Université de Montréal

Une signature du polymorphisme structural d'acides ribonucléiques non-codants permettant de comparer leurs niveaux d'activités biochimiques

par

Paul Dallaire

Informatique et Recherche Opérationnelle

Faculté des Arts et Sciences

Thèse présentée à la Faculté des Arts et des Sciences
en vue de l'obtention du grade de Doctorat

en Informatique

Mai, 2014

Résumé

Des évidences expérimentales récentes indiquent que les ARN changent de structures au fil du temps, parfois très rapidement, et que ces changements sont nécessaires à leurs activités biochimiques. La structure de ces ARN est donc dynamique. Ces mêmes évidences notent également que les structures clés impliquées sont prédites par le logiciel de prédiction de structure secondaire MC-Fold.

En comparant les prédictions de structures du logiciel MC-Fold, nous avons constaté un lien clair entre les structures presque optimales (en termes de stabilité prédites par ce logiciel) et les variations d'activités biochimiques conséquentes à des changements ponctuels dans la séquence.

Nous avons comparé les séquences d'ARN du point de vue de leurs structures dynamiques afin d'investiguer la similarité de leurs fonctions biologiques. Ceci a nécessité une accélération notable du logiciel MC-Fold. L'approche algorithmique est décrite au chapitre 1. Au chapitre 2 nous classons les impacts de légères variations de séquences des microARN sur la fonction naturelle de ceux-ci. Au chapitre 3 nous identifions des fenêtres dans de longs ARN dont les structures dynamiques occupent possiblement des rôles dans les désordres du spectre autistique et dans la polarisation des œufs de certains batraciens (*Xenopus* spp.).

Abstract

Recent experimental evidence indicates that RNA structure changes, sometimes very rapidly and that these changes are both required for biochemical activity and captured by the secondary structure prediction software MC-Fold. RNA structure is thus dynamic.

We compared RNA sequences from the point of view of their structural dynamics so as to investigate how similar their biochemical activities were by computing a signature from the output of the structure prediction software MC-Fold.

This required us to accelerate considerably the software MC-Fold. The algorithmic approach to this acceleration is described in chapter 1. In chapter 2, point mutations that disrupt the biochemical activity of microRNA are explained in terms of changes in RNA dynamics. Finally, in chapter 3 we identify dynamic structure windows in long RNA with potentially significant roles in autism spectrum disorders and separately in *Xenopus* ssp. (species of frogs) egg polarisation.

Keywords : NCM, Nucleotides Cyclic Motifs, Dynamic programming, microRNA, dynamic structure, RNA, VegT mRNA, neuroligins.

Table des matières

Liste des Tableaux

Liste des Figures

Le coût de la lumière est la profondeur des ombres.

Remerciements

Mes tout premiers remerciements vont à mon directeur de thèse Dr. François Major. Sans son ouverture, sa disponbilité sans borne, son imagination débordante, sa clairvoyance scientifique et son support intarissable, rien de tout cela n'aurait été possible. Je lui suis éternellement redevable.

Merci à Cosima d'être elle-même et d'enraciner ma vie chaque jour dans l'importance de ses fous rires.

Merci à ma mère Gaby sans qui je serais certainement mort en chemin.

Merci à Monique ma très chère sœur dont la ténacité exemplaire, l'intense besoin de vivre (et incidement l'accès au chalet) m'élèvent un peu plus haut, me permettant d'entrevoir des horizons neufs.

Remerciements pour le chapitre 1

Je tiens à souligner les diverses contributions aux idées et commentaires techniques donnés par les individus suivants. Yann Ponty pour la première idée quant à la manière d'adapter le modèle des NCM à la programmation dynamique. Jérome Waldisphul pour des commentaires éclairants au sujet de la table de partition des paires de bases qui nous ont éclairé sur la nature du problème. Stefanie Schrimer a implantée le modèle des NCM dans le système ADP et a clarifié un point particulièrement nébuleux au sujet des combinaisons improbables de NCM. Cédric Saule qui a implanté MC-Fold2 une première version de la programmation dynamique sur le modèle des NCM. Marc-Frédéric Blanchet pour de nombreuses discussions très utiles et pour ses talents de canard en caoutchouc (*rubber ducking*) ainsi que, de concert avec Maria Abella Angela de Los Angeles, pour ses contributions à l'identification de failles (bogues) dans l'implantation et Nicolas Scott pour une relecture critique de certaines parties du manuscrit.

Remerciements pour le chapitre 2

Merci à nos collaborateurs Peng Jin, Huiping Tan, Keith Szulwach, Christopher Ma qui ont effectués les expériences de biologie moléculaires.

Merci à Julie Pelloux et Julie Robitaille qui ont reproduit les résultats expérimentaux sur miR-125a et qui travaillent à vérifier l'effet délétaire de SNP sur des microARN que nous avons identifié dans genbank et dbSNP.

Remerciements pour le chapitre 3

Merci à nos collaborateurs Malgorzata Kloc et Arkadiy Reunov pour les travaux expérimentaux sur le projet de *vegt* chez *X. leavis*.

Merci en particulier Nahum Sonenberg et Christos Gkogkas ainsi qu'à tous nos collaborateurs sur le projet des neuroligines.

Liste des Symboles

$a(f)$ arité de la fonction f

A arêtes d'un graphe

$\mathcal{A}$ un algorithme en programmation dynamique

$\mathcal{A}_p$ un algorithme en programmation dynamique solutionnant le problème P

c une constante dans $\mathbb{R}$

$\mathcal{C}$ l'ensemble des règles de composition

$\mathcal{C}_p$ l'ensemble des règles de composition du problème P

Δ (voir aussi Δx) seuil dans le problème est Δ-sous-optimaux (de Waterman-Byers)

Δx variation discrète de x

dx/dy taux de variation continue de x relativement à y

$\mathcal{D}$ un problème de *décoration*

$D(P)$ une *décomposition* du problème P

$D(\gamma)$ une *décomposition* de l'instance γ

$\mathfrak{D}$ distance/similarité entre signatures

Σ l'alphabet de *sortie*

Σ_{PP} alphabet du langage parenthèse-point. ($\Sigma_{PP} = \{'.', '(', ')'\}$)

E énergie libre (thermodynamique)

f fonction quelconque

f_x fréquence de x (son décompte dans un ensemble donné)

γ un mot sur Γ^n, une instance d'un problème

Γ l'alphabet d'un problème

Γ_{ARN} l'alphabet des ARN

G énergie libre de Gibbs (thermodynamique)

$g(\dots)$ poids de la partie résolue d'un arbre

$\mathcal{G}$ une grammaire

$\mathcal{G}_p$ une grammaire dont les règles sont issues de $\mathcal{C}_p$

$\mathbb{G}^+$ et $\mathbb{G}^-$ ensembles de gènes positifs et négatifs

H enthalpie

i, j, k, l, m, o, p, q des constante dans $\mathbb{N}$

k_B constante de Boltzmann (thermodynamique)

$\mathcal{I}_p$ ensemble d'*itérateurs* (valeurs permises d'indices pour les règles dans $\mathcal{C}_p$)

N nœuds d'un graphe (sommets); symboles non-terminaux d'une grammaire

$\mathbb{N}$ les nombres naturels

$\mathcal{N}$ l'ensemble de tous les NCM

$\mathfrak{N}$ une suite de NCM contigues

$\mathcal{O}$ un problème d'optimisation

$O(\dots)$ ordre de

p_x probabilité *a priori* de x

P un problème

ϕ un prédicat quelconque

ψ_x le poids de la partie non résolue d'un sous arbre dont la racine est x

φ, Π des piles

Φ pseudo-énergie (énergie interne estimée selon la probabilité)

q quantité de chaleur (thermodynamique)

$\mathcal{Q}$ cycle nucléotidique minimal $\langle A_Q, N_Q \rangle$ (un graphe)

r une règle

R constante universelle des gaz parfaits, version molaire de k_B (thermodynamique)

$\mathbb{R}$ les nombres réels

$\mathcal{R}_s$ graphe d'ARN 2D (séquence et paires de nucléotides)

ξ états intermédiaires utilisés dans l'énumération de solutions

s, t des séquences d'ARN (sur $\{A, C, G, U\}^+$)

S entropie (thermodynamique)

$S(\gamma)$ les solutions au problème dont γ est une instance

$S^{\phi}(\gamma)$ sous ensemble de $S(\gamma)$ pour lequel le prédicat ϕ est vrai

$S^{\Delta,OPT}(\gamma)$ sous ensemble de $S(\gamma)$ dont les éléments ont des poids qui ne sont pas plus loin que Δ d'un optimal selon la politique d'optimisation OPT

$S^{\kappa,OPT}(\gamma)$ sous ensemble donnant les membres de $S(\gamma)$ et dont les éléments ont des poids qui ne sont pas plus loin que τ^{κ} d'un optimal selon la politique d'optimisation OPT

$S^{\kappa*,OPT}(\gamma)$ sous ensemble de $S^{\kappa,OPT}(\gamma)$ de taille κ et respectant certaines contraintes (voir texte)

$\mathfrak{S}$ suite de $\mathfrak{N}$

σ un mot sur Σ^+

σ^s une signature pour la séquence s

T température (thermodynamique)

τ, μ des arbres

τ^o un arbre optimal

τ^{κ} le $\kappa^{ième}$ arbre dans une suite d'arbres ordonnée selon leurs poids

$\mathcal{T}_p$ les tables de *mémoisation* de l'algorithme de programmation dynamique $\mathcal{A}_p$

Θ les catégories de NCM

θ un élément de Θ

$\mathcal{U}_p$ une variante de $\mathcal{T}_p$

$\mathcal{V}_p$ autre variante de $\mathcal{T}_p$

ω une fonction de poids; longueur d'une fenêtre

$\mathcal{W}_{\mathcal{C}_P}$ les fonctions de poids en correspondance avec les règles de $\mathcal{C}_p$

W le poids d'un arbre

χ le poids de la partie non résolue d'un arbre

$[\cdots]$ une suite ordonnée

$\langle\cdots\rangle$ un tuple

$\{\cdots\}$ un ensemble

$|\cdots|$ valeur absolue, longueur d'un mot, nombre d'éléments dans un objet (ensemble, vecteur)

$f: \mathcal{X}^x\mathcal{Y}^y \ldots \rightarrow \mathcal{V}^v\mathcal{W}^w \ldots$ une fonction f ayant $\mathcal{X}^x\mathcal{Y}^y \ldots$ pour domaine et $\mathcal{V}^v\mathcal{W}^w \ldots$ pour image

$A[i,j]$ sous-suite de A commençant avec le $i^{ième}$ élément de A et de longueur $j - i + 1$

X^* étoile de Kleine donnant un mot de longueur quelconque sur l'alphabet X

X^+ mot de X^* de longueur au moins 1

$\mathcal{X}^x$ espace défini sur les $\mathcal{X}$ et de dimensionnalité x

a^l un mot de longueur l

$\vec{x}$ variable explicitement vectorielle

$\rightarrow$ dérivation

$\rightarrow_r$ dérivation par r

$\rightarrow^*$ dérivation en multiples étapes

$g \rightarrow d$ règle de grammaire dont g (la partie gauche) et d (la partie droite) sont des mots

$\leftarrow$ assignation

$\alpha \lhd \beta$ sous arbre, α est un sous arbre de β

$\alpha \bowtie \beta$ non sous arbre, α n'est pas un sous arbre de β et β n'est pas un sous arbre de α

X•Y paire. Le nucléotide X paire avec le nucléotide Y

Liste des Acronymes

2D$^+$ Ensemble des toutes les paires de nucléotides d'une molécule d'ARN. (Avec γ une molécule d'ARN, $2D^c(\gamma) \subseteq 2D(\gamma) \subseteq 2D^+(\gamma)$.)

2D Ensemble maximal des paires de nucléotides d'une molécule d'ARN excluant tout pseudo-nœuds et plates formes (triplets, quadruplets).

2D^c Ensemble maximal des paires canoniques d'une molécule d'ARN excluant tout pseudo-nœud et plates formes (triplets, quadruplets).

3D Dénote une structure tridimensionnelle d'une molécule. ie : le positionnement de tous ses atomes dans un espace en 3 dimensions.

ADN Acide désoxyribonucléique.

ARN Acide ribonucléique.

ARNm ARN messager.

ARNt ARN de transfert.

ASD *Autism like Syndrome Disorder*. *Autistic Syndrome Disorder.* Syndrome de type autistique.

MC-Flashfold Version accélérée (décrite ici) du logiciel MC-Fold.

MC-Fold Logiciel de prédiction de structures $2D$ d'ARN basé sur les NCM.

MFE *Minimum Free Energy*. L'énergie libre la plus faible possible pour un ARN.

miARN, microARN Petits ARN découverts à la fin des années 1990 et impliqués dans la régulation de l'expression de gènes chez les eucaryotes.

MPR Meilleur Poids Réalisable avec la solution partielle courante.

nt. Abréviation de nucléotides. Unité de mesure de la longueur d'une séquence d'ARN ou d'ADN.

NCM *Nucleotides Cyclic Motif*. Motif cyclique de nucléotides.

PDB *Protein Data Bank.* Base de donnée conservant les structures connues de macromolécules biologiques.

PO Poids Optimal. Poids d'une solution optimale.

RMN Résonance Magnétique Nucléaire.

RNAsubopt Un logiciel de prédiction de structures $2D^c$ sur le modèle des plus proches voisins qui a la particularité d'énumérer rapidement les solutions dans $S^{\Delta,MIN}(\gamma)$.

SDL Structure Dynamique Locale.

SNP *Single Nucleotide Point mutation*. Mutation ponctuelle dans une séquence. Ce terme est normalement compris comme dénotant de *petites* mutations incluant de *courtes* délétions, insertions et remplacements dans une séquence relativement à une séquence référence. (Ici, petites et courtes réfèrent à des longueurs de l'ordre d'une dizaine de nucléotides ou moins. La vaste majorité des SNP sont de longueur 1.)

0 Introduction

Dans un premier temps nous présentons une introduction générale au problème de la dynamique structurale des ARN en mettant une emphase sur son importance lors de la maturation des microARN. Ensuite nous revenons sur la structure des ARN un peu plus en détails avant d'exposer des approches à la prédiction de structure.

0.1 L'ARN et les microARN

L'ADN encode et maintient l'information génétique à travers les générations cellulaires mais puisque la double hélice découverte par Watson et Crick (Watson and Crick 1953) est essentiellement inerte, elle est incapable d'agir sur son environnement et seule sa transcription peut avoir des effets. Cette transcription se produit invariablement vers l'ARN.

Ces effets sont entre autres la conséquence d'activités chimiques particulières nommées catalyses qui sont le plus souvent opérés après la traduction de l'information génétique en protéines via la nature enzymatique de celles-ci. On distingue d'une part les effets indirects de l'ARN qui sont effectués par le biais de la réalisation de leurs messages génétiques par des protéines et d'autre part les effets directs de l'ARN sur l'environnement chimique de la cellule.

$$ADN \xrightarrow{transcription} ARN \rightarrow \begin{cases} Protéines \rightarrow \text{effets} \\ \text{effets directs de l'ARN} \end{cases}$$

Les composantes encodant une protéine sont lues à la queue leu leu, par triplets dans un type d'ARN (l'ARN messager ou ARNm) au cours d'un processus qui déplie l'ARNm pour y lire l'information génétique. Il existe une correspondance depuis les nucléotides, l'alphabet des acides nucléiques (ADN et ARN), $\Gamma_{ADN/ARN} = \{A, C, G, T/U\}$ vers celui des vingt acides aminés encodant pour les protéines Ψ.

$$\Gamma_{ADN/ARN} \xrightarrow{traduction} \Psi$$

Les effets directs de l'ARN sont très nombreux, souvent orientés vers la traduction et son contrôle ou le maintient de l'intégrité chromosomique (Cech and Steitz 2014). Un de ces effets se nomme interférence (Lee, Feinbaum et al. 1993; Ambros 2001). L'interférence consiste au blocage spécifique de la traduction d'un ARNm par un second ARN plus court d'une classe qu'on nomme microARN ou miARN.

$$miARN \dashv traduction$$

($\alpha \dashv \beta$ indique que α inhibe β) Nous savons que les règles établissant le ciblage des ARNm par les miARN sont basées sur la complémentarité de séquence (essentiellement A-U, G-C) entre un site de l'ARNm et le miARN (Bartel 2009) mais les subtilités de cette reconnaissance sont élusives et sont toujours sous étude (voir par exemple (Li, Kim et al. 2014)). Bien que les ARN soient des copies de régions de l'ADN, les transcrits primaires sont traités et littéralement édités vers leurs formes matures dans le noyau des cellules eucaryotes. Dans la voie canonique de synthèse, les miARN sont transcrits depuis l'ADN sous une forme primitive et cette molécule subit des étapes de maturation dont le résultat final est le miARN mature localisé au cytoplasme, là où se produit la traduction (Wilson and Doudna 2013). Maturation des miARN :

$$ADN \xrightarrow{transcription} pri\text{-}miARN \xrightarrow{clivage} pre\text{-}miARN \xrightarrow{exportation} pre\text{-}miARN \xrightarrow{clivage} duplex \xrightarrow{sélection\ de\ brin} miARN$$

pri-miARN : transcrit primaire du miARN, *pre-miARN* : précurseur du miARN. Les structures adoptées par ces différents intermédiaires sont reconnues par des enzymes et cette reconnaissance détermine leurs taux de maturation (Han, Lee et al. 2006; Finnegan and Pasquinelli 2013). Cette dépendance du taux de maturation des pri-miARN et des pre-miARN sur les structures adoptées par ces molécules nous permet d'espérer identifier, dans la séquence d'ADN, les régions qui génèrent des miARN grâce à nos connaissances sur la structure des ARN (voir notamment (Song, Wang et al. 2013)).

Nous savons que les molécules d'ADN viennent normalement en paires et qu'elles adoptent la structure en double hélice. A l'opposé, les molécules d'ARN sont presque toujours simple brin. Elles adoptent des structures hautement spécifiques, dépendantes de la séquence de nucléotides qui les composent. Lors de la transcription, une nouvelle molécule d'ARN est synthétisée et nous nommons *repliement* le processus au cours duquel cette molécule passe à l'état structuré.

ARN à la structure indéfinie ⇋ ARN structuré

Au sens thermodynamique, pour une température et pression données, cette relation est *spontanée*. Ceci veut dire que si l'on ne considère pas la durée de la réaction, le côté droit de l'équation est favorisé et conséquemment que la réaction de repliement libère de l'énergie. La structure adoptée par un ARN est donc dépendante de sa séquence en nucléotides et il existe une fonction d'énergie déterminant sa stabilité. Autrement dit, il doit exister une fonction d'énergie optimisant la structure d'une séquence d'ARN. Évidemment, les séquences d'ARN étant des objets combinatoires cette fonction d'énergie est soumise aux problèmes d'explorations de surfaces et cette surface présente des minimums locaux.

Les ARN changent dynamiquement leurs structures (Al-Hashimi and Walter 2008). L'extensivité de ces changements est inconnue et est certainement variable, selon la séquence concernée. Nous postulons que ces variations dans la structure leur permettent d'interagir avec différents partenaires moléculaires et que ces structures sont proches des minimums locaux dans leurs fonctions d'énergie de repliement. Au moins pour le cas des microARN, les structures prédites par les programmes de prédiction de structures en tant que solutions sous-optimales s'approchent suffisamment des configurations fonctionnelles pour refléter leur fonction. Mais nous ne savons pas choisir parmi les prédictions, celles qui sont réalisées par les ARN dans la cellule.

Nous allons donc interpréter les sorties de programmes de prédiction de structures pour des microARN comme des ensembles de structures dont plusieurs sont réalisées par les microARN au cours de leurs maturations dans le noyau.

Ainsi, nous pensons que:

1. Les microARN au cours de leur maturation doivent occuper des configurations multiples.
2. Ces configurations sont reflétées dans leurs séquences et approchées par les prédicteurs courants dans leurs listes de structures sous optimales.

Nous disposons d'un schéma de programmation dynamique en $O(n^3)$ pour calculer la structure secondaire la plus probable d'ARN étant donnée une fonction d'énergie (Zuker and Stiegler 1981). L'algorithme trouve la structure qui maximise la libération d'énergie lors du repliement de la molécule. Cet algorithme, de même qu'un grand nombre d'autres approches, vise à élucider une structure unique d'énergie optimale déterminé par la séquence. L'atteinte de ce but est très difficile et, malgré les nombreuses améliorations obrtenues au fil du temps au sein d'une littérature riche, les résultats sont imparfaits. L'évaluation de la justesse des prédictions fait elle même l'objet d'un certain nombre de bancs d'essais. Le lecteur intéressé par ces travaux est invité à consulter (Doshi, Cannone et al. 2004) où il trouvera un compendium des résultats de plusieurs logiciels de prédiction testés sur des jeux d'essais variés. Cela ne nous éclaire cependant pas beaucoup sur la structure ou la fonction réelles de molécules d'ARN au final.

L'algorithme peut être étendu pour tenir compte de certaines classes topologiques de pseudo-nœuds avec accroissement conséquent des ressources nécessaires à l'exécution de l'algorithme ($O(n^6)$ en temps et $O(n^4)$ en espace) (Rivas and Eddy 1999) ($O(n^4)$ en temps et $O(n^2)$ en espace pour une sous-classe d'entre eux (Reeder and Giegerich 2004)) mais le cas général des pseudo-nœuds dans le modèle d'énergie 'des plus proches voisins' est NP complet (Lyngso and Pedersen 2000). Un certain nombre d'articles de revue concernant la prédiction de structure secondaire des ARN à partir de la séquence sont

disponibles, voir notamment (Mathews, Moss et al. 2010; Rivas 2013) pour une description des éléments essentiels des algorithmes de prédiction de structure d'ARN.

Nous pouvons lister un ensemble de pistes d'explication du mauvais fonctionnement des prédictions de structures :

1. L'algorithme est applicable mais la fonction d'énergie est trop imprécise conduisant à des résultats erronés.
2. L'algorithme n'est pas applicable car les chemins cinétiques empruntés par la molécule d'ARN se repliant l'empêchent de trouver le minimum global.
3. Nous interprétons mal les sorties du programme. Soit, notre règle d'or pour juger des prédictions est fausse. (Par exemple une structure déterminée par cristallographie en présence de métaux lourds est elle vraiment réalisée *in vivo*?)

Nous pensons que, au moins en ce qui concerne les miARN, la troisième option s'applique. *In extenso* il n'est pas raisonnable d'assumer que nous sachions comment évaluer la structure réelle des miARN *in vivo* et par conséquent les jeux de données grâce auxquels nous évaluons la justesse des prédictions peut être questionnée. Leurs structures laissent croire que la fonction d'énergie doit être applicable puisqu'elles ne présentent pas les cas où la fonction d'énergie est difficile à mesurer et imprécise (Zeng and Cullen 2004; Han, Lee et al. 2006). Et nous n'entrevoyons pas de complexité particulière dans la cinétique des chemins vers sa structure d'énergie minimale.

Dans le second chapitre nous proposons une manière d'interpréter la sortie des programmes de prédiction de structure lorsque les ARN analysés sont dynamiques. Avec des modèles d'énergie à la fine pointe nous constatons que les résultats obtenus convergent sur des conclusions similaires pour ces ARN. Dans cette expérience clé nous proposons une nouvelle manière de lire les prédictions de structure et d'y associer directement une interprétation sur la fonction biologique des ARN concernés permettant ainsi l'interprétation des prédictions pour des ARN polymorphes.

Dans notre troisième chapitre, nous utilisons des variations de l'approche développée dans le second chapitre pour identifier les segments d'ARN structurés dans des sections non codante d'ARN messagers.

0.2 Structure de l'ARN

L'ARN est en général structuré. C'est à dire que la molécule d'ARN se replie sur elle-même et que ce repliement permet un positionnement de ses atomes de manière conséquente aux activités qui l'impliquent. Les ARN de transfert par exemple (ARNt) entrent en contact avec un certain nombre de partenaires et le détail des interfaces entre l'ARNt et ces partenaires sont précis et distinguent un parmi les différents ARNt.

Le squelette d'un ARN est constitué d'une chaines (polymères) de ribose phosphatés attachés par leurs résidus phosphates. Chaque ribose de la chaine est lié chimiquement à une base azotée et il existe 4 telles bases bien connues (Saenger 1984). La suite des bases dans la chaine détermine son identité laquelle est à son tour déterminée par le segment d'ADN dont elle est issue. Toutes choses étant égales par ailleurs, la suite de bases détermine les possibilités de repliement de la molécule. Les bases azotées ressemblent essentiellement à des disques plus ou moins équarris d'épaisseurs réduites relativement à leurs largeurs. Elles sont hydrophobes ce qui leur confère une propension à l'agrégation par empilement. De plus, leurs faces exposent quelques orbitales électronégatives et électropositives qui trouvent complémentarité dans les faces d'autres bases et quelque fois dans des riboses ou phosphates à proximité en partageant des électrons dans un type de lien chimique nommé le *pont hydrogène*. Le phénomène est similaire dans l'ADN où un appariement parfait des deux brins permet la formation d'une double hélice continue. Mais les molécules d'ARN sont typiquement simple brin et doivent satisfaire les ponts hydrogènes avec des bases qui ne leurs sont pas nécessairement compatibles (Figure 1). Ainsi les différentes faces des bases azotées sont mises à profit en formant des structures souvent étonnantes. L'énergie avec laquelle deux bases empilées résistent à la séparation (*stacking energy*) ainsi que celle contenue dans les ponts hydrogènes liant deux bases appariées (*pairing energy*) ont d'abord été estimées (Delisi and

Crothers 1971; Tinoco, Uhlenbeck et al. 1971) puis, elles ont été mesurées sur des brins d'ARN appariés en solution pour différentes séquences dans une suite de travaux dont le plus important est possiblement (Mathews, Sabina et al. 1999) et ces paramètres font toujours l'objet de raffinements (revus dans (Mathews, Moss et al. 2010; Turner and Mathews 2010)). Ces énergies varient bien sûr en fonction de l'identité des bases impliquées dans ces relations mais également selon les bases trouvées dans le voisinage de celles-ci et sont disponibles sous forme de tables d'énergies. Les contributions de segments à l'appariement imparfait ou simples brins (dans certaines boucles notament), ainsi que la contribution de paires de bases qui ne sont empilées que d'un seul côté (comme cela doit être le cas aux bouts de segments empilés) sont difficiles à estimer.

Les détails de la structure des ARN sont mieux connus depuis l'an 2000 où en particulier la cristallographie en rayon-X sur de gros ARN a commencée à être réalisée permettant la visualisation à la résolution atomique, des bases dans celles-ci (Ban, Nissen et al. 2000; Wimberly, Brodersen et al. 2000). Ces travaux ont significativement contribué au catalogue des structures d'ARN connues, augmentant la fiabilité potentielle de l'estimation de l'énergie des sous structures par échantillonnage de sous-structures parmi les structures résolues. Dans cette thèse, nous utilisons le modèle d'énergie du logiciel de prédiction de structures d'ARN MC-Fold (Parisien and Major 2008), lequel est essentiellement un dictionnaire associant de petits blocs récurants (les NCM) à leurs fréquences d'observations et de co-occurences dans ces structures.

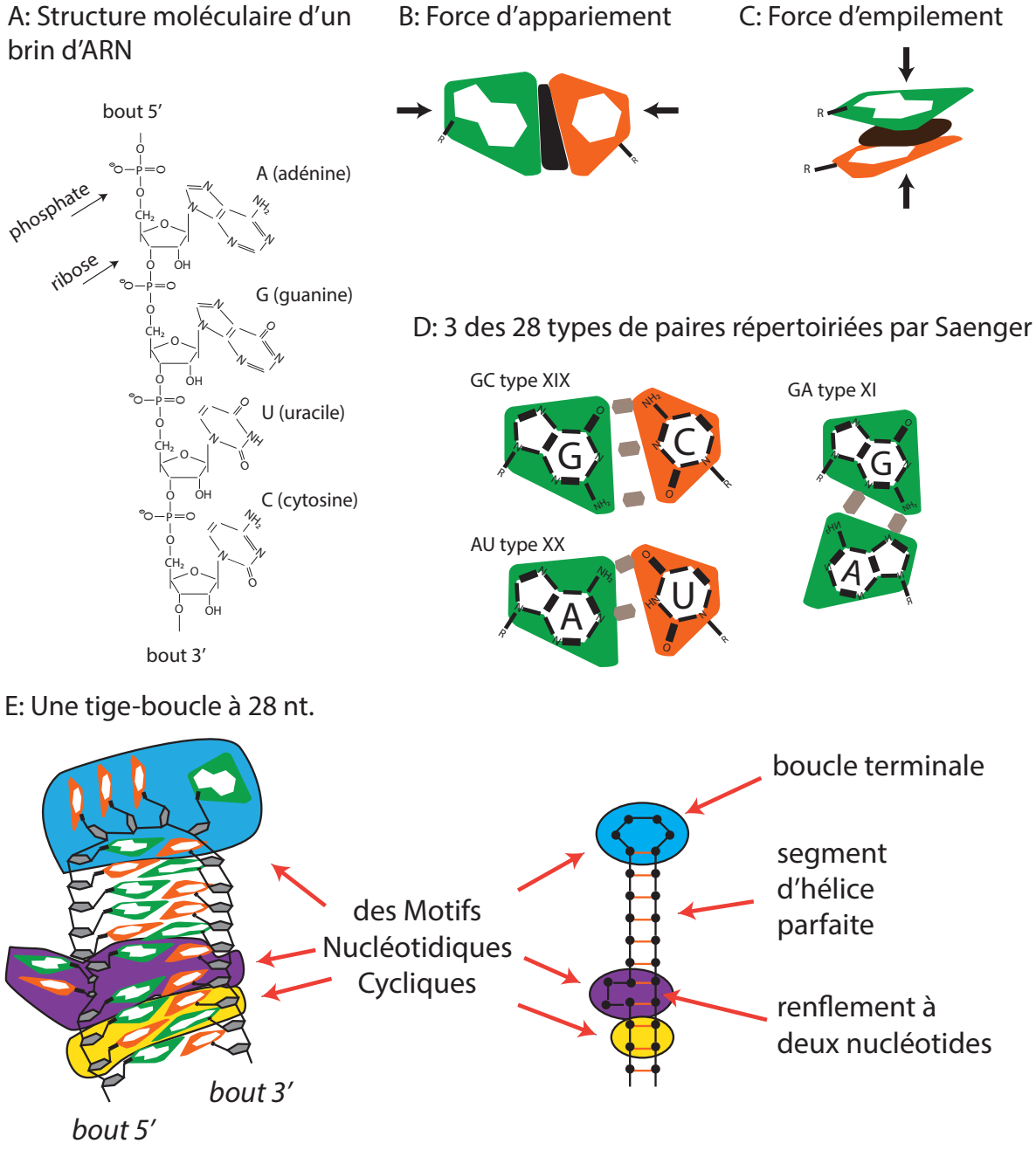

Figure 1. Éléments de la structure des molécules d'ARN.

A : La structure d'un brin de molécule d'ARN de séquence AGUC montrant la plupart des atomes (à l'exception de la plupart des atomes de carbone). B : La force d'appariement s'exerce perpendiculairement à l'axe des faces jointes par des pont H. C : La force d'empilement entre deux bases. D : Trois des 28 types de paires répertoriées par Saenger

(Saenger 1984) sont montrées : les paires GC et AU canoniques ainsi que la paire GA dans une de ses configurations. E : Le diagramme d'une tige boucle imaginaire de 28 nt. est montrée dans deux représentations simplifiées. À gauche on peut constater l'effet de l'énergie d'empilement, la déstabilisation causée par un renflement de deux nucléotides dans une tige autrement parfaite ainsi que la stabilisation de la boucle terminale conséquente à l'énergie d'empilement de trois de ses nucléotides. Ces effets sur l'énergie globale de la molécule sont captés par les NCM. Les envergures de trois des NCM de la structure sont montrées. L'énergie associée à chacun de ces NCM capture l'ensemble des interactions survenant dans ces zones.

0.2.1 Méthodes d'observation

De nombreuses méthodes d'analyse chimico-physiques donnent des informations partielles sur la structure des macromolécules et sont parfois clés à la modélisation structurales. Par exemple la diffusion à petits angles de rayons X (*SAXS : Small Angle X-ray Scattering*) peut-être utilisé pour contraindre la modélisation (Yang, Parisien et al. 2010; Parisien and Major 2012).

Quatre méthodes principales sont utilisées pour investiguer la structure des molécules d'ARN : les sondages chimiques et enzymatiques (*probing*), la cristallographie aux rayons X, la résonance magnétique nucléaire (RMN) et la microscopie électronique.

0.2.1.1 Sondage chimique et enzymatique

Le sondage chimique révèle partiellement la structure d'un ARN en identifiant la configuration de certains de ses nucléotides. La molécule d'ARN subit une agression chimique douce (attaque acide, oxydation, etc) ou enzymatique qui brise la molécule d'ARN dans des régions où se trouvent exposés des liens sensibles (Xu and Culver 2009). Les lieux sensibles aux coupures sont ensuite identifiés par électrophorèse, séquençage ou même résonance magnétique nucléaire. L'interprétation de ces profiles d'attaque chimique est difficile mais on peut en tirer néanmoins des informations utiles (Bindewald, Wendeler et al. 2011). Ces méthodes jouissent d'un engouement certain et les progrès sont régulièrement évalués (Wilkinson, Merino et al. 2006; Deigan, Li et al. 2009; Sukosd, Swenson et al. 2013). Par exemple une nouvelle approche consiste à mettre en lien les informations obtenues par mutagénèse dirigée exhaustive avec celles obtenues par attaque chimique (Kladwang and Das 2010; Kladwang, VanLang et al. 2011).

0.2.1.2 Cristallographie en rayons X

La cristallographie révèle la position exacte de tous les atomes d'une molécule cristallisée. Cette méthode n'est pas idéale parce que (1) elle n'identifie qu'une seule

conformation et (2) elle impose un contexte chimique particulier lequel peut induire des modifications à la structure native. Les conditions nécessaires pour obtenir un cristal incluent en particulier l'addition de sels et des concentrations élevées de molécules purifiées. Ces conditions peuvent induire des changements majeurs dans la structure des ARN et, d'autre part, les occurrences (Reyes, Garst et al. 2009) de plusieurs conformations et états transitoires constituent des obstacles sérieux à la cristallisation (Felden 2007). Les cartes produites ne sont vraies que pour la seule configuration trappée dans le cristal. Les modèles moléculaires dont nous disposons sont souvent obtenus par cristallographie et en particulier sans cette méthode la structure du ribosome à ce niveau de détail (résolution de quelques angströms) serait inaccessible.

0.2.1.3 Résonance magnétique nucléaire

Cette méthode est très utilisée en chimie organique. La plupart des nombreuses molécules de cette discipline sont cataloguées par leurs spectres RMN agissant comme des signatures distinctives. L'étude des macromolécules reste limitée en RMN à cause de limites techniques sur la taille des molécules. Ceci est une méthode de choix pour observer la dynamique moléculaire des petits ARN (Felden 2007). De récents progrès ont permis, par exemple, la détermination de motifs non-canoniques d'ARN (Sripakdeevong, Cevec et al. 2014) et la dynamique de certains autres motifs (Al-Hashimi and Walter 2008; Nikolova, Kim et al. 2011; Dethoff, Chugh et al. 2012).

0.2.1.4 Microscopie électronique

La microscopie électronique est très utilisée en biologie cellulaire et en virologie car elle permet de visualiser les structures des organelles et des capsides virales directement dans des tranches de cellules fixées dans la paraffine (Goldsmith and Miller 2009). Plus récemment, des techniques de traitement d'image et l'amélioration de la qualité des images permettent d'appliquer cette méthode à l'étude des gros ARN (voir par exemple (Zhang, Bettadapura et al. 2012)). La résolution reste très faible cependant et le positionnement des atomes est questionnable.

0.2.2 Topologie

Les bases s'apparient en formant des paires, le plus souvent de proche en proche. Le repliement de la chaine se produit spontanément dès sa synthèse. Certains rapportent (Xayaphoummine, Bucher et al. 2005a; Lai, Proctor et al. 2013) que le repliement serait concomitant à la transcription depuis l'ADN. En effet, l'ARN est normalement synthétisé comme une copie conforme de régions de l'ADN chromosomique dans un processus nommé transcription et il n'existe pas de raison *a priori* pour que la chaine naissante attende d'être complètement transcrite pour commencer à se replier. Bien au contraire, la forme non-repliée est énergétiquement défavorisée. Cependant, les chaines d'ARN ne sont pas automatiquement auto-complémentaires comme le sont deux brins d'ADN complémentaires. Ainsi, le repliement se produit par optimisation le long d'un chemin de fouille qui est peut-être invariable. Il existe néanmoins des situations où des facteurs extrinsèques entrent en ligne de compte. Par exemple, le cas de pause dans la transcription permet au segment de chaine déjà transcrit de trouver un minimum avant que ne soit poursuivie la synthèse de la molécule (Martinez-Rucobo and Cramer 2013).

Une méthode récente permet la mesure directe de l'énergie déployée par la chaine d'ARN pour retenir sa structure lorsque étirée mécaniquement (Onoa, Dumont et al. 2003; Tinoco 2004; Manosas, Wen et al. 2007). Cette méthode permet d'appliquer une traction à un bout d'une molécule (dont l'autre bout est fixé) et d'observer la résistance à cette traction. Bien que récente, cette méthode confirme directement la propension de la molécule à se replier sur elle-même en absence de facteurs extrinsèques.

Le repliement est donc spontané et ne présente pas de topologie complexe telle que les nœuds lesquels seraient possibles si le brin était brisé et réparé par exemple. Les nœuds ne sont pas impossibles à strictement parler mais ils n'ont toujours pas été observés et rien ne laisse supposer qu'ils trouvent leurs places parmi les structures d'ARN. Les sous-structures communes comprennent : l'hélice, la boucle terminale, la boucle interne à k-branchements ($k > 1$), le segment non-apparié et le pseudo-nœud. Certains assemblages fréquents de ces structures ont des noms communs dont en particulier la tige-boucle aussi

dite boucle en épingle à cheveux (*hairpin loop*) qui est simplement une hélice coiffée d'une boucle terminale (Waterman 1978). Le *Y-shape* qui consiste en deux épingles à cheveux consécutives dont les bases sont jointes dans une boucle interne à trois branches et dont la troisième branche est une hélice. Dans une hélice, l'appariement peut être interrompu et exclure un ou plusieurs nucléotides de la suite d'appariements. Techniquement cette interruption est une boucle interne à deux branches mais cela est tellement fréquent qu'on parle plutôt de renflement (ou *bulge*). Cette interruption dans l'appariement peut n'impliquer qu'un seul brin d'une hélice ou les deux brins à la fois et comprend des nombres variables mais limités (approximativement 1-5) de nucléotides sur chaque brin.

0.2.3 Représentation des structures

Les coordonnées spatiales des atomes et possiblement des liens chimiques qu'ils partagent constituent la structure en trois dimensions de la molécule. Ceci est la *structure tertiaire* ou 3D.

Il arrive que des molécules s'agglomèrent conséquemment à la complémentarité (géométrique et chimique) de leurs surfaces. Ce genre d'interaction est fréquent, spécifique et souvent nécessaire à la fonction biologique. Les sous unités du ribosome en sont un exemple flagrant. La structure de ces complexes se nomme la *structure quaternaire*.

Posons $\Gamma_{ARN} = \{A, C, G, U\}$ l'alphabet des mots d'ARN où chaque lettre correspond à un nucléotide et $s \in \Gamma_{ARN}^{n}$ une séquence d'ARN de longueur n. La séquence est parfois nommée *structure primaire*. Notons $s[i] \bullet s[j]$ la paire formée des nucléotides en position i et j dans s. Les paires possibles sont $\{x \bullet y | x, y \in \Gamma_{ARN}\}$ mais l'ensemble des *paires canoniques* est $PC = \{G \bullet C, A \bullet U\}$ (parfois $\{G \bullet C, A \bullet U, G \bullet U\}$).

La définition de *structure secondaire* (2D) varie selon les auteurs. Nous utilisons les définitions suivantes :

2D^{+} : Ensemble de toutes les paires de nucléotides d'une molécule d'ARN. Avec $s \in \Gamma_{ARN}^{+}$:

$$2D^{+}(s) = \{(i,j)|s[i] \bullet s[j]\}$$

2D : Ensemble maximal des paires de nucléotides d'une molécule d'ARN excluant tout pseudo-nœuds et plates formes (triplets, quadruplets) :

$$2D(s) = MAX\{(i,j)|s[i] \bullet s[j], \nexists(k,l) \in 2D(s), k \leq i \leq l \leq j \text{ ou } i \leq k \leq j \leq l\}$$

$2D^{c}$: Ensemble maximal des paires canoniques d'une molécule d'ARN excluant tout pseudo-nœud et plates formes (triplets, quadruplets) :

$$2D^{c}(s) = \{(i,j)|(i,j) \in 2D(s), s[i] \bullet s[j] \in PC\}$$

Ces annotations sont incomplètes et imprécises si nous cherchons à décrire complètement les interactions entre nucléotides d'une molécule d'ARN mais elles reflètent les capacités de calculs des algorithmes de prédiction de structure. Elles sont incomplètes car elles ne tiennent pas compte de certains liens chimiques entre nucléotides qui contribuent grandement à la stabilité des ARN. L'empilement des bases azotées est négligé de même que les ponts hydrogènes qui ne forment pas des paires de nucléotides. Elles sont imprécises car elles excluent spécifiquement les interactions entre bases qui ne sont pas des paires, en particulier les triplets ne sont pas considérés et il faut choisir l'une ou l'autre des interactions d'un triplet. Un autre aspect de l'imprécision réside dans l'absence d'annotation des faces impliquées dans la relation entre nucléotides.

Si la structure tertiaire est une structure en 3 dimensions et la structure primaire une représentation en 1 dimension de la molécule, on attend de la structure secondaire qu'elle soit représentable en 2 dimensions, sur le plan. Évidemment, si les lignes ont le privilège de se croiser, il n'y a pas de problème hormis la catastrophe de la surcharge. Les liens entre nucléotides des molécules d'ARN sont souvent non conformes à des dessins dont les lignes ne se croisent pas. Ceci est dû aux triplets, aux interactions atypiques et aux pseudo-nœuds.

On utilise souvent un mot de parenthèses bien balancées de même longueur que la séquence d'ARN pour décrire sa structure 2D (ou $2D^{c}$). La représentation des structures $2D^{+}$ dépasse les capacités de cette représentation. Chaque caractère d'un tel mot donne

l'information 2D du nucléotide correspondant dans la séquence. Nous parlerons d'une *décoration* de la séquence. Ces mots sont sur l'alphabet $\Sigma_{PP} = \{'.', '(', ')'\}$ et sont tels qu'à toute parenthèse ouvrante correspond une parenthèse fermante. Le point correspond à un nucléotide qui ne participe pas à une paire. La parenthèse ouvrante est assignée à un nucléotide dont le partenaire de pairage est situé à sa droite et la parenthèse fermante est assignée à un nucléotide dont le partenaire est situé à sa gauche. Ainsi pour la séquence $s \in \Gamma_{ARN}^{+}$, avec $(i,j) \in 2D(s)$, la paire $s[i] \bullet s[j], i < j$, le mot $\lambda \in \Sigma_{PP}^{|s|}$ décorant s est tel que $\lambda[i] = '('$ et $\lambda[j] = ')'$ et avec $1 \leq k \leq |\lambda|$, si $\lambda[k] \notin \{'(', ')'\}$ alors $\lambda[k] = '.'$. Il s'ensuit que λ est un mot bien parenthèsé comprenant autant de parenthèses ouvrantes que fermantes et qu'aucun de ses préfixes ne compte plus de parenthèses fermantes qu'ouvrantes.

Ces mots sont en grands nombres et leur décompte croit à une allure exponentielle avec leurs longueurs. Si l'on fait fi des contraintes de proximités dans les boucles, le nombre de structures pour une séquence longue de n est le $n^{ième}$ nombre de Motzkin ($M_{n+1} = M_n + \sum_{i=0}^{n-1} M_i M_{n-i-i}$). Une discussion plus complète est donnée par (Waterman 1978; Zuker and Sankoff 1984).

0.2.4 Prédiction de structure

Idéalement il serait possible d'inférer les mécanismes d'action et les fonctions des ARN étant données leur séquence. Et en fait, à mesure que grandit le catalogue des ARN connus, la comparaison de séquences nous approche de ce but (Burge, Daub et al. 2013). Mais le besoin de comprendre les mécanismes, la récente explosion du nombre des séquences, la découverte de classes de séquences dont les fonctions restent inconnues et le désir de construire des ARN synthétiques font que cette approche restera insuffisante. Les approches courantes cherchent plutôt soit à modéliser la structure tertiaire ou à prédire la structure secondaire. Nous ne nous intéresserons pas ici à la modélisation 3D.

Plusieurs logiciels de prédiction de structure d'ARN ont été publiés. Bien qu'il existe une certaine variété dans les approches utilisées, les principales sont basées sur la

minimisation d'une fonction d'énergie ou sur la covariance des bases formant des paires dans des séquences homologues obtenues d'espèces différentes (Rivas 2013). Les approches basées sur la covariance (par exemple (Nawrocki and Eddy 2013)) ne sont pas très utiles pour nous puisque nous désirons traiter des molécules dont les nombres d'homologues connus ne les justifient pas. Les méthodes qui minimisent une fonction d'énergie quant à elles sont utilisables en absence de quelconque homologue (nonobstant la base de données d'apprentissage) permettant d'approcher les structures en absence de toute autre information que la séquence.

0.2.4.1 Fonctions d'énergie

Au moins deux fonctions d'énergie sont disponibles. L'une dont il a déjà été fait mention, est basée sur des mesures directes d'énergies. L'autre, plus récente, basées sur la compilation d'occurrences des modules structuraux cycliques dans les modèles 3D publiés (Parisien and Major 2008).

0.2.4.1.1 Mesures directes

Nous avons déjà mentionné des travaux expérimentaux visant à établir la contribution énergétique des liens chimiques formés lors du repliement des molécules d'ARN (Mathews, Sabina et al. 1999). L'approche consiste à mesurer directement la quantité de chaleur, notée q, minimale causant la dénaturation de doubles brins d'ARN de séquences données. Cette chaleur est une forme d'énergie et est spécifiée en Joules ou en calories. Le système auquel on injecte cette énergie peut augmenter en température, augmenter en volume, transformer sa structure, etc. Certains systèmes n'engagent pas de changements de pression, de volume, d'activité électrochimique, etc, sur certaines plages de température. Dans ces conditions, q, l'énergie fournie, cause au système (a) un changement de température et (b) un changement interne. Notons q_s la partie de q qui conduit un ARN de séquence s à changer son état interne. Si la structure de s est unique et bien définie alors q_s correspond à l'énergie nécessaire pour que s perde sa structure. Plus q_s est grand et plus la structure de s est *stable*. La quantité q_s est l'inverse additif de

l'énergie libre de la molécule, l'énergie qui est disponible pour effectuer du travail juste avant que la molécule perde sa structure. Un ensemble de méthodes expérimentales est disponible pour effectuer cette mesure soit en mesurant la température à laquelle le double brin subit la dénaturation ou soit en mesurant directement la quantité de chaleur qu'il faut injecter pour que cette dénaturation se produise. Lorsque ces mesures sont effectuées dans certaines conditions et qu'elles sont normalisées pour un certain nombre de molécules, on parle de la variation de *l'énergie libre de Gibbs* notée G.

Un grand intérêt pour nous de cette mesure vient de ce qu'elle lie directement la constante d'équilibre (K) de deux états ($A \leftrightharpoons B$) à l'énergie libre dans

$$K = \frac{[A]}{[B]} = e^{-\frac{\Delta G}{RT}}$$

les crochets indiquent la concentration molaire de A et de B, R est la constante des gaz parfaits et T est la température. RT et ΔG ne sont en fait que des facteurs de normalisation dont l'évaluation nous donne $q_A - q_B$, la différence d'énergie libre de la molécule dans les deux états. L'exponentielle dans $K = e^{\Delta q}$ est une conséquence de l'entropie, laquelle dépend du nombre de micro états dans un système. Ici, le système n'a que deux états mais très souvent, le nombre d'états est plus considérable et est nommé la fonction de partition (notée Z). (voir par exemple (Van holde 1985) pour une introduction à ces concepts).

Ainsi, en connaissant la quantité relative des espèces on peut établir l'énergie nécessaire pour passer de l'une à l'autre. Lors de son repliement la molécule d'ARN libère une certaine quantité de cette énergie car il est connu que les systèmes tendent vers les états accessibles les plus stables. En particulier, la formation de ponts hydrogènes et l'empilement des nucléotides contribuent à la libération d'énergie lors du repliement de l'ARN et l'organisation pour laquelle G est minimale est naturellement favorisée. Une méthode d'importance, la calorimétrie différentielle, permet d'évaluer q pour une séquence donnée dans un environnement contrôlé. Prenons deux échantillons de liquide dans les cellules dont l'environnement est contrôlé A et B mais à l'un d'entre eux (la cellule B), ajoutons un double brin d'ARN d'intérêt. Ajoutons de la chaleur aux deux cellules jusqu'à

ce qu'elles atteignent une température cible. La différence de chaleur $\Delta q > 0$ injectée dans les cellules doit avoir été absorbée par l'ARN de la cellule B et cette chaleur correspond à ΔG^t sur cette plage de température et pour cet ARN. Notons que ces phénomènes sont réversibles et que la quantité de chaleur émise ou absorbée dans les deux directions doivent être l'exact opposé l'un de l'autre. Ainsi on peut parler de libération ou d'absorption d'énergie selon qu'on considère le repliement ou la dénaturation de l'ARN. En effectuant un grand nombre de mesures sur des séquences différentes, l'équipe de Turner (Turner and Mathews 2010) a construite une base de données décrivant le comportement thermique de nombreuses telles expériences.

Étant donné un modèle de l'énergie du repliement des ARN, il est possible d'écrire une équation décrivant la participation de chaque paramètre à modéliser et d'obtenir leurs valeurs par régression. Minimalement on voudra assigner des paramètres aux types de paires de nucléotides formées, leurs nombres ainsi qu'aux nucléotides non impliqués dans les paires et leurs nombres (notamment à cause de leurs tendances à l'empilement). Pour que le modèle soit utilisable en prédiction de structure, le nombre de paramètres doit inclure non seulement les nucléotides impliqués dans les paires mais également les contextes de séquence dans lesquels on les trouve. Le modèle dominant est le modèle des *plus proches voisins* (*nearest neighbors*) dans lequel une paire de nucléotides est paramétrée en fonction de l'identité des nucléotides qui jouxtent la paire que ceux ci soient appariés ou non (Turner and Mathews 2010). Il doit également donner des valeurs d'énergie pour les brins non appariés, les boucles internes et terminales de différentes longueurs et séquences.

0.2.4.1.2 Fréquences de cycles

La disponibilité des coordonnées 3D de plusieurs molécules cristallisées ou déterminées par RMN (Berman, Westbrook et al. 2000) a permise l'énumération directe des structures locales qu'accomplissent les ARN dans ces conditions expérimentales. Cette approche donne l'avantage d'observer directement l'organisation des nucléotides dans des structures réelles. La capture de l'environnement local des paires de nucléotides se fait en

décomptant des unités structurelles stables, minimalement suffisantes pour assurer leurs structures. Ce rôle est assuré par le concept de motifs cycliques de nucléotides (NCM : *Nucleotide Cyclic Motifs*) (Parisien and Major 2008). L'hypothèse sous-jacente est que les interactions entre nucléotides, comprenant les ponts hydrogènes et l'empilement des bases, sont complètement déterminées par les composantes trouvées dans ces organisations. Ainsi, les NCM forment les blocs fondamentaux dont peuvent être recomposées les structures d'ARN, un peu à la manière de blocs LÉGO. Les NCM d'usage courant sont composés de nucléotides extraits d'un ou de deux brins et ils sont formés d'autant de paires de bases que de brins. (détails au chapitre 1). Ils forment des cycles sur le graphe que forme l'union de la structure secondaire de la molécule d'ARN et de sa séquence primaire. C'est à dire un graphe dont les nœuds sont des nucléotides et les arêtes sont soit la paire entre deux bases ou le lien phosphate unissant les nucléotides au long de la séquence. Les NCM sur un seul brin décrivent les boucles terminales alors que les NCM sur deux brins correspondent à des empilements de bases avec ou sans erreur formant des renflements dans les hélices (ou *bulges* en anglais). L'énumération des différentes séquences dans ces cycles est comptabilisée dans une liste d'occurrences. Leurs cooccurrences sont également décomptées. Ces fréquences d'occurrences sont utilisées comme probabilité *a priori* de formation de ces structures dans de nouvelles séquences. L'énergie de Gibbs des structures décrites par des NCM peut être déduite de la fréquence en appliquant la définition de l'énergie de Gibbs où la probabilité d'un état parmi l'ensemble des états possibles est mis en correspondance avec l'énergie libre de cet état.

Définition (Pseudo-énergie) : Soit $p(x)$ la fréquence d'apparition d'un NCM x au sein d'une base de donnée, la pseudo-énergie de Gibbs associée à x est :

$$\Delta G(x) = -RT\ ln\big(p(x)\big)\blacksquare$$

La prédiction de structures probables pour une séquence consiste donc à identifier des arrangements permis de NCM dont la somme des énergies de Gibbs est minimale. La formulation exacte de cette fonction d'énergie ne sera donnée qu'à la section 1.4 car il faut encore considérer les énergies des paires de bases et du chevaugement des NCM.

Les structures étant prédites par recherche de minimums dans les fonctions d'énergie, il est possible de lister un ensemble de structures probables et d'associer à chacune d'entre elle une valeur d'énergie libre de repliement. Les structures dont l'énergie est la plus faible sont donc d'énergie libre minimale ou *minimum free energy* (*MFE*). Par abus de langage, on parle souvent de MFE pour référer aux structures dont l'énergie libre correspond à la MFE. Les autres structures proposées par l'algorithme sont dites sous-optimales.

0.2.4.2 MC-Flashfold

Comme mentionné plus tôt, l'espace des solutions est grand et l'énumération est en général impossible. La minimisation se fait souvent par programmation dynamique ou par une variante sur le thème (Mathews, Moss et al. 2010). MC-Fold (Parisien and Major 2008) utilise plutôt une suite d'étapes algorithmiques dont voici les deux principales. D'abord un ensemble de fragments de tiges (doubles hélices) stables est calculé. Puis, les fragments compatibles sont assemblés. Or seule la seconde étape de ce calcul est accomplie par programmation dynamique. Ces étapes sont très couteuses en mémoire et en temps de calcul. Cette approche originale est due à la difficulté perçue d'adapter le modèle d'énergie des NCM à la programmation dynamique. Dans ce modèle la probabilité d'existence d'un cycle dépend de ses deux voisins et cela se prête naturellement à un modèle markovien pour modéliser chaque tige. En pratique, nous avons besoin de traiter des séquences plus longues que ce que permet MC-Fold et nous avons également besoin de traiter des nombres variables (et grands) de structure sous-optimales à chaque fois.

Il est possible toutefois de transposer l'algorithme en programmation dynamique à MC-Fold. L'énumération des solutions sous-optimales se fera par retour en arrière (*backtracking*) sur les solutions satisfaisant un seuil d'énergie (le seuil de Waterman-Byers (Waterman and Byers 1985)). Or ce retour en arrière doit obéir aux mêmes règles que celles utilisées pendant la programmation dynamique. Afin de minimiser les erreurs de codage il est nécessaire d'abstraire les règles dans une grammaire et d'utiliser systématiquement cette grammaire pour construire d'une part les équations de programmation dynamique et d'autre part les règles du retour en arrière. Cette approche

systématique et l'implantation d'une version accélérée de MC-Fold sont décrites au chapitre 1.

0.2.4.2.1 Structures dynamiques et métrique de comparaison

Une fois les ensembles de structures sous-optimales obtenues pour un certain nombre de séquences, nous voudrons les comparer entre eux, estimer leurs similarités et possiblement donner une classification de leurs relations. Pour cela nous compilons les structures sous-optimales en une signature de la dynamique donnant pour chaque base d'une séquence différentes statistiques quant à ses partenaires d'appariement. Nous pouvons comparer deux à deux les signatures disponibles en comparant les statistiques des positions correspondantes. Si les séquences ne sont pas de même longueurs, il est possible d'utiliser la programmation dynamique pour faire cette comparaison. La programmation dynamique garantissant la meilleure valeur de comparaison pour la paire agira comme une métrique et les tables de distances paire à paire pourront être utilisées dans les procédures des groupages telles que le *clustering hiérarchique*. Nous décrivons ces métriques au chapitre 2 et nous les utilisons dans les chapitres 2 et 3.

Des signatures ont déjà été utilisées dans une version plus simple ne comptabilisant que les occurrences de paires de bases et seulement les paires canoniques prédites (Fontana, Stadler et al. 1993) pour comparer des séquences d'ARN (Sabarinathan, Tafer et al. 2013b). Plus récemment des méthodes basées sur la fonction de partition (une table de probabilité des paires de bases) (Halvorsen, Martin et al. 2010; Sabarinathan, Tafer et al. 2013a; Sabarinathan, Tafer et al. 2013b) ont été utilisées dans des buts qui ne sont pas étrangers aux nôtres.

0.2.4.3 Variations sur le thème de la programmation dynamique

Les premiers logiciels de prédiction de structure $2D^c$ disponibles des années 1970 (par exemple (Pipas and McMahon 1975; Studnicka, Rahn et al. 1978)) consistent en une série d'étapes visant à combiner au mieux les hélices prédites en minimisant un certain

nombre de paramètres mesurés ou simplement estimés *au mieux* à partir des travaux clé publiés au début des années 70 (Delisi and Crothers 1971; Tinoco, Uhlenbeck et al. 1971).

Les premiers logiciels de prédiction de la MFE par programmation dynamique sont apparus en 1978, Smith et Waterman, connus pour leur version de l'alignement de séquences par programmation dynamique ont publié un algorithme pour la minimisation de structure secondaire d'ARN par programmation dynamique (Waterman and Smith 1978) et l'algorithme de Nussinov aurait été découvert simultanément (Nussinov, Pieczenik et al. 1978; Nussinov and Jacobson 1980). Ces deux algorithmes n'utilisent qu'une seule table. Zuker et Stiegler (Zuker and Stiegler 1981) proposent une solution sur deux tables afin de permettre l'implémentation du modèle des plus proches voisins lequel permet d'intégrer l'énergie d'empilement des paires de bases dans le calcul. La fin de la décennie 1980 voit apparaitre l'énumération interactive de plusieurs structures sous-optimale (Zuker 1989). D'autres implantations du modèle voient le jour pendant les années 1990, en particulier RNAstructure (Mathews, Sabina et al. 1999) et RNAFold (Hofacker, Fontana et al. 1994). Wucthy (Wuchty, Fontana et al. 1999) donne une solution systématique pour l'énumération rapide et complète des structures sous-optimales. PKNots (Rivas and Eddy 1999) donne une version de la programmation dynamique qui tient en compte les pseudo-nœuds.

McCaskill (McCaskill 1990) donne une simple transformation de l'algorithme de minimisation en programmation dynamique permettant le calcul de probabilité de chaque paire de base dans une séquence parmi toutes les structures possibles. Il est possible de déduire certains paramètres thermodynamique de son analyse et de comparer des séquences en termes de profils qui en sont extraits (Bonhoeffer, McCaskill et al. 1993). Cette table peut-être très utile et a servi par exemple à l'identification de zones dépourvues de structures pour la sélection d'oligonucléotides de puces d'ADN (Ding and Lawrence 2001). Dans l'approche Sfold (Ding, Chan et al. 2005) cette table est utilisée comme probabilité *a priori* dans un nouveau calcul des structures suivant un processus d'échantillonnage dont le résultat se compose de structures représentatives des groupes des structures dominantes. Il est facile dans ce paradigme d'introduire d'autres probabilités *a priori* sur les paires de bases obtenues avec des méthodes expérimentales telles que l'attaque chimique ou autre.

Plus récemment la recherche de minima locaux (Lorenz and Clote 2011) par exemple, ou la mesure du changement induit dans la table par des mutations ponctuelles (Halvorsen, Martin et al. 2010; Sabarinathan, Tafer et al. 2013a) exploitent cette approche.

En 2000, la publication de la structure 3D du ribosome (Ban, Nissen et al. 2000; Wimberly, Brodersen et al. 2000) est venue changer les cartes il devint possible de concevoir la prédiction de structure d'ARN comme un problème d'apprentissage plutôt qu'un problème de minimisation. Ces méthodes, basées sur la vraisemblance maximale conditionnelle (CML : *conditional maximum likelihood*) ou l'algorithme EM (*expectation maximisation*), dépendent étroitement de la grammaire utilisée car d'une part les paramètres appris sont associés à ses transitions et d'autre part la non-ambigüité du langage est capitale pour le bon fonctionnement de ces systèmes. Huit grammaires sont évaluées dans ce contexte par (Dowell and Eddy 2004) où les différences entre les grammaires influencent leurs capacités relatives à prédire des structures sans atteindre tout à fait le niveau de qualité des prédictions faites en minimisant l'énergie libre. Une grammaire plus sophistiquée n'améliore pas le rendement (Nebel and Scheid 2011). Le logiciel MC-Fold (Parisien and Major 2008) échantillonne les structures 3D sans toutefois s'inscrire dans la mouvance de ces méthodes. L'échantillonnage cherche à remplacer le modèle des plus proches voisins par celui des NCM. L'idée est que ces motifs capturent les interactions d'empilement et de ponts hydrogènes, eux-mêmes dépendants des particularités géométriques des nucléotides, mieux que simplement les empilements de paires de nucléotides. Le modèle des NCM n'a donc pas été conçu en fonction d'une quelconque grammaire et l'implantation du logiciel n'a pas été conçue autour du paradigme de la programmation dynamique. Or il s'avère que les grammaires utilisées dans ces méthodes (Dowell and Eddy 2004; Nebel and Scheid 2011; Rivas 2013) ne sont pas conçues pour capturer l'information des NCM.

0.2.4.4 Algebraic Dynamic Programming

Il existe une étroite relation entre langages, grammaires, piles et la programmation dynamique. L'algorithme CYK (du nom de ses trois découvreurs indépendants vers la fin

des années 1960; voir par exemple (Sikkel 1997)) décide l'appartenance d'un mot w dans un langage hors-contexte $\mathcal{L}$ décrit par une grammaire $\mathcal{G}$ en exploitant la programmation dynamique et donne une suite d'applications de règles qui le produit le cas échéant. L'idée de cet algorithme est de construire l'ensemble des équations booléennes de récurrence d'un algorithme en programmation dynamique $\mathcal{A}$ à partir des règles de production de la grammaire et de déclarer $w \in \mathcal{L}$ ssi $\mathcal{A}(w) \neq 0$. Le cas échéant on peut obtenir une suite productive de règles acceptant w en utilisant la méthode des pointeurs à rebours (voir par exemple (Cormen, Leiserson et al. 1989)). Mais en associant à chaque règle de $\mathcal{G}$ une équation de récurrence dans $\mathbb{R}$ (les réels) plutôt que dans $\mathbb{B}$ (les booléens), il est possible d'augmenter la portée de cette approche pour aborder des problèmes d'optimisation et non seulement des problèmes de décision et de *parsing*. Cela permet notamment d'obtenir un arbre de poids optimal décrivant la syntaxe d'un mot utilisable dans plusieurs situations dont le traitement en langage naturelle et le repliement d'ARN. Une difficulté associée à ces systèmes pour certaines catégories de problèmes consiste à trouver l'écriture adéquate des tables de $\mathcal{A}$ à partir de $\mathcal{G}$ lorsque $\mathcal{G}$ comprends un grand nombre de règles ou demande un ordonnancement de celles-ci qui soit difficile à conceptualiser. Dans ces situations en particulier, un système capable de générer un programme d'ordinateur à partir d'une grammaire et de ses fonctions à optimiser pourrait s'avérer utile. Avec $\mathcal{W}$ les fonctions de poids, la paire $\langle \mathcal{G}, \mathcal{W} \rangle$ suffit pour décrire le problème et le professeur Robert Giegerich à baptisé ces paires et leurs traitement automatique *Algebraic Dynamic Programming* (Giegerich and Meyer 2002). Le système *ADP* (Giegerich, Meyer et al. 2004) devenu *Bellman's GAP* (Sauthoff, Mohl et al. 2013) est un module dans le langage Haskell qui accepte une paire $\langle \mathcal{G}, \mathcal{W} \rangle$, une instance du problème et exécute la minimisation. Il est possible d'obtenir une solution optimale et optionnellement un certain nombre de solutions sous-optimales. Plusieurs stratégies semblent en cours visant à l'optimisation des performances de ce système (voir par exemple (zu Siederdissen 2012)). Ce système entre autre a été utilisé récemment pour produire le logiciel rnaWolf et MC-Fold-DP (Honer zu Siederdissen, Bernhart et al. 2011; zu Siederdissen 2013). Le premier augmente la grammaire de MC-Fold pour tenir compte des triplets de nucléotides et le second utilise le concept des NCM en programmation dynamique. L'équation minimisée par MC-Fold-DP

(Honer zu Siederdissen, Bernhart et al. 2011) ainsi que la grammaire raportée comme étant basée sur l'approche de RNAsubopt (Wuchty, Fontana et al. 1999) semblent adéquats. Toutefois, d'une part nous n'avons pas été en mesure de reproduire les résultats de MC-Fold avec ce système (les auteurs non plus d'ailleurs) et d'autre part l'exécution est considérablement moins rapide que nos attentes. Il est possible que des révisions ultérieures fassent concorder ce système avec les prédictions de MC-Fold et que l'augmentation de la performance de Bellman's GAP améliorent la vitesse d'exécution de celui-ci.

0.3 Notre approche

Au niveau macroscopique la température d'un objet en équilibre thermique est constante car elle dégage autant d'énergie qu'elle en reçoit mais au niveau moléculaire les choses sont différentes. Les quanta d'énergie absorbés et émis d'une molécule changent son énergie interne constamment et ce flux est imprévisible, assimilable à du bruit quantique. Le niveau moyen de ces échanges semble être dans le même ordre de grandeur que l'énergie d'un pont-H. Une molécule change continuellement de structure autour de minima locaux et les solutions presque optimales de MC-Fold captent ces structures (Dethoff, Petzold et al. 2012). De plus les interactions entre molécules induisent des changements de l'énergie interne des partenaires dont l'ampleur dépend de la compatibilité des surfaces. L'ensemble des structures d'ARN accessibles à une température donnée devrait refléter les fonctions biologiques de cette molécule et ces ensembles sont à une certaine distance inconnue Δ kcal/mol de l'optimal. Nous utilisons le modèle des NCM pour obtenir ces ensembles dont nous extrayons des signatures que nous comparons afin de déterminer si elles peuvent avoir la même activité biologique. Bien entendu, cette mesure d'activité dépend ultimement de la position des atomes et non simplement de l'allure de la dynamique 2D. D'autre part, notre notion de *dymamique* n'inclue pas la cinétique de repliement des ARN et cela constitue une limite importante de nos efforts. Le lecteur intéressé par la prédiction de la cinétique du repliement des ARN peut consulter entre autres les travaux d'Isambert (Xayaphoummine, Bucher et al. 2005b; Isambert 2009). Néanmoins nous avons observé une corrélation forte entre la *dynamique* et la perte de

fonction de mutants de miR-125a (chapitre 2) et nous avons appliqué cette méthode à la recherche de modules de dynamique conservée dans des ensembles de séquences (chapitre 3).

0.3.1 Accélération de MC-Fold

Le logiciel MC-Fold (Parisien and Major 2008) calcule les structures secondaires d'énergies minimales selon le modèle des Motifs Cycliques de Nucléotides (NCM) à partir de la séquence. Une des caractéristiques importantes du modèle NCM est sa capacité à capturer les paires de nucléotides dites non-canoniques (celles qui ne sont pas tirées de {AU, CG, GU}). Il est étonnant, en vue de leurs occurrences fréquentes que les outils de prédiction des structures secondaire d'ARN n'en tiennent pas compte (sauf exceptions (Rivas 2013)). Cette capacité à prédire les paires non-canoniques est particulièrement appréciée lors du modelage 3D des ARN, notamment avec le logiciel MC-Sym (Major, Turcotte et al. 1991) lequel nécessite la description de l'ensemble des paires de bases (canoniques ou non) à modéliser. Il est également nécessaire de considérer ces types de paires de nucléotides pour toute analyse fine basée sur la structure secondaire ou, comme nous le verrons dans les prochains chapitres, pour évaluer les probabilités d'appariement de chaque nucléotide des ARN. Malheureusement, MC-Fold n'est pas très rapide d'exécution et utilise de grandes quantités de mémoire vive et cela limite son usage à de courtes séquences d'ARN (environ moins de 100 nt.) et à un nombre relativement petit de structures sous-optimales.

Au chapitre 1, nous décrivons MC-Flashfold, une version accélérée de MC-Fold qui utilise moins de mémoire vive et qui est capable d'énumérer rapidement un grand nombre de structures sous-optimales. Grâce à MC-Flashfold, il est dorénavant possible de traiter les plus longs ARN biologiques structurés connus tels que les sous-unités de ribosomes (quelques milliers de nucléotides) ou d'énumérer des dizaines de millions de structures sous-optimales de riboswitchs sur des ordinateurs ordinaires contemporains. L'accélération suit l'approche du logiciel RNAsubopt (Wuchty, Fontana et al. 1999), un champion de vitesse pour ce qui est du calcul déterministe de structures secondaires d'ARN dont la

fonction objectif est basée sur le modèle d'énergie thermodynamique mentionné en introduction et ne calcule donc de manière satisfaisante que les paires de bases canoniques.

Comme dans RNAsubopt, les structures secondaires sous-optimales sont identifiées comme des chemins de retour en arrière (backtrack) dans les tables de programmation dynamique en utilisant le critère de Waterman-Byers (Waterman and Byers 1985) comme seuil. Ceci impose au programmeur de parfaitement coordonner les règles utilisées pendant la passe de programmation dynamique avec celles utilisées lors de la passe de retour en arrière et cette tâche est sujette à l'erreur pour des règles non triviales. Une manière de réaliser ceci consiste à définir une grammaire dont on dérive les règles de programmation dynamique ainsi que celles utilisées lors du retour en arrière. Certains outils font ceci automatiquement (Sauthoff, Mohl et al. 2013). Nous décrivons ici une méthode pratique permettant de systématiser l'élucidation des règles pour les deux passes algorithmiques ainsi qu'un certain nombre d'exemples de l'application de cette méthode. Ceci serait inutile sans la prescription d'algorithmes pour le retour en arrière. Cette passe du calcul est différente selon la nature des règles de la grammaire. Lorsque les règles de production ne comportent jamais plus d'un non-terminal à droite (le langage hors-contexte correpondant est linéaire), l'algorithme ne nécessite qu'une seule pile. Mais lorsque certaines d'entre elles ont au moins deux non-terminaux à droite, deux piles sont nécessaires. Cette approche est décrite dans un premier temps dans un langage qui se veut intuitif, puis de manière formelle. Trois exemples d'utilisation de l'approche sont donnés en appendice sur des problèmes de difficulté croissante.

Nous développons une grammaire de prédiction des structures secondaires sur le modèle des NCM. La version longue de cette grammaire comporte 25 non-terminaux et quelques règles dont le côté droit comprend deux non-terminaux, nous aurons donc besoin de la version à deux piles de l'algorithme de retour en arrière.

De plus nous introduisons un algorithme qui augmente l'utilisabilité du logiciel en effectuant la détection automatique du seuil de Waterman-Byers correspondant à un nombre donné de structures sous-optimales prescrit.

Il est possible d'améliorer la grammaire avec de nouvelles règles sans détruire le modèle et des travaux éventuels viseront à augmenter la précision et l'exactitude des prédictions en augmentant la portée topologique des règles. Notons que d'autres problèmes de chemins sous-optimaux peuvent bénéficier de notre approche.

0.3.2 Signature de la dynamique

Au chapitre 1 nous verrons donc comment accélérer MC-Fold afin d'obtenir rapidement des ensembles de structures sous-optimales pour des séquences d'ARN. L'idée étant que ces ensembles capturent la dynamique des ARN. Nous décrivons maintenant les outils pour comparer ces ensembles de structures. Ensuite nous donnerons des évidences montrant que la perte de maturation des microARN peut être causée par des changements de dynamique induits par des mutations.

0.3.2.1 Comparer la dynamique des ARN

Nous tentons de mesurer la différence de fonction biologique entre des séquences proches. En particulier nous voulons connaitre l'impact de mutations introduites dans une séquence sur la fonction de la séquence originale. Dans ce qui suit, nous utilisons le symbole → de manière volontairement imprécise, tantôt signifiant *détermine* et tantôt signifiant *prédit.* Soit nous savons que la structure tridimensionnelle (3D) d'une molécule définit sa fonction biologique (*F*). Nous pouvons donc écrire :

$$3D \rightarrow F$$

Mais nous aimerions pouvoir inférer cette connaissance d'une molécule d'ARN simplement à partir de sa séquence (*s)* :

$$f(s) \rightarrow F$$

Cela est hors de portée pour l'instant. Pouvons nous obtenir un prédicteur de structure 3D à partir de la séquence? Un tel prédicteur n'existe pas encore. Une telle fonction serait :

$$f(s) \rightarrow 3D$$

Nous disposons de modélisateurs de structure tridimensionnelle qui permettent d'obtenir un ensemble de modèles 3D sur entrées de séquences annotées (où les annotations sont la structure secondaire, 2D). Un de ces modélisateurs (M) est MC-SYM.

$$M(s, 2D) \rightarrow \{3D\}$$

Où les accolades indiquent qu'on obtient en ensemble de modèles 3D. Nous disposons également de prédicteurs (P) de structures 2D. Ceux-ci génèrent des ensembles d'assignations de paires probables pour une séquence d'ARN fournie en entrée.

$$P(s) \rightarrow \{2D\}$$

Nous disposons donc d'outils chainables dont la sortie est un ensemble de modèles de structures 3D :

$$P(s) \rightarrow \{2D\} \rightarrow \{3D\}$$

L'étape suivante nous est toutefois interdite car nous ne savons pas comment déduire la fonction d'un ensemble de modèles 3D :

$$P(s) \rightarrow \{2D\} \rightarrow \{3D\} \overset{?}{\rightarrow} F$$

Toutefois, puisque dans ce modèle la structure 3D est complètement déterminée par la séquence annotée, on peut écrire:

$$P(s) \rightarrow \{2D\} \overset{?}{\rightarrow} F$$

On ne sait toujours pas comment appliquer cette implication mais nous pouvons espérer de trouver un comparateur $\otimes$ qui mette en évidence les variations de la fonction biologique en calculant les changements induits dans les {2D} par des variations dans la séquence donnée en entrée.

$$P(s) \otimes P(s') \rightarrow \{2D\} \otimes \{2D\} \overset{?}{\rightarrow} \Delta F$$

Ainsi, cherchons une fonction *T* qui prend deux séquences en entrée, applique l'opérateur sur la sortie de P et donne une mesure liée à la variation dans la fonction biologique de la molécule.

$$T(s, s') \overset{?}{\rightarrow} \Delta F$$

Nous implantons T en deux étapes. D'abord nous extrayons une signature σ de {2D} pour *s*. Puis nous comparons les signatures entre elles. Différentes stratégies sont possibles. D'une part l'extraction de la signature peut correspondre à différentes caractéristiques de l'ensemble de structures et d'autre part, la comparaison des signatures peut être implantée de différentes manières. L'une mène à une mesure de distance et l'autre à une mesure de similarité. Il est toujours possible dans ces conditions de transformer une mesure de similarité en mesure de distance, lorsque on considère un groupe de séquences que l'on compare toutes entre elles en leurs soustrayant le MAX des similarités. Nous verrons les différentes stratégies utilisées ainsi que les impacts de ces choix en pratique.

0.3.2.2 Mutations et maturation des microARN

Une mutation dans le gène de miR-125a affecte la résistance aux traitements des personnes atteinte de certains cancers du sein et modifie la probabilité de contracter certains cancers (Li, Duan et al. 2009). Cette mutation change la séquence du microARN mature correspondant et l'équipe de Peng Jin à Atlanta a d'abord cru que le miARN mutant n'agissait pas spécifiquement sur le ou les même(s) gène(s) cible(s) que le miARN non mutant. Mais en cherchant plus avant, il leur est devenu évident que le précurseur mutant subit une maturation abortive (Duan, Pak et al. 2007). Ainsi, ce n'est pas le changement des cibles du microARN mature mais bien la disparition de celui-ci qui est responsable des effets de la mutation. Pourtant, l'ARN est bel et bien synthétisé dans le noyau. La mutation bloque donc la maturation de cet ARN et cela doit se produire suite à des changements dans

la structure de celui-ci. Changements qui l'empêchent d'être reconnu par les enzymes responsables de la maturation des microARN.

L'investigation des prédictions de structures pour les deux versions du gène avec le logiciel MC-Fold, n'a pas révélé de différence de structure pour la MFE. Afin d'expliquer structurellement la différence de comportement, certains membre du laboratoire Major, Marc Parisien et Karine St-Onge ont modélisés des structures 3D probables de ces mutants en cherchant en particulier des variations structurales ou électrostatiques qui auraient pu expliquer des différences d'affinités pour certaines des protéines impliquées dans la maturation. Pourtant, rien ne semble y faire, la mutation n'affecte pas les structures les plus probables suffisamment pour trouver là une solution à la maturation abortive.

Nous avons formulé l'hypothèse voulant que la différence entre les structures se cache dans les nombreuses structures sous-optimales acceptées par les variantes de séquences. Pour vérifier cette idée, nous avons conçu notre système permettant la comparaison de groupes de structures sous-optimales d'une part, et d'autre part nos collaborateurs ont réalisé des expériences sur une liste de mutants augmentée. Ainsi, avec les données biologiques concernant 16 mutants (correspondant aux paires de nucléotides possibles impliquant la position mutante) il devient possible de vérifier si la différence biologique induite par les mutations est reflétée dans l'ensemble de structures sous-optimales obtenues par prédiction logicielle.

1 Les Algorithmes de MC-Flashfold

Ce chapitre est divisé en 7 parties :

1. Introduction au chapitre
2. Description intuitive de l'approche systématique
3. Description formelle
4. Le modèle d'énergie des Motifs Nucléotidiques Cycliques
5. La grammaire de MC-Flashfold
6. Processus complémentaires
7. Conclusion au chapitre

1.1 Introduction

Le logiciel MC-Fold calcule les structures secondaires les plus probables étant donnée une description statistique des NCM décrits dans la base de donnée Protein Data Bank (pdb) (Berman, Westbrook et al. 2000). Sa sortie est un ensemble de chaines de caractères de même longueurs que l'entrée et annotées par l'énergie de repliement de l'ARN. Chaque chaine de caractère correspond à une structure secondaire et est produite dans le format *dot-bracket* (une décoration de $s \in \Gamma_{ARN}^{n}$ sur Σ_{PP}^{n}). Malheureusement MC-Fold est lent et utilise beaucoup de mémoire vive le rendant inapproprié à plusieurs tâches. Nous donnons ici quelques définitions sommaires afin de guider le lecteur.

Définition (NCM) : Un motif cyclique nucléotidique ou NCM est un cycle minimal récurent dans les graphes de structure 2D (Fig 1E).

Définition (Annotation de NCM) : Identification de l'ensemble des NCM d'un graphe de structure 2D d'une séquence (Fig 3).

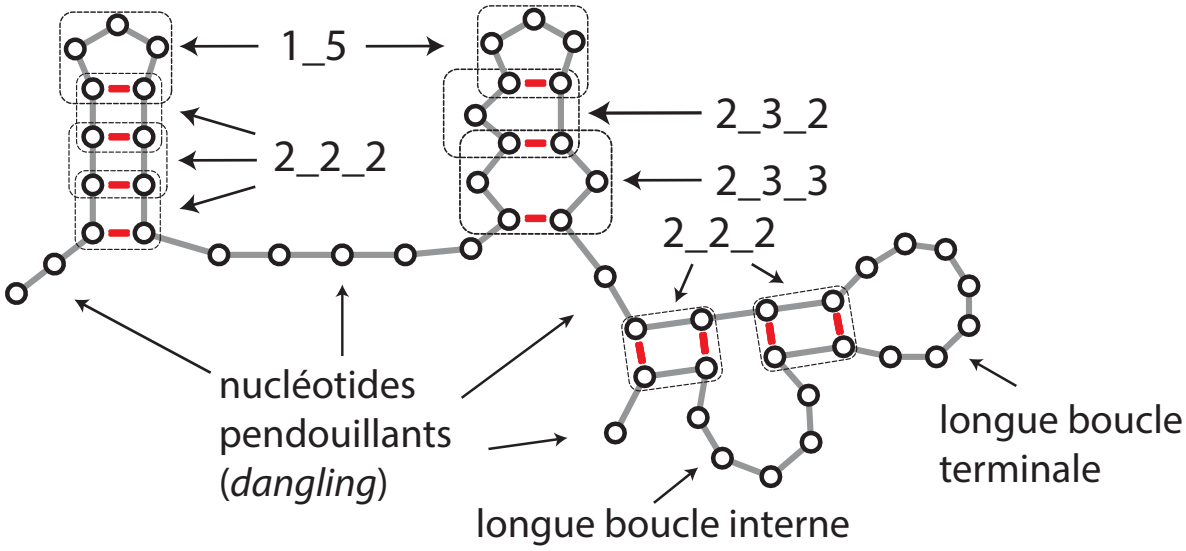

Figure 2: Une structure annotée par ses NCM.

Définition (décoration 2D NCM) : L'unique annotation de paires de bases donnée dans le format parenthèse-point décrite par les NCM de l'annotation.

Définition (MFE) : Décoration 2D de l'annotation NCM de pseudo-énergie mimiale pour une séquence.

Définition (structure sous-optimale) : Décoration 2D d'une annotation NCM dont la pseudo-énergie n'est pas minimale.

En général, il existe plusieurs structures dont l'énergie est proche de la MFE. Nos problèmes sont :

Problème pred-MFE : Identifier une MFE pour une sequence d'ARN donnée; et:

Problème pred-subopt : Identifier les MFE et les structures sous-optimales proches de l'énergie minimale soit en pseudo-énergie ou en nombre. C'est à dire lister les structures dont la difference de pseudo-énergie avec le minimum est inférieure à un seuil donné ou encore lister un nombre donné de structures dont l'énergie s'élloigne du minimum.

Définition (MC-Fold) : Est un premier logiciel solutionant pred-MFE et pred-subopt (Parisien and Major 2008) essentiellement en temps $O(c^n)$.

Définition (MC-Flashfold) : Est un nouveau logiciel solutionant pred-MFE en $O(n^3)$ et pred-subopt en $O(n^3 + n^2\kappa)$ où κ est le nombre de solutions sous-optimales. La mémoire utilisée est $O(n^2)$.

L'ensemble des structures secondaires d'ARN peut être décrit par une grammaire dont chaque terme est associé à une valeur énergétique. Ainsi, puisque l'équation d'énergie peut-être écrite de manière récursive et, puisque l'espace du problème accepte le principe d'optimalité de Bellman (Bellman 1952), la programmation dynamique est applicable et trois implémentations ont été réalisées de manière concourante (MC-Fold2 (Saule, unpublished), RNAWolf (Honer zu Siederdissen, Bernhart et al. 2011) et l'implémentation de Stefanie Schirmer (unpublished)) donnant parfois des accélérations considérables.

Cependant chacun de ces outils souffre d'au moins un problème majeur (temps de calcul excessif, lenteur dans l'énumération des sous-optimaux, incapacité à traiter des séquences longues, manque général de fiabilité). MC-Fold2 est une implantation en programmation dynamique, RNAWolf augmente la définition de structure 2D pour inclure des triplets de nucléotides et autres innovations et l'implantation de Steffie est un travail expérimental basé sur le système Bellman's GAP des grammaires et algèbres d'évaluation (*grammars and evaluation algebras*) (Sauthoff, Mohl et al. 2013). MC-Flashfold est une nouvelle implantation rapide et fiable codée dans le langage C et ce chapitre décrit son architecture algorithmique.

Le gain de vitesse est réalisé en approchant le problème de l'énumération des structures sous-optimales de manière homologue à l'exploration de graphe avec une fonction guide donnant le bénéfice réalisable des choix de chemin à chaque étape. Les algorithmes de la classe A* (A*, fringe, etc) trouvent les chemins les plus courts jusqu'à un nœud donné dans des graphes dont les arêtes sont associées à des coûts (*weighted graphs*) en utilisant l'approche du retour en arrière (*backtracking*) à partir du nœud de départ (Hart, Nilsson et al. 1968). À chaque étape, ces algorithmes sélectionnent la meilleure prochaine étape d'après la valeur d'une fonction heuristique *h()*, un estimé conservateur de la distance du nœud courant à la destination, si bien que la distance totale du chemin ne peut excéder *g()+h()* où *g()* est la valeur du chemin à date. Nous procédons de manière analogue excepté que notre estimation de la distance à la cible est une mesure exacte si bien que la découverte du chemin est linéaire dans le nombre de nucléotides dans la structure. Pour trouver toutes les structures dont la valeur est dans un intervalle Δ de la meilleure distance possible (*Poids Optimal : PO* ou *Minimum Free Energy : MFE*) nous faisons des retours en arrière sur les nœuds de chemins où $MFE + \Delta \leq g() + h()$. Ces valeurs sont des énergies libres au sens de la thermodynamique et la stabilité des structures augmente à mesure que décroit son énergie. Notez que la valeur du seuil (Δ) est positive. La fonction $h()$ est construite sur les tables remplies par un calcul en programmation dynamique et les chemins sont découverts par retour en arrière. Il s'avère que cette approche est presque identique à celle utilisée par Wuchty (Wuchty, Fontana et al. 1999) ce qui explique la vitesse et la

faible consommation de mémoire de RNAsubopt. Nous verrons en détails en quoi les deux approches sont différentes.

L'algorithme est décomposé en deux phases séquentielles distinctes mais les énergies calculées par la première phase (programmation dynamique) doivent exactement correspondre aux énergies calculées pendant la seconde phase (retour en arrière) et cela pose le problème de la correspondance entre les règles appliquées pendant les deux calculs. Wuchty ne s'intéresse pas outre mesure à ce problème, il lui est possible de simplement faire preuve de minutie pendant la programmation. Comme nous le verrons, les règles du modèle NCM sont plus compliquées. Nous avons formalisé une approche systématique (prochaine section) afin de réduire l'impact du problème de la correspondance : nous dérivons les règles utilisées dans les deux phases algorithmiques d'une grammaire commune. Cela nous permet d'associer à chaque règle de la grammaire une règle correspondante en programmation dynamique et une règle correspondante pour le retour en arrière. En particulier, cela permet de documenter adéquatement le code et de le modifier selon les changements apportés à la grammaire.

En premier lieu nous verrons les détails de notre approche et quelques exemples d'applications. Puis nous décrirons le modèle d'énergie des NCM, sa grammaire et les algorithmes associés en programmation dynamique ainsi qu'en retour en arrière tels qu'implantés dans MC-Flashfold. Quelques algorithmes de traitement des sorties du programme seront décrits. Finalement nous considèreront la performance des passes algorithmiques et de l'implantation complète.

1.2 Description intuitive de l'approche systématique

Le processus débute avec une grammaire décrivant le problème sous considération. De cette grammaire nous dérivons systématiquement d'une part les récurrences de la programmation dynamique et les étapes du retour en arrière. Tout changement à la grammaire est ainsi systématiquement traçable dans les deux passes permettant d'assurer leurs coordinations, simplifiant le travail de programmation et minimisant les chances

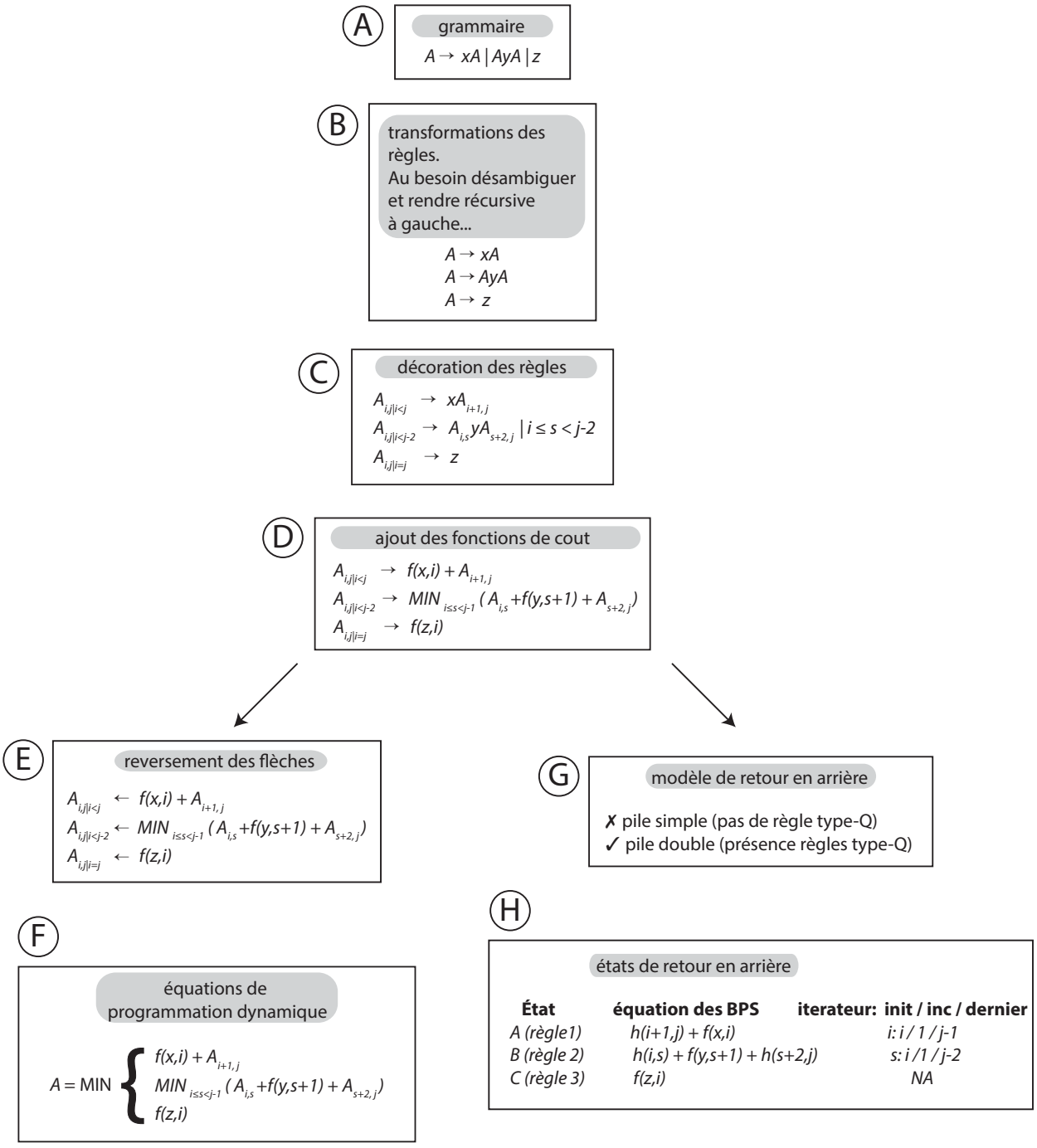

Figure 3. Sommaire des étapes de notre approche.

d'erreurs. Dans cette section, nous décrivons les étapes de la méthode. En annexe, nous la mettons en œuvre avec quelques exemples de difficultés croissantes.

1.2.1 Étapes de l'approche

Notez que, bien qu'il soit entendu que les conditions initiales de toute solution par programmation dynamique sont capitales à la bonne exécution nous n'insisterons pas sur leurs prescriptions exactes puisque cela nécessiterait une documentation encore plus verbeuse qu'elle ne l'est déjà.

Le processus complet est illustré à la Figure 1. Une grammaire est donnée en entrée qui définit le langage des solutions acceptables au problème. On trouve la grammaire en élaborant les règles à partir des mécanismes fondamentaux du problème et l'espace des solutions admissibles. Une grammaire adéquate doit couvrir l'espace des solutions par applications répétées des règles de production. Cette couverture de l'espace des solutions exige une grammaire ambigüe. Parfois la grammaire peut être changée pour une grammaire équivalente produisant les solutions qu'une seule fois et parfois, il est impossible de découvrir une telle grammaire. Cela est une propriété intrinsèque de certains langages hors-contexte (voir page 233 dans (Harrison 1978)). D'autre part, il n'existe pas d'algorithme qui prouverait touijours qu'une grammaire soit dépourvue d'ambigüité (cette preuve est donné en page 202 dans (Aho and Ullman 1972)) et nous sommes pris avec cette redondance de solutions. Il existe bien des algorithmes pouvant déclarer qu'une grammaire est ambiguë ou non, mais ils ne donnent pas toujours une réponse et le problème reste indécidable (Giegerich, Meyer et al. 2004). La plupart des problèmes acceptent des grammaires faciles à identifier et qui cèdent rapidement à la réflexion mais parfois il faut faire preuve d'ingéniosité. Nous donnons quelques exemples en appendice utilisant de telles grammaires. Les types de problèmes qui nous intéressent acceptent des solutions descriptibles comme des annotations le long de séries d'étapes où chaque annotation correspond à un choix pris à l'étape correspondante. Nous parlons alors de *décoration* de l'entrée et cette solution est donnée sous la forme d'un mot $\lambda \in \Sigma^+$ sur un alphabet approprié. Par exemple, avec $s \in \Gamma_{RNA}^n$, la prédiction d'une structure secondaire par MC-

Flashfold est une décoration $\lambda = 2D(s)$ décrite sur Σ_{PP}^{n}. Comme autres exemples, si les solutions décrivent des chemins dans un labyrinthe, elles représenteront les choix faits à chaque bifurcation. La solution à un problème d'alignement de séquences est composée d'annotations de trous dans l'une des séquences ou d'une concordance. À ce niveau d'abstraction, la grammaire n'est pas pondérée mais doit décrire adéquatement la structure de l'espace solution. Cela sera fait à l'étape 4 où un coût (ou un bénéfice) sera associé à chaque application de règle pendant les calculs et la grammaire doit permettre d'ordonner les transformations selon qu'elles sont plus ou moins avantageuses lors de l'évaluation des solutions possibles. La grammaire est décrite en termes de terminaux (T) et de non-terminaux (N) organisés sous forme de règles de production (R) dans un tuple < N, T, R, S > où S est le non-terminal de départ (start) ou alternativement l'ensemble des règles de départ. Ces grammaires peuvent être réduites et la redondance peut parfois être éliminée en utilisant des transformations d'usage commun que nous ne discutons pas avant ici.

Supposons que nous cherchions des solutions dans l'alphabet $\Sigma = \{x, y\}$ sous la forme d'une chaine sur $[xy]^n$ où x et y sont deux états acceptés à l'indice i ($1 < i \leq n$) pour une instance γ d'un problème de longueur n. Un exemple de solution serait *xxxxyxyyxxxyx* et les états pourraient être des directions (gauche et droite) ou de l'information de groupage (inclus et exclus) par exemple. Une grammaire correspondante pourrait prendre la forme :

$$\mathrm{A} \rightarrow x\mathrm{A} \mid y\mathrm{A} \mid x \mid y$$

Dans un premier temps, nous annotons les non-terminaux avec les indices applicables :

$$\mathrm{A}_i \rightarrow x\mathrm{A}_{i\text{-}1} \mid y\mathrm{A}_{i\text{-}1} \mid x \mid y$$

La récurrence de programmation dynamique est obtenue en renversant la flèche

$$\mathrm{A}_i \leftarrow x\mathrm{A}_{i\text{-}1} \mid y\mathrm{A}_{i\text{-}1} \mid x \mid y$$

Puis en convertissant la règle en une fonction de pondération correspondante (donnant dans ce cas une fonction objectif très simple) :

$$A_i = \text{OPT} \{ w_x(\text{`}x\text{'}, A_{i-1}), w_y(\text{`}y\text{'}, A_{i-1}), c_1, c_2 \}$$

Ici OPT pourrait être la fonction MIN dans un problème de minimisation et les c_i pourrait prendre des valeurs nominales adéquates.

Cette récurrence nous donne le meilleur score possible (PO : Poids Optimal) pour l'instance et nous conservons les tables utilisées pendant le calcul parce qu'elles contiennent le PO pour toutes les sous-instances et, comme nous le verrons bientôt, nous utiliserons celles-ci comme valeurs plafonds pour guider le retour en arrière.

Nous générons les solutions par retour en arrière sur toutes les solutions possibles en commençant par la règle la plus abstraite (i.e. : la règle S de la grammaire) en utilisant une pile. Le meilleur score possible (MPR : Meilleur Poids Réalisable avec la solution partielle courante) pour la solution partielle sous considération est :

$$\text{MPR} = g(\text{solution partielle}) + h(\text{sous-instance candidate})$$

Le score de la solution partielle décrite dans la pile (i.e. : $g()$) est combinée à celui de la sous-instance candidate pour obtenir le MPR réalisable avec la solution partielle courante. Si le MPR est à l'intérieur du seuil choisi (Δ) du PO, alors la transition est empilée et l'exploration est continuée avec ce nouveau non-terminal. Toutes les transitions choisies correspondent à un non-terminal dérivé de la grammaire et un indice (ou un ensemble d'indices). Avec notre grammaire, il serait possible de n'utiliser qu'un seul type d'état à mettre sur la pile mais les choses sont un peu plus claires avec une grammaire légèrement modifiée :

$$A_i \rightarrow X_i \mid Y_i \mid x \mid y$$

$$X_i \rightarrow xA_{i-1}$$

$$Y_i \rightarrow yA_{i-1}$$

Le retour en arrière débute en empilant A_n sur la pile et procède tant que la pile n'est pas vide. Les transitions à partir du non-terminal sur la pile dont le MPR $\geq$ PO $- \Delta$

(pour les problèmes de maximisation) sont acceptés en empilant le nouveau non-terminal avec son/ses terminaux émit. Puisque la pile décrit une sous-solution acceptable, lorsque celle-ci est une solution complète elle doit être acceptable et peut être produite en sortie. Lorsque toutes les transitions acceptées à partir du non-terminal sur la pile ont été visitées, la pile est popée.

Puisque toutes les transitions empilées ont un MPR $\geq$ PO $- \Delta$ il s'ensuit que tout élément empilé fait partie d'au moins une solution qui sera éventuellement produite et le nombre d'empilements est linéaire en κn; où κ est le nombre de solutions produites et n est la taille de l'instance. L'analyse grand-O doit tenir compte des calculs additionnels effectués par les règles *itératives*; que nous décrivons maintenant.

On trouve souvent des règles itérative dans les grammaires telles que :

$$\mathrm{P} \rightarrow x\mathrm{P}$$

$$\mathrm{Q}_{i,j} \rightarrow \mathrm{R}_{i,s}\mathrm{S}_{s+1,j}$$

La règle P itère sur un indice quelconque k et la règle Q itère sur s. La transition pour ces états est décorée par la valeur de l'indice de manière à être adéquatement calculée lors du retour en arrière. Ces états doivent être initialisés correctement et leurs définitions doivent comprendre leurs valeurs initiales, leurs incrément et leurs valeurs terminales que nos nommerons collectivement l'itérateur de la règle. Nous annotons donc ces règles ainsi :

$$\mathrm{P}_i \rightarrow x^k\mathrm{P}_{i-k} \mid \forall_{y<k<z}\, k$$

$$\mathrm{Q}_{i,j} \rightarrow \mathrm{R}_{i,s}\mathrm{S}_{s+1,j} \mid \forall_{i<s<j-1}\, s$$

Ceci n'ajoute pas de puissance au retour en arrière mais permet de compacter son usage de la mémoire et d'augmenter son efficacité dans certain cas. Par exemple, des dérivations successives de $\mathrm{P} \rightarrow x\mathrm{P}$ telles que $\mathrm{P} \rightarrow^* xxxxxxxxxx\mathrm{P}$ utilisent 10 éléments sur la pile mais $\mathrm{P}_i \rightarrow x^{10}\,\mathrm{P}_{i+10}$, n'en utilise qu'un.

Le traitement de la règle Q demande plus de logique de la part de l'algorithme parce que lorsque la règle R est poussée sur la pile, nous n'avons nulle part ou disposer de la règle S. Nous parlerons des règles de *type-Q* pour dénoter des productions à deux non-terminaux à droite et des règles *type-P* pour les règles à un seul non-terminal à droite. Notre approche se distingue de celle de Wuchty dans le traitement de ces règles. Nous utilisons une seconde pile destinée à recevoir les règles S lors de la rencontre de règles de type-Q, nous ajoutons la logique nécessaire pour pousser le contenu de la pile secondaire sur la pile principale au besoin et pour dépiler la seconde pile lorsque une règle type-Q est dépilée de la pile principale. Deux piles suffisent pour tout langage exprimable sous la forme normalisée de Chomsky (dans la mesure où la fonction d'évaluation le permet comme nous le verrons plus loin) où les règles compliquées telles que $N_1 \rightarrow N_x N_y N_z$ sont toujours convertibles en une combinaison plus simple (par exemple $N_1 \rightarrow N_x M$ et $M \rightarrow N_y N_x$) par ajout de non-terminaux. Il n'est pas nécessaire de nous limiter à deux symboles non-terminaux à droite, deux piles suffisent au traitement de règles à plusieurs non-terminaux.

La Figure 3 illustre l'approche avec un exemple quelconque ($A \rightarrow xA \mid Ay A \mid z$) dont les solutions sont de la forme $(x^*y)(yx^*z)^*$ de longueurs égales à l'entrée et utilisant une fonction de poids monotone croissante *f(t,i)* donnant la valeur de produire le caractère *t* à la position *i* (la forme de cette équation est discutée dans la section formelle). Cette grammaire simple a trois règles :

règle 1 | $A \rightarrow xA$

règle 2 | $A \rightarrow AyA$

règle 3 | $A \rightarrow z$

qui s'exprime sous la forme annotée :

règle 1 | $A_{i,j|i<j} \rightarrow xA_{i+1,j}$

règle 2 | $A_{i,j|i<j-2} \rightarrow A_{i,s}yA_{s+2,j} \mid \forall_{i\leq s<j-2}s$

règle 3 | $A_{i,j|i=j} \rightarrow z$

laquelle est facilement renversée pour obtenir la récurrence de programmation dynamique :

$$A_{i,j} = MIN \begin{cases} A_{i+1,j} + f(x,i) \mid i < j \\ MIN_{i\leq s<j-1}\left(A_{i,s} + f(y,s+1) + A_{s+2,j}\right) \\ f(z,i) \mid i = j \end{cases}$$

La matrice A est la seule nécessaire pour les calculs en programmation dynamique pour cet exemple. Elle est utilisée pour mémoriser (parfois nommé mémoiser) PO pour tous les sous-problèmes, ainsi la fonction heuristique *h()* se résume à consulter A pour l'intervalle (*i,j*). Mais puisque nous manipulons deux piles, il est nécessaire de modifier l'équation 1 pour évaluer la MPR d'une solution partielle impliquant une règle type-Q sur la pile principale et nous avons :

MPR = *g(*solution partielle) + *h(*sous-solution candidate pile principale*)* + *h(*sous-solution candidate pile secondaire*)*

Où la solution candidate de la seconde pile a le MPR réalisable pour le fragment $A_{s+2,j}$ correspondant à la règle 2. De manière plus générale, on précise l'évaluation de la MPR pour toutes les règles correspondants aux règles de la récurrence de la programmation dynamique :

règle 1 : g(soln partielle) + h(i+1,j) + f(x,i)

règle 2 : g(soln partielle) + h(i,s) + f(y,s+1) + h(s+2,j)

règle 3 : g(soln partielle) + f(z,i)

De cette manière, nous sommes certains que l'étape de retour en arrière correspond exactement avec la fonction heuristique.

Nous avons mentionné plus haut que les règles itératives ont des effets dont il faut tenir compte lors des analyses de performance. L'augmentation de calcul vient des règles type-Q et non des règles type-P dont l'itération n'a pas de conséquence sur l'analyse. Une règle type-Q fait une recherche linéaire sur *s* pour identifier toutes ses valeurs productives. Pour une combinaison de grammaire aux nombreuses règles de type-Q et une instance problématique, ce surcroit de calcul pourrait devenir notable. Considérons une grammaire construite autour d'une règle récursive de type-Q dotée d'un itérateur :

$$Q_{i,i} \rightarrow x$$

$$Q_{i,j} \rightarrow Q_{i,s}Q_{s+1,j} \mid \forall_{i \leq s < j} s$$

L'appel récursif Q→QQ remplira la pile principale jusqu'à ce que l'instance se résolve en branches se terminant en une série de *x*. Le calcul problématique vient de ce que tous les BPS pour toutes les valeurs de *s* à tous les niveaux de la pile et dans chaque branche doivent être vérifiés. L'appel $Q_{1,n} \rightarrow Q_{1,s}Q_{s+1,n}$ entrainera *n* vérifications et pour chaque valeur acceptée de *s* (MPR > PO – Δ; cas de maximisation) un appel récursif causera *n* vérifications pour la somme des deux branches résultantes. Pour une pile profonde de *n*, n^2 vérifications seront nécessaires rien que pour cette seule règle. Remarquez que ceci ne revient pas à dire que l'approche soit exponentielle. Dans notre exemple, si toutes les

valeurs de *s* sont acceptantes, alors le nombre d'états peut devenir exponentiel mais chaque empilement conduit à au moins une solution et le comportement explosif n'est pas la conséquence de l'approche mais de la taille du langage. La complexité de la passe du retour en arrière doit plutôt s'intéresser aux ressources utilisées pour calculer une solution. On s'intéresse alors (a) aux nombres d'états empilés et (b) le nombre de règles type-Q × le nombre d'options dans leurs itérateurs. Dans les cas extrêmes, si la grammaire le permet et si chaque règle consomme un indice ou émet un terminal, alors (avec κ structures sous-optimales) la complexité est $O(\kappa n^2)$ et non $O(\kappa n)$ tel que attendu.

1.2.2 Détails d'implémentation

Les détails d'implantation sont importants lorsqu'on vise les meilleures performances et nous notons ici un élément au sujet de la passe de retour en arrière. Typiquement $\kappa \gg n$ et malgré l'étape polynomiale de programmation dynamique, l'étape de retour en arrière dure beaucoup plus longtemps et il est important de l'optimiser agressivement. Notez que la pile comprend environ autant d'éléments que le nombre de terminaux à émettre. On peut estimer la hauteur maximale de la pile pour une grammaire en termes de la taille de l'instance. Ainsi la mémoire de la pile peut-être réclamée d'un coup, éliminant le besoin de gestion de la mémoire. La hauteur de la pile correspond seulement *à peu près* à la taille de l'instance. Certaines règles, comme les règles type-Q, consomment deux éléments sur la pile et n'émettent aucun terminal. Cependant elles consomment une valeur d'indice et leur nombre sur la pile est donc borné. Le programme MC-Flashfold utilise deux piles qui ensemble consomment *2n* éléments (où *n* est la taille de l'instance) et on s'attend à ce que l'utilisation de structures de données dynamiques non-bornées soit moins efficaces.

Les deux versions de l'algorithme de retour en arrière sont montrées dans la prochaine section. Un premier algorithme est utilisable en absence de règle type-Q. Le second algorithme est légèrement différent et s'utilise en présence de règles type-Q. Avec une règle Q → RS, nous introduisons une seconde pile (φ) qui reçoit le second non-terminal de droite (S). Ceci entraine le besoin de synchroniser le dépilement de R de la pile

principale (τ), avec celui de S sur φ. En effet, lorsque les options de l'itérateur de la règle sont épuisées, c'est R et S qui doivent être dépilés et non seulement R. Il est également nécessaire de vérifier que φ est vide avant de déclarer que la structure décrite par la pile τ est une solution.

RNAsubopt aussi procède au retour en arrière sur une pile principale mais cette pile ne décrit pas la solution courante. Son algorithme est décrit dans la sous-section *Variation sur une pile* plus bas. Brièvement, la pile principale de cet algorithme contient un certain nombre de sous-solutions partielles, toutes faisant l'objet d'un retour en arrière. Dans ce cas, une solution partielle est un 2-tuple $P = < pile, M >$ où M décrit toutes les paires de bases connues à date et *pile* est une pile de règles de production partiellement résolues. RNAsubopt évalue P_i, la structure incomplètement décrite au sommet de la pile, comme suit : (a) Si la règle au sommet de *pile* peut-être résolue immédiatement alors elle l'est, de nouvelles paires de bases viennent possiblement agrandir l'ensemble M puis *pile* est dépilée. (b) sinon, la règle est de type-Q, la règle est dépilée et les non-terminaux à droite de la règle sont empilées; si toutefois plusieurs options doivent être explorées, P*i* est dupliqué autant de fois que nécessaire et les copies sont empilées.

1.3 Description Formelle

Les arbres et les grammaires sont au cœur de l'informatique et les preuves les plus fondamentales de pratiquement tous les champs de l'informatique en dépendent. Voir par exemple (Sikkel 1997) pour une présentation intégrée de différentes approches aux problèmes de parsing, (Aho, Sethi et al. 1985) pour une approche appliquée et (Sipser 1997) pour l'étroite connexion entre, d'une part les grammaires et langages et, d'autre par la complexité de calcul. Nous n'en faisons pas le tour ici mais définissons simplement un formalisme suffisant pour décrire clairement l'approche déployée dans ce chapitre. Notre approche recoupe plusieurs travaux, certains de nature plutôt théorique (Rozenberg and Salomaa 1997; Comon, Dauchet et al. 2007) et d'autres plus appliqués et utilisant la programmation fonctionnelle (Giegerich, Meyer et al. 2004) au sujet des grammaires d'arbres.

Dans un premier temps, nous décrivons le type des problèmes que nous pouvons aborder et nous en décrivons certaines propriétés. Puis nous établissons la correspondance entre ces problèmes, leurs grammaires, leurs solutions en programmation dynamique et l'algorithme d'énumération des solutions sous-optimales. Finalement nous considérons les différences avec RNAsubopt (Wuchty, Fontana et al. 1999) et l'approche *k-best* formalisée sur les hyper-graphes (Huang and Chiang 2005).

1.3.1 Arbres et règles des problèmes d'intervalles

Soit P un problème, $\gamma \in \Gamma^n$ l'encodage d'une instance de P et $S(\gamma)$ les solutions à γ.

Certains problèmes sont solubles par composition de sous-problèmes grâce à des règles que nous diront *règles de composition* (cette propriété est parfois nommée sous-structure optimale) . Avec $\gamma = \gamma_a\gamma_b$, la règle r compose les solutions de γ :

$$S(\gamma) \leftrightarrow r\big(S(\gamma_a), S(\gamma_b)\big)$$

Considérant une décomposition d'un problème $D(P)$ en un ensemble de sous-problèmes $D(P) = \{P_1, P_2, \dots, P_i, \dots, P_m\}$, nous dirons que le problème est *local* si les $S(P_i)$ sont indépendantes des contextes des P_i (les $D(P)\backslash P_i$).

Un problème est *ordonné* lorsque l'ordre des sous-problèmes est prescrit par sa nature et la décomposition prend alors la forme d'un vecteur de sous-problèmes: $D(P) = [P_1, P_2, \dots]$.

Par exemple, un problème d'horaire et le problème des correspondances de Post ne sont pas ordonnés mais l'alignement de séquence ou la détermination d'un parenthèsage optimal pour la multiplication matricielle à la chaine le sont.

Un problème d'*intervalles* est un problème local et ordonné qui accepte des solutions dont les sous-problèmes sont définis sous forme d'intervalles et la composition

recouvre complètement l'intervalle sans laisser de trous. Nous dirons alors que la composition est valide sur l'intervalle. Les compositions

$$[(1, x_1), (x_1 + 1, x_2), (x_2 + 1, x_3), \cdots, (x_k + 1, n)]\ , x_1 \geq 1, x_k < n, \forall 1 < i \leq k, x_{i-1} < x_i$$

sont valides sur l'intervalle $(1, n)$.

Si les sous-problèmes sont eux-mêmes soit décomposables ou solubles en temps constant, alors le problème est *récursif* et ses solutions prennent la forme d'*arbres de composition*.

$$r_1(r_2(r_3(r_4(\cdots), r_x(\cdots), \cdots), r_{x'}\ (\cdots), \cdots), r_{x''}(\cdots), \cdots) \in S(\gamma)$$

Notons $\mathcal{C}$ l'ensemble des règles de composition. Les règles dans l'espace de solution du problème P forment $\mathcal{C}_p$. Une règle de composition est soit de la forme $r: \gamma^+ \to \mathcal{C}$ ou soit de la forme $r: \mathcal{C}^a \to \mathcal{C}, a > 1$ où a est l'*arité* de r, l'*image* de r est dans $\mathcal{C}$ et où son $domaine$ est dans $\mathcal{C}^a$. Ainsi, $\forall \tau \in S(\gamma)$, les feuilles de τ sont produites par des règles de la forme $\gamma^+ \to \mathcal{C}$ et les nœuds internes sont de la forme $\mathcal{C}^a \to \mathcal{C}$.

Les problèmes que nous abordons sont tous des problèmes d'intervalles, récursifs et locaux. La difficulté algorithmique consiste à trouver les sous-problèmes et leurs compositions. Autrement dit, solutionner P, consiste à énumérer les $\tau \in S(\gamma)$.

1.3.2 Énumération des solutions

Nous procédons par construction. À partir de solutions partielles initiales ξ_0, construisons des suites de solutions partielles par applications successives de $r \in \mathcal{C}_P$ jusqu'à ce qu'il ne soit plus possible d'appliquer de règle : $\xi_0, \xi_1, \ldots, \xi_f$. Si ξ_f décrit un arbre τ_f qui est connecté et dont les feuilles forment une $D(\gamma)$, alors $\tau_f \in S(\gamma)$. De tels algorithmes doivent spécifier : la nature des ξ, la construction des ξ_0, les transformations $\xi_i \Rightarrow \xi_{i+1}$ et la détection des ξ_f. De nombreuses variantes sont possibles mais seulement deux approches nous sont nécessaires: *bottom-up* et *top-down.* Pour chacune de ces

méthodes, un grand nombre de variantes algorithmiques sont possibles et nous ne donnons initialement que des approches gloutonnes par énumération.

L'approche *à partir des feuilles* (*bottom-up*) consiste à construire $\tau \in S(\gamma)$ en fusionnant les feuilles d'une solution préalablement identifiée.

Définition des ξ: Ici les états intermédiaires sont des vecteurs de solutions partielles $\xi_i = [\tau_1, \tau_2, \dots, \tau_x, \dots, \tau_m]$ dont chaque τ_x est un sous arbre de $\tau_f \in S(\gamma)$ et décrivant un intervalle valide sur $(1, n)$.

Construction de ξ_0: en utilisant les règles $\gamma_{i,j} \to r \in \mathcal{C}$, trouver les

$$\xi_0 = \left[r_1(\gamma_{1,x_1}), r_2(\gamma_{x_1+1,x_2}), r_3(\gamma_{x_2+1,x_3}), \cdots, r_i(\gamma_{k+1,n})\right],$$
$$1 \le x_1 < x_2 < x_3 < \cdots < x_k < n$$

Transformations $\xi_i \Rightarrow \xi_{i+1}$: En utilisant les règles $\mathcal{C}^a \to \mathcal{C}$, pour chaque ξ_i, trouver les arrangements de *réduction*s :

$$\cdots, r_i(\dots), r_{i+1}(\dots), \cdots, r_{i+a}(\dots), \cdots \Rightarrow \cdots, r\big(r_i(\dots), r_{i+1}(\dots), \cdots, r_{i+a}(\dots)\big), \cdots$$

Détection des τ_f: Chaque réduction, diminue le nombre d'éléments disjoints dans la solution partielle : $|\xi_i| - |\xi_{i+1}| = a - 1$. La solution est complète lorsque $|\xi_i| = 1$. Alors $\xi_i = \xi_f$ et $\xi_f[1] = \tau_f \in S(\gamma)$ est une solution.

L'approche *à partir de la racine* (*top-down*) consiste à spécifier les sous arbres incomplets à partir de toutes les racines possibles.

Définition des ξ: Ici, les ξ sont des arbres *possiblement incomplets* (notés $\hat{\tau}$) dont le seul élément complet est $\hat{\tau}_f = \tau_f$. Un arbre est incomplet si un ou plusieurs de ses sous arbres a (ont) été remplacé(s) par les intervalles valides qu'ils recouvrent. Posons que la relation $\hat{\tau}_a \subset \hat{\tau}_b$ est vraie si tous les nœuds compris dans $\hat{\tau}_a$ sont compris dans $\hat{\tau}_b$ mais que $\hat{\tau}_b$ comprends plus de nœuds que $\hat{\tau}_a$.

Construction de ξ_0: On choisit les $\hat{\tau}_r \leftrightsquigarrow r \in \mathcal{C}_p$ valides pour γ sur l'intervalle $(1, n)$. Avec la relation $\leftrightsquigarrow$ sur l'intervalle (k, l) définie comme suit: Si r est de la forme $\Gamma^{l-k+1} \to \mathcal{C}$, alors $\hat{\tau}_r = r\big(\gamma_{k,l}\big)$. Si $r \in \mathcal{C}^a \to \mathcal{C}$,alors tous les $\hat{\tau}_r = r\Big(r_1\big((k, x_1)\big), r_2\big((x_1 + 1, x_2)\big), , \cdots, r_a\big((x_{a-1} + 1, l)\big)\Big)$ valides. Ici, la notation $r\big((i, j)\big)$ indique qu'une règle r s'applique sur l'intervalle (i, j) mais sans que soit précisé la décomposition de l'intervalle.

Transformations $\xi_i \Rightarrow \xi_{i+1}$: Remplacer un intervalle $r\big((k, l)\big) \in \xi_i$ par les $\hat{\tau}_r \leftrightsquigarrow r$ sur (k, l). (Notons ici que puisque cette opération conserve tous les nœuds de ξ_i mais en rajoute au moins un dans ξ_{i+1}, il s'ensuit que $\forall_i\ 0 \leq i < f, \xi_i \subset \xi_{i+1}$ et $i < n$.)

Détection des τ_f: Si ξ_i n'est pas incomplet, alors $\xi_i = \xi_f = \tau_f \in S(\gamma)$ est une solution.

1.3.3 Poids des règles

Les poids permettent de transformer un problème d'énumération des solutions acceptables en problème d'optimisation.

Avec α, β des arbres quelconques, notons $\alpha \lhd \beta$ si α est un sous arbre de β et $\alpha \bowtie \beta$ si ni $\alpha \lhd \beta$, ni $\beta \lhd \alpha$.

Avec la fonction $\omega: \alpha \to \mathbb{R}$ donnant le poids d'un arbre. Soit $\tau, \mu \in S(\gamma), \tau \neq \mu$, $\tau_a, \tau_b \lhd \tau, \mu_1 \lhd \mu$ et $\tau_a \bowtie \tau_b$, alors $\exists 1 \leq i \leq j < k \leq l \leq n \mid \tau_a \in S(\gamma[i, j]), \tau_b \in S(\gamma[k, l])$. La localité de P nous dit que $\nexists f: \alpha \to \mathbb{R} | f(\tau_1) = \omega(\tau_2)$. Mais nous voulons que si, de plus $\mu_1 \in S(\gamma[i, j])$ alors $\exists \nu \in S(\gamma)$, avec $\tau_1 \lhd \nu$ et ν autrement identique à μ, alors si $\omega(\tau_1) < \omega(\mu_1) \Rightarrow \omega(\nu) < \omega(\mu)$, $\omega(\tau_1) > \omega(\mu_1) \Rightarrow \omega(\nu) > \omega(\mu)$ et $\omega(\tau_1) = \omega(\mu_1) \Rightarrow \omega(\nu) = \omega(\mu)$. Alors, pour être générale, ω doit être définie de manière récursive en correspondance avec les $r \in \mathcal{C}_p$ comme une combinaison indépendante de ses paramètres.

L'ensemble des poids est la bijection $\mathcal{W}_{\mathcal{C}_p} = \left\{\omega \mid \forall r \in \mathcal{C}_p \Leftrightarrow \mathcal{W}_{\mathcal{C}_p}(r) = \omega\right\}$. ω est de la forme $\mathcal{C}_p^a \to \mathbb{R}$ si r est de la forme $\mathcal{C}_p^a \to \mathcal{C}_p$ et de la forme $\gamma^l \to \mathbb{R}$ si r est de la forme $\gamma^l \to \mathcal{C}_p$. Et nous utilisons la combinaison linéaire :

$$\begin{aligned}\omega_i(\tau_{i.1}, \tau_{i.2}, \dots, \tau_{i.a}) \\ &= c_i + \omega_{i.1}(\tau_{i.1.1}, \tau_{i.1.2}, \dots) + \omega_{i.2}(\tau_{i.2.1}, \tau_{i.2.2}, \dots) + \cdots \\ &+ \omega_{i.a}(\tau_{i.a.1}, \tau_{i.a.2}, \dots)\end{aligned}$$

En particulier, cette forme est toujours monotone (avec $\omega_i \leq \omega_i'$ lorsque $\omega_{i.j} \leq \omega'_{i.j}$ $1 \leq j \leq a$) au sens du principe d'optimalité de la programmation dynamique (Cormen, Leiserson et al. 1989).

L'expansion d'un terme (ici $i.1$) révèle en particulier sa constante :

$$\begin{aligned}\omega_i(\tau_{i.1}, \tau_{i.2}, \dots, \tau_{i.a}) \\ &= c_i + \left(c_{i.1} + (\omega_{i.1.1}\tau_{i.1.1}) + (\omega_{i.1.2}\tau_{i.1.2}) + \cdots + (\omega_{i.1.a}\tau_{i.1.a})\right) + \omega_{i.2}(\tau_{i.2}) \\ &+ \cdots + \omega_{i.a}(\tau_{i.a})\end{aligned}$$

De même, l'expansion d'un ensemble $\mathcal{E}$ de termes quelconques laisse possiblement des termes non-explorés X, correspond à l'évaluation de l'arbre incomplet $\hat{\tau}$ et prends la forme :

$$\omega(\hat{\tau}) = \sum_{e \in \mathcal{E}} c_e + \sum_{\nabla \in X} \omega(\tau_\nabla)$$

L'expansion d'un terme $\hat{\nabla}$, correspond à l'évaluation d'un nouvel arbre incomplet $\hat{\tau}'$. Avec $g = \sum c_e$ et $\chi = \sum \omega(\tau_\nabla)$:

$$\omega(\hat{\tau}) = g(\hat{\tau}) + \chi(\hat{\tau})$$

$$\omega(\hat{\tau}') - \omega(\hat{\tau}) = c_{\hat{\nabla}} + \left(\sum_{1 \le i \le arité(\hat{\nabla})} \omega_i(\tau_i) \right) - \omega_{\hat{\nabla}}(\tau_{\hat{\nabla}})$$

notons :

$$\psi_{\hat{\nabla}} = \sum_{1 \le i \le arité(\hat{\nabla})} \omega_i(\tau_i)$$

ainsi :

$$g(\hat{\tau}') = g(\hat{\tau}) + c_{\hat{\nabla}}$$

$$\chi(\hat{\tau}') = \chi(\hat{\tau}) + \psi_{\hat{\nabla}} - \omega(\tau_{\hat{\nabla}})$$

chaque expansion de terme ajoute un terme constant $c_{\hat{\nabla}}$ à g, retire $\omega(\tau_{\hat{\nabla}})$ et ajoute des termes $\psi_{\hat{\nabla}}$ à χ.

Pour un $\tau \in S(\gamma)$, $\chi = 0$ et $\omega(\tau) = g(\tau)$ que nous notons $W(\tau)$ l'évaluation récursive du poids de τ par les $\omega \in \mathcal{W}_{\mathcal{C}_p}$.

1.3.4 Catégories de problèmes

Un problème de *décision* s'intéresse à l'existence même d'un arbre. Par exemple on peut chercher à savoir si un ensemble donné de règles peut être construit sur $\gamma \in P$: Une instance a-t-elle une solution? Existe-t-il un chemin entre deux nœuds d'un graphe? L'algorithme CYK, voir par exemple (Sikkel 1997), solutionne des problèmes de ce type.

Un problème d'*évaluation* applique des transformations données pour chaque règle et produit un résultat pour chaque combinaison de sous-problèmes $W(\tau), \forall \tau \in S(\gamma)$. Cela peut prendre la forme d'un résultat numérique unique comme dans l'évaluation d'une fonction numérique récursive comme dans le calcul du n^{ième} nombre des suites de Fibonacci ou de Catalan.

Un problème d'*optimisation* $\mathcal{O}(\gamma)$ demande les solutions dont l'évaluation est optimale par rapport à une politique d'optimisation donnée $OPT \in \{MIN, MAX\}$. Avec $OPT: \{\tau\} \to \{\tau\}$ on a $\mathcal{O}^{OPT}(\gamma) = \{\tau | \tau \in OPT\{S(\gamma)\}\}$, la valeur optimale est unique et est obtenue par $W(\tau), \exists \tau \in \mathcal{O}^{OPT}(\gamma)$. On peut obtenir $W(\tau)$, par $\mathcal{A}_p$, l'algorithme en programmation dynamique dont la fonction objectif réalise les règles $r \in \mathcal{C}_p$ et utilise les poids de $\mathcal{W}_{\mathcal{C}_p}$. Nous verrons plus loin comment obtenir $\mathcal{A}_p$.

Un problème de *décoration* ($\mathcal{D}$) demande un résultat sous la forme d'un mot $\sigma \in \Sigma^*$ où Σ est un *alphabet de sortie*. $\mathcal{D}: \tau \to \sigma$. Par exemple, pour un graphe : donner un chemin décrivant la solution. Pour un ARN : énumérer les nucléotides formant des bases dans une structure secondaire $\tau \in S(\gamma)$. Pour un alignement de mots : montrer les caractères en correspondance et positionner les trous.

Nous faisons ici la distinction entre un problème de décoration et un problème de traduction. La traduction implique la conservation sémantique mais ce n'est pas le cas d'un problème de décoration. Dans un problème de décoration, la syntaxe associée aux $\sigma \in \Sigma^*$ est *plaquée sur* les nœuds des τ alors qu'un problème de traduction consiste à chercher un σ pour lequel il existe un arbre de composition qui soit le plus similaire possible a τ et cela ouvre tout un autre champs de travail. Nos problèmes sont des problèmes de décoration.

Si on s'intéresse à la décoration d'une solution optimale $\mathcal{D}^{OPT}(\tau), \exists \tau \in \mathcal{O}^{OPT}(\gamma)$ on peut l'obtenir en appliquant la méthode des pointeurs à rebours dans $\mathcal{A}_p$ (Cormen, Leiserson et al. 1989). Cependant on s'intéresse aux décorations de plus d'une solution mais on ne dispose pas de méthode générale pour énumérer les sous-ensembles quelconques $\mathcal{S}^{\phi} = \{\tau | \tau \in \phi(S(\gamma))\}$ puisque d'une part $|S(\gamma)| \in O(x^{|\gamma|})$ rendant inefficace l'énumération et d'autre part les constructions par exploration des règles ne sont pas *a priori* garanties de terminer.

Nous sommes intéressés principalement par l'ensemble des Δ-sous-optimaux, une forme particulière de ϕ dont il est possible d'énumérer les membres par construction sans énumérer les $S(\gamma)$:

$$\mathcal{S}^{\Delta,MIN}(\gamma) = \{\langle \mathcal{D}(\tau), W(\tau)\rangle | \tau \in S(\gamma), W(\tau) \leq W(\tau^{\circ}) + \Delta, \tau^{\circ} \in \mathcal{O}^{OPT}(\gamma), \Delta \geq 0\}$$

$$\mathcal{S}^{\Delta,MAX}(\gamma) = \{\langle \mathcal{D}(\tau), W(\tau)\rangle | \tau \in S(\gamma), W(\tau) \geq W(\tau^{\circ}) - \Delta, \tau^{\circ} \in \mathcal{O}^{OPT}(\gamma), \Delta \geq 0\}$$

Notre intérêt portera également sur le problème des κ meilleures solutions. Soit $W(\tau^{\kappa})$ la valeur du $\kappa^{ième}$ meilleur τ selon l'ordonnancement sur les $W(\tau)$,

$$\mathcal{S}^{\kappa,OPT}(\gamma) = \left\{\langle \mathcal{D}(\tau), W(\tau)\rangle \middle| \tau \in S(\gamma), W(\tau) \begin{matrix} \leq_{OPT=MIN} \\ \geq_{OPT=MAX} \end{matrix} W(\tau^{\kappa})\right\}$$

Et une version moins précise $\mathcal{S}^{\kappa^*,OPT}(\gamma)$ donnant un des sous-ensembles $\in \mathcal{S}^{\kappa,OPT}(\gamma)$ de taille exactement κ.

1.3.5 Une grammaire pour $\mathcal{C}_p$

À $\mathcal{C}_p$ correspond une grammaire $\mathcal{G}_p$. Une grammaire hors-contexte est un tuple $\mathcal{G} = \langle N, T, S, R\rangle$ où N est l'alphabet des symboles *non-terminaux*, T est l'alphabet des symboles *terminaux*, R est l'ensemble des *règles de production* données sous la forme $R: N \rightarrow (N \cup T)^*$ et $S \subseteq R$ est l'ensemble des règles *de départ*.

Pour établir la correspondance entre $\mathcal{C}_p$ et $\mathcal{G}_p$, il suffit de construire le tuple de $\mathcal{G}_p$:

$$N = \{symbole(\alpha) | r \in \mathcal{C}_p, \alpha = image(r)\}$$

où $symbole(\alpha)$ donne le symbole unique représentant α et $image(f)$ renvoie l'image de la fonction f,

$$T = \Gamma$$

$$R = \{g \rightarrow d | r \in \mathcal{C}_p, g = symbole(image(r)), d = symboles(domaine(r))\}$$

où $symboles(\mathcal{C}^a)$ donne le mot $w \in N$, correspondant aux règles,

$$S = \{\rho | \tau \in S(P), \rho \leftrightarrow racine(\tau)\}$$

Une telle grammaire peut souvent (sous réserve de l'équivalence des fonctions de poids) être simplifiée par des méthodes classiques telles que la normalisation de Chomsky (Harrison 1978). Par exemple, il est possible d'éliminer les règles de la forme $r: n \to t$ où $n \in N$ et $t \in T^*$ en remplaçant les occurrences de n dans le côté droit des $r \in R$ et de remplacer les règles $r: n \to mw$ (ou encore $r: n \to wm$) où $n \in N, m \in N, w \in (N \cup T)^*$ par des paires équivalentes de règles en introduisant de nouveaux symboles $N' = \eta \cup N$, $R' = \{\eta \to w, r' \to \eta m\} \cup R \backslash r$ (ou encore $r' \to m\eta$). À partir de maintenant, nous considérons que toutes les grammaires $\mathcal{G}_p$ sont dans la forme simplifiée.

L'application d'une règle $r \in R$ sur un mot $w \in (N \cup T)^*$ se nomme dérivation et consiste à remplacer une occurrence de $n \in N$ dans w par la partie à droite de la flèche dans r par sa partie de gauche et se note $w \to_r w'$ et la dérivation grâce à une règle quelconque tirée de R est notée $w \to w'$ alors que la dérivation successive $w \to w_{r \in R} \to \cdots \to w_i \cdots \to w'$ et notée $w \to^* w'$.

Une dérivation décrit un *arbre de dérivation*, noté A, où chaque $n \in N$ correspond à un nœud interne.

Si un mot est dépourvu de symboles tirés de N, alors il ne peut être dérivé et on le note w^{final}.

La visite *en ordre* des feuilles de l'arbre A obtenu par la suite de dérivations $\to_{s \in S} \cdots \to w^{final}$ donne le mot $w^{final} \in T^*$ alors, considérant $\gamma \in P$, si $\gamma = w^{final}$, alors l'arbre de dérivation de w^{final} est une visite d'un arbre $\tau \in S(\gamma)$ et cet arbre est une solution du problème. Nous pourrions donc énumérer les $\tau \in S(\gamma)$ grâce à $\mathcal{G}_p$ mais $|S(\gamma)| \in \mathcal{O}(x^n)$ (et même généralement $\in \Theta(x^n)$). Nous allons plutôt énumérer $\tau \in \mathcal{S}^{\Delta}(\gamma)$ (que nous noterons souvent simplement $\mathcal{S}^{\Delta}(\gamma)$) en appliquant les $r \in \mathcal{G}_p$ aux solutions partielles (les w_i) seulement si w_{i+1} est sur une dérivation dont $w^{final} \in \mathcal{S}^{\Delta}(\gamma)$. Mais comment savoir si w_{i+1} est dans une telle dérivation?

1.3.6 Trouver une solutions dans $S^{\Delta,OPT}(\gamma)$

Pour un problème de minimisation. Considérant la dérivation $\rightarrow_{s\in S} w_1 \cdots \rightarrow w_i \rightarrow_{r\in R} w_{i+1} \rightarrow \cdots \rightarrow w^{final}$. Avec w^{final} est une visite de $\tau \in S^{\Delta}(\gamma)$, où l'intervalle $\hat{\hat{V}}$ est remplacé par une décomposition valide $\left[\hat{\hat{V}}_1, \hat{\hat{V}}_2, \ldots, \hat{\hat{V}}_a\right]$ par l'application de $r \in R$, alors avec $\mathcal{W}_{\mathcal{C}_p}(r) = \omega_r$

$$\mathcal{O}(\gamma) + \Delta \geq W(\tau) = g(w_i) + \chi(w_i) = g(w_{i+1}) + \chi(w_{i+1})$$

et, puisque

$$g(w_{i+1}) = g(w_i) + c_r$$

alors, $w_i \rightarrow_r w_{i+1}$ est explorée si

$$\chi(w_{i+1}) \leq \mathcal{O}(\gamma) + \Delta - g(w_i) - c_r$$

Nous ne pouvons pas connaitre $\chi(w_{i+1})$ à moins de connaitre τ (ce que nous cherchons) mais nous pouvons obtenir la valeur de la meilleure dérivation possible à partir de w_{i+1} grâce à $\mathcal{A}_p$ (un algorithme en programmation dynamique que nous décrirons bientôt). Avec, $\chi^{OPT} w_{i+1} = OPT\big(\chi(w_{i+1})\big)$ la valeur de la meilleure suite possible de dérivations donnant un w^{final}, choisissons d'explorer $w_i \rightarrow_r w_{i+1}$ si

$$\chi^{MIN}(w_{i+1}) \leq \mathcal{O}(\gamma) + \Delta - g(w_i) - c_r$$

On évalue $\chi^{MIN}(w_{i+1})$ à partir de $\chi^{MIN}(w_i)$ grâce à la relation

$$\chi^{MIN}(w_{i+1}) = \chi^{MIN}(w_i) + \psi_r^{MIN} - \omega_r\big(\tau_{\hat{\hat{V}}}\big)$$

laquelle est la somme d'un ensemble de termes où à chaque dérivation, le terme $\omega_r\left(\tau_{\hat{\gamma}}\right)$ est retiré et l'ensemble (possiblement vide) ψ_r^{MIN} est ajouté, le nombre d'opération à chaque étape est bornée par a (l'arité de r).

Remarquons qu'il existe possiblement plus qu'une suite de dérivation pour chaque $\tau \in S(\gamma)$. En fait il existe autant de suites de dérivations pour τ qu'il y a de visites de ses nœuds. Nous choisissons la visite qui applique toujours la première dérivation à gauche et elle est nécessairement unique pour un τ donné. Notons que ceci est différent que d'affirmer que la suite de dérivations soit unique à τ.

Ces observations suggèrent immédiatement l'algorithme suivant pour obtenir un $\tau \in S^A(\gamma)$. Plus tard, nous verrons comment généraliser pour obtenir $\{\tau |\, \tau \in S^A(\gamma)\}$. Construisons successivement les $\hat{\tau}_1, \hat{\tau}_2, \ldots, \hat{\tau}_i, \ldots, \tau$ en utilisant l'approche *à partir de la racine*. Remarquons d'abord qu'il n'est pas nécessaire d'encoder explicitement les ξ_i (lesquels correspondent aux w_i de la suite de dérivation) ni de conserver la suite des τ_i. Notons que τ est complètement décrit par un ensemble de règles instanciées définies comme des triplets $< r, i, j >$ où $r \in R$ et (i, j) est l'intervalle sur lequel la règle s'applique. Nous utilisons un ensemble X de triplets décrivant les termes de χ_i.

Notons $D_r^*(i, j)$ une décomposition valide sur l'intervalle (i, j) pour la règle $r \in R$ qui soit telle que $\omega_r(i, j) = c_r + \omega_{r.1}(i, x_1) + \cdots + \omega_{r.a}(x_{a-1} + 1, j)$ et ψ_r la suite des termes en $\omega_{r.k}$ de cette somme. Puisque $\omega_r(i, j)$ est connu, cette somme existe aussi et elle est découvrable par énumération des intervalles valides.

Avec $a(x)$ l'arité de la règle $x \in R$, l'algorithme de base (à partir de la racine) prend la forme :

$\tau \leftarrow \emptyset, X \leftarrow \emptyset, n \leftarrow |\gamma|$
Prendre $r \in S\ |\omega_r(1,n) \leq \mathcal{O}^{MIN}(\gamma) + \Delta$
SI r n'existe pas retourner $\emptyset$
$X \leftarrow [X, \langle r, 1, n\rangle]$
TANT QUE $X \neq \emptyset$
    $\langle t, i, j\rangle \leftarrow prendre\ un\ élément\ de\ X$
    $\psi_t \leftarrow D_t^*(i,j)$
    $\forall k \leftarrow a(t)$ downto 1 $(step\ 1)$
        $X \leftarrow X \cup \psi_t[k]$
    $\tau \leftarrow \tau \cup \langle t, i, j\rangle$
FIN TANT QUE
retourner τ

1.3.6.1 L'algorithme de base produit une dérivation légale

L'ensemble X contient une entrée pour chaque règle dont l'intervalle est connu mais dont la décomposition est indéterminée (forme $r\big((i,j)\big)$. Chaque tour de boucle remplace une de ces règles par une décomposition valide ($D_r^*(i,j)$). Lorsque $X = \emptyset$, τ est la dérivation d'un arbre complet.

La racine $r\big((1,n)\big)$ d'un arbre τ dans $\mathcal{S}^{\Delta,MIN}(\gamma)$ est sélectionnée si $\mathcal{S}^{\Delta,MIN}(\gamma) \neq \emptyset$ puisque $\omega_r(1,n) \leq \mathcal{O}^{MIN}(\gamma) + \Delta$. Puisque toutes dérivations sont obtenues en respectant la contrainte $\omega_r(i,j) = c_r + \omega_{r.1}(i,x_1) + \cdots + \omega_{r.a}(x_{a-1}+1,j)$, $W(\tau) \leq \mathcal{O}^{MIN}(\gamma) + \Delta$.

1.3.6.2 La complexité de la découverte des $D_r^*(i,j)$ dépend de l'arité maximale des règles

Nous distinguons trois cas de figure sur l'arité maximale des règles de la grammaire $a^{MAX} = \text{MAX}(a(r), r \in R)$:

Cas 1. $a^{MAX} = 1$: Dans ce cas, d'une part $|X| \leq 1$ éliminant le besoin d'entretenir un ensemble et, d'autre part, les D_r^* sont toujours complètement spécifiés par le côté droit des règles. L'arbre est alors obtenu en $O\big(noeuds(\tau)\big) \in O(|\gamma|)$.

Cas 2. $a^{MAX} = 2$: Chaque $D_r^*(i,j)$ est trouvée en $O(|\gamma|)$. Avec une décomposition par nœud, τ est obtenu en $O(noeuds(\tau)|\gamma|)$. Avec $O\big(noeuds(\tau)\big) \in O(|\gamma|)$, l'arbre est obtenu en $O(|\gamma|^2)$. Ce cas est le plus commun car si le language est régulier, alors il existe une grammaire telle que $a^{MAX} = 2$.

Cas 3. $a^{MAX} > 2$: Les décompositions valides d'un intervalle i,j sont en $O\big(a^{|\gamma|}\big)$.

1.3.6.3 Une grammaire avec itérateurs

L'algorithme de base ne produit qu'une solution $\exists \tau \in S^A(\gamma)$ mais nous voulons lister toutes les solutions (ie : $\forall \tau \in S^A(\gamma)$). Nous fournissons maintenant deux algorithmes faisant cela. Le premier pour le cas où $a^{MAX} = 1$ et l'autre pour les cas où $a^{MAX} = 2$. Cette énumération se fait par retour en arrière (*backtracking*) sur les sous arbres possibles de τ. Pour cela, d'une part nous représentons τ sous la forme d'une pile mais afin que cette pile supporte les retours en arrière, il nous faut ajouter la notion d'*itérateurs* à la grammaire. D'autre part, nous ordonnons les règles de manière à ce que toutes les règles compatibles (ayant le même symbole à gauche) soient visitées en ordre.

La pile que nous utilisons est un arrangement pré-ordre des nœuds de l'arbre avec la racine au bas de la pile. Cette pile décrit un $\hat{\tau}$ le long d'une suite de dérivations culminant en $\tau \in S^{\Delta,OPT}(\gamma)$, (ici avec OPT=MIN), donc $\mathcal{O}(\gamma) + \Delta \geq g(\hat{\tau}) + \chi(\hat{\tau})$ et nous acceptons une transition $\hat{\tau} \rightarrow_r \hat{\tau}'$ si $\chi(\hat{\tau}') \leq \mathcal{O}(\gamma) + \Delta - (g(\hat{\tau}) + c_r)$. La dérivation $\dots A \dots \rightarrow_r \dots BC \dots$, correspond à l'application d'une règle $r(i,j) \rightarrow (r_1(i,x_1), r_2(x_1+1,j))$ et à $\mathcal{W}_{C_p}(r) = \omega_r$ avec $\omega_r(i,j) = c_r + \omega_{r_1}(i,x_1) + \omega_{r_2}(x_i+1,j)$. Cette dérivation a lieu avec $\langle r,i,j \rangle$ au sommet de la pile pour toute décomposition valide de i,j, les $x_1 \in D_r^*(i,j) = \{x_1 | c_r + \omega_{r_1}(i,x_1) + \omega_{r_2}(x_1+1,j) \leq \mathcal{O}(\gamma) + \Delta - (g(\hat{\tau}) + c_r)\}$. Nous savons que $D_r^*(i,j) \neq \emptyset$ puisque $\langle r,i,j \rangle$ est sur la pile mais nous ne savons pas quelles valeurs de x_1 sont dans $D_r^*(i,j)$ il nous faut donc, d'une part annoter les $r \in R$ avec $\mathcal{I}_r$ (les *itérateurs* de r) les étendues permises pour les x_i et d'autre part associer les valeurs des $\overrightarrow{x_i}$ aux éléments de la pile afin de permettre les retours en arrière. Notre grammaire augmentée prend la forme $\langle N,T,S,R,\mathcal{I} \rangle$ et les éléments de la pile sont des $\langle r,i,j,\vec{x} \rangle$.

Lorsque les D_r^* du nœud au sommet de la pile ont été énumérés par le retour en arrière, la pile est popée. Mais cela n'est pas la fin de l'histoire car une autre règle avec le même non-terminal à gauche de sa flèche pourrait être disponible dans R. Cette énumération est possible si les règles sont toujours visitées dans le même ordre et à partir du début. Pour cela, un ordre quelconque est imposé à R, lequel devient techniquement un vecteur de règles et non pas un ensemble. Cependant, comme la plupart des problèmes ont relativement peu de règles avec le même côté gauche, il est avantageux de pré-spécifier une règle unique à explorer lors du retour en arrière pour chaque règle et c'est ce qu'incarne la fonction $suivant: R \rightarrow R$.

Rappelons que la grammaire $\mathcal{G}_p = \langle N, T, R, S, \mathcal{I} \rangle$ solutionne le problème P dont γ est une instance de longueur $|\gamma| = n$ et qu'une règle $r \in R$ est de la forme $r: g \rightarrow d$ avec $g \in N$ et $d \in (N \cup T)^*$. Nous accédons au caractère à gauche avec $gauche(r)$ et au mot à droite avec $droite(r)$. Une solution est $\tau \in S^{\Delta}(\gamma)$. Nous construisons les τ avec la méthode *à partir de la racine* dont les états intermédiaires ξ_i sont des arbres possiblement incomplets notés ici $\hat{\tau}$. La structure de donnée décrivant $\hat{\tau}$ est une pile dont chaque élément est une application d'une règle sur un intervalle donné pris sur $1 \ldots n$ et encodé sous la forme du tuple $\langle r, i, j, \vec{x} \rangle$ où r est la règle, $i \ldots j$ est l'intervalle et où $\vec{x}$ est le vecteur donnant la position des itérateurs courants appliqués aux sous arbres de r et dont les intervalles permis sont définis dans $\mathcal{I}$. La méthode $suivant(r)$ retourne la règle à visiter après la visite de r lors des pas de retour en arrière.

Notons que l'$\mathcal{I}$ d'une $\mathcal{G}_p$ définie pour un langage dont $a^{MAX} = 1$ est généralement 0 et c'est cette forme que nous montrons dans l'algorithme 1.

1.3.7 Algorithme pour les grammaires linéaires

Un grammaire linéaire est une grammaire n'ayant j'amais une règle comprenant plus d'un symbole non-terminal à droite. L'allignement de séquences est un exemple de problème adapté à de telles grammaires (voir appendices). Les solutions prennent la forment d'une liste (ou d'un arbre dont aucun nœud n'a plus d'un enfant). L'énumération des solutions sous-optimales pour des problèmes accèptant des grammaires linéaires est un cas particulier des grammaires hors-contextes. Ce cas accèpte un algorithme plus efficace que l'algorithme pour des langages hors-contexte que nous décrivons ici.

1.3.7.1 Retour en arrière pour les grammaires linéaires

Nous avons vu comment obtenir une solution quelconque dans $\mathcal{S}^{\Delta,MIN}(\gamma)$ grâce à l'*algorithme de base* par l'approche *à partir de la racine*. Puis nous avons vu comment augmenter la grammaire avec les itérateurs. Modifions maintenant l'algorithme comme suit :

1. Les solutions $\hat{\tau}$ sont conservées dans une pile dont seule la règle au sommet est possiblement de la forme $r\big((i,j)\big)$.
2. Lorsque la pile ne comprend que des règles complètement spécifiées, elle décrit une suite de dérivations acceptable. ie : $\hat{\tau} = \tau^{final} \in \mathcal{S}^{\Delta,MIN}(\gamma)$.
3. Les solutions alternatives sont obtenues par retour arrière sur les décompositions possibles à chaque nœud. Ces décompositions sont obtenues grâce aux itérateurs associés.

1.3.7.2 Pseudo-code de l'algorithme 1

Note : dans les algorithmes qui suivent, les piles sont encodées sur des vecteurs et pour la pile p, le dessus de la pile est $p[|p|]$, on en dépile un élément avec $p \leftarrow p[1 \dots |p| - 1]$ et on y empile q avec $p \leftarrow [p, q]$ et on remplace le dessus de la pile par l'élément r avec $p[|p|] \leftarrow r$, la pile est vide si $p = \emptyset$ que l'on prend pour synonyme de $|p| = 0$.

<u>algorithme 1 ($a^{MAX} = 1$)</u>

$\langle r, 1, n, 0\rangle \leftarrow$ la première règle $\in S | \omega_r(1,n) \leq \mathcal{O}(\gamma) + \Delta$
SI $\nexists r$ ALORS retourner $\emptyset$
$\hat{\tau} \leftarrow [\langle r, 1, n, 0\rangle]$

TANT QUE $\hat{\tau} \neq \emptyset$
    DÉPILER $\leftarrow vrai$
    SI $droite(\hat{\tau}[|\hat{\tau}|]) \in T^*$
        $imprimer(\hat{\tau})$
    SINON
        SI $g(\hat{\tau}) + \chi\left(droite(\hat{\tau}[|\hat{\tau}|])\right) \leq \mathcal{O}(\gamma) + \Delta$
            $\hat{\tau} \leftarrow [\hat{\tau}, droite(\hat{\tau}[|\hat{\tau}|])]$
            DÉPILER $\leftarrow faux$
        FIN SI
    FIN SI
    SI DÉPILER $= vrai$
        TANT QUE $\hat{\tau} \neq \emptyset$ ET $suivant(\hat{\tau}[|\hat{\tau}|]) = \emptyset$
            $\hat{\tau} \leftarrow \hat{\tau}[|\hat{\tau}| - 1]$
        FIN TANT QUE
        SI $\hat{\tau} \neq \emptyset$
            $\hat{\tau}[|\hat{\tau}|] \leftarrow suivant(\hat{\tau}[|\hat{\tau}|])$
        FIN SI
    FIN SI
FIN TANT QUE

1.3.7.3 L'algorithme 1 énumère $\{\boldsymbol{\tau}|\boldsymbol{\tau} \in \boldsymbol{\mathcal{S}^{\Delta}(\gamma)}\}$ pour les grammaires linéaires en $\boldsymbol{O(|\mathcal{S}^{\Delta,OPT}(\gamma)|\times|\gamma|^2)}$

Lorsqu'une solution partielle est une solution complète, celle-ci est produite en sortie en énumérant ses nœuds en $O\big(noeuds(\tau)\big) \in O(|\gamma|)$ (ou, alternativement, en produisant sa décoration en $O(|\gamma|)$).

L'utilisation de l'oracle garanti que chaque nœud poussé sur $\hat{\tau}$ conduit nécessairement à au moins une solution et à chaque nœud correspond l'exploration des

règles alternatives données dans la grammaire pour la règle au sommet de $\hat{\tau}$. En suposant qu'aucune solution ne partage de sous-solution (ce qui est invressemblable) le nombre de nœuds explorés est $O\left(|\mathcal{S}^{\Delta,OPT}(\gamma)|\times noeuds\left(\hat{\tau}\right)\times\left|\mathcal{G}_p\right|\times|\gamma|\right)$ et, pour une grammaire donnée : $O(|\mathcal{S}^{\Delta,OPT}(\gamma)|\times|\gamma|^2)$.

1.3.7.4 Retour en arrière pour les grammaires hors-contexte

Lorsque $a^{MAX}=2$, il y a trois types de règles qui peuvent s'appliquer. Voyons d'abord le cas le plus général. Une règle r dont $a(r)=2$ (de la forme $r\in N\rightarrow N$) est $r(i,j)\leftarrow r_1(i,x_1)+r_2(x_1+1,j)$, correspondant à un nœud (r) avec deux enfants $(r_1 \text{ et } r_2)$. Mais $\hat{\tau}$ est une visite pré-ordre de $\tau\in S(\gamma)$. Lorsque $\hat{\tau}_i\rightarrow_r\hat{\tau}_{i+1}$ la pile $\hat{\tau}_i=[\dots,r]$ donne la pile $\hat{\tau}_{i+1}=[\dots,r,r_1]$. La visite du sous arbre dont r_1 est la racine sera suivie de la visite dont r_2 est la racine produisant un $\hat{\tau}_j=[\dots,r,\dots,r_1,\dots,r_2]$ avec $1<i<j$. Mais il n'y a pas d'information donnant r_2 au dessus de la pile $\hat{\tau}_{j-1}=[\dots,r,\dots,r_1,\dots]$. Nous pourrions fouiller $\hat{\tau}_{j-1}$ pour localiser r et en déduire que r_2 n'a pas été exploré mais cela serait inefficace. Nous introduisons plutôt une seconde pile φ pour recevoir r_2 pendant que les sous arbres de r_1 sont explorés.

Avec $\omega\left(\hat{\tau}\right)=g\left(\hat{\tau}\right)+\chi\left(\hat{\tau}\right)$, isolons un terme $\hat{\tau}_1$ dans $\chi\left(\hat{\tau}\right)$: $\omega\left(\hat{\tau}\right)=g\left(\hat{\tau}\right)+\omega\left(\hat{\tau}_1\right)+\overline{\chi\left(\hat{\tau}\right)}$. On peut donc écrire $\omega\left(\hat{\tau}\right)=g\left(\hat{\tau}\right)+c_r+\chi\left(\hat{\tau}_1\right)+\overline{\chi\left(\hat{\tau}\right)}$. Les meilleurs poids des termes non explorés sont connaissables grâce aux tables de $\mathcal{A}_p$ et le critère pour choisir l'exploration d'une règle $\hat{\tau}\rightarrow_r\hat{\tau}'$ avec $r\in N\rightarrow N^+$ prends, pour $OPT=MIN$, la forme :

$$g\left(\hat{\tau}\right)+c_r+\chi^{MIN}\left(\hat{\tau}_1\right)+\overline{\chi^{MIN}\left(\hat{\tau}\right)}\leq\mathcal{O}^{MIN}(\gamma)+\Delta$$

et, pour $OPT = MAX$:

$$g(\hat{\tau}) + c_r + \chi^{MAX}(\hat{\tau}_1) + \overline{\chi^{MAX}(\hat{\tau})} \geq \mathcal{O}^{MAX}(\gamma) - \Delta$$

Nous gardons les termes non-explorés dans la pile φ et, comme préalablement, les termes explorés dans la pile $\hat{\tau}$. La valeur de $g(\hat{\tau})$ est une somme mise à jour à chaque empilement/dépilement de $\hat{\tau}$ et celle de $\chi(\hat{\tau})$ est une somme mise à jour à chaque empilement/dépilement de φ. L'évaluation du critère se fait donc en temps constant.

Selon la normalisation appliquée sur l'ensemble des règles, on pourrait avoir choisi de conserver certaines règles de la forme $r \in N \rightarrow N$. Cela ne pose pas de problème car le critère défini s'applique tel quel. Les règles de la forme $r \in N \rightarrow T^+$ sont acceptées si les terminaux coïncident avec le sous-problème ($droite(r) = \gamma[i,j]$) et si

$$g(\hat{\tau}) + c_r + \overline{\chi^{MIN}(\hat{\tau})} \leq \mathcal{O}^{MIN}(\gamma) + \Delta$$

ou, pour les cas où $OPT = MAX$:

$$g(\hat{\tau}) + c_r + \overline{\chi^{MAX}(\hat{\tau})} \geq \mathcal{O}^{MAX}(\gamma) - \Delta$$

Lorsqu'une dérivation (de la forme $r \in N \rightarrow NN$) $O \rightarrow_r PQ$ satisfait le critère, pour une valeur de son itérateur $x_1 \in \mathcal{I}_r$, la partie P de $droite(r)$ est poussée sur $\hat{\tau}$ et la partie Q est poussée sur φ. Les autres formes de règles ne font pas intervenir φ. On dira simplement que la transition $SATISFAIT(\langle r, i, j, x_1 \rangle)$ le critère et que l'on $POUSSE(\hat{\tau}, \varphi)$.

Le retour en arrière $BACKTRACK$ consiste à dépiler une règle α du sommet de $\hat{\tau}$ et à visiter ses alternatives applicables. Si α est de la forme $N \rightarrow NN$, alors la pile φ est

simultanément popée. Ces alternatives sont de deux types : incrémentation de son itérateur ou soit remplacement par une autre règle obtenue par la fonction $suivant(\alpha)$. Si une de ces options s'applique, alors cette nouvelle valeur est empilée sur $\hat{\tau}$. Si aucune de ces options ne s'applique pour α, alors le processus est repris avec la pile résultante tant que celle-ci n'est pas vide.

Une solution partielle $\hat{\tau}$ est $\in S^{\Delta,OPT}(\gamma)$ si $g(\hat{\tau}) \leq \mathcal{O}^{MIN}(\gamma) + \Delta$ (ou $g(\hat{\tau}) \geq \mathcal{O}^{MAX}(\gamma) - \Delta$), $\hat{\tau} \neq \emptyset$, $\chi^{OPT}(\hat{\tau}) = 0$ (c'est à dire que $\varphi = \emptyset$) et s'il n'y à pas de dérivation possible. On dira alors qu'on $ACCEPTE(\hat{\tau})$.

1.3.7.5 Pseudo-code de l'algorithme 2

L'algorithme complet prend la forme suivante :

algorithme 2 ($a^{MAX} > 1$)

$\varphi \leftarrow \emptyset$
$\langle r, 1, n, x_1 \rangle \leftarrow$ la première règle $\in S|SATISFAIT(\langle r, 1, n, x_1 \rangle)$
SI $\nexists r$ ALORS retourner $\emptyset$
$\hat{\tau} \leftarrow [\langle r, 1, n, 0 \rangle]$
TANT QUE $\hat{\tau} \neq \emptyset$
    SI $SATISFAIT(\hat{\tau}[|\hat{\tau}|])$
        SI $ACCEPTE(\hat{\tau})$
            $imprimer(\hat{\tau})$
            $BACKTRACK(\hat{\tau}, \varphi)$
        SINON
            $POUSSE(\hat{\tau}, \varphi)$
        FIN SI

SINON

$BACKTRACK(\hat{\tau}, \varphi)$

FIN SI

FIN TANT QUE

Nous avons fait usage implicite d'une fonction $imprimer(\hat{\tau})$ sans la définir. Nous sommes intéressés par une décoration $\mathcal{D}^{\Delta,OPT}(\gamma)$ pour un alphabet de sortie Σ^*, or les tuples de $\hat{\tau}$ sont $\langle r, i, j, x_1 \rangle$ définissant complètement l'assignation des règles dans une décomposition valide du problème $D(\gamma)$. Pour obtenir la décoration, il suffit de produire un dictionnaire $symbole(r)\times\mathbb{N}\times\mathbb{N} \rightarrow \Sigma^*\times\Sigma^*$ acceptant les $\langle r, i, j, x_1 \rangle \in \hat{\tau}$ et produisant la sortie sur un vecteur $\Sigma^{|\gamma|}$ correspondant. Toutefois, selon la grammaire, il peut exister plusieurs arbres avec la même décoration et cela peut causer des difficultés. Nous y reviendrons dans la section concernant l'ambigüité.

1.3.7.6 L'algorithme 2 énumère $\{\tau|\tau \in S^{\Delta}(\gamma)\}$ pour les grammaires d'arité 2 en temps $O(n^2\kappa)$ et en mémoire $O(n)$

Chaque itération de la boucle TANT QUE soit empile ou dépile un élément sur $\hat{\tau}$. Chacun de ces éléments correspond à une dérivation ... $w_u \rightarrow_{r_u} w_v$ ... valide dans l'arbre de dérivation donnant $w^{final} \in S^{\Delta,OPT}(\gamma)$. En négligeant les dérivations de la forme $r(i,j) \rightarrow r'(i,j)$ (puisqu'il existe une forme normalisée de $\mathcal{G}_p$), il y a au plus $|\gamma|$ étapes de dérivations $\forall\tau \in S^{\Delta,OPT}(\gamma)$ et la dérivation par une règle demande le balayage $\in O(\mathcal{I})$ de ses itérateurs. Sans tenir compte du travail effectué pour préparer $\mathcal{O}(\gamma)$, avec $\kappa = |S^{\Delta,OPT}(\gamma)|$ et $n = |\gamma|$, l'algorithme est $\in O(\ O(\mathcal{I})\ n\kappa)$. Et maximalement, un itérateur

consiste en l'itération des décompositions valides soit $O(cte)$ pour une règle d'arité $a(r) = 1$, $O(n)$ pour une règle d'arité $a(r) = 2$, et en général $O\left(n^{MAX_{r \in \mathcal{C}_p}(a(r)-1)}\right)$.

Pour une grammaire avec au plus deux règles à droite, l'algorithme 2 est $\in O(n^2\kappa)$.

La quantité de mémoire nécessaire est bornée par la hauteur des piles. Or à tout moment $|\hat{\tau}| + |\varphi| \leq |\tau|$ où τ est l'arbre de w^{final} et le nombre de dérivations $\ldots \rightarrow_{r_u} \ldots w^{final}$ est $\leq n$. Il s'ensuit que la quantité de mémoire maximale utilisée par $\hat{\tau}$ et φ est $\in O(n)$. Cette borne sur l'espace permet une pré-allocation de l'espace mémoire total nécessaire à l'exécution de l'algorithme. En pratique, cela transforme la structure de données dynamique en structure de données statique dans le sens que ceci élimine tout besoin de gestion de la mémoire pendant l'exécution.

1.3.8 Programmation dynamique

La programmation dynamique est une méthode algorithmique qui accélère le calcul de plusieurs types de problèmes d'intervalles récursifs locaux. En particulier des problèmes de décision, d'évaluation et d'optimisation. L'observation clé tient de ce que la solution à un sous-problème local est toujours la même, indépendamment de son contexte. Ainsi, si la solution d'un sous-problème est requise dans le calcul des sous-problèmes qui l'englobent, en gardant cette valeur en mémoire, il devient superflu de calculer à nouveau sa valeur. On fait parfois référence à la *mémoisation* de ces valeurs. Notons que les valeurs des sous-problèmes de tous les intervalles i, j avec $1 \leq i, j \leq |\gamma|$ sont inscrits dans un tableau de taille $|\gamma|^2$. (Il peut arriver qu'un certain nombre d'alternatives soient encodées dans des dimensions additionnelles comme c'est le cas dans une table de MC-Flashfold mais notons

qu'il ne s'agit alors que d'une manière compacte de décrire la situation. Nous nous attardons ici au cas général.)

Il suffit de décrire la méthode par laquelle les cases de ces tableaux sont remplies et le lieu où la réponse se trouve à la fin de l'exécution pour décrire un algorithme en programmation dynamique. Cela se fait en spécifiant des équations de récurrence dans la forme :

$$T_r[i,j] = OPER\begin{cases} f(i,j) & lorsque\ conditions\ x \\ \vdots & \vdots \\ f'(i,j) & lorsque\ conditions\ y \end{cases}$$

où $OPER$ est typiquement choisi parmi des opérateurs tels que $MIN, MAX, SUM, MULT, etc$, où la fonction f est appliquée seulement lorsque la condition x est vraie et où la fonction f' est appliquée seulement lorsque la condition y est vraie. L'ordre de remplissage des tables doit être tel que toutes les conditions nécessaires au calcul d'une $T_r[i,j]$ soient préalablement remplies. Cela proscrit en particulier les agencements cycliques de règles.

1.3.8.1 $\mathcal{A}_p$ un Oracle pour les algorithmes de retour en arrière

Notre but est de disposer des tables de mémoisation pour les $r \in \mathcal{C}_p$ donnant $\mathcal{O}^{OPT}(\gamma[i,j])\ \ \forall_{i,j} 1 \leq i \leq j \leq |\gamma|$ en temps constant tel que l'avons utilisé dans les algorithmes donnant les $\mathcal{S}^{\Delta,OPT}(\gamma)$. Après le remplissage des tables de mémoisation, l'algorithme $\mathcal{A}_p$ donne ces valeurs.

La spécification de $\mathcal{A}_p$ nécessite $\mathcal{C}_p$, $\mathcal{W}_{\mathcal{C}_p}$ et l'ensemble des itérateurs $\mathcal{I}_p$.

L'ensemble des tables de mémoisation est :

$$\mathcal{T}_p = \{T_r | r = symbole(image(x)), \forall x \in \mathcal{C}_p, x \text{ est de la forme } \mathcal{C}_p \rightarrow \mathcal{C}, \} \cup T_\gamma$$

avec $T_\gamma[i,j] = \omega(\gamma[i,j])$, le poids des règles terminales si elles existent sur $\gamma[i,j]$.

L'équation de récurrence de $T_r \in \mathcal{T}_p \backslash T_\gamma$ sur i,j avec $1 \leq i \leq j \leq |\gamma|$ trouve le poids optimal parmi les règles $r_x \in \mathcal{C}_p$ dont $symbole(image(r_x)) = r$:

$$T_r(i,j) = OPT \begin{cases} \underset{x_1, x_2, \ldots \in \mathcal{I}_{r_1}}{OPT} \omega_{r_1}(i, x_1, x_2, \ldots, j) & \text{si } i \leq x_1 \leq x_2 \leq \cdots \leq j \\ \vdots & \vdots \\ \underset{x_1, x_2, \ldots \in \mathcal{I}_{r_m}}{OPT} \omega_{r_m}(i, x_1, x_2, \ldots, j) & \text{si } i \leq x_1 \leq x_2 \leq \cdots \leq j \end{cases}$$

où $\mathcal{I}_{r_x}$ est l'itérateur de la règle $r_x \in \mathcal{C}_p$ dont $\mathcal{W}_{\mathcal{C}_p}(r_x) = \omega_{r_x}$.

1.3.8.2 L'oracle pour l'instance γ du problème P est construit en temps $O(|\gamma|^{MAX(a(\mathcal{C}_p)+1,2)})$

Le travail effectué par la récurrence est fonction de la complexité des récurrences. Si les récurrences sont linéaires (deux symboles non-terminaux), alors la complexité est $\in O(|\gamma|)$, si elles sont quadratiques elle est $\in O(|\gamma|^2)$, etc. Plus généralement, $O(\mathcal{I})$ est la complexité des itérateurs et avec R règles, la récurrence est $\in O(O(\mathcal{I})\ R)$

$\mathcal{A}_p$ fonctionne comme suit. (a) Les $T \in \mathcal{T}_p$ sont initialisées à la pire valeur possible pour OPT. Si $OPT = MAX \in \mathbb{R}$ alors $T[i,j] = -\infty$, si $OPT = MIN \in \mathbb{R}$ alors $T[i,j] = \infty$, etc. (b) $\forall i,j,$ où $1 \leq i \leq j \leq |\gamma|, T_\gamma[i,j] = \omega_{\gamma^l}(\gamma[i,j]), l = j - i + 1$ si $\gamma[i,j] \rightarrow \mathcal{C} \in \mathcal{C}_p$. (c) Les récurrences T_r sont calculées pour des valeurs de $i \ldots j$ grandissantes et couvrant $1 \ldots |\gamma|$ à partir des plus petits intervalles possibles et dans un tel ordre que toutes les

valeurs nécessaires au calcul de $T_r(i,j)$ soient toujours disponibles. (d) Avec γ une instance de P, la valeur optimale pour la solution de $\mathcal{O}^{OPT}(\gamma)$ est $OPT\left(T_{r_s}[1,n]\right)$ avec T_{r_s} correspondants aux symboles de $S \in \mathcal{G}_p$ et la meilleure valeur possible au sous problème $\gamma[i,j]$ via la règle $r \in \mathcal{C}_p$ est $T_r[i,j] = \mathcal{O}^{OPT}(\gamma[i,j])$.

Notons qu'avec $a(\mathcal{C}_p) = MAX\{arité(r) | r \in \mathcal{C}_p^a \rightarrow \mathcal{C}_p\}$, le temps nécessaire pour remplir ces tableaux est $O\left(|\gamma|^{MAX(a(\mathcal{C}_p)+1,2)}\right)$ et de même pour la complexité spatiale. Après le remplissage des tables de mémoisation, $\mathcal{O}^{OPT}(\gamma[i,j])$ est $O(cte)$.

1.3.9 Variation sur une seule pile

Nous avons donné un algorithme qui construit une solution $\hat{\tau}$ dans une pile et qui énumère les $\tau^{final} \in S^{\Delta,OPT}(\gamma)$ par retour en arrière sur cette pile en gardant les parties non explorées des règles dans une seconde pile φ. Nous présentons maintenant l'alternative exploitée par Wuchty dans l'implantation du logiciel RNAsubopt.

Cette alternative consiste à conserver plusieurs solutions partielles le long de la suite de dérivations $w \rightarrow_{s \in S} w_1 \rightarrow_{r_1} \ldots \rightarrow_{r_{u-1}} w_u \rightarrow_{r_u} \ldots \rightarrow_{r_x} w^{final}$ et de procéder à leurs raffinements itératifs en s'assurant que chaque transition appliquée satisfait le critère Δ pour la politique d'optimalité OPT. Lors d'une transition $\ldots w_u \rightarrow_{r_u} w_v \ldots$ les choses suivantes se produisent :

(a) Un symbole non-terminal $\lambda \in N$ associé à l'intervalle $\langle i,j \rangle$ est remplacé par $droite(r_u)$ pour une décomposition satisfaisante $\langle i, x_1, x_2, \ldots, j \rangle$.

(b) Une solution partielle (correspondant à nos $\hat{\tau}$) $\hat{\dot{\sigma}}_u \in S^{\Delta,OPT}(\gamma)$ est augmentée d'une partie de solution donnant $\hat{\dot{\sigma}}_v$.

On construit une pile Π de tuples $\langle w_u, \hat{\dot{\sigma}}_u \rangle$. À chaque itération, on remplace $\langle w_u, \hat{\dot{\sigma}}_u \rangle$ du sommet de la pile ($\Pi[|\Pi|]$) par toutes les dérivations possibles du symbole non-terminal λ choisi dans w_u :

$$\Upsilon = \left\{ \langle w_v, \hat{\dot{\sigma}}_v \rangle \middle| SATISFAIT\left(w_v, \hat{\dot{\sigma}}_v\right), w_u \rightarrow_r w_v, symbole\left(gauche(r)\right) = \lambda \right\}$$

Cet algorithme procède comme suit.

$\Pi \leftarrow \left\{ \langle w_s, \emptyset \rangle \; sur \; 1 \ldots |\gamma| \middle| \; SATISFAIT(w_s, \emptyset) \right\}$
TANT QUE $\Pi \neq \emptyset$
    $\langle w_u, \hat{\dot{\sigma}}_u \rangle \leftarrow \Pi[|\Pi|]$
    $\Pi \leftarrow \Pi[1 \ldots (|\Pi| - 1)]$
    $\lambda \leftarrow$ un symbole non-terminal $\in w_u$
    SI $\lambda = \emptyset$
        $imprimer\left(\hat{\dot{\sigma}}_u\right)$
    SINON
        $\Upsilon \leftarrow$ construire Υ pour $\langle w_u, \hat{\dot{\sigma}}_u \rangle$ et λ
        $\Pi \leftarrow \Pi \cup PILE(\Upsilon)$
    FIN SI
FIN TANT QUE

La version à deux piles et la version alternative effectuent toutes deux le même nombre de décompositions d'intervalles et puisque chaque dérivation n'est effectuée qu'une fois le long d'un arbre de dérivation, elles sont toutes deux $O \in (n\Delta)$ en temps.

L'application de règles $r, arité(r) = 2$, nécessite $j - i$ éléments dans Π. Lorsqu'une telle règle est appliquée à répétition l'espace mémoire grandit $|\Pi| \in O(n^2)$. Avec deux telles règles, $|\Pi| \in O\left(n^{2^2}\right)$ et avec ϕ telles règles $|\Pi| \in O\left(n^{2^\phi}\right)$. Ce qui pourrait s'avérer un clair désavantage pour la version alternative.

1.3.10 Variation sur $\mathcal{A}_p$ (k-best parsing)

Nous avons spécifié l'utilisation de $\mathcal{A}_p$ pour obtenir $\mathcal{O}^{\Delta,OPT}(\gamma[i,j])$ dans le but d'obtenir en temps constant les transitions satisfaisantes à la construction *top-down* des solutions partielles $\hat{\tau}$ et $\hat{\sigma}$ compatibles avec les $\tau \in S^{\Delta,OPT}(\gamma[i,j])$. Il est également possible de modifier $\mathcal{A}_p$ de manière à produire directement les ξ_i dans l'approche *bottom-up*. Une approche particulièrement intéressante utilisant des queues de priorités directement dans les cellules des tables de programation dynamique nous avait échapée (Huang and Chiang 2005) et nous en présentons l'idée ici.

Pour réaliser cette méthode, les équations de récurrences et les tables sont augmentées afin de mémoiser non seulement les valeurs optimales mais également les ensembles des sous arbres $\xi_{T_r[i,j]} = \{\tau_{\langle i,j\rangle_1}, \tau_{\langle i,j\rangle_2}, \dots, \tau_{\langle i,j\rangle_x}\}$ dont les fusions correspondent aux règles $r \in \mathcal{C}_p$. Ici, les $\xi_{T_r[i,j]}$ font références aux ensembles d'arbres, solutionnant le sous problème $\gamma[i,j]$, dont les racines sont prescrites par la règle $r \in \mathcal{C}_p$ tel que mémoisé dans la table T_r.

L'idée générale est d'augmenter la méthode des pointeurs à rebours utilisés lorsqu'on ne désire qu'une seule solution $\tau \in S^{OPT}(\gamma)$. Il est possible d'encoder $\tau°$ dans des tables $\widehat{B_r} \in \widehat{\mathcal{B}_p}$ donnant les décompositions appliquées à la règle r sur l'intervalle

$(i,j) \in \tau^{\circ}$, les tuples $< r, i, x_1, x_2, \ldots, j >$ en correspondance avec les $T_r \in \mathcal{T}_p$. Étant données les $\widehat{\mathcal{B}_p}$, il est facile de reconstruire τ°. De même, connaissant les κ solutions dans $\mathcal{S}^{\kappa*,OPT}(\gamma)$, construisons les $B_r \in \mathcal{B}_p$ dont chaque entrée est une queue de priorité donnant au plus les $1..\kappa$ meilleures solutions selon OPT.

Par retours en arrière on retrouve les éléments de $\mathcal{S}^{\kappa*,OPT}(\gamma)$ en connaissant les $\mathcal{B}_p$. Or on peut calculer $\mathcal{B}_p$ en même temps que l'on remplit les $T_r \in \mathcal{T}_p$ pendant l'exécution de $\mathcal{A}_p$ en modifiant les fonctions de poids $\omega_r \in \mathcal{W}_p$ et les équations de récurrences mais l'algorithme strictement naïf correspondant est inefficace. La combinaison de trois accélérations donne un algorithme efficace pour lister $\mathcal{S}^{\kappa*,OPT}(\gamma)$, algorithme 3 dans (Huang and Chiang 2005)

Ces trois accélérations sont :

1. Les ω_r retournent désormais les vecteurs des κ meilleures solutions obtenues $\overrightarrow{\omega_r} = c + OPT\left(\overrightarrow{\omega_{r.1}}(\ldots) + \overrightarrow{\omega_{r.2}}(\ldots) + \cdots + \overrightarrow{\omega_{r.a}}(\ldots)\right)$ et il existe κ^a telles combinaisons. Mais puisque les $\overrightarrow{\omega_{r.\imath}}(\ldots)$ sont ordonnées, leurs combinaisons optimales constituent une frontière sur les indices dans les vecteurs des $\overrightarrow{\omega_{r.\imath}}(\ldots)$. Seules $a\kappa$ telles combinaisons doivent être considérées.

2. Les équations de récurrences choisissent les κ meilleures solutions parmi les règles et les décompositions. Certainement, aucune des solutions retenues ne sera moins bonne que la $\kappa^{ième}$ retenue. Les valeurs calculées par les $\overrightarrow{\omega_r}$ peuvent être limitées à celles qui sont meilleures qu'un seuil maintenu dans une queue de priorité.

3. Il n'est pas nécessaire de calculer toutes les κ entrées pour les intervalles qui ne font pas partie d'au moins une solution. En ne calculant les $B_r[i,j]$ qu'au dernier moment possible pendant l'énumération des $\tau \in \mathcal{S}^{\kappa*,OPT}(\gamma)$ et en propageant ce calcul de manière paresseuse, il est possible de ne remplir que les cases qui se trouvent à la frontière. Il reste néanmoins nécessaire de calculer les valeurs optimales de chaque règle pour chaque indice avec $\mathcal{A}_p$.

L'efficacité rapportée en pire cas est $O(E + |D_{MAX}|\kappa \log \kappa)$ où E est la complexité de $\mathcal{A}_p$ et $|D_{MAX}|$ et la longueur du plus long itérateur (notre $O(\mathcal{I})$). Pour une grammaire hors-contexte, la complexité est $O(n^3 + n\kappa \log \kappa)$.

1.3.11 Une approche top-down pour trouver $S^{\kappa,OPT}(\gamma)$ et $\left(S^{\kappa*,OPT}(\gamma)\right)$

Il arrive que le taux de croissance de W relativement au nombre de solutions $\left(dW = \frac{d|S^{\Delta,OPT}(\gamma)|}{d\Delta}\right)$ soit inconnu. Notamment, dans le cas de la prédiction de structures secondaires d'ARN, ce taux de croissance dépend de la séquence elle-même. Aussi il est impossible d'évaluer analytiquement *a priori* la valeur de Δ^κ pour avoir $\left|S^{\Delta^\kappa,OPT}(\gamma)\right| = \kappa$.

Cependant il est possible d'approcher Δ^κ en invoquant l'algorithme à deux piles avec des estimés $0 \leq \cdots < \Delta' < \Delta'' < \cdots \leq \Delta^\kappa$ de plus en plus rapprochés tout en mesurant dW à la frontière. Avec $T^\Delta(t)$ le temps pour calculer $S^{\Delta=t,OPT}(\gamma)$ et $T^\kappa(t)$ le temps pour calculer $S^{\kappa,OPT}(\gamma)$, si $T^\Delta(\Delta'') \leq 2T^\Delta(\Delta')$, alors avec cette approche, $T^\kappa(\Delta^\kappa) \leq 2T^\Delta(\Delta^\kappa)$ puisque

$$\sum_{i=0}^{n} \frac{1}{2^i} \leq 2$$

Lors de la dernière itération, il est possible que la valeur de Δ soit surestimée. On met alors les solutions dans un tas (*heap*) de taille κ et on rejette les instances τ dont $W(\tau) > W(\tau^\kappa)$. Cet travail est $O(\kappa \log \kappa)$. On énumère $S^{\kappa,OPT}(\gamma) \in O\big(\, O\big(\mathcal{A}_p\big) + O(\mathcal{I})\, |\gamma|\, \kappa + \kappa \log \kappa\big)$ ou, avec $|\gamma| = n$, si l'arité de la grammaire est 2 : $O(n^3 + n^2\kappa + \kappa \log \kappa)$.

Pour obtenir $S^{\kappa*,OPT}(\gamma)$, il suffit de tronquer la solution après κ solutions.

1.3.12 Ressources

Algorithme	Ensemble énuméré	Temps	Mémoire
Algo 2	$\mathcal{S}^{\Delta,OPT}(\gamma)$	$O(n^3 + n^2\kappa)$	$O(n^2)$
Algo 2	$\mathcal{S}^{\kappa,OPT}(\gamma)$ ou $\mathcal{S}^{\kappa*,OPT}(\gamma)$	$O(n^3 + n^2\kappa + \kappa \log \kappa)$	$O(n^2 + \kappa)$
k-best	$\mathcal{S}^{\kappa,OPT}(\gamma)$	$O(n^3 + n\kappa \log \kappa)$	$O(n^2\kappa)$ note : voir texte

Sur entrée $\gamma = \Gamma^n$, l'algorithme 2 utilise les ressources suivantes.

L'énumération des décompositions possibles données par les itérateurs $O(\mathcal{I})$ est déterminée par l'arité maximale d'une grammaire par $O\left(n^{MAX_{r\in\mathcal{C}_p}(a(r))-1}\right)$. Pour une grammaire d'arité 2, $O(\mathcal{I}) = O(n)$.

$\mathcal{A}_p \in O\left(n^2 O(\mathcal{I})\right)$ et pour une grammaire d'arité 2, $\in O(n^3)$.

L'énumération des $\tau \in S^{\Delta,OPT}(\gamma)$ avec Δ spécifié en entrée et $\Delta = \Delta^{\kappa}$ (ie : $|S^{\Delta,OPT}(\gamma)| = \kappa$), le retour en arrière est $O\left(O\left(\mathcal{A}_p\right) + O(\mathcal{I})n\kappa\right)$ et avec une grammaire d'arité 2 $O(n^3 + n^2\kappa)$.

L'énumération des $\tau \in S^{\kappa,OPT}(\gamma)$ est $O\left(\ O\left(\mathcal{A}_p\right) + O(\mathcal{I})\ |\gamma|\ \kappa + \kappa \log \kappa\right)$ et avec une grammaire d'arité 2 : $O(n^3 + n^2\kappa + \kappa \log \kappa)$.

La complexité raportée pour l'algorithme *k-best* de (Huang and Chiang 2005) pour le même problème est $\in O(n^3 + n\kappa \log \kappa)$ et celui-ci serait asymptotiquement $\frac{n}{\log \kappa}$ fois plus rapide ce qui lui confère un avantage certain. Selon les constantes cachées et le ratio $\frac{n}{\kappa}$, l'avantage sera plus ou moins marqué cependant. Avec $n \gtrsim \kappa$, les deux approches sont dominées par le cube et l'effet aura le plus d'importance lorsque $n \ll \kappa$ pour éventuellement s'estomper dans la situation invressamblable où $\log \kappa \to n$.

D'autre part, les besoins en mémoire sont possiblement aussi grand que $\in O(n^2\kappa)$ pour *k-best* (un heap par cellule des tables de programmation dynamique) mais seulement $O(n^2 + \kappa)$ (les tables plus un seul heap, de taille κ) pour notre algorithme 2 (en configuration $S^{\kappa,OPT}(\gamma)$) et $O(n^2)$ (en configuration $S^{\Delta,OPT}(\gamma)$). Cela pourrait limiter la taille réalisable de κ avec l'approche *k-best*.

Par exemple, nous avons récement eu besoin d'énumérer 10^7 structures d'un riboswitch de longueur environ 100nt. et nous envisageons des cas plus difficiles. L'accélération attendue $O\left(\frac{n}{\log \kappa}\right)$ dans cette situation est proportionnelle à $c\frac{100}{\log 10^7} \approx 5c$ pour une valeur inconnue de c mais la quantité de mémoire en pire cas pour *k-best* est inconnue et pourrait être $\in O(n^2\kappa)$ ce qui pourrait defenir problématique pour de telles instances.

1.3.13 Intégrer des sous-solutions prédéfinies (les masques)

Une solution peut être partiellement prédéfinie pour des raisons étrangères au problème d'optimisation. Par exemple la recherche d'un plus court chemin pour livrer une paquet peut se faire sous la contrainte d'utiliser un certain cargo entre deux villes ou des

connaissances antérieures peuvent indiquer que certains nucléotides forment une paire dans une molécule d'ARN. Pour spécifier ces contraintes, nous utilisons des *masques* qui prennent la forme de fragments de décorations $\hat{\hat{\delta}} \in \mathcal{D}(\gamma)$. L'implantation des masques se fait en limitant les choix possibles dans $\mathcal{A}_p$ et ne posent pas de difficultés.

1.3.14 Ambigüité, complétude et redondance

Nous avons montré que les problèmes d'intervalles récursifs sont solvables par la combinaison d'un ensemble de règles $r \in \mathcal{C}_p$ soit de la forme $\gamma^l \rightarrow \mathcal{C}$ pour les sous problèmes solvables en temps constant ou de la forme $\mathcal{C}^a \rightarrow \mathcal{C}$ pour les sous problèmes décomposables. De $\mathcal{C}_p$ nous avons montré comment construire $\mathcal{G}_p$et $\mathcal{A}_p$ avec pour but d'énumérer les arbres de composition $\tau \in S(\gamma)$ lesquels sont les solutions a l'instance γ du problème P et nous exprimons ces solutions grâce à sa décoration $\mathcal{D}(\tau)$. Mais il nous reste plusieurs questions. Sommes nous bien capable d'identifier $\mathcal{C}_p$ pour modéliser un système physique comme le repliement de l'ARN? Dans quelle mesure sommes nous certains que $\mathcal{C}_p$ couvre toutes les possibilités? Existe-t-il un meilleur ensemble $\mathcal{C}'_p$ pour solutionner P? Quel est le comportement normal attendu et quels seraient les comportements anormaux? Bref quel niveau de confiance peut-on accorder au système résultant?

Le problème de l'ambiguité des langages est vaste, vieux et son traitement est toujours d'actualité. Bien que le sujet mérite une discussion approfondie, bornons nous ici a mentionner deux catégories d'ambiguité d'importance pour nous. Selon la nommenclature retenue par Christian Höner zu Siederdissen dans sa dissertation de Doctorat (zu Siederdissen 2013) deux niveaux sont à retenir : l'*ambiguité syntactique* et l'*ambiguité structurale*. L'*ambiguité sémantique* y est définie comme une généralisation de l'ambiguité structurale et dépasse le contexte de nos problèmes.

Une grammaire est *syntactiquement ambiguë* si elle accèpte plus qu'un *parse tree* (nos τ ou ensemble de composition de règles) pour une entrée donnée. Dans plusieurs situations on demande à la grammaire d'être dépourvu d'ambiguité syntactique. Un

compilateur serait bien inutile s'il interprétait du code en langage C de plusieurs manières différentes! La question est différente pour nous puisque nos grammaires sont nécessairement syntactiquement ambiguës, autrement nous ne pourrions produire qu'une seule structure d'ARN par séquence.

Une grammaire est *structurellement ambiguë* si deux *parse trees* conduisent à la même décoration (nos $\mathcal{D}(\tau)$). Ce type d'ambiguité ne dépend pas seulement de la grammaire mais également de l'algorithme de décoration.

L'ambiguité syntactique et l'algorithme 2. Le langage qu'une grammaire peut générer est $L(\mathcal{G}_p) = \{\gamma | \gamma \in \Gamma^+, \exists \rightarrow_{s \in S} \rightarrow^* \gamma\}$ et plusieurs grammaires peuvent décrire le même langage $L(\mathcal{G}_{P1}) = L(\mathcal{G}_{P2})$. Si $\mathcal{G}_{P1}$ comprend $S \rightarrow SS$ et $\mathcal{G}_{P2}$ comprends $S \rightarrow SS$ et $S \rightarrow SSS$ par exemple. Mais les arbres trouvés par $\mathcal{G}_{P2}$ seront plus nombreux que ceux trouvés par $\mathcal{G}_{P1}$ et clairement les deux grammaires ne sont pas équivalentes. Plusieurs applications informatiques demandent des grammaires donnant un arbre unique sur toutes entrées. Par exemple, un langage de programmation doit spécifier une seule sémantique si l'on veut encoder des algorithmes précis. Une grammaire qui ne peut garantir cette unicité est dit *ambiguë*. Et il s'avère qu'on ne peut pas toujours prouver qu'une grammaire soit dépourvue d'ambiguité (Aho and Ullman 1972) bien qu'il existe des outils informatiques permettant dans certains cas de faire cette vérification (Braband, Giegerich et al. 2010). Le plus embarassant est certainement le constat que pour certains langages, dits intrinsèquement ambigus, il n'existe pas de grammaire dépourvu d'ambiguité (Harrison 1978). Cela n'est pas un problème pour nous car nos grammaires sont toutes ambiguës (sinon, il ne ferait aucun sens de chercher de nombreuses solutions pour une instance). La complétude de l'ensemble est l'autre face de ce problème. Nous devons choisir les règles de manière systématique de manière à ce que nos grammaires couvrent bien toutes les possibilités et seulement les possibilités permises par le problème. Le problème du *sinon pendouillant* (*dangling else*) est bien connu et demande d'augmenter la spécificité des règles afin de circonscrire exactement une sémantique précise.

Un problème plus criant concerne l'ambiguité structurale qui prend la forme suivante dans nos travaux : une solution $\mathcal{S}^{\phi}(\gamma) = \{\langle \mathcal{D}(\tau), W(\tau)\rangle | \tau \in \phi(S(\gamma))\}$ est structurellement ambiguë (que nous disons aussi une solution redondante) si au moins une de ses décorations apparait plus d'une fois. Remarquons que l'approche avec représentation en forêt n'encours pas cette complication.

Prenons $s_1, s_2 \in \mathcal{S}^{\phi}(\gamma)$ avec $\mathcal{S}^{\phi}(\gamma)$ une solution redondante, $s_1 = \langle \mathcal{D}(\tau_1), W(\tau_1)\rangle$ et $s_2 = \langle \mathcal{D}(\tau_2), W(\tau_2)\rangle$ avec $\mathcal{D}(\tau_1) = \mathcal{D}(\tau_2)$. $\tau_1 \neq \tau_2$ puisqu'ils ont été produits par retour en arrière soit avec l'algorithme sur deux piles ou soit sur les V_r avec l'algorithme κ-OPT.

Ces doublons peuvent facilement être éliminés en triant $\mathcal{S}^{\phi}(\gamma)$. Ils sont le résultat attendu d'une grammaire ambiguë. Dans certaines situations, il peut s'avérer nécessaire de réviser la grammaire pour les éliminer. Mais ce faisant, pour des raisons évidentes, il est impératif de s'assurer que le processus ne nuit pas à la complétude de $S(\gamma)$. Notamment on doit réviser les règles dans les situations suivantes :

1. Les doublons sont en nombres est trop grand et leurs présence augmente considérablement le temps de calcul.

2. On a besoin d'un nombre exact de solutions pour un Δ donné. Notons que l'énumération des $\mathcal{S}^{\kappa,OPT}(\gamma)$ ne nécessite qu'une approximation.

3. On veut ré-estimer les poids *a posteriori* (*reranking*).

4. On veut calculer la fonction de partition.

1.3.14.1 Note sur la représentation en forêt et les doublons

Une représentation en forêt (dénotons la $\mathcal{F}$) de tous les arbres de dérivation (plutôt que notre simple pile représentant $\hat{\tau}$) pendant la passe de retour en arrière éliminerait la possibilité des doublons. Une telle représentation est possible avec une grammaire non-ambiguë. L'énumération des décorations des arbres dans $\mathcal{F}$ peut être réalisée sans répéter les régions communes des arbres partageant une même racine ce qui laisse envisager un

gain de vitesse. Un certain compromis dans la balance des ressources nécessaires (mémoire/temps) accompagne alors l'accroissement de complexité du retour en arrière puisqu'il faut toujours conserver l'ensemble de la forêt en mémoire. Cela dit, la pile $\hat{\tau}$ de l'algorithme 2 effectue une visite de $\mathcal{F}$ sans que celle-ci n'existe explicitement. Il est donc possible d'énumérer les décorations à la même vitesse que si la forêt était gardée en mémoire. La stratégie (le calcul incrématal des décorations) consiste à construire la décoration pendant le retour en arrière et d'en effacer les régions révisées (les $r\big((i,j)\big)$) lorsque $\hat{\tau}$ est popée. Cette stratégie d'autant plus efficace que les arbres dans $\mathcal{S}^{\Delta,OPT}(\gamma)$ ne diffèrent que par des décompositions sur de petits intervales. Nous utilisions cette stratégie dans une version antérieure de nos algorithmes mais nous l'avons abandonée pour deux raisons d'ordre pratique. (1) Le gain d'efficacité est mitigé en pratique car chaque décoration doit ultimement être exprimée soit à l'écran ou dans un fichier. Cette expression est sans contredit au moins $O(c|\gamma|)$ où c est hors du contrôle de l'algorithme nécéssitant l'intervention du système d'exploitation. Et (2) on souhaite évaluer des décorations alternatives (que nous ne décrivons pas ici) pour certaines situations. Or l'évaluation incrémetale de la décoration doit être effectuée pendant la phase de retour en arrière et cela entremêle la logique de retour en arrière avec celle de la décoration. En pratique, ces deux considérations compliquent donc l'encodage de l'algorithme sans que l'efficacité réelle soit améliorée.

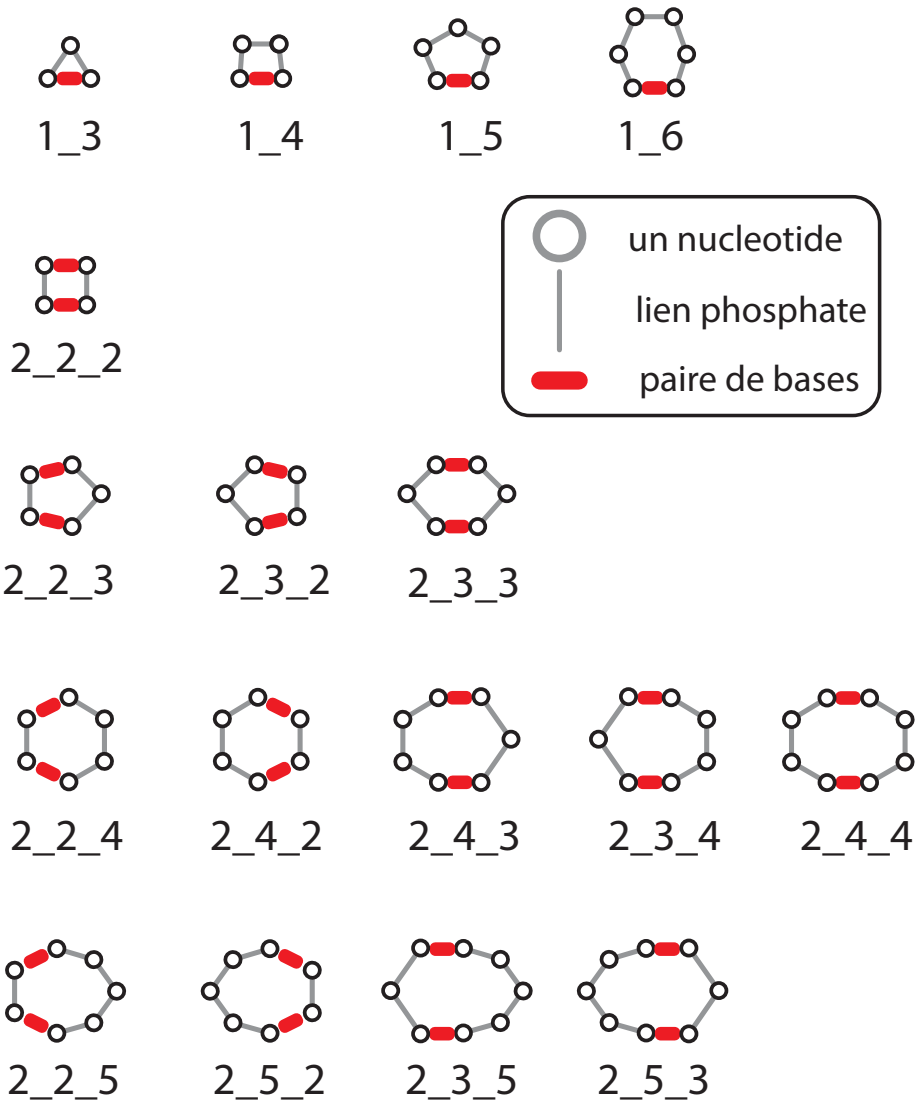

Figure 4. Les différents types de NCM.

1.4 Le modèle d'énergie des Motifs Nucléotidiques Cycliques

Avant de détailler la grammaire de MC-Flashfold nous devons comprendre les détails du modèle d'énergie des NCM et ce modèle est le sujet de cette section. Chaque nucléotide en interaction (dé-)stabilise la structure, certaines séquences sont mieux adaptées à certains arbres que d'autres. Ce biais est l'effet cumulatif de la stabilité de chaque nucléotide dans son environnement chimique. Dans le modèle de structure secondaire nous estimons la qualité de l'assignation d'un arbre à une structure en faisant la somme des contributions de chaque nucléotide. Cette contribution prend la forme d'un niveau d'énergie ou, de manière équivalente, d'une valeur de probabilité. De nombreux autres facteurs contribuent à la qualité de cette assignation. Certains sont intra-moléculaires (pseudo-nœuds, triplets et interactions rares) et d'autres sont inter-moléculaires (interactions avec des métabolites et/ou des macromolécules, variations de force ionique, etc) dictant une borne supérieure sur l'exactitude de la prédiction de structure secondaire.

Il est impossible de calculer indépendamment et explicitement la qualité d'assignation de chaque arbre à chaque séquence car les arbres et les séquences sont bien trop nombreux. Au lieu de cela, nous utilisons la programmation dynamique et le retour en arrière tels que décrits dans la section précédente pour calculer les meilleurs arbres pour des séquences données (Zuker and Stiegler 1981; Wuchty, Fontana et al. 1999; Rivas 2013).

Le modèle des NCM a été introduit dans (Parisien and Major 2008) où deux groupes de NCM sont énumérés. Il y a ceux qui se composent d'une seule chaine de nucléotides et ceux composés de deux chaines (la Figure 4 montre les différents types de NCM).

Avec $\mathcal{R}_s = \langle N, A \rangle$ le graphe (avec $N = \{s_i, 1 \leq i \leq n\}$ et $A = \{(s_i, s_{i+1}) | 1 \leq i < n\} \cup \{(s_i, s_j) | (i,j) \in 2D^+(s)\}$) donnant la combinaison d'une annotation des paires dans une structure connue $3D(s)$ d'une séquence $s \in \Gamma_{ARN}^n$ et de sa séquence primaire (Gendron, Lemieux et al. 2001), le *cycle nucléotidique minimal* (Lemieux and Major 2006) $\mathcal{Q} = \langle N_Q, A_Q \rangle$ est un sous-graphe de degré exactement 2 de $\mathcal{R}_s$

dont il est impossible de remplacer un de ses sous-graphes par un plus petit sous-graphe de $\mathcal{R}_s$. De plus, un cycle Q est un *motif* s'il est récurent dans un échantillonnage non redondant des structure $3D$ connues. Les NCM ont été choisis parmi les motifs dépourvus de triplets et contenant au plus deux paires de nucléotides dans A_Q.

Les NCM sont concrétisés en instanciant les types de NCM montrés à a Figure 4. Ceci est fait en leurs assignant des séquences de nucléotides; notés 1_4:GAGA, 2_2_2:GAGA et ainsi de suite. L'idée derrière le modèle de NCM est que ceux-ci capturent les détails des interactions d'empilements et de pairages entre nucléotides ainsi que les autres interactions atomiques de type pont-H, coordinations électrostatiques et géométriques. La promiscuité atomique révélée dans ces cycles permet en effet la réalisation préférentielle de certaines configurations à la défaveur d'autres configurations. Les préférences pour certains NCM observés parmi les ARN dont la structure tridimensionnelle est connue indiquent qu'en effet certains types de NCM sont rarement instanciés par certaines suites de nucléotides et cet effet est une conséquence des affinités des bases entre elles. Il est donc possible d'énumérer les NCM et d'en établir une liste des fréquences naturelles, constituant un véritable alphabet probabiliste utilisé pour la prédiction de structure. Ainsi, la détermination de la structure secondaire d'un ARN consiste à trouver l'arrangement de types de NCM qui maximise la fréquence naturelle de ceux-ci étant donnée la séquence. Notons que les NCM se chevauchent (Figure 4) et que les fréquences de chevauchement doivent également être tenues en ligne de compte.

La figure 4, énumère Θ, les types de NCM. Chaque $\theta \in \Theta$ est un graphe $\langle N_\theta, A_\theta \rangle$ (analogue à Q mais dont les nœuds sont rendus génériques) et qui peut être *instancié* en NCM en lui assignant une séquence $\gamma \in \Gamma_{ARN}^{l}$ avec $l = |N_\theta|$. L'instanciation des éléments de Θ par toutes les séquences possibles, génère $\mathcal{N}$, l'ensemble des NCM. La fréquence f_x d'observation de chaque $x \in \mathcal{N}$ dans les $\mathcal{R}_s$ d'un ensemble OBS non-biaisé et représentatif de structures connues est utilisé pour en dériver sa *probabilité a priori,* $p_x = \frac{f_x}{\sum_{y \in \mathcal{N}} f_y}$, qui est mise en adéquation avec la stabilité thermodynamique de $x \in \mathcal{N}$ via le *facteur de Boltzmann* $p_x = e^{-\Phi_x/RT}$. Ici, Φ_x est utilisé plutôt que G_x simplement pour indiquer qu'il

s'agit d'une pseudo-énergie, mesurée par échantillonnage des états et non une mesure thermique directe. Notons que la somme des fréquences utilisées pour normaliser les p_x est aussi connue sous le nom de *fonction de partition* et représentée par le symbole Z. L'importance du choix d'OBS, l'ensemble non-biaisé et représentatif de structures, détermine ce qu'on entend par *probabilité a priori*. Idéalement, OBS reflèterait une distribution uniforme des séquences possibles, cependant cet ensemble est constitué des observations disponibles (Berman, Westbrook et al. 2000) et n'est pas explicitement normalisé sur Γ^*_{ARN}. Il est par contre élagué de séquences trop homologues par un processus *ad hoc* (Parisien and Major 2008) et les $f_x | x \in \mathcal{N}$ sont initialisées à 1 plutôt qu'à 0 lors de leurs décompte pour tenir compte de la taille limitée de OBS. Les p_x doivent donc être comprises comme des probabilités *étant donné* que les séquences sont d'origines biologiques et représentées dans OBS.

La minimisation d'une fonction de probabilité, nécessite la multiplication de tout petits nombres à la chaine. Afin d'éviter cette situation, nous remplaçons les produits par des sommes de logarithmes. Ainsi, la pseudo-énergie de $x \in \mathcal{N}$ est obtenue à partir des fréquences observées des NCM dans OBS selon $\Phi_x = -RT \ln \frac{f_x}{Z}$.

L'assignation de NCM à une séquence est illustrée à la Figure 2. Dans notre modèle, des suites de NCM sont séparées par des segments de séquences auxquelles aucun NCM ne correspond. Notons $\mathfrak{N} = [\alpha_1, \alpha_2, \dots, \alpha_v]$ une suite $\alpha_i \in \mathcal{N}$ de NCM contigües telle que les paires (α_i, α_{i+1}) partagent la paire A•B de nucléotides et $\mathfrak{S}(s) = [\mathfrak{N}_1, \mathfrak{N}_2, \dots, \mathfrak{N}_w]$ la suite des $\mathfrak{N}$ assignées de gauche à droite le long d'une séquence $s \in \Gamma^n_{ARN}$. L'ordre des $\mathfrak{N}_i$ dans $\mathfrak{S}(s)$ est déterminé par l'ordre des plus petits indices dans s des nucléotides instanciant les $\alpha \in \mathfrak{N}_i$. De cette manière, à une assignation de NCM $\mathfrak{S}(s)$, correspond une et une seule décoration $2D(s)$.

Si les énergies des NCM d'une structure étaient indépendantes, en première approximation, nous pourrions obtenir l'énergie totale d'une structure comme étant la somme de l'énergie de ses composantes, mais nous savons que cela est inexact :

$$\Phi\big(\mathfrak{S}(s)\big) \neq \sum_{\mathfrak{N}_i \in \mathfrak{S}(s)} \sum_{x \in \mathfrak{N}_i} \Phi_x$$

Il a été observé que les NCM partageant une paire de nucléotides s'influencent mutuellement de proche en proche par le biais de leurs paires communes, favorisant certaines combinaisons et en défavorisant d'autres. Il a également été constaté que la contribution individuelle des paires de bases dans OBS ne peut être négligée. Une meilleure approximation de l'énergie d'un segment de NCM est donnée par

$$\Phi(\mathfrak{N}) = \Phi(A \bullet B) + \sum_{1 \leq i < |\mathfrak{N}|} \big(\Phi(\alpha_i) + \Phi(\alpha_i, \alpha_{i+1}) + \Phi(C \bullet D)\big) + \Phi\big(\alpha_{|\mathfrak{N}|}\big) + \Phi(E \bullet F)$$

où $\alpha_i, \alpha_{i+1} \in \mathfrak{N}$, $\Phi(C \bullet D)$ est la pseudo-énergie de la paire de nucléotides partagée par α_i, α_{i+1} calculée par échantillonnage dans OBS et normalisée comme décrit précédemment pour les NCM. $\Phi(A \bullet B)$ et $\Phi(E \bullet F)$ sont les pseudo-énergies des paires terminales (n'appartenant qu'à un seul NCM). La paire $E \bullet F$ n'existe pas si le dernier NCM de la suite n'a qu'une paire de base et alors $\Phi(E \bullet F) = 0$. L'expression $\Phi(\alpha_i, \alpha_{i+1})$ est problématique cependant car notre liste de NCM comporte environ $3{\times}10^5$ NCM, leurs valeurs de Φ utilisent $\approx 1Mo$ de mémoire. La liste de leurs cooccurrences comprendrait $9{\times}10^{10}$ valeurs occupant $\approx 400Go$ rendant sont utilisation difficile. MC-Fold approxime $\Phi(\alpha_i, \alpha_{i+1}), \alpha \in \mathfrak{N}$ avec $\Phi(\theta_i, \theta_{i+1}, A, B)$, où $\theta \in \Theta$ et $A, B \in \Gamma_{ARN}$, $A \bullet B$ réduisant la taille du domaine de la fonction de $9{\times}10^{10}$ à $|\Theta|^2{\times}\Gamma_{ARN}^2 = 305$.

$$\Phi(\mathfrak{N}) = \Phi(A \bullet B) + \sum_{1 \leq i < |\mathfrak{N}|} \big(\Phi(\alpha_i) + \Phi(\theta_i, \theta_{i+1}, C, D) + \Phi(C \bullet D)\big) + \Phi\big(\alpha_{|\mathfrak{N}|}\big) + \Phi(E \bullet F)$$

MC-Fold utilise un terme supplémentaire nommé *junction* dans cette équation. Junction est une $\Phi(\theta_i, \theta_{i+1})$ donnant l'énergie de transition du type de NCM θ_i au type de NCM θ_{i+1}. Remarquons que cette valeur est incluse dans $\Phi(\theta_i, \theta_{i+1}, C, D)$ avec $\Phi(\theta_i, \theta_{i+1}, C, D) = junction + hinge$. Dans cette version, la partie *hinge* collige les différents *types* de paires de nucléotides (les configurations géométriques variées voir

(Saenger 1984) pour de plus amples détails) permis à la paire $C \bullet D$ partagée par les NCM θ_i et θ_{i+1}. Puisque nous cherchons à accélérer le calcul de MC-Fold et non pas le réécrire, nous utilisons ses tables d'énergies ce qui nous donne la version suivante finale de la fonction objectif à minimiser pour la séquence s :

$$\Phi\big(\mathfrak{S}(s)\big) = \sum_{\mathfrak{N}\in\mathfrak{S}(s)} \Bigg(\Phi_{pair}(A \bullet B) + \sum_{1\le i<|\mathfrak{N}|} \Big(\Phi_{NCM}(\alpha_i) + \Phi_{junction}(\theta_i, \theta_{i+1}) + \Phi_{hinge}(\theta_i, \theta_{i+1}, C, D) + \Phi_{pair}(C \bullet D) \Big) + \Phi_{NCM}\big(\alpha_{|\mathfrak{N}|}\big) + \Phi_{pair}(E \bullet F) \Bigg)$$

La Figure 3 montre d'autres éléments structuraux que les NCM et ces structures sont reconnues par la grammaire mais pour l'heure seule la fonction objectif donnée contribue au calculs de MC-Flashfold. MC-Fold considère en plus l'énergie d'empilement coaxial qu'on observe lorsque deux hélices entrent en interaction par empilement (*co-axial stacking*).

La méthode pour sommer les contributions énergétiques de chaque élément de la fonction objectif est montrée à la Figure 5. Nous y montrons deux modèles d'application de la fonction objectif : le modèle de fermeture des NCM (utilisé dans MC-Flashfold) et le modèle de fermeture des paires de bases (utilisé dans MCFold2). Le modèle de fermeture des paires de bases à l'avantage de permettre de calculer simplement la table de partition des paires de bases mais nécessite de calculer les paires de bases dans toutes leurs configurations possibles; i.e. : à la Figure 5 le calcul de l'extension de la paire de base B-F nécessite le calcul de toutes les combinaisons de NCM-I avec toutes les combinaisons de NCM-II. Un fragment de l'arbre des énumérations nécessaires pour le calcul sur le modèle de la fermeture des NCM est montré à la Figure 6 et les étapes clés y sont montrées.

La fréquence de certains *hinges* et certains NCM est 0 et il est incertain que ceci indique qu'il est strictement impossible d'observer ces situations ou si plutôt cela repose sur un échantillonnage de taille finie. Nous permettons l'utilisation de ces constructions en utilisant une valeur d'énergie de min $(1, hinge + pair)$ pour toute paire (*i,j*) assurant que, si le contexte en dicte l'usage, il soit possible de les utiliser néanmoins.

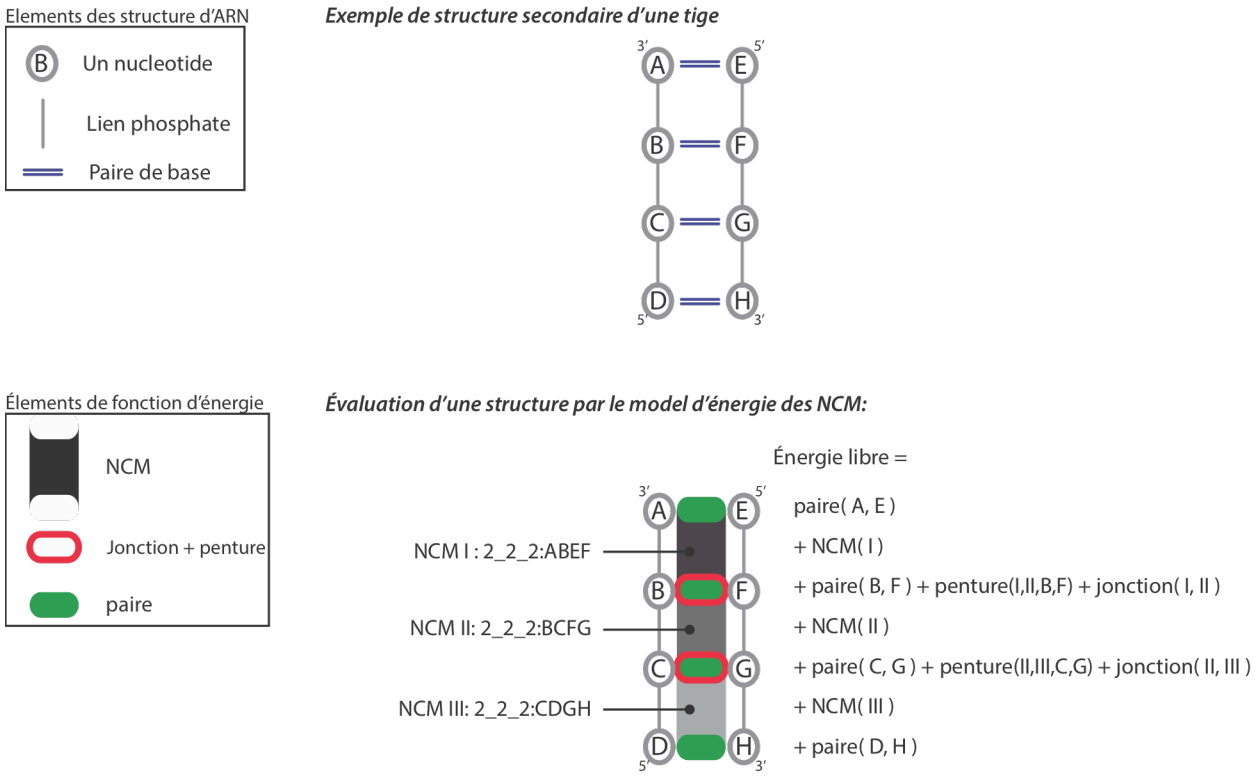

Figure 5: Le modèle d'énergie des NCM.

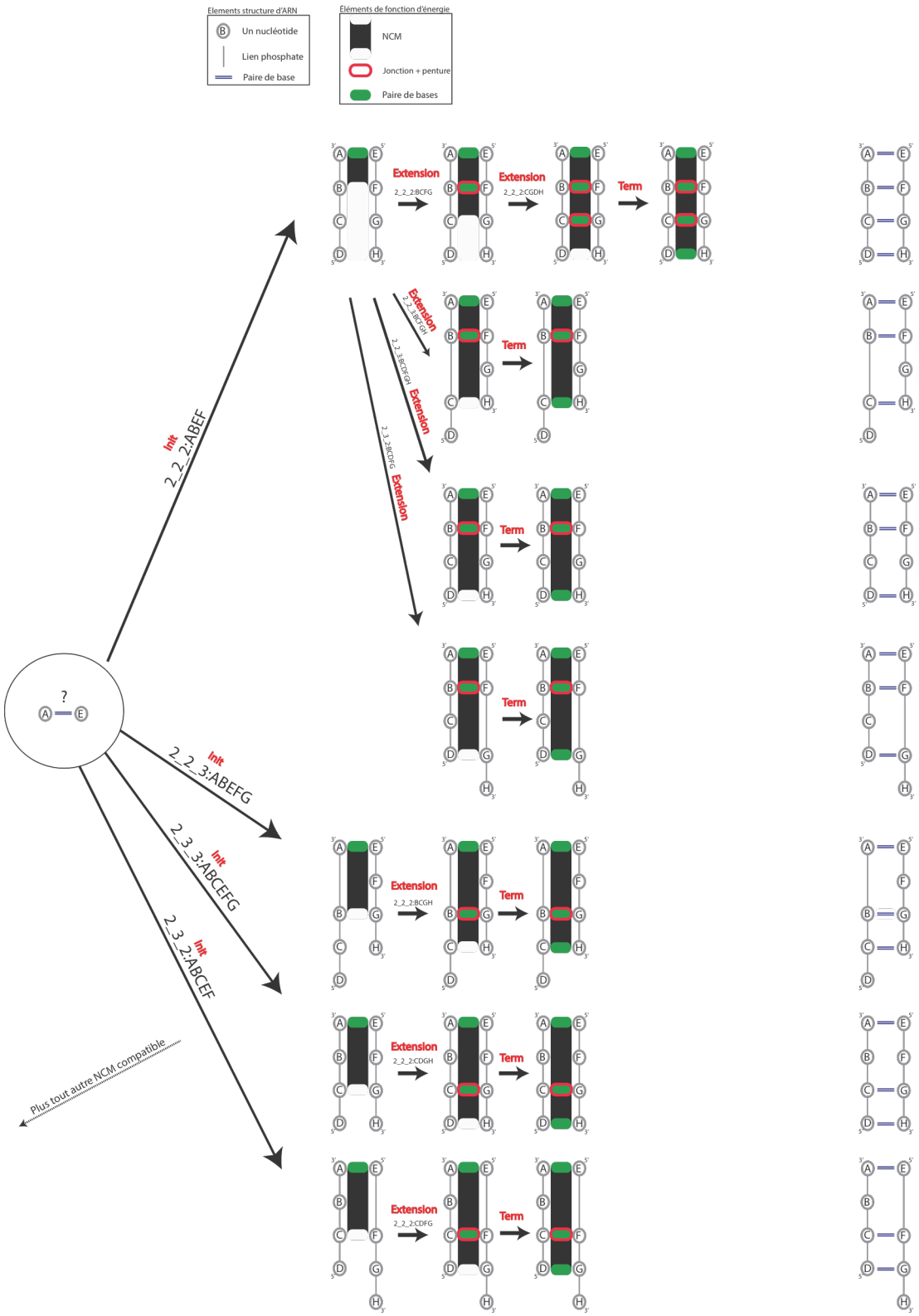

Figure 6: Étapes d'énumération de NCM.

1.5 La grammaire de MC-Flashfold

La grammaire suivante suffit à décrire les structures d'ARN composées de paires de bases et dépourvues de pseudo-nœuds. Le problème à résoudre est un problème de décoration donnant des mots sur Σ_{PP}^{n} sur entrée $\gamma \in \Gamma_{ARN}^{n}$ (rappelons que $\Sigma_{PP} = \{'.', '(', ')'\}$ et que $\Gamma_{ARN} = \{A, C, G, U\}$). Nous définissons ici la grammaire de décoration et la grammaire de sortie de la même manière en remplaçant les symboles terminaux de ces grammaires par les nouveaux symboles x, p, q ayant les équivalences suivantes dans Σ_{PP} : $x = '.'$, $p = '('$ et $q = ')'$. Dans Γ_{ARN}, chaque symbole peut prendre n'importe quelle valeur et les règles de cette grammaire sont en ce sens ambiguës, la préférence parmi les règles étant dictée par la fonction objectif et non par la grammaire. C'est ce que nous voulons dire lorsque nous spécifions les symboles terminaux comme suit $T = < x, p, q >$, S une structure, *pq* une paire de base et *x* une base non appariée :

$$S \rightarrow pSq \mid x \mid xS \mid Sx \mid SS$$

Cette grammaire cependant est structurellement très ambiguë et si un algorithme reconnaissant les éléments de ce langage était produit, il pourrait utiliser différents arbres d'analyse (*parse tree*) pour reconnaitre une même structure. Un générateur des éléments reconnus par le langage donnerait plusieurs copies de la même structure en sortie pendant son exploration. Par exemple, il existe plusieurs arbres pour la structure *x*S*xx*S*x*. Ce comportement peut devenir un problème pour MC-Flashfold si ces copies redondantes sont nombreuses et nous résolvons cela en produisant une nouvelle grammaire. Pour cela nous introduisons de nouveaux non-terminaux et nous séparons les règles dans un processus similaire à celui que nous avons fait dans l'exemple des parenthèses balancées (donné en annexe).

$$S \rightarrow M$$

$$M \rightarrow LM \mid R$$

$$L \rightarrow xL \mid pMq$$

$$R \rightarrow L \mid Rx \mid T$$

$$T \rightarrow xT \mid x$$

Nous pouvons énumérer toutes les structures dont nous avons besoin à partir de cette grammaire mais pour évaluer la contribution des NCM, nous avons besoin des règles additionnelles I, X and C dans la prochaine grammaire. Dans cette grammaire (toujours incomplète) nous commençons à tenir compte des NCM somme suit : L'état I correspond à l'initiation d'une double hélice (tige/*stem*), l'état X permet l'extension des NCM de type 2_2_2 et l'état C tient compte de tous les NCM à un brin. Ceci introduit de nouvelles redondances par exemple en permettant aux boucles terminales dont la longueur le permet d'être reconnues soit par les règles C ou par les règles T.

$$S \rightarrow M$$

$$M \rightarrow LM \mid R$$

$$L \rightarrow xL \mid I$$

$$I \rightarrow X$$

$$X \rightarrow pXq \mid C \mid M$$

$$C \rightarrow pxq \mid pxxq \mid pxxxq \mid pxxxxxq$$

$$R \rightarrow L \mid Rx \mid T$$

$$T \rightarrow xT \mid x$$

Ajoutons maintenant les autres NCM. Nous introduisons également un ensemble de tables *ouvertes*. Ces tables joueront le même rôle que leurs homologues *ordinaires* mais elles seront limitées en ce sens qu'il leur sera interdit de dupliquer les boucles (terminales ou internes) qui sont définies par des NCM. Ceci augmente le nombre de tables mais avec peu de conséquences sur la performance.

Embranchement	$M \rightarrow LM \mid R \mid T$
Boucle Gauche	$L \rightarrow xL \mid I$
Boucle Droite	$R \rightarrow L \mid Rx$

Boucle Terminale	$T \rightarrow xT \mid x$
Embranchement *Ouvert*	$M^o \rightarrow LM \mid R^o \mid T^o$
Boucle Droite *Ouvert*	$R^o \rightarrow xxxxLx \mid xxxLxx \mid xxLxxx \mid Rxxxx$
Boucle Terminale *Ouvert*	$T^o \rightarrow xxxxT$
Tige Initiation	$I \rightarrow X^k$
Tige Extension	$X^{2\_2\_2} \rightarrow pX^kq \mid prM^osq$
	$X^{2\_3\_2} \rightarrow pxX^kq \mid pxrM^osq$
	$X^{2\_2\_3} \rightarrow pX^kxq \mid prM^osxq$
	$X^{2\_3\_3} \rightarrow pxX^kxq \mid pxrM^osxq$
	$X^{2\_2\_4} \rightarrow pX^kxxq \mid prM^osxxq$
	$X^{2\_3\_4} \rightarrow pxX^kxxq \mid pxrM^osxxq$
	$X^{2\_4\_4} \rightarrow pxxX^kxxq \mid pxxrM^osxxq$
	$X^{2\_4\_2} \rightarrow pxxX^kq \mid pxxrM^osq$
	$X^{2\_4\_3} \rightarrow pxxX^kxq \mid pxxrM^osxq$
	$X^{2\_2\_5} \rightarrow pX^kxxxq \mid prM^osxxxq$
	$X^{2\_3\_5} \rightarrow pxX^kxxxq \mid pxrM^osxxxq$
	$X^{2\_5\_2} \rightarrow pxxxX^kq \mid pxxxrM^osq$
	$X^{2\_5\_3} \rightarrow pxxxX^kxq \mid pxxxrM^osxq$
Tige Terminaison	$X^{1\_3} \rightarrow pxq$
	$X^{1\_4} \rightarrow pxxq$
	$X^{1\_5} \rightarrow pxxxq$
	$X^{1\_6} \rightarrow pxxxxq$

où les indice k dans les expression X^k est une variable acceptant n'importe quel type de NCM.

Bien que cette grammaire soit complète elle est un peu verbeuse et nous introduisons un itérateur sur les règles combinées d'extension et de terminaison des tiges donnant une version plus compacte, dont le code est plus facile à maintenir. La règle T^o est éliminée par simplification. Cela constituera notre grammaire finale :

Embranchement	$M \rightarrow LM \mid R \mid T$
Boucle Gauche	$L \rightarrow xL \mid I$

Boucle Droite	$R \rightarrow L \mid Rx$
Boucle Terminale	$T \rightarrow xT \mid x$
Embranchement *Ouvert*	$M^o \rightarrow LM \mid R^o \mid xxxxT$
Boucle Droite *Ouvert*	$R^o \rightarrow xxxxLx \mid xxxLxx \mid xxLxxx \mid Rxxxx$
Tige Initiation	$I \rightarrow X^k$
Tige Extension et Terminaison	$X^k \rightarrow p \cdots X^k \cdots q \mid p \cdots r M^o s \cdots q \mid p \cdots q$

Nous pouvons décorer cette grammaire avec les itérateurs pour obtenir notre architecture de retour en arrière :

Embranchement	$M_{i,j} \rightarrow \forall_{i<s<j-1} L_{i,s} M_{s+1,j} \mid R_{i,j} \mid T_{i,j}$
Boucle Gauche	$L_{i,j} \rightarrow xL_{i+1,j} \mid I_{i,j}$
Boucle Droite	$R_{i,j} \rightarrow L_{i,j} \mid R_{i,j-1}x$
Boucle Terminale	$T_{i,j} \rightarrow xT_{i+1,j} \mid x$
Embranchement *Ouv.*	$M^o_{i,j} \rightarrow \forall_{i<s<j-1} L_{i,s} M_{s+1,j} \mid R^o_{i,j} \mid xxxxT_{i+4,j}$
Boucle Droite *Ouv.*	$R^o_{i,j} \rightarrow xxxxL_{i+4,j-1}x \mid xxxL_{i+3,j-2}xx \mid xxL_{i+2,j-3}xxx \mid R_{i,j-4}xxxx$
Tige Initiation	$I_{i,j} \rightarrow \forall_{k \in NCM} X^k_{i,j}$
Tige Ext. et Term.	$X^k_{i,j} \rightarrow \forall_{k' \in NCM} p \cdots X^{k' \in \{2\_x\_x\}} \cdots q \mid p \cdots r M^o_{i+2+\cdots,j-2-\cdots} s \cdots q \mid \forall_{k \in NCM} p \cdots X^{k' \in \{1\_x\}} \cdots q$

où l'expression $\{2\_x\_x\}$ tient pour l'ensemble des NCM à deux brins et où $\{1\_x\}$ celui des NCM à un brin.

Et en inversant les flèches, nous obtenons le squelette des récurrences de la programmation dynamique.

Embranchement	$M_{i,j} \leftarrow \forall_{i<s<j-1} L_{i,s} M_{s+1,j} \mid R_{i,j} \mid T_{i,j}$
Boucle Gauche	$L_{i,j} \leftarrow L_{i+1,j} \mid I_{i,j}$
Boucle Droite	$R_{i,j} \leftarrow L_{i,j} \mid R_{i,j-1}$
Boucle Terminale	$T_{i,j} \leftarrow T_{i+1,j}$
Embranchement *Ouv.*	$M^o_{i,j} \leftarrow \forall_{i<s<j-1} L_{i,s} M_{s+1,j} \mid R^o_{i,j} \mid T_{i+4,j}$
Boucle Droite *Ouv.*	$R^o_{i,j} \leftarrow L_{i+4,j-1} \mid L_{i+3,j-2} \mid L_{i+2,j-3} \mid R_{i,j-4}$
Tige Initiation	$I_{i,j} \leftarrow \forall_{k \in NCM} X^k_{i,j}$
Tige Ext. et Term.	$X^k_{i,j} \leftarrow \forall_{k' \in NCM} X'^{k \in \{2\_x\_x\}} \mid$

$$M^{o}_{i+2+\cdots,j-2-\cdots} \mid \forall_{k' \in NCM} X^{k' \in \{1\_x\}}$$

La table T n'est pas explicitement nécessaire et les tables X et I peuvent être fusionnées en une seule (F) donnant les équations de programmation dynamique suivantes où les fonctions de coût sont basées sur la fonction objectif donné à la section précédente.

Notons : (1) considérant le NCM inscrit sur i, j, son NCM suivant (partageant une paire) est localisé sur i+Δ,j-∇, (2) E_k : l'énergie du NCM k, (3) $J_{k \to k'}$: la jonction du NCM k vers le NCM k' (4) $H_{i+\Delta,j-\nabla|k,k'}$: le *hinge* de la paire de nucléotides formée des nucléotides i+Δ et j-∇ dans le contexte de la jointure des NCM k et k' et (5) $P_{i+\Delta,j-\nabla}$ est la valeur de la paire formée entre les nucléotides i+Δ et j-∇.

$$F_{i,j,k} = MIN \begin{cases} MIN_{k' \in 2_{NCM}}(F_{i+\Delta,j-\nabla,kk} + E_{k'} + J_{k \to k'} + H_{i+\Delta,j-\nabla|k,k'} + P_{i+\Delta,j-\nabla}) \\ MIN_{k' \in 2_{NCM}}(M^{o}_{i+\Delta,j-\nabla,k'} + E_{k'} + P_{i+\Delta,j-\nabla}) \\ MIN_{k' \in 1_{NCM}}(E_{k'}) \end{cases}$$

$$L_{i,j} = MIN \begin{cases} MIN_{k \in NCM}(F_{i,j,k} + P_{i,j}) \\ L_{i+1,j} \end{cases}$$

$$R_{i,j} = \begin{cases} R_{i,j-1} \\ L_{i,j} \\ 0\ (j - i > 4; longue\ boucle\ terminale) \end{cases}$$

$$R^{o}_{i,j} = MIN \begin{cases} MIN(R_{i+4,j}, R_{i+3,j-2}, R_{i+2,j-3}, R_{i,j-4}) \\ 0\ (j - i"4;\ longue\ boucle\ terminale) \end{cases}$$

$$M_{i,j} = MIN \begin{cases} MIN_{s,i<s<j}(L_{i,s} + M_{s+1,j}) \\ R_{i,j} \end{cases}$$

$$M^{o}_{i,j} = MIN \begin{cases} MIN_{s,i<s<j}(L_{i,s} + M_{s+1,j}) \\ R^{o}_{i,j} \end{cases}$$

Nous obtenons les MPR directement à partir des annotations donnant la description des états de retour en arrière.

État	Code	ID	MPR	Pile	Pile2	Suivant	itérateurs
Start Embr.	MB	1	$L_{i,s} + M_{s+1,j}$	4	1	2	$i < s < j\text{-}1$
Start Droite	MB	2	$R_{i,j}$	6		3	
Start Term.	MB	3	0	12			
Boucle Gauche	LEFT	4	$g() + L_{s,j}$	4		5	$i < s < j$
Boucle Gauche Init.	LEFT	5	$g() + I_{i,j}$	13			
Boucle Droite	RIGHT	6	$g() + R_{i,s}$	6		7	$i < s < j$
Droite Boucle gauche	RIGHT	7	$g() + L_{i,j}$	4			
Term. Boucle Droite	TLRIGHT	8	$g()$				
Branchement ouvert	MB	9	$g() + L_{i,s} + M_{s+1,j}$	4	9	10	$i \leq s < j\text{-}1$
Branchement ouvert Droite	MB	10	$g() + R^{o}_{i,j}$	11			
Droite Boucle ouvert	MB	11	$g() + L_{p,q}$	4		12	$(p,q) \in \begin{cases} (i+4,j), \\ (i+3,j-2), \\ (i+2,j—3) \end{cases}$
Droite Boucle ouvert Droite	MB	12	$g() + R_{i,j\text{-}4}$	6			
Tige Initiation	K	13	$g() + P_{i,j} + F_{i,j,k}$	14			k ∈ { 2_x_x NCM }
Tige Extension	KK	14	$g() + E_k + J_{k \to k'} + H_{i+\Delta,j-\nabla \mid k,k'} + P_{i+\Delta,j-\nabla} + F_{i+\Delta,j-\nabla,k'}$	14		15	k' ∈ { 2_x_x NCM }
Tige Terminaison	KK	15	$g() + E_{k'}$			16	
Tige Terminaison ouvert	KK	16	$g() + E_{k'} + P_{i+\Delta,j-\nabla} + M^{o}_{i+\Delta,j-\nabla,k'}$	9			k' ∈ { 2_x_x NCM }

Un nombre plus restreint d'états explicites sont utilisés dans le code (colonne code) mais de manière équivalente. Un *parse* (arbre d'analyse) donnant la structure d'une séquence est donné à la Figure 7.

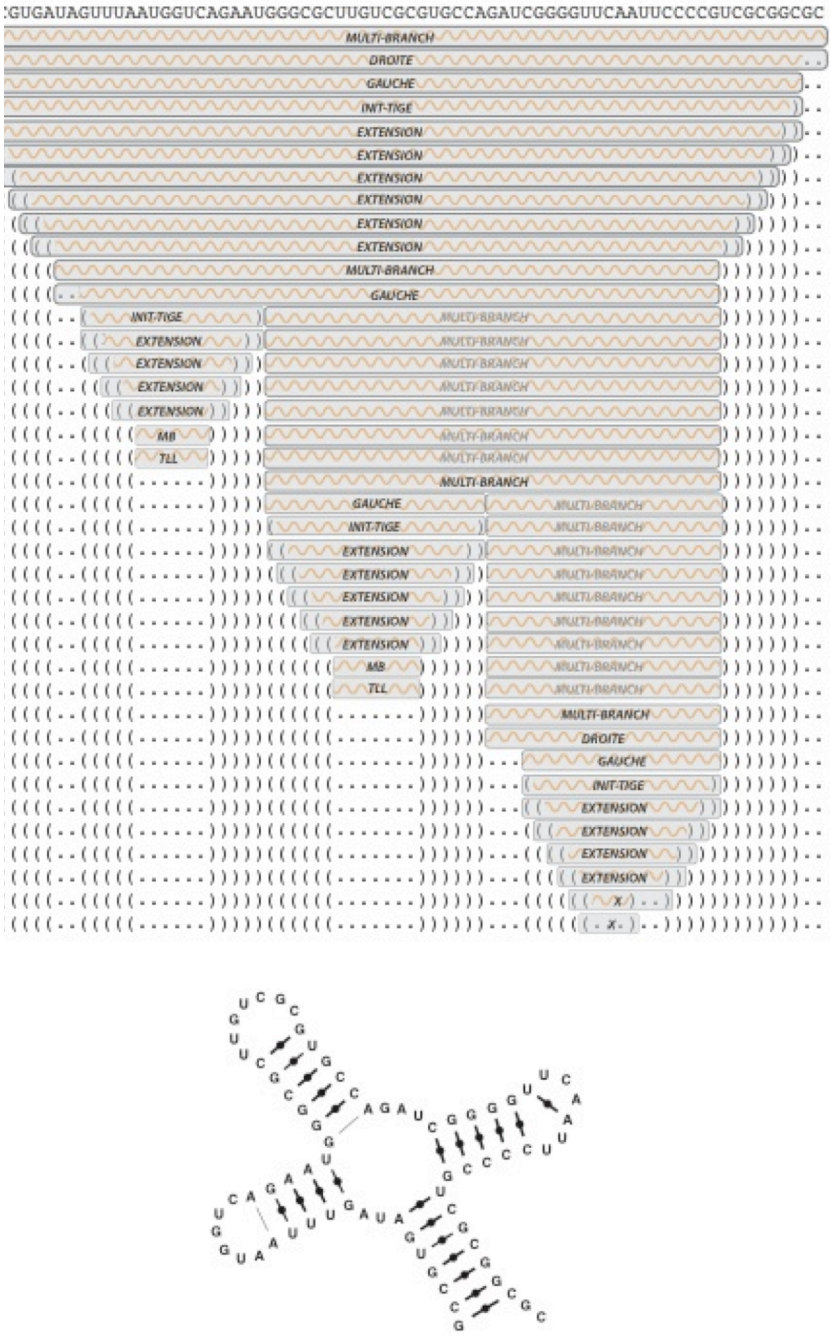

Figure 7: Assignation d'une structure à une séquence (un parse).

1.6 Processus complémentaires

1.6.1 Les masques de MC-Flashfold

Un masque est un objet percé qui limite l'action d'un agent sur un matériel placé en dessous et partiellement protégé au cours d'un traitement manufacturier tel que la gravure ou la photographie. La prédiction de structure d'ARN utilise des masques. Ceux-ci prennent la forme de solutions partiellement spécifiées donnés accessoirement en entrée et que les solutions produites doivent respecter. Les masques sont utilisés pour fournir à l'algorithme des connaissances additionnelles au sujet des ARN et ceci peut améliorer grandement l'exactitude des prédictions. Ils peuvent être utilisés pour refléter des résultats d'analyses physico-chimiques ou comme une aide en modelage par homologie.

Tel que décrit dans la section *Intégrer des sous-solutions prédéfinies*, les masques peuvent être appliqués pendant les calculs (masques complets, *full masks*) ou utilisés pour filtrer les sorties (masques de sortie, *output masks*). Notons au passage que l'application de masques directement pendant les calculs correspond à modifier la grammaire en la combinant à une grammaire explicitant les contraintes voulues, autrement dit à faire le produit de deux grammaires (Giegerich, Meyer et al. 2004). Ce processus est justement implémenté dans le programme RapidShapes (Janssen and Giegerich 2010) donnant la probabilité de *shapes* données pour une séquence d'ARN. De plus nous distinguons les masques qui s'appliquent aux paires de nucléotides (masques balancés, *balanced masks*) de ceux qui ne s'appliquent qu'aux nucléotides pris individuellement (masques non-balancés, *unbalanced masks*). L'usager peux spécifier le type de paire de base (canonique, non-canonique ou non-pairé) ainsi que l'orientation du pairage (vers une position en 5' ou en 3') pour tout nucléotide ou paire de nucléotides dans la séquence.

La filtration des sorties utilisant des masques balancés est une instance d'algorithmes reconnaissant le langage des parenthèses balancées, nécessitant un automate à pile et ne peux donc pas être remplacé par des appels à des outils basés sur des automates

fini tels que grep et sed. C'est pourquoi MC-Flashfold fournit les outils nécessaires à ce traitement.

Un désavantage marqué de la filtration des sorties est que les solutions doivent tout de même être énumérées avant qu'elles puissent être vues par le filtre. Le nombre de sorties étant possiblement grand, ce système peut-être parfois insuffisant. Notons MFE (Minimum Free Energy) l'énergie d'une des structures les plus stables (valeur que nous avons notée $\mathcal{O}(\tau^o)$ dans la description générale de l'approche) en absence de masque. Si une séquence accepte sa première solution respectant le masque de sortie a une énergie Φ_m mais que le seuil est fixé à une valeur MFE – $\Delta < \Phi_m$ alors aucune solution ne sera identifiée. Explorer des valeurs de seuil plus grandes s'avère une mauvaise stratégie en général puisque le nombre de sorties croit rapidement en fonction du seuil, ralentissant l'exécution du logiciel. Les masques complets éliminent ce problème en appliquant les contraintes de paires de bases pendant les deux phases (programmation dynamique et retour en arrière) du logiciel. Les calculs n'explorent alors que les solutions qui respectent le masque. Dans ces conditions, la MFE est celle d'une structures les plus stables respectant le masque. Cette approche est facilement intégrée au code même si celui-ci devient quelque peu verbeux.

Ces approches affectent différentes sections du code et tous les types de masques peuvent être appliqués lors d'une même invocation. En plus des masques complets qui limitent les possibilités pendant la programmation dynamique et le retour en arrière, nous avons implémenté également les masques de sorties comme un reconnaisseur qui ajoute des étiquettes aux sorties qui respectent les masques correspondants et non pas comme un filtre qui enlèverait les sorties non reconnues. De cette manière, plusieurs masques de sorties peuvent être spécifiés simultanément. Cependant seuls un masque complet balancé et un masque complet non-balancé peuvent être spécifiés en même temps (l'utilisateur peut les construire facilement au besoin).

Il est important que l'usager comprenne que certains masques décrivent des solutions qu'aucun arrangement de NCM ne peut satisfaire et qu'alors aucune sortie ne sera générée.

1.6.2 Fixer un estimé du nombre de solutions

Le nombre de structures générées κ grandit rapidement en fonction du seuil Δ de Waterman-Byers donné par l'usager et cette croissance rapide peut devenir un problème pour l'utilisabilité du logiciel. Parce que le taux de croissance dépend de la séquence, parce qu'il est plutôt difficile d'estimer avec précision cette croissance et parce que cette croissance est très abrupte, il est facile de surestimer le seuil applicable avec des conséquences drastiques sur la durée des calculs. Bien sûr l'usager peut s'intéresser à la variation du nombre de sorties en fonction du seuil mais nous nous intéressons ici au seuil à utiliser pour obtenir un nombre donné de solutions. Dans une procédure de balayage où des invocations répétées sont effectuées avec de nombreuses séquences, la variation de vitesse d'exécution est problématique. Deux stratégies distinctes sont utilisées par MC-Flashfold pour adapter la valeur de Δ utilisée pour obtenir un nombre donné de solutions κ^{Δ}.

La première stratégie consiste à énumérer non pas les solutions mais seulement leurs énergies et à ajouter ces énergies à un tas (*heap*) de taille κ^{Δ}. Lorsque le tas est plein, la pire énergie est prise comme Δ. À partir de ce moment, toute nouvelle solution dont l'énergie est meilleure est ajoutée au tas et la nouvelle pire valeur du tas est prise comme nouveau seuil. Cette approche coupe drastiquement l'espace de fouille. À la fin de cette première passe, la valeur de Δ^{κ} est connue et utilisée pour calculer les structures. Le remplacement de chaque nouvelle énergie dans le tas est $O(\log \kappa^{\Delta})$ et si toutes les énergies étaient ordonnées malicieusement, la complexité du retour en arrière serait augmenté d'autant. En général cependant le gain en vitesse relatif aux seuils devinés est impressionnant même lorsque les valeurs initiales de Δ^{κ} sont surestimées de quelques ordres de grandeur. Nous verrons des résultats expérimentaux montrant ceci.

La deuxième stratégie (décrite à la section *Une approche top-down pour trouver* $\mathcal{S}^{\kappa,OPT}(\gamma)$ *et* $\mathcal{S}^{\kappa*,OPT}(\gamma)$) consiste à estimer Δ^{κ} itérativement à partir d'une petite valeur. À chaque itération, les énergies des solutions sont utilisées pour estimer $d\kappa/d\Delta$ le taux de croissance de κ en fonction de Δ. Un $\Delta\Delta$ (incrément de seuil) est estimé à partir de la queue de la distribution des énergies et ajouté au seuil (Δ) pour l'itération suivante. Lorsque

$\Delta \geq \Delta^{\kappa}$ la première stratégie (celle utilisant un tas) est utilisée pour la ronde suivante. La valeur initiale de Δ est soit fournie par l'usager ou estimée de manière conservatrice à partir de la longueur de la séquence.

La détermination automatique du seuil est l'approche recommandée pour invoquer MC-Flashfold lorsque un nombre donné de solutions est requis.

1.7 Conclusion pour le chapitre 1

Wuchty a décrit (Wuchty, Fontana et al. 1999) une méthode pour énumérer $2D^{c}(\gamma)$ les structures d'ARN (à paires de bases canoniques) sous-optimales par retour en arrière sur un espace solution en utilisant une fonction guide basée sur les tables de programmation dynamique mais sans expliquer comment obtenir la synchronisation des règles entre le retour en arrière et celles de la programmation dynamique. Ceci rend difficile d'apporter des changements à la grammaire sous-jacente et au modèle d'énergie de manière nécessaire à adapter le modèle d'énergie de MC-Fold. D'autre part, Giegerich (Sauthoff, Mohl et al. 2013) décrit un système qui fait cette synchronisation pour un langage donné de manière automatique mais qui s'avère difficile à rendre efficace pour le modèle des NCM.

Nous avons tenté de combiner les forces de ces deux approches en donnant une description systématique des transformations à appliquer aux règles de grammaire nécessaires pour écrire la routine de programmation dynamique ainsi que les règles de retour en arrière dans un style de programmation impératif. Ceci rend flexible la grammaire d'entrée et le modèle de coût et permet l'implantation directe du retour en arrière, permettant une implantation performante. Nous avons décrit les méthodes nécessaires en détail et nous avons fourni un nombre suffisant d'exemples pour rendre le processus clair et facilement applicable à de nouveaux problèmes.

Notre but était de mettre au point une version améliorée de MC-Fold qui calcule plus rapidement, utilise moins de mémoire vive, qui accepte des séquences plus longues en entrée et qui génère rapidement des structures sous-optimales. Nous avons donc revu ici le modèle d'énergie des NCM et défini une grammaire correspondante utilisable en

programmation dynamique et dans la phase de retour en arrière. Nous avons implanté MC-Flashfold, le logiciel correspondant. Nous avons analysé et mesuré la croissance de l'usage des ressources en fonction de diverses longueurs des séquences d'entrée n et nombre de sous-optimaux κ et seuils d'énergie Δ sur des entrées pseudo-aléatoires de distributions uniformes en nucléotides (en annexe). Dans ces conditions, MC-Flashfold est beaucoup plus rapide que MC-Fold. Le sujet de bancs d'essais comparatifs exacts est complexe et nous n'avons pas abordé ce sujet ici. Il serait donc difficile de comparer précisément les différences entre les deux outils (MC-Flashfold et RNAsubopt) car leurs modèles d'énergie sont différents. Le modèle des NCM accepte toutes les paires possibles alors seules les paires canoniques sont acceptées par RNAsubopt. Ceci a des conséquences marquées sur la taille brute de l'espace solution à explorer (sans toutefois changer la classe de complexité de celui-ci) et donc sur le temps de calcul nécessaire pour le retour en arrière. Nous avons tout de même fait quelques comparaisons simples entre MC-Flashfold et RNAsubopt

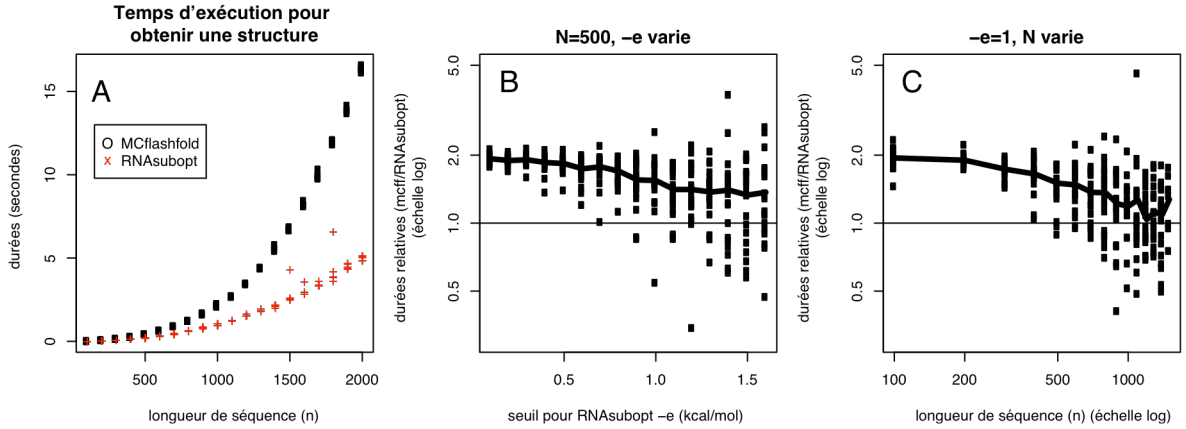

Figure 8: Temps d'exécution de MC-Flashfold et de RNAsubopt pour des mêmes séquences. mcff : MC-Flashfold

(Wuchty, Fontana et al. 1999)(un champion de rapidité de prédictions sous-optimales) avec des conclusions favorables.

La panneau A de la Figure 8 montre le temps nécessaire pour calculer la structure d'énergie MFE. (Les points apparemment aberrants sont ceux de séquences dont plusieurs structures sont d'énergie MFE.) Nous avons également comparé MC-Flashfold et RNAsubopt en ce qui concerne la vitesse de calcul incluant l'énumération d'un même nombre de structures sous-optimales dans les panneaux B et C. Dans un premier temps, RNAsubopt est utilisé avec un seuil fixe et un seuil correspondant pour le MC-Flashfold (donnant le même nombre de structures sous-optimales) est calculé puis utilisé pour exécuter le calcul avec MC-Flashfold sur les mêmes séquences. Le ratio du temps prit par MC-Flashfold versus celui prit par RNAsubopt est exprimé en ordonné. Le seuil (Δ, noté `-e` le paramètre de ligne de commande dans RNAsubopt en kcal) varie dans le panneau B alors que la longueur de l'entrée (n) varie dans le panneau C. Dans les deux cas, la tendance est que pour un grand nombre de structures sous-optimales ou pour des séquences suffisamment longues le ratio de la vitesse d'exécution des deux outils converge. Ainsi, lorsque le retour en arrière domine les calculs, notre implantation est au moins aussi efficace que celle implantée dans RNAsubopt et le choix d'un outil ne devrait pas se faire sur la base des capacités de calcul disponibles mais bien sur le mérite du modèle d'énergie. Nos résultats sont donnés pour une stratégie de compilation donnée et pour des séquences d'entrées aléatoires et uniformément distribuées. Il n'est donc pas impossible que ces valeurs s'avèrent différentes dans d'autres situations mais nous ne nous attendons qu'à de variations mineures.

2 Comparaison de structures

Ce chapitre est divisé en quatre parties :

1. Introduction au chapitre
2. Description des approches algorithmiques
3. L'article 'Structural dynamics controls the microRNA maturation pathway'
4. Conclusion au chapitre

2.1 Introduction au chapitre 2

Un SNP a été identifié dans le gène de miR-125a (un microARN) qui bloque sa maturation causant des effets biologiques indésirables. Nous comparons ici la maturation de plusieurs versions du gène grâce à une mesure de leurs ensembles de structures $2D$ obtenus avec MC-Flashfold en configuration S^{Δ} ou S^{κ} sous l'hypothèse que ce blocage est dû à des changements dans les structures dynamiques. Pour comparer efficacement ces ensembles nous obtenons d'abord des signatures simples extraites de ceux-ci. Si la méthode est suffisamment rapide et qu'elle donne une mesure de similarité qui soit biologiquement sensée, alors nous pourrons l'utiliser pour identifier des séquences partageant des fonctions biochimiques dans des grands ensembles de gènes. Nous sommes donc plus intéressé par la capture de signal d'importance biologique et par la rapidité d'exécution que par l'exactitude théorique de la comparaison.

Supposons qu'on veuille comparer entre elles les séquences de l'ensemble $\mathbb{G} = \{s | s \in \gamma_{ARN}^{n}\}$ en utilisant leurs ensembles $S^{\Delta}(s)$ (où $\Delta = \Delta^{\kappa}$ est un seuil donnant κ structures).

Comparaison en $O(\kappa + n)$ et hypergraphes

Idéalement avec une approche naïve, nous comparerions entre elles chaque structure de chaque ensemble $S^{\Delta}(s_i), s_i \in \mathbb{G}$. Pour cela, supposons qu'un algorithme efficace compare deux structures $2D$ sur des séquences longues de n en temps $O(n)$ (il est difficile de faire mieux). Alors, deux ensembles de taille κ sont comparés en environ $O\left(\frac{\kappa^2 n}{2}\right)$ et la matrice de similarité est obtenue en $O(\kappa^2 n|\mathbb{G}|^2)$. Il est probable qu'on puisse faire mieux en représentant les κ structures sous la forme de forêt compactée (des hypergraphes). Dans ces structures, au moins un nœud est unique à chacune des κ solutions sous-optimales et il y a au moins une solution de taille $O(n)$; ces structures sont donc de taille $O(n + \kappa)$. En supposant qu'on puisse les comparer en temps linéaire, la matrice de similarité est obtenue $\in O\big((n + \kappa)|\mathbb{G}|^2\big)$.

Comparaisons en $O(n^2)$ et table de partition

Une autre approche consiste à calculer une table Π_{s_i} de taille $n^2/2$ donnant la fréquence de la paire $(i,j), 1 \leq i < j \leq n$ dans les $S^{\Delta}(s_i)$. Les Π_{s_i} sont obtenues en $O(\kappa n)$ et la similarité est alors obtenue en $O(n^2)$ et prends à la base une forme simple pour deux séquences $u, v \in \mathbb{G} : \sum_{(i,j)\in(1..n,1..n)}(\Pi_u + \Pi_v - |\Pi_u - \Pi_v|[i,j])$. La matrice de similarité est $O(n^2|\mathbb{G}|^2)$. En apparence, si $\kappa \gg n$, cette approche semble supérieure mais elle souffre de deux problèmes en pratique. D'une part elle ne décrit que les configurations des nucléotides qui sont pairés et ne tient pas compte des nucléotides non appariés. D'autre part, si $|u| \neq |v|$ la simple somme est innaplicable et il faut invoquer une autre méthode de comparaison qui sera cette fois beaucoup plus coûteuse. Par exemple (Hofacker, Bernhart et al. 2004) aligne les tables de partitions de deux séquences de longueurs différentes en se guidant sur l'alignement de leurs séquences.

Rappelons nous que nos solutions sont de la forme $\langle \mathcal{D}(\tau), W(\tau) \rangle \in S^{\Delta}(s)$, où τ est une structure de NCM, $\mathcal{D}(\tau) \in \Sigma_{PP}^{n}$ décrit ses paires de bases sous la forme d'un mot sur le langage parenthèse-point et $W(\tau)$ est la pseudo-énergie (en kcal/mol) de cette solution.

La table de partition des paires de bases est une table P de taille $n^2/2$ donnant les probabilités $p(i,j)$ des paires $s[i] \bullet s[j]$ pour l'ensemble $S^{\Delta}(s)$ avec $\Delta = \infty$ correspondant à $P[i,j] = \sum_{\langle d,w\rangle \in S^{\Delta}(s)} \sum_{A \bullet B \in d} \frac{e^{-\frac{w}{RT}}}{Z_s}$, avec $Z_s = \sum_{\langle d,w\rangle \in S^{\Delta}(s)} e^{-\frac{w}{RT}}$ la fonction de partition de la séquence s. (Un résultat étonnant attribué à McCaskill utilisant l'algorithme inside-outside (McCaskill 1990; Bonhoeffer, McCaskill et al. 1993) donne $P(s)$ essentiellement en $O(n^3)$ en modifiant soigneusement les récurrences de $\mathcal{A}_p$ (assumant une grammaire $\mathcal{G}_p$ dépourvue d'ambiguité) pour obtenir la somme des énergies plutôt que le min de celles-ci, tout en normalisant cette valeur sur le nombre de sous-structures (le nombre des τ en i,j).) Ainsi on pourrait obtenir P_s^{Δ}, la table de partition tronquée pour un seuil Δ à partir de Π_s ou encore ré-écrire MC-Flashfold pour obtenir efficacement directement P_s.

Ceci étant dit, (Sabarinathan, Tafer et al. 2013b) discutent extensivement les méthodes de comparaisons de P_s et donnent une méthode qui inclus une normalisation extensive sur l'espace des séquences naturelles dans le but d'évaluer la probabilité qu'un SNP soit biologiquement significatif.

Comparaison en $O(n)$

L'extraction de signatures à partir de P_s n'est pas une nouvelle idée. (Bonhoeffer, McCaskill et al. 1993) utilisait déjà cette idée pour améliorer l'alignement de séquences sur la base de leurs structures. (Dalli, Wilm et al. 2006; Kruspe and Stadler 2007) produisent des alignement de séquences multiple d'ARN sur la base de similarité des P_s et (Halvorsen, Martin et al. 2010) l'utilise pour établir la perturbation causée à P_s par une variation de séquence. (Sabarinathan, Tafer et al. 2013b) discutent extensivement ces utilisations. La signature que nous avons retenue est une version augmentée des $\xi^{<>}$ qu'utilisent (Halvorsen, Martin et al. 2010) et laquelle est une paire de vecteurs donnant d'une part $\xi^{<}[j] = \sum_{i<j} P[i,j]$ et d'autre part $\xi^{>}[i] = \sum_{i<j} P[i,j]$. Des signatures $\xi_{s_1}^{<>}$et $\xi_{s_2}^{<>}$ pour des séquences s_1 et s_2 peuvent être comparées rapidement avec diverses normes et corrélations généralement en $O(n)$ si $|s_1| = |s_2|$ mais demandent des schémas de comparaison plus compliqués lorsque $|s_1| \neq |s_2|$ qui utilisent par exemple la programmation dynamique en $O(n^2)$ ou $O(2n)$. Des signatures peuvent être compilées à partir de P_s, de Π_s ou de $S^{\Delta}(s)$ en $O(n^2)$, $O(n^2)$ et $O(n\kappa)$ respectivement.

La comparaison des $s \in \mathbb{G}$ pour des séquences de longueurs égales est donc, à partir des P_s ou des Π_s, $\in O(|\mathbb{G}|n^2 + |\mathbb{G}|^2 n)$ et, à partir des $S^{\Delta}(s)$, $\in O(|\mathbb{G}|\kappa n + |\mathbb{G}|^2 n)$.

Notre intérêt se porte notamment sur l'information des nucléotides non-pairés dans les $S^{\Delta}(s)$. Les nucléotides non-pairés en bouts de tiges (les boucles terminales) et dans les renflements (*bulges*) sont déterminants pour l'interaction avec de molécules externes (voir par exemple (Dethoff, Petzold et al. 2012)). Cette information n'est pas disponible dans P_s ni dans Π_s et il nous faut les énumérer en analysant les $\mathcal{d}$ dans $\langle \mathcal{d}, w \rangle \in S^{\Delta}(s)$.

Éventuellement on pourrait envisager d'associer nos signatures avec les comparaisons de P_s. Car si nos signatures captent les fréquences auxquelles les nucléotides sont non-pairés, les signatures ne captent pas l'identité du partenaire de pairage d'un nucléotide pairé, seulement la fréquence et la direction de cet appariement.

2.1.1 Extraction de signature

La signature cherche à capter le comportement des nucléotides dans les structures sous-optimales. Pour y arriver, nous compilons des statistiques sur chaque nucléotide. Chaque nucléotide dans chaque structure est soit apparié ou non. S'il est apparié, son partenaire peut-être en amont (du côté 5') ou en aval (du côté 3'). Les nucléotides non appariés sont dans des segments entre les tiges, aux extrémités de la molécule, dans un virage au bout d'une tige ou dans une tige formant un renflement ou une boucle intérieure et alors soit du côté 5' du virage ou soit du côté 3' de celui-ci. Nous colligeons donc des informations sur ces 7 possibilités.

La signature est composée des occurrences parmi les structures sous-optimales de chaque situation pour chaque base. Elle est donc aussi longue que la séquence et à chaque base correspond un 7-tuple. Nous comptabilisons les 5 situations où la base est non-appariée en comptant le nombre d'occurrences de la situation parmi les structures dans l'ensemble. Les cas de bases appariées nécessitent une considération supplémentaire car pour chaque base dans chaque structure il existe un certain nombre de possibilités d'appariement. Étant donnée une séquence de longueur *n,* pour une structure donnée, la base *i* peut s'apparier avec les *n*-*i*-*B* (normalement, *B* =1, le nombre minimal de bases dans une boucle terminale) bases en 3' et avec *i*-*B*-1 bases en 5'. Les occurrences d'appariement en 3' et en 5' sont donc des vecteurs de valeurs et non des scalaires comme c'est le cas pour les situations non-appariées.

Ceci alourdit la comparaison de séquences de longueurs différentes car pour chaque base, il devient nécessaire de comparer des vecteurs de longueurs différentes. Un certain nombre de stratégies sont envisageables. Nous avons choisi de prendre une valeur

représentative pour chacun de ces vecteurs. Cela donne une représentation compacte et uniforme de chaque base. (Notons au passage qu'il existe une certaine redondance entre ces valeurs et les valeurs capturées aux autres positions dans la signature puisque une paire de base fréquente $s[i] \bullet s[j]$ est captée comme 3' en i mais également comme 5' en j.) L'importance du choix de partenaires d'appariement peut-être capté de différentes manières. Dans le cas le plus simple nous prenons la somme des occurrences de pairage avec tous les partenaires de manière équivalente, que l'évènement soit fréquent ou non. Mais on peut être sensible à l'argument fréquentiste donnant le partenaire le plus fréquent comme étant le plus important (maximum). En continuant avec de telles réflexions, on peut penser que la moyenne, l'écart-type ou une autre statistique soit plus signifiante au sujet de l'affinité de pairage d'une base. Nous avons effectué des tests sur ce point et les résultats sont donnés un peu plus loin.

Après avoir extrait une valeur d'occurrence de chaque situation pour une position donnée, le 7-tuple est normalisé. Cette normalisation est nécessaire afin qu'il soit possible de comparer les signatures de séquences de longueurs différentes. La normalisation rend incompatibles les signatures comptabilisées en utilisant différentes statistiques.

Ainsi la signature est une matrice $7 \times n$ de scalaires dont chaque colonne somme à 1.

La signature d'une séquence *s* est notée σ^s. Le 7-tuple du $i^{ème}$ nucléotide est σ_i^s. Les valeurs individuelles du 7-tuple sont accédées en indiçant la position correspondante dans la signature. La $k^{ième}$ valeur est $\sigma_{i,k}^s$.

2.1.1.1 Calcul d'une signature

Le calcul d'une signature procède en trois étapes :

1. Collecte des configurations,
2. Compilation d'une statistique de pairage,
3. Normalisation.

2.1.1.1.1 *Collecte des configurations*

Cette étape consiste à identifier et compter, pour chaque nucléotide et chaque structure sous-optimale le type de configuration. Le nucléotide peut être dans l'une des 5 configurations non appariées ou dans une configuration appariée.

Il est facile de déterminer quelle situation s'applique à chaque nucléotide dans une structure 2D. L'algorithme ne nécessite qu'une pile. Les nucléotides formant une paire avec un partenaire 3' causent un empilement. Les nucléotides formant une paire avec un partenaire 5' causent un dépilement. En conservant un minimum d'information contextuelle il est facile de déterminer la situation des nucléotides non-appariés. Cette étape de l'extraction de la signature, le *parsing*, d'une séquence longue de n et couvrant κ structures sous-optimales est $O(n\kappa)$ en temps et $O(n^2)$ en espace. Le terme quadratique correspond à une matrice carrée où les occurrences de pairage sont comptabilisées. Au cours du *parsing* les paires $s[i] \bullet s[j]$ sont décomptées dans une matrice $P^{n\times n}$ et les nucléotides non-appariés sont comptés directement dans σ.

Après cette étape, σ donne les occurrences des nucléotides non-appariés dans leurs diverses configurations et P donne les occurrences de pairages de $s[i] \bullet s[j]$.

2.1.1.1.2 Compilation d'une statistique de pairage

Nous voulons un scalaire r_i reflétant les occurrences de pairage à rebours du nucléotide i ($s[i]\bullet s[k]$, 1≤k<i) et un autre f_i pour les pairages par en avant (i●k, i<k≤n) : $r_i = \Psi(P_{i,1..i-1})$ et $f_i = \Psi(P_{i,i+1..n})$. Nous avons testé $\Psi \in \{\Sigma,\ MAX,\ MEAN,\ SD,\ Q1,\ MEDIAN,\ AAD1,\ AAD2\}$ (Σ la somme, MEAN la moyenne arithmétique, SD la déviation standard à la moyenne $\left(\sqrt{VAR}\right)$, Q1 le premier quartile, AAD1 la moyene de la déviation absolue à la médiane $\left(\frac{\sum|x_i - MEDIANE(x)|}{n}\right)$, AAD2 la moyene de la déviation absolue à la moyenne $\left(\frac{\sum|x_i - \bar{x}|}{n}\right)$ avec des résultats très similaires sauf pour MEDIAN qui s'est avéré plus volatile. Ce résultat est présenté après l'article car il utilise le contexte expérimental qui y est présenté (Figure 14).

r_i et f_i sont utilisés pour compléter les entrées de σ.

2.1.1.1.3 Normalisation

La normalisation est simplement :

$$\forall_{i=1}^{n}\ \forall_{k=1}^{7}\ \overline{\sigma_{i,k}} = \frac{\sigma_{i,k}}{\sum_{x=1}^{7}\sigma_{i,x}}$$

2.1.2 Comparaison de signatures de même longueur

La situation typique que nous considérons ici consiste en une séquence principale dont certaines bases sont changées pour donner un variant. On recherche alors le niveau de similarité qu'elles partagent ou le niveau de distance qui les sépare.

La similarité entre deux signatures σ^1 et σ^2 est

$$S^{MIN} = \sum_{i=1}^{n}\left\{\sum_{k=1}^{7} MIN(\sigma_{i,k}^{1}, \sigma_{i,k}^{2})\right\}$$

Plus les valeurs comprises dans les 7-tuples de la position *i* sont similaires et plus S^{MIN} tends vers *n*. La transformation de similarité en distance pour *E* signatures est simplement $D^{MIN} = MAX_{1\leq i<j\leq n}\, S^{MIN}(E_i, E_j) - S^{MIN}$

On peut également utiliser d'autres opérateurs. Afin de vérifier l'éventuel impact de différentes formulations de la mesure, nous avons instancié le squelette suivant de différentes manières :

$$\mathfrak{D}(\sigma^1, \sigma^2) = \underset{i=1}{\overset{n}{\Sigma/\prod}}\left\{\sqrt[p]{\sum_{k=1}^{7}\left(\Theta(\sigma_{i,k}^{1}, \sigma_{i,k}^{2})\right)^{p}}\right\}$$

Cette forme nous permet de combiner trois éléments de l'équation donnant tantôt des mesures de similarité et tantôt des mesures de distances:

Opérateur de métrique Θ minimum, maximum, différence

Puissance de la norme	p	¼, ½, 1 (Manhattan), 2 (Euclidienne), 4
Combinaison	$\Sigma/\prod$	Σ (somme), $\prod$ (produit)

2.1.3 Comparaison de signatures de longueurs différentes

Lorsque les signatures sont de longueurs différentes ou lorsque les positions ne correspondent pas, l'approche mentionnée ci-haut n'est pas applicable. Nous utilisons alors la programmation dynamique d'une manière analogue à l'alignement semi-global de séquences avec trous à cout affine (avec $\mathfrak{T}, \mathfrak{S} < 0$, les coûts d'initiation et d'élongation de trous). Nous montrons ici une implantation simple en $O(n^2m^2)$ pour deux séquences longues de m et n respectivement (avec Θ le MIN des 7-tuples). Cet algorithme pourrait être accéléré au besoin (voir par exemple l'accélération de Gotoh donnée en annexe pour l'alignement de séquences).

$$S_{i,j}^{pd} = MAX \begin{cases} \Theta\left(\sigma_i^1, \sigma_j^2\right) + \begin{cases} 0, si\ i < 2\ ou\ j < 2 \\ S_{i-1,j-1}, sinon \end{cases} \\ MIN_{1 \le k < i} S_{k,j} + \mathfrak{T} + (i-k)\mathfrak{S}, si\ i > 1 \\ MIN_{1 \le k < j} S_{i,k} + \mathfrak{T} + (j-k)\mathfrak{S}, si\ j > 1 \end{cases}$$

L'entrée de valeur maximale sur la dernière colonne ou la dernière rangée de la table $\left(S_{m,\cdot}^{pd}, S_{\cdot,n}^{pd}\right)$ donne le meilleur accord entre les signatures. S^{pd} peut-être pris comme une métrique dans l'espace des signatures puisqu'elle respecte nécessairement l'inégalité du triangle. (Ceci est dit sous la réserve que la signature représente effectivement l'ensemble, mais nous savons que la signature est une approximation des pairages.) En mesurant S^{pd} pour toutes les paires d'un ensemble Σ de signatures on pourra donc construire un arbre de groupage hiérarchique (*clustering*) et cet arbre aura différents usages dont l'établissement d'un alignement multiple des séquences basé sur la dynamique des structures.

Le meilleur chemin est utile si on veut obtenir l'alignement entre les signatures. On laisse alors des pointeurs à rebours dans la table. On peut également construire des

alignements multiples de ces structures. Pour cela nous utilisons l'arbre de groupage dont nous construisons, pour chaque nœud un alignement multiple.

Les contraintes sur les positions des alignements (les α), $\left(\sum_{\eta=1}^{7} DIFF\left(\alpha_{\eta,i}^{1}, \alpha_{\eta,k}^{1}\right)\right) > \delta$ garantie qu'aucun trou ne sera inséré entre deux positions dont les 7-tuples sont plus similaire que le seuil (choisi arbitrairement à δ=0.5) donnant les récurrences suivantes (également optimisables):

$$S_{i,j}^{pd} = MAX \begin{cases} \Theta\left(\sigma_i^1, \sigma_j^2\right) + \begin{cases} 0, si\ i < 2\ ou\ j < 2 \\ S_{i-1,j-1}, sinon \end{cases} \\ MIN_{1 \le k < i} S_{k,j} + \mathfrak{T} + (i-k)\mathfrak{S}, \quad si \begin{cases} i > 1, et\ si \\ \left(\sum_{\eta=1}^{7} DIFF\left(\alpha_{\eta,i}^{1}, \alpha_{\eta,k}^{1}\right)\right) > \delta \end{cases} \\ MIN_{1 \le k < j} S_{i,k} + \mathfrak{T} + (j-k)\mathfrak{S}, si \begin{cases} j > 1, et\ si \\ \left(\sum_{\eta=1}^{7} DIFF\left(\alpha_{\eta,j}^{2}, \alpha_{\eta,k}^{2}\right)\right) > \delta \end{cases} \end{cases}$$

2.1.4 Faut-il assigner un poids à chaque structure?

Chaque paire de structures sous-optimales est séparée par une barrière d'énergie de hauteur donnée et qui lui est propre. Hauteur que nous n'avons pas cherché à estimer et qui serait nécessaire à établir l'abondance relative de la population dans un état particulier. Une question qui vient à l'esprit demande s'il serait possible d'approximer ces abondances relatives en utilisant l'énergie prédite par le modèle NCM pour les structures sous-optimales. Cette idée mélange deux concepts : 1- la stabilité thermodynamique, et 2- la cinétique structurale.

Lorsque une molécule d'ARN change de structure, un certain nombre de forces sont en jeu. Les interactions intramoléculaires tels que les ponts hydrogènes et les interactions d'empilement des bases azotées tendent à stabiliser la structure courante. Mais, d'une part,

ces molécules n'existent pas en isolation et interagissent avec des myriades de molécules aux géométries et caractéristiques physicochimiques variées et, d'autre part, elles baignent dans un bassin thermique soumis à un bruit continuel dont l'ampleur est difficile à mesurer. Une vue réaliste de la structure d'ARN doit considérer qu'une paire de base donnée fluctue entre l'existence et la non-existence. Il arrive que certains systèmes oscillent à des rythmes constants entre un nombre fini d'états mais *a priori* il ne nous est pas possible d'assumer que cela soit le cas pour une paire de base donnée. Ce que nous savons, c'est que certaines molécules occupent certaines structures lorsqu'elles cristallisent et, pour les plus petites molécules étudiables en RMN, que certains de leurs atomes sont en général rapprochés. Les changements successifs qu'une molécule peut subir dans sa structure l'emmènent sur des chemins qui augmentent son énergie interne et lui donnent des structures improbables (au sens de la thermodynamique). Au fil de cette exploration, il est possible qu'elle fréquente plus souvent ou, même reste prise, dans des puits plus profonds (la MFE en particulier). La molécule subit en continu ces changements et un ensemble de molécules occupe donc un nuage de structures et les échanges entre les structures se produisent aux rythmes dictés par l'énergie interne des molécules et la hauteur des barrières d'énergies qui séparent les structures (effet de la cinétique). Il est donc improbable que nous puissions capturer les effets cinétiques sur les réseaux de structures simplement en assignant à chaque structure la probabilité calculée par un outil prédisant la stabilité thermodynamique. Nous rappelons ici au lecteur que notre propos n'est pas de produire une signature représentant fidèlement de l'espace structural au sens énergétique. Nous cherchons plutôt à mettre en évidence les ensembles de structures possiblement visitées dans une bande d'énergie proche de l'optimal.

2.1.5 Choix du nombre de sous-optimaux (κ)

Le nombre idéal de structures sous-optimales est celui qui donne le meilleur signal de dynamique. Mais, à ce jour, les méthodes de laboratoire sont incapables d'identifier les ensembles de structures sous-optimales effectivement visitées par les molécules. Nous savons que si $|S^4(s)|$ est trop petit, il ne reflètera pas convenablement la dynamique

structurale. D'autre part, s'il est trop grand, à la limite, il contiendra toutes les structures possibles pour une séquence de longueur donnée et n'offrira pas d'information quant aux particularités structurales issues de la composition de la séquence. Combien de structures sous-optimales faut-il choisir alors?

En absence d'argument convaincant nous sommes réduits à choisir une approche pragmatique et ciblée pour chaque nouveau problème. Notamment, la recherche d'homologies dans la dynamique de fenêtres (voir chapitre 3) nécessitera une approche différente que celle utilisée pour des transcrits de microARN (article sur miR-125a). En aucun cas est-il possible d'offrir une garantie quant à la représentativité de l'ensemble $S^{\Delta^{\kappa}}(s)$ sur la dynamique car nous ne connaissons pas, pour une séquence donnée, la forme du profil énergétique des structures sous-optimales. Un profil énergétique aplati permet une dynamique impliquant un plus grand nombre de structures qu'un profil en entonnoir par exemple.

Δ^{κ} donne l'écart entre la MFE et la structure la moins stable dans $S^{\Delta^{\kappa}}(s)$. Si deux molécules ont des dynamiques similaires, alors leurs Δ^{κ} devraient aussi être similaires mais cela ne serait pas un indicateur suffisant pour établir que les dynamiques soient similaires.

Les possibilités sont :

1. Fixer κ le nombre de sous-optimaux. L'avantage principal de cette approche vient de ce que le temps de calcul est prévisible lorsque un grand nombre de séquences doivent être comparées (comme c'est le cas lors de recherche dans des génomes par exemple).
2. Fixer Δ en kcal/mol permet d'émettre des hypothèses sur la quantité d'énergie impliquée dans la dynamique. Par exemple en stipulant que les contributions thermiques additionnées aux contributions de l'environnement chimique sont limitées à une valeur fixe. Cette approche souffre d'un certain manque de réalisme car, d'une part les contributions de l'environnement sont variables selon les affinités de l'ARN pour les molécules de son environnement et d'autre part les effets cinétiques sont

invisibles à notre approche. Pour les travaux de (Dethoff, Petzold et al. 2012) un Δ fixé entre 3 et 6 kcal/mol (pseudo-énergie) semblent raisonable mais il n'est pas possible de trouver une valeur universelle de Δ donnant des temps raisonnables de calcul à cause de la variabilité de dW (le taux de croissance de $|S^{\Delta}(s)|$ en fonction de Δ).

3. Fixer Δ à une fraction de la MFE. Cette approche fonctionne bien en pratique pour des séquences de longueurs et de comportement qui se ressemblent mais dont le profil énergétique des structures sous-optimales varie.

2.1.6 Faut-il considérer toutes les structures?

Nous pouvons connecter les structures entre elles formant des groupes disjoints de structures liées par une arête si un changement mineur (déplacement d'une paire de base) connecte deux structures (Flamm, Hovacker et al. 2002). Après un tel groupage, choisissons dans chaque composante de ce graphe disjoint, une structure représentative. Nous avons choisi la structure la plus stable dans chaque composante, notée cMFE. Notons $\widehat{S^{\Delta^{\kappa}}(s)}$ l'ensemble des cMFE et les signatures correspondantes $\widehat{\sigma^{s}}$.

Dans l'article qui suit, à la figure 3F, nous explorons des mesures pour comparer des ensembles de structures 2D : Le nombre de structures (κ), le nombre de composantes, la taille moyenne des composantes et la dynamique sur les $\widehat{\sigma^{s}}$. L'analyse des $\widehat{\sigma^{s}}$ est prometteuse mais donne un signal moins stable que celui obtenu par les σ^{s}.

Une autre question d'intérêt consiste est à savoir s'il ne vaudrait pas mieux utiliser une approche basée sur la fonction de partition (McCaskill 1990) que d'énumérer comme nous le faisons, certains nombres de structures sous-optimales pour en extraire des signatures. Rapellons que nous sommes intéressé par les fluctuations spontanées et induites dans la structure secondaire dans une bande d'énergie proche de l'optimal. Dans les prochaines sections nous reprennons certaines expériences clés en résonnance magnétique nucléaire qui indique clairement que ces structures existent bel et bien. Il n'est pas clair que de telles structures soient bien représentées par la fonction de partition. Dans (Dallaire and

Major 2015) nous montrons que l'importance des structures sous-optimales peut être dramatiquement réduite lorsqu'on considère un grand nombre de structures.

2.2 Structural Dynamics Controls the MicroRNA Maturation Pathway

Paul Dallaire[1], Huiping Tan[2], Keith Szulwach[2], Christopher Ma[2], Peng Jin[2], and François Major[1]

[1]Institute for Research in Immunology and Cancer, and Department of Computer Science and Operations Research, Université de Montréal, PO Box 6128, Downtown Station, Montréal, Québec, Canada H3C 3J7

[2]Department of Human Genetics, Emory University School of Medicine, Atlanta, GA 30322 USA

Keywords: MicroRNA; microRNA biogenesis; gene expression; cancer; miR-125a; single-nucleotide polymorphism; structural dynamics.

2.2.1 SUMMARY

MicroRNAs (miRNAs) are crucial gene expression regulators and first-order suspects in cancer development and progression. Many miRNAs found deleted or greatly down-regulated have been shown to lead to oncogenic conditions. The loss of mature miRNAs can result from single-nucleotide polymorphisms (SNPs). However, the reasons why a SNP can modulate the maturation of a miRNA are weakly understood. Here, using a quantitative model of RNA structural dynamics, we compared the dynamics of miRNA variants with their maturation levels. The comparison revealed a high correlation, suggesting that transient structures control miRNA maturation efficacy. The high correlations also reveal this model of RNA dynamics to be effective in classifying miRNAs according to maturation levels. Applied systematically to reported genetic variations in human miRNAs, we predict that at least about 25% may result in loss of expression.

[136 words]

2.2.2 INTRODUCTION

The secondary (2D) structure of a ribonucleic acid (RNA) sequence is defined by the Watson-Crick interactions forming between complementary bases within itself, A●U and G●C (Gralla, Steitz et al. 1974). However, functional RNA in cells are rarely fully complementary, giving rise to single-stranded dynamic regions that ease conformational changes inherent to RNA function(Williamson 2000; Mandal and Breaker 2004). Recently, transitions between low-abundance excited structural states (ESs) were visualized by

nuclear magnetic resonance (NMR) relaxation dispersion (Dethoff, Chugh et al. 2012). RNA dynamics were linked to function by showing that sequence mutations affecting the relative abundance of the ESs result in modulations of function (Dethoff, Chugh et al. 2012).

The recent NMR studies were made on rather short RNAs of lengths of less than 40 nucleotides. The number of ESs increases with RNA size (Dethoff, Chugh et al. 2012), and it is uncertain whether NMR studies will soon be applicable to larger RNAs. On the other hand, computational methods are free from the limitations of experimental techniques. For instance, RNA 2D structure prediction programs are known to generate and evaluate very large numbers of alternative conformations (McCaskill 1990; Hofacker, Bernhart et al. 2004; Mathews 2004; Ding 2006). RNA structures have been studied computationally, providing new insights and steering further functional investigations (Sim, Minary et al. 2012). Interestingly, the ESs observed by NMR are also predicted computationally, and there is a high correlation between computational and experimental energy changes between ground states (GSs) and ESs (Dethoff, Chugh et al. 2012). The correlation is observed in sets of suboptimal conformations that include non-canonical base pairs (Parisien and Major 2008), which were found to be the main differences between the ESs of the studied sequences (Dethoff, Chugh et al. 2012).

These results from NMR studies prompted us to formalize the impact of RNA sequence changes on function in terms of transition states. Consider the transition network of the twelve most stable structures of the *Escherichia coli* 16S ribosomal RNA A-site (Figure 1A). In this network, a pair of structures is linked if it is connected by a single basic

transition, either a base pair formation or a bulge migration. The A-site structure bound to a mRNA::tRNA complex (Kondo and Westhof 2008), labelled BS, was not observed by NMR relaxation dispersion (Dethoff, Chugh et al. 2012). The NMR conditions without the ribosomal context favoured the path through the bulge migration and towards the ES. The crystallographic structure of the *E. coli* ribosome bound to a mRNA::tRNA complex induces the base pair transition towards the BS. Interestingly, the GS, which corresponds to the A-site conformation of the ribosome unbound to the complex, is the most stable state observed, by NMR and X-ray crystallography, as well as computationally. It is noteworthy that the computed network in Figure 1A includes the structures visualized in different experimental conditions.

The basic transitions that define the network of transient states are fully determined by the set of structures that constitute it, and thus this set of structures contains the same information than does the network. Consider two RNA sequences that induce different sets of structures. These two sequences define two different networks and a direct comparison of the two sets of structures thus provides information about their relative dynamics. For such comparisons, we developed a metric. Given two sets of structures, it returns a value that represents the distance between their dynamics.

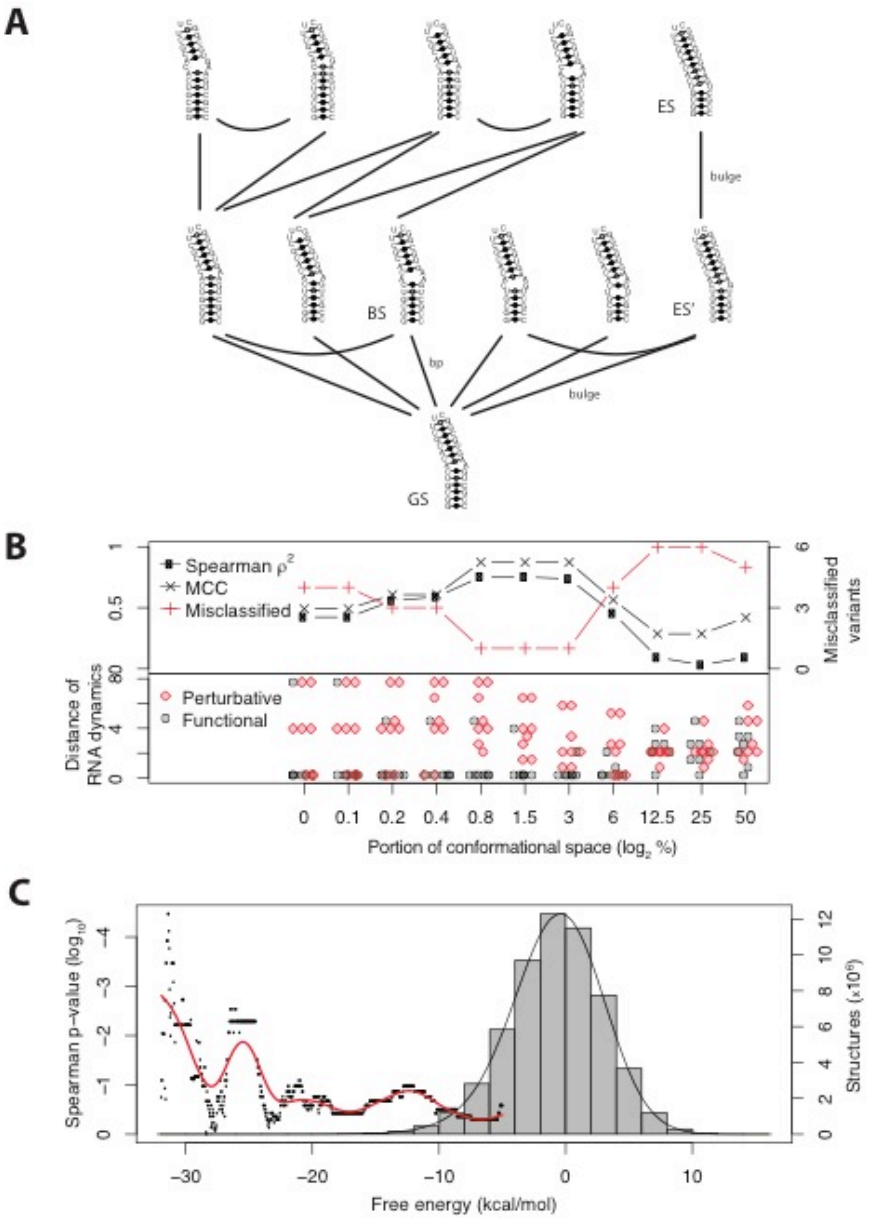

Figure 9. (Fig.1) Model of RNA dynamics.

(A) The twelve most stable predicted structures of the Escherichia coli 16S ribosomal RNA A-site include the ground state (GS), the mRNA::tRNA bound (BS) and unbound states (GS), and the excited state (ES) observed by NMR relaxation dispersion. ES' is a necessary transition state between the GS and the ES. The GS and BS are linked by a base pair formation transition (bp); a A◦A bp in the GS is lost in the BS. The GS and ES', as well as the ES' and ES are linked by single bulge migration transitions (bulge); from 5' to 3',

A bulge in the GS, then G in the ES', and then U in the ES. Canonical Watson-Crick base pairs are indicated by black dots; Wobble base pairs by grey dots; and, non-canonical base pairs by white dots.

(B) Top: The likelihood to predict the functionality of fifteen A-site sequence variants using increasing portions of their conformational space, expressed in square Spearman correlation (ρ2) (two-sided configuration), Matthews correlation coefficient (MCC), and number of misclassified variants. Bottom: Separation between functional (grey circles) and perturbative (red diamonds) sequence variants at different portions of the conformational space considered.

(C) Spearman correlation p-values as the portion of conformational space considered increases (black dots). The red line is a smoothing of the p-values. The histogram indicates the number of structures at different energy levels.

2.2.3 RESULTS

2.2.3.1 Model of RNA dynamics

An interesting question is how many structures must be considered to get an accurate comparison of structural dynamics? In the absence of *a priori* knowledge, we used increasing portions of the conformational space of fifteen A-site sequence variants, some of which are known to perturb translation, by either affecting binding to the translation initiation factor, or promoting read-through and frame-shifting (O'Connor, Thomas et al. 1997). Figure 1B shows the separation of the A-site variants as a function of conformational portion. As we see, the curves reach a plateau where all but one sequence variant is misclassified. Interestingly, the left and right sides of the plateau indicate, respectively, smaller and larger portions of the conformational space where more variants are misclassified. This indicates that the structures defined within 1% of the conformational space contains the information needed to classify properly the variants, while taking too

many structures introduces noise. Figure 1C shows the exploration of a much larger partition of the conformational space. Again, we see that small sets of stable structures contain more information about the function of the variants (small Spearman p-values, left side of the red line), than larger sets of structures that include less and less stable structures (higher Spearman p-values, right side of the red line). Increasing the considered portion of the conformational space incurs noise and reduces the signal.

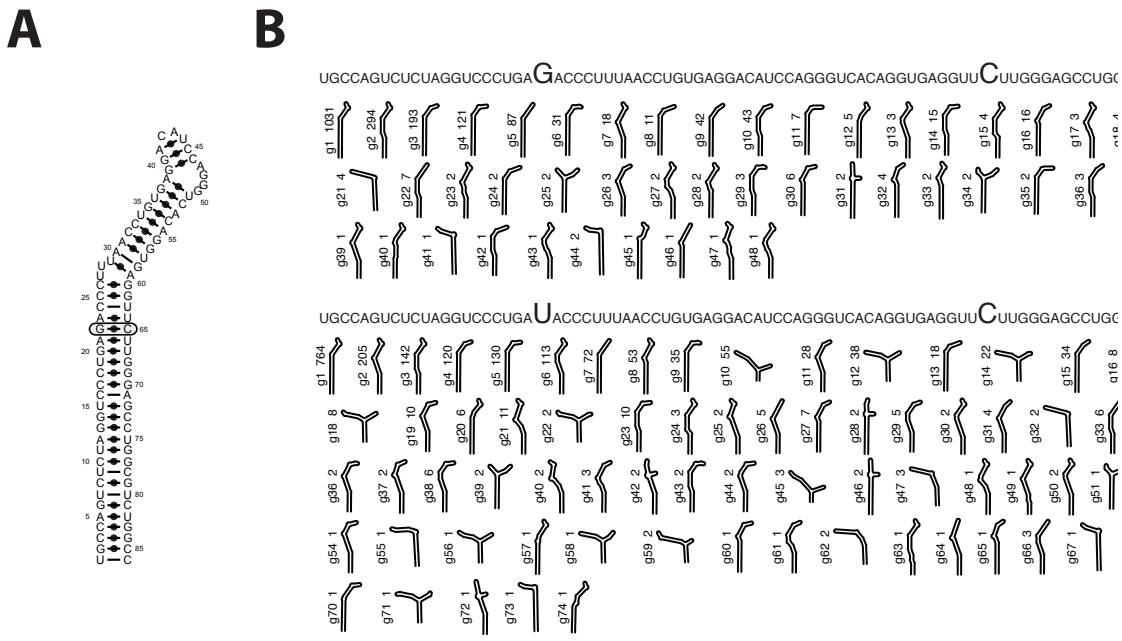

Figure 10. (fig 2) Energy minima of human pre-miR-125a structures.

(A) Minimum free energy structure (global minimum) of human pre-miR-125a alleles. The G22U SNP is circled.

(B) The 2,000 most stable 2D structure predictions of the major allele define 48 energy minima (top), while those of the minor allele define 74 (bottom). The basin defined by each local minimum is represented by its 2D structure, energy rank (g1, g2, ...) , and size.

2.2.3.2 Maturation efficacy

MiRNAs are transcribed from genomic DNA in primary transcripts (pri-miRNA), which are recognized and transformed by many proteins and complexes before they result in mature products of about 22 nucleotides (Lee, Ahn et al. 2003; Denli, Tops et al. 2004; Gregory, Yan et al. 2004). Each one of the miRNA protein partners requires a suitable and possibly induced-fit structure that exhibits various structural features for both specific recognition and processing. Among the first miRNA processing steps, the pri-miRNA is recognized and cleaved to a precursor form (pre-miRNA) by the DROSHA/DGCR8 microprocessor complex. Then, EXPORTIN-5 exports pre-miRNAs to the cytoplasm (Yi, Qin et al. 2003; Lund, Guttinger et al. 2004), where the RNA-Induced Silencing Complex, which includes DICER and an Argonaute protein, cleaves their loops and loads the mature products (Bernstein, Caudy et al. 2001; Grishok, Pasquinelli et al. 2001; Hutvagner, McLachlan et al. 2001; Ketting, Fischer et al. 2001). Other proteins have been shown to be involved in the maturation of specific miRNAs, such as LIN28 that mediates the terminal uridylation and blockage of let-7 (Heo, Joo et al. 2008; Viswanathan, Daley et al. 2008), SMAD that promotes miRNA maturation by DROSHA (Davis, Hilyard et al. 2010), and hnRNP A17 that is required for the processing of miR-18a (Guil and Caceres 2007; Michlewski, Guil et al. 2008). The complete maturation process has not yet been fully characterized, and additional steps and factors may yet be discovered.

A reduced abundance of miR-125a mature products caused by a single-nucleotide polymorphism (SNP) found in a minor allele has been suggested to be involved in breast cancer tumorigenesis (Li, Duan et al. 2009). The SNP is found in the seed region of the mature miRNA, and was initially thought to hinder the miRNA interaction with its target

mRNAs. Surprisingly, it was shown to block DROSHA processing instead (Duan, Pak et al. 2007). The major and minor alleles of pri-miR-125a fold into the same GS. In the minor allele, a G●C base pair is substituted by an isosteric non-canonical U●C base pair that slightly affects the energy and stability of the GS (Figure 2A). The loss in maturation is not due to the loss of a canonical base pair, because a substitution into the A●U base pair also leads to a weak maturation efficacy (Table S1).

Recently, the structural requirements for DROSHA processing of a few pri-miRNAs were shown to be subject to the dynamics of a specific region (Quarles, Sahu et al. 2013). We thus asked whether miRNA dynamics could be linked to other steps in miRNA processing and overall maturation efficacies. We computed the first 2,000 most stable structures of the major and minor alleles of miR-125a. A comparison of the transition networks defined by these sets of structures indicated huge differences, both in the size and energy of local minima (Figure 2B). Two of the most striking differences between the major and minor alleles are the change in the numbers of minima, from 48 to 74, and reduced size of the structures connected to the GS, from 1031 to 764. We also noted that the seventh minimum of the major allele, g7 (Figure 2B top), ranks 2nd among the minima of the minor allele, g2 (Figure 2B bottom), with an important increase in the number of structures connected to it, from 18 to 205.

To compare dynamics with maturation efficacies, we engineered the two miR-125a alleles, as well as the fourteen other variants to address all possible base pairs at the position of the SNP. We expressed each variant in cultured cells, and measured their maturation levels by qRT-PCR (Table S1). We then compared the maturation data with the distance of miRNA

dynamics for various portions of their conformational space (Figure 3A). The maturation levels were lower for each variant when compared to the major allele. We observed a strong link between maturation levels and relative distances of miRNA dynamics, revealed by the high correlation between the two factors (ρ = -0.75) and ability to use pairwise distances of dynamics to classify the variants of similar maturation levels (Figure 3B). Except for variants GG and GU, all variants whose processing is proficient (> 20% compared to the major allele) share a cluster that includes the major allele. The minor allele, which is one of the least processed, shares a cluster with other lowly expressed variants that are located far from the major allele.

We applied the same hierarchical clustering approach to other data sets used in previous studies of miRNA processing by the DROSHA/DGCR8 microprocessor. The results corroborated with those of the miR-125a variants, i.e. we observed a strong correlation between distance of miRNA dynamics and maturation efficacies: miR-30a (Lee, Ahn et al. 2003) (Figure 3C) and miR-21 (Zeng and Cullen 2003) (Figure 3D). Relative expression levels of miR-30 *in vitro* were reported to be either processed or not. In the dendrogram of the clustering result, the two unprocessed variants (labeled M4 and M5) are clearly set apart from the processed variants. The seven variants of miR-21 were cloned, expressed in HEK293 cells, and their processing measured by northern blotting. The normalized distances of the miR-21 variants' dynamics to the major allele were computed using the precursor sequence, flanked by the complete expressed primary transcript of 312 nucleotides. The dendrogram clearly separated the variants into two clusters, one of which

contains the two lowly processed variants (labeled G1 and GGU), and the other contains the highly processed variants (all but G2).

The strong link we observe between distances of miRNA dynamics and maturation efficacies suggests that: 1) multiple transient structures are involved in miRNA maturation; 2) these transient structures are encoded in miRNA primary transcripts; and, 3) the transitions between these structures determine miRNA fate. As we can see in Figure 3E, the dynamic profiles of the miR-125a variants exhibit a great degree of variability, quantified as single strandedness. At the moment, it is unclear which miRNA region participates in which maturation step. The G22●C65 base pair in the major allele stabilizes the surrounding region of each partner. They are indicated by the blue line in the low single strandedness area centered at the dotted lines. The variants that stabilize these regions (blue lines) are processed more efficiently by the DROSHA/DGCR8 complex than those that destabilize them (red lines). Whether the unstable regions interfere with the interaction between the miRNA and the DROSHA/DGCR8 complex, or favor an interaction with another yet unknown factor in the nucleus is unclear and will need to be determined experimentally.

Interestingly, we see that other regions need flexibility (rather than stability) to enable efficient miRNA maturation. See for instance the regions in Figure 3E where the blue lines have higher single strandedness than the red lines (e.g. near nucleotides 9, 16, 19, and 48). In particular, the region near nucleotide 9 is next to a DROSHA cut site and is flexible in most miRNAs (Han, Lee et al. 2006). The flexibility of this region in miR-16-1, miR-30a, and miR-107 was confirmed experimentally by selective 2'-hydroxyl acylation by primer

extension chemistry (Quarles, Sahu et al. 2013). Furthermore, variants of these three miRNAs that stabilize the region were shown to mature less efficiently than their corresponding endogenous miRNAs *in vitro* (Quarles, Sahu et al. 2013).

Another interesting aspect that can be observed from the dynamic profiles of miR-125a variants is the scope SNPs can have in distant regions of the sequence (Figure 3E). This is shown by the difference span in single strandedness in many regions (e.g. near nucleotides 16, 28, 48, 57, 59, 72).Whether the loss in DROSHA processing of the minor allele of miR-125a is due to the dynamics introduced at the SNP position or in another region (e.g. 57, 59, 72), or to increased stability induced by the SNP (e.g. 16, 28, 48) will also need to be determined experimentally.

Finally, to determine which statistical aspects of the sets of predicted structures best captured the maturation signal of miR-125a variants, we questioned whether the correlation is observed using simpler statistics or less data. Surprisingly, we found that the local minimum structures alone contains sufficient information to classify the miR-125a variants according to their maturation efficacies (Figure 3F). This further emphasizes that a small number of the most stable transient structures control the maturation fate of pri-miRNA sequences.

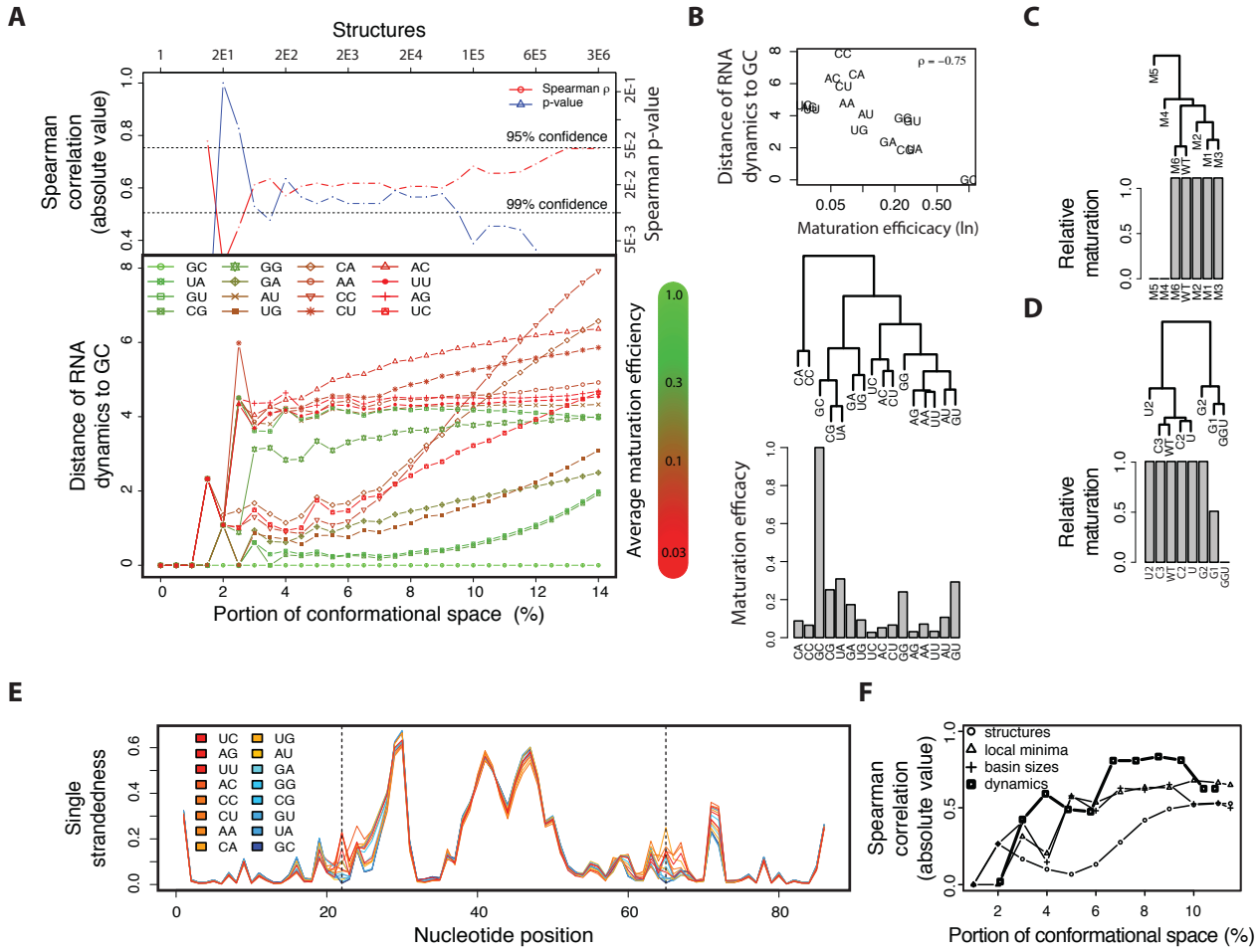

Figure 11. (fig 3) MiRNA dynamics and maturation efficacies.

(A) Bottom: comparative dynamics of miR-125a variants at increasing portions of conformational space (up to 14%, 3×106 structures) colored according to the log of their maturation efficacy from low (red) to high (green). Top: corresponding Spearman correlation (ρ) and p-value. (B) Top: the distance of RNA dynamics to the major allele (GC) increases as maturation efficacy decreases. Middle: hierarchical clustering based on pairwise distances of RNA dynamics. Bottom: maturation efficacies.

(C) Hierarchical clustering and maturation efficacies of seven miR-30a variants using the top 1,000 predicted structures. MiR-30a major allele indicated as WT. (D) Hierarchical clustering and maturation efficacies of eight miR-21 variants using the top 1,000 predicted structures. MiR-21 major allele indicated as WT. (E) Dynamic profiles of miR-125a variants as the single strandedness of each nucleotide. The vertical dotted lines indicate the mutated 22::65 base pairing partners. Each line corresponds to a variant and is colored according to its maturation efficacy, from low (red) to high (blue). (F) Spearman correlation at increasing portions of conformational space using the number of structures, number of local minima, average

number of structures connected to local minima (basin sizes), and distance of RNA dynamics using local minima alone (dynamics).

Tableau 1. (table S1) Expression data of miR-125a variants.

Mature	*GC*	*GA*	*GT*	*GG*	*AC*	*AA*	*AT*	*AG*	*CC*	*CA*	*CT*	*CG*	*TC*	*TA*	*TT*	*TG*	*PSM2*
group1	1,00	0,64	1,35	0,26	0,07	0,08	0,20	0,12	0,19	0,27	0,10	0,71	0,09	1,14	0,10	0,23	0,09
group2	1,00	0,20	0,34	0,32	0,21	0,08	0,18	0,09	0,08	0,14	0,10	0,42	0,04	0,59	0,05	0,11	0,00
group3	1,00	0,28	0,44	0,28	0,08	0,04	0,16	0,06	0,15	0,14	0,12	0,58	0,05	1,09	0,03	0,09	0,05
group4	1,00	0,40	0,48	0,40	0,12	0,07	0,19	0,07	0,09	0,15	0,09	0,92	0,06	1,66	0,07	0,14	0,02
group5	1,00	0,34	0,41	0,50	0,10	0,04	0,17	0,04	0,06	0,12	0,07	0,54	0,04	1,04	0,03	0,07	0,03
Pri																	
group1	1,00	3,95	3,55	1,67	3,08	1,12	1,74	3,23	3,43	2,44	1,63	3,53	2,21	2,97	2,19	1,90	0,16
group2	1,00	1,54	1,23	1,01	2,15	1,14	1,87	2,40	1,37	1,56	0,96	1,71	1,61	2,23	1,26	1,15	0,09
group3	1,00	1,67	2,18	1,37	1,99	0,63	1,57	2,23	1,36	1,83	1,93	1,66	2,34	7,44	1,31	0,95	0,10
group4	1,00	0,96	1,10	0,98	0,77	1,16	1,05	1,64	0,87	1,70	0,76	1,47	0,78	1,51	0,48	0,53	0,03
group5	1,00	1,45	1,31	1,75	2,13	0,50	1,49	1,69	1,63	1,60	1,89	2,56	1,65	2,36	1,26	1,18	0,07
Pri+Pre																	
group1	1,00	3,57	3,79	1,54	1,47	1,12	1,11	1,90	2,36	2,08	1,08	2,66	1,48	3,19	1,57	1,74	0,10
group2	1,00	1,26	1,18	0,90	1,62	0,96	1,20	1,43	0,91	1,79	0,93	1,31	1,16	2,27	0,84	1,18	0,11
group3	1,00	1,67	2,41	1,21	1,49	0,80	1,36	1,90	1,59	1,49	1,65	1,66	2,92	7,66	0,90	1,07	0,08
group4	1,00	1,23	1,22	1,03	0,89	1,09	1,47	1,57	0,80	1,54	0,72	1,11	0,80	2,35	0,58	0,71	0,05
group5	1,00	1,67	1,35	1,39	1,46	0,77	0,85	1,45	1,18	1,88	1,12	1,52	0,90	2,67	1,03	1,06	0,06

Measures of the mature, primary transcripts (Pri) and precursors (Pri+Pre) were taken in five replicates (group1 to group5). Precursor measures were labeled Pri+Pre since precursor primer hybridization with the primary transcripts is unavoidable.

2.2.3.3 Loss of miRNA maturation due to genetic variations in humans

The point mutation A40G in pre-miR-27a (dbSNP #RS895819) is associated with a reduced familial breast cancer risk (Yang, Schlehe et al. 2010), but gastric cancer susceptibility (Sun, Gu et al. 2010). The expression level of miR-27a was found decreased by about 8% in gastric tumor samples of patients with the G40 alleles compared with patients with the A40 alleles, whereas it was found in between G40 and A40 homozygote patient levels in heterozygote patients. This shows that common differences in miRNA sequences that induce even moderate maturation efficacies can have significant consequences on disease risks. These observations prompted us to investigate the scope on miRNA maturation of other reported SNPs in human miRNAs.

Usually, it is difficult to predict the effect of genomic variations that map to non protein-coding genes (Raychaudhuri 2011). We used our metric of RNA dynamics to predict if genetically altered miRNAs, as available in dbSNP (Sayers, Barrett et al. 2012), could have perturbed maturation. We considered 2,017 different genomic positions. For each position, we kept the SNPs that induce the most disturbing effect on RNA dynamics. We found that the dynamics of almost half of these SNPs (964/2,017) are affected at least as much as that of the miR-125a minor allele, and that the majority (1,223/2,017) do not induce changes in GS. The intersection includes 316 SNPs (Figure 4**;** tab. S2). The effect of these variants on miRNA maturation would be difficult to predict otherwise than with using our metric of RNA dynamics. These variations are potentially clinically relevant. It is thus reasonable to further investigate this group of mutated miRNAs (tab. S2), which are likely to be involved in health risks and diseases.

Data not shown

Table S2. Ground state relative change and distance of RNA dynamics of miRNA SNPs.

The 2,219 miRNA SNPs are listed with their maximum dissimilarity to the GS (eq. 1) and distance of RNA dynamics using 1,000 suboptimal predicted structures.

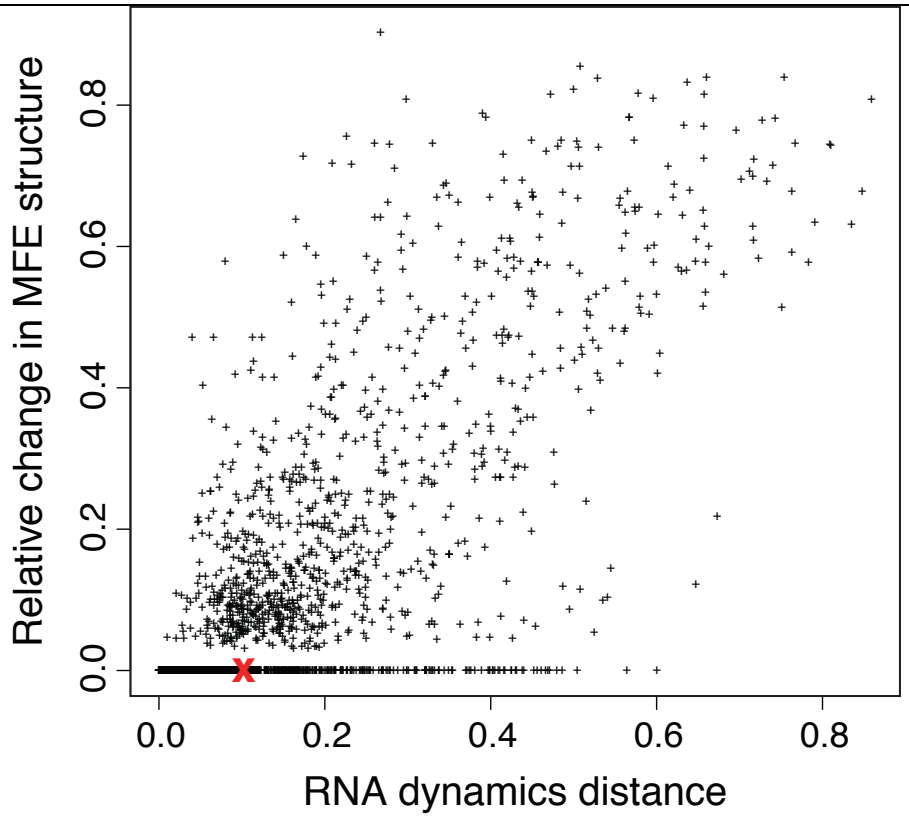

Figure 12. (fig 4) Predicted effects on ground state and dynamics of genetic variations in human miRNAs.

SNPs from dbSNP that collocate at the same genomic position were grouped, and the maximum signal computed using eq. 1 is reported. The miR-125a minor allele is shown with a large red cross. 1,223 SNPs on the 2,017 analyzed showed no change in GS, and only five showed no change in dynamics. The dynamics of 964 SNPed miRNAs are more affected than that of the miR-125a minor allele. The intersection of these two sets includes 316 groups.

2.2.4 DISCUSSION

Since miRNAs were discovered about a decade ago (Lee, Feinbaum et al. 1993), characterizing the mechanisms behind miRNA biogenesis has been the focus of a lot of research (Breving and Esquela-Kerscher 2010). The realization of the role of miRNA dynamics in their mechanisms of action brings a new perspective into this puzzling challenge. Using a quantitative model, we captured changes in miRNA maturation that are direct consequences of their dynamics. Our results suggest that the miRNA maturation pathway is highly sensitive to sequence variations that modulate the topology of a miRNA's network of transient structures. We focused on a SNP in miR-125a that interferes with maturation and is almost exclusive to breast cancer patients (Li, Duan et al. 2009), the ribosomal A-site, and a few other miRNA examples. In general, our model of RNA dynamics most certainly applies to many, if not all, RNAs, and links RNA dynamics and function (Kloc, Dallaire et al. 2011; Gkogkas, Khoutorsky et al. 2013). Besides, the model of RNA dynamics suggests similarities between RNA and intrinsically disordered proteins, which fold and function upon specific chemical signals shown crucial in ligand binding (Ferreon, Ferreon et al. 2013).

The computational determination of RNA structure is often understood as a search for the most stable, native, or active conformation among a huge conformational space. Many researchers are now hypothesizing that changes inducing variations in this space can be linked to perturbations in RNA function. We provide new insights indicating that, 1) the most stable transient structures included in a small fraction of the conformational space are relevant to function; and, 2) non-canonical base pairs characterize the transitions between these transient conformational states. Understanding the effect of sequence variations on RNA function thus relies on the analysis of such networks of transient structures. Finally, we find remarkable that, given 2D structure

prediction approaches are based on static RNA structures (Parisien and Major 2008) and short oligonucleotide fragments in solution (Xia, SantaLucia et al. 1998), it is now possible to venture into the dynamic behavior of RNA molecules at the cellular level using sequence data alone.

2.2.5 EXPERIMENTAL PROCEDURES

2.2.5.1 Metric of RNA dynamics

The metric calculates the degree of similarity between two RNA sequences by comparing their sets of predicted stable secondary (2D) structures. To accomplish this, we: 1) predict the sets of stable 2D structures, including non-canonical base pairs, using the MC-Fold software (Parisien and Major 2008); 2) compute a statistical description of the base pairing patterns of each nucleotides from these sets (signature); and, 3) compute a similarity value between pairs of signatures. In some instances we were required to compute the local minima from the MC-Fold predicted sets of structures. These were obtained by brute force calculation (see below), in $O(Ns^2)$, where N is the sequence length for *s* enumerated structures. A fast algorithm for this problem has been reported (Flamm, Hovacker et al. 2002).

We predict either fixed numbers of structures (typically a few thousands), or all structures below an energy threshold expressed as a percentage of the minimum free energy. We build for each sequence a signature composed of seven values for each nucleotide, two for the paired and five for the unpaired conformations. When a nucleotide is involved in a base pairing conformation (i.e. either the 5' or 3' partner), it can interact with a number of different partners, yielding N-1 counts of base pairing events for a sequence of length N. For the paired conformations, we compute a statistic that summarizes these base pairing counts. We compared different base pairing statistics, and they showed very similar numerical behavior with the exception of the

median that is more volatile; we chose max. The unpaired nucleotides can occur in various structural contexts, and we distinguish between: (*i*) terminal loop; (*ii*) loop in the 5' arm of a stem; (*iii*) loop in the 3' arm of a stem; (*iv*) multi-loop junction; or, (*v*) dangling at either end of the structure. The seven values for a nucleotide were normalized to frequencies between 0 and 1, so that the signature is independent of sequence length.

Measuring the distance between two positions in these signatures can be done in several non-equivalent ways, and we chose to use the sum of the absolute differences in corresponding conformations. The result of comparing nucleotides represented as normalized counts in vectors **v** and **w** is thus simply:

$$\sum_{i=1..7} \left| \mathbf{v}_i - \mathbf{w}_i \right|,$$

and gives a distance score. To compare two signatures of equal length, we sum the results of these sums at every position.

Alternatively we used a similarity score:

$$\sum_{i=1..7} \mathrm{MIN} \left| \mathbf{v}_i - \mathbf{w}_i \right|$$

$$\sum_{i=1}^{7} MIN(v_i - w_i)$$

in a dynamic programming routine to compare signatures obtained from unequal lengths using an algorithm that is similar to sequence alignment. This yields the best possible nucleotide pairwise assignments and optimal score. We used the affine gap model, where a gap has two distinct penalty costs, gap initiation (-4.5) and gap elongation (-2.5), that are added to the overall

distance. This scheme allows for the selection of solutions with minimum numbers of gaps. Since the signatures could be of highly dissimilar lengths, we developed a version where a target length can be specified. Our computational scheme accepts a number of variations, some of which have been found useful previously to compare complete base pairing partition tables for computing guide trees for RNA structure alignments (Bonhoeffer, McCaskill et al. 1993), and to detect changes in the partition table induced by sequence variation (Halvorsen, Martin et al. 2010). These variations and relative pros and cons were recently discussed (Sabarinathan, Tafer et al. 2013a).

To a certain extent, our calculations aim at comparing truncated partition tables, in the same spirit as proposed by Lorenz and Clote (Lorenz and Clote 2011). However, there is no algorithm that computes the necessary partition table for MC-Fold. The use of MC-Fold was required because it was shown to predict correctly sparsely populated transient structural states which were visualized by NMR. This was possible in particular because of the MC-Fold's ability to compute the set of isosteric non-canonical base pairs (Parisien and Major 2008). To obtain large numbers of structures, we developed a very fast version of the MC-Fold program using dynamic programming (in preparation). The algorithm of the new version is similar to that developed by Wuchty and coworkers (Wuchty, Fontana et al. 1999).

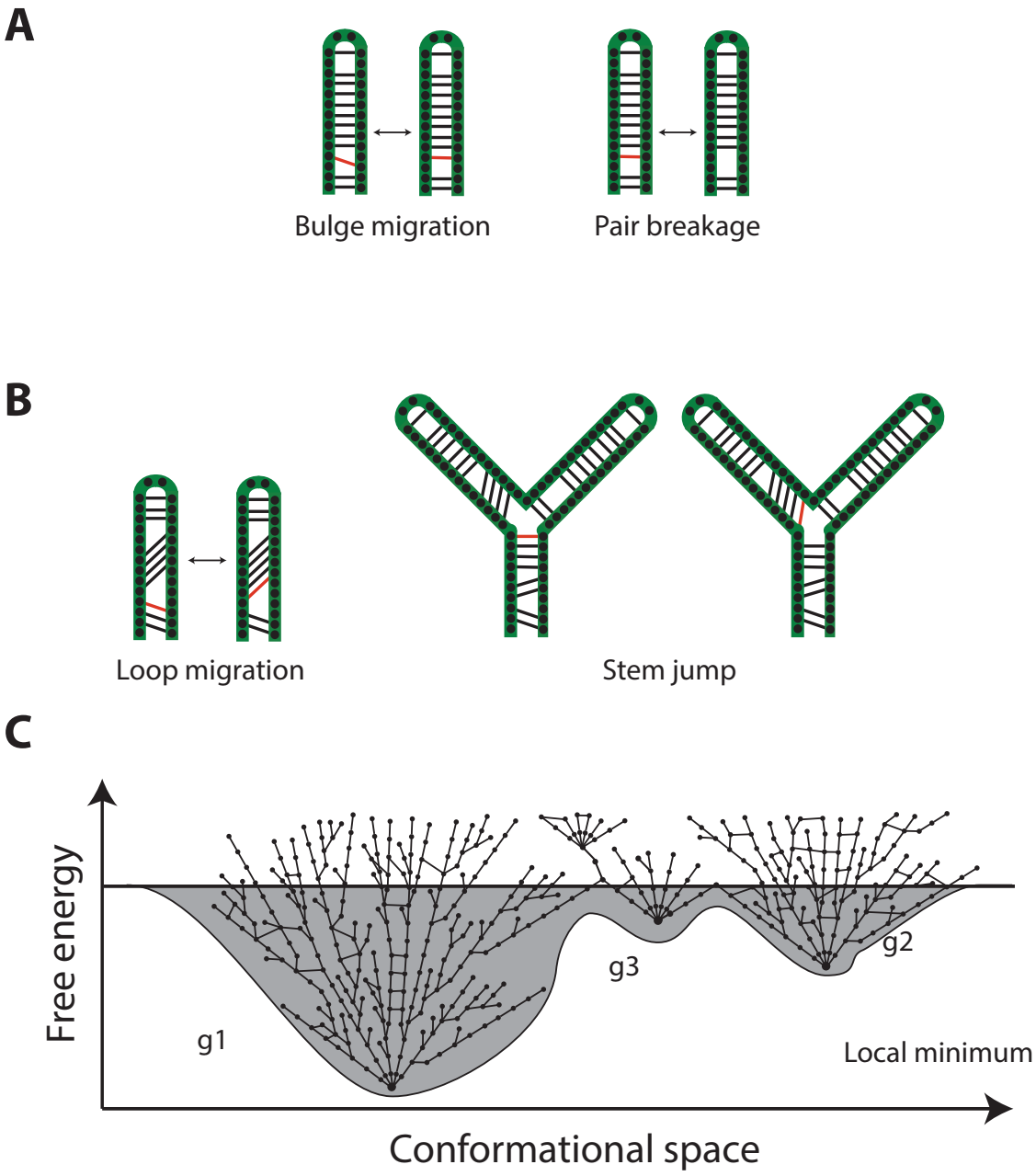

Figure 13. (fig 5) Networks of transient structures and local minima.

(A) Simple transformations considered in building networks of transient structures.

(B) Like for the bulge migration, three values change in the vector representation of two 2D structures connected by a loop migration or a stem jump. They differ from the bulge migration by the indices they modify in the vector representation.

(C) Connected transient structures. Using only structures below a fixed energy threshold (grey area) form groups (g1, g2 and g3). The most stable structures in each group define the local minima.

2.2.5.2 Networks of transient structures and local minima

A 2D RNA structure is represented as a vector **r**, where each nucleotide is assigned a number according to its pairing partner. If a nucleotide, *x,* is unpaired, then **r**[*x*] = -1. If nucleotide *i* is paired with nucleotide *j*, then **r**[*i*] = *j* and **r**[*j*] = *i*. A base pair breakage or formation implies that exactly two values in **r** are altered. Conversely, two vectors **r** and **s** that differ at exactly two indices are linked by a single base pair change. If a bulge migration occurs (Figure 5A), then three values in **r** are changed. However, events other than bulge migration, for instance a loop migration or stem jump (Figure 5B) also cause three values to change in **r** and their detection requires verification of positional invariants: two vectors differ solely by a bulge migration if and only if the values at three indices are changed and one pair of indices involves adjacent nucleotides interchanging their values. The detection of such differences between two structures from a set is computed in time proportional to the length of the vectors, *O(n)*. We incrementally compute groups of structures from a set by comparing all its member pairs so that all members in a group are connected by a series of single changes and no pair of structures taken from different groups are connected by such a suite of changes. From these groups, the most energetically stable structure is a local minimum (Figure 5C).

2.2.5.3 Dissimilarity of miRNA variant dynamics

The miRNA variants were folded using MC-Fold and distances of RNA dynamics were computed using 1,000 predicted structures, and compared to their respective reference sequence. Sequence variants from dbSNP found to occur at the same locations were grouped. This yielded 2,017 groups for further analysis. A score of dissimilarity was computed:

$$\text{Dissimilarity} = 1 - (\text{d}(\text{var}, ref)^2 / (\text{d}(ref, ref) \times \text{d}(\text{var}, \text{var}) \quad \textbf{(Eq. 1)},$$

where d is the distance of RNA dynamics, var is the variant and *ref* is the reference miRNA sequence. d is computed for each variant and the maximum for each group is reported on the x axis. The change in predicted structure for the minimum free energy structure is computed in the same manner.

2.2.5.4 DNA plasmids

To generate pSM2-miR-125a-G, sequence corresponding to the miR-125a precursor and 125 bp flanking region on each side was amplified from normal human genomic DNA and inserted into the pSM2 vector (Open Biosystems). Plasmid pSM2-miR125a-U was generated by introducing a G-T substitution at the sequence corresponding to the eighth position of the mature miR-125a (Stratagene, La Jolla, CA, USA). A double mutant (pSM2-miR125a-U-M) contained the G-T polymorphism in addition to a complementary C-A base change, reported (Duan, Pak et al. 2007) and other mutants were prepared in the same manner.

Cell culture, transfection, and luciferase assay

Human HEK293 cells were grown in Dulbecco's modified Eagle's medium (GIBCO) supplemented with 10% FBS and penicillin/streptomycin. Plasmids were transfected into cells by Lipofectamine 2000 (Invitrogen) following the manufacturer's protocol. Reporter plasmids and pRL-TK plasmid were used for transfection, and each sample was transfected in triplicate.

2.2.5.5 Quantitative RT–PCR

Total RNA from cultured cells were isolated with Trizol (Invitrogen). RNA samples were reverse-transcribed into cDNA with the ThermoScript First-Strand Synthesis System (Invitrogen). Real-time PCR was performed with gene-specific primers and Power SYBR Green PCR Master Mix using the 7500 Standard Real-Time PCR System (Applied Biosystems). The primers to detect the pri- and pre-miRNAs were:

5'-AATGTCTCTGTGCCTATCTCCATCT-3 '(F1; pri-miRNA only),
5'-GTCCCTGAAGCCCTTTAACC-3'(F2; pri- and pre-miRNAs), and
5'-AACCTCACCTGTGACCCTG-3'(R).

Relative quantities of mature miRNAs were determined using Applied Biosystems TaqMan microRNA Assays. Relative quantities of miRNA were determined using the DDCt method, as provided by the manufacturer. Individual assays were performed using primers and probes provided by Applied Biosystems.

2.2.6 AUTHORS CONTRIBUTIONS

Method design (PD and FM), method implementation (PD), experimental work (HT, KS, CM and PJ), and data analysis and writing (PD and FM).

2.2.7 ACKNOWLEDGMENTS

We thank Bryan R. Cullen for providing the flanking sequences of miR-21. This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (Discovery program), the Canadian Institutes of Health Research (CIHR) (grant number MOP-93679), the National Institutes of Health (grant number R01GM088813), and the Human Frontier Science Program (RGP0002/2009-C).

2.3 Résultats complémentaires

Dans l'introduction au chapitre, nous avons mentionné des mesures faites sur la compilation des statistiques de paires de bases dans le contexte du calcul de la signature.

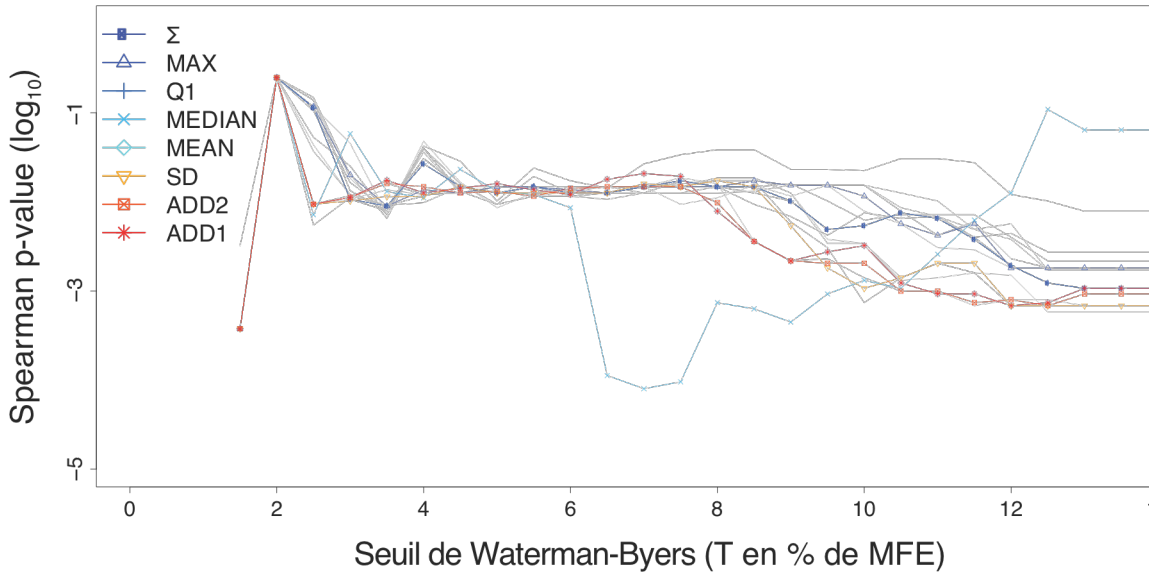

Figure 14 : Impact de la statistique de paires de bases.

Le taux de maturation des 16 variants de miR-125a est corrélé avec les signatures calculées pour des ensembles de structures secondaires pour ces séquences en utilisant des seuils croissants de Waterman-Byers donnés en pourcentage de la MFE. La p-valeur (two-sided) est donnée pour chaque correlation. La signature est calculée en utilisant diverses statistiques sur les paires de bases. Les lignes grisées sont les résultats de calculs sur des signatures sur des 9-tuples augmentées pour tenir compte de toutes les paires possibles de statistiques. La médiane ainsi que certaines combinaisons de paires de statistiques sont à éviter mais autrement, le système est

2.4 Conclusion au chapitre 2

L'hypothèse que les molécules d'ARN, sous l'influence du bruit thermique environnant et de leurs interactions intermoléculaires, changent dynamiquement de structures 2D a été démontrée au moins pour certaines molécules grâce à une approche novatrice de la résonance magnétique nucléaire. De plus, il a été montré que les structures sous-optimales prédites grâce au modèle des NCM identifient ces structures.

Dans ce chapitre, nous avons conçu un système qui compare les ensembles de structures sous-optimales prédites pour des séquences et qui calcule les distances qui les séparent. Plus cette distance est petite et plus leurs dynamique structurale est similaire. Nous avons mesuré les effets sur cette mesure de divers opérateurs de combinaison des signatures, de diverses statistiques sur les paires de bases dans la composition des signatures, de divers opérateurs de métrique dans la comparaison de signature, de l'effet de la puissance de la norme pendant la comparaison de nucléotides ainsi que l'effet du type de combinaison des bases dans la comparaison de signatures. La plupart de ces variations sur la méthode ont des effets mineurs sur la qualité de la capture du signal indiquant que le signal de dynamique moléculaire est robuste.

Lorsque la distance en dynamique structurale est importante entre deux molécules on peut s'attendre à ce que leurs fonctionnalités soient différentes. Cette idée a été testée avec succès sur des transcrits primaires de microARN. Nous avons démontré que des mutations qui affectent la maturation de miR-125a, de miR-30 et de miR-21 le font en accord avec la distance en dynamique structurale prédite par MC-Fold tel que mesuré par notre approche. Ainsi, à la question

$$T(s, s') \overset{?}{\rightarrow} \Delta F$$

on peut répondre par l'affirmative.

Ce résultat permet de tenter de prédire l'effet de mutations ponctuelles dans le génome sur des séquences non-codantes. En effet, une question d'actualité (Raychaudhuri 2011) consiste justement à évaluer le potentiel délétère de mutations de ce genre. Grâce aux avancées dans le domaine du séquençage d'ADN, on identifie un grand nombre de mutations. Il est difficile

d'identifier le sous-ensemble de celles ayant des impacts biologiques. Les mutations qui changent les séquences de protéines sont des candidats flagrants mais pour les autres mutations (et elles sont nombreuses), la question est ouverte. Nous avons classées les mutations connues affectant les précurseurs de microARN en fonction de leurs effets sur la dynamique. Les mutations qui affectent le plus la dynamique des microARN sont de bon candidats à des anomalies de maturation comme l'est la mutation naturelle de miR-125a et qui est presque exclusivement présente chez des personnes affectées par le cancer du sein.

3 Structures locales

Ce chapitre est divisé en cinq parties :

1. Introduction au chapitre
2. Approche algorithmique à la découverte de structures dynamiques locales par balayage
3. L'article 'Structural messenger RNA contains cytokeratin polymerization and depolymerization signals.'
4. Un module structural conservé dans les ARN de neuroligines?
5. Conclusion au chapitre

3.1 Introduction au chapitre 3

Nous avons vu au chapitre 1 un outil pour générer rapidement des ensembles de prédictions de structures 2D en utilisant le modèle des NCM. Puis au chapitre 2 que les ARN sont dynamiques et qu'en comparant leurs ensembles de structures 2D sous-optimales nous pouvons mesurer l'homologie fonctionnelle de deux séquences d'ARN et ce sans comparer directement leurs séquences. Mais en plus d'être dynamiques, les structures d'ARN sont d'au moins deux types que nous introduisons ici: elles peuvent être *locales* ou *globales*.

Lorsque l'ensemble de la séquence d'ARN présente une structure définie, nous parlerons de structure globale. Ceci est le cas par exemple des ARN de transfert et des ARN ribosomiques. Tous les nucléotides de ces molécules participent à la structure. D'un autre côté, plusieurs structures d'ARN sont locales. Elles n'impliquent pas tous les nucléotides de la molécule mais seulement ceux du segment concerné. Les transcrits primaires des microARN sont de bons exemples de structures locales. En effet la molécule peut atteindre des milliers de nucléotides de long mais la structure caractéristique en tige boucle ne fait qu'une centaine de nucléotides.

La détection de structures dynamique locales (SDL) a un certain nombre d'applications pratiques d'intérêt pour le biologiste moléculaire. Deux de ces questions nous intéressent en particulier : Peut-on identifier une SDL le long d'une séquence? Et étant donné plusieurs séquences, peut-on identifier une signature récurrente?

Pour la première de ces questions, nous avons joué un jeu de devinettes avec la grenouille. On nous a demandé d'identifier un segment de structure locale dans l'ARNm du gene *vegt* de *Xenopus laevis*. Bien que l'ARNm de *vegt* encode pour une protéine, des évidences expérimentales indiquent que indépendamment de la protéine, la molécule d'ARNm participe à la polarisation des ovocytes de *Xenopus* spp. (des grenouilles) lors de la première division de l'œuf. Pour cela, nous avons conçu une approche de balayage au long de la molécule à la recherche de segments dont la dynamique de structure est insensible aux variations causées par les bouts de séquences. Ce travail a donné lieu à la

publication d'un article que nous présentons dans son intégralité mais en préambule, nous introduirons tout de même l'approche algorithmique employée.

La seconde question consiste à savoir comment identifier des segments de structures homologues entre plusieurs séquences? Les ARNm des gènes de neuroligines (*Nlgn*) dans certains neurones de la souris (les neurones pyramidales) continuent d'être traduits efficacement en protéines même lorsque le facteur normalement essentiel eIF4E est artificiellement réduit en abondance alors que cela n'est pas le cas pour de nombreux autres ARNm. Une hypothèse veut qu'une structure dynamique locale au début des ARNm soit responsable de cet effet. Or la structure pourrait être en différentes positions et en différentes cardinalité dans chaque gène. Nous avons utilisé une méthode de groupage pour vérifier la plausibilité de l'hypothèse et ces travaux ont fait partie d'un article. Nous ne présentons pas l'article complet ici mais seulement notre approche et nos résultats.

3.2 Découverte de structures dynamiques locales par balayage

Les signatures englobant une SDL sont homologues dans le segment de la SDL impliquant que leurs similarités soient bornées par le haut. En éliminant les coûts d'insertion en bouts de séquences, pour une séquence α :

$$\exists\, SDL_{p,q} \rightarrow \mathfrak{D}^{SIM}\left(\sigma^{\alpha[s,u]}, \sigma^{\alpha[t,v]}\right) \leq \mathfrak{D}^{SIM}\left(\sigma^{\alpha[t,u]}, \sigma^{\alpha[t,u]}\right); avec\ s \leq t \leq p < q \leq u \leq v$$

Prenons le cas d'une SDL de longueur $l = 90$. Les 11 signatures l'englobant de longueur $\omega = 100$ partagent le signal de la SDL et ce signal domine l'homologie entre les paires de signatures.

On peut utiliser ce fait pour localiser des SDL le long d'ARN (α). Fixons ω, calculons les $\sigma^{\alpha[i,i+\omega-1]}\forall_{1\leq i\leq n-\omega}$ et leurs $\mathfrak{D}^{SIM}$ de proche en proche pour un pas $p \ll \omega$. En présence de SDL de longueur $l \leq \omega - p$, débutant dans une position $\geq i + p$, la valeur de $\mathfrak{D}^{SIM}\left(\sigma^{\alpha[i,\dots]}, \sigma^{\alpha[i+p,\dots]}\right)$ est bornée par le haut. Ainsi, cherchons les valeurs de ω, de p et de positions où des suites consécutives de $\mathfrak{D}^{SIM}$ sont maximales.

Cell Tissue Res (2011) 346:209-222

DOI 10.1007/s00441-011-1255-x

3.3 Structural messenger RNA contains cytokeratin polymerization and depolymerization signals

Malgorzata Kloc, Paul Dallaire, Arkadiy Reunov and François Major

Received: 5 August 2011 / Accepted: 16 September 2011 / Published online: 11 October 2011

M. Kloc was supported by NSF grant 0904186. The High Resolution Electron Microscopy Facility at UTMDACC was supported by an Institutional Core Grant (no. CA16672).

M. Kloc (*)

Department of Surgery, The Methodist Hospital and The Methodist Hospital Research Institute, 6565 Fannin Street, Houston, TX 77030, USA

M. Kloc The University of Texas M D Anderson Cancer Center, Houston, TX, USA

P. Dallaire : F. Major Institute for Research in Immunology and Cancer, Department of Computer Science and Operations Research, Université de Montréal, Montréal, Canada

A. Reunov Department of Embryology, A.V. Zhirmunsky Institute of Marine Biology, Vladivostok, Russia

3.3.1 Abstract

We have previously shown that VegT mRNA plays a structural (translation-independent) role in the organization of the cytokeratin cytoskeleton in Xenopus oocytes. The depletion of VegT mRNA causes the fragmentation of the cytokeratin network in the vegetal cortex of Xenopus oocytes. This effect can be rescued by the injection of synthetic VegT RNA into the oocyte. Here, we show that the structural function of VegT mRNA in Xenopus oocyte depends on its combinatory signals for the induction or facilitation and for the maintenance of the depolymerization vs. polymerization status of cytokeratin filaments and that the 300-nucleotide fragment of VegT RNA isolated from the context of the entire molecule induces and maintains the depolymerization of cytokeratin filaments when injected into Xenopus oocytes. A computational analysis of three homologous Xenopus VegT mRNAs has revealed the presence, within this 300-nucleotide region, of a conserved base-pairing (hairpin) configuration that might function in RNA/protein interactions.

3.3.2 Keywords

Structural RNA . Cytokeratin . Oocyte . VegT mRNA . Xenopus

3.3.3 Introduction

Our previous studies have established a new paradigm for the function of RNA. We have demonstrated a novel structural role of coding and noncoding localized RNAs in the

organization and maintenance of the cytokeratin cytoskeleton in Xenopus oocytes (Kloc, Wilk et al. 2005; Kloc, Bilinski et al. 2007; Kloc 2008; Kloc 2009). A structural role for various RNAs has also been found in dividing Xenopus and HeLa cells and in Drosophila embryos (Blower, Nachury et al. 2005; Jenny, Hachet et al. 2006; Blower, Feric et al. 2007; Lecuyer, Yoshida et al. 2007). These data indicate that the structural function of RNA in the organization of the cytoskeleton is probably a widespread phenomenon.

Localized non-coding Xlsirts and coding VegT RNAs have both previously been shown (Kloc, Spohr et al. 1993; Kloc and Etkin 1994; Zhang and King 1996; Heasman, Wessely et al. 2001) to play a structural role in the maintenance and organization of the cytokeratin network in the vegetal cortex of Xenopus oocytes (Kloc, Wilk et al. 2005; Kloc, Bilinski et al. 2007). Removal of VegT mRNA (whose protein product func- tions as a maternal regulator of endoderm initiation in Xenopus embryo) causes fragmentation of the cytokeratin network and the removal of Xlsirts RNA causes a collapse of the multidimensional cytokeratin network (Kloc and Etkin 2005; Kloc, Bilinski et al. 2007). The fragmentation of the cytokeratin network, in turn, results in the release of Vg1 mRNA, another vegetally localized RNA and germinal granules from their cytokeratin anchor (Elinson, King et al. 1993; Alarcon and Elinson 2001; Heasman, Wessely et al. 2001; Kloc, Dougherty et al. 2002; Kloc and Etkin 2005; Kloc, Bilinski et al. 2007; Kloc 2008; Kloc 2009). Further studies by using morpholino methodology, which blocks translation and leaves RNA intact, have revealed that the structural function of VegT mRNA is translation-independent (Heasman, Wessely et al. 2001; Kloc, Wilk et al. 2005). In addition, by using fluorescent molecular beacons applied as an RNA marker (Bratu, Cha et al. 2003), we have shown that Xlsirts and VegT RNA are integrated into the cytokeratin network (Kloc, Wilk et al. 2005; Kloc 2008; Kloc 2009). We have also demonstrated that the synthetic VegT RNA injected into VegT RNA-depleted oocytes is not only able to rescue the cytokeratin network fragmentation phenotype and reconstitute the normal cytokeratin network (Kloc, Wilk et al. 2005) but also to accelerate cytokeratin polymerization in vivo and polymerize cytokeratin filaments in vitro (Kloc, Foreman et al. 2011).

These data raise the question of whether the entire molecule or just certain fragment(s) of the VegT mRNA are necessary for maintaining the organization of the

cytoker- atin network in the Xenopus oocyte. Here, we show that a single 300-nucleotide (nt) fragment of VegT mRNA plays a role in the depolymerization of the cytokeratin network in Xenopus oocytes and that the whole VegT RNA molecule contains combinatory signals for polymerization/depoly- merization of cytokeratin filaments. A TopoScan analysis of three homologous Xenopus VegT mRNAs has revealed the presence, within this 300-nt region, of a conserved base- pairing (hairpin) configuration that might function in RNA/ protein interactions.

3.3.4 Materials and methods

3.3.4.1 Animals, oocytes and eggs

Xenopus laevis wild-type oocyte-positive females (Nasco) were anesthetized and pieces of ovary were surgically removed and placed in oocyte medium (OCM: 480 ml Liebovitz L-15 medium, 320 ml sterile deionized water, 0.32 g bovine serum albumin [BSA], 4 ml glutamine from 200 mM stock, 4 ml penicillin-streptomycin solution, pH adjusted to 7.6–7.8 with 5 M NaOH).

Stage VI oocytes were defolliculated manually and incubated in the same medium. Matured oocytes were either acquired by spawning or by in vitro maturation with progesterone. To induce ovulation, female frogs were injected with 800 U human chorionic gonadotropin. On the next day, the ovulated eggs were dejellied in 2% L-cysteine in TRIS buffer pH 7.8. For in vitro maturation, defolliculated oocytes were incubated overnight in 2 μM progesterone in OCM.

3.3.4.2 Oocyte injections

The vegetal pole of manually defolliculated stage VI oocytes or naturally or in vitro (with progesterone) matured oocytes (see above) was injected with 10 ng of various RNAs in 10 nl 1× Barth solution consisting of 88 mM NaCl, 1 mM KCl, 2.4 mM NaHCO3, 0.82 mM MgSO4, 0.33 mM Ca(NO3)2, 0.41 mM CaCl2 and 20 mM HEPES-TRIS, pH 7.5 (Kloc and Etkin 1994). Immediately after the injection, the oocytes were placed and incubated in OCM (Kloc, Wilk et al. 2005).

3.3.4.3 Light and electron microscopy cytokeratin immunostaining

Oocytes were fixed in 1% formalin in 100% methanol overnight at −20°C. The blocking, antibody incubation and washing steps were performed exactly as described in (Bilinski, Jaglarz et al. 2010). Light microscopy cytokeratin immunostaining was performed by using fluorescein isothiocyanate (FITC)- conjugated C11 anti-pan cytokeratin antibody (cat. no. F 3418, Sigma) at 1:400 dilution. The oocyte vegetal cortexes were observed under a Nikon fluorescence microscope. Electron microscopy immunostaining was performed as described in detail in (Bilinski, Jaglarz et al. 2010).

3.3.4.4

3.3.4.5 RNA constructs and RNAs

Synthetic VegT RNA (open reading frame + 20-nt 5′ untranslated region), which is known to rescue antisense VegT depletion, was prepared by using a Megascript kit (Ambion) as described previously (Kloc, Wilk et al. 2005). In addition, RNAs were synthesized, by using the Megascript kit, from pBluescriptII SKII + constructs containing frag- ments of VegT cDNA (see Results).

3.3.4.6 Functional and structural analysis

The presence of functional RNA structure in homologous mRNAs was detected by first predicting the optimal and suboptimal secondary structures of 80-nt windows by using MC-Fold (Parisien and Major 2008), from which conserve

d base-pairing configurations and their signals were computed (TopoScan program is available upon request from F. Major). The TopoScan signals revealed the presence of RNA function. The three Xenopus mRNAs were aligned by using lsearch36 from fasta36 alignement software (Pearson 2000). These alignements were mapped to the TopoScan signals of X. leavis. In the case of a peak signal, the consensus structure for each sequence was identified by MC-Cons (Parisien and Major 2008).

```
   1  atattgagca ttcacctgca ggggcccatg tgatagccta tagctacagt cgtggaatga
  61  gaaactgctg tcgggaatgc ggcctctctg cggggcatct ggaaccagag gcttcttcta
 121  actgtgcatc agatgtaaag tcttccccag acatggacag tgtctccagc caagactccc
 181  tctacctgcc caacactgtt ggtgcctccc tggaagacca agatctatgg tcccagttcc
 241  accaggaggg gacagagatg atcatcacca aatctggaag gaggatgttt cctcagtgta
 301  agatccgtct cttcggtctt catccttatg ccaagtacat gctgctggtg gactttgttc
 361  ctctggacaa ctttaggtat aagtggaata agaaccagtg ggaagcagca ggtaaagcag
 421  aacctcaccc accctgcagg acgtatgtcc acccagattc acctgctcct ggtgcccact
 481  ggatgaagga tccgatctgc tttcaaaagc tcaaactcac caacaacaca ttggatcaac
 541  aaggccatat tatcttgcat tcaatgcatc gctacaagcc caggttccat gtagttcagt
 601  ctgatgacat gtacaattct ccatggggat tggtacaagt gtttagcttc ccagagacag
 661  agtttacttc agtgactgcc taccagaatg aaaagattac taaactgaaa attaatcaca
 721  acccatttgc taaaggattc cgggagcagg aaaggagtca caagagggat gatgttttaa
 781  agattctaca acaaagtcct agtaaaaggc agaagaggaa gaagtgggag gacagtcctg
 841  aggctgatat ttcagatttc cccaaggcta tatgtgtgaa ggaggaatcc attatggacc
 901  cagcaggagt ttatcagaac tgggtttcag atcacgaggc taaccaaggc ttgacacccc
 961  actcccctga gtctgagggt gccaatcagg agcagcaagt ccccacatct tcctctaact
1021  tctacaacaa gagccattat cgaaggagtt cccaacatct ctcctcgcca tttgaattgg
1081  gagagccctc tagcagacgt cttacccctg acattgctac agtgccggat tccgatccag
1141  attctttagc agtgtttcat gtcattccaa cacagaattc tgctccagaa cgcacatgtt
1201  ctatgaactt ttccatggaa gctccaatga aacagcccct ccggggtgcc atgtacagcc
1261  cttatggagc agatcagtgg ttggttccag ctcaaggtca atatcgacct gtgggctata
1321  ctgcataccc aacagactta agcacacaag gagcagtagc tcacccacat agtgcgatgt
1381  cagactggag ccaatactct ctcttcccct acagctgttg gtaaatgggt taagggaaat
1441  gtgtatccac agtccttacc catccatgcc aatacctcag tcctgatttc ttgttggggg
1501  ggggggggagc gggctgtgtg gttttgcatc tctgaggcaa aacctcagga cctccttaaa
1561  aaccctgcat tgggaggttg catggagagg tttgattaca atatttgttg gactaaaagg
1621  tgtcacttta tcagcacaaa cagtaggttg cccctgtgct tgtgatcagg ggaatctcta
1681  cctgtacttg aactaaatgt atttatttta aagattagca agtaaatgtg agaaaccgtg
1741  gggggtgcg ttaaagcaga tctccagaag agaacccggg aaagcagtta tggatttata
1801  gaacaaagta atggggccat gaacagaagc agttttctct tgactttaag tggtggtaca
1861  gccatctaac tctccatttt cttaatgtcc ccaattattg cctttcactc aattatacta
1921  tggtctgttt tgtttttgtt tttttaaaac tataattttg attacagttt tgcaccttac
1981  attttccagg gtggtggttt gcagcactaa tctgtgacag tgtgtacagc aggtgtttgt
2041  gcttttgtgc actgctcggg aaatgtgtgt gtatttgggg gggagtgtgt gttcactaaa
2101  gattgtactg ctgactatga gactaacgct gtgcaatatt cacagggata ggcaaaattt
2161  ggtgttccat ctaaagcaaa gctacagctt ccagtattgc ctctagatga gagcgctact
2221  tctgcaatag cagggtcacc aaaggctgcc taactctgca tattttcaca ttagcataac
2281  tctggcaatg attgggacaa ttgactccaa aagtggggta aacatgtaac aaatattaat
2341  ccttgcattc tgtgattagt taaataaaca tgggatggtt tgtttgctct ttgttctaca
2401  gattcagtag cagctcagca gttaaagggt ttgtgtaatt gcttgtaaag ctaggttttg
2461  tgggagggct gcaggcagaa cagtatgtat tctgtataga acacatctgt aaatagattt
2521  tatttaagaa ttgtttttga ctatgaaatg tgttttaatg tgacttttat ttaatggcat
2581  gaactgagga aggaaacaat gaagtgtcat ttgttttgtc tccatctggt tatacacctt
2641  caataaatgt tttccttatt tcaaaaaaaa aaaaaaaaaa aaaaaaaaaa a
```

Figure 15. (fig 1) Sequences of VefT mRNA fragments.

The sequence of nine fragments (shown in alternating regular and bold type) of VegT mRNA used. The sequence of fragment no. 8 is marked in red. The short sequence in blue indicates the sequence that was mutated for the rescue experiments to prevent binding to anti-sense deoxynucleotides used in the original studies (Kloc et al. 2005; Heasman et al. 2001).

3.3.5 Results

We were interested in determining whether the structural role of VegT mRNA in the organization of the cytokeratin network depended on the entire molecule or its fragments.

A 300-nt fragment of exogenous VegT RNA induces or facilitates depolymerization of the cytokeratin network in Xenopus oocytes We divided the entire molecule into nine 300-bp fragments (Fig. 1). In the first set of experiments, we injected the synthetic RNAs transcribed from these fragments into the vegetal pole of stage VI oocytes. Between 100 and 150 oocytes from two to five different frogs were injected with each RNA. After overnight incubation, we fixed the oocytes and stained the cytokeratin network by using a FITC- conjugated anti-pan cytokeratin antibody.

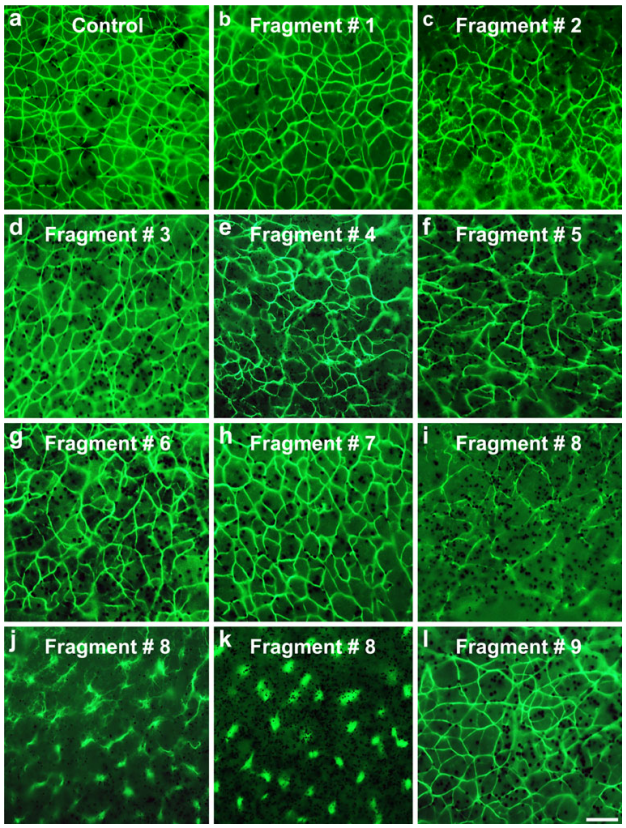

Figure 16. (fig 2) Effect of injection of nine fragments of VegT RNA on the cytokeratin network in the Xenopus oocyte.

a–h, l Injection of fragment nos. 1-7 and fragment no. 9 had no visible effect on the morphology of cytokeratin network. i–k The injection of fragment no. 8 caused (in 30% of oocytes) profound changes in the polymerization status of the network: loose network (i), “long comets” (j) and foci (k). Bar 16 μm.

We found that the injection of fragment nos. 1–7 and fragment no. 9 had no visible effect on the cytokeratin network (Fig. 2). However, the injection of fragment no. 8 had a profound effect on the appearance of the cytokeratin network in approximately 30% of the oocytes (Figs. 2, 3). In some oocytes, the cytokeratin network was depolymer- ized and cytokeratin was visible as distinct foci (Figs. 2, 3). These foci appeared identical to the foci observed in mature oocytes (eggs, see below) and in oocytes injected with the anti-cytokeratin antibody (compare with Figs. 2a, 3b, 3d in (Kloc, Wilk et al. 2005)). Some oocytes injected with fragment no. 8 were in different "phases" of cytokeratin depolymerization/ polymerization and showed cytokeratin in the form of "long comets" or a "loose network" (Figs. 2, 3). In addition, some oocytes contained a much thicker "hyper-polymerized" network (Fig. 3).

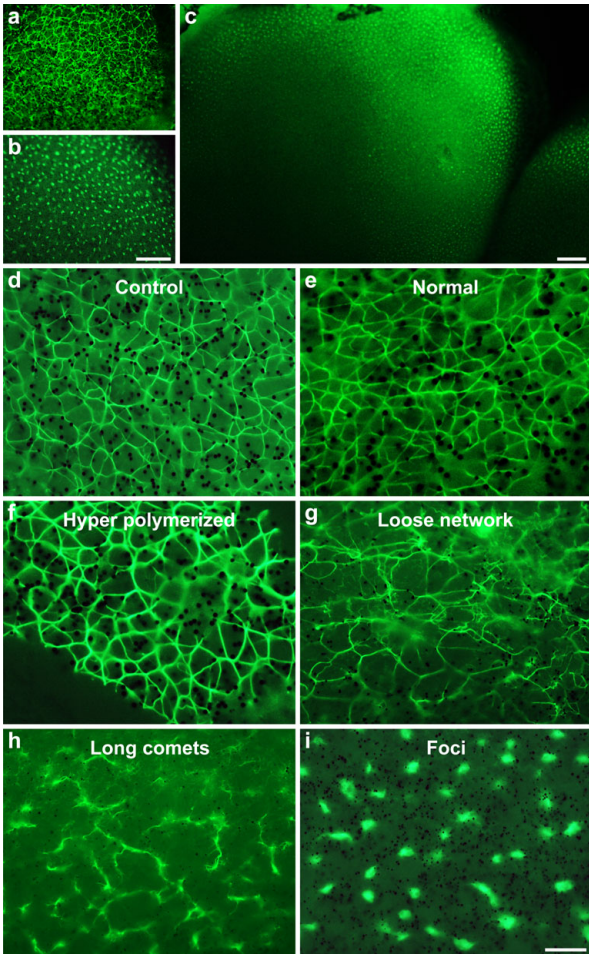

Figure17. (fig 3) Effect of the injection of frament no. 8 of VegT RNA on the cytokeratin network in the oocyte vegetal cortex.

Oocytes immunostained with fluorescein isothiocyanate (FITC)-conjugat- ed anti-pan cytokeratin antibody. a Low magnification image of the vegetal pole surface of con- trol oocyte showing an intricate cytokeratin network. b Similar view of the oocyte injected with VegT RNA fragment no. 8 showing distinct cytokeratin foci instead. c Lower magnification of the vegetal pole surface of two oocytes injected with VegT RNA fragment no. 8 showing distinct cytokeratin foci. d Cytokeratin network in a control oocyte. e–i Examples of grada- tion of the effect of injection of VegT RNA fragment no. 8 on the appearance of the cytokera- tin network. e A total of 150 oocytes were injected. A normal-appearing network was present in 101 (67.3%) oocytes. f A "hyper polymerized" net- work was present in 5 (3.3%) oocytes. g A "loose network" with a large mesh was present in 5 (3.3%) oocytes. h "Long comets" of a partially depoly- merized network were present in 10 (6.6%) oocytes. i Cytokeratin foci were present in 29 (19.3%) oocytes. Bars 40 µm (a, b),83 µm (c), 16 µm (d–i).

Electron microscopy analysis of nanogold immunos- tained samples confirmed a dramatic depolymerizing effect of the injection of fragment no. 8 of VegT RNA on the morphology of the cytokeratin network (Fig. 4). These data indicated that a 300-nt

fragment of VegT RNA was able to induce or facilitate the depolymerization of cytokeratin filaments in the vegetal cortex of Xenopus oocytes.

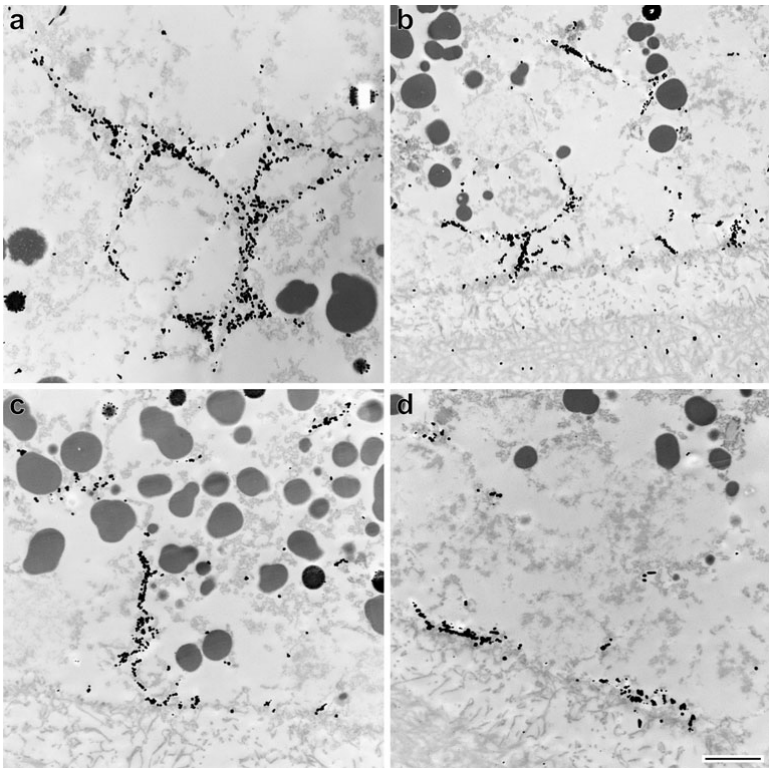

Figure18. (fig 4) Effect of the injection of the fragmetn no. 8 on the ultrastructure of cytokeratinin hte oocyte vegetal cortex.

Electron microscopy images of fragments of the vegetal cortex of stage VI oocytes. Cytokeratin was stained with anti-cytokeratin antibody and nanogold-conjugated secondary antibody and silver- enhanced. a Cytokeratin net- work in a control oocyte. b, c Various stages of cytokeratin depolymerization in oocytes injected with fragment no. 8 of VegT RNA: “loose network” (b), “long comets” (c) and foci (d). Bar 2 μm

A 300-nt fragment of exogenous VegT RNA prevents recon- stitution of the cytokeratin network in activated eggs During in vivo or in vitro (after incubation with progesterone) maturation of Xenopus oocytes, the cytokeratin network gradually depolymerizes, cytokeratin filaments become shorter and finally, in fully matured oocytes (eggs), the cytokeratin is visible as distinct foci (Figs. 5, 6, 7). However, upon activation by sperm or by pricking with a needle, matured oocytes reconstitute the cytokeratin net- work,

although the morphology of the network is slightly different than in stage VI oocytes (Klymkowsky and Maynell 1989; Klymkowsky, Maynell et al. 1991; Clarke and Allan 2003; Kloc, Wilk et al. 2005).

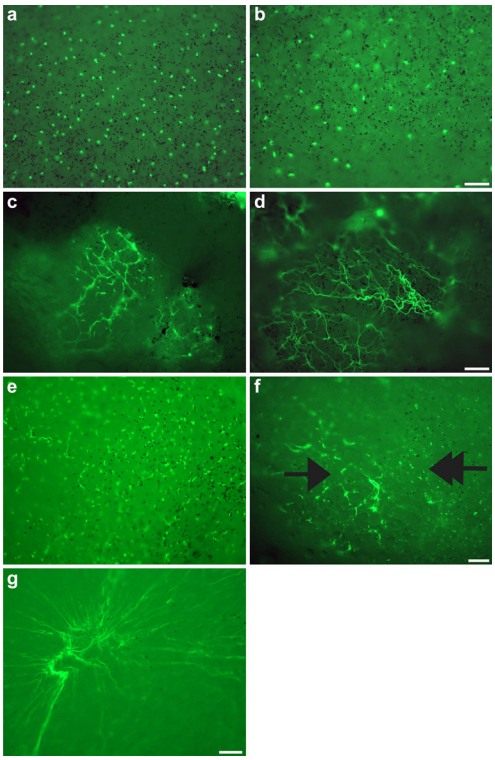

Figure 19. (fig 5) Fragment no. 8 of VegT RNA injected into ovulated eggs (naturally matured oocytes) prevents reconstitution of the cytokeratin network upon egg activation.

Ovulated eggs immunostained with a FITC- conjugated anti-pan cytokeratin antibody. a, b Images of the vegetal pole of a control ovulated egg (mature oocyte) showing distinct foci of cyto- keratin at 0 h (a) and after 24 h incubation (b). c Ovulated egg activated by pricking and subsequently incubated for 24 h: note reconstitution of the cytokeratin network. d Ovulated egg activated by injection with VegT RNA fragment no. 5 and incubated for 24 h; note recon- stitution of the cytokeratin network. e–g Ovulated eggs activated by injection with VegT RNA fragment no. 8 and subsequently incubated for 24 h; note lack of reconstitution of the cytokeratin network. Cytokera- tin is visible as differently sized foci (e), short filaments (arrow) in the vicinity of foci (double arrow) (f), or long filaments (g). Bars 16 μm.

We injected VegT RNA fragment no. 8 into naturally matured (ovulated) eggs. After incubation for 24 h, we examined the appearance of the cytokeratin by immunostaining with a FITC-conjugated anti-pan cytokeratin antibody (Fig. 5). In control

noninjected eggs, cytokeratin was visible (at 0 h and after 24 h of incubation) as distinct foci (Fig. 5a, b). The eggs activated by pricking or by injection with control RNA (VegT RNA fragment no. 5, which we know does not have any effect on the cytokeratin network when injected into stage VI oocytes; see above) reconstituted the cytokeratin network after a 24-h incubation (Fig. 5c, d). In contrast, the eggs activated by the injection of VegT RNA fragment no. 8 were unable to reconstitute the cytokeratin network. In these activated eggs, cytokeratin was visible as foci and filaments of various lengths (Fig. 5e–g). These results indicate that fragment no. 8 of VegT RNA is able to keep cytokeratin in the depolymerized state.

Eggs derived from the oocytes injected with a 300-nt fragment of VegT RNA reconstitute the cytokeratin network upon activation In another set of experiments, we injected

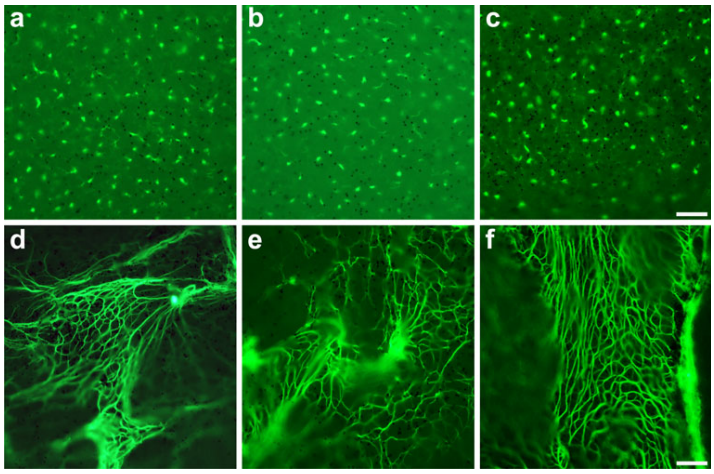

Figure 20. (fig 6) Oocytes injected with fragment no. 8 of VegT RNA are able to reconstitute the cytokeratin network upon maturation and activation.

Images show fragments of oocyte vegetal cortex. Oocytes immunos- tained with FITC-conjugated anti-pan cytokeratin antibody. a Control oocyte incubated for 24 h and subsequently matured with progester- one for 24 h; cytokeratin is visible as distinct foci. b Oocyte injected with VegT RNA fragment no. 5, incubated for 24 h and subsequently matured with progesterone for 24 h; cytokeratin is visible as distinct foci. c Oocyte injected with VegT RNA fragment no. 8, incubated for 24 h and subsequently matured with progesterone for 24 h; cytokeratin is visible as distinct foci. d Control oocyte incubated for 24 h, matured for 24 h, subsequently activated by pricking and then incubated for another 24 h. Note reconstitution of the cytokeratin network. e, f Oocytes injected with VegT RNA fragment no. 5 or fragment no. 8, respectively, incubated for 24 h, matured for 24 hr, subsequently activated by pricking and then incubated for another 24 h. Note reconstitution of the cytokeratin network. Bars 16 μm.

RNA into stage VI oocytes. After a 24-h incubation, the oocytes were matured in vitro with progesterone for 24 h, activated by pricking with a needle and incubated for an additional 24 h (total incubation time of 72 h; Figs. 6, 7). In control oocytes incubated for 24 h and then matured with progesterone for 24 h, cytokeratin was visible as distinct foci (Fig. 6a). When such eggs where activated by pricking and incubated for another 24 h, they reconstituted the cytokeratin network (Fig. 6d). Similarly, the cytokeratin formed distinct

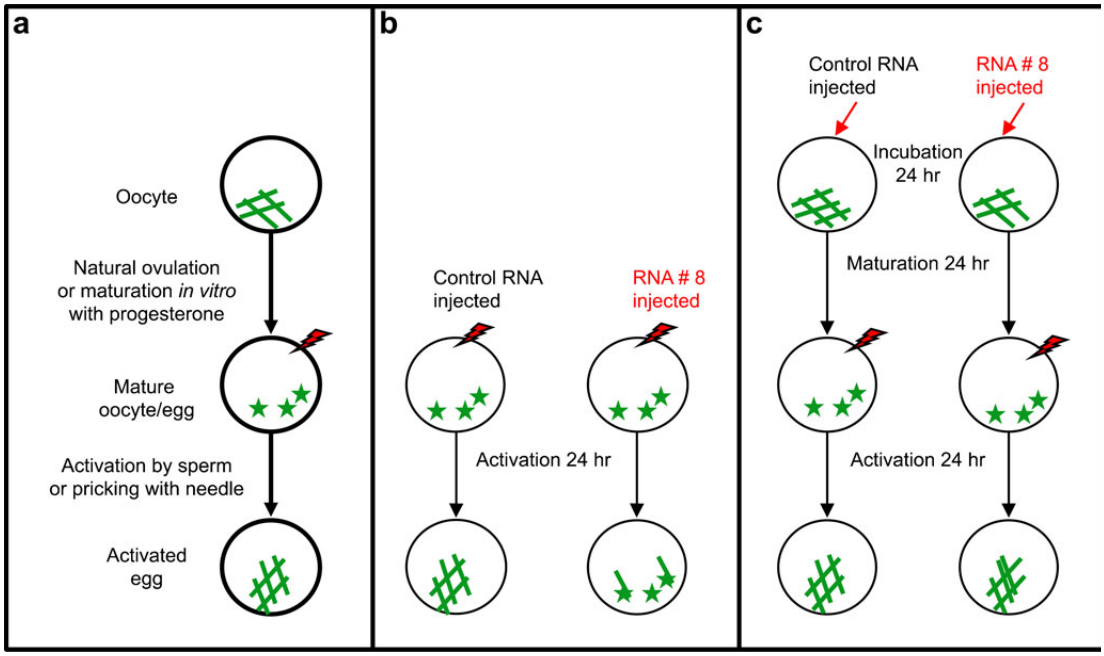

Figure 21. (fig 7) Summary of the effect of injection of VegT RNA fragment no. 8 (RNA #8) into oocytes and eggs on hte appearance of cytokeratin.

a Cytokeratin changes occurring during normal oocyte maturation and activation; the cytokeratin network present in the stage VI oocytes is replaced by cytokeratin foci in naturally matured (ovulated) or in vitro matured (by progesterone) oocytes. The cytokeratin network is reconstituted in eggs activated by sperm or by pricking. The appearance of the network in stage VI oocytes and activated eggs differs slightly (Kloc et al. 2005). **b** Cytokeratin changes in injection experiments with naturally matured (ovulated) eggs is recapitulated. When mature oocytes (eggs) are activated either by pricking or injection of control RNA, the cytokeratin network is reconstituted. Eggs injected with VegT RNA fragment no. 8 are unable to reconstitute the cytokeratin network; cytokeratin remains in the form of foci and short filaments. **c** Cytokeratin changes in experiments in which RNA is injected into stage VI oocytes that are subsequently matured and activated by pricking. After prolonged incubation (72 h), control oocytes and oocytes injected with VegT RNA fragment no. 8 reconstitute the cytokeratin network.

foci in oocytes that were injected with control VegT RNA fragment no. 5 or with VegT RNA fragment no. 8, incubated for 24 h and then matured with progesterone for another 24 h (Fig. 6b, c). When such eggs were activated by pricking and incubated for 24 h, they all reconstituted the cytokeratin network (Fig. 6e, f). These results indicate that the ability of VegT RNA fragment no. 8 to keep cytokeratin in the depolymerized state wanes with time (Fig.7; see Discussion for a further explanation of this result).

In summary, all these experiments indicate that the 300-nt- long fragment of VegT mRNA is able to influence the polymerization status of cytokeratin filaments.

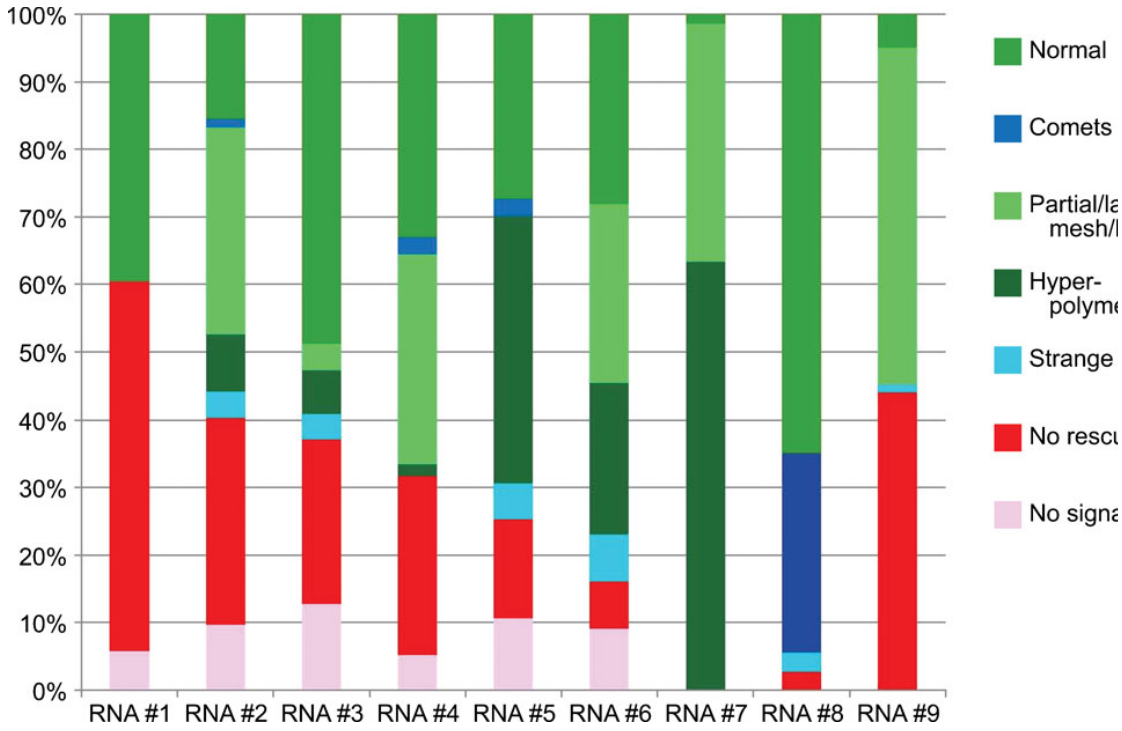

Figure 22. (fig 8) Summary of rescuing activity of the nine fragments (#1-#9) of VegT RNA.

Stage VI oocytes were pre-injected with VegT antisense deoxyoligonucleotides to remove endogenous VegT mRN A; 24 h later the rescuing RNAs were injected and, after an overnight incubation, oocytes were stained with FITC-conjugated anti-pan cytokeratin antibody and scored for the morphology of the cytokeratin network in the vegetal cortex.

Rescuing ability of 300-nt-long fragments In the next set of experiments all nine 300-nt-long RNA fragments (nos. 1–9; described above) were tested for their ability to rescue the cytokeratin network in stage VI oocytes pre-injected (24 h earlier) with VegT-anti sense deoxynucleotides. The results of these rescue experiments are shown in Table 1, Fig. 8. The data indicate that, although the most “rescuing or cytokeratin polymerization activity” seems to be contained within fragment no. 7, none of the fragments when isolated from the molecular context of the entire molecule reconstitute a normal-looking cytokeratin network. This, in turn, indicates that the structural function of VegT mRNA relies on the combinatory signals for depolymerization and polymerization of cytokeratin filaments.

Structural analysis of VegT mRNA In order to determine whether the VegT mRNA contains any particular structural motives, which may function in RNA/protein interaction, we performed structural analysis of VegT mRNA from three different Xenopus species (X. laevis, X. tropicalis, and X. borealis).

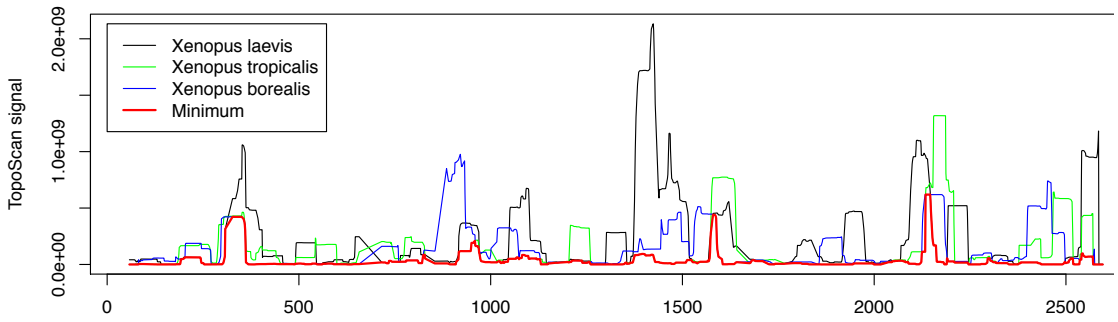

Figure 23. (fig 9) Conserved structure in the three Xenopus VegT mRNAs.

The smoothed TopoScan signal generated by using 80-nt windows is shown for three Xenopus mRNAs (red line minimum of the three signals). Peaks can be observed in the coding sequence and at two sites in the 3′ untranslated region. The largest of these starts near nucleotide (nt) 2100.

The TopoScan analysis (see Materials and methods) of the three homologous Xenopus VegT mRNAs revealed the presence of a conserved base-pairing configuration in the 2110–nt to 2170-nt region (Fig. 9), which was located within the described above fragment no. 8. An analysis of the peak signals indicated eight possible regions of structure-function conservation (Fig. 10). For each peak, the consensus structure of the aligned sequences in each species were computed by MC-Cons. Among the eight structure-function hypotheses, only two hypothetical hairpins were detected in overlapping windows: nt 2095–2144 and nt 2111–2184 (Figs. 11, 12). These results suggest that the conserved hairpin configuration is responsible for the functional interaction of VegT RNA with the cytokeratin.

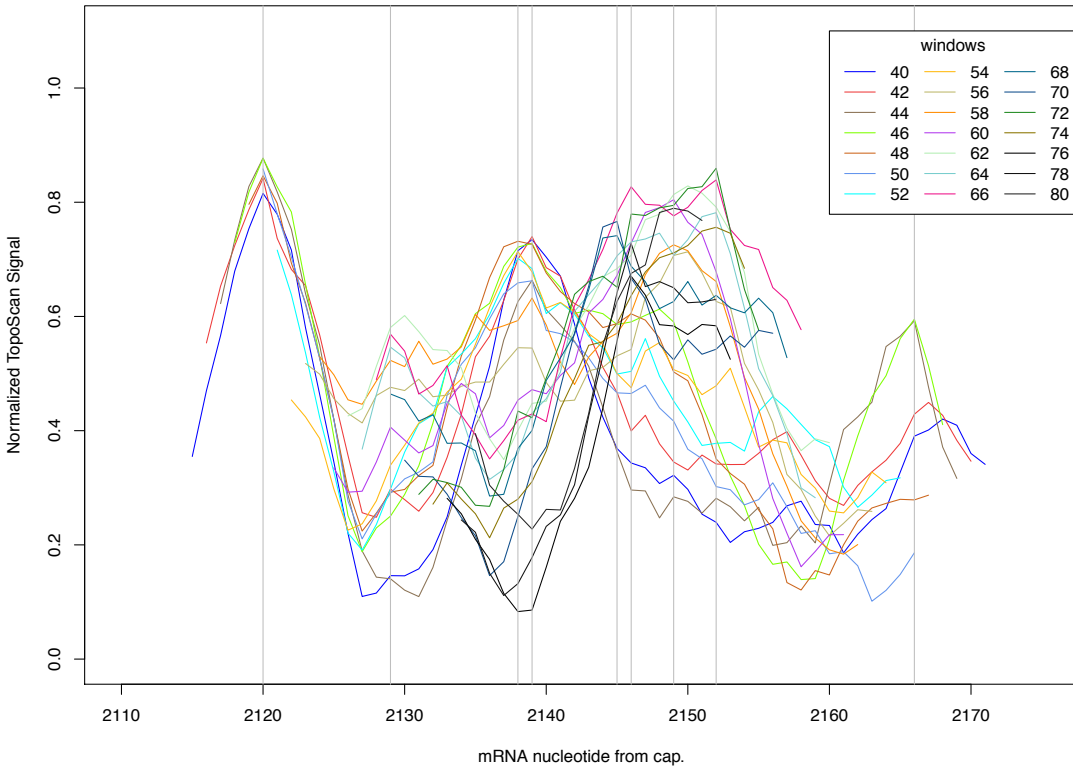

Figure 24. (fig 10) Largest peak region of VegT mRNAs.

A zoom of the largest peak region, showing numerous MC-Fold window lengths (gray lines local maxima)

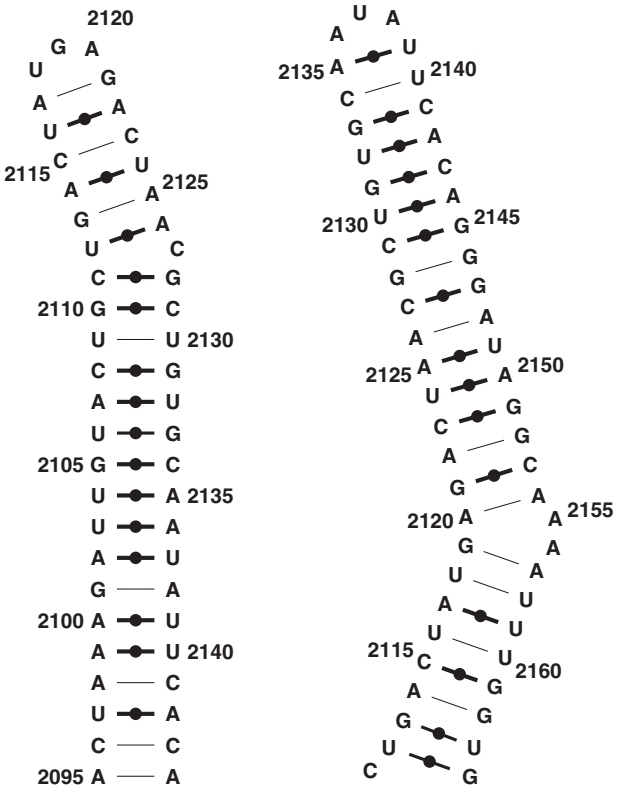

Figure 25. (fig 12) Secondary structure of the two conserved hairpin hypotheses.

Left For window nt 2095–2144. Right For window nt 2111–2184.

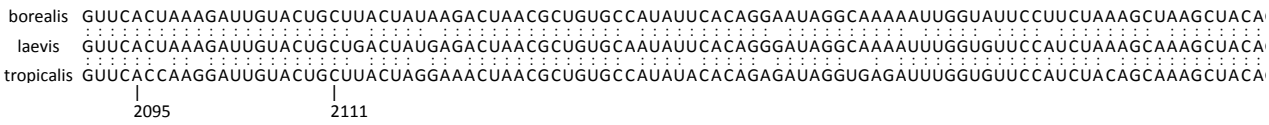

Figure 26. (fig 11) Three-sequence alignment in the largest peak region.

The alignment of VegT mRNA sequences in the largest peak region from top to bottom: X. borealis, X. laevis, and X. tropicalis.

3.3.6 Discussion

Our previous studies have shown that VegT RNA has a structural function in the maintenance and organization of the cytokeratin network at the vegetal cortex of Xenopus oocytes. Depletion of endogenous VegT mRNA with anti- sense deoxynucleotides causes the fragmentation of this network. This effect can be rescued by injection of exogenous synthetic VegT RNA (Kloc, Wilk et al. 2005; Kloc, Bilinski et al. 2007). Our earlier investigations have also demonstrated that, when injected into stage IV oocytes, which have an under- developed cytokeratin network when compared with that of stage VI oocytes, VegT RNA induces the formation of a fully developed, stage-VI-oocyte-type network. In addition, our most recent studies have shown that synthetic VegT RNA promotes the formation of cytokeratin filaments in in vitro assay and that when co-injected with cytokeratin into stage VI oocytes, it accelerates the polymerization of the cytokeratin network in the oocytes (Kloc, Foreman et al. 2011). Taken together, all these data indicate that VegT RNA promotes or facilitates cytokeratin polymerization.

Our present study has revealed that the injection of the 300-nt fragment of VegT RNA (fragment no. 8) causes dramatic changes in the cytokeratin network polymerization status quo in 30% of injected oocytes. Some oocytes exhibit complete depolymerization of the network. This phenotype appears similar to the phenotype that naturally occurs in mature oocytes and to the phenotype caused by the injection of an anti-cytokeratin antibody (Kloc, Wilk et al. 2005). Populations of oocytes showing various stages of depolymerization/polymerization of the network are also present among oocytes injected with fragment no. 8, appearing as longer comets, loose network and a hyper- polymerized (thicker) network. We believe that VegT RNA fragment no. 8 binds to cytokeratin and keeps it in an unpolymerized or partially polymerized state. The shifting balance between the polymerization and depolymerization states is observed as different forms of cytokeratin phenotypes: foci, long comets and the loose network. Because the injected oocytes also contain endogenous intact VegT mRNA that probably has both de-polymerization and pro-polymerization signals, the network is eventually reconstituted. In addition, because the normally occurring balance between polymerization and

depolymerization is disrupted, an abnormal hyper- polymerized network is formed on rare occasions. It is possible that fragment no. 8 does not directly interact with cytokeratin but interacts either with endogenous VegT mRNA, with VegT interacting proteins, or with other RNAs or titrates VegT mRNA regulatory factors. However, this scenario is highly unlikely in view of the results of in vitro experiments showing that VegT RNA promotes cytokeratin filament formation in the absence of any regulatory proteins or other RNAs (Kloc, Foreman et al. 2011).

The results of the fragment no. 8 injection experiments raises the question of why only approximately 30% of the oocytes show the depolymerized cytokeratin-phenotype, whereas the majority of the oocytes (approximately 70%) have a normal network. We believe that these results reflect the different states of network polymerization vs. depolymerization in individual oocytes. The definition of a stage VI oocyte is based on its morphological appearance and size but not all stage VI oocytes are physiologically equal. Because stage VI oocytes are destined to mature (enter meiosis), we assume that some oocytes are more advanced in their preparation for maturation than others. Indeed, several studies of the dynamics of oocyte maturation confirm our assumption. Marteil et al. (Marteil, D'Inca et al. 2010) have shown that stage VI oocytes from different frogs (and even from the same frog) treated with a low dose (10 nM) of progesterone exhibit the entire spectrum of the germinal vesicle breakdown (GVBD) response, from no GVBD to 100% GVBD. Furthermore, a comparison of the proteomes of stage VI oocytes shows a difference in protein composition between stage VI oocytes belonging to potentially “well maturing” and “poorly maturing” groups (D'Inca, Marteil et al. 2010). The cytokeratin network depolymerizes upon oocyte maturation and so we believe that, in those oocytes that are more advanced in their preparation for maturation, the cytokeratin network is labile and more likely to shift toward a depolymerized state. When such oocytes are injected with fragment no. 8 of VegT RNA, the network will depolymerize; this is corroborated by the results of our maturation experiments. Matured oocytes (eggs) are known to reconstitute the cytokeratin network upon activation with sperm or needle pricking (Klymkowsky and Maynell 1989; Klymkowsky, Maynell et al. 1991; Clarke and Allan 2003). Thus, the injection of RNA into eggs should cause the reconstitution of the network. The finding that

the eggs injected with fragment no. 8 of VegT RNA (in contrast to eggs injected with fragment no. 5) are unable to reconstitute the cytokeratin network indicates that fragment no. 8, when separated from its whole molecule context, keeps cytokeratin in the depolymerized state. We have previously demonstrated that oocytes injected with an anti-cytokeratin antibody, matured with progesterone and activated by pricking do not reform the normal cytokeratin network, suggesting that the binding of the antibody to cytokeratin prohibits the full-scale polymerization of the cytokeratin (Kloc, Wilk et al. 2005). Thus, the injection of fragment no. 8 of VegT RNA into a mature oocyte mimics the effect of the injection of the anti-cytokeratin antibody.

The data from our second maturation experiment in which RNA was injected into stage VI oocytes that were subsequently matured and activated showed that the depolymerizing effect of fragment no. 8 was not permanent. The most probable explanation for this result is that when fragment no. 8 is injected into the oocyte, which is subsequently matured, the depolymerization event caused by the injected RNA "enters" the normal cycle of cytokeratin depolymerization during maturation. Thus, when an egg receives an activation signal, all of the (naturally or experimentally) depolymerized cytokeratin enters into the polymerized state and reconstitutes the cytokeratin network.

The results of the rescue experiments indicate that, although the most "rescuing activity" seems to be contained within fragment no. 7, none of the fragments when separated from the molecular context of the entire molecule have the normal rescuing activity of the entire VegT mRNA molecule.

In summary, our data suggest that the structural function of VegT mRNA relies on combinatory signals for the depolymerization and polymerization of cytokeratin filaments.

Although further extensive studies and an experimental proof are needed, we believe that the potential "signal" for polymerization vs. depolymerization is exerted by specific conformational properties of the structural RNA and not by sequence specificity per se. Numerous studies of bacteria, fungi, plants and humans have shown that various regulatory RNAs, including riboswitches, can be quorum and temperature sensors and that

they are extremely dynamic and able to switch rapidly between various conformations (Altuvia and Wagner 2000; Helmann 2007; Bevilacqua and Russell 2008; Romby and Wagner 2008; Ray, Jia et al. 2009; Waters and Storz 2009). Interestingly, many of these regulatory RNAs belong to a general category of "localized RNAs", which, like VegT mRNA, reside in specific subcellular compartments and/or structures, such as Cajal bodies and the P bodies of somatic cells and embryos (for a review, see (Waters and Storz 2009)). The participation of RNA in the regulation of cytoskeleton polymerization status might be a remnant of the "RNA world", if RNA with its information and catalytic properties predated DNA/ protein-based life. Because RNA is able to act both as a sentinel and as an executor, the RNA-based response is probably much faster than the multi-step protein-based response. The RNA-based response, which relies on existing localized RNA, would be the preferred choice whenever an extremely rapid response is required. Thus, VegT structural RNA, which is also localized RNA, would have been a perfect player in the "bet-hedging" strategy of evolution (Dobrzynski, Bernatowicz et al. 2011). On the basis of the many discoveries of new localized RNAs with potential roles in the organization of cellular architecture and function (Lecuyer, Yoshida et al. 2007), we can speculate that, by being the progeny of the original RNA-based world, the phenotype of a cell or organism can be regarded as being binary (Kloc 2008; Kloc 2009; Kloc, Foreman et al. 2011), i.e., the same structural and regulatory functions are executed by evolutionary ancient RNA-based pathways and, in parallel, by evolutionary younger and more sophisticated but more sluggish, protein-based responses.

3.3.7 Acknowledgements

We thank Mr. Kenneth Dunner Jr for his electron microscopy work.

3.4 Un module structural conservé dans les ARN de neuroligines?

3.4.1 Introduction

À tout moment, le taux de traduction de chaque ARNm d'une cellule dépend d'un nombre de facteurs. Notre intérêt porte ici sur la compétition entre les ARNm pour une protéine clé dans le processus d'initiation de la traduction: EIF4E (normalement notée eIF-4E pour des raisons historiques). Cette protéine reconnait une modification chimique (nommée le *cap*) présente au tout début de tous les ARNm eucaryotes et cette reconnaissance induit l'agglomération de nombreux autres facteurs culminant en la mise en place du ribosome et la traduction de l'ARNm. Or, la modification artificielle de la concentration de eIF-4E dans certains neurones de souris change le taux de traduction de la plupart des ARNm et on peut construire deux ensembles de gènes selon que leur traduction est diminuée ou non lorsque eIF-4E est inhibée. Ce phénomène est lié à l'excitabilité du neurone affecté et cause des symptômes neurologiques assimilables à un syndrome de type autistique (*autism like syndrom disorder* ou *ASD*). Il est postulé que les ARNm dont l'expression n'est pas réduite sont porteurs d'une particularité qui leurs confèrent une affinité particulière pour eIF-4E. Cette structure serait présente dans la région de la tête des ARNm (la région non-traduite située en 5' ou 5'-UTR).

Nos travaux se sont bornés à établir s'il est possible qu'il existe au moins une SDL répondant aux critères et non d'établir hors de tout doute qu'il s'agit bien de l'entité responsable des effets. Ils ont été publiés dans le journal Nature en 2013 sous le titre « *Autism-related deficits via dysregulated eIF4E-dependent translational contro*l » (Gkogkas et al. 2013).

3.4.2 Le problème

Le problème se pose comme une ennuyeuse énumération combinatoire. Un ensemble de gènes positifs $\mathbb{G}^+ = \{\mathbb{g}_1^+, \mathbb{g}_2^+, \ldots, \mathbb{g}_g^+\}$ comprends au moins une instance de SDL qui est absente d'un ensemble de gènes négatifs $\mathbb{G}^- = \{\mathbb{g}_1^-, \mathbb{g}_2^-, \ldots, \mathbb{g}_h^-\}$ si

$$\exists \left(\{(w_1,l_1),(w_2,l_2),\dots,(w_g,l_g)\} \left| \forall_{1\le i<j\le g,1\le p\le g,1\le q\le h} \mathcal{D}\left(\sigma^{\mathbb{g}_i^+[w_i,l_i]}, \sigma^{\mathbb{g}_j^+[w_j,l_j]}\right) < \mathcal{D}\left(\sigma^{\mathbb{g}_p^+[w_p,l_p]}, \sigma^{\mathbb{g}_q^-[\cdot,\cdot]}\right)\right.\right)$$

C'est à dire qu'il existe un groupe de positions (les (w,l)) donnant les structures locales dans $\mathbb{G}^+$dont les signatures commençant aux positions *w* et s'étendant jusqu'aux positions *l* $\left(\sigma^{\mathbb{g}^+[w,l]}\right)$ où les distances dynamiques ($\mathcal{D}$) entre toutes les paires sont plus petites que la distance entre chacune d'elle et n'importe quelle tirée de $\mathbb{G}^-$. Notons $N = |\mathbb{g}_1^+| + |\mathbb{g}_2^+| + \cdots + |\mathbb{g}_g^+|$ la somme des longueurs des séquences dans $\mathbb{G}^+$ et $M = |\mathbb{g}_1^-| + |\mathbb{g}_2^-| + \cdots + |\mathbb{g}_h^-|$ celle dans $\mathbb{G}^-$. Il y a

$$\widehat{N} = (|\mathbb{g}_1^+|^2 - |\mathbb{g}_1^+|)/2 + (|\mathbb{g}_2^+|^2 - |\mathbb{g}_2^+|)/2 + \cdots + \left(|\mathbb{g}_g^+|^2 - |\mathbb{g}_g^+|\right)/2 \in O\left(MAX(|\mathbb{g}^+|^2)\right)$$

fenêtres possibles dans $\mathbb{G}^+$ et similairement $\widehat{M} = O\left(MAX(|\mathbb{g}^-|^2)\right)$ fenêtres possibles dans $\mathbb{G}^-$. Pour des fenêtres de longueurs fixes $\left(\omega \ll MIN\left(|\mathbb{g}^{+/-}|\right)\right)$, on a $N - \left(\mathbb{g}(\omega-1)\right) \approx N$ dans $\mathbb{G}^+$et M fenêtres dans $\mathbb{G}^-$.

Il existe $\prod_{1\le i\le g} \frac{\left(|\mathbb{g}_i^+|^2 - |\mathbb{g}_i^+|\right)}{2} \in O(MAX(|\mathbb{g}^+|)^{2g})$ solutions possibles en fenêtres variables et $O(MAX(|\mathbb{g}^+|)^g)$ pour les fenêtres fixes. Il n'est donc pas pratique de les énumérer. On peut couper l'espace de fouille en construisant les solutions par fusion de sous-solutions qui respectent la contrainte de distance sur $\mathbb{G}^-$ mais cette approche n'offre aucune garantie sur la complexité du calcul et n'offre à perte de vue qu'un infini marasme.

Fondamentalement, une SDL partagée dans un ensemble $\mathbb{G}^+$ forme une clique pour un seuil de $\mathcal{D}^{SIM}$ choisi tel qu'aucun membre de $\mathbb{G}^-$ ne soit dans la clique et tous les membres de $\mathbb{G}^+$ y soient. Construisons le graphe dont les sommets sont les fenêtres dans $\mathbb{G}^+$ et les arêtes sont les $\mathcal{D}^{SIM}$ entre elles. Enlevons de ce graphe les arêtes dont la valeur de l'étiquète est plus grande que la plus petite distance séparant un des sommets avec n'importe quelle fenêtre dans $\mathbb{G}^-$. Les cliques comprenant au moins une fenêtre de chaque gène dans $\mathbb{G}^+$ sont des SDL partagées. Malheureusement le graphe contient $O(N)$ nœuds et

$O(N^2)$ arêtes et l'identification de cliques maximales est un des prototypes des problèmes NP-difficiles (Karp 1972).

Simplifions le problème en procédant à un groupage sur toutes les fenêtres dans $\mathbb{G}^+$ et en éliminant les groupes dont les signatures auraient des similarités *trop grandes* avec des signatures tirées de $\mathbb{G}^-$. *A priori* cela nécessite le calcul et le stockage de la distance de toutes les paires de fenêtres dans $\mathbb{G}^+$ en $N^2/2$, suivi du groupage lui-même en $O(N^3)$ et du calcul des distances entre ces fenêtres et celles dans $\mathbb{G}^-$ soit $N \times M$ donnant un complexité en espace $O(MN + N^2)$ et en temps $O(MN + N^3)$. En pratique, puisque la taille de l'instance qui nous intéresse le permet sur des fenêtres de tailles fixes, nous pouvons simplement appliquer le groupage sur $\bigcup(\mathbb{G}^+, \mathbb{G}^-)$ en $O(M^3N^3)$ et identifier les groupes qui incluent tous les membres de $\mathbb{G}^+$ mais aucun membre de $\mathbb{G}^-$. Au chapitre 2, nous avons décrit comment calculer un alignement multiple de signatures nous n'avons donc qu'à modifier la représentation de l'arbre de groupage hiérarchique et de chercher des solutions dans celui-ci.

3.4.3

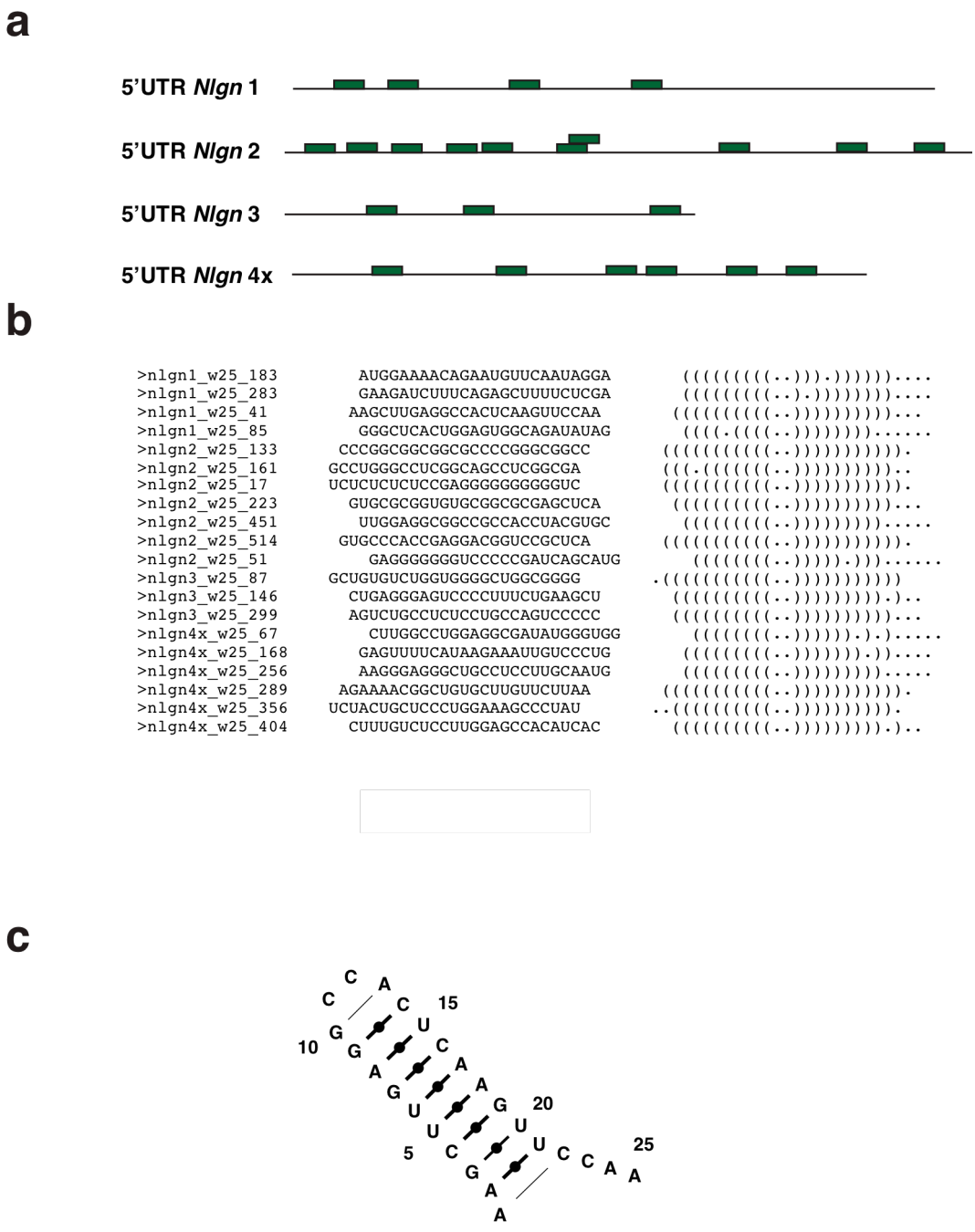

Figure 27. SDL des neuroligines

(a) La SDL identifiée. (b) Structures choisies grâce au logiciel MC-Cons. (c) Structure typique (*Nlgn1*, position 41.

3.4.4 Résultats

Nous avons cherché une SDL partagée parmi les 5'UTR des ARMm de neuroligines $\mathbb{G}^+ = \{$ *NLGN1, NLGN2, NLGN3, NLGN4X, NLGN4Y* $\}$ dont la traduction est élevée en absence de eIF-4E mais absente d'un ensemble dont l'expression est diminuée $\mathbb{G}^- = \{$ *PSD95*-v1, *PSD95*-v2, *ACTB*, *GEPHYRIN*, *SHANK2*-v1, *SHANK2*-v2, *SAPAP3*-v1, *SAPAP3*-v2, *SAPAP3*-v3, *NRXN1*-v1, *NRXN1*-v2, *NRXN2*-v1, *NRXN2*-v2, *NRXN3*-v1, *NRXN3*-v2 $\}$. Nous avons obtenues les séquences des ARNm dans la base de donnée refseq (Sayers, Barrett et al. 2012) dont les indications précisent la position des régions non-traduites; notez que les régions 5'UTR de la plupart de ces gènes sont variables.

Nous avons calculé et comparées entre-elles les signatures de taille fixes choisies parmi $\omega \in \{ 15, 20, 25, \ldots, 80 \}$ avec 20 structures sous-optimales. Les signatures ont été comparées avec l'algorithme en programmation dynamique afin de permettre les insertions. Un arbre de groupage hiérarchique a été construit pour chaque ω. Dans ces arbres nous avons identifié toutes les branches comprenant au moins une signature extraite de $\mathbb{G}^+$ (au moins une de chaque) mais aucune de $\mathbb{G}^-$. Ce processus a identifié 7 SDL candidates. Sur chacune de ces solutions, nous avons utilisé le logiciel MC-Cons (Parisien and Major 2008) pour identifier des structures hautement conservées.

Nous avons retenu la SDL dont la 2D représentative est montrée à la figure 27. Ce candidat ressemble à d'autres structures connues pour interagir avec des protéines; comme par exemple la tige-boucle TAR chez HIV rencontrée au chapitre 2.

3.5 Conclusion au chapitre 3

Les nœuds des arbres de groupage hiérarchiques dont les sous arbres sont composés de tous les membres de $\mathbb{G}^+$ et d'aucun membre de $\mathbb{G}^-$ pour l'instance qui nous concerne, sont rares. Nous en avons identifiés quelques uns et au moins l'un d'entre eux correspond à une SDL candidate et à la fonction biologique probante. Évidemment, ce travail ne constitue pas une preuve de fonctionnalité ou même d'existence de ces structures. Nous attirons simplement l'attention des chercheurs en biochimie sur le fait qu'il est possible que

les SDL induisent des interactions moléculaires directes avec d'autres macromolécules et qu'on peut alors détecter ces modules simplement en analysant les séquences d'ARN. Des observations expérimentales de ces phénomènes sont nécessaires.

4 Conclusion

Depuis plus de quarante ans nous essayons d'écrire le programme d'ordinateur qui serait capable de donner en sortie l'assignation des paires de bases de la structure d'un ARN étant donnée sa séquence en entrée. Encore à ce jour il n'existe pas de tel logiciel (Rivas 2013). Certaines approches y arrivent pour de rares instances mais ce but reste élusif. Et si la question était mal posée? Considérons que les ARN sont polymorphes et qu'ils changent dynamiquement de structures selon des probabilités qui varient selon leurs séquences et leurs environnements. La question devient alors: Quels sont les ensembles de structures probablement occupées par une molécule étant donnée sa séquence? Malheureusement, cette question nous conduit dans des zones ardues en général car les travaux expérimentaux nécessaires pour vérifier les hypothèses qui en découlent sont technologiquement lourds. Néanmoins, certains travaux récents (Dethoff, Chugh et al. 2012) montrent la relation directe entre les énergies relatives des structures 2D prédites par MC-Fold et les observations faites grâce à une utilisation sophistiquée de la RMN.

Les prédictions 2D du logiciel MC-Fold sont calculées sur le modèle des NCM et ce modèle capture les structures 2D vérifiées expérimentalement par Al-Hashimi (Dethoff, Chugh et al. 2012). Mais le calcul n'est pas aussi rapide qu'espéré et est limité à des instances de tailles réduites. C'est pourquoi, au premier chapitre, nous avons décrit une ré-écriture de ce logiciel. MC-Flashfold augmente la capacité de calcul sur le modèle des NCM au niveau des outils les plus rapides. En quelques secondes il nous est dorénavant possible de calculer les ensembles de prédictions de structures $2D$ proches de l'optimal sur le modèle d'énergie des NCM. Cela nous permet en particulier d'envisager des balayages de génomes pour trouver des zones dont les dynamiques structurales se ressemblent. Cela était hors de portée avant MC-Flashfold.

Il reste de la place pour l'amélioration toutefois car l'énumération des ensembles de solutions pourrait être accélérée d'un facteur $\frac{n}{\log \kappa}$ si les constantes cachées, la gestion de mémoire et la base du log le permettent dans l'algorithme *k-best* de (Huang and Chiang 2005). Des expérimentations seraient nécessaires pour le savoir. D'autres améliorations

toutefois sont plus criantes. Par exemple, nous souhaiterions intégrer une probabilité *a priori* des NCM donnée par des expériences de sondage chimique et augmenter le modèle avec des valeurs d'énergie pour des motifs particuliers (*k-turns* par exemple) et des jonctions de tiges. Certains de ces raffinements au modèle d'énergie auront des effets marqués sur la phase de programmation dynamique.

Qu'en est-il alors de la différence entre deux séquences du point de vue de leurs ensembles probables de structures 2D? Cela peut-il induire des changements en termes de fonction biologique qui puissent être prédits par un logiciel? Cette question a de nombreuses applications d'intérêt immédiat en biologie. En particulier: Quel est l'impact d'une mutation sur la fonction d'une molécule d'ARN? Voir par exemple à ce sujet les travaux de Waldispül (Waldispuhl, Devadas et al. 2008) Et : Quels sont les modules fonctionnels présents dans des ensembles de gènes?

Au chapitre 2, nous avons montré que l'impact fonctionnel de mutations ponctuelles sur des microARN est prévisible en inspectant les changements associés à ces mutations dans les structures 2D probables. Pour cela, nous avons montré comment calculer une signature représentative de ces ensembles et comment comparer ces signatures entre-elles. Nous avons vérifié que la similarité entre les signatures corrèle avec l'activité biologique.

Le troisième chapitre était consacré à ce que nous avons appelé les signatures dynamiques locales (SDL). Ces SDL sont des régions d'ARN plus longs dont la dynamique particulière est indépendante du contexte de séquence dans lesquelles elles se trouvent. Nous avons vu une méthode de balayage par fenêtres juxtaposées pour établir un profil révélateur de leurs positions et une application de la méthode à la détection du signal de polarisation présent dans l'ARNm du gène *vegt* chez *Xenopus leavis*. Puis, nous avons utilisé le groupage hiérarchique comme heuristique dans la découverte de SDL répétés dans les ARNm des gènes de neuroligines impliqués dans certains désordres neurologiques chez des modèles transgéniques de souris.

Appendices

Appendice I : Exemples de la méthode

Dans cette section, nous décrivons trois exemples de la méthode algorithmique décrite au chapitre 1.

1- Retour en arrière avec itérateurs sur une seule pile. Nous considérons le problème d'alignement de séquences avec trous sur une fonction quelconque d'évaluation de la valeur des trous. Cette fonction pourrait inspecter la séquence des brins d'un trou par exemple mais pour garder les choses simples, nous utiliserons une fonction logarithmique sur sa longueur.

2- Retour en arrière sans itérateur sur une seule pile. Nous revisitons l'alignement global de séquences avec trous où la valeur des trous est déterminée selon le modèle affine où le poids d'ouverture d'un trou est ajouté à la longueur de celui-ci multiplié par un poids d'élongation. Pour varier nous permettrons aux trous de changer de brin dans le sens où un segment non-apparié immédiatement suivi d'un segment non-apparié mais situé sur l'autre brin sont considérés comme ne formant qu'un seul trou. Ainsi le poids d'ouverture n'est appliqué qu'une seule fois sur ces trous plutôt que deux fois. Des trous se faisant face entrent en compétition avec les valeurs de non-concordance (missmatch) et ceci permet de trouver un équilibre entre des segments fonctionnellement différents et des segments ou les non-concordances pourraient dominer.

3- Retour en arrière sur deux piles. Nous donnons un algorithme pour corriger certains types d'erreurs dans des chaines de parenthèses bien balancées en énumérant les séquences sous-optimales où l'on introduit des corrections. Cet exemple a des similitudes avec le problème du repliement d'ARN et introduit l'utilisation de deux piles.

Alignement de séquences sous diverses métriques de trous

Pour deux séquences ($A = a_1a_2\cdots a_i\cdots a_n$ et $B = b_1b_2\cdots b_j\cdots b_m$) la grammaire suivante décrit un alignement :

$$M \rightarrow (a,b)M \mid (a,-)M \mid (-,b)M \mid \epsilon$$

où ϵ est un mot de longueur 0. Cette grammaire décrit les alignements sur deux séquences comme une suite de correspondances (a,b), d'insertion (a,-) et de suppression (-,b). Dans ce schéma, il est impossible d'évaluer globalement le coût des trous (par exemple sur la longueur des trous) car les étapes successives sont indépendantes les unes des autres. La grammaire permet d'utiliser un coût fixe pour chaque émission du symbole de trou. Mais nous voulons évaluer les trous en fonction de leurs longueurs alors nous regroupons les règles de trous et nous introduisons un non-terminal A de départ pour obtenir la grammaire $\mathcal{A}$:

$$A \rightarrow M \mid G$$

$$M \rightarrow (a,b)M \mid (a,b)G \mid \epsilon$$

$$G \rightarrow (a,-)A \mid (-,b)A$$

Dans les deux prochains exemples, nous construirons à partir de $\mathcal{A}$.

Méthode générique dans $O(n^2m^2)$

Nous commençons par décorer les règles de $\mathcal{A}$ avec des indices :

$$A_{i,j} \rightarrow \quad M_{i,j} \mid G_{i,j}$$

$$M_{i,j} \rightarrow \quad (a,b)M_{i-1,j-1} \mid (a,b)G_{i-1,j-1} \mid \epsilon$$

$$G_{i,j} \rightarrow \quad (a,-)A_{i-1,j} \mid (-,b)A_{i,j-1}$$

Puis nous introduisons les itérateurs :

$$A_{i,j} \rightarrow \quad M_{i,j} \mid G_{i,j}$$

$$M_{i,j} \rightarrow \quad (a,b)M_{i-1,j-1} \mid (a,b)G_{i-1,j-1}$$

$$G_{i,j} \rightarrow \quad (a,-)^{i-p}A_{p,j} \qquad \mid \forall\ 1 \leq p \leq i$$

$$\qquad\quad \mid (-,b)^{j-q}A_{i,q} \qquad \mid \forall\ 1 \leq q \leq j$$

mais cette version est imparfaite puisqu'il est possible pour des séries d'insertions et de suppressions de se juxtaposer par applications successives de G → A → G prévenant l'usage de la fonction globale sur la longueur du trou. Une façon de s'en sortir consiste à grouper l'insertion et la suppression dans une même règle et de forcer la présence d'au moins une correspondance entre deux trous :

$$A_{i,j} \rightarrow \quad M_{i,j} \mid G_{i,j}$$

$$M_{i,j} \rightarrow \quad (a,b)M_{i-1,j-1} \mid (a,b)G_{i-1,j-1}$$

$$G_{i,j} \rightarrow \quad [(a,-)(-,b)]^{i-p+j-q}\, M_{p,q} \qquad \forall 1 \leq p \leq i,\ 1 \leq q \leq j$$

Il est maintenant possible d'utiliser une fonction générique pour établir le coût d'un trou mais ceci n'est possible qu'en augmentant la complexité (la routine de programmation dynamique est maintenant $\mathcal{O}(n^2m^2)$). On évalue les coûts avec deux fonctions, la correspondance ($\mathcal{M}$) et la longueur du trou ($\mathcal{F}$). Par exemple :

$$\mathcal{M}(a,b) = \begin{cases} 3 \mid si\ a \equiv b \\ -2 \mid sinon \end{cases}$$

$$\mathcal{F}(l) = 5 + log_{10}(l+1)$$

Nous ajoutons maintenant les fonctions de coûts aux règles (nous nommons les règles et omettons les terminaux pour simplifier) :

Start Corresondance $A_{i,j} \rightarrow M_{i,j}$

Start Trou $| G_{i,j}$

Extension $M_{i,j} \rightarrow \mathcal{M}(a,b)\, M_{i-1,j-1}$

Extension Trou $| \mathcal{M}(a,b) G_{i-1,j-1}$

Trou $G_{i,j} \rightarrow MAX_{i \leq p \leq i, 1 \leq q \leq j | (i-p)+(j-q) > 0} \mathcal{F}(i-p+j-q) + \mathcal{M}_{p,q}$

Donnant les récurrences $\mathcal{O}(n^2m^2)$en programmation dynamique :

$$A_{i,j} = MAX \begin{cases} M_{i,j} \\ G_{i,j} \end{cases}$$

$$M_{i,j} = MAX \begin{cases} \mathcal{M}(a,b) + M_{i-1,j-1} \\ \mathcal{M}(a,b) + G_{i-1,j-1} \end{cases}$$

$$G_{i,j} = MAX_{i \leq p \leq i, 1 \leq q \leq j | (i-p)+(j-q) > 0} \mathcal{F}(i-p+j-q) + M_{p,q}$$

Et les états correspondants de retour en arrière :

État	ID	MPR	Pile	Suivant	Itérateurs
Start Corr.	1	$g() + M_{i,j}$	3	2	
Start Trou	2	$g() + G_{i,j}$	5		
Extension	3	$g() + \mathcal{M}(a,b) + M_{i-1,j-1}$	3	4	
Ext. Trou	4	$g() + \mathcal{M}(a,b) + G_{i-1,j-1}$	5		

Trou	5	$g() + \mathcal{F}(i-p+j-q) + M_{p,q}$	3		$1 \leq p \leq i, 1 \leq q \leq j \mid (i-p)+(j-q) > 0$

Note : dans les tableaux de description de retour en arrière, nous donnons un identificateur unique aux états (ID). Ces identificateurs donnent l'état qui doit être empillé (colone Pile) lorsque $MPR \leq PO - T$ (minimisation). Lors du dépilement d'un état, si une règle alternative est décrite par la grammaire, alors celle-ci est évaluée (colone Suivant). Cela correspond simplement aux règles formulées par alternatives dans la grammaire. Les règles *type-Q* nécessitent une colone Pile2 pour décrire l'état destiné à la seconde pile.

Métrique affine dans $\mathcal{O}(nm)$

L'alignement en $O(nm)$ est atribué à Gotoh. Voir (Gotoh 1982; Gotoh 1990).

Ce modèle de coût des trous nous demande de séparer le coût d'initiation et le coût d'élongation des trous. Pour cela, nous ajoutons une règle à la grammaire $\mathcal{A}$.

A →	M | G	Start avec corresondance ou trou
M →	(a,b)M	Extension
	| (a,b)G	Fermeture de trou
G →	M	Initiation de trou
	(a,-)G	Extension de trou dans B
	(-,b)G	Extension de trou dans A

Où les règles sont décorées comme suit :

$$\begin{aligned} A_{i,j} &\rightarrow M_{i,j} \mid G_{i,j} \\ M_{i,j} &\rightarrow (a,b)M_{i-1,j-1} \\ &\quad \mid (a,b)G_{i-1,j-1} \\ G_{i,j} &\rightarrow M_{i,j} \end{aligned}$$

$| (a,-)G_{i-1,j}$
$| (-,b)G_{i,j-1}$

La nouvelle fonction de coût est

$\mathcal{M}(a,b) = \begin{cases} 3 \mid si\ a \equiv b \\ -2 \mid sinon \end{cases}$ Correspondance et différence (match et missmatch)

$\mathcal{I} = -0.5$ Initiation de trou

$\mathcal{G} = -0.5$ Extension de trou

Que nous introduisons dans la grammaire :

$A_{i,j} \rightarrow \quad M_{i,j} \mid G_{i,j}$

$M_{i,j} \rightarrow \quad \mathcal{M}(a,b) + (a,b)M_{i-1,j-1}$

$\mid \mathcal{M}(a,b) + (a,b)G_{i-1,j-1}$

$G_{i,j} \rightarrow \quad \mathcal{I} + M_{i,j}$

$\mid \mathcal{G} + (a,-)G_{i-1,j}$

$\mid \mathcal{G} + (-,b)G_{i,j-1}$

Cette grammaire est parfaitement adéquate lorsque nous ne considérons qu'un meilleur alignement mais entraine deux situations indésirables. Premièrement, la règle $G_{i,j} \rightarrow M_{i,j}$ est un choix sous-optimal pour le retour en arrière car elle introduit des états sur la pile qui ne consomme aucun indice et n'émettent pas de symboles terminaux. Cette règle permet l'énumération de solutions ne différend que par la position de départ des trous et du coût de l'alignement mais qui sont

fonctionnellement équivalents. Ceci est solutionné en combinant cette règle au départ du trou dans l'une ou l'autre des séquences :

$$A_{i,j} \rightarrow \quad M_{i,j} \mid G_{i,j}$$

$$M_{i,j} \rightarrow \quad \mathcal{M}(a,b) + (a,b)M_{i-1,j-1}$$

$$\mid \mathcal{M}(a,b) + (a,b)G_{i-1,j-1}$$

$$G_{i,j} \rightarrow \quad \mathcal{I} + \mathcal{G} + (a,-)M_{i-1,j}$$

$$\mid \mathcal{I} + \mathcal{G} + (-,b)M_{i,-1j}$$

$$\mid \mathcal{G} + (a,-)G_{i-1,j}$$

$$\mid \mathcal{G} + (-,b)G_{i,j-1}$$

Le second problème est une énumération des différents arrangements possibles de insertions/suppressions, tous équivalents. Pour exprimer toutes ces possibilités par une seule solution, remplaçons les règles de trous (G) par des paires de règles coordonnées.

Start Correspondance	$A_{i,j} \rightarrow$	$M_{i,j}$
Start Insertion		$\mid V_{i,j}$
Start Délétion		$\mid H_{i,j}$
Correspondance	$M_{i,j} \rightarrow$	$\mathcal{M}(a,b) + (a,b)M_{i-1,j-1}$
Correspondance Insertion		$\mid \mathcal{M}(a,b) + (a,b)V_{i-1,j-1}$
Correspondance Délétion		$\mid \mathcal{M}(a,b) + (a,b)H_{i-1,j-1}$
Insertion	$V_{i,j} \rightarrow$	$\mathcal{I} + \mathcal{G} + (a,-)M_{i-1,j}$
Insertion Extension		$\mid \mathcal{G} + (a,-)V_{i,-1j}$

Délétion	$H_{i,j} \rightarrow$	$\mathcal{I} + \mathcal{G} + (-,b)M_{i,j-1}$
Délétion Extension		$\mid \mathcal{G} + (-,b)G_{i,j-1}$
Délétion Insertion		$\mid \mathcal{G} + (-,b)X_{i,j-1}$
Croisement de trou	$X_{i,j} \rightarrow$	$\mathcal{G} + (a,-)V_{i-1,j}$

Ce qui donne les récurrences suivantes :

$$A_{i,j} = MAX \begin{cases} M_{i,j} \\ H_{i,j} \\ V_{i,j} \end{cases}$$

$$M_{i,j} = MAX \begin{cases} \mathcal{M}(a,b) + M_{i-1,j-1} \\ \mathcal{M}(a,b) + V_{i-1,j-1} \\ \mathcal{M}(a,b) + H_{i-1,j-1} \end{cases}$$

$$V_{i,j} = MAX \begin{cases} \mathcal{I} + \mathcal{G} + M_{i-1,j} \\ \mathcal{G} + V_{i-1,j} \end{cases}$$

$$H_{i,j} = MAX \begin{cases} \mathcal{I} + \mathcal{G} + M_{i,j-1} \\ \mathcal{G} + V_{i-1,j} \\ \mathcal{G} + X_{i,j-1} \end{cases}$$

$$X_{i,j} = \mathcal{G} + V_{i-1,j}$$

Et les états de retour en arrière correspondants.

État	ID	BPS	Pile	État suivant
Start Correspondance (START_MATCH)	1	$g() + M_{i,j}$	4	2
Start Insertion (START_V)	2	$g() + V_{i,j}$	8	3
Start Délétion (START_H)	3	$g() + H_{i,j}$	10	
Correspondance (EXTENSION)	4	$g() + \mathcal{M}(a,b) + M_{i-1,j-1}$	4	5
Correspondance Insertion (CLOSING_V)	5	$g() + \mathcal{M}(a,b) + V_{i-1,j-1}$	8	6
Correspondance Délétion (CLOSING_H)	6	$g() + \mathcal{M}(a,b) + H_{i-1,j-1}$	10	
Insertion (INITGAP_V)	7	$g() + \mathcal{I} + \mathcal{G} + M_{i-1,j}$	4	
Insertion Extension (EXT_V)	8	$g() + \mathcal{G} + V_{i-1,j}$	8	7
Délétion (INITGAP_H)	9	$g() + \mathcal{I} + \mathcal{G} + M_{i,j-1}$	4	11
Délétion Extension (EXT_H)	10	$g() + \mathcal{G} + H_{i,j-1}$	10	9
Délétion Insertion (EXT_HX)	11	$g() + \mathcal{G} + X_{i,j-1}$	12	
Croisement de trou (EXT_XV)	12	$g() + \mathcal{G} + V_{i-1,j}$	8	

Cet algorithme constitue un modèle de base simple et parfait pour l'étude. Son code source en langage c (align.c) est inclus dans la distribution de MC-Flashfold et fourni une introduction simple au code de MC-Flashfold. Dans le tableau des états, les noms entre parenthèses sont utilisés dans le code.

Retour en arrière (backtracking) sur deux piles; le cas des parenthèses balancées

$$S \rightarrow (S) \mid SS \mid \epsilon$$

Portons notre attention sur un langage de parenthèses balancées < {S}, {'(',')'}, R, S > avec les règles suivantes dans R :

> Ce langage nous intéresse pour les raisons suivantes : (1) c'est un prototype de langage hors contexte, (2) il utilise une règle type-Q et (3) les fondements du langage du repliement d'ARN suivent le même squelette.
>
> Pour notre exemple nous construirons une grammaire qui suggère des corrections aux chaines de parenthèses balancées pourvues d'erreurs. Par exemple, sur entrée (((( )) (( ))) une solution corrigée serait (((( )) (( )))) .
>
> La grammaire suivante permet d'identifier toutes les chaines erronées :

$$S \rightarrow (S) \mid (S \mid S) \mid SS \mid \epsilon$$

où ϵ est un mot de longueur 0. Remarquons que les structures décrites sont composées de troncs et de branches. Un tronc est un segment maximal de parenthèses appariées tels que …((((…))))… ou ((((())))). Les troncs sont reconnus par applications successives de la règle S → (S). Les branches sont des embranchements et ne sont donc pas composées de terminaux. Ce sont des bifurcations entre les troncs et les troncs sont donc délimités par des branches. Elles sont responsables des structures en Y telles que ((((()) (())))) et sont les conséquences d'applications de la règle S → SS. Une parenthèse manquante dans un brin gauche d'un tronc pourrait être reconnue à différentes positions du brin droit correspondant, générant des ensembles de solution équivalentes potentiellement grands (par exemple, avec la parenthèse angulée [marquant la position de l'insertion, le mot erroné donné en entrée produit un ensemble de solutions équivalentes : (((…)))) ⇒ { [(((…)))), ([((…)))), (([(…)))), (([…)))) }). Nous allons réduire le nombre de positions où il est possible d'introduire les corrections afin de réduire la taille des sorties. Nous réarrangeons les règles et les décorons d'indices comme suit :

$$S_{i,j} \rightarrow (S_{i+1,j-1}) \mid R_{i,j} \mid L_{i,j} \mid \epsilon$$

$$R_{i,j} \rightarrow (R_{i+1,j} \mid T_{i,j} \mid \epsilon$$

$$L_{i,j} \rightarrow L_{i,j-1}) \mid T_{i,j} \mid \epsilon$$

$$T_{i,j} \rightarrow S_{i,s}S_{s+1,j}$$

De laquelle on enlève les non-terminaux et à laquelle on ajoute une fonction de coût :

La valeur d'une paire balancée	$\mathcal{M} = 2$
La valeur de corriger une parenthèse	$\mathcal{I} = 1$

Pour obtenir l'ensemble final de règles décorées:

Extension	$S_{i,j} \rightarrow$	$\mathcal{M} + S_{i+1,j-1}$
Extension à droite		$\mid R_{i,j}$
Extension à gauche		$\mid L_{i,j}$
Fin d'extension		$\mid \epsilon$
Droite	$R_{i,j} \rightarrow$	$\mathcal{I} + R_{i+1,j}$
Droite branche		$\mid T_{i,j}$
Droite fin		$\mid \epsilon$
Gauche	$L_{i,j} \rightarrow$	$\mathcal{I} + L_{i,j-1}$
Gauche branche		$\mid T_{i,j}$
Gauche fin		$\mid \epsilon$
Branche	$T_{i,j} \rightarrow$	$S_{i,s}S_{s+1,j}$

dont on obient les récurrences suivantes :

$$T_{i,j} = MAX_{i<s<j-1} S_{i,s} + S_{s+1,j}$$

$$S_{i,j} = MAX \begin{cases} 0 & | \ i \geq j \\ MAX \begin{cases} \mathcal{M} + S_{i+1,j-1} \\ R_{i,j} \\ L_{i,j} \end{cases} & | \ i < j \end{cases}$$

$$R_{i,j} = MAX \begin{cases} 0 & | \ i \geq j \\ MAX \begin{cases} \mathcal{I} + R_{i+1,j} \\ T_{i,j} \end{cases} & | \ i < j \end{cases}$$

$$L_{i,j} = MAX \begin{cases} 0 & | \ i \geq j \\ MAX \begin{cases} \mathcal{I} + L_{i,j-1} \\ T_{i,j} \end{cases} & | \ i < j \end{cases}$$

Et les états de retour en arrière :

État	ID	BPS	Pile	Pile2	Suivant	Itérateurs
Extension	1	$g() + \mathcal{M} + M_{i+1,j-1}$	1		2	
Extension Droite	2	$g() + R_{i,j}$	4		3	
Extension Gauche	3	$g() + L_{i,j}$	7			
Droite	4	$g() + \mathcal{M} + R_{i+1,j}$	4		5	
Droite Branche	5	$g() + T_{i,j}$	10		6	
Droite Fin	6	$g()$				
Gauche	7	$g() + \mathcal{M} + L_{i,j-1}$	7		8	
Gauche Branche	8	$g() + T_{i,j}$	10		9	
Gauche Fin	9	$g()$				

Branche	10	$g() + S_{i,s} + S_{s+1,j}$	1	1		$i<s<j-1$

Apendice II : Tests de performance sur des entrées aléatoires

Voici deux bancs d'essais de MC-Flashfold sur des entrées aléatoires. Dans le premier, nous mesurons la performance des différentes étapes de calcul et dans le second nous mesurons le temps pris par le programme en différentes situations pour effectuer le traitement complet.

Les temps d'exécution sont obtenus par des appels à la fonction `get_rusage()` du langage c avant et après les fonctions respectives ou au début et à la fin de l'exécution du programme. Les temps rapportés sont ceux utilisés par le CPU pendant l'exécution du code. Aussi, afin de limiter l'impact éventuel de péremption par des fils d'exécution concurrents sur l'antémémoire causant des absences d'information (*cache misses*) les tests ont été effectués sur un ordinateur autrement inoccupé. Tous les tests ont été effectués sur des entrées pseudo-aléatoires dont la composition est environ $\frac{1}{4}$ pour chaque nucléotide dans { A, C, G, U }. Lorsque des valeurs de Δ pour un κ donné étaient nécessaires, celles-ci ont été calculées en prétraitement par MC-Flashfold en utilisant la stratégie du seuil flottant. Les polynômes ajustés (*fitted*) aux données soit pour en extraire les paramètres ou pour en faire des dessins graphiques l'ont été avec la fonction *lm()* disponible dans le langage R. Tous les résultats présentés ici sont les résultats d'exécution sur un même ordinateur physique (Intel Core i5 cadencé à 3.41 GHz équipé de 16 Gb de mémoire vive et géré par le système d'exploitation Mac OS X version 10.8.3).

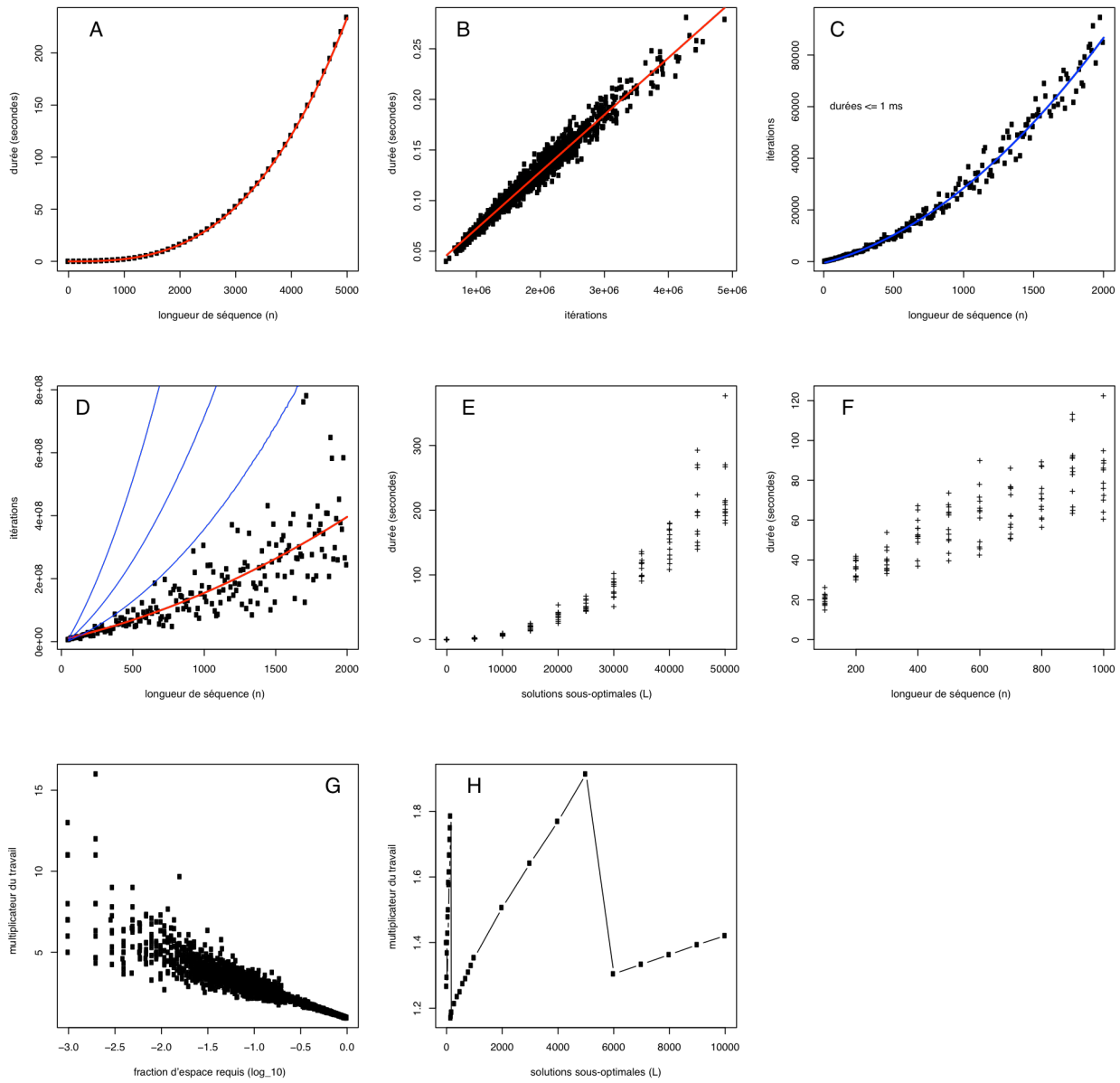

Figure II.1: Durées d'exécution des étapes algorithmiques.

Figure II.1, panneau A ($0 < n \leq 5000$, $Pts = 51$)

Temps pour compléter la passe de programmation dynamique sur des séquences aléatoires de longueurs variées.

Ces valeurs donnent aussi presque exactement le temps nécessaire pour générer les structures de MFE pour des séquences aléatoires de longueurs montrées.

Figure II.1, panneau B ($n = 500,\ réplicats = 1000$, $\kappa = 1000$)

Dans le texte, nous utilisons le nombre d'itérations autour de la boucle principale de retour en arrière comme estimateur du travail effectué. Afin de vérifier cela, nous avons mesurer le temps pris en fonction du nombre d'itérations. La variation observée dans le nombre d'itérations dépends des traits fondamentaux de la séquence d'entrée et semble corréler linéairement avec le temps de calcul. (Une correction est appliquée lorsque $\kappa \neq 1000$. Cela peut se produire lorsque plus d'une séquence ont une énergie de valeur Δ. Dans tous les cas, ce facteur est $\lesssim \frac{1}{200}$.)

Figure II.1, panneau C ($50 \leq n \leq 200$, $Pts = 163$)

Nombre d'itérations pour le calcul d'une seule structure.

Le polynôme est $\mathcal{I} = \frac{n^2}{67} + 14n - 520$. Dans le prochain panneau, nous utiliserons cette équation pour estimer la hauteur de la pile qui est réutilisée pendant les retours en arrière lorsque $\kappa > 1$.

Figure II.1, panneau D ($50 \leq n \leq 200$, $Pts = 196$, $\kappa = 50000$)

Nous sommes intéressés par le nombre d'éléments qui sont réutilisés sur la pile lors du retour en arrière pendant l'énumération de plusieurs structures sous-optimales.

Le nombre d'itérations effectuées pour des séquences aléatoires de diverses longueurs lorsque 50000 structures sont calculées est montré. La ligne du haut correspond aux itérations nécessaires lorsqu'aucun élément de la pile n'est réutilisé (basé sur le polynôme du panneau C) : $\kappa\mathcal{I}$. La seconde ligne est $\frac{L}{2}\times\mathcal{I}$ et la troisième est $\frac{L}{4}\times\mathcal{I}$. Ceci indique que la situation est encore meilleure que nous l'envisagions et approche $\frac{L}{10}\times\mathcal{I}$ pour cet ensemble spécifique de conditions.

Figure II.1, panneau E ($n = 500$, $Pts = 132$, $1 \leq \kappa \leq 50000$) (κ est noté L sur l'image)

Temps nécessaire pour le groupage en $n\kappa^2$ pour une taille fixe $n = 500$ et un nombre de structures sous-optimales variant uniformément. La variation pour un κ donné est dû à la variation intrinsèque au calcul de deux séquences distinctes.

Figure II.1, panneau F ($100 \leq n \leq 1000$, $\kappa = 25000$, $Pts = 120$)

Temps nécessaire pour le groupage en $n\kappa^2$ pour $\kappa = 25000$ et des longueurs qui augmentent.

Figure II.1, panneau G ($n = 250$, $10 \leq \kappa \leq 10000$, $Pts = 2810$)

Ici nous montrons le travail excédentaire effectué par l'algorithme du tas lorsque le seuil initial Δ est surestimé. Supposons qu'un usager cherche un nombre κ de structures mais qu'au lieu de fournir le seuil adéquat Δ^{κ}, il/elle fourni un seuil plus grand Δ^{+} et utilise l'algorithme de la pile pour trouver le seuil adéquat. Dans ces conditions, la quantité de travail nécessaire est $\mathcal{H}(L, \Delta^{+})$ pour lister les structures étant donné le seuil trop grand. Le

ratio $\frac{\mathcal{H}(L,\Delta^+)}{\mathcal{H}(L,\Delta^L)}$ est donné en ordonnée en fonction de l'ordre de grandeur du ratio de la taille des espaces de solution des seuils Δ^+ et Δ^κ; soit $\log_{10}\left(\frac{\kappa^{\Delta^\kappa}}{\kappa^{\Delta^+}}\right)$. En particulier, ceci indique que si $k \approx 2$ dans le raffinement itératif de Δ_i, alors $T^\kappa(\Delta^\kappa)$ peut être borné. Ceci est le sujet du prochain panneau.

Figure II.1, panneau H ($n = 250$, $10 \leq \kappa \leq 10000$, $Pts = 39$) (κ est noté L sur l'image)

Les deux stratégies (l'itération et le tas) sont appliquées conjointement pour trouver Δ^κ. On y voit que le travail ne dépasse pas un (petite) constante multiplicative du travail effectué lors du dernier retour en arrière. Le ratio du temps mit pour trouver Δ^κ et de celui mit pour énumérer les sous-optimaux étant donné Δ^κ est donné en ordonné en fonction de κ pour une séquence dont $n = 500$. Il est a prévoir cependant que sur certaines entrées, la dernière itération surestimera Δ^κ ce qui sera responsable de temps d'exécution supérieurs.

Le second banc d'essais s'intéresse aux différents ensembles de paramètres qui pourraient faire varier le temps global d'exécution. Dans tous ces panneaux, le temps d'exécution en secondes est donné en ordonnée. Les abscisses des panneaux des colonnes 1 et 3 sont le nombre de solutions sous-optimales κ. Les abscisses des panneaux des colonnes 2 et 4 sont n les tailles des séquences d'entrée. Ces tracés comprennent chacun 20 répétitions de soit une série monotone de 21 valeurs de κ ($500 \leq \kappa \leq 10000$) pour une taille fixe de séquence ($n = 250$) pour les colonnes 1 et 3 ou de 11 valeurs de n ($50 \leq n \leq 501$) pour un nombre fixe de structures sous-optimales ($\kappa = 2500$) pour les colonnes 2 et 4. Ces valeurs sont calquées sur un usage typique. Les colonnes 1 et 2 sont obtenues avec un seuil (Δ) fixe alors que les colonnes 3 et 4 le sont pour des nombres fixes de sous-optimaux (κ fixe) et qui nécessitent la détermination de Δ^κ. Comme précédemment, chaque point est le résultat d'un calcul sur une nouvelle séquence pseudo-aléatoire.

Dans chaque panneau, un polynôme de degré 3 est tracé. Ces tracés sont de valeur théorique limitée mais permettent de comparer les panneaux entre-eux. Ces courbes sont reprises dans les panneaux I lorsque κ varie et J lorsque n varie. Ces courbes sont bleues pour un κ fixe et rouge pour un Δ fixe. Les patrons de pointillés de ces lignes sont les mêmes entre ces deux panneaux.

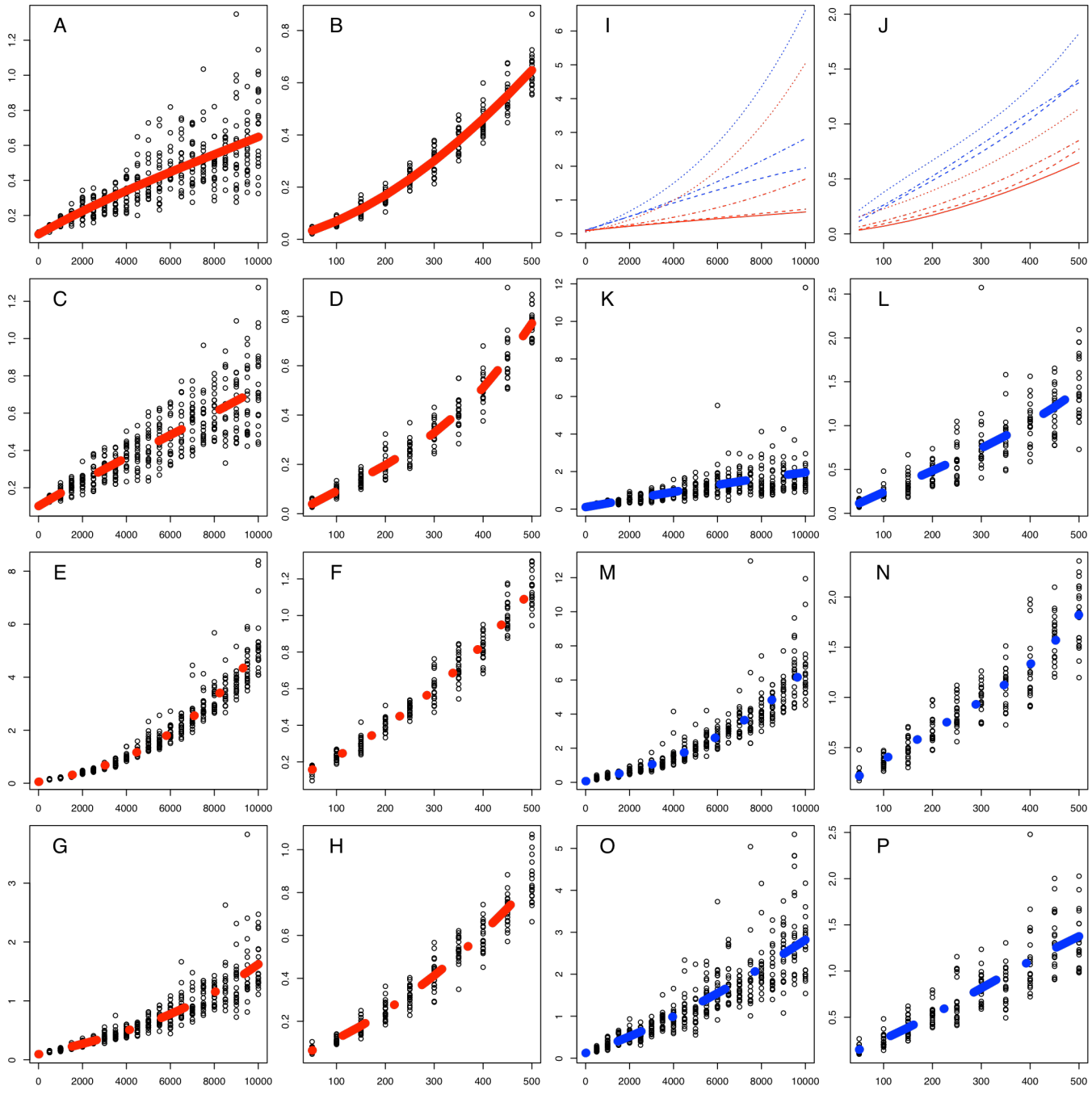

Figure II.2: Temps moyens d'exécution.

Figure II.2, panneaux A et B

La manière la plus rapide d'invoquer MC-Flashfold consiste à donner une valeur de Δ à la ligne de commande et de spécifier le paramètre `–no sort`. Évidemment, dans ce cas, les solutions ne sont pas triées et le nombre de structures produites en sortie n'est pas borné. Cette approche est utile si κ est grand et si l'on possède une connaissance *a priori* sur la relation entre Δ et κ. Cette option n'est pas disponible avec un κ fixe car elle n'a pas été implantée dans MC-Flashfold. Notez également que le temps nécessaire pour obtenir la solution MFE est presque exactement la même (± 1 ms) que celui de la passe de programmation dynamique (Figure 8 panneau A).

Figure II.2, panneaux C, D, K et L

Ces résultats sont presque identiques à ceux des résultats non-triés. Les points marginaux (*outliers*) des panneaux K et L sont des conséquences attendues pour certaines séquences d'entrées de surestimation de Δ^{κ} par la routine de raffinement itératif. L'amplitude de ces points est consistante avec un rapport dans l'ordre d'environ $\pm 2\times$ sur Δ^{κ}. Ceci est dû aux différences importantes dans le taux de croissance de κ versus Δ pour différentes séquences.

Figure II.2, panneaux E, F, G, H, M, N, O et P

La fraction κ^2 du coût du groupage des solutions par similarité (cet algorithme n'a pas été montré et est avantageusement remplacé par l'approche (Flamm, Hovacker et al. 2002)) est clairement visible dans les panneaux E et M. Le léger incrément de traitement obtenu par l'algorithme basé sur la structure de donnée métrique est évident en comparant les panneaux G/E et O/M.

Figure II.2, panneaux I et J

L'utilisabilité du programme dans cette plage de valeurs pour les paramètres explorées ici est bonne pour un usager en interaction avec un terminal d'ordinateur. Pour les situations où le logiciel est utilisé pour balayer un grand nombre de séquences, il serait avisé de choisir des paramètres dont les courbes croissent le plus lentement dans les panneaux I et J.

Bibliographie

Aho, A. and J. D. Ullman (1972). The Theory of Parsing, Translation and Compiling (volume I: Parsing), Prentice-Hall.

Aho, A. V., R. Sethi and J. D. Ullman (1985). Compilers. Principles, Techniques, and Tools, Addison Wesley.

Al-Hashimi, H. M. and N. G. Walter (2008). "RNA dynamics: it is about time." Curr. Opin. Struct. Biol. **18**: 321-329.

Alarcon, V. B. and R. P. Elinson (2001). "RNA anchoring in the vegetal cortex of the Xenopus oocyte." J Cell Sci **114**(Pt 9): 1731-1741.

Altuvia, S. and E. G. Wagner (2000). "Switching on and off with RNA." Proc Natl Acad Sci U S A **97**(18): 9824-9826.

Ambros, V. (2001). "microRNAs: tiny regulators with great potential." Cell **107**(7): 823-826.

Ban, N., P. Nissen, J. Hansen, P. B. Moore and T. A. Steitz (2000). "The complete atomic structure of the large ribosomal subunit at 2.4 A resolution." Science **289**(5481): 905-920.

Bartel, D. P. (2009). "MicroRNAs: target recognition and regulatory functions." Cell **136**: 215-233.

Bellman, R. (1952). "On the Theory of Dynamic Programming." Proc Natl Acad Sci U S A **38**(8): 716-719.

Berman, H. M., J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov and P. E. Bourne (2000). "The Protein Data Bank." Nucleic Acids Res **28**(1): 235-242.

Bernstein, E., A. A. Caudy, S. M. Hammond and G. J. Hannon (2001). "Role for a bidentate ribonuclease in the initiation step of RNA interference." Nature **409**(6818): 363-366.

Bevilacqua, P. C. and R. Russell (2008). "Editorial overview: exploring the vast dynamic range of RNA dynamics." Curr Opin Chem Biol **12**(6): 601-603.

Bilinski, S. M., M. K. Jaglarz, M. T. Dougherty and M. Kloc (2010). "Electron microscopy, immunostaining, cytoskeleton visualization, in situ hybridization, and three-dimensional reconstruction of Xenopus oocytes." Methods **51**(1): 11-19.

Bindewald, E., M. Wendeler, M. Legiewicz, M. K. Bona, Y. Wang, M. J. Pritt, S. F. Le Grice and B. A. Shapiro (2011). "Correlating SHAPE signatures with three-dimensional RNA structures." RNA **17**(9): 1688-1696.

Blower, M. D., E. Feric, K. Weis and R. Heald (2007). "Genome-wide analysis demonstrates conserved localization of messenger RNAs to mitotic microtubules." J Cell Biol **179**(7): 1365-1373.

Blower, M. D., M. Nachury, R. Heald and K. Weis (2005). "A Rae1-containing ribonucleoprotein complex is required for mitotic spindle assembly." Cell **121**(2): 223-234.

Bonhoeffer, S., J. S. McCaskill, P. F. Stadler and P. Schuster (1993). "RNA multi-structure landscapes. A study based on temperature dependent partition functions." Eur Biophys J **22**(1): 13-24.

Braband, C., R. Giegerich and A. Moller (2010). "Analyzing ambiguity of context-free grammars." Journal of Computer Programming(75): 176-191.

Bratu, D. P., B. J. Cha, M. M. Mhlanga, F. R. Kramer and S. Tyagi (2003). "Visualizing the distribution and transport of mRNAs in living cells." Proc Natl Acad Sci U S A **100**(23): 13308-13313.

Breving, K. and A. Esquela-Kerscher (2010). "The complexities of microRNA regulation: mirandering around the rules." Int J Biochem Cell Biol **42**(8): 1316-1329.

Burge, S. W., J. Daub, R. Eberhardt, J. Tate, L. Barquist, E. P. Nawrocki, S. R. Eddy, P. P. Gardner and A. Bateman (2013). "Rfam 11.0: 10 years of RNA families." Nucleic Acids Res **41**(Database issue): D226-232.

Cech, T. R. and J. A. Steitz (2014). "The Noncoding RNA Revolution-Trashing Old Rules to Forge New Ones." Cell **157**(1): 77-94.

Clarke, E. J. and V. J. Allan (2003). "Cytokeratin intermediate filament organisation and dynamics in the vegetal cortex of living Xenopus laevis oocytes and eggs." Cell Motil Cytoskeleton **56**(1): 13-26.

Comon, H., M. Dauchet, R. Gilleron, F. Jacquemard, D. Lucgiez, C. Löding, S. Tison and M. Tommasi (2007). Tree Automata Techniques and Applications.

Cormen, T. H., C. E. Leiserson and R. L. Rivest (1989). Introduction to Algorithms.

D'Inca, R., G. Marteil, F. Bazile, A. Pascal, N. Guitton, R. Lavigne, L. Richard-Parpaillon and J. Z. Kubiak (2010). "Proteomic screen for potential regulators of M-phase entry and quality of meiotic resumption in Xenopus laevis oocytes." J Proteomics **73**(8): 1542-1550.

Dallaire, P. and F. Major (2015). Exploring alternative RNA structure sets using MC-Flashfold and db2cm. Methods and Protocols (to appear). Springer. United States.

Dalli, D., A. Wilm, I. Mainz and G. Steger (2006). "STRAL: progressive alignment of non-coding RNA using base pairing probability vectors in quadratic time." Bioinformatics **22**(13): 1593-1599.

Davis, B. N., A. C. Hilyard, P. H. Nguyen, G. Lagna and A. Hata (2010). "Smad proteins bind a conserved RNA sequence to promote microRNA maturation by Drosha." Mol Cell **39**(3): 373-384.

Deigan, K. E., T. W. Li, D. H. Mathews and K. M. Weeks (2009). "Accurate SHAPE-directed RNA structure determination." Proc. Natl. Acad. Sci. USA **106**(1): 97-102.

Delisi, C. and D. M. Crothers (1971). "Prediction of RNA secondary structure." Proc Natl Acad Sci U S A **68**(11): 2682-2685.

Denli, A. M., B. B. Tops, R. H. Plasterk, R. F. Ketting and G. J. Hannon (2004). "Processing of primary microRNAs by the Microprocessor complex." Nature **432**(7014): 231-235.

Dethoff, E. A., J. Chugh, A. M. Mustoe and H. M. Al-Hashimi (2012). "Functional complexity and regulation through RNA dynamics." Nature **482**(7385): 322-330.

Dethoff, E. A., K. Petzold, J. Chugh, A. Casiano-Negroni and H. M. Al-Hashimi (2012). "Visualizing transient low-populated structures of RNA." Nature **491**(7426): 724-728.

Ding, Y. (2006). "Statistical and Bayesian approaches to RNA secondary structure prediction." RNA **12**(3): 323-331.

Ding, Y. and C. E. Lawrence (2001). "Statistical prediction of single-stranded regions in RNA secondary structure and application to predicting effective antisense target sites and beyond." Nucleic Acids Res **29**(5): 1034-1046.

Ding, Y. E., C. Y. Chan and C. E. Lawrence (2005). "RNA secondary structure prediction by centroids in a Boltzmann weighted ensemble." RNA **11**(8): 1157-1166.

Dobrzynski, M., P. Bernatowicz, M. Kloc and J. Z. Kubiak (2011). "Evolution of bet-hedging mechanisms in cell cycle and embryo development stimulated by weak linkage of stochastic processes." Results Probl Cell Differ **53**: 11-30.

Doshi, K. J., J. J. Cannone, C. W. Cobaugh and R. R. Gutell (2004). "Evaluation of the suitability of free-energy minimization using nearest-neighbor energy parameters for RNA secondary structure prediction." BMC Bioinformatics **5**: 105.

Dowell, R. D. and S. R. Eddy (2004). "Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction." BMC Bioinformatics **5**: 71.

Duan, R., C. Pak and P. Jin (2007). "Single nucleotide polymorphism associated with mature miR-125a alters the processing of pri-miRNA." Hum. Mol. Genet. **16**(9): 1124-1131.

Elinson, R. P., M. L. King and C. Forristall (1993). "Isolated vegetal cortex from Xenopus oocytes selectively retains localized mRNAs." Dev Biol **160**(2): 554-562.

Felden, B. (2007). "RNA structure: experimental analysis." Curr Opin Microbiol **10**(3): 286-291.

Ferreon, A. C., J. C. Ferreon, P. E. Wright and A. A. Deniz (2013). "Modulation of allostery by protein intrinsic disorder." Nature **498**(7454): 390-394.

Finnegan, E. F. and A. E. Pasquinelli (2013). "MicroRNA biogenesis: regulating the regulators." Crit Rev Biochem Mol Biol **48**(1): 51-68.

Flamm, C., I. L. Hovacker, P. F. Stadler and M. T. Wolfinger (2002). "Barrier Trees of Degenerate Landscapes." Z. Phys. Chem. **216**: 155-173.

Fontana, W., P. F. Stadler, E. G. Bornberg-Bauer, T. Griesmacher, I. L. Hofacker, M. Tacker, P. Tarazona, E. D. Weinberger and P. Schuster (1993). "RNA folding and combinatory landscapes." Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics **47**(3): 2083-2099.

Gendron, P., S. Lemieux and F. Major (2001). "Quantitative analysis of nucleic acid three-dimensional structures." J Mol Biol **308**(5): 919-936.

Giegerich, R. and C. Meyer (2002). Algebraic dynamic programming. Algebraic Methodolgy And Software Technology. Berlin Heidelberg, Springer**:** 349-364.

Giegerich, R., C. Meyer and P. Steffen (2004). "A discipline of dynamic programming over sequence data." Science of Computer Programming(51): 215-263.

Gkogkas, C. G., A. Khoutorsky, I. Ran, E. Rampakakis, T. Nevarko, D. B. Weatherill, C. Vasuta, S. Yee, M. Truitt, P. Dallaire, F. Major, P. Lasko, D. Ruggero, K. Nader, J. C. Lacaille and N. Sonenberg (2013). "Autism-related deficits via dysregulated eIF4E-dependent translational control." Nature **493**(7432): 371-377.

Goldsmith, C. S. and S. E. Miller (2009). "Modern uses of electron microscopy for detection of viruses." Clin Microbiol Rev **22**(4): 552-563.

Gotoh, O. (1982). "An Improved Algorithm for Matching Biological Sequences." Journal of Molecular Biology **162**: 705-708.

Gotoh, O. (1990). "Optimal sequence alignment allowing for long gaps." Bull Math Biol **52**(3): 359-373.

Gralla, J., J. A. Steitz and D. M. Crothers (1974). "Direct physical evidence for secondary structure in an isolated fragment of R17 bacteriophage mRNA." Nature **248**(445): 204-208.

Gregory, R. I., K. P. Yan, G. Amuthan, T. Chendrimada, B. Doratotaj, N. Cooch and R. Shiekhattar (2004). "The Microprocessor complex mediates the genesis of microRNAs." Nature **432**(7014): 235-240.

Grishok, A., A. E. Pasquinelli, D. Conte, N. Li, S. Parrish, I. Ha, D. L. Baillie, A. Fire, G. Ruvkun and C. C. Mello (2001). "Genes and mechanisms related to RNA interference regulate expression of the small temporal RNAs that control C. elegans developmental timing." Cell **106**(1): 23-34.

Guil, S. and J. F. Caceres (2007). "The multifunctional RNA-binding protein hnRNP A1 is required for processing of miR-18a." Nat Struct Mol Biol **14**(7): 591-596.

Halvorsen, M., J. S. Martin, S. Broadaway and A. Laederach (2010). "Disease-associated mutations that alter the RNA structural ensemble." PLoS Genet. **6**(8): e1001074.

Han, J., Y. Lee, K. H. Yeom, J. W. Nam, I. Heo, J. K. Rhee, S. Y. Sohn, Y. Cho, B. T. Zhang and V. N. Kim (2006). "Molecular basis for the recognition of primary microRNAs by the Drosha-DGCR8 complex." Cell **125**(5): 887-901.

Harrison, M. A. (1978). Introduction to Formal Language Theory. Addison-Wesley.

Hart, P. E., N. J. Nilsson and B. Raphael (1968). "A Formal Basis for the Heuristic Determination of Minimum Cost Paths." IEEE Transactions on Systems Science and Cybernetics SSC4 **4**(2): 100-107.

Heasman, J., O. Wessely, R. Langland, E. J. Craig and D. S. Kessler (2001). "Vegetal localization of maternal mRNAs is disrupted by VegT depletion." Dev Biol **240**(2): 377-386.

Helmann, J. D. (2007). "Measuring metals with RNA." Mol Cell **27**(6): 859-860.

Heo, I., C. Joo, J. Cho, M. Ha, J. Han and V. N. Kim (2008). "Lin28 mediates the terminal uridylation of let-7 precursor microRNA." Mol. Cell. **32**(2): 276-284.

Hofacker, I. L., S. H. Bernhart and P. F. Stadler (2004). "Alignment of RNA base pairing probability matrices." Bioinformatics **20**(14): 2222-2227.

Hofacker, I. L., W. Fontana, P. F. Stadler, S. Bonhoeffer, M. Tacker and P. Schuster (1994). "Fast Folding and Comparison of RNA Secondary Strutures." Monatshefte f. Chemie **125**: 167-188.

Honer zu Siederdissen, C., S. H. Bernhart, P. F. Stadler and I. L. Hofacker (2011). "A folding algorithm for extended RNA secondary structures." Bioinformatics **27**(13): i129-136.

Huang, L. and D. Chiang (2005). "Better k-best Parsing." Proceedings of the Ninth International Workshop of Parsing Technologies (IWPT): 53-64.

Hutvagner, G., J. McLachlan, A. E. Pasquinelli, E. Balint, T. Tuschl and P. D. Zamore (2001). "A cellular function for the RNA-interference enzyme Dicer in the maturation of the let-7 small temporal RNA." Science **293**(5531): 834-838.

Isambert, H. (2009). "The jerky and knotty dynamics of RNA." Methods **49**(2): 189-196.

Janssen, S. and R. Giegerich (2010). "Faster computation of exact RNA shape probabilities." Bioinformatics **26**(5): 632-639.

Jenny, A., O. Hachet, P. Zavorszky, A. Cyrklaff, M. D. Weston, D. S. Johnston, M. Erdelyi and A. Ephrussi (2006). "A translation-independent role of oskar RNA in early Drosophila oogenesis." Development **133**(15): 2827-2833.

Karp, R. M. (1972). Reducibility among combinatorial problems. Complexity of Computer Computations. R. E. Miller and J. W. Thatcher, Plenum**:** 85-103.

Ketting, R. F., S. E. Fischer, E. Bernstein, T. Sijen, G. J. Hannon and R. H. Plasterk (2001). "Dicer functions in RNA interference and in synthesis of small RNA involved in developmental timing in C. elegans." Genes Dev **15**(20): 2654-2659.

Kladwang, W. and R. Das (2010). "A mutate-and-map strategy for inferring base pairs in structured nucleic acids: proof of concept on a DNA/RNA helix." Biochemistry **49**(35): 7414-7416.

Kladwang, W., C. C. VanLang, P. Cordero and R. Das (2011). "A two-dimensional mutate-and-map strategy for non-coding RNA structure." Nat Chem **3**(12): 954-962.

Kloc, M. (2008). "Emerging novel functions of RNAs, and binary phenotype?" Dev Biol **317**(2): 401-404.

Kloc, M. (2009). "Teachings from the egg: new and unexpected functions of RNAs." Mol Reprod Dev **76**(10): 922-932.

Kloc, M., S. Bilinski and M. T. Dougherty (2007). "Organization of cytokeratin cytoskeleton and germ plasm in the vegetal cortex of Xenopus laevis oocytes depends on coding and non-coding RNAs: three-dimensional and ultrastructural analysis." Exp Cell Res **313**(8): 1639-1651.

Kloc, M., P. Dallaire, A. Reunov and F. Major (2011). "Structural messenger RNA contains cytokeratin polymerization and depolymerization signals." Cell Tissue Res **346**(2): 209-222.

Kloc, M., M. T. Dougherty, S. Bilinski, A. P. Chan, E. Brey, M. L. King, C. W. Patrick, Jr. and L. D. Etkin (2002). "Three-dimensional ultrastructural analysis of RNA distribution within germinal granules of Xenopus." Dev Biol **241**(1): 79-93.

Kloc, M. and L. D. Etkin (1994). "Delocalization of Vg1 mRNA from the vegetal cortex in Xenopus oocytes after destruction of Xlsirt RNA." Science **265**(5175): 1101-1103.

Kloc, M. and L. D. Etkin (2005). "RNA localization mechanisms in oocytes." J Cell Sci **118**(Pt 2): 269-282.

Kloc, M., V. Foreman and S. A. Reddy (2011). "Binary function of mRNA." Biochimie **93**(11): 1955-1961.

Kloc, M., G. Spohr and L. D. Etkin (1993). "Translocation of repetitive RNA sequences with the germ plasm in Xenopus oocytes." Science **262**(5140): 1712-1714.

Kloc, M., K. Wilk, D. Vargas, Y. Shirato, S. Bilinski and L. D. Etkin (2005). "Potential structural role of non-coding and coding RNAs in the organization of the cytoskeleton at the vegetal cortex of Xenopus oocytes." Development **132**(15): 3445-3457.

Klymkowsky, M. W. and L. A. Maynell (1989). "MPF-induced breakdown of cytokeratin filament organization in the maturing Xenopus oocyte depends upon the translation of maternal mRNAs." Dev Biol **134**(2): 479-485.

Klymkowsky, M. W., L. A. Maynell and C. Nislow (1991). "Cytokeratin phosphorylation, cytokeratin filament severing and the solubilization of the maternal mRNA Vg1." J Cell Biol **114**(4): 787-797.

Kondo, J. and E. Westhof (2008). "The bacterial and mitochondrial ribosomal A-site molecular switches possess different conformational substates." Nucleic Acids Res **36**(8): 2654-2666.

Kruspe, M. and P. F. Stadler (2007). "Progressive multiple sequence alignments from triplets." BMC Bioinformatics **8**: 254.

Lai, D., J. R. Proctor and I. M. Meyer (2013). "On the importance of cotranscriptional RNA structure formation." RNA **19**(11): 1461-1473.

Lecuyer, E., H. Yoshida, N. Parthasarathy, C. Alm, T. Babak, T. Cerovina, T. R. Hughes, P. Tomancak and H. M. Krause (2007). "Global analysis of mRNA localization reveals a prominent role in organizing cellular architecture and function." Cell **131**(1): 174-187.

Lee, R. C., R. L. Feinbaum and V. Ambros (1993). "The C. elegans heterochronic gene lin-4 encodes small RNAs with antisense complementarity to lin-14." Cell **75**: 843-854.

Lee, Y., C. Ahn, J. Han, H. Choi, J. Kim, J. Yim, J. Lee, P. Provost, O. Radmark, S. Kim and V. N. Kim (2003). "The nuclear RNase III Drosha initiates microRNA processing." Nature **425**(6956): 415-419.

Lemieux, S. and F. Major (2006). "Automated extraction and classification of RNA tertiary structure cyclic motifs." Nucleic acids research **34**(8): 2340-2346.

Li, J., T. Kim, R. Nutiu, D. Ray, T. R. Hughes and Z. Zhang (2014). "Identifying mRNA sequence elements for target recognition by human Argonaute proteins." Genome Res.

Li, W., R. Duan, F. Kooy, S. L. Sherman, W. Zhou and P. Jin (2009). "Germline mutation of microRNA-125a is associated with breast cancer." J. Med. Genet. **46**(5): 358-360.

Lorenz, W. A. and P. Clote (2011). "Computing the partition function for kinetically trapped RNA secondary structures." PLoS One **6**(1): e16178.

Lund, E., S. Guttinger, A. Calado, J. E. Dahlberg and U. Kutay (2004). "Nuclear export of microRNA precursors." Science **303**(5654): 95-98.

Lyngso, R. B. and C. N. Pedersen (2000). "RNA pseudoknot prediction in energy-based models." J Comput Biol **7**(3-4): 409-427.

Major, F., M. Turcotte, D. Gautheret, G. Lapalme, E. Fillion and R. Cedergren (1991). "The combination of symbolic and numerical computation for three-dimensional modeling of RNA." Science **253**(5025): 1255-1260.

Mandal, M. and R. R. Breaker (2004). "Gene regulation by riboswitches." Nat Rev Mol Cell Biol **5**(6): 451-463.

Manosas, M., J. D. Wen, P. T. Li, S. B. Smith, C. Bustamante, I. Tinoco, Jr. and F. Ritort (2007). "Force unfolding kinetics of RNA using optical tweezers. II. Modeling experiments." Biophysical journal **92**(9): 3010-3021.

Marteil, G., R. D'Inca, A. Pascal, N. Guitton, T. Midtun, A. Goksoyr, L. Richard-Parpaillon and J. Z. Kubiak (2010). "EP45 accumulates in growing Xenopus laevis oocytes and has oocyte-maturation-enhancing activity involved in oocyte quality." J Cell Sci **123**(Pt 10): 1805-1813.

Martinez-Rucobo, F. W. and P. Cramer (2013). "Structural basis of transcription elongation." Biochim Biophys Acta **1829**(1): 9-19.

Mathews, D. H. (2004). "Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization." Rna **10**(8): 1178-1190.

Mathews, D. H., W. N. Moss and D. H. Turner (2010). "Folding and finding RNA secondary structure." Cold Spring Harb Perspect Biol **2**(12): a003665.

Mathews, D. H., J. Sabina, M. Zuker and D. H. Turner (1999). "Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure." J Mol Biol **288**(5): 911-940.

McCaskill, J. S. (1990). "The equilibrium partition function and base pair binding probabilities for RNA secondary structure." Biopolymers **29**(6-7): 1105-1119.

Michlewski, G., S. Guil, C. A. Semple and J. F. Caceres (2008). "Posttranscriptional regulation of miRNAs harboring conserved terminal loops." Mol Cell **32**(3): 383-393.

Nawrocki, E. P. and S. R. Eddy (2013). "Infernal 1.1: 100-fold faster RNA homology searches." Bioinformatics **29**(22): 2933-2935.

Nebel, M. E. and A. Scheid (2011). "Evaluation of a sophisticated SCFG design for RNA secondary structure prediction." Theory Biosci **130**(4): 313-336.

Nikolova, E. N., E. Kim, A. A. Wise, P. J. O'Brien, I. Andricioaei and H. M. Al-Hashimi (2011). "Transient Hoogsteen base pairs in canonical duplex DNA." Nature **470**(7335): 498-502.

Nussinov, R. and A. B. Jacobson (1980). "Fast algorithm for predicting the secondary structure of single-stranded RNA." Proc. Natl. Acad. Sci. USA **77**(11): 6309-6313.

Nussinov, R., G. Pieczenik, J. R. Griggs and D. J. Kleitman (1978). "Algorithms For Loop Matchings." SIAM J. Appl. Math. **35**(1): 68-82.

O'Connor, M., C. L. Thomas, R. A. Zimmermann and A. E. Dahlberg (1997). "Decoding fidelity at the ribosomal A and P sites: influence of mutations in three different regions of the decoding domain in 16S rRNA." Nucleic Acids Res **25**(6): 1185-1193.

Onoa, B., S. Dumont, J. Liphardt, S. B. Smith, I. Tinoco, Jr. and C. Bustamante (2003). "Identifying kinetic barriers to mechanical unfolding of the T. thermophila ribozyme." Science **299**(5614): 1892-1895.

Parisien, M. and F. Major (2008). "The MC-Fold and MC-Sym pipeline infers RNA structure from sequence data." Nature **452**(7183): 51-55.

Parisien, M. and F. Major (2012). "Determining RNA three-dimensional structures using low-resolution data." J Struct Biol **179**(3): 252-260.

Pearson, W. R. (2000). "Flexible sequence similarity searching with the FASTA3 program package." Methods Mol. Biol. **132**: 185-219.

Pipas, J. M. and J. E. McMahon (1975). "Method for predicting RNA secondary structure." Proc Natl Acad Sci U S A **72**(6): 2017-2021.

Quarles, K. A., D. Sahu, M. A. Havens, E. R. Forsyth, C. Wostenberg, M. L. Hastings and S. A. Showalter (2013). "Ensemble analysis of primary microRNA structure reveals an extensive capacity to deform near the Drosha cleavage site." Biochemistry **52**(5): 795-807.

Ray, P. S., J. Jia, P. Yao, M. Majumder, M. Hatzoglou and P. L. Fox (2009). "A stress-responsive RNA switch regulates VEGFA expression." Nature **457**(7231): 915-919.

Raychaudhuri, S. (2011). "Mapping rare and common causal alleles for complex human diseases." Cell **147**(1): 57-69.

Reeder, J. and R. Giegerich (2004). "Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics." BMC Bioinformatics **5**: 104.

Reyes, F. E., A. D. Garst and R. T. Batey (2009). "Strategies in RNA crystallography." Methods Enzymol **469**: 119-139.

Rivas, E. (2013). "The four ingredients of single-sequence RNA secondary structure prediction. A unifying perspective." RNA Biol **10**(7): 1185-1196.

Rivas, E. and S. R. Eddy (1999). "A dynamic programming algorithm for RNA structure prediction including pseudoknots." J Mol Biol **285**(5): 2053-2068.

Romby, P. and E. G. Wagner (2008). "Exploring the complex world of RNA regulation." Biol Cell **100**(1): e1-3.

Rozenberg, G. and A. Salomaa (1997). Handbook of Formal Languages (Beyond Words).

Sabarinathan, R., H. Tafer, S. E. Seemann, I. L. Hofacker, P. F. Stadler and J. Gorodkin (2013a). "The RNAsnp web server: predicting SNP effects on local RNA secondary structure." Nucleic Acids Res **41**(Web Server issue): W475-479.

Sabarinathan, R., H. Tafer, S. E. Seemann, I. L. Hofacker, P. F. Stadler and J. Gorodkin (2013b). "RNAsnp: efficient detection of local RNA secondary structure changes induced by SNPs." Hum Mutat **34**(4): 546-556.

Saenger, W. (1984). Principles of nucleic acid structure. Cantor, C. R., Springer-Verlag New-York Inc.

Sauthoff, G., M. Mohl, S. Janssen and R. Giegerich (2013). "Bellman's GAP--a language and compiler for dynamic programming in sequence analysis." Bioinformatics **29**(5): 551-560.

Sayers, E. W., T. Barrett, D. A. Benson, E. Bolton, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. Dicuccio, S. Federhen, M. Feolo, I. M. Fingerman, L. Y. Geer, W. Helmberg, Y. Kapustin, S. Krasnov, D. Landsman, D. J. Lipman, Z. Lu, T. L. Madden, T. Madej, D. R. Maglott, A. Marchler-Bauer, V. Miller, I. Karsch-Mizrachi, J. Ostell, A. Panchenko, L. Phan, K. D. Pruitt, G. D. Schuler, E. Sequeira, S. T. Sherry, M. Shumway, K. Sirotkin, D. Slotta, A. Souvorov, G. Starchenko, T. A. Tatusova, L. Wagner, Y. Wang, W. J. Wilbur, E. Yaschenko and J. Ye (2012). "Database resources of the National Center for Biotechnology Information." Nucleic Acids Res **40**(Database issue): D13-25.

Sikkel, K. (1997). Parsing Schemata, Springer-Verlag.

Sim, A. Y., P. Minary and M. Levitt (2012). "Modeling nucleic acids." Curr Opin Struct Biol **22**(3): 273-278.

Sipser, M. (1997). Introduction to the theory of computation.

Song, X., M. Wang, Y. P. Chen, H. Wang, P. Han and H. Sun (2013). "Prediction of pre-miRNA with multiple stem-loops using pruning algorithm." Comput Biol Med **43**(5): 409-416.

Sripakdeevong, P., M. Cevec, A. T. Chang, M. C. Erat, M. Ziegeler, Q. Zhao, G. E. Fox, X. Gao, S. D. Kennedy, R. Kierzek, E. P. Nikonowicz, H. Schwalbe, R. K. Sigel, D. H. Turner and R. Das (2014). "Structure determination of noncanonical RNA motifs guided by (1)H NMR chemical shifts." Nat Methods **11**(4): 413-416.

Studnicka, G. M., G. M. Rahn, I. W. Cummings and W. A. Salser (1978). "Computer method for predicting the secondary structure of single-stranded RNA." Nucleic Acids Res **5**(9): 3365-3387.

Sukosd, Z., M. S. Swenson, J. Kjems and C. E. Heitsch (2013). "Evaluating the accuracy of SHAPE-directed RNA secondary structure predictions." Nucleic Acids Res **41**(5): 2807-2816.

Sun, Q., H. Gu, Y. Zeng, Y. Xia, Y. Wang, Y. Jing, L. Yang and B. Wang (2010). "Hsa-mir-27a genetic variant contributes to gastric cancer susceptibility through affecting miR-27a and target gene expression." Cancer Sci. **101**(10): 2241-2247.

Tinoco, I., Jr. (2004). "Force as a useful variable in reactions: unfolding RNA." Annu Rev Biophys Biomol Struct **33**: 363-385.

Tinoco, I., Jr., O. C. Uhlenbeck and M. D. Levine (1971). "Estimation of secondary structure in ribonucleic acids." Nature **230**(5293): 362-367.

Turner, D. H. and D. H. Mathews (2010). "NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure." Nucleic Acids Res **38**(Database issue): D280-282.

Van holde, K. E. (1985). Physical Biochemistry. Englewood Cliffs, NJ 07632, Prentice-Hall, Inc.

Viswanathan, S. R., G. Q. Daley and R. I. Gregory (2008). "Selective blockade of microRNA processing by Lin28." Science **320**(5872): 97-100.

Waldispuhl, J., S. Devadas, B. Berger and P. Clote (2008). "Efficient algorithms for probing the RNA mutation landscape." PLoS Comput Biol **4**(8): e1000124.

Waterman, M. S. (1978). "Secondary structure of single-stranded nucleic acids." Adv. Math. Suppl. Stud. **1**: 167-212.

Waterman, M. S. and T. H. Byers (1985). "A Dynamic Programming Algorithm to Find All Solutions in a Neighborhood of the Optimum." Mathematical Biosciences **77**: 179-188.

Waterman, M. S. and T. F. Smith (1978). "RNA secondary structure: a complete mathematical analysis." Math. Biosci. **42**: 257-266.

Waters, L. S. and G. Storz (2009). "Regulatory RNAs in bacteria." Cell **136**(4): 615-628.

Watson, J. D. and F. H. Crick (1953). "Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid." Nature **171**(4356): 737-738.

Wilkinson, K. A., E. J. Merino and K. M. Weeks (2006). "Selective 2'-hydroxyl acylation analyzed by primer extension (SHAPE): quantitative RNA structure analysis at single nucleotide resolution." Nat Protoc **1**(3): 1610-1616.

Williamson, J. R. (2000). "Induced fit in RNA-protein recognition." Nat. Struct. Biol. **7**(10): 834-837.

Wilson, R. C. and J. A. Doudna (2013). "Molecular mechanisms of RNA interference." Annu Rev Biophys **42**: 217-239.

Wimberly, B. T., D. E. Brodersen, W. M. Clemons, Jr., R. J. Morgan-Warren, A. P. Carter, C. Vonrhein, T. Hartsch and V. Ramakrishnan (2000). "Structure of the 30S ribosomal subunit." Nature **407**(6802): 327-339.

Wuchty, S., W. Fontana, I. L. Hofacker and P. Schuster (1999). "Complete suboptimal folding of RNA and the stability of secondary structures." Biopolymers **49**(2): 145-165.

Xayaphoummine, A., T. Bucher and H. Isambert (2005a). "Kinefold web server for RNA/DNA folding path and structure prediction including pseudoknots and knots." Nucleic Acids Res **33**(Web Server issue): W605-610.

Xayaphoummine, A., T. Bucher and H. Isambert (2005b). "Kinefold web server for RNA/DNA folding path and structure prediction including pseudoknots and knots." Nucleic Acids Res. **33**(Web Server issue): W605-610.

Xia, T., J. SantaLucia, Jr., M. E. Burkard, R. Kierzek, S. J. Schroeder, X. Jiao, C. Cox and D. H. Turner (1998). "Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson-Crick base pairs." Biochemistry **37**(42): 14719-14735.

Xu, Z. and G. M. Culver (2009). "Chemical probing of RNA and RNA/protein complexes." Methods Enzymol **468**: 147-165.

Yang, R., B. Schlehe, K. Hemminki, C. Sutter, P. Bugert, B. Wappenschmidt, J. Volkmann, R. Varon, B. H. Weber, D. Niederacher, N. Arnold, A. Meindl, C. R. Bartram, R. K. Schmutzler and B. Burwinkel (2010). "A genetic variant in the pre-miR-27a oncogene is associated with a reduced familial breast cancer risk." Breast Cancer Res. Treat. **121**(3): 693-702.

Yang, S., M. Parisien, F. Major and B. Roux (2010). "RNA structure determination using SAXS data." J Phys Chem B **114**(31): 10039-10048.

Yi, R., Y. Qin, I. G. Macara and B. R. Cullen (2003). "Exportin-5 mediates the nuclear export of pre-microRNAs and short hairpin RNAs." Genes Dev **17**(24): 3011-3016.

Zeng, Y. and B. R. Cullen (2003). "Sequence requirements for micro RNA processing and function in human cells." RNA **9**(1): 112-123.

Zeng, Y. and B. R. Cullen (2004). "Structural requirements for pre-microRNA binding and nuclear export by Exportin 5." Nucleic acids research **32**(16): 4776-4785.

Zhang, J. and M. L. King (1996). "Xenopus VegT RNA is localized to the vegetal cortex during oogenesis and encodes a novel T-box transcription factor involved in mesodermal patterning." Development **122**(12): 4119-4129.

Zhang, Q., R. Bettadapura and C. Bajaj (2012). "Macromolecular structure modeling from 3D EM using VolRover 2.0." Biopolymers **97**(9): 709-731.

zu Siederdissen, C. H. (2012). Sneaking around concatMap: Efficient combinators for dynamic programming. Proceedings of the 17th ACM SIGPLAN international conference on Functional programming. New York, ACM**:** 215-226.

zu Siederdissen, C. H. (2013). Grammatical Approaches to Problems in RNA Bioinformatics. Doktor der Naturwissenshaften, Universität Wien.

Zuker, M. (1989). "On finding all suboptimal foldings of an RNA molecule." Science **244**(4900): 48-52.

Zuker, M. and D. Sankoff (1984). "RNA secondary structures and their prediction." Bulletin of Mathematical Biology **46**(4): 591-621.

Zuker, M. and P. Stiegler (1981). "Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information." Nucleic Acids Res. **9**(1): 133-148.