Université de Montréal

Algorithme de comparaison de structures secondaires d'ARN

Par
Valentin Guignon

Département de Biochimie
Faculté de Médecine

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de M.Sc.
en bio-informatique

mars, 2006

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé:

Algorithme de comparaison
de structures secondaires d'ARN

présenté par
Valentin Guignon

a été évalué par un jury composé des personnes suivantes:

Miklós Csürös
président-rapporteur

Sylvie Hamel
directrice de recherche

Cedric Chauve
codirecteur

Mathieu Blanchette
membre du jury

RESUME

Ce mémoire décrit une nouvelle approche et un nouvel algorithme de comparaison de structures secondaires d'ARN. Cette première est caractérisée par une décomposition en tiges-boucles des structures permettant la mise au point d'un algorithme de programmation dynamique calculant une *distance d'édition* entre 2 tiges-boucles. La complexité de cet algorithme est de $O(n^4)$ en temps et en espace malgré l'introduction de nouvelles *opérations d'édition* et aucune contrainte sur les schémas de scores utilisés pouvant exclure une de ces opérations. L'algorithme est ensuite employé par une heuristique pour retrouver les meilleurs alignements entre deux structures. Une base de données et un serveur de comparaison ouverts au public ont été mis en place pour pouvoir tester notre méthode. Les résultats ainsi obtenus illustrent le bien fondé de nos travaux et suggèrent de nouvelles perspectives de recherches.

Mots clés: structure secondaire d'ARN, tiges-boucles, alignement, distance, opération d'édition, schéma de scores, programmation dynamique.

ABSTRACT

This bioinformatics master thesis presents a new approach and a new algorithm for RNA secondary structure comparison. This approach is characterized by a stem-loop decomposition of structures which brings a new dynamic programming algorithm that computes an edit distance between 2 stem-loops. The algorithm time and space complexity is in $O(n^4)$ in spite of the introduction of novel *edit operations* and no score scheme constraints that could exclude an operation. Then a heuristic uses the algorithm to find the bests alignments between two structures. To test our method, we built a public database and a public comparison server. Obtained results show the well-fund of our work and suggest new research perspectives.

Keywords: RNA secondary structure, stem-loops, alignment, distance, edit operation, score scheme, dynamic programming.

TABLE DES MATIERES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES

ADN	Acide DésoxyriboNucléique
ANSI	American National Standards Institute
ARN	Acide RiboNucléique (voir annexe pour les différents types d'ARN)
BLAST	Basic Local Alignment Search Tool
INFERNAL	INFErence of RNA secondary structure aLignment
ISO	du grec "iso" (et non pas International Standards Organization!), signifiant égal
IUB	International Union of Biochemistry
IUPAC	International Union of Pure and Applied Chemistry
PCR	Polymerase Chain Reaction
RFAM	RNA Families Database of Alignments and Covariance Models
RNAStrAT	RNA Structure Analysis Toolkit
RISC	RNA-induced silencing complexe
SQL	Structured Query Language

NOTATIONS

T_i	: un arbre.
$\|T_i\|$	: taille de l'arbre T_i (son nombre de nœuds).
L_i	: nombre de feuilles de l'arbre T_i.
D_i	: profondeur de l'arbre T_i.
I_i	: degré maximal de l'arbre T_i.

à mes parents qui m'ont donné les moyens de réussir

REMERCIEMENTS

Mes remerciements vont tout d'abord à Sylvie, ma directrice et amie, qui m'a donné ma chance en acceptant de me diriger alors qu'elle ne disposait que de peu d'éléments sur moi. Sans sa confiance, son enthousiasme et sa générosité, mon séjour à Montréal n'aurait jamais été aussi enrichissant, agréable et productif.

Je tiens ensuite à remercier Cedric, mon co-directeur et ami, qui par sa rigueur et son savoir faire m'a poussé à remettre en question ce que je donnais pour évident et à avoir une démarche un peu plus prudente.

Je remercie ensuite Génome Québec pour avoir financé mes travaux par le biais d'une bourse de co-direction. Cet argent m'a permis d'aborder mes recherches plus sereinement.

Je remercie ma famille qui m'a offert les moyens de réussir et qui me donne tous les jours les motivations pour faire de mon mieux.

Un merci particulier à Emmanuelle pour son aide.

Je remercie enfin mes amis de bio-informatique ainsi que ceux de la Clef-des-Champs, qui ont fait de mon passage au Québec et à Montréal une expérience inoubliable et peut-être même la meilleure de ma vie. Leur contribution à mon bien-être m'a donné l'énergie dont j'avais besoin pour réussir mes études et m'épanouïr. Les amis comme vous sont rares et précieux!

Chapitre 1

INTRODUCTION

Les êtres vivants sont très complexes et comprendre leur fonctionnement est un défi en biochimie. D'innombrables mécanismes biologiques sont mis en œuvre pour perpétrer la vie et la maintenir. Au cours des 20 dernières années, avec l'apparition de la technique d'amplification génique PCR (1984), le volume des données biologiques, issues des programmes de séquençages entre autres, a explosé posant un nouveau problème, celui du traitement de ces données.

Pour pouvoir être capable de manipuler et d'utiliser toutes ces données cumulées, les biochimistes ont dû faire appel à l'informatique spécialisée, donnant ainsi naissance au domaine de la bio-informatique. Parmi les nombreux thèmes abordés par cette jeune discipline, il y a celui de l'étude des *ARN**.

Les acides ribonucléiques sont présents dans toutes les cellules vivantes et chez de nombreux virus. Il existe deux grandes catégories d'ARN: les ARN codants qui sont issus de la transcription de gènes de l'ADN (acide désoxyribonucléique) et qui peuvent être traduits en protéines pour exprimer ces gènes, et les ARN non-codants qui ne sont pas traduits en protéines mais qui peuvent effectuer diverses autres fonctions. Les activités exercées par ces derniers dépendent de leur conformation. C'est pourquoi, l'étude de leur structure spatiale est incontournable pour appréhender leur fonctionnement.

Malheureusement, il est difficile d'obtenir des informations sur les structures tridimensionnelles des ARN car ces molécules sont généralement très flexibles et sensibles à leur milieu. En revanche, il est plus facile de déterminer chimiquement ou de calculer certains liens intramoléculaires appelés *appariements de bases*. Ils peuvent être représentés sous la forme d'une

*: tous les mots en italique sont définis dans le glossaire situé en annexe.

structure qualifiée de *secondaire*. Pour étudier et caractériser ces structures, il est possible de les comparer entre elles.

Les premiers algorithmes destinés à comparer les structures secondaires d'ARN datent des années 80. L'algorithme de Zhang et Shasha [42] publié en 1989 permettait cette comparaison en modélisant les structures sous forme d'arbres. Une amélioration réduisant les calculs du pire cas de cet algorithme a été proposée par Klein [18] 9 ans plus tard. Récemment une autre méthode proposée par Liu et al. [28] a permis d'effectuer cette même tâche avec une complexité encore plus réduite en $O(n^2)$. Cependant, ces trois algorithmes n'apportaient qu'une réponse partielle au problème de la comparaison de structures secondaires car ils ne permettaient pas de détecter certains types de modifications structurelles. Un premier pas vers la prise en compte de ces *opérations d'édition* a été proposé par Jiang et al. [22]. Les nouvelles opérations introduites rendent cependant le *probleme NP-complet* dans un cadre général. D'autres opérations encore plus complexes ont été introduites dans les calculs de comparaison par Allali et Sagot [1]. En contrepartie, les calculs ont été alourdis. Parallèlement à la comparaison globale de structures, d'autres algorithmes destinés à la recherche de petits motifs ont été développés comme celui de Hochsmann et al. [18].

Bien qu'il existe maintenant un certain nombre d'outils dédiés à la comparaison de structures secondaires, plusieurs aspects restent à améliorer. De plus, alors que la comparaison massive de séquences est possible en temps raisonnable grâce à des algorithmes ou heuristiques comme BLAST [2], aucun équivalent n'est disponible pour les structures secondaires.

Pour répondre à ce dernier problème, nous avons tout d'abord envisagé de "*bit-vectoriser*" un algorithme existant. Notre approche a été de commencer par décomposer les structures secondaires complexes en sous-éléments plus faciles à manipuler: les tiges-boucles. L'idée sous-jacente à cette approche est d'utiliser les alignements entre sous-éléments comme des ancres pour obtenir plus

rapidement un alignement global aussi proche que possible d'un alignement optimal. La simplicité structurelle des tiges-boucles nous a finalement menés à un nouvel algorithme capable d'intégrer de nouvelles *opérations d'édition* sans trop augmenter la complexité.

Pour tester l'efficacité et la fiabilité de notre algorithme, nous avions besoin d'un vaste ensemble de données. Plusieurs sites Internet fournissent des structures secondaires et il s'agit souvent de structures consensus construites à partir d'alignements multiples comme c'est le cas pour la RFam [14]. Puisque ces données sont dispersées et dans des formats variés peu propices aux calculs, nous avons jugé nécessaire de créer une nouvelle base de données regroupant ces structures secondaires d'ARN dans un même format optimisé pour un traitement informatique. Enfin, nous avons mis en place un serveur doté de plusieurs outils dédiés à la comparaison de structures et fournissant en même temps un accès public à cette base de données.

Le second chapitre de ce mémoire sera consacré à la présentation de l'ARN. Nous verrons tout d'abord les caractéristiques de cette molécule, ses différents rôles et nous aborderons quelques exemples de familles d'ARN. Puis nous discuterons des perspectives de recherches sur ces biomolécules, comment elles peuvent être représentées et l'intérêt de leur comparaison sous forme de structures secondaires.

Le troisième chapitre exposera l'état de l'art de la comparaison des structures secondaires d'ARN. Nous commencerons par recenser les différents types de comparaisons et leur finalité. Nous continuerons par un descriptif exhaustif des façons de modéliser les structures secondaires. Nous poursuivrons par l'étude des *opérations d'édition* ainsi que la façon de les quantifier. Pour terminer, les principaux algorithmes de comparaison de structures secondaires d'ARN seront passés en revus.

Le chapitre suivant sera consacré à l'algorithme que nous avons développé, à la base de données et au serveur de comparaison. Des résultats viendront illustrer le bien fondé de nos travaux.

Enfin, nous conclurons avec un regard critique sur notre contribution au domaine de l'étude des ARN et les perspectives ouvertes à l'issue de notre recherche.

Chapitre 2

L'ARN

2.1 La molécule d'ARN

L'ARN ou acide ribonucléique est avant tout un polymère composé d'une longue chaîne linéaire de nucléotides de 4 types: adénine, uracile, guanine et cytosine. Ces 4 acides nucléiques, symbolisés respectivement par les lettres **A**, **U**, **G** et **C**, sont présents dans un ARN en quantité variable. Ils partagent tous la même structure composée d'une base azotée, d'un sucre (ribose) et d'un phosphate. Leur polymérisation se fait en liant de façon covalente le ribose d'un acide nucléique au phosphate d'un autre acide nucléique. La macromolécule ainsi formée peut être vue comme une séquence de nucléotides comportant deux extrémités; l'une appelée 5' correspondant au carbone 5' du sucre sur lequel est normalement lié le phosphate effectuant la liaison avec l'acide nucléique précédent, et l'autre appelée 3' correspondant au carbone 3' du sucre sur lequel vient normalement se fixer le phosphate de l'acide nucléique suivant (Figure 1).

Comme les nucléotides sont des résidus présentant des charges, la molécule d'ARN va avoir tendance à se replier sur elle-même de façon à maximiser la stabilisation de ces charges en les annulant entre elles. Ce repliement obéit à des règles physiques précises et va dépendre des conditions environnementales et de la séquence nucléotidique. À cause de leurs cycles carbonés leur conférant une conformation planaire, les bases azotées vont avoir tendance à s'empiler. De plus, l'adénine et l'uracile présentent chacune deux dipôles qui peuvent être mis en vis-à-vis de différentes façons pour se complémenter et former des ponts hydrogènes. Il en est de même pour la guanine et la cytosine qui comportent quant à elles 3 dipôles. Ces bases vont ainsi former des paires stables dont les

plus courantes sont les paires dites "Watson-Crick" A-U et G-C (Figure 2a.).

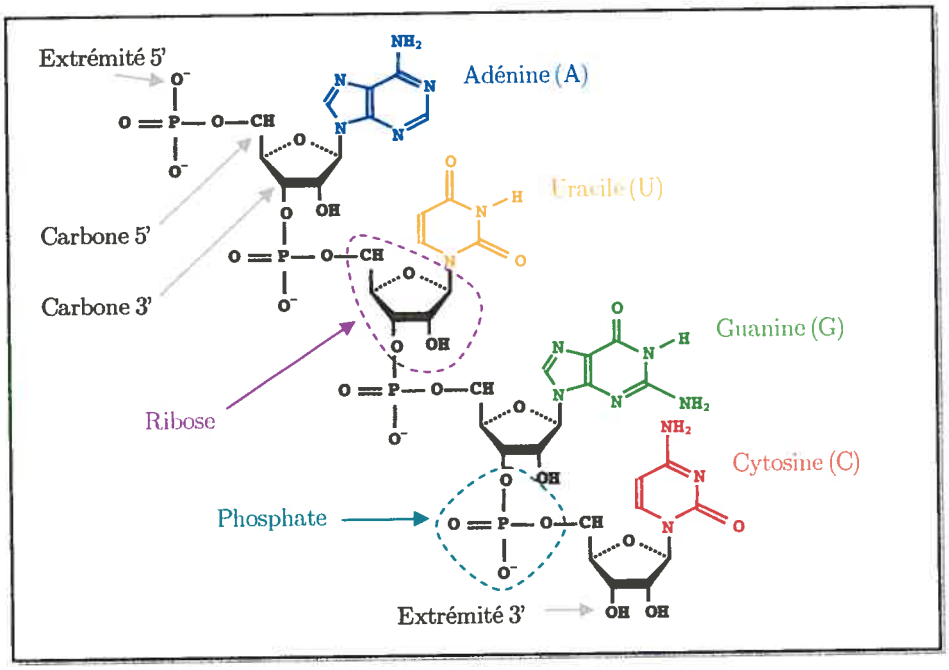


Figure 1 Composition d'un acide ribonucléique. (inspirée de [39])

Il existe de nombreux autres types d'*appariements* ([27]) comme les paires dites de "Wobble" G-U moins stables donc moins répandues (Figure 2b.).

Une fois repliée sur elle-même, la molécule d'ARN adopte une forme qui ne varie que très peu. Cette conformation peut lui permettre de participer à un processus biochimique particulier. Les molécules d'ARN peuvent jouer de très nombreux rôles allant de la catalyse de réactions chimiques (RNase P) au transport de l'information génétique (ARNm) en passant par l'inhibition de gènes (miARN), mais il est rare qu'un même ARN assume plus d'un de ces rôles (ARNtm).

2.2 La théorie du monde à ARN

Tout d'abord relayé par les chercheurs au rang de simple intermédiaire de l'information génétique et d'outil pour l'exprimer, l'ARN a vu son rôle dans la

vie cellulaire réévalué en particulier au cours de ces 10 dernières années au point

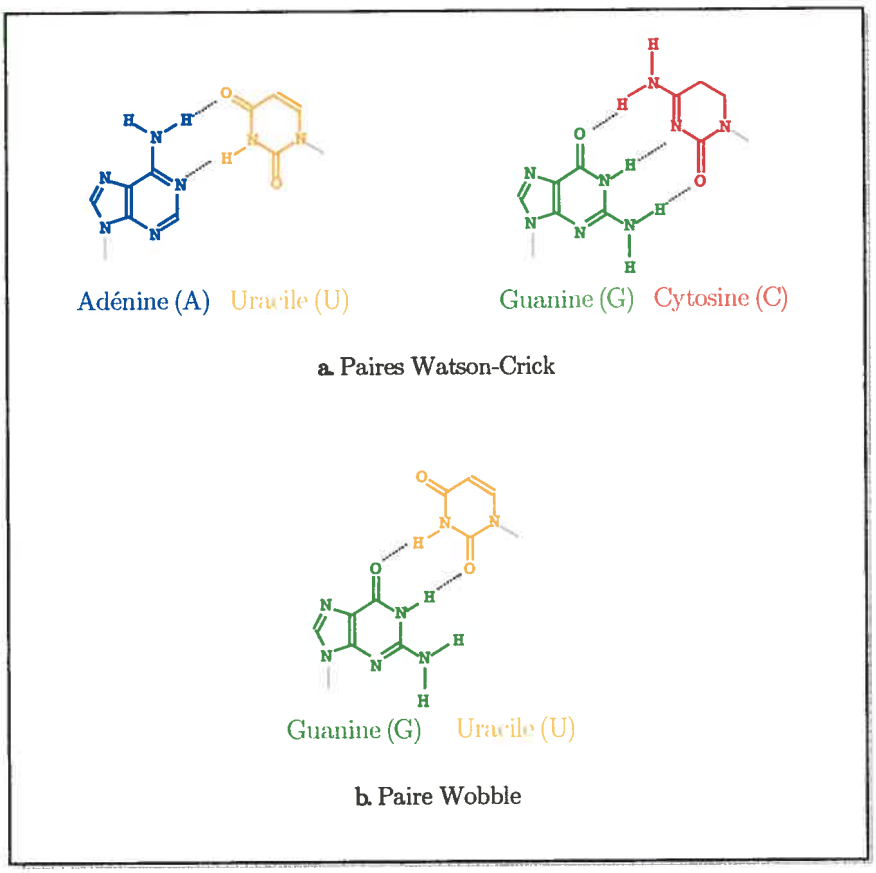


Figure 2 Appariements de ribonucléotides les plus courants. Les ponts hydrogènes sont symbolisés par des pointillés. (inspirée de [39])

de devenir un élément, sinon l'Élément central de la vie. Plusieurs découvertes laissent supposer que l'ARN aurait été à l'origine même de l'apparition de la vie sur Terre ([13] et [40]). Selon cette théorie, les premières molécules capables de se répliquer auraient été des ARN primitifs composés d'une variété de bases plus riche que celle que l'on rencontre actuellement chez les êtres vivants. Ces ARN primitifs auraient devancé l'utilisation des acides aminés par le vivant (Figure 3).

Les ARN encore présents de nos jours représenteraient les vestiges de ces temps reculés. Aujourd'hui, l'ADN a été identifié comme le support de l'information génétique, les protéines comme l'expression des gènes et l'ARN

comme le principal intermédiaire entre l'ADN et les protéines, ce qui forme le dogme central de la biologie moléculaire (Figure 4).

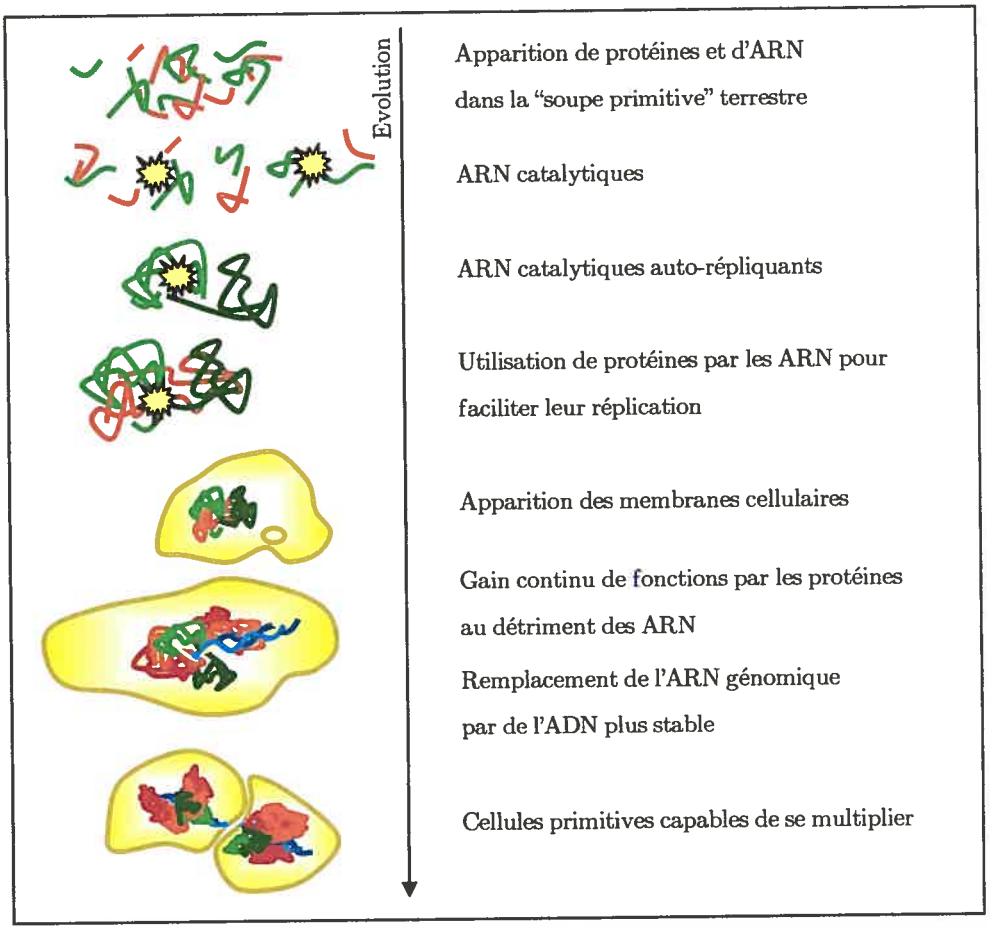


Figure 3 Une théorie de l'origine de l'apparition de la vie à partir des molécules d'ARN.

Cependant, ce schéma a été grandement remis en question et même modifié par des découvertes récentes. Tout d'abord, il a été démontré que les ARN sont plus nombreux que ce que l'on croyait au départ et assument une plus grande variété de tâches. Ils sont capables notamment de catalyser diverses réactions chimiques, d'intervenir dans les phénomènes d'épissage de gènes (*introns* de groupe I et II), de réguler l'expression de gènes, de corriger des problèmes de traductions (ARNtm) et bien d'autres rôles leurs sont attribués. Un article de Lolle et al. [29] datant de 2005 pose même la question de la transmission possible

à une descendance d'une partie de l'information génétique par l'ARN! Ainsi, même si dans de nombreuses tâches, l'ARN a été supplanté par les protéines ou l'ADN, il reste un acteur clé dans les organismes contemporains.

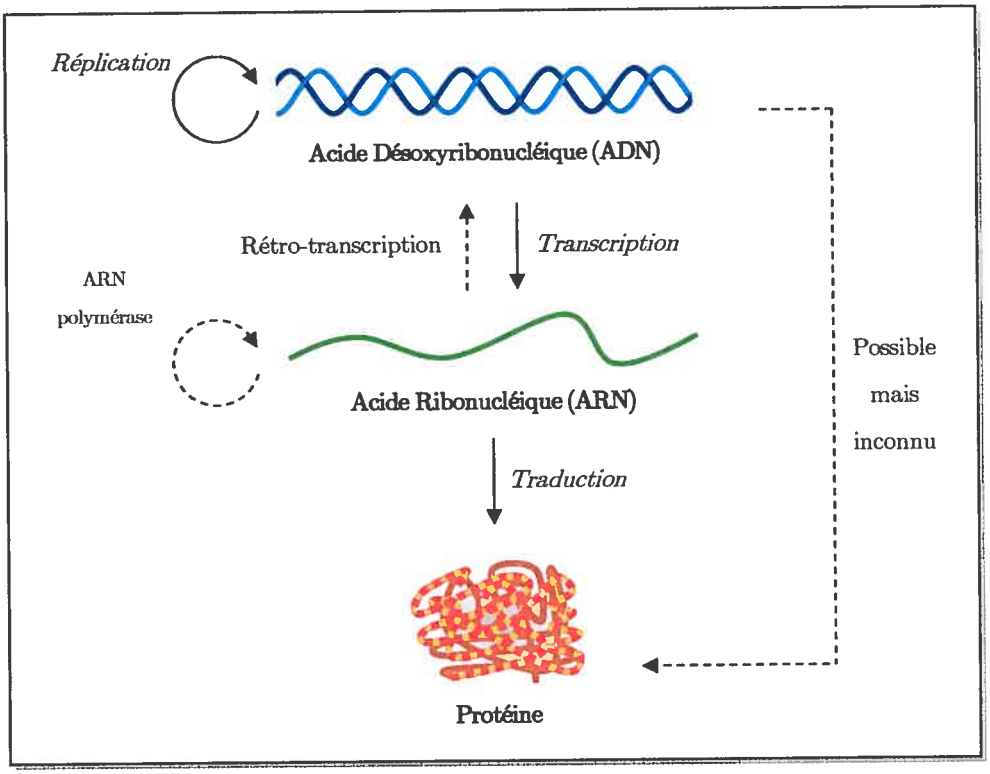


Figure 4 Dogme central de la biologie moléculaire. (inspirée de [39])

2.3 Différents types d'ARN

Comme nous l'avons évoqué précédemment, il existe une grande variété d'ARN effectuant des opérations toutes aussi variées. Cependant, nombre de ces fonctions sont communes à plusieurs organismes et on peut ainsi regrouper les ARN en familles. La plus grande famille d'ARN et également l'une des plus connue est celle des ARN codants composée des ARN messagers (ARNm). Ces ARN contiennent l'information sur la séquence d'une protéine et servent de principal intermédiaire entre les gènes de l'ADN et leur expression sous forme de protéines. On retrouve les ARNm chez les eucaryotes aussi bien dans le noyau

que dans le cytoplasme. Cette famille doit sa grande taille au fait que toutes les protéines exprimées d'un génome sont issues d'un ARNm leur correspondant.

Toutes les autres familles d'ARN sont composées d'ARN dits "non-codants" car ils ne "codent" pas pour des protéines. La famille d'ARN non-codants la plus connue et représentant autour de 80% de tous les ARN produits par une cellule est sans conteste celle des ARN ribosomaux (ou ARNr). Ils sont eux aussi impliqués dans le phénomène de traduction et servent en quelque sorte de "tête de lecture" de la machinerie de traduction. Ils vont s'associer avec des protéines pour former des ribosomes et parcourir les ARNm d'une extrémité à l'autre. Ils vont faire appel à une autre famille d'ARN bien connue, celle des ARN de transfert (ou ARNt) pour traduire le code génétique en protéine. Les ARNt sont des ARN dont certaines bases ont été modifiées afin de leur permettre de reconnaître des triplets de nucléotides sur l'ARNm grâce à une de leurs extrémités et de leur faire correspondre un acide aminé particulier greffé sur une autre de leurs extrémités.

Mais ces 3 types d'ARN ne sont pas les seuls à avoir une fonction essentielle pour les cellules. Avant de pouvoir être fonctionnels, les ARNt ont besoin d'être maturées par plusieurs enzymes dont des endonucléases comme par exemple la RNase P ([8], Figure 10) qui n'est rien d'autre qu'une molécule d'ARN capable de catalyser (ribozyme) le clivage d'un précurseur d'ARNt.

Une autre famille d'ARN est également impliquée dans le processus de traduction, lorsqu'un ARN messager problématique bloque un ribosome; il s'agit des ARN transfert-messagers. Comme son nom l'indique, l'ARN transfert-messager (ARNtm, précédemment connu sous le nom d'ARN 10S, [35]) joue un double rôle. Lorsqu'un ARN messager se retrouve bloqué dans un ribosome, l'ARNtm peut alors s'associer à ce ribosome là où normalement viendrait s'insérer un ARN de transfert normal. En effet, l'ARNtm possède un domaine structuralement proche de celui d'un ARN de transfert normal. Son autre domaine quant à lui, se comporte comme un ARN messager et est localisé là où

se trouverait normalement le triplet de nucléotides reconnu par un ARN de transfert. Il va donc prendre la place de l'ARN messager et permettre à la traduction de s'achever, libérant ainsi le ribosome et la protéine défectueuse qu'il était en train de synthétiser (Figure 5).

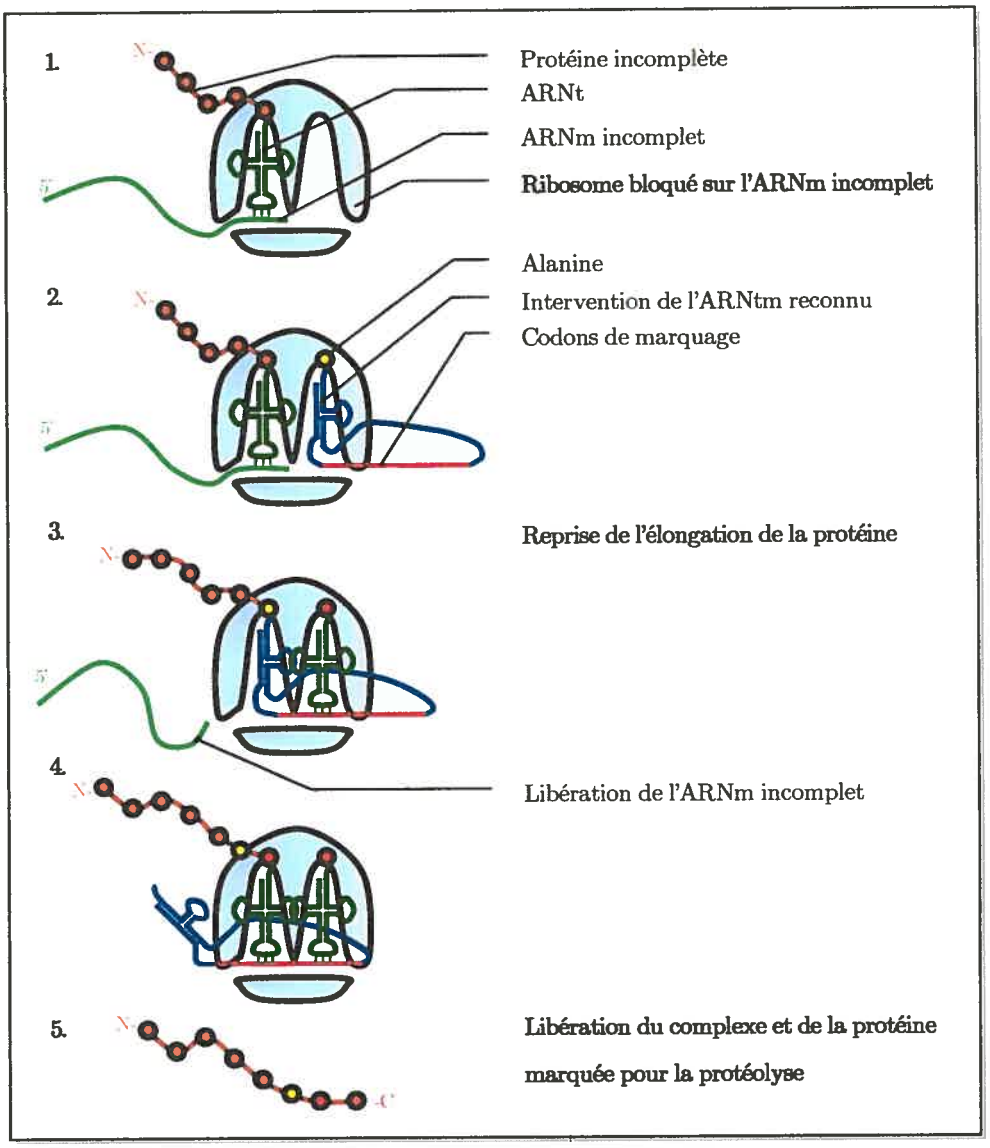


Figure 5 Fonctionnement des ARNtm (inspiré de http://www.indiana.edu).

Plus en amont de la traduction, les précurseurs d'ARN messagers peuvent présenter des *introns*, c'est-à-dire des sous-séquences qui seront absentes de la

séquence de l'ARN messager mature. Ces *introns*, séparés en groupes I et II, sont en fait des ARN capable de catalyser une réaction leur permettant de s'extraire d'eux-mêmes de la séquence du futur ARN messager (Figure 6).

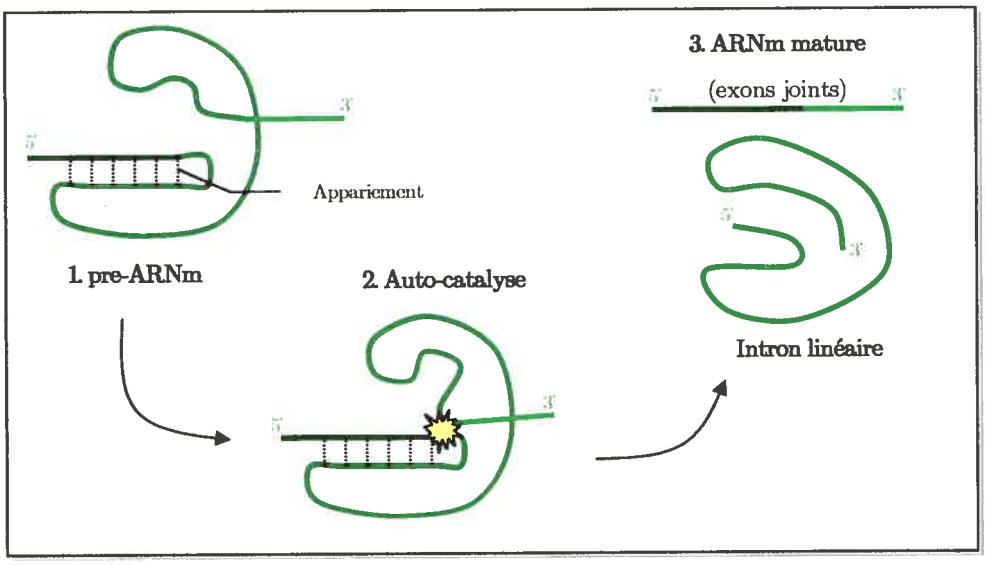


Figure 6 Réaction auto-catalytique d'un intron de Groupe I. (inspirée de [39])

Enfin une famille d'ARN qui prend de plus en plus d'importance est celle des micro-ARN (miARN, [6]). Il s'agit de courtes séquences d'ARN (généralement moins d'une centaine de bases) formant une structure très simple appelée "précurseur" dont une sous-séquence est excisée pour former un miARN mature. Ce dernier est capable de s'apparier avec un ARN messager et d'en rendre impossible sa traduction (Figure 7). Il s'agit d'un phénomène d'interférence par ARN qui bloque la production d'une protéine en particulier.

Bien d'autres familles d'ARN ont été découvertes, certaines sont impliquées dans la reconnaissance de signaux, d'autres appartiennent à des virus, il en reste quelques-unes dont les fonctions sont inconnues et beaucoup restent encore probablement à découvrir. Dans sa version 7.0 datant de mars 2005, la base de données publique RFam ([14]) recensait 503 familles d'ARN et ce nombre ne cesse de croître depuis plus de 3 ans. Cela justifie l'intérêt grandissant des chercheurs pour les ARN.

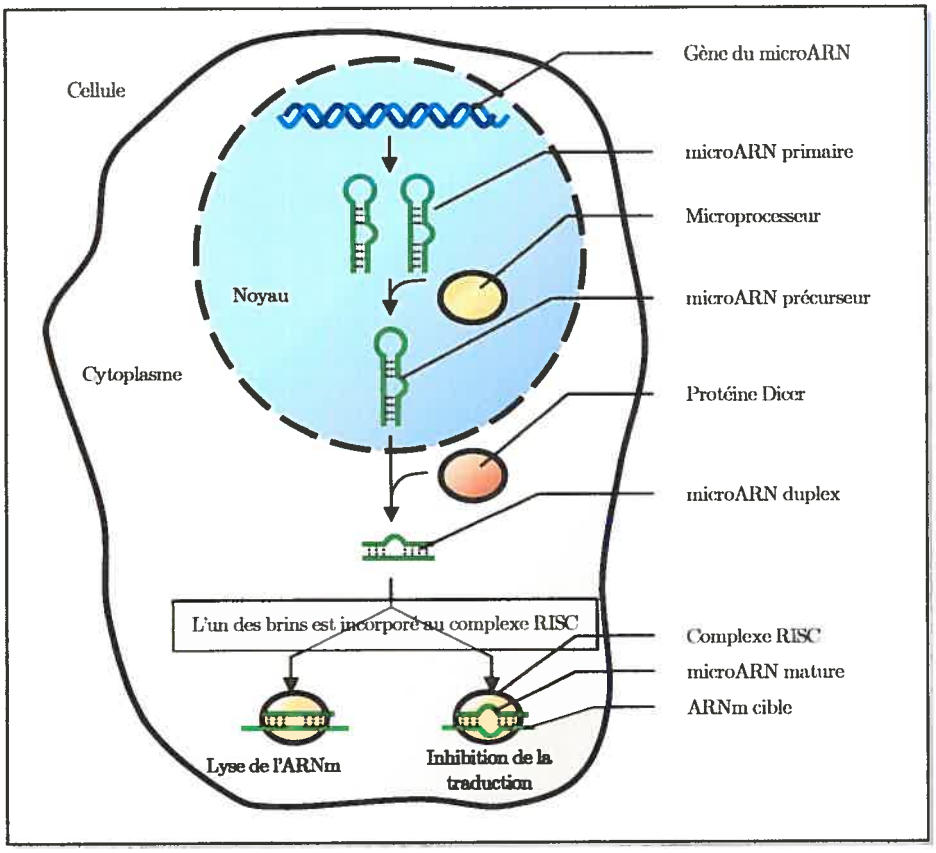


Figure 7 Principe de fonctionnement de la régulation d'un gène par un microARN. (inspiré de [3])

2.4 Les perspectives de recherche sur l'ARN

Il y a beaucoup d'enjeux dans la recherche axée sur les ARN non-codants. Tout d'abord, mieux appréhender comment fonctionnent ces molécules permettrait de comprendre certains dysfonctionnements cellulaires liés à des ARN défectueux et cela ouvrirait éventuellement la voie vers des thérapies. De même, de nombreux virus agissent principalement grâce à des ARN et connaître leur mode opératoire permettrait de trouver des moyens de les bloquer et donc d'offrir un traitement thérapeutique.

La thérapie génique fonde beaucoup d'espoir sur les ARN. Il s'agit de les utiliser dans le but de corriger l'action de gènes défectueux. Plusieurs voies de

recherche sont ouvertes. L'une d'elles serait d'utiliser des *introns* de groupe II modifiés capables de s'insérer à des sites spécifiques de l'ADN (le génome) pour y apporter des corrections ([17]). Une autre méthode serait de concevoir et d'utiliser des petits ARN d'interférence en utilisant le même principe que celui des micro-ARN pour inhiber l'expression de gènes défectueux.

Dans tous les cas, la compréhension et la maîtrise des mécanismes employés par les différents ARN passent par l'étude de leurs structures qui leurs confèrent des propriétés biochimiques.

2.5 Représentations de l'ARN non-codant

Contrairement aux ARN messagers, dont la séquence nucléotidique est primordiale, les ARN non-codants effectuent leurs tâches parce qu'ils peuvent adopter une structure spatiale particulière. De ce fait, les structures des ARN non-codants seront les seules à présenter un réel intérêt pour nous et nous allons à partir de maintenant ne plus tenir compte des ARN messagers lorsque nous parlerons d'ARN en général.

Dans la plupart des cas, un ARN ne va pas travailler seul et il va au contraire s'associer à d'autres molécules pour être actif. Il peut s'agir d'autres ARN ou de protéines ou des deux. Ces associations forment un complexe que l'on qualifie de structure quaternaire (Figure 8).

Un ARN seul, dans des conditions environnementales normales, se présente sous la forme d'une chaîne d'acides ribonucléiques repliée sur elle-même dans l'espace pour former une espèce de pelote. Le maintient de la forme adoptée est assurée par diverses interactions entre les atomes de la molécule d'ARN. Cette forme est appelée structure tertiaire (Figure 9).

En cassant virtuellement certaines des interactions intramoléculaires de l'ARN, il est possible de représenter de façon schématique la molécule à plat. Dans cette représentation, chaque base de la séquence est remplacée par un code

d'une lettre correspondant à la nature de l'acide ribonucléique et les interactions

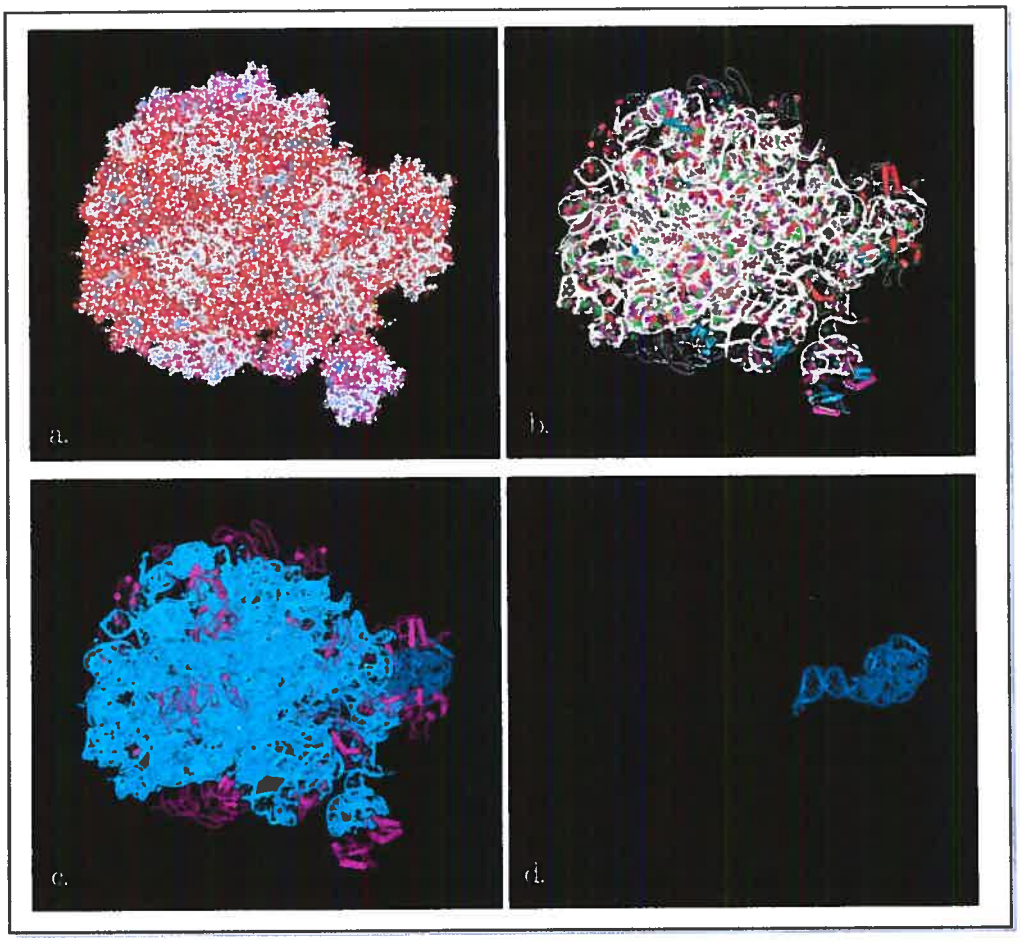


Figure 8 Différentes représentations du complexe ribo-protéique de la sous-unité 50S du ribosome 70S de E. Coli. (source: RCSB-PDB, acc=2AWB, [37])

a. représentation CPK (Corey-Pauling-Koltun);

b. représentation schématique des protéines et ARN (tiges et rubans);

c. coloration: protéines en rouge, ARNr 23S en vert et ARNr 5S en bleu.;

d. ARNr 5S isolé du complexe.

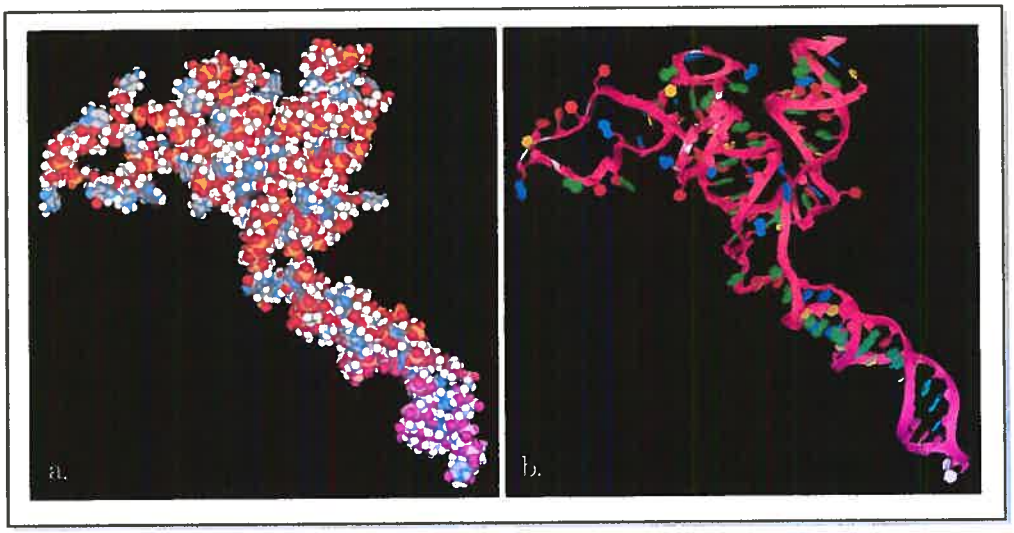


Figure 9 Structure tertiaire de l'ARNr 5S de E. Coli (source: RCSB-PDB, acc=1C2X, [5]).

a. représentation CPK;

b. représentation schématique (rubans et anneaux avec coloration par types de base).

sont symbolisées par des tirets, des points ou d'autres figures particulières. On parle alors de structure secondaire (Figure 10). Les structures secondaires d'ARN peuvent présenter plusieurs degrés de détails. Certaines structures, très bien étudiées et documentées, comportent toutes les interactions entre les bases, tandis que d'autres au contraire, seront épurées et ne présenteront que 2 types d'interaction, celles des paires de type Watson-Crick et celles des paires de type Wobble. En règle générale, lorsque l'on parle de structure secondaire, on ignore toutes les interactions faisant intervenir plus de deux bases. De plus, il existe des cas particulier d'éléments de structures secondaires: les *pseudonœuds* (en anglais: pseudoknots) et les boucles jointes. Le *pseudonœud* est composé de deux tiges-boucles dont certaines bases appariées de l'une forment une partie des bases de la boucle terminale de l'autre (Figure 16b.). Les boucles jointes, quant à elles, comportent également deux tiges-boucles mais ces dernières ne sont reliées entre elles que par *appariements* entre les bases de leurs boucles terminales (Figure

16b.). Bien que ces éléments soient composés de paires de bases, ils sont à la frontière entre structure secondaire et tertiaire. C'est pourquoi, on considère normalement qu'ils ne font pas partie des structures secondaires même si malgré tout, ils sont souvent représentés sur ces dernières.

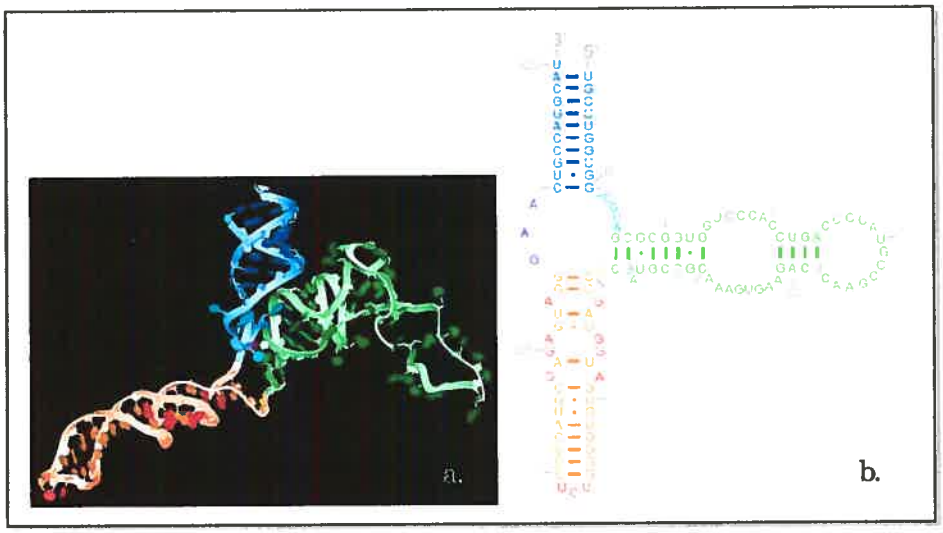


Figure 10 Structure tertiaire et secondaire correspondante de l'ARNr 5S de E. Coli. (source: RCSB-PDB, acc=1C2X, [5]).
a. structure tertiaire (représentation schématique, chaque branche a été colorée pour mieux identifier les correspondances avec la structure secondaire);
b. structure secondaire faisant apparaître des appariements non canoniques ou de Wobble.

Enfin, lorsque les interactions entre nucléotides sont ignorées, il ne reste plus qu'une séquence de nucléotides qui forment la structure primaire. L'ARN est alors vu comme une suite de lettres dont chacune code pour un type d'acide ribonucléique (Figure 11).

UGCCUGGCGGCCGUAGCGCGGUGGUCCCACCUGACCCCAUGCCGAACUCAGAAGUGAAACGCCGUAGCGC
CGAUGGUAGUGUGGGGUCUCCCCAUGCGAGAGUAGGGAACUGCCAGGCAU

Figure 11 Séquence de l'ARNr 5S de E. Coli. (V00336) (source: NCBI, acc=D12500). Les couleurs correspondent à celles employées Figure 10.

2.6 Eléments des structures secondaires

Comme nous l'avons vu dans la section 2.5, les ARN non-codants adoptent une forme tridimensionnelle qui peut être modélisée de façon simplifiée dans un plan. Lors de cette simplification, des informations structurelles sont perdues. Cependant, il existe différents niveaux de simplifications permettant de réduire l'information structurelle suivant les besoins.

Tout d'abord, lors du passage de la structure moléculaire tridimensionnelle à la structure secondaire, les bases azotées sont remplacées par des lettres symboliques: un **A** pour l'adénine, un **U** pour l'uracile, un **G** pour la guanine et un **C** pour la cytosine. Cependant, dans certains cas, l'identification d'un acide ribonucléique à une position donnée n'est pas triviale et il arrive également que des mutations sur un nucléotide apparaissent au sein d'une même espèce sans qu'un consensus puisse être déterminé. Parfois, des modifications post-transcriptionnelles empêchent également d'identifier la nature d'un acide

Symbole	Signification	Origine de la désignation (anglais)
G	G	Guanine
A	A	Adenine
T	T	Thymine
C	C	Cytosine
U	U	Uracile
R	G ou A	puRine
Y	T ou C	pYrimidine
M	A ou C	aMino
K	G ou T	Keto
S	G ou C	Strong interaction (interaction forte, 3 ponts H)
W	A ou T	Weak interaction (interaction faible, 2 ponts H)
H	A ou C ou T	non-G (H suit G dans l'alphabet)
B	G ou T ou C	non-A (B suit A)
V	G ou C ou A	non-T et non-U (V suit U)
D	G ou A ou T	non-C (D suit C)
N	G ou A ou T ou C	aNy (n'importe lequel)

Tableau I Code IUPAC-IUB pour les acides nucléiques (source: [20]).

nucléique. En fait, divers cas de figures peuvent se poser et empêcher l'utilisation d'un symbole déterminant un acide nucléique précis à une position donnée. Pour palier à ce problème, d'autres symboles ont été introduits et appartiennent à un standard dénommé code IUPAC-IUB. Ce standard ne prend pas en compte les bases modifiées mais il permet tout de même l'utilisation de lettres représentant plusieurs types de bases possibles (Tableau I).

Le passage de la structure moléculaire tridimensionnelle à la structure secondaire enlève également des informations sur les positions relatives des bases azotés entre elles. En effet, une base forme souvent des ponts hydrogènes avec une ou plusieurs autres bases pour se stabiliser et les liens non-covalents ainsi formés peuvent adopter un grand nombre de conformations. Les liens les plus courants sont les *appariements* entre deux bases complémentaires. Il s'agit ici des appariements de type Watson-Crick (ou respectivement Wobble) qui sont symbolisés par un trait court (ou respectivement un point) entre deux bases. Cependant, il existe de nombreux autres types d'*appariements* et certaines structures secondaires bien annotées peuvent les faire apparaître. Pour les symboliser et pouvoir les différencier, une nomenclature standardisée a été établie (Figure 12).

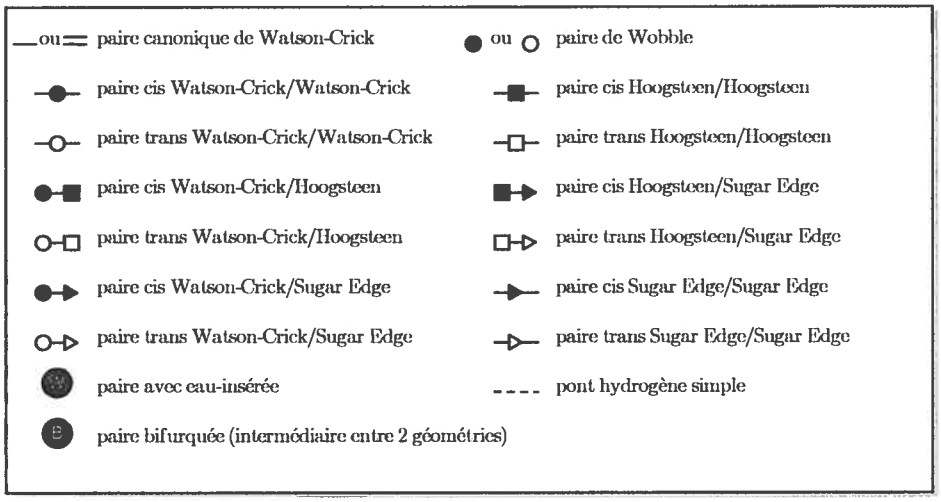


Figure 12 Annotation des appariements de bases des structures secondaires d'ARN (source: [27]).

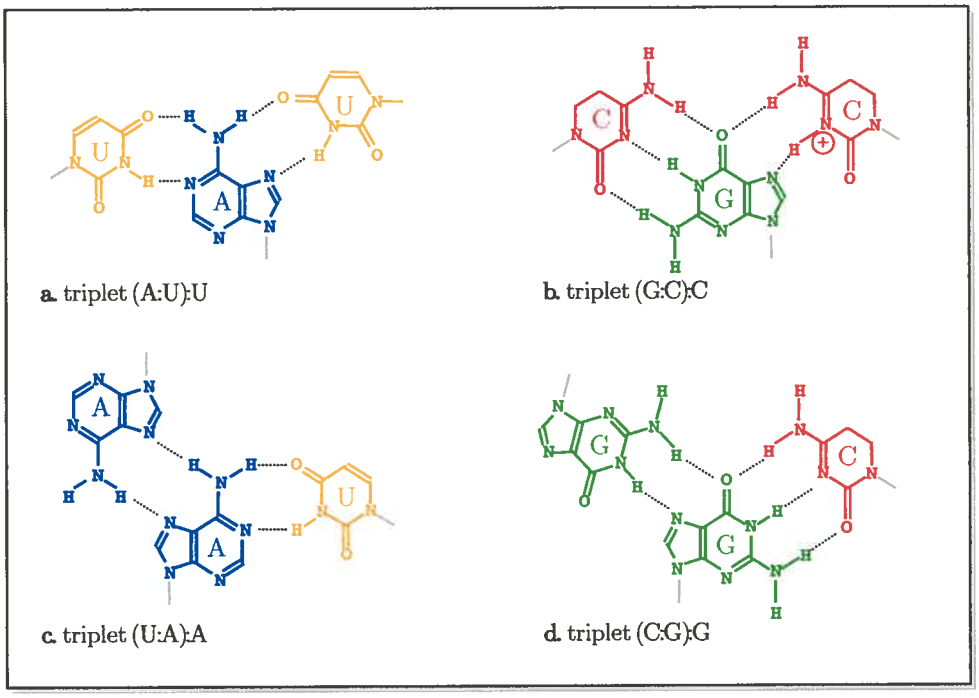


Figure 13 Exemples d'interactions observées entre triplets de bases.

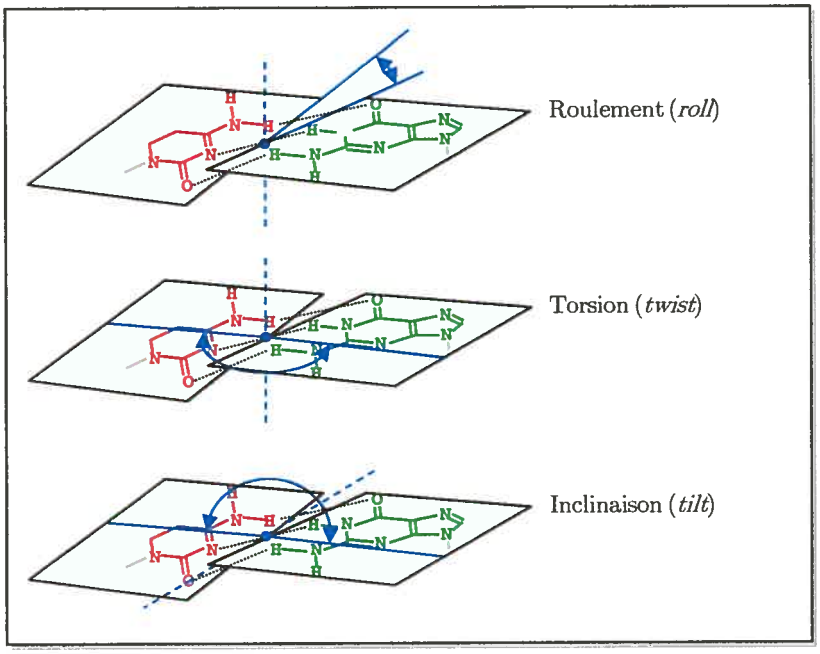


Figure 14 Angles de mesures des variations spatiales d'un appariement canonique.

Les *appariements de bases* font intervenir deux bases mais il arrive que trois bases ou plus interagissent entre elles (Figure 13). Ces interactions sont parfois symbolisées sur les structures secondaires par de longs traits reliant les bases concernées, néanmoins ces interactions sont considérées comme tertiaires et ne devraient pas être considérées comme faisant parti de la structure secondaire.

Pour compliquer encore les choses, il existe de nombreuses variations dans la façon dont deux bases sont appariées. Ces variations peuvent être caractérisées par des angles (Figure 14) mais cette information est normalement absente dans la modélisation en structure secondaire.

Bien entendu, certaines bases ne présentent pas d'interactions notables avec d'autres bases d'une même structure. Ces bases sont appelées bases libres. Cependant, il est bon de noter qu'une base peut apparaître comme libre sur une structure secondaire mais avoir des interactions avec d'autres bases en réalité.

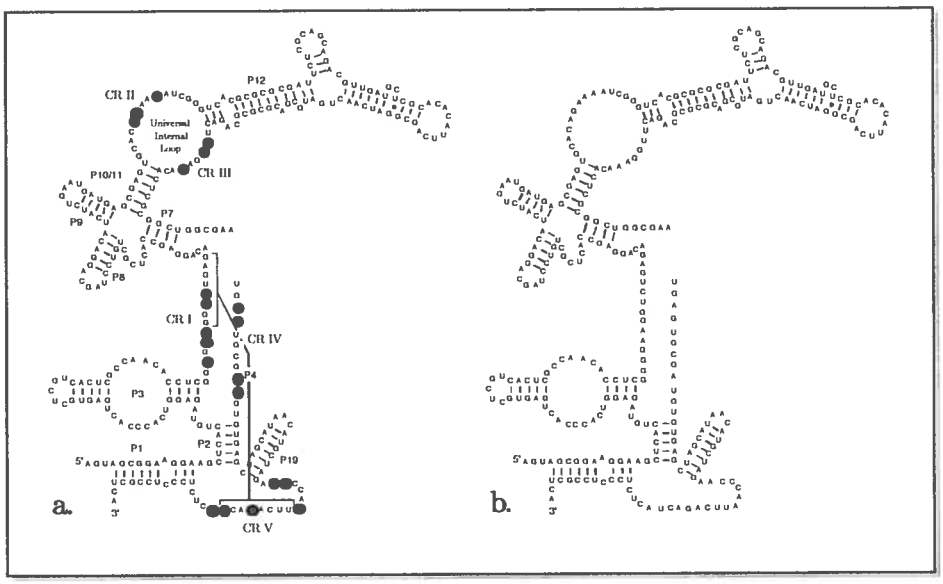


Figure 15 Deux versions d'une même structure secondaire de RNase P de Danio Rerio.

a. Version hautement annotée;

b. Version simplifiée.

Ces interactions ont simplement été retirées de la structure secondaire pour en simplifier la lecture ou encore parce qu'elles sont mineures ou incertaines.

Généralement, une structure secondaire ne fait apparaître que les interactions de type Watson-Crick et Wobble et des bases libres (Figure 15).

Ainsi simplifiées, les structures secondaires présentent moins d'informations mais sont en contrepartie bien plus faciles à manipuler par un programme informatique et également bien plus lisibles par un humain. Une structure peut être décomposée en plusieurs éléments et sous-éléments structuraux (Figure 16). Les bases libres situées aux extrémités de la séquence de la structure sont appelées bases externes (en anglais: external bases). Il en est de même pour les bases libres qui n'appartiennent à aucun autre élément structurel. Les bases appariées sont appelées paires empilées (en anglais: stacked pairs) en référence à la façon dont elles se superposent dans la réalité pour former une hélice (en anglais: helix). Les renflements (en anglais: bulges) correspondent à une séquence d'une ou plusieurs bases libres consécutives délimitées par deux paires de bases reliées entre elles. Les boucles internes (en anglais: internal loops) correspondent quant à elles à deux séquences non consécutives d'une ou plusieurs bases délimitées par les deux mêmes paires de bases; chacune de ces paires est liée en 5' et en 3' à l'une ou l'autre de ces deux séquences. Une sous-structure composée exclusivement d'une combinaison de ces éléments reliés les uns aux autres constitue ce que l'on appelle une tige (en anglais: stem). Les bases libres consécutives reliant les deux bases d'une paire située à l'extrémité d'une tige forment une boucle terminale (en anglais: hairpin loop). La sous-structure comportant une tige et une boucle terminale est appelée tige-boucle (en anglais: stem-loop). Les bases libres formant la jonction entre plusieurs tiges ou tiges-boucles composent les boucles multiples (en anglais: multibranched loops). Enfin, lorsqu'une des bases d'une paire appartient à la sous-séquence située entre deux autres bases appariées mais que l'autre base de la paire ne s'y trouve pas, nous sommes en présence d'un *pseudonœud* ou de boucles jointes.

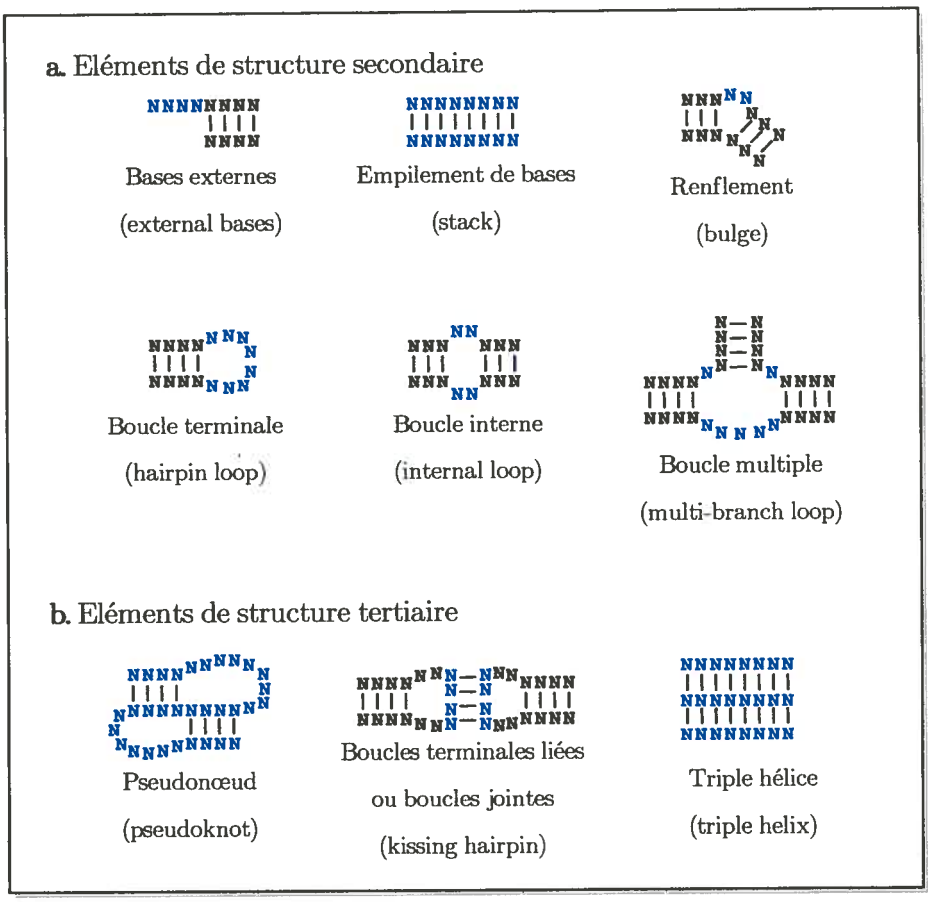


Figure 16 Eléments structuraux retrouvés sur les structures secondaires d'ARN. En bleu, les bases impliqués et entre parenthèse la dénomination anglophone équivalente.

Généralement, les *pseudonœuds* (et les boucles jointes couramment assimilées à des sortes de *pseudonœuds*) ne sont pas considérés comme faisant partie de la représentation en structure secondaire même s'ils jouent souvent un rôle primordial dans le bon fonctionnement d'un ARN non-codant. Par exemple, le *pseudonœud* de la composante ARN de la télomérase humaine joue un rôle critique dans son activité ([10]). Mais la prise en compte des *pseudonœuds*, que cela soit dans le domaine de la prédiction de structures ou de la comparaison, complique énormément les problèmes. Ainsi, dans le cadre de la prédiction de structures secondaires incluant des *pseudonœuds*, le problème est NP-complet

pour une large classe de modèles raisonnables de *pseudonœuds* ([30]) et dans le cadre de la comparaison de structures secondaires, le problème devient également NP-complet lorsque les *pseudonœuds* sont pris en compte ([43]). Par conséquent, pour pouvoir offrir des algorithmes efficaces, les chercheurs sont contraints à reléguer les *pseudonœuds* aux travaux portant sur les structures tertiaires.

Tous ces éléments peuvent être utilisés pour décrire ou caractériser une structure secondaire. Ainsi, la forme adoptée par les structures secondaires d'ARN de transfert est assez caractéristique car elle comporte normalement quatre tiges-boucles reliées à une même boucle multiple, ce qui lui a value l'appellation de structure en forme de trèfle (en anglais: cloverleaf).

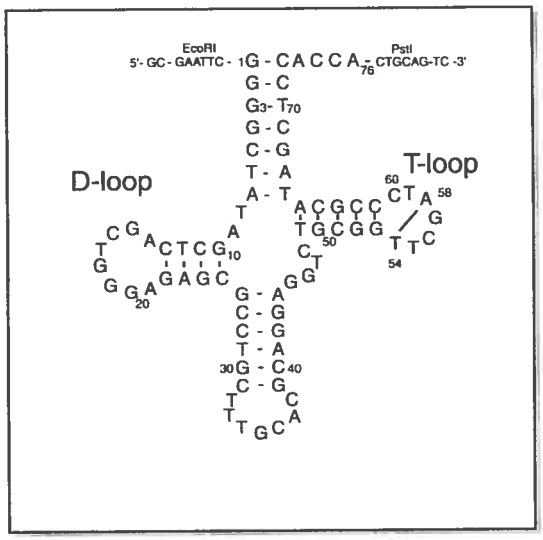


Figure 17 Structure en forme de trèfle typique des ARN de transfert.
Ici l'ARNt$^{Ala}_{UGC}$ de E. Coli (source: [41] avec l'aimable autorisation de reproduction de F. Doyon).

Chapitre 3

LA COMPARAISON

3.1 Les fondements de la comparaison de structures secondaires

Pour étudier les ARN non-codants, on s'intéresse en général à leur structure, qui est primordiale pour leur fonction. En effet, la structure adoptée par un ARN provient principalement des *appariements* entre ses bases et non de la nature de ces bases. Par exemple, il est souvent possible de remplacer une paire G-C par une paire A-U sans que la structure n'en souffre trop: la séquence est alors modifiée en deux points séparés mais pas la structure! Plusieurs mutations de ce genre peuvent amener des morceaux de deux séquences distinctes d'ARN, dont les structures sont similaires, à se ressembler énormément même s'ils occupent des positions complètement différents dans ces structures. De ce fait, la comparaison d'ARN non-codants est en général centrée sur leur structure plutôt que sur leur séquence seule.

Il serait plus judicieux d'étudier leurs structures quaternaires ou tertiaires. Malheureusement, il est à l'heure actuelle techniquement très difficile d'obtenir des images de tels complexes ou structures au complet. De plus, étudier des modèles en 3 dimensions n'est pas chose facile et les comparer entre eux est une tâche encore plus délicate, même avec l'aide de la bio-informatique. En revanche, les chercheurs sont capables d'identifier ou de prédire les *appariements* de certaines bases de l'ARN. Par exemple, la séquence d'un ARN est relativement facile à obtenir et il est possible d'utiliser des algorithmes bio-informatiques pour calculer les *appariements de bases* qui minimisent l'énergie libre de la structure (Figure 18) et donc la rendraient plus stable et plus probable dans la nature ([45]). Ces *appariements*, qu'ils aient été déterminés expérimentalement

ou prédits, ne fournissent pas d'informations tridimensionnelles mais forment un premier pas vers elles. C'est pour toutes ces raisons que le concept de structure secondaire est apparu.

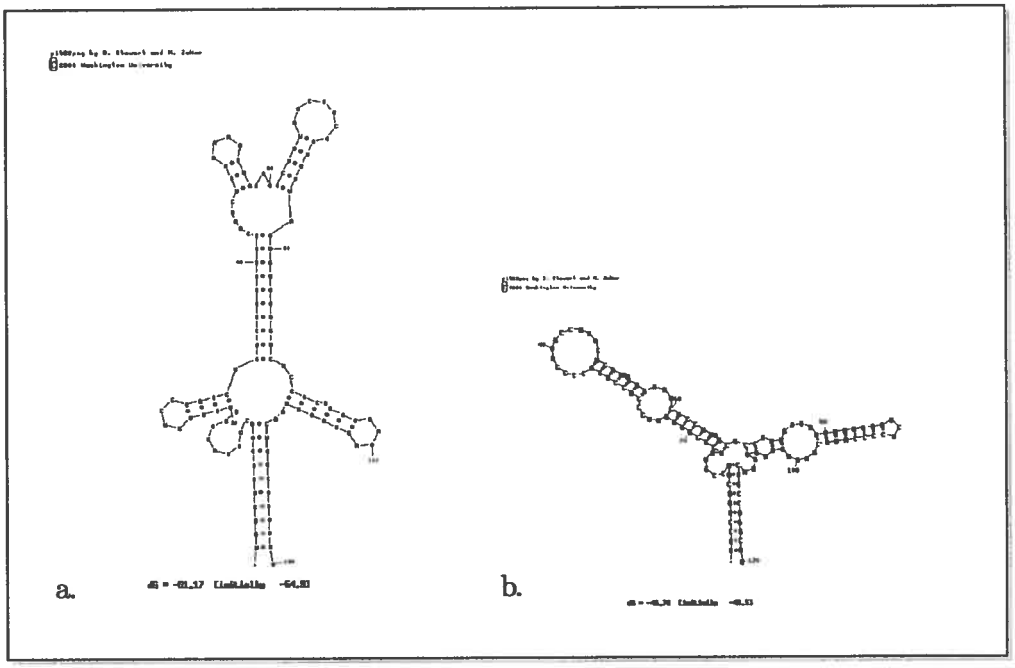


Figure 18 Prédictions du repliement de l'ARNr 5S de E. Coli à partir de sa séquence par MFold. (version 3.2, [44] et [45])
a. à gauche la meilleure prédiction avec les paramètres par défaut;
b. à droite la structure secondaire réelle prédite en introduisant une seule contrainte forçant la formation de la paire 70-106 (commande MFold: "F 70 106 1").

Les structures secondaires sont une représentation simplifiée sous forme de graphe en 2 dimensions des molécules d'ARN. Cette modélisation offre de nombreux avantages. Elle peut être représentée sur une feuille manuellement, la séquence de nucléotides est lisible et les principales interactions qui caractérisent une structure y figurent. De surcroît, elles sont plus faciles à modéliser informatiquement et elles sont également plus simples et plus rapides à manipuler avec des algorithmes que des structures en 3 dimensions.

Ce dernier point est d'autant plus important que la recherche génère de plus en plus de résultats sur les structures d'ARN. Il est envisageable que dans quelques années, obtenir des images tridimensionnelles de structures sera aussi simple que de séquencer des gènes aujourd'hui. Même si on est capable de fournir des outils informatiques pour travailler sur des modèles en trois dimensions, ils devront gérer un niveau de complexité supérieur à celui des structures secondaires. Étant donné que les structures secondaires renferment déjà les informations principales sur la structure d'un ARN, est-il pertinent de vouloir absolument travailler sur des modèles 3D qui prendront bien plus de temps à être analysés mais dont les résultats seront plus précis?

La réponse à cette question n'est pas triviale et mérite réflexion. Néanmoins, l'expérience montre qu'avoir des outils rapides pour traiter de grands ensembles de données est une nécessité, et ce même s'ils ne sont pas parfaits. Aujourd'hui, on ne discute plus l'intérêt de BLAST pour comparer des séquences en grand nombre même s'il s'agit d'une heuristique, c'est-à-dire que ses comparaisons ne sont pas parfaites et peuvent produire des erreurs. De manière analogue, face à une augmentation des données sur les structures d'ARN, comparer leurs structures secondaires ne fournira pas une information parfaite mais malgré tout très utile aux chercheurs et sera bien plus rapide qu'une comparaison d'un niveau supérieur impliquant une complexité également supérieure.

Si en termes de rapidité, la comparaison de structures secondaires est un bon choix, nous sommes dans le droit de nous poser la question si ce compromis sur les détails structuraux est un bon choix. Quelles informations pouvons-nous tirer en comparant des structures secondaires? Les réponses sont nombreuses!

Tout d'abord, en l'état actuel des connaissances scientifiques dans le domaine de la biochimie, il est pratiquement impossible de prédire la fonction d'un ARN à partir simplement de sa structure. Cela requiert nécessairement des expériences pour démontrer une fonction. Cependant, par une comparaison avec un ensemble de structures connues, il est possible d'avoir une petite idée de la

fonction assurée par un ARN, surtout si des membres de sa famille sont déjà connus et font parti de l'ensemble testé. À défaut de posséder, dans une banque de données, les membres de la famille d'un nouvel ARN d'intérêt, le comparer à un grand ensemble de structures connues peut faire ressortir des similarités locales avec d'autres structures. Ces similarités pourraient permettre de déceler les zones actives de l'ARN étudié, ou d'extrapoler sa forme tridimensionnelle. La comparaison de structures secondaires peut ainsi aider à prédire les fonctions d'ARN inconnus.

Retrouver des similarités locales entre structures peut s'avérer utile lorsque l'on est en présence d'un ARN dont on connaît la structure et que l'on suppose qu'il peut affecter des cellules en interférant avec un autre ARN fonctionnel de cette cellule. Il peut s'agir par exemple d'un ARN viral ou encore d'un futur médicament. Là encore, la comparaison rapide de la structure secondaire de l'ARN d'intérêt avec une base de donnée permettrait de faire ressortir les ARN présentant des zones similaires et d'identifier là où pourrait agir l'ARN étudié.

Si nous retournons maintenant dans l'optique d'une possible automatisation de l'obtention de structures d'ARN, il sera nécessaire de pouvoir classer et annoter automatiquement ces structures. Là encore, la comparaison de structures secondaires peut apporter des réponses. En effectuant un alignement structurel entre une nouvelle structure secondaire fraîchement obtenue et les structures déjà annotées de la base de données les plus similaires, il devient possible d'identifier de manière automatique la famille de la nouvelle structure ainsi que ses zones clés et de les annoter.

Un autre domaine d'intérêt est celui de la prédiction des structures tertiaires d'ARN à partir de leur séquence. À l'heure actuelle, il s'agit encore d'un casse-tête pour les chercheurs car aucun outil existant n'est capable de donner de bons résultats dans tous les cas de séquences fournies. Cependant, il est possible de comparer une structure secondaire prédite avec une base de données de structures pour voir si elle est proche d'autres structures de la base, ce qui

indiquerait que le repliement a des chances d'être bon. Cette voie n'a cependant pas encore été totalement explorée mais il s'avère que la détection de zones localement bien repliées pourrait aider le calcul du repliement global en invalidant les zones mal repliées pour qu'elles soient recalculées. Cette validation par la comparaison de structures secondaires pourrait avoir à être répété plusieurs fois sur de grosses structures, nécessitant l'emploi d'algorithmes de comparaisons rapides.

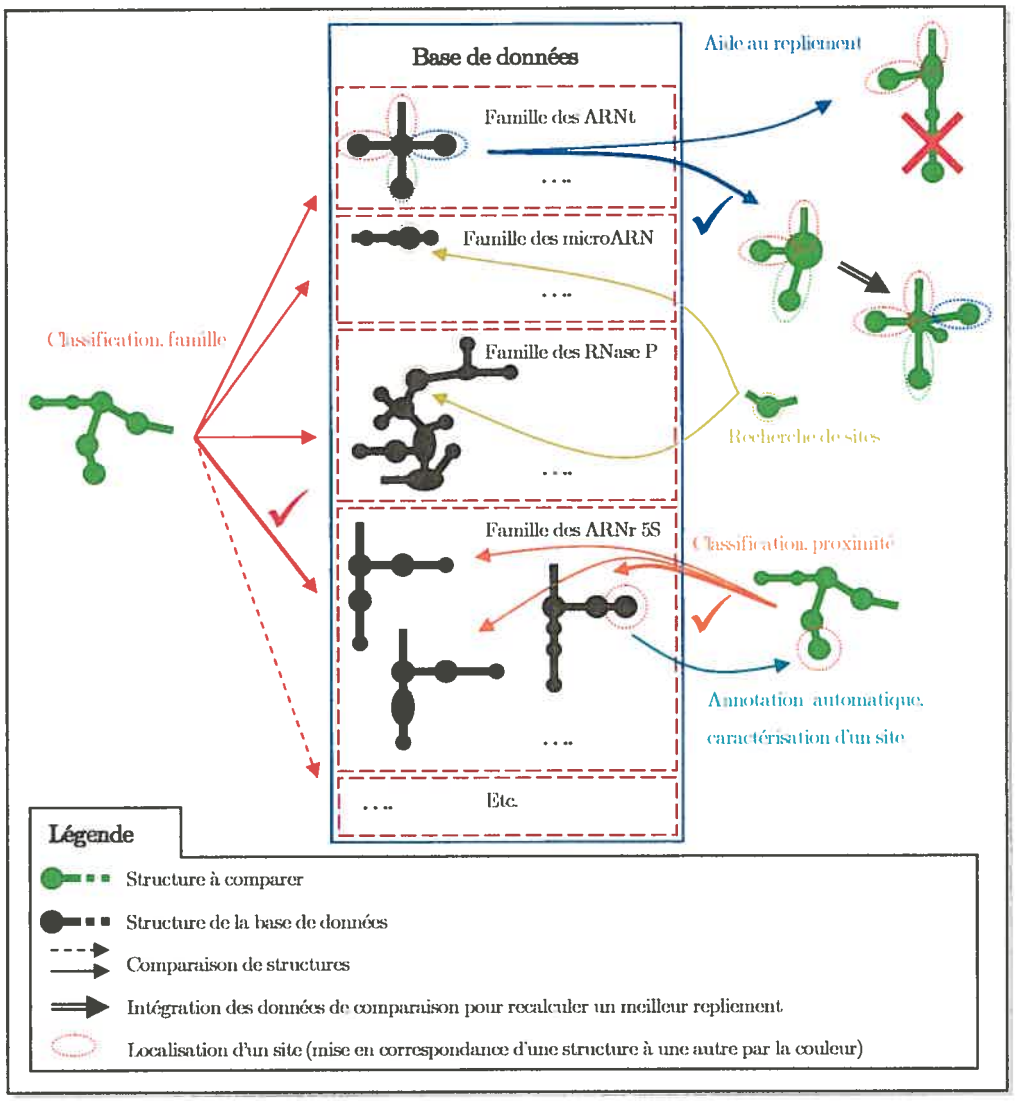


Figure 19 Quelques applications de la comparaison de structures secondaires.

Comme nous l'avons vu, la comparaison de structures secondaires est loin d'être un simple amusement scientifique. Les intérêts derrières ces comparaisons sont bien réels, variés et nombreux (Figure 19). De plus, la nécessité d'un algorithme rapide permettant de travailler sur une grosse base de données de structures secondaires est évidente. Il faut maintenant s'intéresser à la façon par laquelle vont s'effectuer les comparaisons.

3.2 Types de comparaisons

Il existe une grande variété de techniques pour comparer les structures secondaires d'ARN. Le choix d'une technique par rapport à une autre dépend essentiellement du but recherché. Ainsi, suivant les applications, un chercheur peut avoir besoin de calculer une *distance d'édition* entre des structures ou un score de similarité; il peut également vouloir comparer des structures dans leurs intégralités ou identifier des parties communes à plusieurs structures ou encore rechercher un motif particulier dans un ensemble de structures. De plus, son travail peut porter sur une structure particulière, un petit groupe de structures ou encore un vaste ensemble de structures provenant d'une base de données. Il paraît évident que comparer une structure secondaire calculée pour la valider ne se fera pas de la même façon que de rechercher un motif spécifique à une fonction biochimique. Néanmoins, de façon générale, la finalité de la comparaison de structures secondaires est de quantifier des modifications, appelées également mutations, entre une structure et une autre.

L'approche la plus simple est de mesurer le nombre minimal de changements à effectuer pour transformer une structure de référence A en une autre structure B. Il faut tout d'abord définir un jeu d'*opérations d'édition*, c'est-à-dire de modifications élémentaires permettant, en les combinant, de passer de la structure A à la structure B et inversement. Un coût est ensuite associé à chacune de ces opérations, définissant ainsi un schéma de score. Finalement, un algorithme est utilisé pour déterminer le nombre minimal et le type d'opérations

à effectuer pour changer *A* en *B*. Un tel algorithme est normalement basé sur la programmation dynamique et va chercher à minimiser les coûts d'édition. La somme des coûts de chacune des opérations mises en jeu donne une *distance d'édition* entre les structures *A* et *B* (Figure 20). Typiquement, les coûts des opérations sont positifs pour qu'une *distance d'édition* reflète une distance évolutive. De tels scores peuvent par exemple être utilisés pour des études phylogéniques.

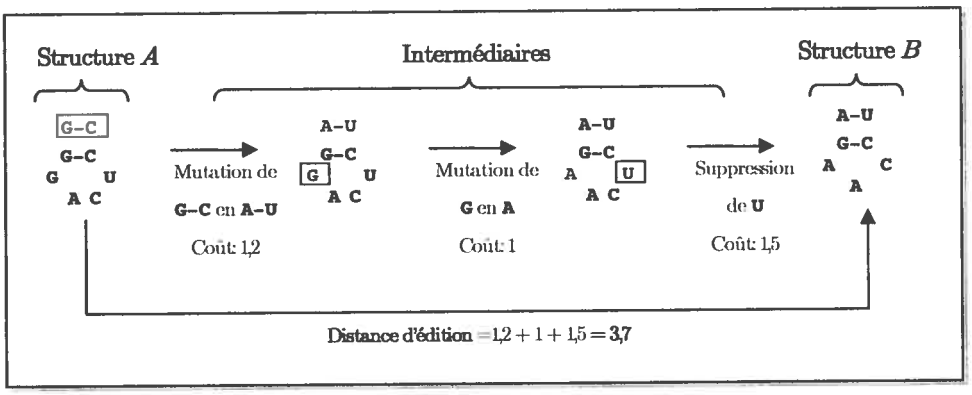


Figure 20 Exemple de scénario de mutations permettant de transformer une structure *A* en *B*.
La distance d'édition calculée est basée sur un schéma de score arbitraire.

Cependant, de tels scores ne reflètent que le nombre de modifications subies par une structure. Ils ne fournissent aucune information sur la quantité d'éléments qui ont été conservés d'une structure à l'autre. Pour obtenir un score qui tienne compte à la fois des modifications structurales et des parties inchangées il faut calculer un score de similarité. Ce calcul se fait en modifiant le schéma de score pour que chaque *opération d'édition* ait un coût négatif et chaque correspondance ait un coût positif. L'algorithme utilisé ne cherchera alors plus un coût minimum mais un score maximum signifiant un maximum de similarités pour un minimum d'éditions. En d'autres termes, plus des structures présenteront un haut degré de similarité, plus leur score de comparaison sera positivement élevé et inversement, plus des structures seront différentes plus

leur score sera bas ou négativement élevé. Un tel score serait par exemple utile lorsqu'une structure A doit être comparée à un ensemble de structures pour pouvoir déterminer avec laquelle A partage le plus de similarité. Un score de similarité permet ainsi de classer des ARN par rapport à leurs structures secondaires.

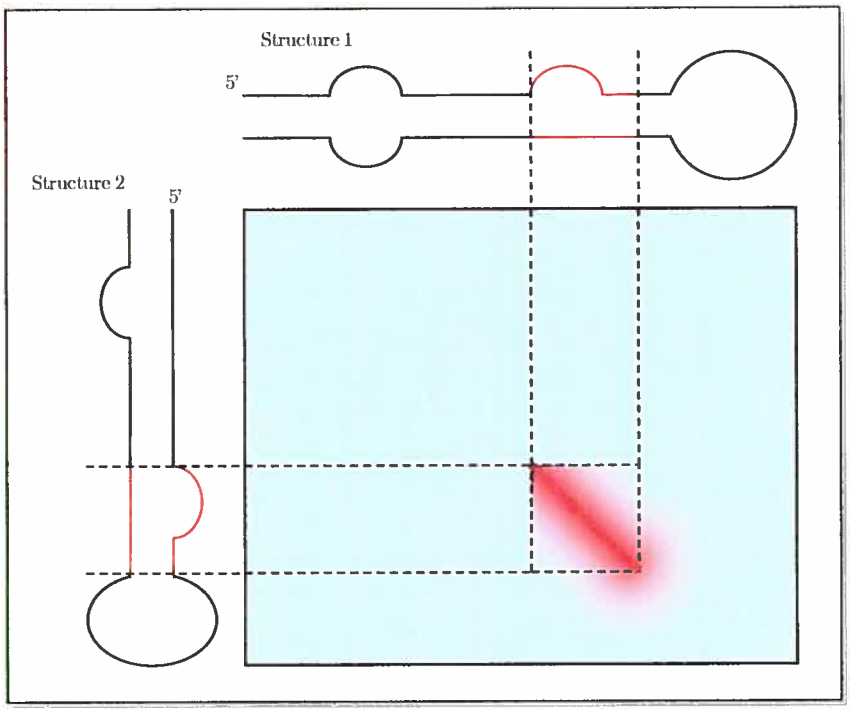


Figure 21 Schéma montrant le principe de l'alignement local entre deux structures secondaires d'ARN.
Sur ces deux structures, seule les parties en rouge sont communes, le reste est différent. Dans la pseudotable de programmation dynamique (cadre coloré), les scores nuls ou proches de 0 sont indiqués en bleu très clair et inversement, plus un score est élevé, plus il apparaît en rouge intense.

Il faut cependant observer que la comparaison de structures dont les tailles varient beaucoup peut fournir des scores reflétant mal les similarités. Il faudrait dans ce cas utiliser une formule pour "normaliser" les résultats. En retraçant un alignement optimal entre deux structures, les nombres d'éléments associés a (correspondances et mutations) et non-associés n (insertions et suppressions)

sont connus; il est donc relativement simple d'en extraire un pourcentage de similarité entre les deux structures: $similarité\,(\%) = \frac{a}{a+n} \times 100$. Ce taux de similarité est un indice plus parlant pour les chercheurs mais il ne faut pas perdre de vue que le score de similarité fournit plus d'information. En effet, il est possible que pour un même taux de similarité entre un couple (*A*, *B*) et un couple (*A*, *C*), le couple (*A*, *B*) présente un score de similarité plus élevé que celui de (*A*, *C*).

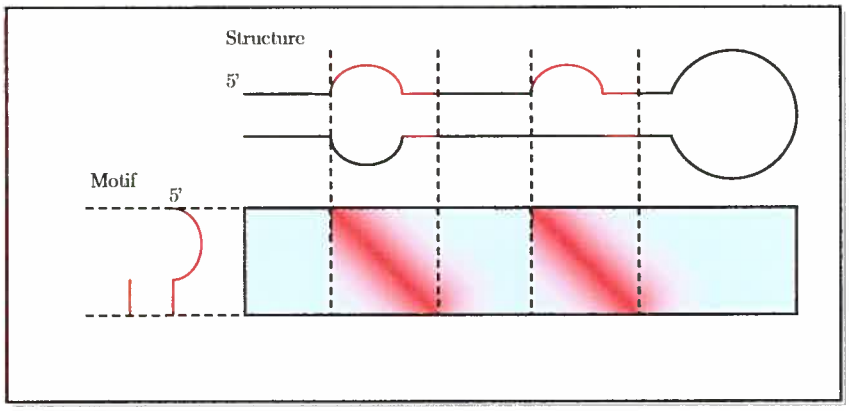


Figure 22 Schéma montrant le principe de l'alignement d'un motif avec une structure secondaire d'ARN.
Dans la pseudotable de programmation dynamique (cadre coloré), les scores nuls ou proches de 0 sont indiqués en bleu très clair et inversement, plus un score est élevé, plus il apparaît en rouge intense. Dans cet exemple conceptuel, le motif a été retrouvé à deux endroits dans la structure.

L'algorithme calculant la similarité entre deux structures peut être modifié pour faire ressortir les alignements locaux et donc les sous-éléments structurels similaires entre deux structures. Pour ce faire, quelque soit le point de départ d'un alignement sur chacune des structure, il ne doit pas commencer par une valeur handicapante. Techniquement, cela revient à empêcher les valeurs négatives d'apparaître dans une table de programmation dynamique en les remplaçant par 0 le cas échéant. Les alignements locaux peuvent être repérés en recherchant dans la table, les valeurs les plus élevées. Pour une même famille

d'ARN comportant une grande variété de forme de structures secondaires, ce type de comparaison est un moyen de faire ressortir les sites conservés, fournissant ainsi des indices sur les parties structurelles clés de cette famille.

L'alignement local peut également être utile lors de la recherche de motifs. Dans ce cas, la méthode de calcul est une sorte d'intermédiaire entre l'alignement global et local. L'algorithme cherchera à aligner intégralement le motif à partir de tout point de la structure étudiée. Ainsi, comme c'était le cas lors de l'alignement global, plus un alignement commencera après le début du motif, plus il sera pénalisé mais en revanche, comme pour les alignements locaux, un alignement pourra commencer en tout point de la structure d'intérêt sans être pénalisé.

But recherché	**Méthode la plus appropriée**
Construire un arbre phylogénique	Distances d'édition ou taux ou score de similarité sur des alignements globaux
Rechercher un ensemble de structures similaires à une structure de référence	Scores de similarité sur des alignements globaux
Rechercher la famille ou la sous-famille d'une structure	Taux et scores de similarité sur des alignements globaux
Rechercher les éléments communs à une même famille d'ARN	Scores de similarités sur des alignements locaux
Aider les calculs de repliement	Scores de similarités sur des alignements locaux
Rechercher un motif particulier sur une ou plusieurs structures	Scores de similarité pour de la recherche de motifs (alignement global du motif mais local sur la structure)
Annoter de façon automatisée	Distances d'édition ou scores de similarité sur des alignements globaux
Comparer dans le détail deux structures pour étudier les scénarios possibles d'évolution entre elles	Distance d'édition sur un alignement global

Tableau II Choix des méthodes d'alignement en fonction des buts recherchés.

Comme nous l'avons vu, il existe diverses façons d'estimer les différences et ressemblances entre structures. Une personne désirant comparer deux structures peut travailler avec des *distances d'édition*, des scores de similarités ou encore des taux de similarité. De plus, elle est libre d'établir son propre schéma de score par lequel les alignements optimaux seront recherchés. Enfin, elle peut choisir différents types d'algorithmes pour des alignements globaux, locaux ou pour la recherche de motifs. Finalement, ses choix dépendront essentiellement du but qu'elle recherche (Tableau II).

3.3 Modélisation des structures secondaires

Comme nous l'avons décrite au départ, la molécule d'ARN est une chaîne d'acides ribonucléiques. En termes mathématiques, on dit qu'il s'agit d'une chaîne de caractères issus d'un alphabet à 16 ou à 4 lettres suivant si le code IUPAC-IUB ([20] et Tableau I) est utilisé ou non. Cette chaîne comporte des arcs reliant les bases appariées.

La première modélisation des structures secondaires découle directement de cette description mathématique de la structure. Il s'agit des séquences arcs-annotées (Figure 23). Comme son nom l'indique, sa représentation fait apparaître une séquence où des arcs relient les deux bases d'une paire.

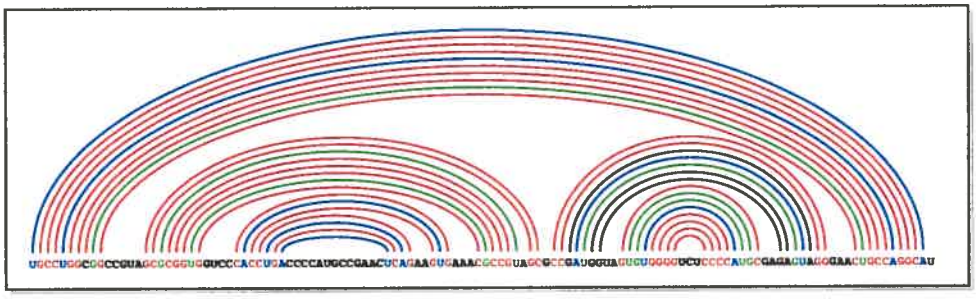


Figure 23 Séquence arcs-annotée.

Les séquences arc-annotées peuvent également être représentées sous forme circulaire (Figure 24). Elles occupent ainsi moins d'espace et seront pour certaines personnes plus faciles à visualiser.

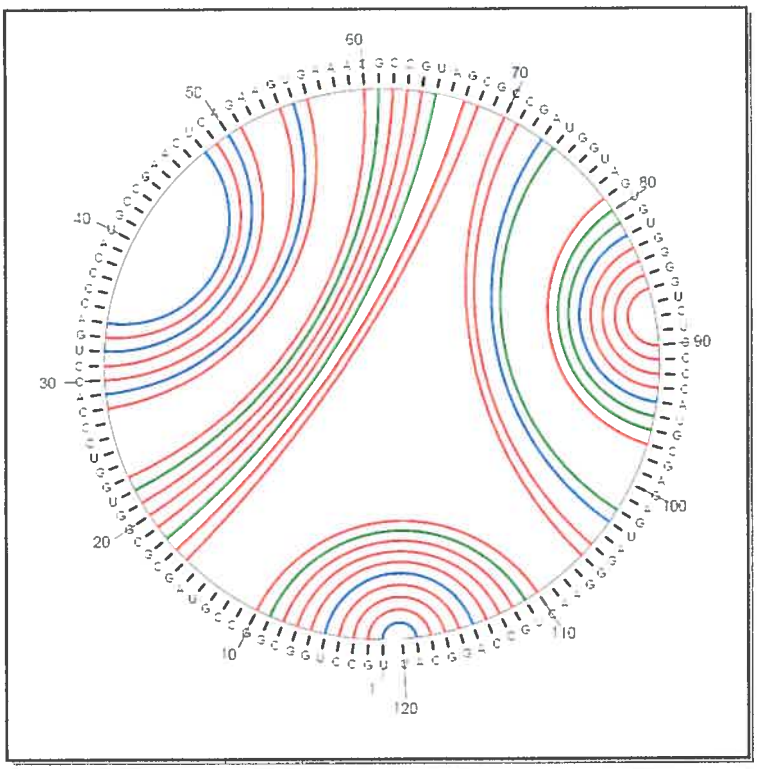


Figure 24 Modélisation circulaire d'une structure secondaire.

Une légère variante des séquences arcs-annotées a été conçue pour une représentation informatique de ces séquences; ce sont les séquences annotées points-parenthèses (Figure 25). Chaque caractère de la séquence est annoté par un autre caractère: il s'agit d'un point lorsque la base est libre, d'une parenthèse ouvrante (ouverture d'arc) lorsque l'on rencontre la première base d'une paire, et enfin d'une parenthèse fermante (fermeture d'arc) lors qu'il s'agit de la seconde base de la paire. Cette modélisation, fait apparaître deux séquences, l'une correspondant aux acides nucléiques et l'autre aux points-parenthèse, qui peuvent être traitées indépendamment si nécessaire.

Figure 25 Séquence annotée points-parenthèses.

Pratiquement jamais utilisée car elle ne présente que peu d'intérêt, la modélisation en montagnes d'une structure est également possible (Figure 26).

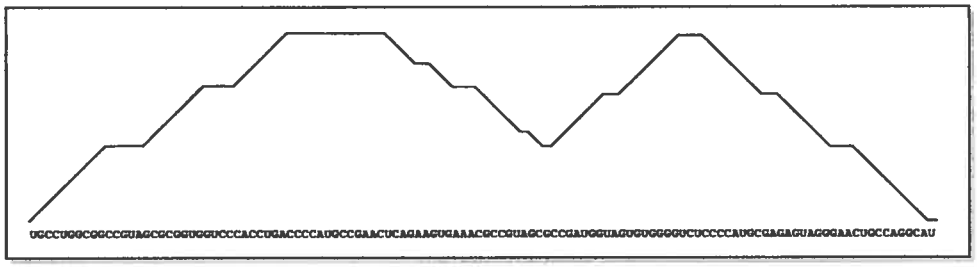


Figure 26 Modélisation en montagne d'une structure secondaire.

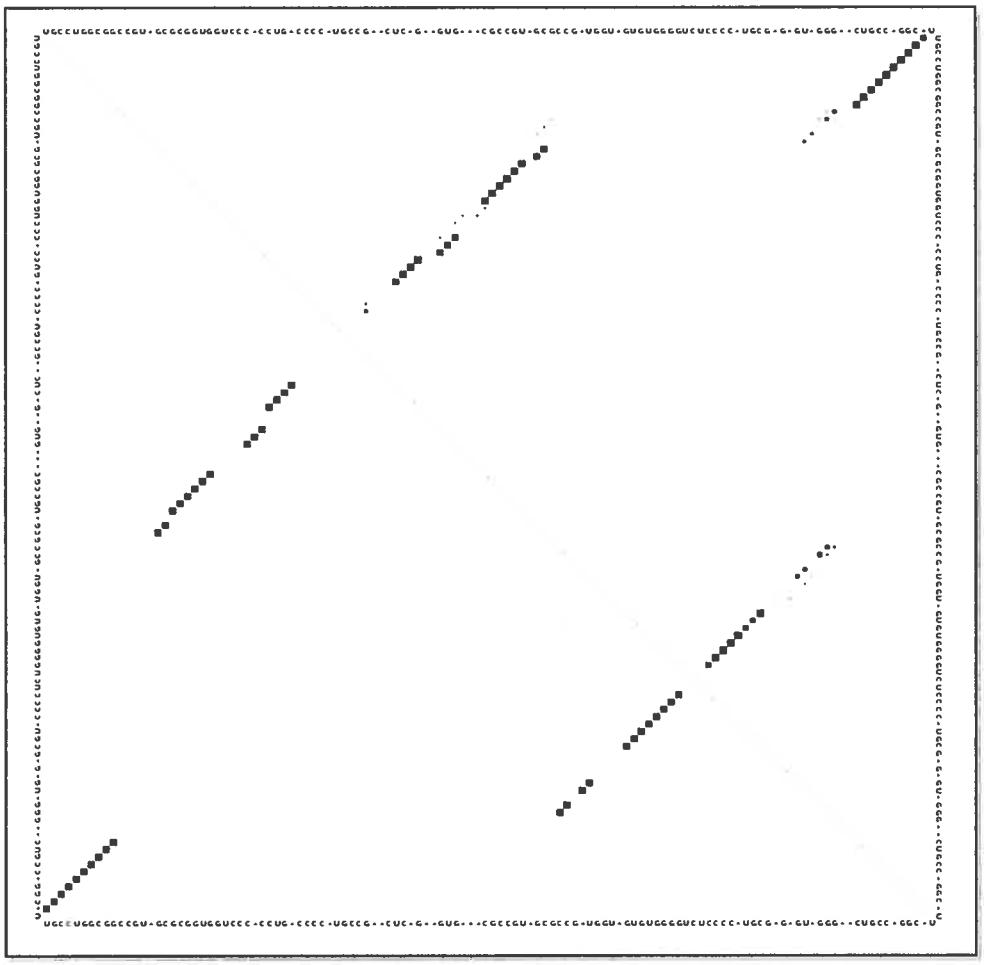


Figure 27 Matrice des appariements.

Dans cette modélisation, chaque nouvelle paire ajoute un niveau et chaque paire terminée en retire un. Le graphique ainsi obtenu permet de faire ressortir les ruptures dans les empilements de bases ainsi que les boucles terminales (sommets de montagnes).

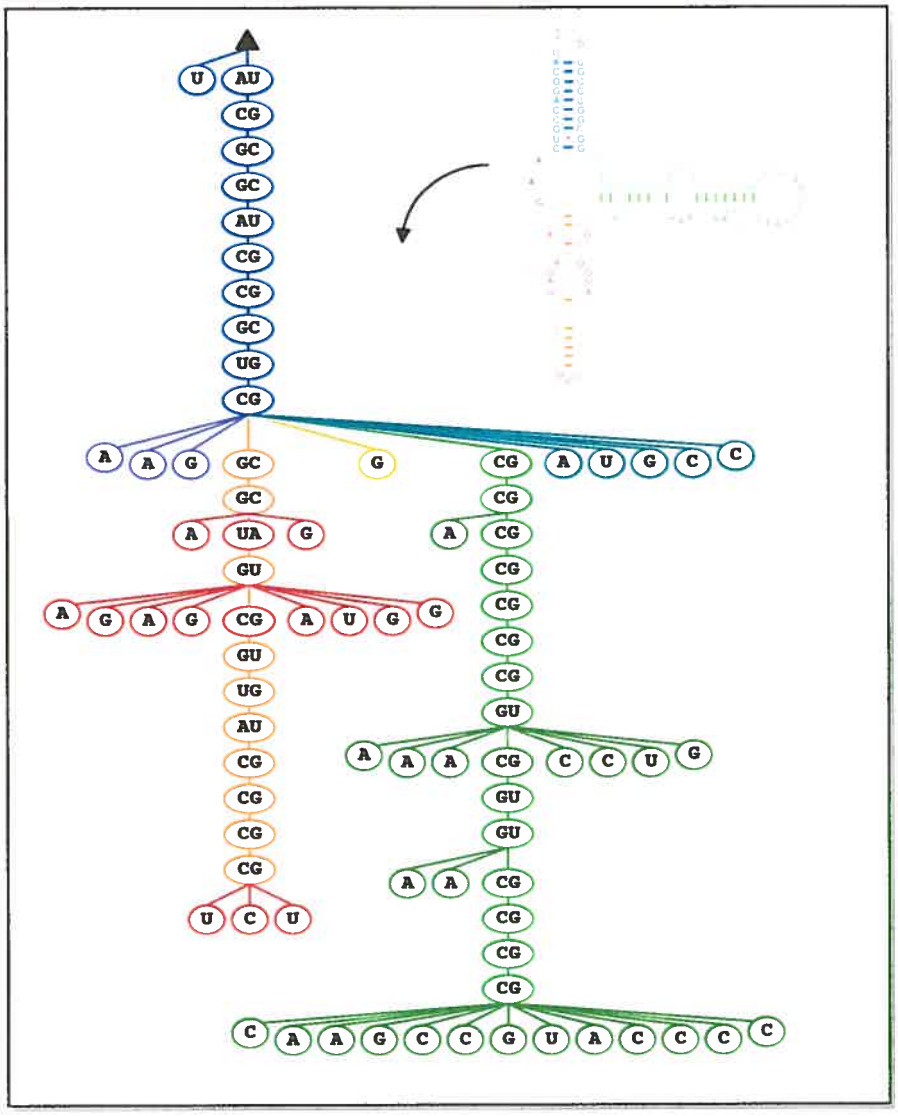


Figure 28 Représentation en arbre la plus courante d'une structure secondaire d'ARNr 5S de E. Coli (en haut à droite).
Chaque nœud interne représente une paire de base et chaque feuille une base libre. Les couleurs permettent de mieux visualiser les correspondances entre la structure et sa représentation.

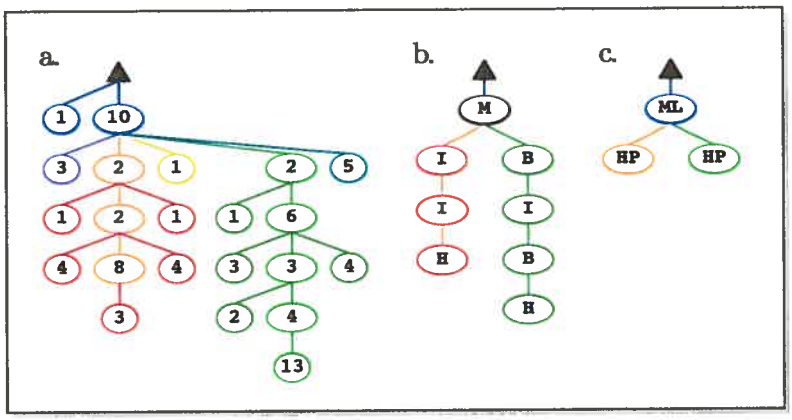


Figure 29 Représentation en arbre à faible niveau de détail d'une structure secondaire d'ARNr 5S de E. Coli présentée Figure 28. Les couleurs mettent en évidence les correspondance entre les différentes représentations.

a. Chaque nœud interne correspond au nombre de paires de bases entre deux sous-éléments structuraux et chaque feuille au nombre de bases libres entre les empilements de paires;

b. chaque nœud correspond à un sous-élément structurel. M pour les boucles multiples (multi-banch loops), I pour les boucles internes (internal loops), B pour les renflements (bulge) et H pour les boucles terminales (hairpin loops). Les empilements de paires sont représentés par les tiges de l'arbre;

c. Les sous-éléments structuraux sont regroupés en macro-sous-éléments: ML pour les boucles multiples et HP pour les tiges-boucles (hairpins). Les empilements de paires sont également représentés par les tiges de l'arbre.

Pour calculer les *appariements de bases* possibles, les algorithmes de prédiction de repliement d'ARN comme RNAFold ([19]) ou MFold ([44]) travaillent avec des tables semblables à celle que produirait un alignement de la séquence de la structure avec sa version complémentaire en termes de bases. De telles matrices *d'appariements* peuvent être également utilisées pour modéliser une structure secondaire d'ARN. Chaque colonne et chaque ligne correspondent à un caractère de la séquence; un point dans la matrice indique alors un pont hydrogène entre les deux bases. Lorsqu'il s'agit d'une matrice de prédiction des

appariements possibles, l'importance du point reflète la probabilité de formation d'un pont hydrogène.

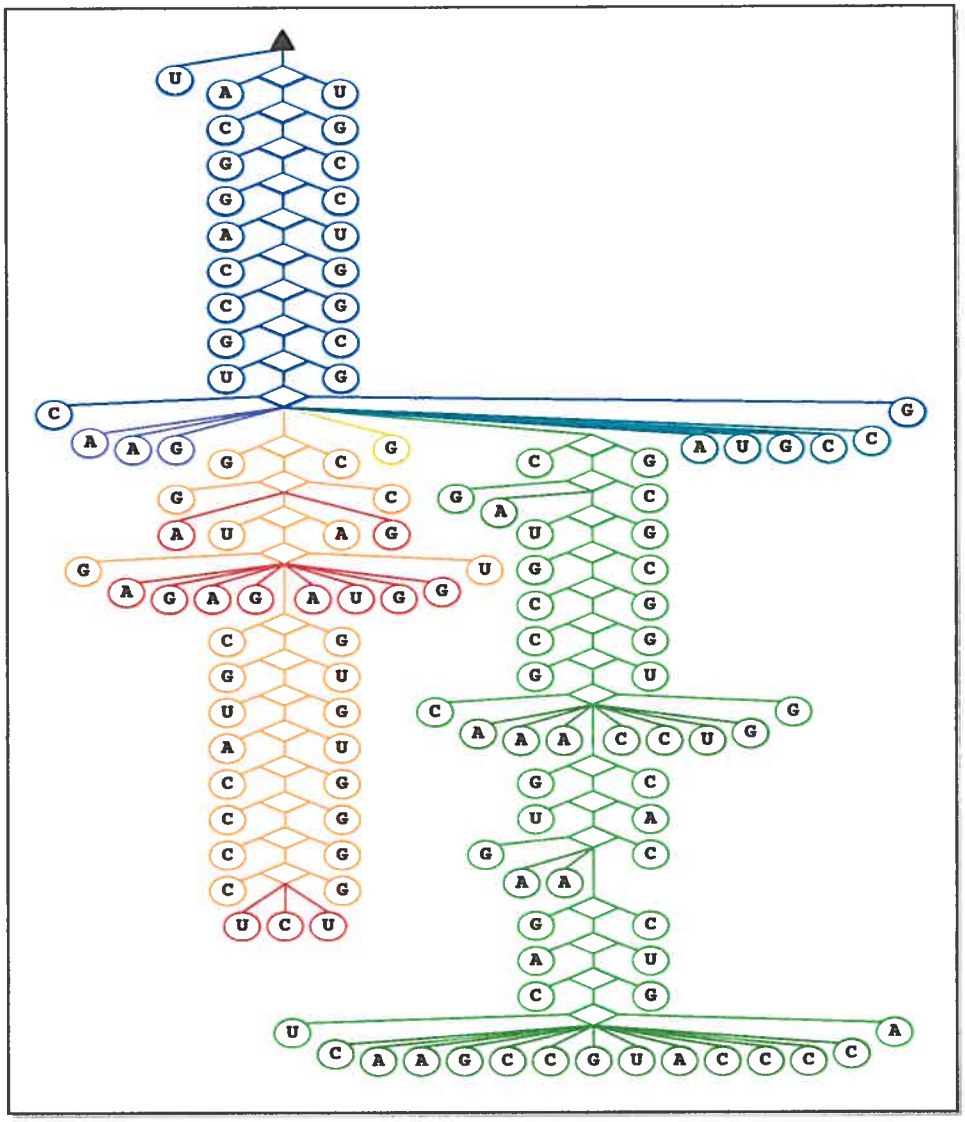


Figure 30 Représentation en arbre détaillant les ponts hydrogènes d'une structure secondaire d'ARNr 5S de E. Coli. présentée Figure 28. Les couleurs mettent en évidence les correspondance avec la structure.
Les ponts hydrogènes sont modélisés parles nœuds internes de l'arbre (losanges).

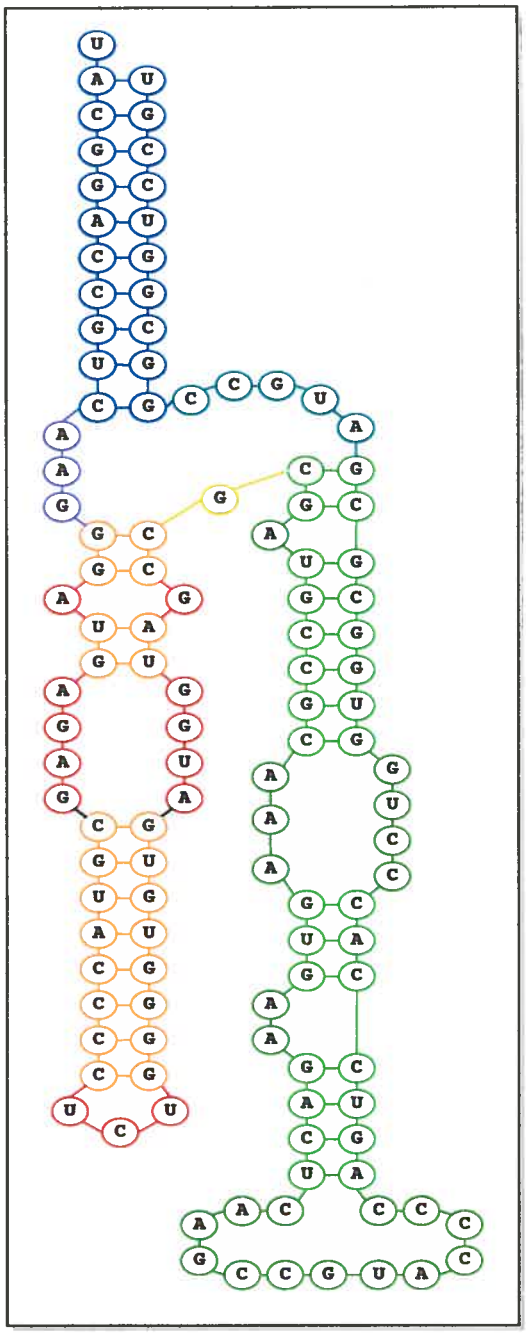


Figure 31 Représentation en graphe d'une structure secondaire d'ARNr 5S de E. Coli. présentée Figure 28. Les couleurs mettent en évidence les correspondance avec la structure.

La modélisation en arbre des structures secondaires est une autre forme de représentation qui offre de nombreux avantages. Tout d'abord, elle permet plusieurs niveaux de détails structurels (5 représentations de la même structure Figure 28, Figure 29 et Figure 30). Ensuite, son apparence est plus proche de la structure secondaire réelle que les autres représentations vues précédemment. Enfin, différents outils mathématiques ont été fournis pour manipuler ce modèle.

La modélisation la plus proche du schéma d'une structure secondaire est sans conteste celle en graphe (Figure 31). Son principal problème est qu'elle est plus complexe à manipuler qu'un arbre.

Parmi toutes les modélisations vues précédemment, la représentation en arbre est celle qui a emporté le plus de succès dans la littérature ([41]). Les séquences arc-annotées ([22]) et la modélisation sous forme de graphe ([28]) ont également fait l'objet d'algorithmes intéressants mais en moindre mesure. D'un point de vue théorique, à l'exception des modélisations en arbre ne faisant pas apparaître les étiquettes des bases, toutes ces modélisations renferment les mêmes informations et ne diffèrent que par leur aspect visuel ou pédagogique et par la façon de formaliser les outils pour les manipuler.

Pour nos travaux, nous nous sommes d'abord basés sur la modélisation en arbre de la Figure 28 qui était le modèle utilisé par l'algorithme que nous voulions améliorer. Nous nous sommes ensuite tournés vers une représentation originale sous forme d'automate suite à une nouvelle idée d'algorithme pour finalement revenir à une représentation en arbre en simplifiant ce que nous avions trouvé.

3.4 Opérations d'édition

Quelque soit le modèle considéré, une opération sur ce modèle correspond à une mutation biologique sur la structure de l'ARN. Les mutations les plus

simples correspondent aux modifications impliquant un seul nucléotide libre sur la séquence:

- Substitution d'une base libre par une autre (réétiquetage) (Figure 32a.);
- Suppression d'une base libre (Figure 32b.);
- Insertion d'une base libre (Figure 32c.);

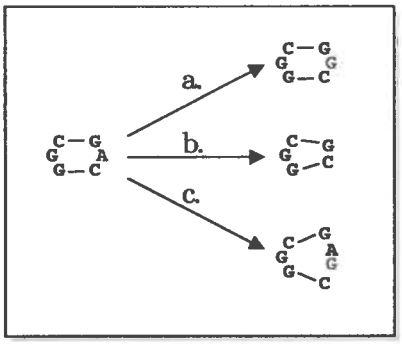


Figure 32 Opération d'édition correspondant à la mutation d'un seul nucléotide libre.
a. Substitution (ou réétiquetage) d'une base libre;
b. Suppression d'une base libre;
c. Insertion d'une base libre.

Des mutations portant sur un seul nucléotide peuvent aussi avoir un impact sur une paire:

- Substitution d'une base appariée par une autre base (Figure 33a.);
- Suppression d'une base appariée (altération) (Figure 33b.);
- Insertion d'une base permettant à une base libre déjà présente de s'apparier (complémentation) (Figure 33c.);

Deux bases peuvent également faire l'objet d'une seule *opération d'édition*:

- Substitution d'une paire par une autre (Figure 34a.);
- Suppression d'une paire de bases (Figure 34b.);
- Insertion d'une paire de bases (Figure 34c.);
- Création d'une paire à partir de deux bases libres (Figure 34d.);

- Bris d'une paire conduisant à la formation de deux bases libres (Figure 34e.);

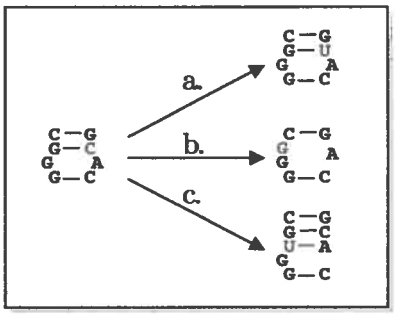


Figure 33 Opérations d'édition portant sur une base appariée.
a. Substitution d'une base appariée;
b. Altération d'une paire;
c. complémentation d'une paire.

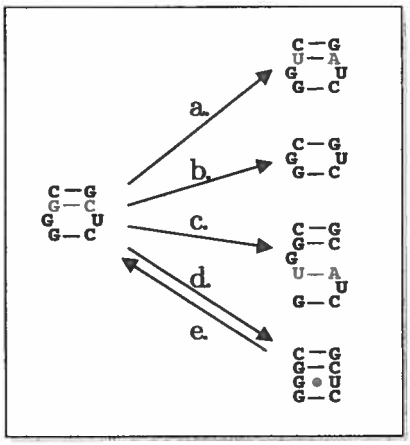


Figure 34 Opération d'édition touchant une paire de bases.
a. substitution d'une paire;
b. suppression d'une paire;
c. insertion d'une paire;
d. formation d'une paire à partir de deux bases libres;
e. bris d'une paire.

Un autre type d'*opération d'édition* portant sur 3 bases est à notre connaissance, absent de la littérature mais bien présent dans la réalité:

- le changement d'*appariement d'une base* (Figure 35);

Cette opération met en jeu deux bases appariées et une base libre généralement voisine. L'une des deux bases de la paire devient libre alors que l'autre s'apparie avec la base libre du voisinage. Bien qu'apparaissant comme complexe et certainement rare, cette opération peut se produire (Figure 36) et nous sommes capables de la prendre en compte dans notre modélisation.

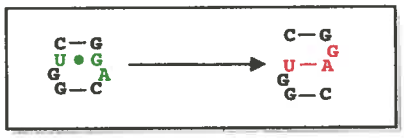


Figure 35 Changement d'appariement d'une base.

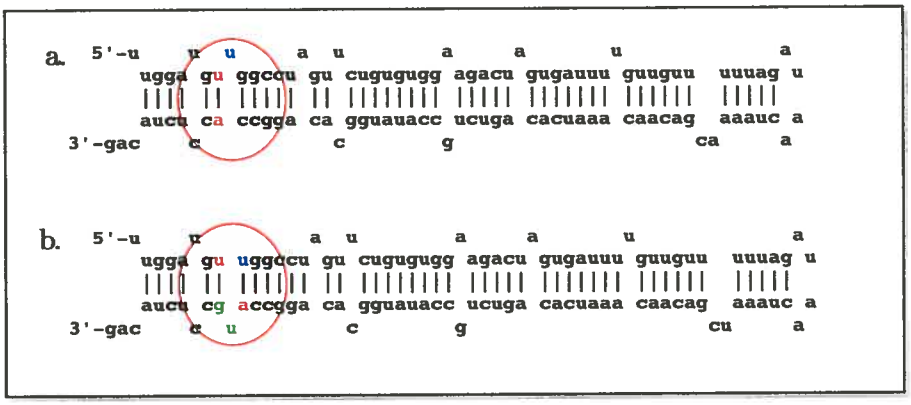


Figure 36 Exemple de changement d'appariements entre 2 miARN.

a. Mus Musculus miR-7-1 (mmu-mir-7-1);

b. Homo Sapiens miR-7-1 (hsa-mir-7-1).

L'insertion d'un fragment "ug" (en vert) dans le miARN humain a vraisemblablement induit un double rappariement:

- l'uracile en rouge sur le miARN de souris s'apparie avec la guanine insérée;
- l'adénine en rouge, s'apparie alors avec l'uracile en bleu qui était libre.

Ces différentes opérations peuvent être classées, du point de vue de leur complexité en 2 catégories: les opérations simples regroupant les opérations décrites dans [42] (i.e. les insertions, suppressions ou substitutions d'une base ou d'une paire de bases) et les opérations complexes comportant le reste des opérations. La prise en compte des opérations simples offre un bon niveau de

détail pour la comparaison de structures secondaires mais le plus haut niveau de détail, le plus précis, n'est obtenu qu'en prenant également en compte les opérations complexes. Cependant, quelque soit leur catégorie, ces mutations se situent à l'échelle des nucléotides et ont, isolément, un impact limité sur la structure.

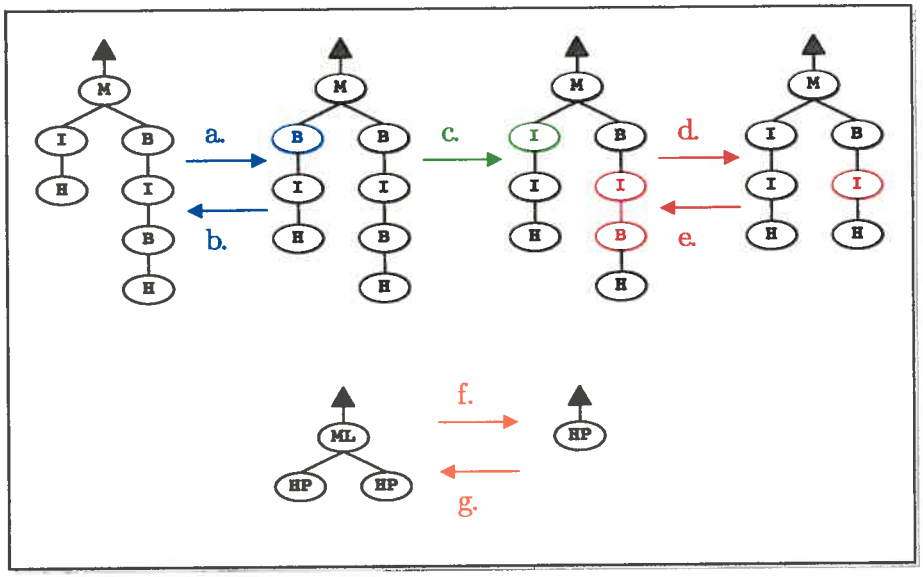


Figure 37 Macro-opérations d'édition.

a. Insertion d'un renflement;

b. suppression d'un renflement;

c. transformation d'un renflement en boucle interne;

d. fusion d'un renflement avec une boucle interne pour former une grande boucle interne (cela se produit par exemple lorsque les paires entre ces deux éléments sont brisées);

e. explosion d'une boucle interne en deux éléments (cela se produit par exemple lorsque des appariements se forment parmi les bases d'une boucle interne);

f. fusion de deux tiges-boucles pour former une seule grande tige-boucle (cela se produit par exemple lorsque toutes les appariements d'une petite tige-boucle sont brisés);

g. explosion d'une tige-boucle en trois éléments (cela se produit par exemple lorsque des bases appartenant à un renflement ou situées d'un même côté d'une boucle interne forment des appariements entre elles);

Si l'on considère les structures secondaires de façon plus abstraite comme étant des assemblages de sous-éléments structuraux, d'autres opérations peuvent apparaître, que l'on peut qualifier de "macro-opérations" (Figure 37, [1]):

- remplacement d'une sous-structure par une autre (Figure 37c.);
- insertion d'un sous-structure (Figure 37a.);
- suppression d'une sous-structure (Figure 37b.);
- fusion de 2 sous-structures séparées par une sous-structure différente supprimée (Figure 37d.);
- éclatement d'une sous-structure en 2 sous-structures séparées par une autre sous-structure insérée (Figure 37g);

Pour conclure cette section sur les *opérations d'édition*, nous remarquerons que chacune de ces opérations peut être obtenue à l'aide d'une combinaison d'autres opérations (Figure 38).

3.5 Schémas de score

Une fois le modèle et les opérations à appliquer à celui-ci choisis, il faut associer un poids à chacune des opérations de manière à calculer leur emploi. Ceci constitue ainsi un schéma de score.

Le choix d'un schéma de score n'est pas trivial. En effet, comme nous venons de le voir, il est par exemple possible de remplacer l'altération d'une paire par deux autres opérations consécutives: la suppression de la paire et l'insertion d'une base libre. D'un point de vu conceptuel, ceci pose un problème car il s'agit de deux opérations distinctes qui n'ont pas le même sens biologique même si elles conduisent in-fine au même résultat. Pour qu'un algorithme soit capable de déceler une altération, il faudra donc que le coût imparti à celle-ci soit plus faible que celui additionné de l'opération de suppression d'une paire et de l'insertion d'une base libre. Le choix des coûts est d'autant plus complexe que le nombre d'opérations autorisé est élevé. En effet, plus on dispose d'opérations,

plus les combinaisons de compositions possibles sont élevées et plus il est possible de trouver une composition défavorisant une opération unique.

En plus des contraintes sur la métrique des *opérations d'édition*, il faut également prendre en compte leur aspect biologique. Certaines opérations apparaîtront plus fréquemment que d'autres. Ceci peut être lié à des contraintes géométriques ou chimiques de la molécule d'ARN, ou encore à des contraintes biologiques impliquant des molécules extérieures. Par exemple, on peut

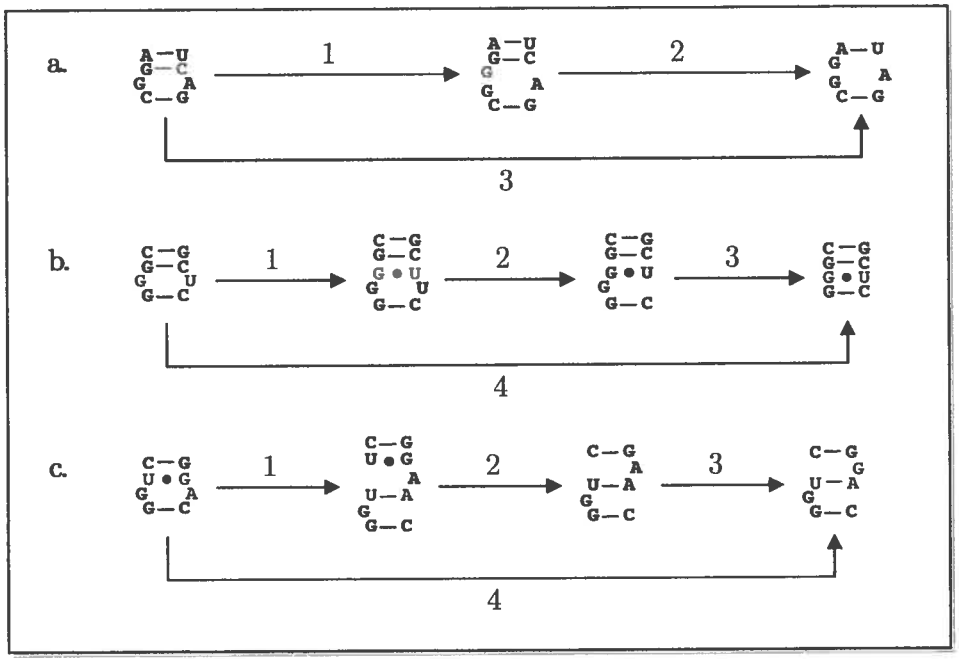


Figure 38 Quelques opérations d'édition et de compositions d'opérations menant à des résultats finaux identiques.

a. L'altération d'une paire **GC** (a.3) peut être obtenue par la composition de l'insertion d'une base **G** (a.1) suivie de la suppression de la paire **GC** (a.2);

b. la création d'une paire **GU** (b.4) peut être obtenue par la composition de l'insertion d'une paire **GU** (b.1) suivie des suppressions successives des bases libres **U** (b.2) et **G** (b.3);

c. le rappariement de la base **U** avec la base **A** (c.4) peut être obtenu par la composition de l'insertion d'une paire **UA** (c.1) suivie de la suppression de la paire **UG** (c.2) et de la mutation de la base libre **A** en **G** (c.3).

concevoir facilement que remplacer une paire G-U par une paire A-U coûte moins que son remplacement par une paire G-C car sur le plan physique, la paire G-C présentant 3 ponts hydrogènes est plus rigide et solide que les paires G-U ou A-U n'en possédant que 2. Il s'agit pourtant ici de la même *opération d'édition*! Le schéma de score comprend donc un coût associé à chaque opération mais ce coût peut varier en fonction des opérandes de l'opération.

On peut pousser le vice plus loin en attribuant des pénalités de coût suivant la région de la structure où s'opèrent des éditions. Imaginons par exemple qu'une tige-boucle soit caractéristique d'une famille de structures et que toute modification sur celle-ci dénaturerait la fonction de la molécule. Lors d'une comparaison dont le but serait de déterminer si oui ou non des structures appartiennent à cette famille, on voudra rapidement éliminer les structures présentant trop de mutations sur la tige-boucle d'intérêt. Un coût d'édition supplémentaire et spécifique à une région de la structure solutionnerait bien ce problème.

Et pour corser encore un peu le problème du schéma de score, il ne faut pas oublier qu'un évènement comme l'insertion ou la découpe d'une sous-séquence dans la séquence d'une structure peut se produire. C'est par exemple le cas lors du processus de maturation des ARNt. Si les versions précurseur et mature sont comparées, il est fort probable qu'un élément structurel entier apparaisse ou disparaisse entre elles. Mais le point important est que les mutations détectées seront consécutives et sans interruption. Pour faciliter la détection d'un tel évènement, il peut être intéressant de pondérer les coûts d'opérations consécutives. Ainsi, supprimer 5 bases libres consécutives devrait coûter moins que supprimer 5 bases libres non-consécutives par exemple. Ce problème est équivalent à celui des "pénalités de trous" (gaps penalty) pour les alignements de séquences ([12]).

Dès lors, le choix du coût à attribuer à chaque opération en fonction de ses opérandes devient extrêmement critique. Un tel choix, bien qu'essentiel, est trop

complexe pour faire parti de ce mémoire. Pour nos études, nous nous contenterons d'un schéma simple et arbitraire laissant une chance à peu près égale à chaque opération de se produire, notre but n'étant pas de fournir des alignements les plus exacts possibles mais plutôt de montrer que notre algorithme offre la précision nécessaire pour obtenir ces alignements. Pour que chaque opération puisse être utilisée, nous nous assurerons que son coût soit légèrement inférieur à n'importe quelle composition de d'autres opérations produisant un résultat équivalent. Par exemple, le coût de l'altération d'une paire sera légèrement inférieur à celui de la supression d'une paire plus l'insertion d'une base. Nous ajouterons que notre algorithme peut intégrer n'importe quel schéma de score ne prenant pas en compte les trous et que la construction d'un schéma de score idéal pour les structures secondaires pourrait s'inspirer des travaux de M. Dayhoff ([11]) et dépendrait des familles d'ARN mises en jeu ainsi que du domaine d'application de la comparaison de structures (i.e. le schéma ne serait pas le même pour de l'aide au repliement ou de la classification).

3.6 Quelques algorithmes

3.6.1 Alignements globaux

Le premier algorithme destiné à calculer une *distance d'édition* entre deux arbres est dû à Kuo-Chung Tai [26]. Mais un algorithme plus simple et moins gourmant en espace et en temps a été proposé par la suite par Zhang et Shasha [42]. Ce dernier était à l'origine destiné principalement à la comparaison de structures secondaires d'ARN. Basé sur la programmation dynamique, il calcule le nombre minimal d'opérations simples permettant de transformer une structure en une autre. Ce calcul s'effectue par une approche "bas-haut" (en anglais "bottom-up") en décomposant les 2 arbres d'une façon particulière, depuis les feuilles vers la racine de l'arbre. Le découpage de chaque arbre se fait en commençant à partir de la feuille la plus à gauche et en remontant jusqu'à

un nœud sans parent. Les nœuds ainsi parcourus forment un sous-arbre linéaire qui est extrait de la structure, laissant ainsi une forêt. L'opération est répétée récursivement sur la forêt restante jusqu'à ce qu'elle soit vide. L'algorithme compare ensuite dans des tables chaque sous-arbre d'une structure à ceux de l'autre et complète au fur et à mesure une table de distance globale entre les deux arbres. La complexité de cet algorithme a fait l'objet d'une étude approfondie dans [12] et est en $O(|T_1||T_2|)$ en espace et en $O(|T_1||T_2|\min(D_1, L_1).\min(D_2, L_2))$ en temps (où $|T_i|$, L_i, D_i et I_i représentent respectivement la taille, le nombre de feuille, la profondeur et le degré maximal de l'arbre T_i (cf. section notations)).

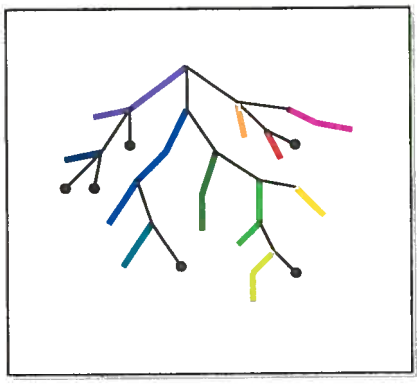


Figure 39 Décomposition d'un arbre suivant l'algorithme de Zhang-Shasha. Chaque feuille en gras correspond à un sous-arbre trivial et chaque ensemble d'arêtes en gras d'une même couleur correspond à un sous-arbre unique et distinct des autres.

9 ans plus tard, Klein [24] a proposé une autre façon de décomposer les arbres permettant de réduire la complexité des calculs dans le pire des cas. Plutôt que de décomposer un arbre à partir de ses feuilles gauches, cet algorithme le décompose à partir de ses chemins les plus lourds. Un chemin lourd est défini par le parcours descendant depuis un nœud vers une feuille en choisissant à chaque nœud de passer par le nœud fils le plus lourd (cf. Figure 40). Le poids d'un nœud correspond au nombre de nœuds contenus dans le sous-arbre dont il

est racine. La complexité en temps des calculs se trouve ainsi réduite en $O(|T_1|^2 \cdot |T_2| \log(|T_2|))$ pour une complexité en espace maintenue en $O(|T_1| \cdot |T_2|)$.

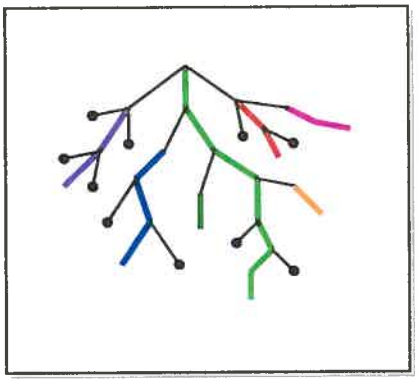


Figure 40 Décomposition d'un arbre en chemins lourds. chaque chemin lourd est indiqué d'une couleur en gras. Les points indiquent les chemins lourds triviaux composés d'une seule feuille. (source: [24])

Une autre voie a été ouverte par Jiang et al. [23]: le calcul de la distance d'alignement entre arbres plutôt que celui de la *distance d'édition*. La distance d'alignement est un cas particulier de la *distance d'édition* où les opérations d'insertion doivent être effectuées avant les opérations de suppression.

Très récemment, une autre technique a été proposée par Liu et al. [28] pour aligner des structures secondaires avec une complexité en temps en $O(n^2)$. Pour ce faire, l'algorithme décompose les structures en sous-éléments atomiques organisés à l'aide d'un arbre permettant de conserver l'information structurelle globale de la molécule donnant une sorte de graphe. Néanmoins, ce modèle ne prend en compte que les opérations simples d'édition et ignore complètement les opérations plus complexes.

Une première étape vers les opérations plus complexes a été franchie par Jiang et al. [22] en introduisant les opérations de bris/création de paires et les opérations d'altération/complémentation de paires. Dans leur article, ils démontrent que la prise en compte des opérations comme les bris ou les créations de paires rend le problème de l'alignement de structures secondaires

NP-complet. En imposant des contraintes sur le schéma de scores (i.e. en choisissant un schéma de score qui force le remplacement des opérations de bris-création de paires par une combinaison d'autres opérations), ils ont été en mesure de proposer un algorithme en temps polynomial en $O(n_1^3.n_2)$ capable de détecter les opérations d'altérations/complémentation de paires. Cependant, aucun algorithme, même exponentiel, n'a été proposé pour prendre également en compte les autres opérations complexes.

L'algorithme d'Allali et al. [1], qui est d'une complexité exponentielle en temps et en espace de $O((2I)^{\ell}.|T_1|.\min(D_1, L_1).|T_2|.\min(D_2, L_2))$ (où $I = \max(I_1, I_2)$ et ℓ le nombre maximal de fusions consécutives autorisées), ouvre la voie vers la prise en compte des macro-opérations. Cet algorithme basé sur celui de Zhang-Shasha [42] s'intéresse plutôt aux structures secondaires représentées à un faible niveau de détail.

3.6.2 Alignements locaux (recherche de motifs)

L'application la plus évidente des algorithmes de comparaison de structures secondaires est de fournir une distance entre des structures ou un alignement ou les deux. Ces informations sont utiles aux biologistes qui souhaitent connaître les distances évolutives ou encore savoir où se situent les différences entre deux structures. Cependant, d'autres biologistes peuvent être intéressés par certains motifs structuraux d'ARN qui peuvent par exemple représenter la cible ou encore être le point d'ancrage d'une autre molécule. Dans ce cas, il peut s'avérer utile de rechercher un motif dans une structure, ce qui peut être effectué par un alignement local de celle-ci avec le motif cherché.

Höchsmann et al. [18] traitent se problème en proposant un algorithme en temps $O(|T_1|.|T_2|.D_1.D_2.(D_1+D_2))$ et en espace $O(|T_1|.|T_2|.D_1.D_2)$. Leur algorithme est basé sur celui présenté par Jiang et al. [23] qu'ils ont modifié pour effectuer des alignements locaux. De plus, ils présentent une nouvelle modélisation en arbre des structures secondaires permettant d'employer toutes les *opérations d'édition*

décrites dans [22] tout en s'affranchissant des contraintes sur le schémas de score. Cependant, cette amélioration s'effectue au détriment d'un arbre plus lourd à manipuler.

Backofen et al. [36] ont également proposé un algorithme d'alignement local basé quant à lui sur [22]. Cet algorithme fondé sur des principes biologiques est capable de détecter des motifs qui ne peuvent pas être décelé par [22]. Sa complexité en temps est en $O(n_1^2.n_2^2.\max(n_1, n_2))$ et en espace en $O(n_1.n_2)$ (où n_1 et n_2 sont respectivement les longeurs de la première et de la seconde séquence de la comparaison).

3.6.3 Bilan

Dans cette section, nous avons vu divers approches, modèles et niveaux de possibilités d'algorithmes pour comparer des structures secondaires d'ARN. Le Tableau III résume les principales caractéristiques de ces algorithmes.

Référence	Complexité en temps	Complexité en espace	Type	Modèle	Jeu d'opérations
ZS89 [42]	$O(\lvert T_1\rvert.\min(L_1,D_1).\lvert T_2\rvert.\min(L_2,D_2))$	$O(\lvert T_1\rvert.\lvert T_2\rvert)$	Global	Arbres	Simples
Kle98 [24]	$O(\lvert T_1\rvert^3.\lvert T_2\rvert.\log(\lvert T_2\rvert))$	$O(\lvert T_1\rvert.\lvert T_2\rvert)$	Global	Arbres	Simples
JWZ95 [23]	$O(\lvert T_1\rvert.\lvert T_2\rvert.(I_1+I_2)^2)$	$O(\lvert T_1\rvert.\lvert T_2\rvert.(I_1+I_2)^2)$	Global	Arbres	Simples
LWHT05 [28]	$O(n_1.n_2)$	$O(n_1.n_2)$	Global	Mixte arbre-graphe	Simples
JLMZ01 [22]	$O(n_1^3.n_2)$	$O(n_1^3.n_2)$	Global	Séquence arc-annotée	Partiellement complexes
AS04 [1]	$O(I_1'.\lvert T_1\rvert.\min(L_1,D_1).\lvert T_2\rvert.\min(L_2,D_2))$	$O(I_1'.\lvert T_1\rvert.L_2'.\lvert T_2\rvert)$	Global	Arbres	Macro-opérations
HTGK03 [18]	$O(\lvert T_1\rvert.\lvert T_2\rvert.D_1.D_2.(D_1+D_2))$	$O(\lvert T_1\rvert.\lvert T_2\rvert.D_1.D_2)$	Local	Arbre particulier	Complexes
BW04 [36]	$O(n_1^2.n_2^2.\max(n_1, n_2))$	$O(n_1, n_2)$	Local	Séquence arc-annotée	Complexes

Tableau III Récapitulatif des principaux algorithmes destinés à la comparaison de structures secondaires (source: [6]).
n représente le nombre de bases dans la séquence d'ARN;
$\lvert T_i\rvert$, L_i, D_i et I_i représentent respectivement la taille, le nombre de feuilles, la profondeur et le degré maximal de l'arbre T_i (cf. section notations);
ℓ représente le nombre maximal autorisé de fusions consécutives.

Chapitre 4

TRAVAUX EFFECTUES ET CONTRIBUTION

4.1 But initial recherché

Lorsque ce projet de recherche a débuté, il avait pour but de mettre au point un algorithme très rapide de calcul de *distance d'édition* entre deux structures secondaires d'ARN. En effet, aucun algorithme à notre connaissance n'était disponible pour comparer rapidement et massivement des structures secondaires entre elles de façon analogue à ce qu'effectuait BLAST sur des séquences biologiques. De plus, comme nous l'avons mentionné à la section 3.1, les applications possibles de la comparaison de structures secondaires sont vastes et il y a fort à parier que les données sur les structures secondaires vont croître dans les années à venir de façon exponentielle à l'instar de ce qui s'est passé avec les séquences nucléotidiques. C'est pour cela que nous avons donc jugé opportun de nous pencher sur ce problème.

L'idée de départ était de mettre au point une version *bit-vectorisée* de l'algorithme de Zhang-Shasha [42], en se basant sur des travaux similaires effectués pour le calcul de *distance d'édition* entre séquences ([4]). Un algorithme *bit-vectorisel* effectue des opérations sur des vecteurs de bit, c'est-à-dire qu'il applique simultanément une même opération à plusieurs données concaténées en une seule. En pratique, ce procédé permet aux processeurs de traiter parallèlement x données stoquées dans un même mot processeur et donc de diviser théoriquement le temps de calcul par x. Dans sa version originale, l'algorithme de Zhang-Shasha se prête mal à la bit-vectorisation (des valeurs à calculer consécutivement ne sont pas toujours sujettes au même traitement), c'est pourquoi nous avons eu l'idée de travailler sur les tiges-boucles plutôt que sur des structures globales. Nous espérions ainsi pouvoir tirer profit des

caractéristiques structurelles des tiges-boucles pour simplifier l'algorithme de Zhang-Shasha et atteindre notre but.

Le découpage en tige-boucle des structures n'est pas dénué de sens. La fonction assurée par une molécule d'ARN non codante dépend essentiellement de sa configuration spatiale. Celle-ci est étroitement liée à la formation d'*appariements de bases* que l'on retrouve dans les structures secondaires. Le nombre, les dimensions et l'ordre des tiges-boucles sur une structure sont la signature de sa fonction. Ainsi, les ARNt sont caractérisés par 4 tiges-boucles reliées à une même boucle multiple, les ARNr 5S présentent quant à eux 3 tiges-boucles et les RNase P ont également une forme caractéristique. Bien entendu, il y a toujours des exceptions mais elles sont très rares. Il existe par exemple des ARNt comprenant une $5^{ème}$ tige-boucle même si celle-ci est plutôt courte. Le faible nombre de cas particuliers nous a confortés dans l'idée que la décomposition en tige-boucle des structures capture l'essentiel de l'information structurelle.

Pour récapituler, les tiges-boucles sont des éléments sans ramifications et apparaissent ainsi plus simples et donc plus rapides à comparer. Notre intuition a donc été de découper des structures en tiges-boucles, de les comparer entre elles, de détecter les tiges-boucles similaires et de s'en servir comme ancre pour construire un alignement global final. Nos efforts se sont portés sur l'algorithme de comparaison des tiges-boucles dans le but de le bit-vectoriser pour le rendre extrêmement rapide.

4.2 Algorithme

Dans ce sous-chapitre, nous allons dans un premier temps expliquer le cheminement de nos travaux qui nous a permis d'obtenir un algorithme pour comparer des structures secondaires d'ARN. Ensuite, nous exposerons cet algorithme en deux phases. La première phase décrira l'algorithme servant à calculer la distance exacte entre deux tige-boucles. La deuxième phase

introduira un second algorithme intégrant le premier et permettant d'estimer une distance globale entre deux structures secondaires d'ARN.

4.2.1 Cheminement

Nos tentatives pour apporter une version bit-vectorisée se sont révélées infructueuses. En effet, même si la bit-vectorisation nous est apparue possible, elle se retrouve limitée à des portions des matrices des distances. L'algorithme à mettre au point se révèle alors assez compliqué car il faut tout d'abord délimiter les zones des matrices sur lesquelles il peut être appliqué et au final, le gain de temps ne nous a pas paru énorme.

Nous avons ensuite modifié notre approche en laissant de côté l'algorithme de Zhang-Shasha et sa bit-vectorisation pour nous tourner vers une autre idée, celle d'exploiter la linéarité apparente des tiges-boucles afin de trouver un autre algorithme. Le résultat a été un algorithme ([15]) basé sur une modélisation des tiges-boucles sous forme d'*automate des mélanges* ([21]), c'est-à-dire d'un automate qui permet de fusionner deux mots en un seul de toutes les façons possibles tout en respectant l'ordre des lettres de chacun des mots. L'idée sous-jacente à cette approche est qu'une tige-boucle peut être représentée par un ensemble de séquences de symboles où chaque symbole est soit une paire de base, soit une base libre à laquelle nous assignons un côté (Figure 41b.). La construction d'un automate complet décrivant une structure secondaire est trop longue pour pouvoir être abordée dans ce mémoire et a fait l'objet d'un article dédié et publié [15]. Le lecteur pourra cependant se faire une idée du procédé à l'aide de la Figure 41.

En observant ce modèle, nous avons remarqué qu'il pouvait servir à calculer la distance entre deux tiges-boucles sur une seule table et de surcroît prendre en compte les *opérations d'édition* complexes d'altération/complétion et de bris/création de paires tout en se limitant à une complexité en temps et en espace en $O(n^4)$. Cette complexité égale à celle de l'algorithme de Zhang-Shasha,

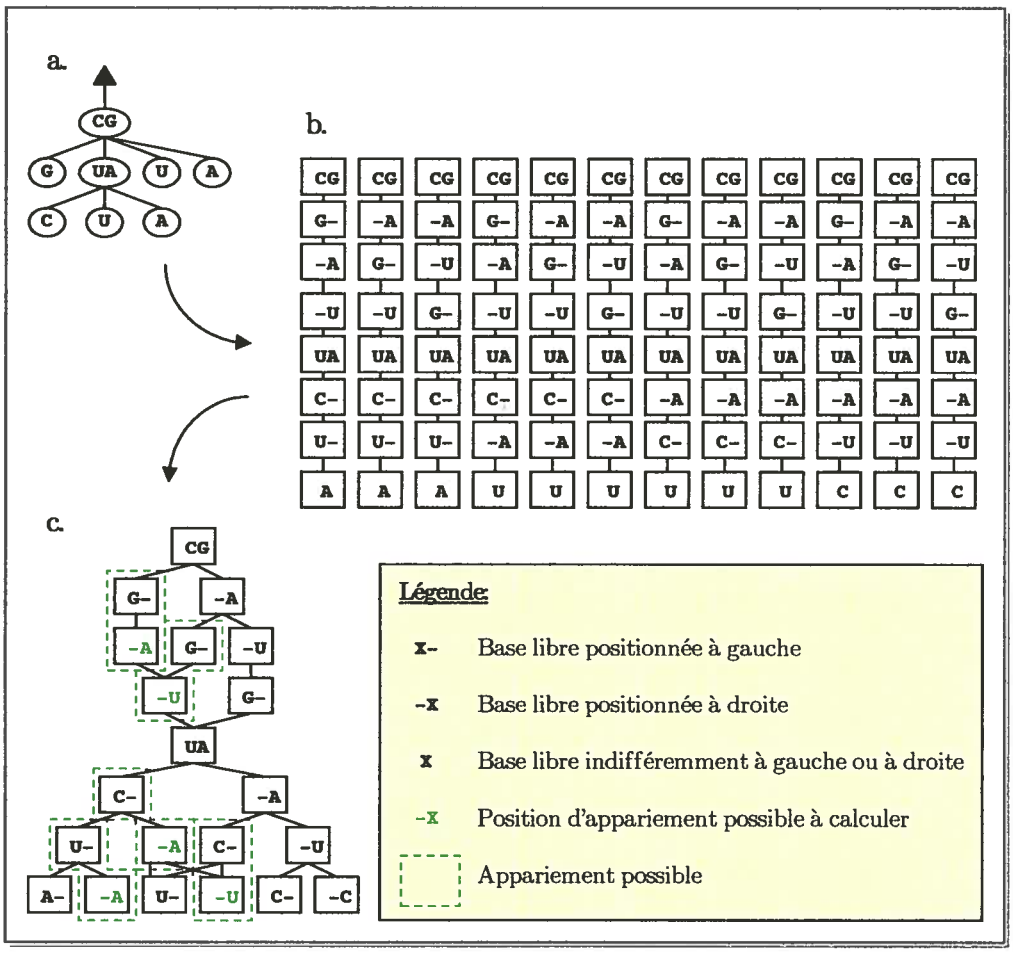


Figure 41 Construction d'un *automate des mélanges* représentant une tige-boucle d'ARN.

La structure modélisée sous forme d'arbre (a.) peut être vue comme une liste exhaustive de séquences (b.) qui peuvent être fusionnées sous la forme d'un *automate des mélanges* (c.).

ne reflète cependant pas la réalité sur des exemples concrets. En fait, moins les tiges-boucles présentent de bases libres dans leurs boucles, plus l'algorithme à un comportement quadratique. En pratique sur certains exemples, notre algorithme basé sur les *automates des mélanges* requiert de 2 à 7 fois moins de *cycles de calculs* que l'algorithme de Zhang-Shasha et offre en plus la prise en compte d'opérations complexes. Cependant, sur d'autres exemples comportant

des grandes boucles, il s'avère bien moins rapide à cause des calculs des opérations complexes de bris-création de paires.

Même s'il est plus rapide que celui de Zhang-Shasha, qu'il travaille sur une seule table de distance en mémoire et qu'il fonctionne avec un jeu d'opérations plus large, notre algorithme n'est pas encore parfait. Tout d'abord, les calculs des opérations complexes se font suivant une contrainte de "localité". Ainsi, la création d'une paire ne peut se faire qu'avec deux bases libres situées de part et d'autre d'une même boucle (Figure 42). Une autre contrainte que nous avons dénommée "gauche-droite", ne permet pas l'alignement d'une base située d'un côté d'une tige-boucle avec une base située d'un autre côté d'une autre tige-boucle (Figure 43). En revanche, les bases de la boucle terminale n'ont pas de

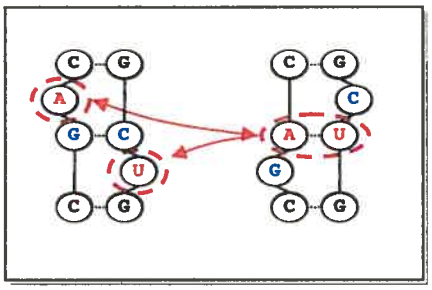


Figure 42 Contrainte de localité.
Cette contrainte interdit par exemple l'alignement des bases **A** et **U** à gauche avec la paire de base **AU** à droite.

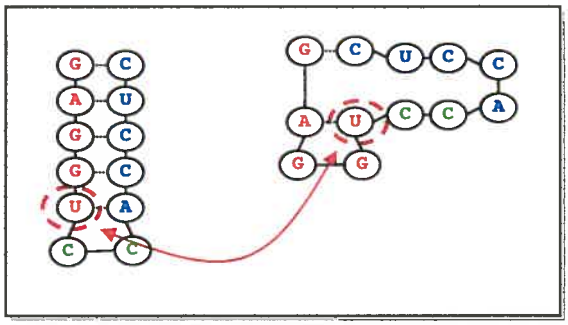


Figure 43 Contrainte gauche-droite.
Cette contrainte interdit par exemple l'alignement des **U** encerclés.

côté et peuvent être alignées indépendamment à gauche ou à droite. De plus, les calculs de bris/création de paires sont parfois redondants. Ensuite, l'opération de changement d'appariement n'est pas prise en compte. Et pour terminer, une tige-boucle ne peut pas être alignée avec plus d'une autre tige-boucle.

En reformulant notre algorithme, nous en avons trouvé un autre s'inspirant du principe de l'*automate des mélanges* cherchant à aligner la structure d'une extrémité à l'autre sans la décomposer. Cet algorithme ([16]) utilise une modélisation en arbre et introduit la notion de "paires indexantes" qui sera décrite plus loin dans ce mémoire. Ces paires sont un moyen de modéliser entièrement une structure tout en offrant la possibilité de détecter des opérations complexes de modifications.

Dans sa première version publiée, notre algorithme basé sur les "paires indexantes" était sujet aux mêmes contraintes que celui des *automates des mélanges*. Néanmoins, les calculs de rappariement n'étaient plus redondants, ce qui a divisé en moyenne par 2 le nombre de *cycles de calculs* nécessaires (moyenne calculée sur un ensemble de 200 tiges-boucles de miARN présentant moins de 40 bases libres dans une même boucle). En approfondissant nos recherches, nous avons trouvé comment s'affranchir des contraintes de localité et "gauche-droite" et également comment prendre en compte l'opération de changement d'appariement. Ceci s'effectue en "relaxant" les deux tiges-boucles, c'est-à-dire en brisant artificiellement toutes les paires, et en imposant des coûts d'édition en fonction des bases mises en jeu. Par exemple, un appariement A-U est brisé en A et U; le coût pour reformer cette paire est nul tandis que le coût pour former une nouvelle paire avec l'une de ces bases seule sera additionné de la moitié du coût du bris de la paire A-U, l'autre moitié étant ajouté lors du calcul du coût l'*opération d'édition* effectué sur l'autre base de la paire. Nous remarquerons au passage, sans le démontrer, qu'il suffirait en fait de relaxer une seule des deux structures pour arriver au même résultat.

Il reste un problème qui n'a pas encore été abordé, celui de l'alignement global de structures secondaires qui comportent plus d'une tige-boucle. Comme nous l'avions mentionné précédemment, toute structure secondaire peut être décomposée en un ensemble de tiges-boucles (Figure 44a.). Notre première approche pour construire des alignements globaux a été de décomposer chaque structure en ensemble de tiges-boucles ordonnées suivant leur rencontre lors du parcours de la séquence de l'ARN correspondant. Pour comparer 2 structures, nous comparions chaque tige-boucle d'un ensemble avec celles de l'autre. Nous construisions ensuite l'alignement des tiges par une table de programmation dynamique cherchant la correspondance optimale entre une tige-boucle d'une structure avec une tige-boucle de l'autre structure. Nous obtenions ainsi un alignement excluant cependant les bases libres des boucles multiples et incapable de faire correspondre une tige-boucle à plusieurs tiges-boucles ce qui pouvait s'avérer nécessaire dans certains cas.

Le problème de la prise en compte des bases libres de boucles multiples peut être résolu de plusieurs façons plus ou moins élégantes. La première consisterait à aligner comme des séquences les bases libres entre deux tiges-boucles d'une structure avec les bases libres des tiges-boucles correspondante sur l'autre structure. Une autre idée serait d'assigner une série de bases libres d'une boucle multiple à la tige-boucle qui les précède (ou qui les suit). Il est également possible de répartir ces bases entre chacune des 2 tiges entre lesquelles elles se trouvent.

En ce qui concerne l'alignement d'une tige-boucle avec un ensemble de tiges-boucles, le problème est plus délicat. En effet, l'alignement d'une seule tige-boucle avec plusieurs n'a de sens que si l'extrémité terminale de celle-ci est intégralement alignée avec une seule tige-boucle de l'ensemble et que la partie restante est alignée avec une ou plusieurs tiges consécutives de l'ensemble en amont de la tige-boucle déjà alignée. Il faut donc être capable d'aligner une tige-boucle depuis son extrémité terminale vers sa racine ce que nous pouvons justement faire avec notre algorithme. Il suffit pour cela d'inverser le sens de

calcul de la table de distance. L'alignement partiel optimal d'une tige ou tige-boucle avec une autre pourra être déterminé en regardant la dernière ligne ou la dernière colonne de la table de distance et en y localisant la valeur minimale. Cette valeur délimitera l'alignement complet d'une tige ou tige-boucle avec un préfixe de l'autre. Il est alors trivial d'extraire le suffixe qui n'a pas été aligné pour en faire une nouvelle tige ou tige-boucle artificielle. Ce nouvel élément artificiel pourra ensuite être aligné suivant la même méthode avec la tige mère de la tige-boucle intégralement alignée et provenant de l'autre structure.

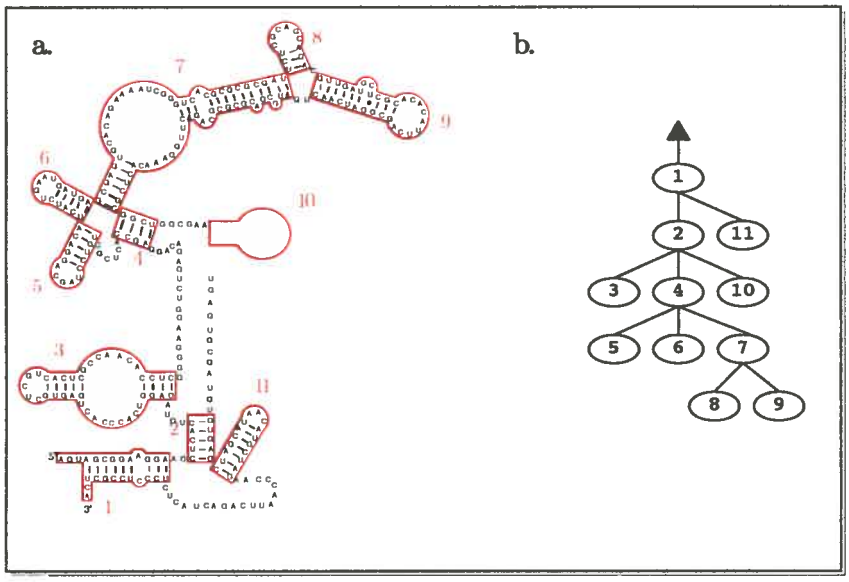


Figure 44 Décomposition d'une structure secondaire en arbre de tiges-boucles.
a. Chaque tige ou tige-boucle est encerclée en rouge. Le lecteur remarquera que les bases libres des boucles multiples sont exclues de cette décomposition cependant, elles pourraient tout à fait être incluses à la tige-boucle qui se trouve en leur extrémité 5' par exemple;
b. les tiges et tiges-boucles sont numérotées et positionnées dans un arbre de tiges-boucles reflétant la structure secondaire associée.

Pour faciliter la recherche des tiges parentes, nous allons modéliser une structure sous forme d'arbre où chaque feuille ou nœud interne représente une tige-boucle ou une tige. L'abre ainsi obtenu sera appelé "arbre de tiges-boucles"

(Figure 44b). Nous pouvons ensuite nous baser sur l'algorithme classique d'alignement d'arbre de Zhang-Shasha [42] pour aligner deux arbres de tiges-boucles car nous connaissons les *distances d'édition* entre chaque nœud de ces arbres et nous pouvons en plus prendre en compte lors des calculs les alignements d'une tige ou tige-boucle avec plusieurs autres si de tels alignements existent.

4.2.2 Formulation de l'algorithme

4.2.2.1 Distance entre tiges-boucles

Soient un ensemble d'opérations d'édition E, un schéma de score S associant à chaque opération un réél positif ou nul (coût) donné et une fonction de coûts γ prenant en paramètre une séquence d'opérations d'édition de E et retournant la somme de leurs coûts associés par S.

Nous contraignons S pour que γ soit une mesure de distance, c'est-à-dire:

(1) $\gamma(m \to m) = 0,\ \gamma(m \to n) \geq 0$;

(2) $\gamma(m \to n) = \gamma(n \to m)$;

(3) $\gamma(m \to q) \leq \gamma(m \to n) + \gamma(n \to q)$.

Soient deux tiges-boucles S_1 et S_2 représentées respectivement par des arbres ordonnés T_1 et T_2 tels que pour chaque arbre la paire de bases nulle " – – " soit racine et chaque base a de la séquence de la tige-boucle correspondante soit une feuille de la racine en conservant l'ordre issu de la séquence.

Remarque: T_1 et T_2 comportent au plus un seul nœud interne.

Définition: la *distance d'édition* D_e entre T_1 et T_2 est le coût minimal d'une séquence d'opération Ω parmi l'ensemble des séquences d'opérations permettant de transformer T_1 en T_2.

$$D_e(T_1, T_2) = \min_{\Omega}\{\gamma(\Omega) | \Omega(T_1) = T_2\}$$

Définition: soit $p(x)$ la fonction se lisant "prédécesseur de x" qui pour un nœud x d'un arbre retourne son prédécesseur, c'est-à-dire son frère

immédiatement à sa gauche s'il en a un ou son parent dans le cas contraire. Dans le cas particulier où x est un nœud sans parent, $p(x) = x$.

Définition: soit $s(x)$ la fonction se lisant "successeur de x" qui pour un nœud x d'un arbre retourne son successeur, c'est-à-dire son frère immédiatement à sa droite s'il en a un ou son parent dans le cas contraire. Dans le cas particulier où x est un nœud sans parents, $s(x) = x$.

Soit x et y des nœuds de T_1.

Définition: x est enfant gauche de y ssi x est une feuille qui se trouve à gauche de tout nœud fils de y qui n'est pas une feuille. Symétriquement, x est enfant droit de y ssi x est une feuille qui se trouve à droite de tout nœud fils de y qui n'est pas une feuille.

Définition: (x, y) est une paire indexante de T_1 dans l'un des cas suivants:

- x est un nœud interne et $x = y$;
- x est un nœud interne et y son enfant droit;
- y est un nœud interne et x son enfant gauche;
- x est une feuille gauche et y une feuille droite d'un parent commun;
- x et y sont des feuilles correspondant à la boucle terminale de la tige-boucle avec x à gauche de y.

Soit P_1 la liste ordonnée de toutes les paires indexantes p_i de T_1 telle que:

(1) $p_0 = (--,--)$;

(2) $\forall (p_i = (x, y), p_j = (u, v)),\ x = u \text{ et } y = v \Rightarrow i = j$;

(3) $\forall p_i = (x, y) | i > 0,\ \exists p_j = (p(x), s(y)) | i < j$;

(4) si x est une feuille de T_1, $\forall p_i = (x, y) | i > 0,\ \exists p_j = (p(x), y) | i < j$;

(5) si y est une feuille de T_1, $\forall p_i = (x, y) | i > 0,\ \exists p_j = (x, s(y)) | i < j$.

Remarque: l'ordre des paires indexantes imposé ici est l'inverse de celui présenté par [16] (à l'exception de la première paire) et permet un alignement depuis la boucle terminale d'une tige-boucle.

Soit P_2 la liste ordonnée de toutes les paires indexantes p_i de T_2 définie de façon analogue à P_1.

Soit D une table bidimensionnelle indexée par les paires indexantes de P_1 en abscisse et celles de P_2 en ordonnées.

Soient a, b, c et d des bases telles que $a \neq b$, $c \neq d$, a et b soient orientées 5'-3' et c et d soient orientées 3'-5'.

Soient les opérations d'éditions suivantes et leurs coûts associés:

- correspondance exacte d'une base, $\gamma(a \rightarrow a)$ ou $\gamma(c \rightarrow c)$;
- réétiquetage d'une base, $\gamma(a \rightarrow b)$ ou $\gamma(c \rightarrow d)$;
- suppression d'une base, $\gamma(a \rightarrow --)$ ou $\gamma(c \rightarrow --)$;
- insertion d'une base, $\gamma(-- \rightarrow a)$ ou $\gamma(-- \rightarrow c)$;
- correspondance exacte d'une paire de bases, $\gamma(ac \rightarrow ac)$;
- réétiquetage d'une bases appariée, $\gamma(ac \rightarrow bc)$ ou $\gamma(ac \rightarrow ad)$;
- réétiquetage d'une paire de bases, $\gamma(ac \rightarrow bd)$;
- suppression d'une paire de bases, $\gamma(ac \rightarrow --)$;
- insertion d'une paire de bases, $\gamma(-- \rightarrow ac)$;
- altération d'une paire de bases, $\gamma(ac \rightarrow a)$ ou $\gamma(ac \rightarrow c)$;
- complémentation d'une paire de bases, $\gamma(a \rightarrow ac)$ ou $\gamma(c \rightarrow ac)$;
- bris d'une paire de bases, $\gamma(ac \rightarrow (a,c))$;
- formation d'une paire de base, $\gamma((a,c) \rightarrow ac)$;
- réappariement d'une base, $\gamma((ac,d) \rightarrow (ad,c))$ ou $\gamma((ac,b) \rightarrow (bc,a))$.

Nous imposons les contraintes supplémentaires suivantes sur le schéma de score de manière à ce que γ calcule toujours une distance:

$$\gamma(ac \rightarrow a) = \gamma(ac \rightarrow (a,c)) + \gamma(c \rightarrow --),$$

$$\gamma(ac \to c) = \gamma(ac \to (a,c)) + \gamma(a \to --),$$
$$\gamma(a \to ac) = \gamma(-- \to c) + \gamma((a,c) \to ac),$$
$$\gamma(c \to ac) = \gamma(-- \to a) + \gamma((a,c) \to ac),$$
$$\gamma((ac,d) \to (ad,c)) = \gamma(ac \to (a,c)) + \gamma((a,d) \to ad),$$
$$\gamma((ac,b) \to (bc,a)) = \gamma(ac \to (a,c)) + \gamma((b,c) \to bc).$$

Soit δ une fonction de coûts prenant les mêmes types de paramètres que γ telle que, si ac et bd sont deux paires de bases, e est une base libre orientée 5'-3', f est une base libre orientée 3'-5' et x est indépendamment une base libre ou une paire de bases, alors:

$$\delta(a \to x) = \delta(x \to a) = \frac{\gamma(ac \to (a,c))}{2} + \gamma(a \to x);$$
$$\delta(c \to x) = \delta(x \to c) = \frac{\gamma(ac \to (a,c))}{2} + \gamma(c \to x);$$
$$\delta(a \to --) = \delta(-- \to a) = \frac{\gamma(ac \to (a,c))}{2} + \gamma(a \to --);$$
$$\delta(c \to --) = \delta(-- \to c) = \frac{\gamma(ac \to (a,c))}{2} + \gamma(c \to --);$$
$$\delta(ac \to x) = \delta(x \to ac) = \gamma(ac \to x);$$
$$\delta(ac \to --) = \delta(-- \to ac) = \gamma(ac \to --);$$
$$\delta((a,c) \to ac) = 0;$$
$$\delta(ac \to (a,c)) = \gamma(ac \to (a,c));$$
$$\delta((a,f) \to af) = \frac{\gamma(ac \to (a,c))}{2} + \gamma((a,f) \to af);$$
$$\delta(af \to (a,f)) = \frac{\gamma(ac \to (a,c))}{2};$$
$$\delta((e,c) \to ec) = \frac{\gamma(ac \to (a,c))}{2} + \gamma((e,c) \to ec);$$
$$\delta(ec \to (e,c)) = \frac{\gamma(ac \to (a,c))}{2};$$

$$\delta((a,d)\rightarrow ad)=\frac{\gamma(ac\rightarrow(a,c))}{2}+\frac{\gamma(bd\rightarrow(b,d))}{2}+\gamma((a,d)\rightarrow ad);$$

$$\delta(ad\rightarrow(a,d))=\frac{\gamma(ac\rightarrow(a,c))}{2}+\frac{\gamma(bd\rightarrow(b,d))}{2}.$$

Lemme 1: (1) $D[0,0]=0$;

(2) $\forall i, D[\phi,p_i]=D[p_i,\phi]=D[\phi,\phi]=\infty$ où ϕ est un index de D indéfini et p_i une paire indexante;

(3) $$D[(x,y),0]=\min\begin{pmatrix}D[(p(x),y),0]+\gamma(x\rightarrow--),\\ D[(x,s(y)),0]+\gamma(y\rightarrow--)\end{pmatrix};$$

(4) $$D[0,(x,y)]=\min\begin{pmatrix}D[0,(p(x),y)]+\gamma(x\rightarrow--),\\ D[0,(x,s(y))]+\gamma(y\rightarrow--)\end{pmatrix}.$$

Preuve 1: trivial.□

Lemme 2:

$$D[(x,y),(u,v)]=\min\begin{pmatrix}D[(p(x),y),(p(u),v)]+\delta(x\rightarrow u),\\ D[(x,s(y)),(u,s(v))]+\delta(y\rightarrow v),\\ D[(p(x),y),(u,v)]+\delta(x\rightarrow--),\\ D[(x,p(y)),(u,v)]+\delta(y\rightarrow--),\\ D[(x,y),(p(u),v)]+\delta(--\rightarrow u),\\ D[(x,y),(u,s(v))]+\delta(--\rightarrow v),\\ D[(p(x),s(y)),(p(u),s(v))]+\delta((x,y)\rightarrow xy)+\delta((u,v)\rightarrow uv)+\delta(xy\rightarrow uv),\\ D[(p(x),s(y)),(p(u),v)]+\delta((x,y)\rightarrow xy)+\delta(xy\rightarrow u),\\ D[(p(x),s(y)),(u,s(v))]+\delta((x,y)\rightarrow xy)+\delta(xy\rightarrow v),\\ D[(p(x),y),(p(u),s(v))]+\delta((u,v)\rightarrow uv)+\delta(x\rightarrow uv),\\ D[(x,s(y)),(p(u),s(v))]+\delta((u,v)\rightarrow uv)+\delta(y\rightarrow uv)\end{pmatrix}$$

Preuve 2: nous cherchons à calculer la distance entre la sous-structure S'_1 comprenant les bases entre x et y et la sous-structure S'_2 comprenant les bases entre les bases u et v en connaissant les distances suivantes:

(S'_1 privée de x, S'_2 privée de u);

(S'_1 privée de y, S'_2 privée de v);

(S'_1 privée de x, S'_2);

(S'_1 privée de y, S'_2);

(S'_1, S'_2 privée de u);

(S'_1, S'_2 privée de v);

(S'_1 privée de x et de y, S'_2 privée de u et de v);

(S'_1 privée de x et de y, S'_2 privée de u);

(S'_1 privée de x et de y, S'_2 privée de v);

(S'_1 privée de x, S'_2 privée de u et de v);

(S'_1 privée de y, S'_2 privée de u et de v).

La distance entre S'_1 et S'_2 se calcule à partir des distances connues entre les structures inférieures ou égales à S'_1 et S'_2 auxquels le ou les éléments manquants sont ajoutés à l'aide d'une opération d'édition. Cette distance correspond donc à la distance minimale entre tous les cas possibles suivant correspondants aux 11 cas du lemme 2:

(1) distance (S'_1 privée de x, S'_2 privée de u) + coût de la substitution de x en u;

(2) distance (S'_1 privée de y, S'_2 privée de v) + coût de la correspondance entre y et v;

(3) distance (S'_1 privée de x, S'_2) + coût de la suppression de x;

(4) distance (S'_1 privée de y, S'_2) + coût de la suppression de y;

(5) distance (S'_1, S'_2 privée de u) + coût de la suppression de u;

(6) distance (S'_1, S'_2 privée de v) + coût de la suppression de v;

(7) distance (S'_1 privée de x et de y, S'_2 privée de u et de v) + coût de la correspondance entre xy et uv;

(8) distance (S'_1 privée de x et de y, S'_2 privée de u) + coût de l'altération de xy en u;

(9) distance (S'_1 privée de x et de y, S'_2 privée de v) + coût de l'altération de xy en v;

(10) distance (S'_1 privée de x , S'_2 privée de u et de v) + coût de la complémentation de x en uv;

(11) distance (S'_1 privée de y , S'_2 privée de u et de v) + coût de la complémentation de y en uv.

Les formation et bris de paires ainsi que les réappariements sont pris en compte et calculés par la fonction de coût δ. Ainsi, par construction, elle nous assure que toute formation de paire entre deux bases déjà appariées ensemble sera nul et que toute opération portant sur une seule des deux bases d'une paire sera affecté de la moîtié du coût du bris de cette paire, l'autre moîtié de ce coût étant ajouté lors d'une opération necessairement effectuée sur l'autre base de la paire. Le résultat de chacun des 11 cas est donc une somme de deux distances et le minimum de ces 11 cas correspond bien à la distance entre S'_1 et S'_2. □

Lemme 3: la distance entre S_1 et S_2 est égale à $D[(x,y),(u,v)]$, avec x et y respectivement les première et dernière bases de S_1 et u et v respectivement les première et dernière bases de S_2.

Preuve 3: découle directement de la preuve 2.

L'algorithme de calcul de distance entre tiges-boucles est donc exact pour un schéma de score comportant toutes les opérations d'édition portant sur des bases ou des paires de bases ou les deux. La complexité en temps et en espace de cet algorithme dépend des dimensions de la table D. Cette table a pour dimensions les longueurs prises par les listes des paires indexantes P_1 et P_2. Chaque liste a au plus n^2 paires indexantes, où n est le nombre de base de la structure correspondante. En effet, chaque base d'une structure peut former une paire avec une autre base de la même strucutre, soit n bases avec n autres bases ou encore n^2. Il en découle que la complexité en temps et en espace de l'algorithme est en $O(n^4)$. □

4.2.2.2 Estimation d'une distance globale

Soient 2 structures secondaires d'ARN R_1 et R_2 décomposées en arbres de tiges-boucles ATB_1 et respectivement ATB_2 comme indiqué Figure 44.

Soit *DTB* une table bidimensionnelle contenant les distances entre les tiges-boucles de ATB_1 et ATB_2, indexée en abscisse par tous les nœuds de ATB_1, et en ordonnées par tous les nœuds de ATB_2. La table *DTB* est remplie en prenant chaque tige-boucle de ATB_1 et calculant sa distance avec chaque tige-boucle de ATB_2.

Soit *DMTB* une table bidimensionnelle contenant des estimations de distances entre des ensembles de tiges-boucles adjacentes provenant de ATB_1 et ATB_2. Lorsqu'une tige-boucle tb_1 d'une structure s'aligne complètement avec un préfixe tb_2' d'une tige-boucle tb_2 de l'autre structure, c'est-à-dire qu'il existe une valeur minimale dans la dernière ligne ou (exclusif) la dernière colonne de la table de distance D entre les deux tiges-boucles, et que cette valeur est inférieure ou égale à la distance globale entre les deux tiges-boucles, le suffixe tb_2'' non aligné est extrait. On ajoute le coût de suppression des sœurs de tb_1 à la distance entre tb_1 et tb_2' et on réitère l'opération avec le parent de tb_1 et tb_2''. Le processus s'arrête lorsqu'il n'existe plus d'alignement d'une tige-boucle avec un préfixe de l'autre ou qu'il n'y a plus de tige-boucle parente sur l'une des structure. La distance ainsi estimée est sauvegardée dans la table *DMTB* dans la cellule indexée par les dernières tiges-boucles parentes alignées de ATB_1 et ATB_2 si et seulement si la cellule ne contient pas de valeur ou que la valeur déjà présente est supérieur à la nouvelle valeur calculée. Ainsi, *DMTB* ne contient que certaines estimations de distance entre des ensembles de tiges-boucles. La complexité en temps apportée ces calculs est de $O(m.n^4)$ pour m tiges-boucles comportant chacune au maximum n bases.

L'algorithme de calcul de distance globale entre les structures R_1 et R_2 est basé sur l'algorithme de Zhang-Shasha [42] appliqué aux arbres ATB_1 et ATB_2,

la distance entre chaque nœud étant fournie par la table *DTB*. Cet algorithme est cependant modifié pour mettre dans la table permanente le minimum de la distance entre deux nœuds calculée normalement et la distance estimée entre ces deux nœuds provenant de la table *DMTB* si cette dernière valeur existe. Ceci s'effectue en ce reportant à l'article [42] et en ajoutant après la ligne de l'algorithme *treedist(i,j)*.

```
treedist(i1,j1)=forestdist(T1[l(i)..i],T2[l(j)..j]) /* put in permanent array */
```

les deux lignes suivantes:

```
if DMTB[i1,j1] exists and DMTB[i1,j1] < treedist(i1,j1) then
      treedist(i1,j1) = DMTB[i1,j1]
```

Ainsi, notre algorithme global se décompose en 3 phases. La première phase découpe les structures en arbre de tiges-boucles. La seconde phase calcule les distances entre les tiges-boucles. Enfin, la troisième phase calcule la distance entre les arbres de tiges-boucles en intégrant les fusions possibles détectées lors de la seconde phase. Notre algorithme estime donc une distance entre les deux structures en prenant en compte les *opérations d'édition* simples et complexes et les macro-opérations tout en conservant une complexité polynomiale en $O(n^4)$ pour les calculs entre tiges-boucles et $O(n^5)$ pour la prise en compte des macro-opérations. Remarquons que le problème de calcul de distance entre deux structures est *NP-complet* dans un cas général ([22]) mais que nous calculons ici une estimation de distance et non une distance exacte.

Nous conclurons ce sous-chapitre en discutant de quelques voies qui peuvent être explorées pour formuler des heuristiques encore plus rapide à partir de notre algorithme. Tout d'abord, il n'est souvent pas nécessaire de comparer toutes les tiges entre elles, ce qui suggère qu'il est possible d'économiser un temps précieux en calculs en ne comparant que les tiges-boucles susceptibles de s'aligner (i.e: il est généralement inutile de comparer la première tige-boucle d'une structure avec la dernière d'une autre). Ensuite, pour pouvoir rester dans un nombre de *cycles de calculs* proche ou inférieur à celui de Zhang-Shasha lors du calcul de distance entre tiges-boucles, nous pouvons faire

un compromis au niveau de la relaxation des paires. En effet, d'un point de vue biologique, les changements d'appariements ou les phénomènes de "reptation" de la structure entraînant des appariements entre deux bases libres n'appartenant pas à une même boucle, s'effectuent sur des bases proches; une base libre du début d'une tige boucle ne s'appariera vraisemblablement pas avec une base de la boucle terminal, il n'est donc pas nécessaire de calculer cette possibilité. L'idée est de ne relaxer qu'une des structures et ce localement aux endroits stratégiques. Nous pouvons donc fournir à notre algorithme un paramètre de degré de liberté lui disant de relaxer un certain nombre de paires voisines d'une boucle ou d'un renflement. Si ce paramètre est à 0, les contraintes "gauche-droite" et de "localité" s'imposent et le changement d'appariement ne peut plus être calculé mais l'algorithme est très rapide ; si au contraire ce paramètre est infini, tous les appariements possibles sont calculés sans contraintes mais l'algorithme est plus lent. Ensuite, les différents paramètres pour relaxer les contraintes gauche-droite et de localité pourraient être ajustés de façon automatique en fonction des caractéristiques structures. Enfin, nous n'avons pas non plus exploré la possibilité de bit-vectoriser notre algorithme.

4.3 Bibliothèque de fonctions C++

Pour implémenter et tester notre algorithme, nous avons choisi de développer une petite bibliothèque de fonctions en C++. Le choix d'utiliser C++ plutôt qu'un autre langage est avant tout personnel. Les arguments qui ont motivé notre choix sont:

- une très bonne expérience de ce langage;
- la rapidité d'exécution programmes compilés face aux langages interprétés;
- et enfin la possibilité à long terme d'écrire certaines parties du code en langage assembleur (en particulier dans l'optique d'une bit-vectorisation).

Nous avons ensuite vérifié si des outils C++ avaient déjà été développés. Tout d'abord, il faut savoir que s'il existe bien des librairies BioJava, BioPerl,

BioPython et BioRuby, il n'y a cependant pas de librairie BioC++ rattaché à la Fondation Open Bioinformatics [50]. En revanche, plusieurs tentatives ont été faites pour développer des bibliothèques bio-informatiques en C++.

Il existe un projet source-forge de bibliothèque C++ open-source initialement développé par le Dr. Hongyu Zhang [47]. Les fonctions offertes concernent principalement les protéines et sont relativement mal documentées. De plus, ce projet est critiquable sur plusieurs points: entêtes et implémentations se retrouvent dans un seul fichier d'entête alors que le standard voudrait qu'elles soient séparées; ensuite, le code ne respecte pas le standard ISO/ANSI C/C++.

Le Vienna RNA Package [51] est une autre bibliothèque de fonctions destinées quant à elles à prédire et comparer les structures secondaires d'ARN. Bien que plus fourni en fonctions et récemment mis à jour, le Vienna RNA Package reste néanmoins mal documenté. L'utilisateur désireux de s'en servir constatera que les programmes sont documentés mais ce n'est pas le cas des fonctions. De plus, le code source ne respecte pas non plus le standard ISO/ANSI C/C++ et de nombreuses difficultés apparaissent lorsqu'il s'agit de compiler les programmes sur d'autres plateformes que Unix. Nous avons quand même retenu un programme bien utile de ce package: RNAPlot, un programme destiné à l'affichage de structures secondaires.

Une autre bibliothèque C++ dénommée seq++ [32] offre pour sa part des fonctions destinées à des analyses statistiques des séquences. Bien que respectant le standard ISO/ANSI C/C++ et étant bien documentée, cette bibliothèque ne comportait pas de fonctions intéressantes pour nos travaux.

Nous avons donc commencé le développement d'une bibliothèque C++ bien documentée à l'aide de Doxygen [49] et respectant les standard ISO/ANSI C/C++. Actuellement, elle comporte divers outils pour traiter les séquences et structures nucléotidiques ainsi que divers convertisseurs de format de fichiers. Elle est encore loin d'être achevée mais nous envisageons à court terme de la rendre publique et si possible de l'intégrer à la Fondation Open Bioinformatics

pour que toute la communauté bioinformatique utilisant le C++ puisse en profiter et l'étendre.

4.4 Base de données

Une fois le programme de comparaison de structures secondaires implémenté, nous avions besoin d'une base de données pour le tester. Il existe plusieurs bases de données de structures secondaires à accès publique par Internet. Malheureusement, toutes ces bases ne partage pas un même et unique format de données. De plus, pour effectuer des comparaisons massives, il fallait que ces structures soient accessible localement par notre programme pour éviter les pertes de temps liées aux connexions et aux échanges de données avec ces serveurs. C'est pourquoi nous avons projeté de mettre au point notre propre base de données de structures secondaires en regroupant dans un format unique les données disséminées en plusieurs formats sur plusieurs serveurs.

Notre base de données est de type mySQL. Ce choix a été motivé par le fait que mySQL est un logiciel gratuit, très performant et disponible sur des serveurs du Département d'Informatique et de Recherche Opérationnelle de l'Université de Montréal. Le schéma de nos tables de données a été étudié pour faciliter les recherches et les sélections de structures secondaires en fonction de différents critères comme les familles d'ARN, les espèces, les sources des structures ou encore leurs longueurs. Pour plus de détails concernant l'architecture de la base de données, se référer à l'annexe 5.2.

Présentement, nous avons développé des outils pour importer les données de la RFam ([9]) et les compléter automatiquement. En effet, les données de la RFam ne contiennent pas toutes les informations sur la taxonomie de ses entrées et une seule structure consensus est fournie par famille d'ARN. Notre programme d'importation doit donc adapter la structure consensus à chaque séquence en retirant les paires non valides et en reformant des paires évidentes (Figure 45). Nous noterons que les structures fournies par la RFam sont des

structures consensus obtenues à l'aide du logiciel INFERNAL utilisant un model de covariance. Par conséquent, les structures de la RFam ou celles ajustées par notre programme ne sont pas nécessairement fiable et induisent un biais dont il faut être conscient.

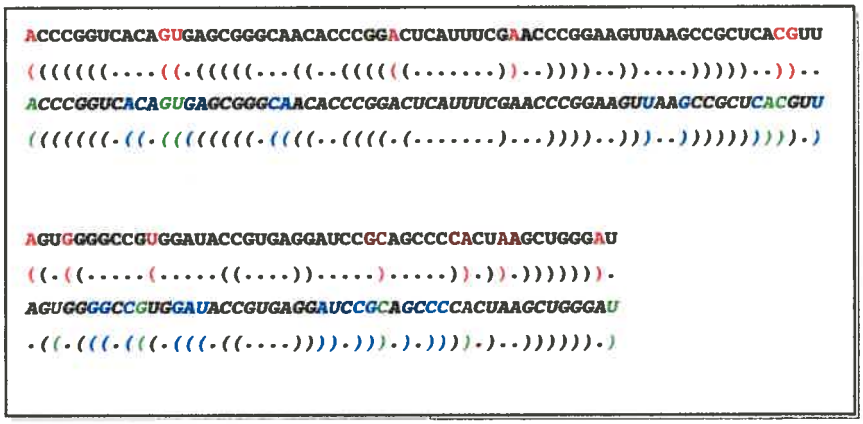


Figure 45 Correction de structure secondaire.
Les deux premières lignes correspondent à la structure extraite de la RFam, les deux lignes suivantes (en italique) à la structure corrigée par notre programme d'importation. En rouge, les paires non valides, en vert les paires corrigées et en bleu les paires ajoutées.

Pour compléter nos données, nous projetons d'importer des structures secondaires plus fiables que celles de la RFam. Nous disposons par exemple d'une base de données de structures secondaires d'ARNt issue de [38], d'ARNtm provenant de [46], d'ARNr fournies par [9] et [31].

4.5 Serveur de comparaison

Le dernier outil de ce projet de maîtrise est le serveur de comparaison de structures secondaires dénommé RNAStrAT pour "RNA Structure Analysis Toolkit". Ce serveur est publiquement accessible par Internet à l'adresse http://www-lbit.iro.umontreal.ca/rnastrat (Figure 46). Il est composé d'une interface graphique réalisée en scripts php, de la base de données mySQL et de programmes pour comparer et afficher des structures secondaires d'ARN. Le site est bilingue français-anglais, est compatible avec les navigateurs les plus

courants sur Mac, PC et Unix et son contenu est facilement imprimable par les utilisateurs grâce à une feuille de style prévue à cet effet.

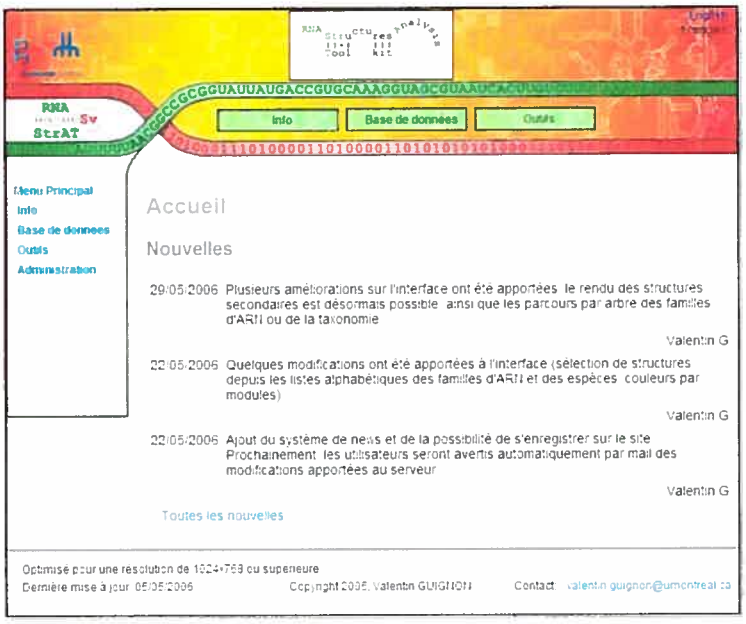


Figure 46 Page d'accueil du serveur RNAStrAT.

L'interface est décomposée en 4 parties. La première, intitulée "Info", regroupe les détails techniques et les articles relatifs aux différents outils du site ainsi que des liens vers d'autres sites appropriés.

La seconde partie renferme l'interface graphique permettant de parcourir la base de données. Ainsi, les utilisateurs peuvent rechercher des structures secondaires suivant certain critères, ils peuvent également les lister par leurs identifiants, leurs espèces ou leurs familles d'ARN, ou encore parcourir les espèces ou les familles d'ARN sous forme d'arbre. Les structures peuvent être affichées d'un simple clique ou être sélectionnées pour une comparaison avec une autre structure. La base de donnée fournit ainsi un large ensemble d'exemples pour les tests et permettra d'offrir dans un futur plus ou moins proche un service de type BLAST où l'utilisateur entrera une structure à rechercher au sein de la base de données.

La partie suivante est celle des outils pour analyser les structures. Les utilisateurs ont la possibilité de comparer des structures avec l'algorithme de

leur choix. Ils peuvent également convertir les formats de leurs données ou encore afficher des structures suivant différents niveaux de détails.

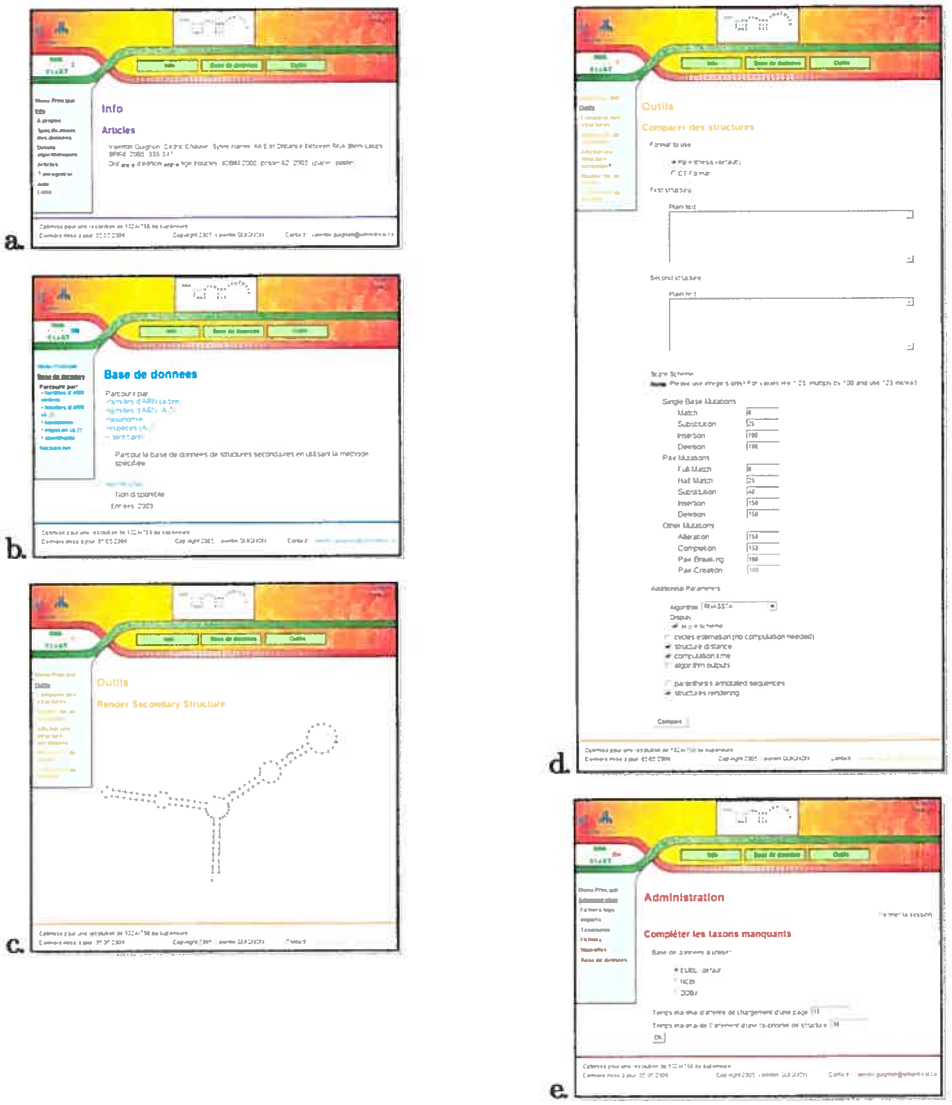


Figure 47 Interface du serveur d'outils d'analyse de structures secondaires d'ARN RNA StrAT.

a. Page de la section "information" (ici la page des articles);

b. page principale de la section "base de données";

c. et **d.** pages de la section "outils" (rendu et comparaison);

e. page de la section "administration".

Enfin, la dernière partie est réservée à l'administration du site. Son interface permet d'importer des structures, de compléter les données manquantes, de gérer les nouvelles du site ou encore de consulter les fichiers de log enregistrant les actions effectuées sur le serveur.

Chapitre 5

CONCLUSION

Dans ce mémoire, nous avons tout d'abord vu l'intérêt profond des chercheurs pour les molécules d'ARN. Il nous semble évident que cet intérêt ne pourra que croître dans les prochaines années aux vues de certaines découvertes récentes comme la transmission d'information génétique par de l'ARN ([8] et [34]) ou encore l'interférence par ARN ([25]) et peut-être même un jour la correction de gènes in-vivo grâce à des techniques de "rétro-homing" ([7]) faisant appel aux *introns*.

Nous avons donc apporté notre contribution aux recherches sur l'ARN en offrant à la communauté scientifique plusieurs outils d'analyse accessibles gratuitement depuis notre serveur. Les utilisateurs ont ainsi accès via une interface de qualité à une vaste base de données de structures secondaires sur laquelle ils peuvent effectuer des recherches et des comparaisons rapides.

Les buts initiaux de ce projet de maîtrise ont donc été remplis avec succès. L'algorithme de comparaison que nous avons mis au point s'avère moins gourmant en calculs que celui de Zhang-Shasha qui nous a servi de référence ([42]). Il est de surcroît plus précis car il est capable de prendre en compte toutes les *opérations d'édition* sur les structures secondaires. Enfin, il ouvre plusieurs perspectives de recherches.

Une première avenue intéressant serait de cherche à bit-vectoriser notre algorithme. Le fait que tous les calculs s'effectuent sur une seule table suggère que la bit-vectorisation serait plus simple qu'avec l'algorithme de Zhang-Shasha. Une bit-vectorisation permettrait de diminuer de façon significative la complexité en temps de notre algorithme et serait donc très utile dans la comparaison à grande échelle de structures.

Une deuxième avenue de recherche se situe au niveau des schémas de score utilisés par notre algorithme. En effet, comme mentionné à la section 3.5, le problème est loin d'être trivial et n'a pas encore fait parti d'une étude appronfondie. L'élaboration d'un algorithme polynomial prenant en compte toutes les *opérations d'édition* sur les structures secondaires rend ce problème plus admissible. La mise au point de schémas de score adaptés pourrait conduire à l'obtention d'alignements structurels plus fiables nécessaires à une meilleure qualité des études scientifiques portant sur les ARN.

Enfin, la comparaison de structures secondaires couplée à une base de données donne accès à de nombreuses applications décrites dans la section 3.1, ne serait-ce que l'aide à la prédiction ou l'annotation automatique de structures secondaires (voir section 3.1 pour plus de détails).

RÉFÉRENCES

[1] Allali, J., Sagot, M-F., *Novel Tree Edit Operations for RNA Secondary Structure Comparison*, WABI 2004, 2004, 412-425.

[2] Altschul, SF., Gish, W., Miller, W., Myers, EW., Lipman, DJ., *Basic local alignment search tool*, J. Mol. Biol., 1990, 215: 403-410.

[3] Bartel, D., *MicroRNAsGenomics, Biogenesis, Mechanism, and Function.*, Cell, 23 Jan 2004, Volume 116, Issue 2, Pages 281-297.

[4] Bergeron, A., Hamel, S., *Vector Algorithms for Approximate String Matching*, International Journal of Foundations of Computer Sc., 2002, 13-1, 53—66.

[5] Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E., *The Protein Data Bank.*, Nucleic Acids Research, 2000, 28 pp. 235-242.

[6] Bille, P., *A survey on tree edit distance and related problems*, Theor. Comput. Sci., Jun 2005, 337, 1-3, 217-239.

[7] Bonen, L., Vogel, J., *The ins and outs of group II introns*, Trends in Genetics, 2001, 17:322-31

[8] Buck, A.H., Dalby, A.B., Poole, A.W., Kazantsev, A.V., Pace, N.R., *Protein activation of a ribozyme: the role of bacterial RNase P protein*, EMBO J., 5 Oct 2005, 24(19):3360-8.

[9] Cannone, J.J., Subramanian, S., Schnare, M.N., Collett, J.R., D'Souza, L.M., Du, Y., Feng, B., Lin, N., Madabusi, L.V., Muller, K.M., Pande, N., Shang, Z., Yu, N., Gutell, R.R., *The comparative RNA web (CRW) site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs*, BMC Bioinformatics, Jul 2002, 3:2, Epub 2002 Jan 17, Erratum in BMC Bioinformatics, 3(1):15.

[10] Chen, J.L., Greiger, C.W., *Functional analysis of the pseudoknot structure in human telomerase RNA*, 2005, Proc Natl. Acad. Sci. USA 102(23):8080-5.

[11] Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C., *A model of evolutionary change in proteins*, Atlas of protein sequence and structure, 1978, supplement 3, National Biomedical Research Foundation, Washington, DC, pp. 345-352.

[12] Dulucq, S., Tichit, L., *RNA secondary structure comparison: exact analysis of the zhang-shasha tree edit algorithm*, Theoretical Computer Science, 2003, pp. 471-V484.

[13] Gilbert, W., *The RNA world*, Nature, 1986, 319, 618.

[14] Griffiths-Jones, S., Bateman, A., Marshall, M., Khanna, A., Eddy, S.R., *RFam: an RNA family database*, Nucleic Acids Research, 2003, 31, 1, 439-441.

[15] Guignon, V., Chauve, C., Hamel, S., *Distance d'édition entre tiges-boucles*, JOBIM 2005, 2005, poster 82.

[16] Guignon, V., Chauve, C., Hamel, S., *An edit distance between RNA stem-loops*, SPIRE 2005, 2005, LNCS 3772:334-345.

[17] Guo, H., Karberg, M., Long, M., Jones, JP. 3rd, Sullenger, B., Lambowitz, A.M., *Group II introns designed to insert into therapeutically relevant DNA target sites in human cells*, Science, 21 Jul 2000, 289(5478):452-7.

[18] Hochsmann, M., Toller, T., Giegerich, R., Kurtz, S., *Local Similarity in RNASecondary Structures*, In Proc of the Computational Systems Bioinformatics (CSB) Conference, Aug 2003, Stanford, CA.

[19] Hofacker, I.L., Fontana, W., Stadler, P.F., Bonhoeffer, S., Tacker, M., Schuster, P., *Fast Folding and Comparison of RNA Secondary Structures*, Monatshefte f. Chemie, 1994, 125:167-188.

[20] IUBMB. Biochem. J., 1985, 229, 281-286; Eur. J. Biochem., 1985, 150, 1-5; J. Biol. Chem., 1986, 261, 13-17; Mol. Biol. Evol., 1986, 3, 99-108; Nucl. Acids Res., 1985, 13, 3021-3030; Proc. Nat. Acad. Sci. (U. S.), 1986, 83, 4-8; Biochemical Nomenclature and Related Documents, 2nd edition, Portland Press, 1992, pp 122-126.

[21] Jowicz, J., *Structural Properties of Shuffle Automata*, Grammars, Vol. 2, Number 1, 1999, pp. 35-51(17).

[22] Jiang, T., Lin, G., Ma, B., Zhang, K., *A general edit distance between RNA structures*, J. Comput. Biol., 2002, 9(2), 371–388.

[23] Jiang, T., Wang, L., Zhang, Z., *Alignment of Trees–An Alternative to Tree Edit*, Proc. Fifth Ann. Symp. Combinatorial Pattern Matching, 1994, pp. 75-86.

[24] Klein, P.N., *Computing the Edit-Distance between Unrooted Ordered Trees*, LNCS 1461, 1998, 91–102.

[25] Krol, J., Sobczak, K., Wilczynska, U., Drath, M., Jasinska, A., Kaczynska, D., Krzyzosiak, W.J., *Structural features of microRNA (miRNA) precursors and their relevance to miRNA biogenesis and small interfering RNA/short hairpin RNA design*, J Biol Chem., 1 Oct 2004, 279(40):42230-9.

[26] Tai, K., *The tree-to-tree correction problem*, J. Assoc. Comput., 26 Mar 1979, pp. 422-433.

[27] Leontis, N.B., Westhof, E., *Geometric nomenclature and classification of RNA base pairs*, RNA, Apr 2001, 7(4):499-512.

[28] Liu, J., Wang, J.T.L., Hu, J., Tian, B., *A method for aligning RNA secondary structures and its application to RNA motif detection*, BMC Bioinformatics 2005, 7 Apr 2005, 6:89.

[29] Lolle, S.J., Victor, J.L., Young, J.M., Pruitt, R.E., *Genome-wide non-mendelian inheritance of extra-genomic information in Arabidopsis*, Nature, 24 Mar 2005, 434(7032):505-9.

[30] Lyngsø, R.B., Pedersen, C.N., *RNA pseudoknot prediction in energy-based models*, J Comput Biol, 2000, 7(3-4): 409-427.

[31] Szymanski, M., Barciszewska, M.Z., Erdmann, V.A., Barciszewski, J., *5S ribosomal RNA database*, Nucleic Acids Res., 2002, 30: 176-178.

[32] Miele, V., Bourguignon, P.Y., Robelin, D., Nuel, G., Richard, H., *seq++: analyzing biological sequences with a range of Markov-related models*, Bioinformatics, 1 Jun 2005, 21(11):2783-4.

[33] Mueller, F., Sommer, I., Baranov, P., Matadeen, R., Stoldt, M., Wohnert, J., Gorlach, M., van Heel, M., Brimacombe, R., *The 3D arrangement of the 23 S and 5 S rRNA in the Escherichia coli 50 S ribosomal subunit based on a cryo-electron microscopic reconstruction at 7.5 Å resolution*, J.Mol.Biol., 2000, v298, pp.35-59.

[34] Rassoulzadegan, M., Grandjean, V., Gounon, P., Vincent, S., Gillot, I., Cuzin, F., *RNA-mediated non-mendelian inheritance of an epigenetic change in the mouse*, Nature, 2006, Vol. 441, No. 7092., pp. 469-474.

[35] Ray, B.K., Apirion, D., *Characterization of 10S RNA: a new stable RNA molecule from Escherichia coli*, Mol Gen Genet, 1979, 174:25-32.

[36] Backofen, R., Will, S., *Local sequence-structure motifs in RNA*, Journal of Bioinformatics and Computational Biology (JBCB), 2004, 2 no. 4 pp. 681-698.

[37] Schuwirth, B.S., Borovinskaya, M.A., Hau, C.W., Zhang, W., Vila-Sanjurjo, A., Holton, J.M., Cate, J.H.D., *Structures of the bacterial ribosome at 3.5 A resolution*, Science, 2005, v310, pp.827-834.

[38] Sprinzl, M., Horn, C., Brown, M., Ioudovitch, A., Steinberg, S., *Compilation of tRNA sequences and sequences of tRNA genes*, Nucleic Acids Res., 1 Jan 1998, 1;26(1):148-53.

[39] Voet, D., Voet, J. G., *Biochimie (2e édition français)*, De Boeck, 2005.

[40] Woese, C.R., *The Genetic Code: The Molecular Basis for Genetic Expression*, Harper and Row, New York., 1967.

[41] Zagryadskaya, E.I., Doyon, F.R., Steinberg, S.V., *Importance of the reverse Hoogsteen base pair 54—58 for tRNA function*, Nucleic Acids Res., 2003, 31, , 3946—3953.

[42] Zhang, K., Shasha, D., *Simple fast algorithms for the editing distance between trees and related problems*, SIAM J. Comput., 1989, 18-6, 1245–1262.

[43] Zhang, K., Wang, L., Ma, B., *Computing similarity between RNA structures*, Theoretical Computer Sciences, mars 2000, 276(1-2):111-132, 2002.

[44] Zuker, M., *Mfold web server for nucleic acid folding and hybridization prediction*, Nucleic Acids Res., 2003, 31 (13), 3406-15.

[45] Mathews, D.H., Sabina, J., Zuker M., Turner, D.H., *Expanded Sequence Dependence of Thermodynamic Parameters Improves Prediction of RNA Secondary Structure*, J. Mol. Biol, 1999, 288, 911-940.

[46] Zwieb, C., Gorodkin, J., Knudsen B., Burks, J., Wower, J., *tmRDB (tmRNA database)*, Nucleic Acids Res, 2003, 31, 446-447.

[47] BioC++, http://biocpp.sourceforge.net

[48] DicoMaths, http://www.bibmath.net/dico

[49] Doxygen, http://www.stack.nl/~dimitri/doxygen

[50] Open Bioinformatics Foundation, http://www.open-bio.org

[51] Vienna RNA Package, http://www.tbi.univie.ac.at/~ivo/RNA

ANNEXES

5.1 Glossaire

Algorithme bit-vectoriel: un algorithme vectoriel est un algorithme qui permet d'obtenir un vecteur de sortie en n'appliquant, sur un vecteur d'entrée, que des opérations vectorielles. Cet algorithme peut alors être implémenté en parallèle, en se servant des opérations sur les vecteurs de bits disponibles dans les processeurs, donnant lieu à des calculs très efficaces. (source: thèse de doctorat de Hamel S.)

Algorithme en temps polynomial: un algorithme est dit en temps polynômial si, pour tout n, pour des données ne prenant pas plus de n octets, l'algorithme s'exécute en moins de $C.n^k$ opérations élémentaires (les constantes C et k étant bien sûr indépendantes de n) (source: [48]).

Appariement de bases: présence de ponts hydrogène entre des bases créant un lien non-covalent entre elles.

ARN : l'acide ribononucléique ou ARN est un polymère de longueur variable de ribonucléotides (bases).

Automate des mélange: un automate des mélange A est un quintuplet (Q, Σ, q_0, q_f, T), où Q est un ensemble fini d'états, $Q = ORD \cup OP \cup CL \cup ST \cup END$; $ORD = \{ord_1, \ldots, ord_r\}$, $r \geq 0$, est l'ensemble des états ordinaires, $OP = \{op_1, \ldots, op_m\}$, $m \geq 0$, est l'ensemble des états d'ouverture, $CL = \{cl_1, \ldots, cl_m\}$, est l'ensemble des états de fermeture, $ST = \{s_1, \ldots, s_k\}$, $k \geq 0$, est l'ensemble des états de départ, $END = \{e_1, \ldots, e_k\}$, est l'ensemble des états de fin, Σ est un alphabet d'entrée fini, $q_0, q_f \in Q$ sont deux états distincts — l'état initial et l'état final,

$$T \subset \{\lambda\} \times OP \times Q \cup \{\lambda\} \times Q \times CL \cup \Sigma \times ORD \times ORD$$

$$\cup \{\lambda\} \times ST \times Q \times Q \cup \{\lambda\} \times Q \times Q \times END$$

est un ensemble fini de transitions. (source: [21])

Coût d'édition : poids associé (ou valeur associée) à une édition.

Cycle de calcul un cycle de calcul correspond à une itération de la boucle principale de calcul d'un algorithme.

Distance d'édition: mesure entre deux structures correspondant à la somme des coûts des *opérations d'édition* utilisées pour transformer une structure en l'autre suivant un *scénario d'édition* et un *schéma de score* donnés.

Intron : sous-séquence d'un ARN (pré-messager) qui est extraite de celui-ci et qui ne fera pas parti de l'ARN messager mature. Les introns sont extraits au cours d'un phénomène appelé épissage des gènes. Les autres segments de l'ARN messager, dénommés exons, sont joints entre eux après l'extraction de l'intron.

Opération d'édition: opération sur une structure menant à sa modification.

Problème NP : on dit qu'un problème est dans NP s'il existe un algorithme pour vérifier qu'une solution donnée convient en un temps polynômial (source: [48]).

Problème NP-Complet: on dit qu'un problème est NP-complet si la résolution de ce problème en temps polynômial entraîne la résolution en temps polynômial de tout problème NP (source: [48]).

Pseudonœud : élément structurel comportant deux tiges-boucles liées par certaines des bases de leurs boucles.

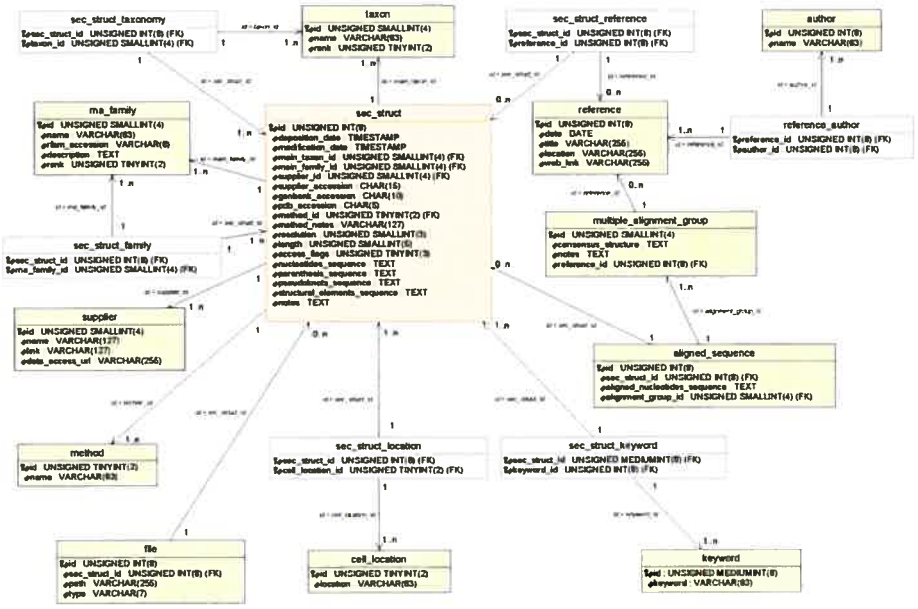
Rétro-homing : technique permettant de remplacer une partie de la séquence ADN d'un gène par de l'ARN à l'aide d'un intron adapaté à ce gène ([7]). La séquence d'ARN sera par la suite réparée en ADN permettant parfois de remplacer le contenu initial du gène par celui apporté par l'intron.

Scénario d'édition séquence ordonnée d'opérations permettant de transformer une structure en une autre.

Schéma de score ensemble des coûts associés aux *opérations d'édition* applicables à un modèle.

Structure secondaire forme de représentation en deux dimension d'un polymère.

5.2 Architecture de la base de données



5.3 Quelques acronymes d'ARN

ncRNA, sRNA, nmRNA: nom générique de la famille des ARN non-codants (en anglais: non-coding), des petits ARN (en anglais: small) ou des ARN non messagers (en anglais: non-messenger).

dsRNA : long ARN double-brin (en anglais: long Double-Stranded RNA).

gRNA : ARN qui joue un rôle dans l'édition d'ARN (en anglais: Guide RNA).

hnRNA : précurseur des ARN messagers trouvés dans le noyau (en anglais: Heterogeneous Nuclear RNA). Ce terme passé mode n'est pratiquement plus employé.

miRNA : ARN qui s'attache à l'ARNm d'un autre gène pour éventuellement l'inhiber (en anglais: MicroRNA).
La famille des miRNA inclu siRNA et stRNA.

mRNA : ARN messager (en anglais: Messenger RNA).

rRNA : ARN ribosomal (en anglais: Ribosomal RNA).

scRNA : ARN structural du cytoplasme (en anglais: Small cytoplasmic RNA).

shRNA : courte tige-boucle d'ARN variante des siRNA utilisée dans l'interférence par ARN (en anglais: Short Hairpin RNA).

siRNA : petit ARN d'interférence (en anglais: Small Interfering RNA). À ne pas confondre avec RNAi qui est le nom de la méthode d'interférence par ARN (en anglais: RNA interference).

snRNA : petit ARN structural du noyau (entre 100 et 300 nucléotides) impliqué dans la maturation de l'ARN messager (en anglais: Small Nuclear RNA).

snoRNA : petit ARN nucléolaire impliqué dans la maturation des ARN pré-ribosomaux (en anglais: Small Nucleolar RNA).

stRNA : petit ARN temporel (en anglais: Small Temporal RNA). ARN semblant réguler la temporisation du développement des animaux bilatéraux.

tRNA : ARN de transfert (an anglais: Transfer RNA).