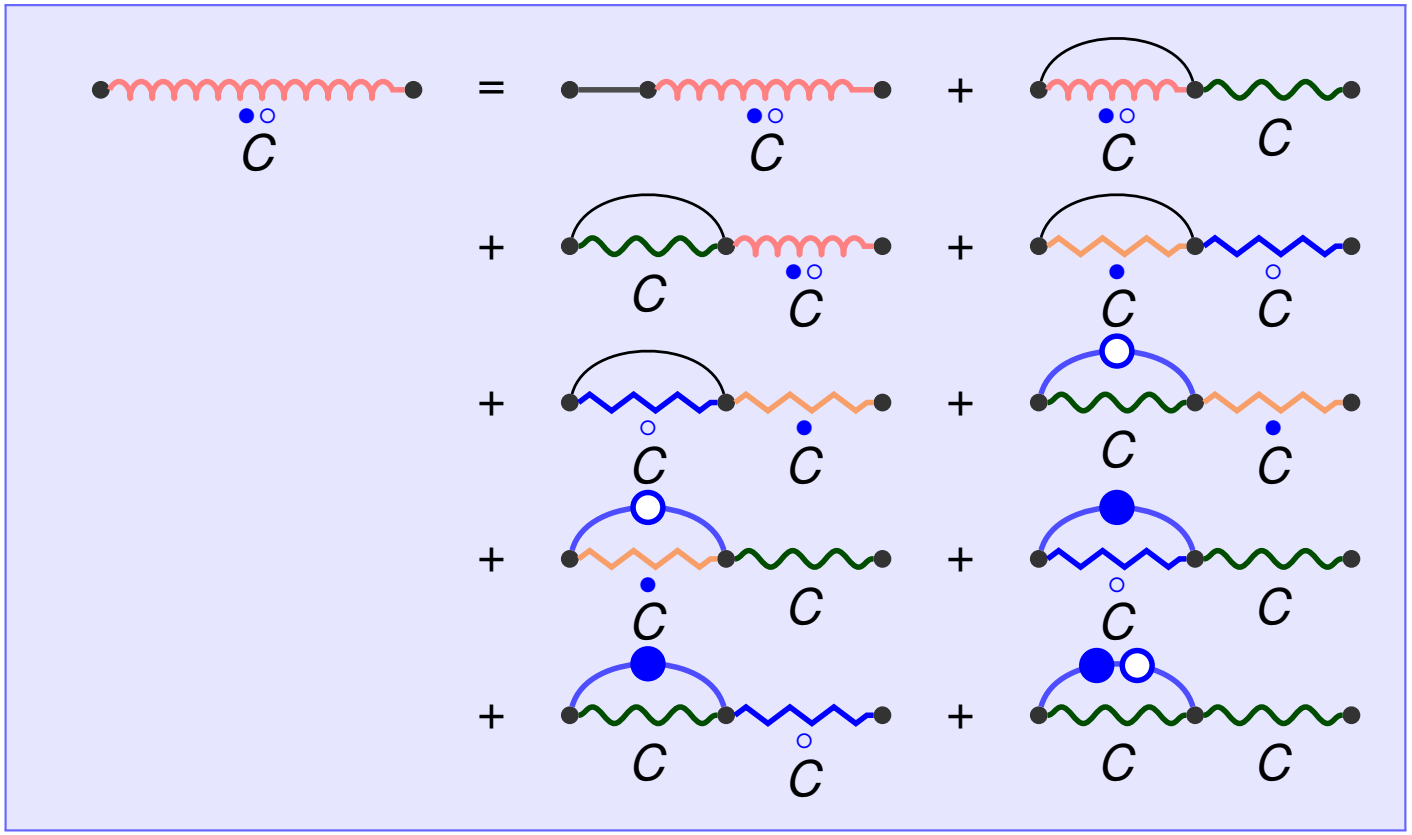
Equation de prog. dyn. :

$$\dot{C}_{i,j} = \dot{C}_{i+1,j} + \sum_{k \in i+\theta+1}^{j} \left(\begin{array}{c} \dot{C}_{i+1,k-1} \cdot \dot{C}_{k+1,j} \\ + C_{i+1,k-1} \cdot \dot{C}_{k+1,j} \\ + \dot{C}_{i+1,k-1} \cdot C_{k+1,j} \end{array} \right)$$

Exercice : Deuxième moment ?

Calcul du premier moment $\rightarrow$ Temps : $\Theta(n^3)$, Mémoire : $\Theta(n^2)$

Calcul du deuxième moment $\rightarrow$ Temps : $\Theta(n^3)$, Mémoire : $\Theta(n^2)$



Ordres supérieurs : Même complexité, mais les constantes explosent !

Des optimisations génériques

Ces principes se généralisent sans difficulté :

- Distribution de Boltzmann (+ Facteur $\mathcal{B}$; fonc. part. $\mathcal{Z}$ au dénom.) ;

Des optimisations génériques

Ces principes se généralisent sans difficulté :

- Distribution de Boltzmann (+ Facteur $\mathcal{B}$; fonc. part. $\mathcal{Z}$ au dénom.) ;
- Toute valuation additive (Attention au #Vals distinctes en distr. exacte) ;

Des optimisations génériques

Ces principes se généralisent sans difficulté :

- Distribution de Boltzmann (+ Facteur $\mathcal{B}$; fonc. part. $\mathcal{Z}$ au dénom.) ;
- Toute valuation additive (Attention au #Vals distinctes en distr. exacte) ;
- Distributions jointes sur plusieurs paramètres (e.g. #Paires vs #Hélices) ;

Moments généraux : $\mathbb{E}(\alpha_1^{m_1}\alpha_2^{m_2}\cdots\alpha_k^{m_k}) = \sum_{p\in\mathcal{P}_s}\frac{\pi(p)}{w_s}\prod_{i=1}^{k}\alpha_i(p)^{m_i} = c_s^{\mathbf{m}}/w_s$

Des optimisations génériques

Ces principes se généralisent sans difficulté :

- Distribution de Boltzmann (+ Facteur $\mathcal{B}$; fonc. part. $\mathcal{Z}$ au dénom.) ;
- Toute valuation additive (Attention au #Vals distinctes en distr. exacte) ;
- Distributions jointes sur plusieurs paramètres (e.g. #Paires vs #Hélices) ;
- Tout schéma non-ambigu de programmation dynamique.
  Ex. : Analyser la diversité des solutions quasi-optimales en optimisation.

Moments généraux : $\mathbb{E}(\alpha_1^{m_1}\alpha_2^{m_2}\cdots\alpha_k^{m_k}) = \sum_{p\in\mathcal{P}_s} \frac{\pi(p)}{w_s} \prod_{i=1}^{k} \alpha_i(p)^{m_i} = c_s^{\mathbf{m}}/w_s$

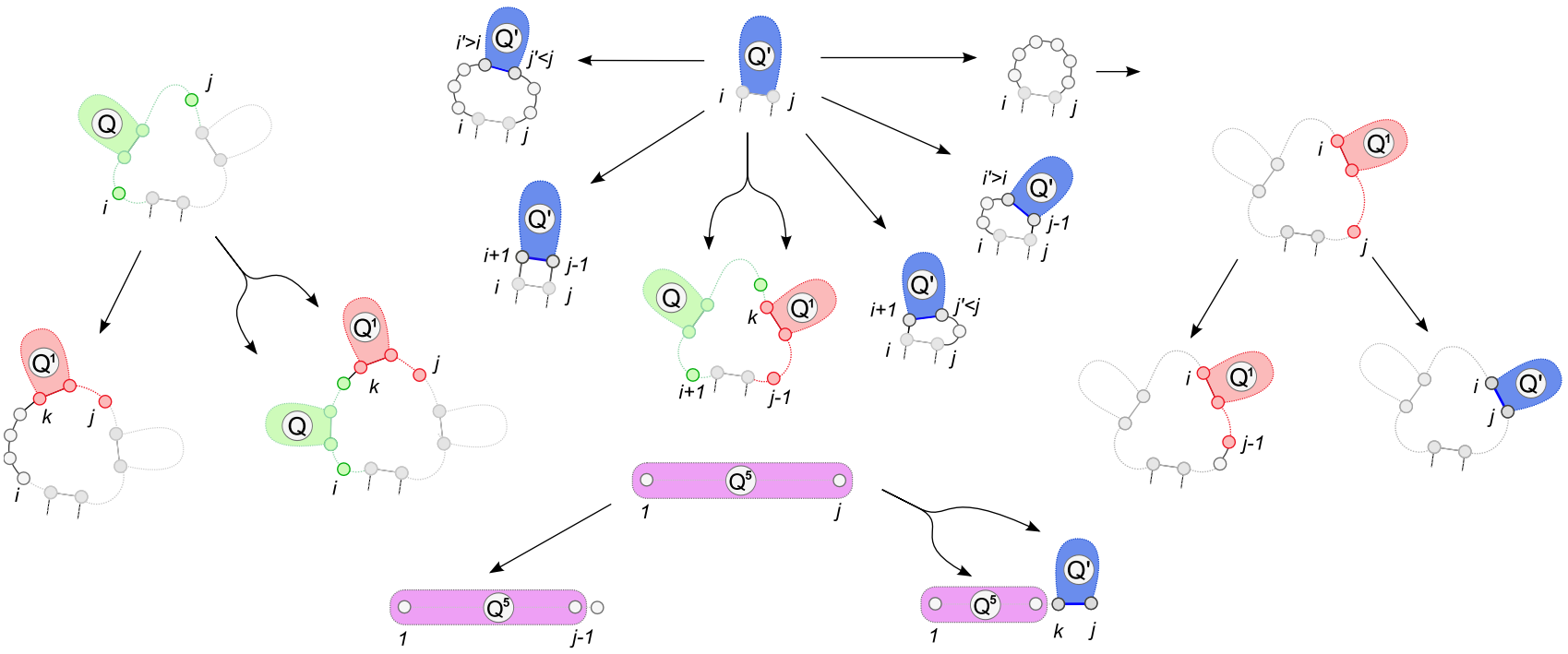
Théorème (Récurrence de calcul des $c_s^{\mathbf{m}}$ [PS11])

$$c_s^{\mathbf{m}} = \sum_{e=(s\to\mathbf{t})} \pi(e) \cdot \sum_{\substack{\mathbf{m}',\left(\mathbf{m}''_1,\cdots,\mathbf{m}''_{|t|}\right)\\ s.\,t.\ \mathbf{m}'+\sum_j \mathbf{m}''_j=\mathbf{m}}} \prod_{i=1}^{k} \binom{m_i}{m'_i, m''_{1,i}, \cdots, m''_{|t|,i}} \cdot \alpha_i(e)^{m'_i} \cdot \prod_{i=1}^{|t|} c_{t_i}^{\mathbf{m}''_i}$$

Temps : $\mathcal{O}\left((|E|+|V|)\cdot k\cdot t^+\cdot \prod_{i=1}^{k} m_i^{t^++1}\right)$ (t^+ = degré sortant max)

Mémoire : $\Theta\left(|V|\cdot\prod_{i=1}^{k} m_i\right)$

Décomposition Mfold/Unafold (modèle de Turner) [PS11]



Application	Algorithm	Weight fun.	Time/Space	Ref.
Energy minimization	Minimal weight	$\pi_{\mathcal{T}}$	$\mathcal{O}(n^{3(4)})/\mathcal{O}(n^2)$	[ZS81]
Partition function	Weighted count	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(n^{3(4)})/\mathcal{O}(n^2)$	[McC90]
Base-pairing probabilities	Arc-traversal prob.	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(n^{3(4)})/\mathcal{O}(n^2)$	[McC90]
Statistical sampling (k-samples)	Random gen.	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(n^{3(4)} + kn \log n)/\mathcal{O}(n^2)$	[DL03, Pon08]
Moments of energy (Mean, Var.)	Moments extraction	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(n^{3(4)})/\mathcal{O}(n^2)$	[MMN05]
m-th moment of additive features	Moments extraction	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(m^3 \cdot n^{3(4)})/\mathcal{O}(m \cdot n^2)$	–
Correlations of additive features	Moments extraction	$e^{\frac{-\pi_{\mathcal{T}}}{RT}}$	$\mathcal{O}(n^{3(4)})/\mathcal{O}(n^2)$	–

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation

2.

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation
- La programmation dynamique y règne sans partage, et suscite des développements génériques et transposables

2.

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation
- La programmation dynamique y règne sans partage, et suscite des développements génériques et transposables
- Au coeur de la démarche de conception en prog. dyn., l'objet crucial est la décomposition, et l'équation de récurrence n'est qu'un produit dérivé

2.

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation
- La programmation dynamique y règne sans partage, et suscite des développements génériques et transposables
- Au coeur de la démarche de conception en prog. dyn., l'objet crucial est la décomposition, et l'équation de récurrence n'est qu'un produit dérivé
- Encore des problèmes difficiles/ouverts, à la lisière de la prog. dyn. : Prédiction avec pseudonoeuds, Design d'ARN, Cinétique des ARN

2.

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation
- La programmation dynamique y règne sans partage, et suscite des développements génériques et transposables
- Au coeur de la démarche de conception en prog. dyn., l'objet crucial est la décomposition, et l'équation de récurrence n'est qu'un produit dérivé
- Encore des problèmes difficiles/ouverts, à la lisière de la prog. dyn. : Prédiction avec pseudonoeuds, Design d'ARN, Cinétique des ARN
- Nombreuses hypothèses testables sur la relation structure/fonction, et au delà de l'ARN : Réconciliations, alignement, sac-à-dos . . . (mais ~~jamais~~ rarement plus de 24h dans une journée)

2.

Conclusion

- L'algorithmique du repliement des ARN soulève des problèmes à la frontière entre combinatoire et optimisation
- La programmation dynamique y règne sans partage, et suscite des développements génériques et transposables
- Au coeur de la démarche de conception en prog. dyn., l'objet crucial est la décomposition, et l'équation de récurrence n'est qu'un produit dérivé
- Encore des problèmes difficiles/ouverts, à la lisière de la prog. dyn. : Prédiction avec pseudonoeuds, Design d'ARN, Cinétique des ARN
- Nombreuses hypothèses testables sur la relation structure/fonction, et au delà de l'ARN : Réconciliations, alignement, sac-à-dos . . . (mais ~~jamais~~ rarement plus de 24h dans une journée)

Merci de votre attention ![2]

2. Réveillez vous ! Tous à la plage !

References I

J. K. Baker.
Trainable grammars for speech recognition.
The Journal of the Acoustical Society of America, 65(S1) :S132–S132, 1979.

Y. Ding, C. Y. Chan, and C. E. Lawrence.
RNA secondary structure prediction by centroids in a boltzmann weighted ensemble.
RNA, 11 :1157–1166, 2005.

Y. Ding and E. Lawrence.
A statistical sampling algorithm for RNA secondary structure prediction.
Nucleic Acids Research, 31(24) :7280–7301, 2003.

Alain Denise, Yann Ponty, and Michel Termier.
Controlled non-uniform random generation of decomposable structures.
Theoretical Computer Science, 411(40) :3527–3552, 2010.

A. V. Finkelstein and M. A. Roytberg.
Computation of biopolymers : a general approach to different problems.
Biosystems, 30(1-3) :1–19, 1993.

J.S. McCaskill.
The equilibrium partition function and base pair binding probabilities for RNA secondary structure.
Biopolymers, 29 :1105–1119, 1990.

István Miklós, Irmtraud M Meyer, and Borbála Nagy.
Moments of the boltzmann distribution for rna secondary structures.
Bull Math Biol, 67(5) :1031–1047, Sep 2005.

Ján Maňuch, Chris Thachuk, Ladislav Stacho, and Anne Condon.
Np-completeness of the direct energy barrier problem without pseudoknots.
Natural Computing : an international journal, 10 :391–405, 2011.

References II

Nicholas R Markham and Michael Zuker.

Unafold : software for nucleic acid folding and hybridization.
Methods Mol Biol, 453 :3–31, 2008.

R. Nussinov and A. B. Jacobson.

Fast algorithm for predicting the secondary structure of single stranded RNA.
Proc. Natl. Acad. Sci. U. S. A., 77(11) :6309–6313, 1980.

S.B. Needleman and C.D. Wunsch.

A general method applicable to the search for similarities in the amino acid sequence of two proteins.
J. Mol. Bio., 48 :443–453, 1970.

Y. Ponty.

Efficient sampling of RNA secondary structures from the boltzmann ensemble of low-energy : The boustrophedon method.
J Math Biol, 56(1-2) :107–127, Jan 2008.

Yann Ponty and Cédric Saule.

A combinatorial framework for designing (pseudoknotted) rna algorithms.
In TeresaM. Przytycka and Marie-France Sagot, editors, *Algorithms in Bioinformatics*, volume 6833 of *Lecture Notes in Computer Science*, pages 250–269. Springer Berlin Heidelberg, 2011.

Georg Sauthoff, Mathias Möhl, Stefan Janssen, and Robert Giegerich.

Bellman's gap—a language and compiler for dynamic programming in sequence analysis.
Bioinformatics, 29(5) :551–560, 2013.

Evan Senter, Saad Sheikh, Ivan Dotu, Yann Ponty, and Peter Clote.

Using the fast fourier transform to accelerate the computational search for rna conformational switches.
PLoS ONE, 7(12) :e50506, 12 2012.

Lioudmila V Sharova, Alexei A Sharov, Timur Nedorezov, Yulan Piao, Nabeebi Shaik, and Minoru S H Ko.

Database for mrna half-life of 19 977 genes obtained by dna microarray analysis of pluripotent and differentiating mouse embryonic stem cells.
DNA Res, 16(1) :45–58, Feb 2009.

References III

T. F. Smith and M. S. Waterman.
Identification of common molecular subsequences.
J Mol Biol, 147 :195–197, 1981.

M. S. Waterman.
Secondary structure of single stranded nucleic acids.
Advances in Mathematics Supplementary Studies, 1(1) :167–212, 1978.

Jérôme Waldispühl and Yann Ponty.
An unbiased adaptive sampling algorithm for the exploration of rna mutational landscapes under evolutionary pressure.
Journal of Computational Biology, 18(11) :1465–1479, 2011.

M. Zuker and P. Stiegler.
Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information.
Nucleic Acids Res, 9 :133–148, 1981.

Contenu indicible

Deux grandes familles de formalismes

- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

- Approche Hypergraphe :
  Espace de recherche = (hyper-)chemins d'un hypergraphe.

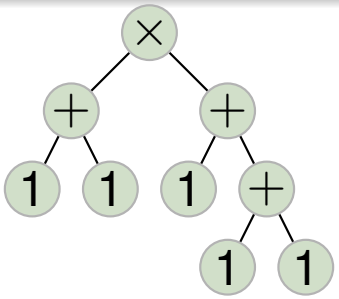
Deux grandes familles de formalismes

- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

  Exemple 1 (Caravane d'El Mamoun) :
  Grammaire $= \{S \rightarrow S + S \mid S \times S \mid 1\}$
  Quelle(s) valeur(s) pour une expression arithmétique sans parenthésage :

$$1 + 1 \times 1 + 1 + 1 = \ ?$$

Deux grandes familles de formalismes

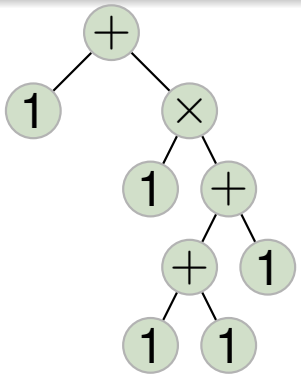


- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

  Exemple 1 (Caravane d'El Mamoun) :
  Grammaire $= \{S \to S + S \mid S \times S \mid 1\}$
  Quelle(s) valeur(s) pour une expression arithmétique sans parenthésage :

$$1 + 1 \times 1 + 1 + 1 = \ = (1 + 1) \times (1 + (1 + 1)) = 6\,?$$

Deux grandes familles de formalismes

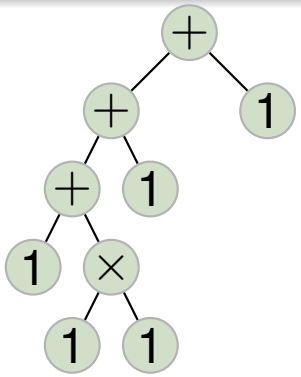


- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

Exemple 1 (Caravane d'El Mamoun) :
Grammaire $= \{S \to S + S \mid S \times S \mid 1\}$
Quelle(s) valeur(s) pour une expression arithmétique sans parenthésage :

$$1 + 1 \times 1 + 1 + 1 = \ = 1 + (1 \times ((1 + 1) + 1)) = 4\,?$$

Deux grandes familles de formalismes



- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

  Exemple 1 (Caravane d'El Mamoun) :
  Grammaire $= \{S \to S + S \mid S \times S \mid 1\}$
  Quelle(s) valeur(s) pour une expression arithmétique sans parenthésage :
  $$1 + 1 \times 1 + 1 + 1 = \ = ((1 + (1 \times 1)) + 1) + 1 = 4\,?$$
  En général : Se demander comment parenthéser l'expression de façon à min (resp.max)-imiser la valeur de l'expression est un problème d'analyse syntaxique (pondéré).

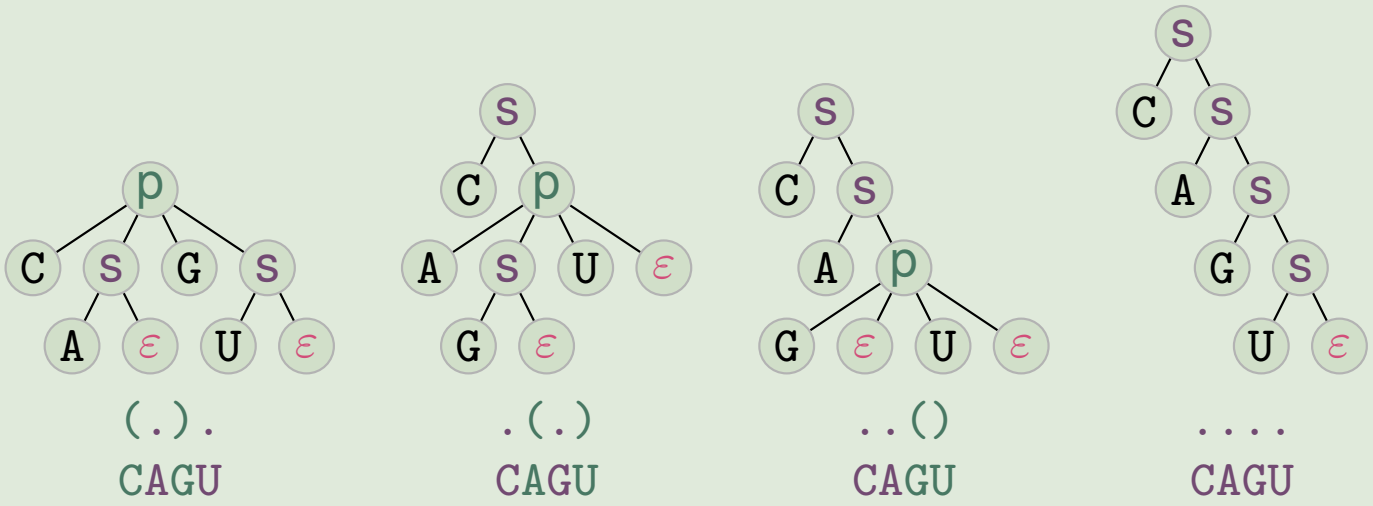
- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

Exemple 2 (ARN) :

$R \rightarrow \mathrm{A}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{A}\,R \mid \mathrm{G}\,R\,\mathrm{C}\,R \mid \mathrm{C}\,R\,\mathrm{G}\,R \mid \mathrm{G}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{G}\,R$ (Règles p)

$\mid \mathrm{A}\,R \mid \mathrm{C}\,R \mid \mathrm{G}\,R \mid \mathrm{U}\,R \mid \varepsilon$ (Règles s)

Arbres de syntaxe abstraite pour CAGU :



Deux grandes familles de formalismes

- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

  Exemple 2 (ARN) :

  $$R \rightarrow \mathrm{A}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{A}\,R \mid \mathrm{G}\,R\,\mathrm{C}\,R \mid \mathrm{C}\,R\,\mathrm{G}\,R \mid \mathrm{G}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{G}\,R \quad \text{(Règles p)}$$
  $$\mid \mathrm{A}\,R \mid \mathrm{C}\,R \mid \mathrm{G}\,R \mid \mathrm{U}\,R \mid \varepsilon \quad \text{(Règles s)}$$

  + Pondération des règles (Ex : 1/0 pour p/s)
  $\Rightarrow$ Repliement à la Nussinov = Parsing pondéré **sans équation explicite !**

  Intérêt :
  - Algorithmique générique en $O(n^3)$
  - Pas de manipulation des indices $\Rightarrow$ moins d'erreurs ...
  - Grammaires plus modulaires : Robustesse au changement de modèle
  - Séparation assez naturelle espace de recherche/score d'évaluation
  - Extensions possibles à certaines grammaires faiblement contextuelles (surcoût algorithmique)

Deux grandes familles de formalismes

- Approches *analyse syntaxique* (parsing) :
  Modèle : Grammaire non-contextuelle
  Espace de recherche = Arbres de syntaxe abstraite (parse trees).

  Exemple 2 (ARN) :

  $$R \rightarrow \mathrm{A}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{A}\,R \mid \mathrm{G}\,R\,\mathrm{C}\,R \mid \mathrm{C}\,R\,\mathrm{G}\,R \mid \mathrm{G}\,R\,\mathrm{U}\,R \mid \mathrm{U}\,R\,\mathrm{G}\,R \quad \text{(Règles p)}$$
  $$\mid \mathrm{A}\,R \mid \mathrm{C}\,R \mid \mathrm{G}\,R \mid \mathrm{U}\,R \mid \varepsilon \quad \text{(Règles s)}$$

  + Pondération des règles (Ex : 1/0 pour p/s)
  $\Rightarrow$ Repliement à la Nussinov = Parsing pondéré **sans équation explicite !**

  Intérêt :
  - Algorithmique générique en $O(n^3)$
  - Pas de manipulation des indices $\Rightarrow$ moins d'erreurs ...
  - Grammaires plus modulaires : Robustesse au changement de modèle
  - Séparation assez naturelle espace de recherche/score d'évaluation
  - Extensions possibles à certaines grammaires faiblement contextuelles (surcoût algorithmique)

  Inconvénients :
  - Apprendre le formalisme
  - Performances : Pertes de constantes d'implémentation, voir d'ordre de grandeur algorithmique pour structures plus complexes
  - Problèmes d'ambiguités *sémantique* déjà complexes à définir

$$
\left.
\begin{array}{l}
V_T = \{0, \cdots, 9, +, \times\}, \\
V_N = S, \\
P = \left\{ \begin{array}{l} S \to S + S \\ S \to S \times S \\ S \to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{array} \right.
\end{array}
\right) \text{Context-free grammar}
$$

$$
\left.
\begin{array}{l}
\mathcal{A} = \mathbb{N} \\
\lambda_{\mathcal{A}} = \left\{ \begin{array}{l} S_{\mathcal{A}}(0) = 0 \\ \vdots \\ S_{\mathcal{A}}(9) = 9 \\ S_{\mathcal{A}}(+, \times) = 0 \end{array} \right. \\
F_P = \left\{ \begin{array}{l} f_{S \to S+S}(xyz) = x + z \\ f_{S \to S \times S}(xyz) = x \times z \\ f_{S \to a \in V_T}(x) = x \end{array} \right.
\end{array}
\right) \text{Attribute system}
$$

[Crédit : J. Waldispühl]

- Grammaires (multibandes) attribuées (Lefebvre, Waldispühl *et al*)
  Calculer/minimiser un score sur un système d'attribut.
  Avantages : Pseudonoeuds simples peuvent être explorés grace à des grammaires multibandes en *synchronisant* les analyses syntaxiques .
  Inconvénient : Induction de surcoût algorithmique substantiel pour certaines familles *légèrement* contextuelles (ex : Repliement avec pseudonoeuds).

`http://people.csail.mit.edu/jw/software.php`

Deux implémentations de l'analyses syntaxique générique

Grammaire d'analyse (yield grammar) :

```
>nussinov78 alg inp = axiom s where
> (nil,left,right,pair,split,h) = alg
> s = tabulated (
>   nil   <<< empty                  |||
>   left  <<< base -~~ s             |||
>   right <<<          s ~~- base    |||
>   (pair <<< base -~~ s ~~- base)
>               `with` basepairing |||
>   split <<<       s +~+ s          ... h)
```

Algèbre d'évaluation (base-pair algebra) :

```
>pairmax :: Nussinov_Algebra Char Int
>pairmax = (nil,left,right,pair,
            split,h) where
>  nil _        = 0
>  left  _ x    = x
>  right   x _  = x
>  pair  _ x _  = x + 1
>  split x y    = x + y
>  h xs         = [maximum xs]
```

- Programmation dynamique algébrique (ADP, Giegerich *et al*)
  Avantages :
  - Applications plus générale grâce à une séparation de grammaires respectivement consacrées à l'analyse syntaxique et à l'évaluation.
  - Cadre formel et implémentation *in-extenso* ($>$ 20 publis, dont 3 thèses).

  Inconvénient :
  - Notations cryptiques, implémentation liée fortement au langage Haskell.
  - Pseudonoeuds : Formalisme doit être *hacké*, perdant une partie des bénéfice de l'approche générique.
  - Structure combinatoire disparaît derrière des notations (trop ?) abstraites.
    $\Rightarrow$ Gros problèmes d'ambiguité, impossible à traiter de façon purement automatisée (Problèmes *indécidables*).

`http://bibiserv.techfak.uni-bielefeld.de/adp/`