Master recherche de Bioinformatique
Université de Rennes 1

Rapport de stage de Master recherche
Janvier - Juin 2005

SIEROCINSKI Thomas

Utilisation des propriétés des acides aminés dans le cadre de la caractérisation et détection de motifs dans les protéines

Résumé

La détection de motifs dans les protéines a actuellement de multiples intérêts en biologie comme l'extraction de séquences ayant une signature protéique commune ou encore la détermination de fonctions de nouvelles protéines par la localisation d'un motif connu dans leurs séquences. L'approche actuelle s'inscrit dans le cadre de la recherche d'expressions régulières sur l'alphabet des acides aminés. Nous nous proposons ici de mettre en place une méthode qui permet la détection de motifs basée sur les propriétés des acides aminés. Celle-ci a été implémentée dans un programme nommé PRAAP pour " Pattern Recognition with Amino Acid Properties " et testée dans deux cas d'applications. Nous avons pu remarquer que les résultats obtenus par notre approche sont sensiblement meilleurs que pour l'approche classique.

Laboratoire d'Etude et de Recherche en Informatique d'Angers
Responsables de stage : **Jean-Michel Richer** et **Jin-Kao Hao**

Remerciements

Je tiens à remercier tout particulièrement :

Jin-Kao Hao, Professeur et directeur du LERIA (Laboratoire d'Etude et de Recherche en Informatique d'Angers), pour m'avoir si bien accueilli au sein de son laboratoire.

Jean-Michel Richer, Maître de conférences et responsable de stage pour son encadrement, son écoute et sa disponibilité. Merci d'avoir mis à notre disposition tous les moyens matériels nécessaires au bon déroulement du stage et surtout de nous faire partager ses connaissances, son enthousiasme et son dynamisme.

Emmanuel Jaspard et **Gilles Hunault** pour leurs conseils et leurs avis critiques très fondateurs pour l'améloration de mon projet.

Marc-Olivier Buob, élève ingénieur à l'IIE, et **Didier Empis**, élève du Master de bioinformatique de Rennes, pour leur aide précieuse en programmation et leur humour au quotidien.

Olivier Cantin, **Hervé Deleau**, **Vincent Derrien**, **Adrien Goeffon**, **Olivier Hû**, **Tony Lambert**, **Frederic Lardeux**, **Thomas Raimbault**, **Antoine Robin** pour leur accueil très chaleureux et leur bonne humeur.

Tous les amis du Master Bioinformatique de Rennes ainsi que les anciens étudiants du DEA Génomique et Informatique et tout particulièrement **Sandrine Pawlicki** pour nous avoir permis de garder le contact par l'intermédiaire du forum étudiant.

Toutes les personnes que j'ai oubliées et qui ont contribué au bon déroulement de ce stage.

Table des matières

Chapitre 1

Introduction

Dans le cadre de la bioinformatique, un motif protéique est une expression définie sur l'alphabet des acides aminés et caractéristique d'une famille ou d'un groupe de protéines. Les motifs sont exprimés par rapport à la structure primaire des protéines c'est-à-dire la séquence polypeptidique. Ils sont souvent liés à une fonction particulière du groupe de protéines en question (site catalytique, site de fixation d'autres molécules...). Il existe deux grands domaines dans l'étude bioinformatique des motifs protéiques :

- la découverte de motifs [36] est un problème complexe qui consiste, à partir d'un ensemble de séquences considérées comme voisines, à déterminer un motif commun qui permet de caractériser ce groupe. Il existe de nombreuses méthodes [7] implémentées dans différents programmes : PRATT [18] [19], TEIRESIAS [11] [12] [31] [32], MEME [2] et son extension MetaMEME [15].

- la détection ou recherche de motifs consiste à rechercher une ou plusieurs occurrences d'un motif donné dans un ensemble de séquences. Il existe des outils qui permettent cette recherche : FUZZPRO [5], ScanProsite [13], Protein Pattern Find [33], PIR Pattern [38]. C'est cet aspect que nous aborderons dans ce rapport.

Devant la croissance exponentielle du nombre de génomes séquencés (desquels sont déduites des séquences protéiques), la demande en traitement informatique est en très forte augmentation. Les intérêts de la détection de motifs dans les protéines sont multiples :

- Essayer de déterminer la fonction d'une nouvelle protéine en localisant un ou plusieurs motifs déjà répertoriés dans des banques de motifs.

- Extraire des séquences protéiques possédant une même signature afin de les regrouper.

- Déterminer l'appartenance d'une protéine inconnue à une famille grâce à la présence d'un motif caractéristique.

Les méthodes de détection classiques (recherche d'expressions régulières) sont très utilisées dans cette optique. Pourtant, il est peut-être restrictif de se baser uniquement sur une lettre qui représente un acide aminé alors que celui-ci est caractérisé par des structures et/ou fonctions. C'est pourquoi nous nous proposons de concevoir un système de détection de motifs protéiques basé sur les propriétés (caractéristiques physico-chimique) des acides aminés.

Dans la suite de ce rapport, nous traiterons de la détection de motifs dans les protéines. Nous verrons tout d'abord comment sont définis les motifs (notation) basés sur l'alphabet des acides aminés, puis un aperçu des méthodes de détection dans les séquences protéiques. Ensuite nous présenterons notre méthode de détection basée sur les propriétés des acides aminés. Enfin, Nous terminerons par quelques applications.

Chapitre 2

La détection de motifs dans les protéines

La détection de motifs a nécessité la mise en place de notations, celle de type PROSITE [1] étant la plus répandue. De nombreuses applications (FUZZPRO[5], ScanProsite[13]...) sont disponibles pour la détection de motifs basée sur cette notation.

2.1 Alphabet des acides aminés

Les acides aminés sont généralement codés sur un caractère parmi vingt :

	code			code	
Nom	3 car.	1 car.	Nom	3 car.	1 car.
Glycine	gly	G	Alanine	ala	A
Valine	val	V	Leucine	leu	L
Isoleucine	ile	I	Méthionine	met	M
Proline	pro	P	Phénylalanine	phe	F
Tryptophane	trp	W	Sérine	ser	S
Thréonine	thr	T	Asparagine	asn	N
Glutamine	gln	Q	Tyrosine	tyr	Y
Cystéine	cys	C	Lysine	lys	K
Arginine	arg	R	Histidine	his	H
Aspartate	asp	D	Glutamate	glu	E

D'autres caractères sont également utilisés :

X = un acide aminé non déterminé
B = aspartate ou asparagine
Z = glutamate ou glutamine

Les caractères B et Z sont moins utilisés mais seront pris en compte dans l'implantation de notre méthode.

2.2 PROSITE et sa notation

La base de données PROSITE [1] [3] rassemble une vaste collection de signatures biologiques décrites par des motifs et des profils. Chaque signature est reliée à une documentation qui contient des informations

utiles sur la famille protéique, le domaine ou le site fonctionnel identifié par la signature.

Deux types de descripteurs sont utilisés pour identifier des régions conservées au sein de séquences polypeptidiques : les motifs et les profils. Un profil est un tableau de coefficients pour les acides aminés et de coûts de gaps[1] pour des positions données. Un motif ou expression régulière est un descripteur quantitatif : il identifie une séquence ou non. Un motif de qualité est habituellement localisé dans une région courte et bien conservée. De telles régions sont en général des sites catalytiques d'enzymes, des sites d'attachement à un groupement prosthétique, des cystéines engagées dans des ponts disulfures ou des régions de fixation d'autres molécules.

La détection de motifs s'inscrit dans la recherche d'expressions régulières au sein des séquences alphabétiques représentant l'enchaînement peptidique des protéines. Un des systèmes de représentation des ces motifs est la notation PROSITE [1] [3]. Les règles définies par cette notation utilisent l'alphabet des acides aminés et obéissent aux règles suivantes :

- Un motif est composé de sous-motifs séparés par un tiret (le tiret améliore la lisibilité de la notation).

- Pour chaque sous-motif :
  - On peut désigner un caractère autorisé ainsi que le nombre de répétitions
    Exemple : `C` autorise une cystéine sur une position.
    `A(1,2)` autorise une alanine sur une à deux positions.

  - On peut désigner un ensemble de caractères autorisés sur une ou plusieurs positions
    Exemple : `[ALC](1)` autorise une alanine, une lysine ou une cystéine sur une seule position.

  - On peut désigner un ensemble des caractères interdits sur une ou plusieurs positions
    Exemple : `{ALC}(2,5)` interdit l'alanine, la lysine et la cystéine sur 2 à 5 positions.

  - On peut aussi autoriser tous les caractères possibles avec la notation `X`
    Exemple : `X` autorise tous les caractères sur une seule position.
    X(1,4) autorise tous les caractères sur une distance de une à quatre positions.

Nous allons prendre le motif de site de fixation du glutamate des Glutamate DésHydrogénases [17] comme exemple suivant les règles de la notation de type PROSITE [17] :

```
K-G-G-X-R-X(12,23)-L-X(6)-K-X(4,6)-P-X-G-G-X-K
```

Cette expression sera capable de reconnaître toutes les séquences commençant par une lysine (K) suivi de deux glycines (G), d'un acide aminé quelconque (X), d'une arginine (R), de douze à vingt-trois acides aminés quelconques, d'une lysine, de six acides aminés quelconques, d'une lysine, de quatre à six acides aminés quelconque, d'une proline (P), d'un acide aminé quelconque, de deux glycines, d'un acide aminé quelconque et d'une lysine. Les acides aminés autres que X présents au sein du motif représentent des acides aminés conservés au sein de la famille protéique en question.

[1] Un gap est, dans le cadre des profils, une position ou un morceau de séquence ou manque un acide aminé. Ils sont généralements introduits dans le cadre d'alignements de séquences afin d'en améliorer la qualité.

Prenons la séquence de glutamate déshydrogénase du maïs (*Zea mays*), le motif ci-dessus permettra de reconnaître le segment de séquence souligné :

MNALAATSRNFKQAAKLVGLDSKLEKSLLIPFREIKVECTIPKDDGTLASYVGFRVQHDNARGPM
KGGIRYHHVDPDEVNALAQLMTWKTAVANIPYGGAKGGIGCSPGDLSISELERLTRVFTQKIHDP
LIGIHTDVCAPDMGTNSQTMAWILDEYSKFHGYSPAVVTGKPVDLGGSLGRDAATGRGVLFATEA
LLAEHGKGIAGQRFVIQGFGNVGSWAAQLISEAGGKVIAISDVTGAVKNVDGLDIAQLVKHSAEN
KGIKGFKGGDAIAPDSLLTEECDVLIPAALGGVINKDNANDIKAKYIIEAANHPTDPEADEILSK
KGVLILPDILANSGGVTVSYFEWVQNIQGFMWDEEKVNAELRTYMTRAFGDVKQMCRSHSCDLRM
GAFTLGVNRVARATVLRGWEA

2.3 Les méthodes de détection de motifs de type PROSITE

La base de données PROSITE est une collection de descriptions de sites protéiques. Pour chaque site, la base de données contient des informations diverses, notamment le motif. Ce motif est une expression formée sur l'alphabet des acides aminés (une classe de caractères) contenant des parties variables, c'est-à-dire les $X(a, b)$. Ces zones variables sont traduites par des gaps de taille bornée au sein des expression régulières représentant le motif. La notation PROSITE des motifs protéiques peut donc être assimilée à une expression formée de Classes de caractères et de Gaps de taille Bornée (CGB). Ces motifs sont utilisés pour rechercher une occurrence dans d'autres séquences protéiques.

Le problème de l'extraction rapide et de la recherche de motifs contenant des CGB est crucial pour l'identification de motifs protéiques dans des bases de données. Il existe deux manières de procéder : la première est de convertir le CGB en expression régulière et de la rechercher dans la séquence, la deuxième est de rechercher directement le CGB.

Généralement les motifs sont considérés comme des expressions régulières à part entière sur un alphabet fixe. Les expressions régulières, aussi appelées expressions rationnelles sont une famille de notations compactes et puissantes pour décrire certains ensembles de chaînes de caractères. Ce qui s'applique particulièrement bien à la détection de motif dans les protéines.

La détection de motifs utilise des algorithmes visant la recherche d'expressions régulières. Celles-ci doivent être converties en automates déterministes ou non déterministes puis recherchées dans la séquence. La majorité des programmes de reconnaissance de motifs de PROSITE utilisent ces techniques [28].

2.3.1 Recherche d'expressions régulières

La manière habituelle de rechercher une expression avec des CGB est de la convertir en automate fini non déterministe. Il est ensuite possible de rechercher le motif dans la séquence en utilisant l'automate ou de le convertir en automate fini déterministe.

Pour construire un automate fini non déterministe à partir d'une expression régulière, différentes techniques sont utilisées. La plus classique est la construction de Thompson [34] qui construit un automate fini non déterministe avec au plus $2m$ états (m représente le nombre de caractères de l'expression régulière) (fig. 2.1). Ces automates finis non déterministes possèdent des propriétés qui ont étés intensivement exploitées pour des algorithmes de recherche de motifs comme ceux de Thompson, Myers [26], Wu et Manber [39].

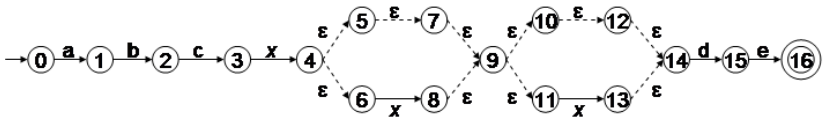

FIG. 2.1 – *Construction de Thompson sur un exemple a-b-c-x(1,3)-d-e.Dans ce cas l'automate construit possède 17 états, la transition entre deux états correspond au caractère de l'expression régulière. Par exemple la transition de l'état 0 à l'état 1 se fait par le caractère a. Le passage de l'état 3 à l'état 4 se fait par une transition x (un x dans l'expression), pour passer de l'état 4 à l'état 9 deux chemins sont possibles : le premier contenant une ϵ-transition ou transition vide, le second contenant une transition x (deux x dans l'expression). Il en est de même pour le passage de l'état 9 à 14.*

La seconde méthode est la construction de Glushkov démocratisé par Berry et Sethi (fig. 2.2) [4], l'automate résultant possède m+1 états.

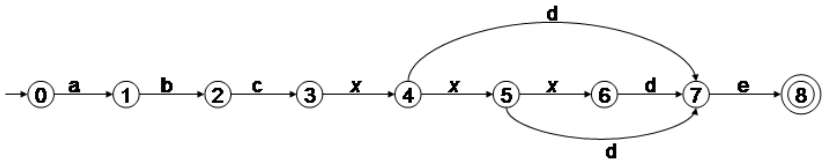

FIG. 2.2 – *Construction de Glushkov sur l'exemple a-b-c-x(1,3)-d-e. Comme pour la construction de Thompson la transition entre deux états correspond au caractère donné de l'expression régulière. Ce qui diffère de la construction précédente est la gestion du x(1,3) : en effet, le premier x apparaît entre l'état 3 et 4 (un x dans l'expression), après l'état 4 l'automate permet d'atteindre directement l'état 7 avec la transition d ou de passer à l'état 5 avec une transition x (deux x dans l'expression). De l'état 5, on peut atteindre directement l'état 7 par une transition d ou passer à l'état 6 avec une transition x (trois x dans l'expression).*

Une autre méthode [37] consiste à réduire la recherche d'expression régulière à une recherche toutes les séquences possibles de longueur donnée qui sont suceptibles d'être reconnues par l'expression régulière. Il existe d'autres méthodes plus adaptées aux CGB.

2.3.2 Méthode de recherche de CGB vers l'avant

Bien que leurs fonctionnalités soient identiques, les automates utilisés pour cette méthode ne correspondent pas aux automates pour la recherche d'expressions régulières. Un exemple de ce type d'automate est donné dans la figure 2.3.

Pour construire l'automate fini non déterministe, on commence par l'état S_0 puis le motif est lu caractère par caractère. Pour chaque caractère un arc est ajouté à l'automate. Après avoir créé l'état S_i, si le caractère suivant est C alors un nouvel état S_{i+1} et un arc annoté C reliant l'état S_i et S_{i+1} sont créés.

Si le caractère suivant est de type x(a,b), b états sont créés $S_{i+1}...S_{i+b}$ ainsi que des arcs annotés ϵ (mot vide) reliant les états S_j à S_{j+1} pour j appartenant à $[i, i+b-1]$. Le dernier état créé dans le processus est l'état final.

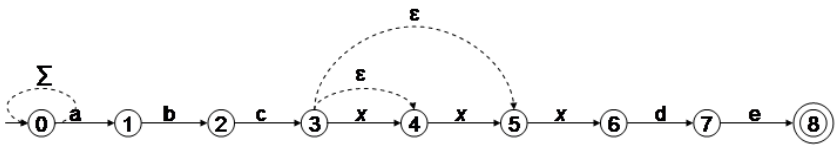

FIG. 2.3 – *Représentation de l'automate non-déterministe pour le motif a-b-c-x(1,3)-d-e. Trois transitions pouvant être suivies de n'importe quel caractère ont été insérées, ce qui correspond à la longueur maximale du "gap" $X(1,3)$. Deux ϵ-transitions (transitions vides) quittent l'état où a-b-c a été reconnu et saute une ou deux transitions. Ce qui permet de sauter de un à trois caractères dans la séquence avant de trouver le c-d à la fin du motif. La boucle sur l'état S_0 permet de débuter la reconnaissance sur n'importe quelle position de la séquence.*

2.3.3 Méthode de recherche de CGB vers l'arrière

Pour cette méthode l'automate parcourt la fenêtre de recherche à l'envers (fig. 2.5), les automates utilisés sont sensiblement identiques à ceux de la méthode de recherche vers l'avant, mais cet automate ne possède pas de boucle sur le premier état (fig.2.4).

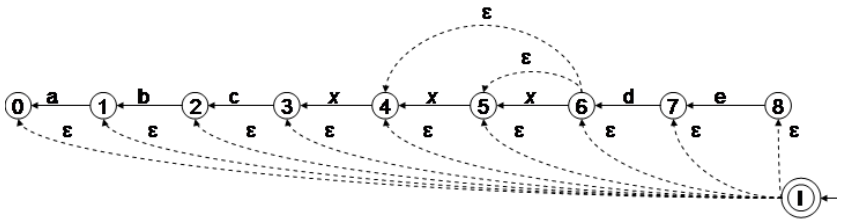

FIG. 2.4 – *Représentation de l'automate non-déterministe pour le motif a-b-c-x(1,3)-d-e. Un état initial I est créé ainsi que des ϵ-transitions de I à chaque état de l'automate. Une séquence lue par cet automate est un facteur du CGB tant qu'il contient au moins un état actif (un état est dit actif lorsque la transition à l'état suivant est possible).*

Bien que les deux dernières méthodes soient plus adaptées (au niveau du temps de calcul) à la recherche de motifs dans les protéines [28], l'utilisation des premières (recherche classique d'expressions régulières) est plus répandue dans les applications actuelles.

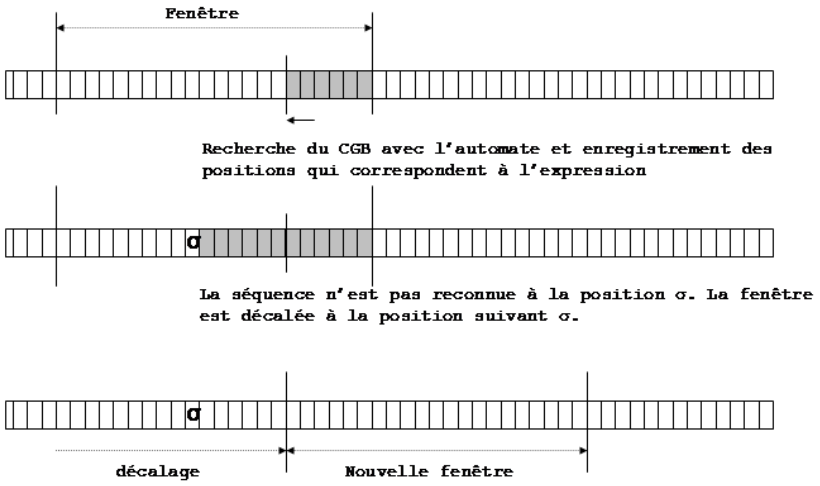

FIG. 2.5 – *La recherche est effectuée en glissant une fenêtre (de gauche à droite) de la taille du plus petit alignement possible (l) sur la séquence. La fenêtre est lue à l'envers (de droite à gauche) afin de reconnaître un facteur du motif. Si le début de la fenêtre est atteint, un alignement est trouvé. Si la recherche échoue, la fenêtre est décalée au début du facteur reconnu le plus long.*

Chapitre 3

La détection de motifs basée sur les propriétés des acides aminés

La détection de motifs basée sur les propriétés des acides aminés poursuit les mêmes objectifs que ceux vus précédemment mais, contrairement à la notation basée sur l'alphabet des acides aminés, nous ne les considérerons pas comme simples caractères. Chaque acide aminé possède ses propriétés physico-chimiques propres. De plus, sachant qu'un motif conservé au sein d'une famille de protéines est en général un site lié à une fonction particulière et caractéristique et que celle-ci est directement en relation avec les propriétés du motif, il est légitime de penser pouvoir détecter des motifs en se basant sur les propriétés des acides aminés, ce qui permet de conserver une approche tant biologique qu'informatique. Nous nous proposons ici de mettre en place une méthode qui permet la détection de motifs basée sur les propriétés des acides aminés.

3.1 Etude des acides aminés

Les propriétés retenues pour cette étude en accord avec **Gilles Hunault**, **Emmanuel Jaspard** et **Jean-Michel Richer** sont uniquement physico-chimiques et sont les suivantes : le poids moléculaire en Dalton noté M, le volume en A^3 noté V, l'hydrophobicité selon l'échelle de Kyte et Doolittle notée H [21], la polarité selon l'échelle de Grantham [14] notée P ainsi que la charge (neutre, positive, négative) notée C.

Afin de pouvoir implémenter cette méthode de détection de motif, nous avons créé des classes d'acides aminés pour chaque propriété étudiée. Ceci permet le regroupement d'acides aminés ayant des caractéristiques proches et aussi la mise en place une "*notation intuitive*". En effet, il sera plus aisé de rechercher dans une séquence un motif faiblement hydrophobe qu'un motif ayant une "*hydropathie de -3.5 ou de -3.9*".

3.1.1 Méthode de création de classes

Dans le but de répartir les acides aminés en classes pour chaque propriété, il est nécessaire d'appliquer une méthode statistique permettant de déterminer le nombre de classes ainsi que leurs bornes. On a choisi d'appliquer ici la technique des différences successives. Cette méthode parcourt les valeurs données pour les propriétés triées dans l'ordre croissant. Pour chaque valeur la différence avec la suivante est calculée et rapportée en pourcentage. Si elle dépasse un seuil fixé alors une séparation des données en groupes est effectuée au niveau de la moyenne arithmétique des deux points. Le nombre de classes est ici induit par

la méthode de découpage et donc est fonction du seuil choisi. Celui-ci a été déterminé selon le nombre approximatif de classes désiré pour les regroupements. Les acides aminés seuls dans un groupe ont étés rattachés à celui dont ils se rapprochaient le plus (par rapport à la moyenne arithmétique des deux valeurs encadrantes), ceci afin d'éviter les regroupements ne contenant qu'un seule valeur. Cette méthode donne des résultats sensiblement équivalents au regroupement via des quartiles.

Le poids moléculaire

Le poids moléculaire est exprimé en Daltons, le seuil choisi pour la séparation des classes est de 6%.

ac. aminé	**PM (Da)**	**Classes**		**ac.aminé**	**PM (Da)**	**Classes**	
G	75	M0	Très légers	D	133	M2	Moyens
A	89	M0		Q	146	M3	Lourds
S	105	M1	Légers	K	146	M3	
P	115	M1		E	147	M3	
V	117	M1		M	149	M3	
T	119	M1		H	155	M3	
C	121	M1		F	165	M4	
L	131	M2	Moyens	R	174	M4	Très lourds
I	131	M2		V	181	M4	
N	132	M2		W	204	M4	

Le volume

Le volume est exprimé en $Angstrom^3$, le seuil choisi pour la séparation des classes est ici de 8%.

ac. aminé	**V** (A^3)	**Classes**		**ac. aminé**	**V** (A^3)	**Classes**	
G	48	V0	Très petit	E	109	V2	Moyen
A	67	V0		H	117	V2	
S	73	V0		L	124	V2	
C	86	V1	Petit	I	124	V2	
P	90	V1		M	124	V2	
N	91	V1		F	135	V3	Volumineux
D	91	V1		K	135	V3	
T	93	V1		Y	141	V3	
V	105	V1		R	148	V3	
Q	109	V2	Moyen	W	163	V3	

L'hydrophobicité

L'hydrophobicité est représentée ici selon l'échelle de Kyte et Doolittle [21], le seuil de séparation de classe est de 11%.

ac. aminé	Hydrophobicité	Classes	
R	-4.5	H0	très faiblement hydrophobe
K	-3.9	H0	
N	-3.5	H1	faiblement hydrophobe
Q	-3.5	H1	
D	-3.5	H1	
E	-3.5	H1	
H	-3.2	H1	
P	-1.6	H1	
Y	-1.3	H2	moyennement hydrophobe
W	-0.9	H2	
S	-0.8	H2	
T	-0.7	H2	
G	-0.4	H2	
A	1.8	H3	fortement hydrophobe
M	1.9	H3	
C	2.5	H3	
F	2.8	H3	
L	3.8	H4	très fortement hydrophobe
V	4.2	H4	
I	4.5	H4	

La polarité

La polarité est représentée selon l'echelle de Gratham [14], le seuil de séparation des classes est de 7%.

ac. aminé	Polarité	Classes		ac. aminé	Polarité	Classes	
L	4.9	P0	Très faiblement polarisé	T	8.6	P1	Faiblement polarisé
I	5.2	P0		G	9	P1	
F	5.2	P0		S	9.2	P1	
C	5.5	P0		H	10.4	P2	Moyennement polarisé
M	5.7	P0		Q	10.5	P2	
V	5.9	P0		R	10.5	P2	
Y	6.2	P0		K	11.3	P3	Fortement polarisé
P	8	P1	Faiblement polarisé	N	11.6	P3	
W	8	P1		E	12.3	P3	
A	8.1	P1		D	13	P3	

La charge

La charge est divisée en trois classes (neutre, positif, négatif), aucune autre séparation n'est nécessaire.

ac. aminé	charge	Classes	ac. aminé	charge	Classes
G	Neutre	C0	T	Neutre	C0
A		C0	N		C0
V		C0	Q		C0
L		C0	Y		C0
I		C0	C		C0
M		C0	K	Positif	C1
P		C0	R		C1
F		C0	H		C1
W		C0	D	Négatif	C2
S		C0	E		C2

3.1.2 Tableau récapitulatif des propriétés

ac. aminé	Polarité	Charge	Hydrophobicité	PM	Volume
Y	P0	C0	H2	M4	V3
C	P0	C0	H3	M1	V1
M	P0	C0	H3	M3	V2
F	P0	C0	H3	M4	V3
V	P0	C0	H4	M1	V1
I	P0	C0	H4	M2	V2
L	P0	C0	H4	M2	V2
G	P0	C0	H2	M0	V0
S	P1	C0	H2	M1	V0
P	P1	C0	H2	M1	V1
T	P1	C0	H2	M1	V1
W	P1	C0	H2	M4	V3
A	P1	C0	H3	M0	V0
Q	P1	C0	H1	M3	V2
R	P2	C1	H0	M4	V3
H	P2	C1	H1	M3	V2
N	P2	C0	H1	M2	V1
K	P3	C1	H0	M3	V3
N	P3	C2	H1	M2	V1
D	P3	C2	H1	M3	V2

Nous pouvons remarquer à quelques exceptions près que certaines propriétés sont liées. Par exemple, la polarité, la charge et l'hydrophobicité. On remarque que dans la répartition qui est faite les acides aminés les plus polaires et chargés sont les moins hydrophobes. Il en est de même pour le poids moléculaire et le volume. Nous pouvons aussi remarquer que la classification de certains acides aminés est identique, ce qui est le cas pour :
– La Leucine et l'Isoleucine (I, L).
– La Proline et la Thréonine (P, T).

Ces acides aminés sont biochimiquement proches au vu de ces caractéristiques et de la répartition effectuée ici, nous avons prévu d'étendre les propriétés prises en compte afin de pouvoir distinguer ces acides aminés les uns des autres (voir conclusion et perspectives).

3.2 Méthodes et développement

3.2.1 Définition de la notation

Comme pour l'exemple de PROSITE, la détection de motifs basée sur les propriétés des acides aminés nécessite la mise en place de règles syntaxiques pour l'écriture des motifs recherchés. La notation utilisée ici est basée sur le regroupement en classes vu précédemment et s'inspire fortement de la notation PROSITE.

Motif :

sous-motif 0 - sous-motif 1 - ..... sous-motif n

Un motif est un ensemble de sous-motifs séparés par un tiret.

Sous motif :

```
[Ensemble de propriétés](Répétition)
X(répétition)
```

Un sous-motif est composé de deux parties :
– la première partie représente les ensembles de propriétés recherchées dans la séquence entre crochets.
– la deuxième partie représente le nombre de répétitions autorisées.

Les ensembles de propriétés :

`ppté1 & ppté2 | ppté3`

Les différentes propriétés d'un ensemble sont séparées par une expression logique & (ET) ou un | (OU). Par exemple la notation H0&V0 ne reconnaîtra que les acides aminés étant faiblement hydrophobes (H0) et minuscules (V0), si l'une ou l'autre des conditions n'est pas respecté alors l'acide aminé n'est pas reconnu. La notation H0|V0 reconnaîtra les acides aminés respectant une des deux conditions ou les deux, dans ce cas tous les acides aminés de classe H0 ainsi que tous les acides aminés de classe V0 seront reconnus. Cela implique que tous ceux qui respectent les deux conditions seront aussi reconnus.

Les répétitions :

`Nombre minimal de répétitions, Nombre maximal de répétitions`

Les répétitions sont représentées par un couple de valeurs qui correspondent au nombre minimal et au nombre maximal de répétitions autorisées. Par exemple : $[H0](2,5)$, permet de reconnaître tous les acides aminés de classe H0 au sein de la séquence sur une longueur de deux positions à cinq positions.
Soit la séquence suivante :

`EHRKEEKKRERKRRHE`

Dans cette séquence et avec la notation donnée, le motif [H0](2,3) possède quatre occurrences :

EHRKEEKKRERKRRHE
EHRKEEKKRERKRRHE
EHRKEEKKRERKRRHE
EHRKEEKKRERKRRHE

Les propriétés :

Les propriétés sont représentées par un caractère alphabétique, qui représente la propriété recherchée, suivi d'un chiffre qui indique la classe.

Les caractères alphabétiques des propriétés :

Les caractères alphabétiques et numériques des propriétés qui composent un motif doivent être choisis parmi ceux implémentés.

Les X :

Les X représentent n'importe quelle propriété sur toutes les classes.

La négation :

La notation ne comprend pas de négation ("`non H0`" par exemple) car elle contient un ensemble fini de valeurs. De cette manière "`non H0`" peut être écrit de la manière suivante : `H1&H2&H3&H4`. L'ajout de cette caractéristique fera partie d'une évolution ultérieure du programme (cf. perspectives).

Nous pouvons définir la syntaxe de description des motifs par la Grammaire BNF (Backus-Naur Form, notation permettant de décrire les règles syntaxiques d'un langage informatique [27]) suivante :

```
motif = sous-motif  "-" sous-motif
sous-motif = "[" expression_logique "]" repetition | "X" repetition
repetition = "(" entier "," entier ")"
entier = chiffre chiffre
chiffre = "0"|"1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"
expression_logique = expression_simple [operateur expression_simple]
expression_simple = propriete chiffre
propriete = "V"|"H"|"M"|"C"|"P"
operateur = "&" | "|"
```

Un exemple de motif :

`[C1](1,1)-[C0&V0](2,2)-X(1,1)-[C1](1,1)`

Le motif présenté ici représente une séquence contenant un acide aminé chargé positivement suivi de deux acides aminés de petit volume et non chargés, d'un acide aminé quelconque et pour finir d'un acide aminé chargé positivement.

3.2.2 Méthode

La méthode a été implémentée dans un programme que l'on nommera PRAAP pour "Pattern Recognition with Amino Acid Properties". Celui-ci est codé en JAVA, langage orienté objet qui a été choisi pour sa portabilité et sa polyvalence. PRAAP fonctionne en ligne de commande et prend en paramètres un fichier de séquences (de type FASTA) et un motif contenu dans un fichier texte et respectant la notation décrite précédemment. Le programme recherche ensuite toutes les occurrences du motif au sein des séquences. Pour chaque séquence, la sortie est de type :

Reference de la séquence
Séquence
Marquage des positions des occurrences du motif

Exemple :

```
gi|7431775|pir||T04342 glutamate dehydrogenase (EC 1.4.1.2) - maize
MNALAATSRNFKQAAKLVGLDSKLEKSLLIPFREIKVECTIPKDDGTLASYVGFRVQHDNARGPM
KGGIRYHHEVDPDEVNALAQLMTWKTAVANIPYGGAKGGIGCSPGDLSISELERLTRVFTQKIHD
LIGIHTDVPAPDMGTNSQTMAWILDEYSKFHGYSPAVVTGKPVDLGGSLGRDAATGRGVLFATEA
LLAEHGKGIAGQRFVIQGFGNVGSWAAQLISEAGGKVIAISDVTGAVKNVDGLDIAQLVKHSAEN
KGIKGFKGGDAIAPDSLLTEECDVLIPAALGGVINKDNANDIKAKYIIEAANHPTDPEADEILSK
KGVLILPDILANSGGVTVSYFEWVQNIQGFMWDEEKVNAELRTYMTRAFGDVKQMCRSHSCDLRM
GAFTLGVNRVARATVLRGWEA
motif1 :
------------------------------------------------------------------
**********************************----------------------------
------------------------------------------------------------------
------------------------------------------------------------------
------------------------------------------------------------------
------------------------------------------------------------------
----------------------
```

3.2.3 Représentation interne

L'interprétation des motifs est effectuée grâce à JFLEX [20], un générateur d'analyseur syntaxique pour JAVA, ce qui permet la prise en charge des motifs par PRAAP. Le motif complet est décomposé en sous-motifs qui sont ensuite représentés sous forme d'arbres binaires auxquels sont associés les nombres de répétitions du sous-motif (fig. 3.1). Les noeuds des arbres sont ici les opérateurs et les feuilles sont les propriétés. Le sens de lecture des arbres est fils gauche - opérateur - fils droit.

3.2.4 Détection du motif

Le système prend en entrée un fichier de type FASTA ainsi qu'un motif et recherche la présence de ce dernier sur chaque séquence. Pour chaque séquence traitée le programme crée une matrice de correspondance de propriétés où chaque ligne représente une propriété donnée et chaque colonne contient la classe de propriété de l'acide aminé correspondant.

Le programme parcourt ensuite la matrice de propriétés créée afin de détecter les positions de la séquence qui correspondent à chaque sous-motif. Les positions respectant les conditions requises par le sous-motif sont retenues dans un tableau (vrai si correspondance, faux sinon). L'opération est répétée

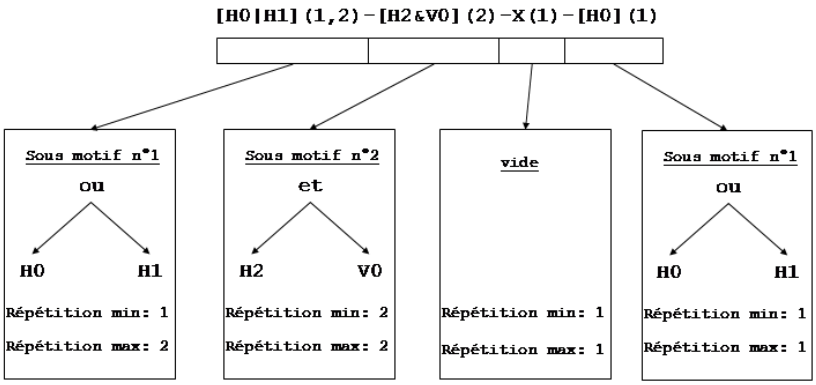

FIG. 3.1 – *Représentation interne de la notation : chaque sous-motif est représenté sous forme d'arbre associé au nombre de répétitions autorisées. Les sous-motifs de type* `X(a,b)` *sont représentés par un arbre vide.*

jusqu'à la fin du motif (dernier sous-motif). On obtient alors les coordonnées de reconnaissance de la séquence pour chaque sous-motif. Pour les sous-motifs de type `X(a,b)`, qui reconnaissent tous les caractères de la séquence, il n'est pas nécessaire de retenir les coordonnées d'identification dans la séquence.

Prenons par exemple le motif suivant :

`[H0|H1](1,2)-[H2&V0](2)-X(1)-[H0](1)`

Et la séquence suivante :

`KASACCSSACRKSGARGSRCAACKCSA`

Pour cette séquence la matrice de correspondance est la suivante :

	KASACCSSACRKSGARGSRCAACKCSA
Masse M	30102211024310040142002321O
Volume V	30001100013300030031001310O
Hydrophobicité H	03233322330022302203333032 3
Polarité P	31110011102311121120110301 1
Charge C	10000000001100010010000100 0

Prenons le premier sous-motif `[H0|H1](1,2)`, les positions détectées dans la matrice de correspondance sont soulignées dans la figure ci-dessus. La détection est ensuite effectuée pour chaque sous-motifs, ce qui donne :

		KASACCSSACRKSGARGSRCAACKCSA
sous-motif 1	[H0—H1](1,2)	*---------**---*--*----*---
sous-motif 2	[H2&V0](2)	------**----**--**---------
sous-motif 4	[H0](1)	*---------**---*--*----*---

Il est évident que toutes ces coordonnées ne correspondent pas au motif entier. Il est donc nécessaire de discerner celles correspondant au motif complet. Sachant que les sous-motifs doivent se présenter dans l'ordre décrit dans le motif et qu'ils ne peuvent être disjoints (sauf lorsqu'un `X(a,b)` sépare deux sous-motifs), Il devient donc plus aisé de retrouver le motif complet dans la matrice.

La méthode utilisée ici est un algorithme récursif borné par le nombre de sous-motifs (fig.3.2) : Celui-ci parcourt la ligne notée l (en commençant par la première) de la matrice d'identification jusqu'à ce qu'il trouve une position (qu'on notera p) qui a été reconnue sur une longueur *(s minimum, s maximum)* par le sous-motif numéro m. On parcourt alors la ligne suivante sur les positions comprises dans *[s min +1 ; s max +1]* (on notera cette positon S), il y a alors deux cas :

1. Le sous-motif suivant est de type *X(a,b)* : dans ce cas la ligne *l+1* est parcourue de la position *p+S+a+1* à *p+S+b+1*. Deux cas se présentent alors :

   (a) Aucune position reconnue ne se situe dans la fenêtre *[p+S+a+1 ; p+S+b+1]* : on retourne alors à la ligne l de la matrice à la position *p+S+1*.

   (b) Une ou plusieurs positions reconnues se situent dans la fenêtre *[p+S+a+1 ; p+S+b+1]* : on effectue un appel (récursif) de cette méthode avec les paramètres courants *(ligne courante et position courante)*.

2. Le sous-motif suivant n'est pas de type *X(a,b)* : la ligne *l+1* est alors parcourue à partir de la position *p+S+1*. Deux cas se présentent :

   (a) Il n'y a pas de position reconnue en *p+S+1* : on retourne alors à la ligne l de la matrice à la position *p+S+1*.

   (b) Une position reconnue se situe en *p+S+1* : on effectue l'appel de cette méthode avec les paramètres courants *(ligne courante et position courante)*.

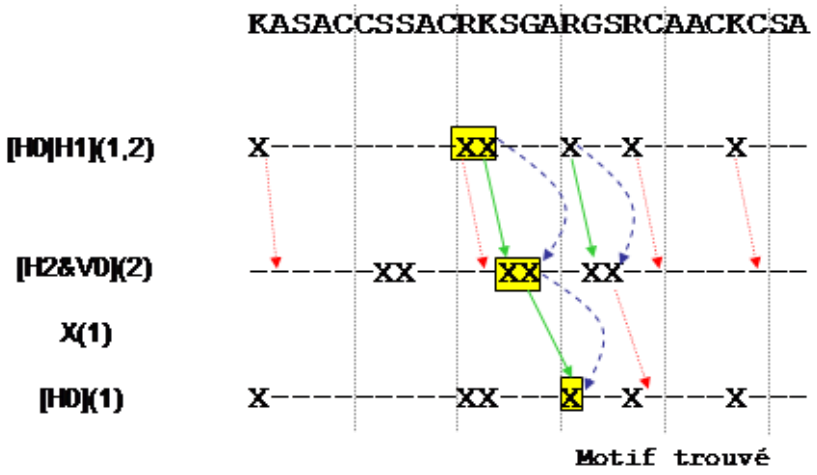

FIG. 3.2 – *Représentation de l'algorithme de recherche du motif complet.*

Algorithm 1 Algorithme de recherche du motif complet

Entrée : Matrice de détection des sous-motifs
Sortie : Coordonnées des occurences du motif complet

- **Détection (sous-motif, position dans la séquence)**
  - Si (sous-motif est le dernier sous-motif ou fin de la séquence)
    Arreter la méthode ;
- Si (sous-motif est de type X(MIN,MAX))
  - pour(i = MIN ; i <= MAX ; i++)
    Détection (sous-motif suivant, position dans séquence+i) ;
- Sinon
  - pour (l = MIN ; l <= MAX ; l++)
    - Si (répétition(sous-motif,position dans la séquence,l)== vrai)
      Retenir la position de l'occurrence du sous-motif et sa longueur ;
    - Si (Motif complet trouvé)
      Afficher la séquence et l'occurrence trouvée ;
    - Détection (sous-motif suivant, position+l) ;
- **répétition (sous-motif, position dans la séquence, l)**
  - Si (position + l > longueur de la séquence) retourne faux ;
  - Si (l == 0) retourne vrai ;
  - pour (i = 0 ; i<l ; i++)
    - si (sous-motif non reconnu à (position+i)) retourne faux ;
  - retourne vrai ;

De cette manière nous sommes capables de reconnaître toutes les occurrences d'un motif dans une séquence protéique.

```
KASACCSSACRKSGARGSRCAACKCSA
----------******-----------
-----------*****-----------
```

Comparativement aux méthodes basées sur la construction d'automates, celle-ci est probablement moins performante en temps d'éxecution mais plus aisée à implémenter (pas de création d'automates). Le temps de calcul pour rechercher toutes les occurrences de motifs dans un fichier de séquences reste néanmoins raisonnable (19 secondes pour 2016 séquences sur une machine de type AMD Sempron 2500+ 1.75GHz).

Chapitre 4

Applications

4.1 La glutamate déshydrogénase

La méthode présentée a été testée, en collaboration avec **Emmanuel Jaspard de l'UMR INRA 1191 Physiologie Moléculaire des Semences de l'Université d'Angers**, sur des séquences de glutamate déshydrogénase. Ces enzymes appartiennent à la famille des déshydrogénases.

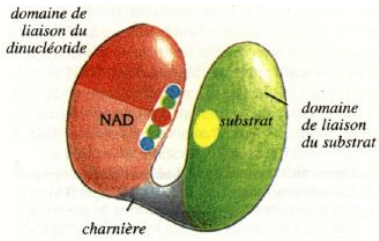

FIG. 4.1 – *Structure schématique d'une sous unité de déshydrogénase.*

Les déshydrogénases (fig 4.1) sont des enzymes multimériques constituées de deux, quatre ou six sous unités identiques qui sont structurées en deux domaines :

- un domaine de fixation du NAD(P)+. Ces domaines ont des structures tridimensionnelles très semblables. Ce site de fixation contient un motif structural appelé "Pli Rossman".
- un domaine de fixation du substrat qui est spécifique de la déshydrogénase considérée (lactate DH, malate DH, alcool DH...). Ces domaines ont des structures très différentes. En effet, chaque domaine a une topologie optimisée pour assurer la spécificité de reconnaissance du substrat.

Les glutamate déshydrogénases sont capables de catalyser la réaction suivante :

$$2 - oxoglutarate + NH_4 + + NAD(P)H + H^+ \leftrightarrow Glutamate + H_2O + NAD(P)^+$$

Il existe trois isoformes de cette enzyme :
- GDH EC 1.4.1.2 (GDH2) qui catalyse essentiellement la formation du 2-oxoglutarate en utilisant le $NAD(P)^+$ [10]. Cette famille d'enzyme se présente sous forme dimérique [35], tétramérique [23] ou hexamérique [8].

– GDH EC 1.4.1.3 (GDH3) qui catalyse la formation du 2-oxoglutarate et la réaction inverse, ceci résultant d'une double spécificité du coenzyme $[NAD(P)^+/NAD(P)H]$ [22]. Les GDH3 sont essentiellement sous forme hexamèrique [25].
– GDH EC 1.4.1.4 (GDH4) catalyse la formation du glutamate en utilisant le $NAD(P)H$ [9] [6]. Les GDH de cette famille se présentent sous forme tétramérique [16] ou hexamèrique [29].

Chaque sous unité de glutamate déshydrogénase comporte un domaine de fixation du NAD(P)+ (appelé pli Rossman) et un domaine de fixation du substrat (Glutamate). Par alignement multiple de séquences (116 réparties en 15 groupes), on peut remarquer que le domaine de fixation du glutamate est très conservé au sein de la famille et qu'il est possible de la caractériser de la manière suivante [17] :

`K-G-G-X-R-X(12,23)-L-X(6)-K-X(4,6)-P-X-G-G-X-K`

Le domaine de fixation du NAD(P)+ est caractérisé de la manière suivante dans la littérature [24] :

`G-X-G-X-X-[GAS]`

4.1.1 Le domaine de fixation du substrat

Le domaine de fixation du substrat de la glutamate deshydrogénase correspond à un motif très conservé au sein de la famille. Ce domaine contient trois résidus lysine :

K_{166}`-G-G-X-R-X(12,23)-L-X(6)-`K_{190}`-X(4,6)-P-X-G-G-X-`K_{202}

Les lysines 190 et 202 sont retrouvées dans tous les groupes. Pour certaines glutamate Déshydrogénases, il existe une substitution conservative de la lysine 166 en arginine et une insertion de neuf acides aminés (au niveau du X(12,23)) pour un sous groupe. Cette région riche en lysines correspond au site de fixation du $[2-oxoglutarate/Glutamate]$. Les lysines 166 et 190 forment des ponts salins avec les deux groupements carboxyles du $2-oxoglutarate$ ou du $glutamate$. La lysine 202 est impliquée dans la catalyse de la réaction plutôt que dans la fixation du substrat [17]. C'est en partant du motif initial ainsi que des propriétés liées aux fonctions ci-dessus que nous avons défini le motif suivant :

`[C1](1)-[C0&V0](2)-X(1)-[C1](1)-X(19,30)-[C1](1)-X(6,8)-[V0](2,3)-[C1](1)`

Ce motif contient des acides aminés de charges positives ($[C1](1)$ propriété nécessaire pour la fixation du glutamate et la catalyse de la réaction), le premier ainsi que le dernier $[C1]$ sont voisins d'acides aminés de petite taille. Les autres positions étant très peu conservées dans les séquences, ont été matérialisées par des X.

Détection du motif

Pour cette étude, nous avons choisi de "comparer" les résultats de la détection de motifs PROSITE avec notre approche. Pour la détection de motif de type PROSITE, nous avons choisi FUZZPRO [5], qui fait partie du package EMBOSS téléchargeable sur Internet [30]. Cet outil a été choisi parce qu'il s'installe en local contrairement à la majorité des outils en ligne (ScanProsite, Protein Pattern Find, PIR pattern) ce qui permet une analyse plus rapide et plus aisée des fichiers. FUZZPRO permet de scanner des fichiers de séquences pour la recherche d'un motif donné, il est possible de paramétrer un nombre de mésappariements du motif avec la séquence (ce qui correspond à remplacer un caractère défini du motif par un X).

Les jeux de données pris pour l'étude sont les suivants :
- Ensemble 1 : 116 séquences de glutamate déshydrogénase réparties en 15 groupes, auxquelles on a rajouté les 15 séquences consensus[1] de chaque sous-famille.
- Ensemble 2 : 2016 séquences autres que glutamate déshydrogénase choisies pour leur absence de site de fixation du glutamate et de pli Rossman : des cyclines dont les A, B, cdc2, des sous-unités α et β d'hémoglobine, tubulines α et β et ATP-synthases procaryotes.

Une détection de motif à été lancée sur chaque jeu de données avec les motifs suivants :
- FUZZPRO : `K-G-G-X-R-X(12,23)-L-X(6)-K-X(4,6)-P-X-G-G-X-K`
- PRAAP : `[C1](1)-[C0&V0](2)-X(1)-[C1](1)-X(19,30)-[C1](1)-X(6,8)-[V0](2,3)-[C1](1)`

Le premier tableau donne les résultats obtenus avec FUZZPRO sur les deux jeux de données en fonction du nombre de mésappariements autorisés. Pour chaque résultat, le taux de rappel ainsi que le taux de précision ont étés calculés de la manière suivante :

$$Taux\ de\ rappel = \frac{Nombre\ de\ GDH}{Nombre\ total\ de\ GDH\ (ensemble1)}$$

$$Taux\ de\ précision = \frac{Nombre\ de\ GDH}{Nombre\ de\ GDH\ +\ Nombre\ de\ non\ GDH)}$$

FUZZPRO				
Nb mesapp.	GDH	non GDH	Rappel	Précision
0	86	0	65.64	100
1	115	0	87.78	100
2	116	0	88.54	100
3	127	7	96.94	94.77
4	129	159	98.47	44.79
5	131	1177	100	10.01

PRAAP			
GDH	non GDH	Rappel	Précision
123	2	93.89	98.4

On constate que, pour FUZZPRO, le rappel augmente avec le nombre de mésappariements autorisés ce qui traduit une augmentation de GDH reconnues. Par contre la précision baisse énormément lorsque plus de quatre mésappariements sont autorisés. On peut considérer, dans ce cas, que FUZZPRO détecte le motif de manière optimale pour un nombre de trois mésappariements autorisés. En comparant ces résultats avec ceux de PRAAP, on constate qu'ils sont sensiblement les mêmes : le rappel est un peu moins bon pour PRAAP que pour FUZZPRO (respectivement 93.89% et 96.94%), la précision quand à elle est meilleure pour PRAAP que pour FUZZPRO (respectivement 98.4% et 94.77%). Dans ce cas, les résultats de la détection de motif basée sur les propriétés des acides aminés donne des résultats sensiblement identiques (moins de rappel mais plus de précision).

[1] Une séquence consensus reflète les acides aminés communs pour chaque position d'un ensemble de séquences alignées. Une séquence consensus représente donc les zones conservées au sein d'un groupe de séquences.

4.1.2 Domaine de fixation du NAD(P)+ : le pli Rossman

Le domaine de fixation du NAD(P)+ est une structure très conservée au sein de la famille des déshydrogénases. Il est caractérisé par une structure en sandwich $\beta\alpha\beta$ (feuillet β, hélice α, feuillet β). Cette structure permet la fixation du NAD(P)+ et est stabilisée par des liaisons H. Ce domaine est caractérisé dans la littérature par le motif G-X-G-X(2)-G ou G-X-G-X(2)-A ou encore G-X-G-X(2)-S [24]. Ces trois motifs peuvent être réunis au sein de la même notation : G-X-G-X(2)-[GAS].

Pour tester ces motifs, les mêmes outils (FUZZPRO, PRAAP) ainsi que les mêmes jeux de données que précédemment ont étés utilisés. Cependant, deux groupes (contenant 10 + 1 séquences + les deux séquences consensus) de glutamate déshydrogénases ont été écartés à cause d'une moins bonne conservation du motif par rapport aux autres. Cela ramène le nombre de séquences de glutamate déshydrogénases à 118. Pour les non GDH (et non déshydrogénase), le nombre de séquences reste inchangé (2016).

Lors de l'alignement des séquences de glutamate déshydrogénases on constate qu'il existe des substitutions conservatives au niveau de certaines propriétés pour certaines positions dans le motif.

```
                                GAGNVA
Alignement des séquences        GFGNVG
consensus :                     GFGNAG
                                GSGNVA
```

En fonction de ces alignements et du rôle du domaine, le motif basé sur les propriétés adopté est le suivant :

`[V0&C0&H2&P1&M0](1)-[H2|H3](1)-[V0&C0&H2&P1&M0](1)-[C0&V1](2)-[C0&V0&P1&M0](1)`

Nous pouvons remarquer que le premier ainsi que le troisième sous-motif regroupent l'ensemble des propriétés de la glycine (on pourra éventuellement élargir la notation *cf. conclusion et perspectives*). Le deuxième sous-motif reconnaîtra les acides aminés faiblement ou moyennement hydrophobes, le cinquième sous-motif reconnaîtra une série de deux petits acides aminés non chargés.

Détection du motif

La détection de ces deux motifs a été lancée sur les deux jeux de données décrits ci-dessus :
- FUZZPRO : `G-X-G-X(2)-[GAS]`
- PRAAP : `[V0&C0&H2&P1&M0](1)-[H2|H3](1)-[V0&C0&H2&P1&M0](1)-[C0&V1](2)-[C0&V0&P1&M0](1)`

Le premier tableau décrit les résultats obtenus avec FUZZPRO. Dans ce cas, au vu de la faible spécificité du motif (seulement trois positions définies) et des résultats obtenus par défaut nous avons choisi de ne pas introduire de mésappariements dans le test.

FUZZPRO			
Motif	Non motif	Rappel	Précision
108	922	91.52	10.48

PRAAP			
Motif	Non motif	Rappel	Précision
101	90	85.59	52.87

Dans le cas de FUZZPRO, on obtient un rappel de 91,5% et une précision de 10,5%. Dans le cas de PRAAP, le rappel obtenu est de 85,6% avec 52,87% de précision. Ces résultats peu précis s'expliquent par la faible spécificité du motif. En effet, le motif est très court et codé sur six caractères ce qui implique que l'on peut retrouver des séquences correspondantes fréquemment même si les protéines en question ne sont pas des déshydrogénases. Nous pourrons quand même noter que même si le rappel est plus faible pour PRAAP (85,59% contre 91,52), la précision est meilleure (52,87% contre 10,48%). On peut donc penser que, dans ce cas, la méthode basée sur les propriétés des acides aminés semble conférer un meilleur taux de précision à la détection de motifs dans les protéines.

Chapitre 5

Conclusion et perspectives

Dans les deux cas d'application ci-dessus, nous pouvons remarquer que les résultats obtenus par l'approche basée sur les propriétés sont sensiblement identiques voire meilleurs que pour l'approche "classique". Ces résultats peuvent s'expliquer par une "flexibilité" du motif, c'est-à-dire que le motif de propriétés peut accepter sur une position un ensemble d'acides aminés (dont souvent des substitutions conservatives au niveau des propriétés). Dans le premier cas d'application, les résultats pour FUZZPRO ont étés exprimés en fonction du nombre de mésappariements autorisés lors de la détection du motif. Nous pouvons assimiler ce nombre de mésappariements à une flexibilité non contrôlée du motif. En effet, les mésappariements peuvent se faire sur n'importe quelle position et pour n'importe quel type d'acide aminé. Dans ce cas ils peuvent se faire sur un acide aminé conservé et important pour une fonction particulière et donc en autoriser un autre (avec des caractéristiques totalement différentes), il en résultera la détection d'un motif erroné. Prenons l'exemple, du domaine de fixation du glutamate : le rôle des lysine est ici très important, c'est leur charge positive qui permet la fixation du substrat. En autorisant des mésappariements, une séquence ne possédant pas de lysine mais une glycine pour une position donnée pourra être reconnue alors qu'elle n'aura pas la fonction biologique de fixation du glutamate.

Dans le cas de la détection de motifs basée sur les propriétés des acides aminés, la flexibilité concerne des positions précises dans le motif. Elle est surtout biologiquement justifiée, c'est-à-dire qu'elle autorise les substitutions conservatives sur les propriétés. Reprenons l'exemple de la lysine 166 dans le motif précédent, avec la notation `[C1](1)` nous autorisons les acides aminés chargés positivement ce qui a un sens biologique. Cette flexibilité contrôlée explique donc les résultats et surtout le meilleur taux de précision de PRAAP, elle justifie aussi l'inutilité d'autoriser des mésappariements avec cette méthode.

Cependant, cette méthode n'est applicable que lorsqu'on dispose d'un motif basé sur les propriétés. Dans cette optique, deux solutions sont envisageables :
- la première consisterait en un système de découverte de motifs comparable à ce qui existe actuellement (PRATT, MEME...) mais axé sur les propriétés des acides aminés.
- la deuxième consisterait en un système de traduction de la notation PROSITE en fonction de propriétés déterminées par un expert (biologiste). C'est cette méthode qui a été appliquée au cours de cette étude.

En ce qui concerne cette méthode, nous pourrons l'affiner en ajoutant des propriétés qui ont été écartées pour l'instant (nous pourrions par exemple ajouter un coefficient de présence pour les hélices α, les feuillets β ou les boucles). Ainsi, il sera possible de caractériser un motif avec plus de propriétés. Même si les propriétés ajoutés à la classification ne permettent pas de mieux caractériser les acides aminés les uns par rapport aux autres, l'utilisation n'en sera que plus étendue (enrichissement du catalogue de propriétés).

Dans ce cadre, la conception du programme PRAAP permet une évolutivité aisée, l'addition d'une ou plusieurs propriétés se résume à l'ajout simple des caractéristiques désirées pour chaque acide aminé dans le code source du programme.

Il est aussi envisageable, pour des raisons de simplicité d'utilisation, de faire évoluer la notation. Dans le cas du pli Rossman, nous avons pu remarquer que des glycines sont très conservées sur la première et troisième position du motif. Avec la notation actuelle, si l'on ne veut détecter que les glycines dans la séquence, il est nécessaire de définir l'ensemble de ses propriétés dans le motif (`[V0&C0&H2&P1&M0](1)` au lieu de `G`). Il serait donc appréciable d'intégrer la notation d'acides aminés. Dans ce cas, afin de ne pas confondre les acides aminés avec les propriétés (par exemple le C de la propriété Charge ou de la Cystéine), nous pourrons envisager d'écrire les propriétés en minuscules et les acides aminés en majuscules. Dans l'exemple du pli Rossman, cette notation serait la suivante :

```
G-[h2|h3](1)-G-[c0&v1](2)-[c0&v0&p1&m0](1)
```

Dans le cadre d'une application plus étendue de cet outil, nous pourrons doter le logiciel d'une interface graphique afin de rendre l'utilisation plus conviviale, d'une interface WEB et d'un module permettant de détecter les motifs dans des bases de données.

Références

[1] *The PROSITE database of protein families and domains,User Manual*. Swiss Institute of Bioinformatics Switzerland.

[2] T. L. Bailey and C. Elkan. Fitting a mixture model by expectation maximisation to discover motifs in biopolymers. *Proceedings of the second international conference on Intelligent Systems for Molecular Biology*, 1994.

[3] A. Bairoch. Prosite : a dictionnary of sites and patterns in proteins. *Nucleic Acids Research*, 19 :2241–2245, 1991.

[4] G. Berry and R. Sethi. From regular expression to deterministic automata. *Theor. Comput. Sci.*, 48 :117–126, 1986.

[5] Alan Bleasby. *fuzzpro, user's manual*. Wellcome Trust Genome Campus, Hinxton, Cambridge, CB10 1SB, UK.

[6] B. Botton and Y. Msatef. Purification and properties of nadp-dependent glutamate dehydrogenase from sphaerostilbe repens. *Physiol. Plant*, 59 :438–444, 1983.

[7] A. Brazma, I. Jonassen, I. Eidhammer, and D. Gilbert. Approaches to the automatic discovery of patterns in biosequences. 1997.

[8] S. Chavez and P. Cau. A nad-specific glutamate dehydrogenase from cyanobacteria. identification and properties. *FEBS Lett*, 285 :35–38, 1991.

[9] J. W. Coulton and M. Kapoor. Studies on the kinetics and regulation of glutamate dehydrogenase of salmonella typhimurium. *Can. J. Microbiol.*, 19 :439–450, 1973.

[10] P. A. Duncan, B. A. White, and R. I. Mackie. Purification and properties of nadp dependent glutamate dehydrogenase from ruminococcus flavefaciens. *Appl Environ Microbiol*, 58 :4032–4037, 1992.

[11] A. Floratos. *Pattern discovery in biology : theory and applications*. PhD thesis, 1999.

[12] A. Floratos and I. Rigoutsos. Research report on the time complexity of the teiresias algorithm. Technical report, IBM Research, 1998.

[13] A. Gattiker, E. Gasteiger, and A. Bairoch. Scanprosite : a reference implementation tool of a prosite scanning tool. *Applied Bioinformatics*, 1(2) :107–108, 2002.

[14] R. Grantham. Amino acid scale : Polarity. *Science*, 185 :862–864, 1974.

[15] W. N. Grundy, T. L. Bailey, C. P. Elkan, and M. E. Baker. Meta-meme : Motif-based hidden markov models of protein families. *Computer applications in the Biosciences*, 13 :397–406, 1997.

[16] R. C. Hudson, L. D. Ruttersmith, and R. M. Daniel. Glutamate dehydrogenase from the extremely thermophilic archaebacterial isolate an1. *Biochim. Biophys. Acta*, 1202 :244–250, 1993.

[17] E. Jaspard. Analysis of the putative structure - function relationship of glutamate dehydrogenase ec 1.4.1.4 from viridiplantae. a probable key role of this isoform in ammonium assimilation by plants. submitted.

[18] I. Jonassen. *PRATT version 2.1 Documentation.*

[19] I. Jonassen, J. F. Collins, and D G. Higgins. Finding flexible patterns in unaligned protein sequences. *Protein Science*, 4 :1587–1595, 1995.

[20] Gerwin Klein. *JFlex, Fast Lexical Analyser Generator, User's Manual.*

[21] J. Kyte and R. F. Doolittle. Amino acid scale : Hydropathicity. *J. Mol. Biol.*, 157 :105–132, 1982.

[22] P. Maulik and S. Ghosh. Nadph/nadh-dependent cold-labile glutamate dehydrogenase in azospirillum brasilense. purification and properties. *Eur. J. Biochem.*, 155 :595–602, 1986.

[23] M. J. Meredith, R. M. Gronostajski, and R. R. Schmidt. Physical and kinetic properties of the nicotinamide adenine dinucleotide-specific glutamate dehydrogenase purified from chlorella sorokiniana. *Plant Physiol.*, 61 :967–974, 1978.

[24] B. Minambres, E. R. Olivera, R. A. Jensen, and J. M. Luengo. A new class of glutamate dehydrogenases (gdh). biochemical and genetic characterization of the first member, the amp-requiring nad-specific gdh of streptomyces clavuligerus. *J. Biol. Chem.*, 275 :39529–39542, 2000.

[25] E. Moyano, J. Cardenas, and J. Munoz-Blanco. Purification and properties of three nad(p)+ isozymes of l-glutamate dehydrogenase of chlamydomonas reinhardtii. *Biochim. Biophys. Acta*, 1119 :63–68, 1992.

[26] E. Myers. A four-russian algorithm for regular expression pattern matching. *J. of the ACM*, 39(2) :430–448, 1991.

[27] Peter Naur. Revisited report on the algorithmic language algol 60. *Communication of the ACM*, 3(5) :299–314, 1960.

[28] G. Navarro and M. Raffinot. Fast and simple character classes and bounded gaps pattern matching, with application to protein searching.

[29] J. H. Park, P. J. Schofield, and M. R. Edwards. Giardia intestinalis : characterization of a nadp-dependent glutamate dehydrogenase. *Exp. Parasitol.*, 88 :131–138, 1998.

[30] P. Rice, I. Longden, and A. Bleasby. Emboss : The european molecular biology open software suite. *Trends in Genetics*, 16 :276–277, 2000.

[31] I. Rigoutsos and A. Floratos. Combinatorial pattern discovery in biological sequences : the teiresias algorithm. *Bioinformatics*, 1997.

[32] I. Rigoutsos, A. Floratos, and C. Ouzounis. Case studies in pattern discovery without alignment results using the teiresias algorithm. Technical report, IBM Research, 97.

[33] Stothard. The sequence manipulation suite : Javascript programs for analyzing and formatting protein and dna sequences. *BioTechniques*, 28 :1102–1104, 2000.

[34] K. Thompson. Regular expression search algorithm. *CACM*, 11(6) :419–422, 1968.

[35] A. J. van Laere. Purification and properties of nad-dependent glutamate dehydrogenase from phycomyces spores. *J. Gen. Microbiol.*, 134 :1597–1601, 1988.

[36] J. T. L. Wang, B. A. Shapiro, and D. Shasha. *Pattern discovery in biomolecular data : tools, techniques and applications.* Oxford University Press, 1999.

[37] B. Watson. *taxonomies and toolkits of regular language algorithm.* PhD thesis, Eindhoven University of Technology, Netherlands, 1995.

[38] Cathy H. Wu, Lai-Su L. Yeh, Hongzhan Huang, Leslie Arminski, Jorge Castro-Alvear, Yongxing Chen, Zhang-Zhi Hu, Robert S. Ledley, Panagiotis Kourtesis, Baris E. Suzek, C. R. Vinayaka, Jian Zhang, and Winona C. Barker. The protein information resource (pir). *Nucleic Acids Research*, 31 :345–347, 2003.

[39] S. Wu and U. Manber. Fast text searching allowing errors. *CACM*, 35(10) :83–91, 1992.