Numéro d'ordre : **2009-ISAL-0059** Année **2009**

Thèse :

Formalisme statistique pour ensembles de structures discrètes

Présentée par
Sébastien Rebecchi

Devant l'
Institut National des Sciences Appliquées (INSA) de Lyon

Pour obtenir le grade de
Docteur de l'Université de Lyon

Spécialité :
Informatique

École doctorale :
InfoMaths

Soutenue publiquement le
23 septembre 2009

Devant le jury composé des membres suivants :

Président	**Pr. Éric Fleury**	LIP	ENS de Lyon
Rapporteurs	**Pr. Luc Brun**	GREYC	ENSICAEN
	Pr. Thierry Lecroq	LITIS	Université de Rouen
Examinateurs	**Pr. Isabelle Bloch**	LTCI	ENST
	Pr. Colin de la Higuera	LINA	Université de Nantes
Directeur	**Pr. Jean-Michel Jolion**	LIRIS	INSA de Lyon

Laboratoire :
Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS)

Remerciements

Je remercie l'ensemble des personnes ayant contribué au bon déroulement de mon parcours doctoral, et plus particulièrement :

- Jean-Michel Jolion, pour m'avoir accordé sa confiance et m'avoir fait profiter de son expérience de la recherche ;
- Luc Brun et Thierry Lecroq, pour avoir accepté de rapporter ma thèse ;
- Isabelle Bloch, Éric Fleury et Colin de la Higuera, pour avoir accepté de participer au jury de soutenance de ma thèse, devant lequel c'est un honneur pour moi de présenter mes travaux ;
- Mario Vento, pour avoir accepté de m'accueillir au sein du laboratoire MIVIA pour un séjour durant lequel j'ai beaucoup appris ;
- l'ensemble des membres des laboratoires LIRIS et MIVIA avec qui j'ai passé de bons moments.

Résumé

Formalisme statistique pour ensembles de structures discrètes

En reconnaissance de formes, le codage de l'information extraite des données est une étape décisive, et l'utilisation de structures semble être le choix le plus pertinent, leur puissance de représentation semblant illimitée. Cependant, le codage sous forme de vecteurs de caractéristiques numériques offre l'avantage de permettre, par la suite, l'utilisation de nombreux algorithmes efficaces développés spécifiquement pour la classification de vecteurs numériques dans des disciplines connexes à la reconnaissance de formes (apprentissage automatique, inférence statistique...).

De ce constat est né un champ de recherche dédié à la caractérisation statistique des espaces de structures. Les travaux les plus notables dans cette catégorie sont ceux basés sur la topologie induite par la distance d'édition. Cette approche se voit cependant assujettie à des problèmes de complexité qui pourraient théoriquement s'avérer insurmontables dans le paradigme calculatoire actuel. Parallèlement, ont vu le jour un ensemble des travaux basés sur une transformation des structures sous forme de vecteurs numériques. Le problème des méthodes proposés jusqu'à présent dans cette philosophie est le manque de pouvoir caractéristique, dans l'espace structurel initial, des informations extraites dans l'espace vectoriel. Enfin, les probabilités sont un paradigme largement utilisé pour la classification de données structurées, via la modélisation de distribution de structures au moyen de machines à états, ou autres graphes aléatoires, et l'utilisation massive du classifieur par maximisation de vraisemblance.

Dans cette thèse, outre le passage en revue des méthodes précitées, nous nous concentrons sur le traitement probabiliste d'ensembles de structures discrètes. Nous proposons plus particulièrement la traduction aux espaces structurels de critères permettant de définir les notions statistiques d'uniformité et de normalité de lois de probabilités. Nous proposons également une réflexion sur la définition de variables aléatoires de structures à valeurs dans un espace vectoriel, avec pour perspective la possibilité d'application, dans le domaine structurel, du théorème central limite, résultat d'importance fondamentale en théorie des probabilités et statistique.

D'un point de vue applicatif, nous évaluons les apports d'une partie de nos travaux pour la résolution de problèmes typiques en reconnaissance de formes, à savoir la classification de séquences d'ADN et la classification d'images de chiffres dessinés à la main. Nous utilisons le classifieur par maximisation de vraisemblance, en estimant la distribution de chaque classe par une loi normale de chaînes, telle que définie dans la partie théorique de cette thèse. Une conclusion négative tirée des expérimentations est notre manque de compétitivité vis-à-vis des méthodes les plus performantes sur chaque problème, notamment celles profitant de l'apport de connaissance experte biologique dans le cas des séquences d'ADN. Cependant, ce point nous donne des idées de perspectives de travail visant à améliorer notre classifieur, comme par exemple le fait de se concentrer plus sérieusement sur la phase d'apprentissage de la fonction de coût, trop simpliste dans cette thèse. Pour ce qui est du point positif, nous montrons qu'il est possible d'améliorer les résultats obtenus par les classifieurs basés sur la distance d'édition, très utilisée en reconnaissance de formes structurelle. Notre classifieur obtient des résultats honorables même lorsque la fonction de coût n'est pas très appropriée au problème, ce grâce à l'apport du cadre probabiliste qui permet de se détacher en partie de l'influence de cette fonction. Ce n'est pas le cas pour le classifieur à la plus proche médiane et le classifieur aux k plus proches voisins, qui échouent fortement lorsqu'ils sont basés sur la distance d'édition selon la même fonction de coût.

Mots clef Structure discrète, chaîne, multiensemble, arbre, graphe, fonction de coût, distance d'édition, statistique, moyenne, médiane, écart-type, vecteur, mesure, probabilité, distribution uniforme, distribution normale, théorème central limite, reconnaissance de formes, classification.

Abstract

Statistical formalism for sets of discrete structures

In pattern recognition, the coding of information extracted from the data is a decisive phase, and the use of structures seems to be the most pertinent choice, since their representative power seems to be unlimited. However, the coding in the form of numeric feature vectors offers the advantage of enabling, in the sequel, the use of numerous efficient algorithms specifically developed for the classification of numeric vectors in fields connected to pattern recognition (machine learning, statistical inference ...).

From this observation was born a research field devoted to the statistical characterization of structure spaces. The most notable works in this category are the ones based on the topology induced by the edit distance. This approach is yet subject to complexity problems that could theoretically be unsolvable within the current computational paradigm. At the same time have been developed a set of works based on a transformation of structures into numeric vectors. The problem of the methods proposed as yet within this philosophy is the lack of characteristic power, in the initial structure space, of information extracted in the vector space. Finally, probabilities are a widely used paradigm for the classification of structured data, via the modelling of distributions of structures by the way of state machines, or other random graphs, and a massive use of the maximum likelihood classifier.

In this thesis, in addition to reviewing the precited methods, we concentrate on the probabilistic processing of sets of structures. We propose more particularly the translation to structure spaces of criteria enabling to define the statistical notions of uniformity and normality of probability laws. We propose as well a reflection on the definition of structural random variables taking their values in a vector space, having in prospect the possibility of applying, in the structural domain, the central limit theorem, a fundamentally important result in probability theory and statistics.

From an applicative point of view, we evaluate the contribution of a part of our work for the resolution of typical problems in pattern recognition, namely the classification of DNA sequences and the classification of images of handwritten digits. We use the maximum likelihood classifier, estimating the distribution of each class by a Gaussian distribution, as defined in the theoretical part of this thesis. A negative conclusion drawn from these experimentations is our lack of competitiveness regarding the most reliable methods on each problem, in particular the ones that take advantage of biological expert knowledge in the case of DNA sequences. However, this point gives us insights for a future work aiming at improving our classifier, such as the need to concentrate more seriously on the cost function learning phase, that is too simplistic in this thesis. As for the positive point, we show that it is possible to improve the results obtained by the classifiers based on the edit distance, widely used in structural pattern recognition. Our classifier obtains honourable results even when the cost function is not well adapted to the problem, this thanks to the contribution of the probabilistic framework that reduces the influence of this function. This is not the case for the nearest median classifier and the k nearest neighbors classifier, both of them hardly failing when being based on the edit distance with respect to the same cost function.

Keywords *Discrete structure, string, multiset, tree, graph, cost function, edit distance, statistic, mean, median, standard deviation, vector, measure, probability, uniform distribution, normal distribution, central limit theorem, pattern recognition, classification.*

Table des matières

Liste des définitions

Liste des algorithmes

Table des figures

Liste des tableaux

Chapitre 1

Introduction

Le domaine d'application de cette thèse est la reconnaissance de formes [Bis06]. Nous nous concentrons plus particulièrement à contribuer au champ de recherche visant à combiner les approches statistiques et structurelles de cette discipline.

1.1 La reconnaissance de formes

L'être humain dispose d'aptitudes très avancées d'identification de signaux de l'environnement. Il est capable de reconnaître un visage, comprendre un discours oral, comprendre un texte écrit à la main ou imprimé, savoir si de la nourriture est fraîche ou périmée en fonction de son odeur etc., cela de manière quasi indépendante des conditions dans lesquelles ont lieu les perceptions des signaux (angle de vue, luminosité, contraste, papier froissé, bruit de fond...). Ces capacités lui permettent de prendre constamment des décisions influençant son évolution dans son environnement. La reconnaissance de formes est la branche de l'intelligence artificielle visant à l'automatisation de ces capacités. Elle consiste donc en l'étude de la manière dont les machines peuvent percevoir leur environnement, en distinguer les signaux ayant de l'intérêt, et prendre des décisions intelligentes sur la catégorie de ces signaux.

Les applications de la reconnaissance de formes sont multiples : bio-informatique (analyse de séquences d'ADN), biométrie (identification de personnes par leur visage, leur voix ou leurs empreintes digitales), médecine (diagnostic assisté par ordinateur), reconnaissance optique de caractères (classification de documents numérisés, tri de lettres par code postal) etc.

Watanabe [Wat85] a défini une *forme* comme « une entité, vaguement définie, à laquelle peut être donné un nom ». Une forme peut donc être une empreinte digitale, un visage, un discours oral, un mot écrit à la main, une séquence d'ADN etc. Une *classe* est un ensemble de formes ayant généralement des caractéristiques et une origine communes, et un *classifieur* est une machine effectuant un processus de *classification*, c'est-à-dire l'affectation d'une forme à une classe. Lorsque les classes sont définies à l'avance, la classification est dite *supervisée*, et lorsque les classes sont apprises en se basant sur la similarité des formes dont l'on dispose, on parle de classification *non supervisée*.

La conception d'un système standard de reconnaissance de formes peut être résumée par l'élaboration des trois étapes suivantes :

1. collecte des données ;
2. représentation des formes ;
3. classification des formes.

L'étape de collecte des données consiste simplement en l'enregistrement de signaux de l'environnement par des capteurs dont la nature (appareil photographique, caméra, microphone, numériseur...) dépend du problème de reconnaissance considéré. La représentation des formes joue un rôle d'importance fondamentale, car c'est lors de cette étape que sont définis, extraits et organisés

les attributs considérés essentiels pour caractériser une forme, et discriminants pour la classifier. De plus, le choix du classifieur est évidemment dépendant du type de représentation choisi, et il existe deux approches majeures dans la littérature, définissant les sous-domaines que sont la reconnaissance de formes *statistique* et la reconnaissance de formes *structurelle*.

1.2 Approche statistique

En reconnaissance de formes statistique [JDM00], une forme est caractérisée par n attributs numériques extraits des données, et représentée par un vecteur d'un espace numérique de dimension n. La construction du classifieur consiste donc en la partition de l'espace des attributs en différentes régions, chacune affectée à une classe.

Toute la difficulté de cette approche réside dans le choix des attributs : de bons attributs doivent permettre de discriminer le plus possible les classes, c'est-à-dire représenter toutes les formes d'une même classe dans une région la plus compacte possible de l'espace, et situer chaque classe dans une région la plus éloignée possible des régions où se situent chacune des autres classes. Cependant certains types de formes sont difficilement descriptibles par le biais de vecteurs d'attributs numériques, rendant leur extraction inefficace.

Par conséquent, cette approche accentue plutôt la phase de classification, la construction d'un classifieur de vecteurs numériques fiable et robuste pouvant être réalisée à l'aide d'un grand nombre d'algorithmes issus de domaines de recherche connexes à la reconnaissance de formes, tels que l'apprentissage automatique (réseau de neurones artificiel [JMM96], machines à vecteurs de support [Vap98], arbre de décision [LBS84, Qui93], *k-means* [Ste56, JMF99]...), l'inférence statistique (estimation de probabilité, décomposition de mélange, règles de Bayes...) etc.

1.3 Approche structurelle

En reconnaissance de formes structurelle [BS90], les formes sont représentées par des structures complexes composées de primitives simples et de relations topologiques entre les primitives. Les primitives, et éventuellement les relations, sont caractérisées par des attributs symboliques extraits des données.

L'avantage de cette approche réside dans l'expressivité de la représentation des formes, qui fournit une description sur la manière dont elles sont construites à partir des primitives. De plus, le nombre de primitives et de relations ne nécessite pas d'être spécifié à l'avance et peut être variable d'une forme à l'autre. Ce qui est l'avantage principal de cette approche peut cependant se révéler être un inconvénient dans certaines situations où il est difficile de choisir a priori selon quel schéma relationnel (séquentiel, hiérarchique...) ou dans quel ordre doivent être arrangées les primitives.

Un système de reconnaissance structurel standard est limité a deux types de classifieurs, à savoir :

- les classifieurs basés sur le ou les meilleurs appariements (isomorphisme, distance d'édition [Lev66, WF74]...) à un ou plusieurs modèles, où un modèle est une forme supposée représenter la classe à laquelle elle appartient ;
- les classifieurs basés sur l'acceptation par une grammaire formelle, qui est un modèle issu de la théorie des langages formels [Har78], selon laquelle une structure est vue comme une phrase, c'est-à-dire une construction respectant les règles syntaxiques d'un langage. Une grammaire représente un langage formel, c'est-à-dire un ensemble de structures respectant certaines règles syntaxiques.

Dans tous les cas, la classification ne peut être que supervisée : il est nécessaire de disposer soit d'au moins un modèle par classe pour l'appariement, soit d'un échantillon de chaque classe pour inférer une grammaire pour chacune d'entre elles. Un autre inconvénient est l'absence de considération statistique, qui rend difficile voire impossible la gestion du bruit ou de la distorsion, résultant principalement d'instabilités pouvant survenir durant le processus d'extraction des primitives et/ou des relations. Enfin, les méthodes de cette approche peuvent être soumises à une explosion combinatoire du nombre de solutions possibles à tester, et s'avérer donc trop coûteuses en termes de temps pour un grand nombre d'applications.

1.4 Vers une combinaison des approches statistiques et structurelles

Nous venons de voir que les deux approches principales de la reconnaissance de formes sont complémentaires. Une approche combinée, qui garderait les avantages des deux tout en supprimant leurs inconvénients, pourrait se révéler efficace pour la plupart des problèmes de reconnaissance.

D'après la discussion précédente, il s'agirait en effet d'améliorer les méthodes structurelles en y incluant des possibilités de gestion du bruit ou des erreurs survenant lors de l'extraction des structures, de manière à rendre plus robuste le processus de classification. Tsai [Tsa90] a dégagé plusieurs manières d'aborder cette problématique, plus ou moins compatibles entre elles, et qui peuvent être regroupées en deux catégories :

1. combinaison au niveau de la représentation des formes : inclure de l'information statistique dans les attributs des primitives et/ou des relations des structures, c'est-à-dire de l'information extraite des données ;
2. combinaison au niveau de la classification des formes : inclure de l'information statistique dans le classifieur, c'est-à-dire de l'information apprise à partir des structures.

Les méthodes de la première catégorie sont dépendantes du processus d'extraction des formes, et on pourrait donc presque imaginer la mise au point d'une méthode spécifique à chaque application visée. Nous nous intéressons à contribuer à la deuxième catégorie dans cette thèse, c'est-à-dire à savoir comment inférer de l'information statistique à partir d'un ensemble de structures appartenant à une même classe.

1.5 Organisation de la thèse

Dans le chapitre suivant, nous allons définir les structures de données utilisées tout au long de cette thèse, avec le vocabulaire et les notations y afférant. Puis, le chapitre 3 est dédié à l'étude des diverses approches définies dans la littérature pour caractériser statistiquement les ensembles de structures discrètes. Nous allons ensuite passer en revue les principaux modèles utilisés en reconnaissance de formes pour représenter et estimer des distributions de structures (chapitre 4). Le cœur et la plus grande partie de notre travail se situent dans les chapitres 5 et 6, où nous proposons des définitions de distributions de structures vérifiant respectivement des critères statistiques d'uniformité et de normalité. Enfin, avant de tirer des conclusions et résumer les perspectives soulevées par cette thèse (chapitre 8), nous évaluerons l'apport de notre travail via des expérimentations de classification de bases de données issues de l'environnement réel (chapitre 7).

Chapitre 2

Structures

Ce chapitre est dédié aux définitions des structures utilisées dans la suite de cette thèse, ainsi qu'à l'introduction des notations et du vocabulaire associés.

De manière générale, une structure est un objet complexe composé d'éléments simples prenant leur valeur dans un ou plusieurs ensembles nommés *alphabets*. Une structure est dite *discrète* dès lors que le ou les alphabets sur lesquels elle est définie sont dénombrables, ce qui sera toujours notre cas.

2.1 Alphabet

Définition 1 (Alphabet)
Un alphabet est un ensemble fini non vide dont les éléments sont nommés lettres.

La notion de lettre est à comprendre au sens large, et nous l'utilisons aussi bien pour désigner un élément de type symbolique que numérique.

Exemple
$\{a, b, c\}$ est un alphabet.

Soit A un alphabet, auquel nous associons un objet spécial, noté λ, nommé *lettre vide*, et qui n'est pas élément de A. Les lettres de A sont utilisées en tant que composantes simples (primitives), qui, associées d'une manière ou d'une autre en fonction du contexte souhaité, nous servent de base à la définition de structures plus complexes. λ est utilisé pour représenter explicitement le vide implicite pouvant être pris en compte lors d'opérations sur de telles structures.

2.2 Multiensemble

Le concept de multiensemble « étend » celui d'ensemble, en ce sens où il permet de formaliser la présence *multiple* d'un élément.

Définition 2 (Multiensemble)
Un multiensemble (étiqueté) sur A est une fonction $A \to \mathbb{N}$.

Soient M un multiensemble sur A, et $l \in A$. $M(l)$ est nommé *multiplicité* de l dans M. De plus, nous optons pour une notation simplifiée, semblable à celle des ensembles (l'ordre n'a aucune importance), mais à base de crochets [i], et à partir de laquelle les multiplicités des lettres de A sont implicitement déduites.

i. Il est d'usage d'utiliser des doubles accolades $\{\!\{ \ldots \}\!\}$, mais nous optons pour le choix des crochets pour une plus grande clarté dans le cas de notations d'ensembles de multiensembles.

FIGURE 2.1 – Une représentation graphique du multiensemble exemple de la définition 2.

Exemple
Soit $A = \{a, b, c\}$. $M = \{(a, 0), (b, 5), (c, 2)\} =_{\text{notation}} [b, b, b, b, b, c, c]$ est un multiensemble sur A, avec $M(a) = 0$, $M(b) = 5$, et $M(c) = 2$. Une représentation graphique possible de M est proposée en figure 2.1.

Remarque
$[]$ est un multiensemble sur A, nommé *multiensemble vide*, l'unique dans lequel toute lettre de A est de multiplicité nulle.

Définition 3 (Taille d'un multiensemble)
Soit M un multiensemble sur A. La taille de M, notée $|M|$, est égale à la somme des multiplicités dans M des lettres de A :

$$|M| = \sum_{l \in A} M(l).$$

La taille d'un multiensemble est également nommée *multicardinal*.

Exemple
Soient $A = \{a, b, c\}$, et $M = [b, b, b, b, b, c, c]$. Nous avons $|M| = 7$.

Soit $n \in \mathbb{N}$. Notons :
- $\mathbb{M}(A)^n$ l'ensemble des multiensembles sur A de taille n ;
- $\mathbb{M}(A)^{\leqslant n}$ l'ensemble des multiensembles sur A de taille au plus n ;
- $\mathbb{M}(A)^*$ l'ensemble des multiensembles sur A.

Dès lors, nous avons :
- $\mathbb{M}(A)^0 = \{[]\}$;
- $\mathbb{M}(A)^{\leqslant n} = \bigcup_{i=0}^{n} \mathbb{M}(A)^i$;
- $\mathbb{M}(A)^* = \bigcup_{i \in \mathbb{N}} \mathbb{M}(A)^i$.

Exemple
Soit $A = \{a, b, c\}$. Nous avons $\mathbb{M}(A)^0 = \{[]\}$, $\mathbb{M}(A)^1 = \{[a], [b], [c]\}$, $\mathbb{M}(A)^2 = \{[a, a], [b, b], [c, c], [a, b], [a, c], [b, c]\}\ldots$

Remarque
Le nombre de multiensembles sur A de taille n est égal au nombre de r-combinaisons de taille n de A (cf. section B.3) :

$$|\mathbb{M}(A)^n| = \text{r-C}^n_{|A|} = \frac{(|A| + n - 1)!}{n! \times (|A| - 1)!}.$$

De manière similaire aux ensembles, nous définissons un ensemble d'opérations binaires agissant sur les multiensembles :

Définition 4 (Opérations binaires multiensemblistes)
Soient $M, N, P \in \mathbb{M}(A)^*$:
- appartenance : $\forall l \in A$:

$$l \in M \iff M(l) \neq 0;$$

- inclusion :

$$M \subseteq N \iff \forall l \in A, M(l) \leqslant N(l),$$
$$M \subset N \iff (M \subseteq N) \wedge (M \neq N);$$

– intersection :
$$(M \cap N) = P \iff \forall l \in A, P(l) = \min\{M(l), N(l)\};$$

– union :
$$(M \cup N) = P \iff \forall l \in A, P(l) = \max\{M(l), N(l)\}.$$

Exemple
Soient $A = \{a, b, c\}$, $M = [a, b, b, c]$, et $N = [a, a, b]$. Nous avons $a \in M$, $c \notin N$, $M \not\subseteq N$, $N \not\subseteq M$, $M \cap N = [a, b]$, et $M \cup N = [a, a, b, b, c]$.

Enfin, nous définissons l'opération d'union-addition, qui produit un nouveau multiensemble à partir d'un multiensemble initial, dans lequel y est inséré une lettre de A, ou un autre multiensemble sur A, et telle que λ est élément neutre de cette insertion.

Définition 5 (Union-addition de multiensembles)
L'union-addition sur A est l'opération binaire $\uplus\colon \mathbb{M}(A)^* \times (\mathbb{M}(A)^* \cup A \cup \{\lambda\}) \to \mathbb{M}(A)^*$ telle que : $\forall M, N, P \in \mathbb{M}(A)^*, \forall l \in A$:

$$\begin{aligned} M \uplus \lambda &= M, \\ M \uplus l &= M \uplus [l], \\ (M \uplus N) = P &\iff \forall m \in A, P(m) = M(m) + N(m). \end{aligned}$$

Comme nous le verrons avec la définition 9, l'union-addition est au multiensemble ce que la concaténation est à la chaîne.

Exemple
Soient $A = \{a, b, c\}$, $l = b$, $M = [a, b, c]$, et $N = [a, b]$. Nous avons $M \uplus l = [a, b, b, c]$, et $M \uplus N = [a, a, b, b, c]$.

Remarque
– $|M \uplus l| = |M| + 1$;
– $|M \uplus N| = |M| + |N|$;
– $[] \uplus M = M \uplus [] = M$.

Grâce à la définition 5, nous pouvons « promouvoir » une lettre de $A \cup \{\lambda\}$ en un multiensemble de $\mathbb{M}(A)^{\leqslant 1}$, simplement en l'union-additionnant à $[]$.

Définition 6 (Promotion en multiensemble)
La promotion en multiensemble sur A est la bijection définie comme suit :

$$\begin{aligned} A \cup \{\lambda\} &\to \mathbb{M}(A)^{\leqslant 1} \\ l &\to [] \uplus l. \end{aligned}$$

De plus, nous disons que l est le *promu* de $[] \uplus l$. Notons que la lettre vide est promue en le multiensemble vide.

2.3 Chaîne

Le concept de chaîne est l'alternative « ordonnée » à celui de multiensemble, en ce sens où il permet de formaliser la prise en compte d'un ordre explicite et discriminant sur les composantes.

Définition 7 (Chaîne)
Soit $n \in \mathbb{N}$. Une chaîne (étiquetée) sur A, de taille n, est une fonction $\{1, \ldots, n\} \to A$.

L'ensemble de départ $\{1, \ldots, n\}$ d'une telle chaîne X est nommé *ensemble des positions* de X, et sa taille est également nommée *longueur*, et notée $|X|$. De plus, nous optons pour une notation simplifiée, où l'ordre des lettres a de l'importance, et à partir de laquelle l'ensemble des positions est implicitement déduit de cet ordre.

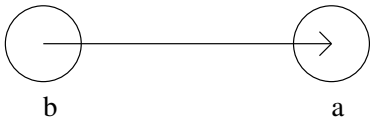

FIGURE 2.2 – Une représentation graphique de la chaîne exemple de la définition 7.

Exemple
Soit $A = \{a, b, c\}$. $X = \{(1, b), (2, a)\} =_{\text{notation}} ba$ est une chaîne sur A de taille 2. Une représentation graphique possible de X est proposée en figure 2.2.

Remarque
$\{\}$ est l'unique chaîne sur A de taille nulle, nommée *chaîne vide*.

Soit $n \in \mathbb{N}$. Notons :
- $\mathbb{S}(A)^n$ l'ensemble des chaînes sur A de taille n ;
- $\mathbb{S}(A)^{\leqslant n}$ l'ensemble des chaînes sur A de taille au plus n ;
- $\mathbb{S}(A)^*$ l'ensemble des chaînes sur A.

Dès lors, nous avons :
- $\mathbb{S}(A)^0 = \{\{\}\}$;
- $\mathbb{S}(A)^{\leqslant n} = \bigcup_{i=0}^{n} \mathbb{S}(A)^i$;
- $\mathbb{S}(A)^* = \bigcup_{i \in \mathbb{N}} \mathbb{S}(A)^i$.

Exemple
Soit $A = \{a, b, c\}$. Nous avons $\mathbb{S}(A)^0 = \{\{\}\}$, $\mathbb{S}(A)^1 = \{a, b, c\}$, $\mathbb{S}(A)^2 = \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}\ldots$

Remarque
Le nombre de chaînes sur A de taille n est égal au nombre de r-arrangements de taille n de A (cf. section B.2) :

$$|\mathbb{S}(A)^n| = \text{r-A}^n_{|A|} = |A|^n.$$

Définition 8 (Sous-séquence, facteur, préfixe, suffixe d'une chaîne)
Une chaîne X' sur A est une sous-séquence d'une chaîne X sur A ssi il existe une injection $f\colon \{1, \ldots, |X'|\} \to \{1, \ldots, |X|\}$ telle que :
- $\forall i \in \{1, \ldots, |X'| - 1\}, f(i + 1) > f(i)$;
- $\forall i \in \{1, \ldots, |X'|\}, X'(i) = X(f(i))$.

Si de plus $\forall i \in \{1, \ldots, |X'| - 1\}, f(i + 1) = f(i) + 1$, alors X' est un facteur de X. Et si de plus $X' = \{\}$ ou $f(1) = 1$ (resp. $X' = \{\}$ ou $f(|X'|) = |X|$), alors X' est un préfixe (resp. suffixe) de X, l'unique de taille $|X'|$.

Nous n'utilisons pas le terme bien connu « sous-chaîne », car nous lui préférons le terme « facteur », cela pour éviter la possible confusion entre « sous-séquence » et « sous-chaîne ».

Exemple
Soient $A = \{a, b, c\}$, $X = abcc$, et $X' = ab$. X' est une sous-séquence de X, avec $f(1) = 1$ et $f(2) = 2$; X' est également un facteur de X, car $f(2) = f(1) + 1$; enfin X' est aussi le préfixe de taille 2 de X, car $f(1) = 1$.

Remarque
- $\{\}$ est sous-séquence, facteur, préfixe, et suffixe de toute chaîne ;
- l'unique sous-séquence de $\{\}$ est $\{\}$.

Nous définissons maintenant l'opération de concaténation, qui produit une nouvelle chaîne à partir d'une chaîne initiale, dans laquelle y est insérée une lettre de A, ou une autre chaîne sur A, et telle que λ est élément neutre de cette insertion.

Définition 9 (Concaténation de chaîne)
La concaténation sur A est l'opération binaire $.: \mathbb{S}(A)^* \times (\mathbb{S}(A)^* \cup A \cup \{\lambda\}) \to \mathbb{S}(A)^*$ telle que : $\forall X, Y \in \mathbb{S}(A)^*, \forall l \in A$:

$$\begin{aligned} X.\lambda &= X, \\ X.l &= X \cup \{(|X|+1, l)\}, \\ X.Y &= X \cup \bigcup_{i=1}^{|Y|} \{(|X|+i, Y(i))\}. \end{aligned}$$

La concaténation est à la chaîne ce que l'union-addition (définition 5) est au multiensemble.

Exemple
Soient $A = \{a, b, c\}$, $l = b$, $X = abc$, et $Y = ab$. Nous avons $X.l = abcb$, et $X.Y = abcab$.

Remarque
- $|X.l| = |X| + 1$;
- $|X.Y| = |X| + |Y|$;
- $\{\}.X = X.\{\} = X$.

Grâce à la définition 9, nous pouvons « promouvoir » une lettre de $A \cup \{\lambda\}$ en une chaîne de $\mathbb{S}(A)^{\leqslant 1}$, simplement en la concaténant à $\{\}$.

Définition 10 (Promotion en chaîne)
La promotion en chaîne sur A est la bijection définie comme suit :

$$\begin{aligned} A \cup \{\lambda\} &\to \mathbb{S}(A)^{\leqslant 1} \\ l &\to \{\}.l. \end{aligned}$$

De plus, nous disons que l est le *promu* de $\{\}.l$. Notons que la lettre vide est promue en la chaîne vide.

2.4 Arbre

Le concept d'arbre « étend » ceux de chaîne ou multiensemble, en fonction de la considération ou non d'un ordre sur sa composition. Dans tous les cas, il permet de formaliser la notion de *hiérarchie* entre les composantes. Nous parlons simplement d'arbre lorsque la précision du caractère ordonné ou non ordonné n'est pas importante.

2.4.1 Cas non ordonné : l'u-arbre

Définition 11 (U-arbre)
Soient $a, d \in \mathbb{N}$. Un u-arbre [ii], ou arbre non ordonné, U, (étiqueté) sur A, d'arité a et profondeur d, est un couple $(r(U), c(U))$ coïncidant avec la définition récursive suivante :
- $(\lambda, [])$ est l'u-arbre vide sur A, c'est-à-dire l'unique u-arbre sur A d'arité a et profondeur 0 ;
- soit $r \in A$. $(r, [])$ est une u-feuille sur A, c'est-à-dire un u-arbre sur A d'arité a et profondeur 1 ;
- soient $r \in A$, et c un multiensemble non vide, de taille au plus a, sur l'alphabet des u-arbres non vides sur A d'arité a et profondeur au plus d. (r, c) est un u-arbre sur A d'arité a, et de profondeur égale à :
$$1 + \max\{\text{profondeur de } C | C \in c\}.$$

Soit U un u-arbre sur A. $r(U)$ est nommé *racine*, ou *parent*, de U, et les arbres appartenant à $c(U)$ sont nommés *arbre enfant*, ou *sous-arbre*, de U. Enfin, notons $d(U)$ la profondeur de U.

Exemple
Soit $A = \{a, b, c\}$. $U = (a, [(b, []), (a, [])])$ est un u-arbre sur A, avec $r(U) = a$, $c(U) = [(b, []), (a, [])]$, et $d(U) = 2$. U est d'arité 2, 3, 4... mais U n'est pas d'arité 0 ou 1. Une représentation graphique possible de U est proposée en figure 2.3.

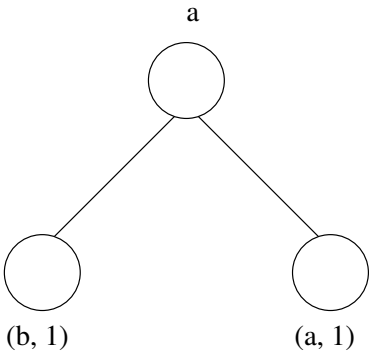

FIGURE 2.3 – Une représentation graphique de l'u-arbre exemple de la définition 11.

Remarque
Si U est d'arité a, alors U est d'arité $a+1$.

Soient $a, d \in \mathbb{N}$. Notons :
- $\mathbb{U}(A)_d^{\leqslant a}$ l'ensemble des u-arbres sur A d'arité a et profondeur d ;
- $\mathbb{U}(A)_{\leqslant d}^{\leqslant a}$ l'ensemble des u-arbres sur A d'arité a et profondeur au plus d ;
- $\mathbb{U}(A)_*^{\leqslant a}$ l'ensemble des u-arbres sur A d'arité a ;
- $\mathbb{U}(A)_d^*$ l'ensemble des u-arbres sur A de profondeur d ;
- $\mathbb{U}(A)_{\leqslant d}^*$ l'ensemble des u-arbres sur A de profondeur au plus d ;
- $\mathbb{U}(A)_*^*$ l'ensemble des u-arbres sur A.

Dès lors, nous avons :
- $\forall j \in \mathbb{N}, \mathbb{U}(A)_0^{\leqslant j} = \{(\lambda, [])\}$;
- $\forall i \in \mathbb{N} \setminus \{0,1\}, \mathbb{U}(A)_i^{\leqslant 0} = \{\}$;
- $\mathbb{U}(A)_d^{\leqslant a} = \bigcup_{j=0}^{a} \mathbb{U}(A)_d^{\leqslant j}$;
- $\mathbb{U}(A)_{\leqslant d}^{\leqslant a} = \bigcup_{i=0}^{d} \bigcup_{j=0}^{a} \mathbb{U}(A)_i^{\leqslant j}$;
- $\mathbb{U}(A)_*^{\leqslant a} = \bigcup_{i\in\mathbb{N}} \bigcup_{j=0}^{a} \mathbb{U}(A)_i^{\leqslant j}$;
- $\mathbb{U}(A)_d^* = \bigcup_{j\in\mathbb{N}} \mathbb{U}(A)_d^{\leqslant j}$;
- $\mathbb{U}(A)_{\leqslant d}^* = \bigcup_{i=0}^{d} \bigcup_{j\in\mathbb{N}} \mathbb{U}(A)_i^{\leqslant j}$;
- $\mathbb{U}(A)_*^* = \bigcup_{i\in\mathbb{N}} \bigcup_{j\in\mathbb{N}} \mathbb{U}(A)_i^{\leqslant j}$.

Exemple
Soit $A = \{a, b, c\}$. Nous avons $\mathbb{U}(A)_0^0 = \{(\lambda, [])\}$, $\mathbb{U}(A)_0^1 = \{(\lambda, [])\}$, $\mathbb{U}(A)_1^0 = \{(a, []), (b, []), (c, [])\}\ldots$

Définition 12 (Liste des nœuds, taille d'un u-arbre)
Soit U u-arbre sur A. La liste des nœuds $n(U)$ de U est le multiensemble sur A égal à :
- [] si U est vide ;
- sinon :

$$[r(U)] \uplus \biguplus_{C \in c(U)} \biguplus_{i=1}^{c(U)(C)} n(C).$$

Ainsi, la taille $|U|$ de U, c'est-à-dire son nombre de nœuds, est égale à $|n(U)|$.

Exemple
Soient $A = \{a, b, c\}$, et $U = (a, [(b, []), (a, [])])$. Nous avons $n(U) = [a, b, a]$, et $|U| = 3$.

Remarque
Si un u-arbre U est d'arité a, alors nous avons :

$$0 \leqslant |U| \leqslant \sum_{i=0}^{d(U)-1} a^i.$$

ii. « u » pour « *unordered* »

Définition 13 (U-arbre complet)
Soient $a, d \in \mathbb{N}$, et taille_max_u-arbre: $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ la fonction associant à (a, d) la taille maximale que peut avoir un u-arbre sur A d'arité a et profondeur d :

$$\text{taille\_max\_u-arbre}(a, d) = \sum_{i=0}^{d-1} a^i.$$

Un u-arbre U sur A est (a, d)-complet ssi ces trois conditions sont satisfaites :
- U est d'arité a ;
- $d(U) = d$;
- $|U| = \text{taille\_max\_u-arbre}(a, d)$.

Un u-arbre (a, d)-complet est l'un des plus « grands » u-arbres d'arité a et profondeur d, en ce sens où aucun autre de ces tels u-arbres ne peut avoir plus de nœuds que lui.

Exemple
Soient $A = \{a, b, c\}$, et $U = (a, [(b, []), (a, [])])$. U est $(2, 2)$-complet.

Soient $a, d \in \mathbb{N}$. Nous définissons maintenant l'opération d'u-enracinement, qui produit un u-arbre sur A d'arité a et profondeur au plus $d + 1$ à partir d'un couple composé d'une lettre de $A \cup \{\lambda\}$, et d'un multiensemble de taille au plus a sur l'alphabet des u-arbres non vides sur A d'arité a et profondeur au plus d. La définition 11 impose que λ ne peut pas avoir d'enfant, et par conséquent que l'u-enracinement résulte dans ce cas en l'u-arbre vide. Sinon, le couple constitue déjà un u-arbre en lui-même :

Définition 14 (Enracinement d'un u-arbre)
Soient $a, d \in \mathbb{N}$. L'u-enracinement sur A est l'opération binaire suivante :

$$\downarrow_d^a : A \cup \{\lambda\} \times \mathbb{M}(\mathbb{U}(A)_{\leqslant d}^{\leqslant a} \setminus \{(\lambda, [])\})^{\leqslant a} \to \mathbb{U}(A)_{\leqslant d+1}^{\leqslant a}$$

$$(l, M) \to l \downarrow_d^a M = \begin{cases} (l, M) & \text{si } l \neq \lambda, \\ (\lambda, []) & \text{sinon.} \end{cases}$$

Exemple
Soient $A = \{a, b, c\}$, $l = a$, et $M = [(b, []), (a, [])]$. Nous avons $l \downarrow_1^2 M = (a, [(b, []), (a, [])])$, et $\lambda \downarrow_1^2 M = (\lambda, [])$.

2.4.2 Cas ordonné : l'o-arbre

Définition 15 (O-arbre)
Soient $a, d \in \mathbb{N}$. Un o-arbre [iii], ou arbre ordonné, O, (étiqueté) sur A, d'arité a et profondeur d, est un couple $(r(O), c(O))$ coïncidant avec la définition récursive suivante :
- $(\lambda, \{\})$ est l'o-arbre vide sur A, c'est-à-dire l'unique o-arbre sur A d'arité a et profondeur 0 ;
- soit $r \in A$. $(r, \{\})$ est une o-feuille sur A, c'est-à-dire un o-arbre sur A d'arité a et profondeur 1 ;
- soient $r \in A$, et c une chaîne non vide, de taille au plus a, sur l'alphabet des o-arbres non vides sur A d'arité a et profondeur au plus d. (r, c) est un o-arbre sur A d'arité a, et de profondeur égale à :

$$1 + \max\{\text{profondeur de } c(i) | i \in \{1, \ldots, |c|\}\}.$$

Soit O un o-arbre sur A. $r(O)$ est nommé *racine*, ou *parent*, de O, et les arbres qui sont lettres de $c(O)$ sont nommés *arbre enfant*, ou *sous-arbre*, de O. Enfin, notons $d(O)$ la profondeur de O.

Exemple
Soit $A = \{a, b, c\}$. $O = (a, (b, \{\})(a, \{\}))$ est un o-arbre sur A, avec $r(O) = a$, $c(O) = (b, \{\})(a, \{\})$, et $d(O) = 2$. O est d'arité 2, 3, 4... mais O n'est pas d'arité 0 ou 1. Une représentation graphique possible de O est proposée en figure 2.4.

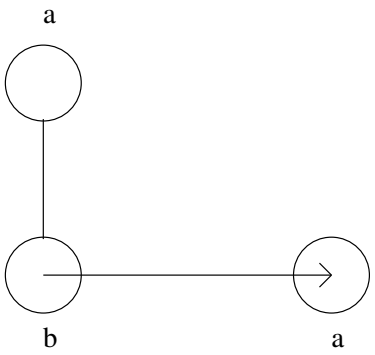

FIGURE 2.4 – Une représentation graphique de l'o-arbre exemple de la définition 15.

Remarque
Si O est d'arité a, alors O est d'arité $a+1$.

Soient $a, d \in \mathbb{N}$. Notons :
- $\mathbb{O}(A)_d^{\leqslant a}$ l'ensemble des o-arbres sur A d'arité a et profondeur d ;
- $\mathbb{O}(A)_{\leqslant d}^{\leqslant a}$ l'ensemble des o-arbres sur A d'arité a et profondeur au plus d ;
- $\mathbb{O}(A)_*^{\leqslant a}$ l'ensemble des o-arbres sur A d'arité a ;
- $\mathbb{O}(A)_d^*$ l'ensemble des o-arbres sur A de profondeur d ;
- $\mathbb{O}(A)_{\leqslant d}^*$ l'ensemble des o-arbres sur A de profondeur au plus d ;
- $\mathbb{O}(A)_*^*$ l'ensemble des o-arbres sur A.

Dès lors, nous avons :
- $\forall j \in \mathbb{N}, \mathbb{O}(A)_0^{\leqslant j} = \{(\lambda, \{\})\}$;
- $\forall i \in \mathbb{N} \setminus \{0,1\}, \mathbb{O}(A)_i^{\leqslant 0} = \{\}$;
- $\mathbb{O}(A)_d^{\leqslant a} = \bigcup_{j=0}^{a} \mathbb{O}(A)_d^{\leqslant j}$;
- $\mathbb{O}(A)_{\leqslant d}^{\leqslant a} = \bigcup_{i=0}^{d} \bigcup_{j=0}^{a} \mathbb{O}(A)_i^{\leqslant j}$;
- $\mathbb{O}(A)_*^{\leqslant a} = \bigcup_{i\in\mathbb{N}} \bigcup_{j=0}^{a} \mathbb{O}(A)_i^{\leqslant j}$;
- $\mathbb{O}(A)_d^* = \bigcup_{j\in\mathbb{N}} \mathbb{O}(A)_d^{\leqslant j}$;
- $\mathbb{O}(A)_{\leqslant d}^* = \bigcup_{i=0}^{d} \bigcup_{j\in\mathbb{N}} \mathbb{O}(A)_i^{\leqslant j}$;
- $\mathbb{O}(A)_*^* = \bigcup_{i\in\mathbb{N}} \bigcup_{j\in\mathbb{N}} \mathbb{O}(A)_i^{\leqslant j}$.

Exemple
Soit $A = \{a, b, c\}$. Nous avons $\mathbb{O}(A)_0^0 = \{(\lambda, \{\})\}$, $\mathbb{O}(A)_0^1 = \{(\lambda, \{\})\}$, $\mathbb{O}(A)_1^0 = \{(a, \{\}), (b, \{\}), (c, \{\})\}\ldots$

Définition 16 (Liste des nœuds, taille d'un o-arbre)
Soit O o-arbre sur A. La liste des nœuds $n(O)$ de O est le multiensemble sur A égal à :
- [] si O est vide ;
- sinon :

$$[r(O)] \uplus \biguplus_{i=1}^{|c(O)|} n(c(O)(i)).$$

Ainsi, la taille $|O|$ de O, c'est-à-dire son nombre de nœuds, est égale à $|n(O)|$.

Exemple
Soient $A = \{a, b, c\}$, et $O = (a, (b, \{\})(a, \{\}))$. Nous avons $n(O) = [a, b, a]$, et $|O| = 3$.

Remarque
Si un o-arbre O est d'arité a, alors nous avons :

$$0 \leqslant |O| \leqslant \sum_{i=0}^{d(O)-1} a^i.$$

iii. « o » pour « *ordered* »

Définition 17 (O-arbre complet)
Soient $a, d \in \mathbb{N}$, et taille_max_o-arbre: $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ la fonction associant à (a, d) la taille maximale que peut avoir un o-arbre sur A d'arité a et profondeur d :

$$\text{taille\_max\_o-arbre}(a, d) = \sum_{i=0}^{d-1} a^i.$$

Un o-arbre O sur A est (a, d)-complet ssi ces trois conditions sont satisfaites :
- O est d'arité a ;
- $d(O) = d$;
- $|O| = \text{taille\_max\_o-arbre}(a, d)$.

Un o-arbre (a, d)-complet est l'un des plus « grands » o-arbres d'arité a et profondeur d, en ce sens où aucun autre de ces tels o-arbres ne peut avoir plus de nœuds que lui.

Exemple
Soient $A = \{a, b, c\}$, et $O = (a, (b, \{\})(a, \{\}))$. O est $(2, 2)$-complet.

Soient $a, d \in \mathbb{N}$. Nous définissons maintenant l'opération d'o-enracinement, qui produit un o-arbre sur A d'arité a et profondeur au plus $d + 1$ à partir d'un couple composé d'une lettre de $A \cup \{\lambda\}$, et d'une chaîne de taille au plus a sur l'alphabet des o-arbres non vides sur A d'arité a et profondeur au plus d. La définition 15 impose que λ ne peut pas avoir d'enfant, et par conséquent que l'o-enracinement résulte dans ce cas en l'o-arbre vide. Sinon, le couple constitue déjà un o-arbre en lui-même :

Définition 18 (Enracinement d'un o-arbre)
Soient $a, d \in \mathbb{N}$. L'o-enracinement sur A est l'opération binaire suivante :

$$\downarrow_d^a : A \cup \{\lambda\} \times \mathbb{S}(\mathbb{O}(A)_{\leqslant d}^{\leqslant a} \setminus \{(\lambda, \{\})\})^{\leqslant a} \to \mathbb{O}(A)_{\leqslant d+1}^{\leqslant a}$$

$$(l, X) \to l \downarrow_d^a X = \begin{cases} (l, X) & \text{si } l \neq \lambda, \\ (\lambda, \{\}) & \text{sinon.} \end{cases}$$

Exemple
Soient $A = \{a, b, c\}$, $l = a$, et $X = (b, \{\})(a, \{\})$. Nous avons $l \downarrow_1^2 X = (a, (b, \{\})(a, \{\}))$, et $\lambda \downarrow_1^2 X = (\lambda, \{\})$.

2.5 Graphe

De manière générale, un graphe est défini sur 2 alphabets, l'un pour ses composantes nommées *sommet*, et l'autre pour ses composantes nommées *arête*, ces dernières permettant de modéliser les intra et inter relations associées aux sommets.

Le graphe est le concept structurel le plus puissant, en ce sens où aucune contrainte ou relation particulière n'est imposée sur les composantes, et les notions d'ordre, hiérarchie, ou autres, ne peuvent être que déduites de l'interprétation de l'utilisateur.

Soit (A_v, A_e) un couple d'alphabets, tel que $\lambda \notin (A_v \cup A_e)$. Tout comme pour A, nous associons également λ à A_v et A_e en qualité de lettre vide.

Définition 19 (Graphe)
Un graphe G (étiqueté) sur (A_v, A_e) est un couple $(vl(G), el(G))$, avec $vl(G)$ une fonction $v(G) \to A_v$ nommée étiquetage des sommets, et $el(G)$ une fonction $e(G) \to A_e$ nommée étiquetage des arêtes, telles que $v(G)$ est un ensemble fini d'éléments de $\mathbb{N}$ nommés sommets, et $e(G)$ un ensemble d'éléments de $v(G)^2$ nommés arêtes.

Définir les sommets d'un graphe comme entiers naturels est sans perte de généralité, car l'ensemble des sommets d'un graphe est fini par définition. Tout autre ensemble infini aurait donc pu convenir, mais le choix de $\mathbb{N}$ s'impose naturellement, ce par volonté de simplicité des notations.

Soient G un graphe sur (A_v, A_e), et $e = (v_{src}, v_{dest})$ une arête de G. Nous disons que $v(G)$ (resp. $e(G)$) est *étiqueté* sur A_v (resp. A_e), et que e *connecte*, ou *relie*, son sommet *origine*, ou *source*, v_{src}, à son sommet *destination* v_{dest}. Enfin, notons $|G|$ la *taille* de G, c'est-à-dire son nombre total de sommets et d'arêtes : $|G| = |v(G)| + |e(G)|$.

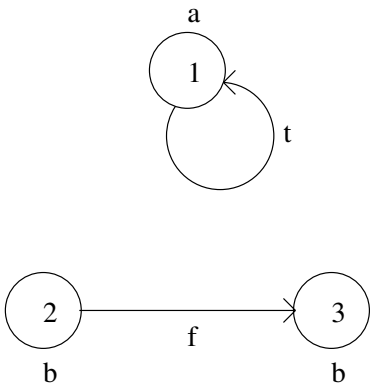

FIGURE 2.5 – Une représentation graphique du graphe exemple de la définition 19.

Exemple
Soient $A_v = \{a, b, c\}$, et $A_e = \{t, f\}$. $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$ est un graphe sur (A_v, A_e), avec $v(G) = \{1, 2, 3\}$, $e(G) = \{(1, 1), (2, 3)\}$, $vl(G)(1) = a$, $vl(G)(2) = b$, $vl(G)(3) = b$, $el(G)(1, 1) = t$, $el(G)(2, 3) = f$, et $|G| = 5$. Une représentation graphique possible de G est proposée en figure 2.5.

Remarque
$(\{\}, \{\})$ est l'unique graphe sur (A_v, A_e) de taille nulle, nommé *graphe vide*.

Définition 20 (Graphe identifié sur un ensemble)
Soit $S \subseteq \mathbb{N}$. Un graphe G sur (A_v, A_e) est identifié sur S ssi $v(G) \subseteq S$.

Soient $S \subseteq \mathbb{N}$, et $n \in \mathbb{N}$. Notons :
- $\mathbb{G}(A_v, A_e, S)^n$ l'ensemble des graphes sur (A_v, A_e), identifiés sur S, et possédant n sommets ;
- $\mathbb{G}(A_v, A_e, S)^{\leqslant n}$ l'ensemble des graphes sur (A_v, A_e), identifiés sur S, et possédant au plus n sommets ;
- $\mathbb{G}(A_v, A_e, S)^*$ l'ensemble des graphes sur (A_v, A_e) identifiés sur S.

Dès lors, nous avons :
- $\mathbb{G}(A_v, A_e, S)^0 = \{(\{\}, \{\})\}$;
- $\mathbb{G}(A_v, A_e, S)^{\leqslant n} = \bigcup_{i=0}^{n} \mathbb{G}(A_v, A_e, S)^i$;
- $\mathbb{G}(A_v, A_e, S)^* = \bigcup_{i \in \mathbb{N}} \mathbb{G}(A_v, A_e, S)^i$.

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, et $S = \{1\}$. Nous avons $\mathbb{G}(A_v, A_e, S)^0 = \{(\{\}, \{\})\}$, $\mathbb{G}(A_v, A_e, S)^1 = \{(\{(1, a)\}, \{\}), (\{(1, a)\}, \{((1, 1), t)\}), (\{(1, a)\}, \{((1, 1), f)\}), (\{(1, b)\}, \{\}), (\{(1, b)\}, \{((1, 1), t)\}), (\{(1, b)\}, \{((1, 1), f)\}), (\{(1, c)\}, \{\}), (\{(1, c)\}, \{((1, 1), t)\}), (\{(1, c)\}, \{((1, 1), f)\})\}\ldots$

Définition 21 (Degré entrant, sortant d'un sommet d'un graphe)
Soit G un graphe sur (A_v, A_e), et v un sommet de G. Le degré entrant $d_{in}(v)$ (resp. sortant $d_{out}(v)$) de v est égal au nombre d'arêtes de G pour lequel v est destination (resp. source) :

$$d_{in}(v) = |\{(v_{src}, v) \in e(G)\}|,$$
$$d_{out}(v) = |\{(v, v_{dest}) \in e(G)\}|.$$

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, et $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$. Nous avons $d_{in}(2) = 0$, et $d_{out}(2) = 1$.

Remarque
Toute arête ayant 1 sommet entrant et 1 sommet sortant, nous avons :

$$\sum_{v \in v(G)} d_{in}(v) = \sum_{v \in v(G)} d_{out}(v) = |e(G)|.$$

Définition 22 (Graphe complet)
Soit $n \in \mathbb{N}$, et nombre_max_arêtes: $\mathbb{N} \to \mathbb{N}$ la fonction associant à n le nombre maximal d'arêtes que peut avoir un graphe sur (A_v, A_e) possédant n sommets :

$$\text{nombre\_max\_arêtes}(n) = n^2.$$

Un graphe G sur (A_v, A_e) est n-complet ssi ces deux conditions sont satisfaites :
- $|v(G)| = n$;
- $|e(G)| = \text{nombre\_max\_arêtes}(n)$.

Un graphe n-complet est l'un des plus « grands » graphes possédant n sommets, en ce sens où aucun autre de ces tels graphes ne peut avoir plus d'arêtes que lui.

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, et $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$. G n'est ni 2-complet, car $|v(G)| = 3 \neq 2$; ni 3-complet, car $|e(G)| = 2 \neq \text{nombre\_max\_arêtes}(3) = 9$. Par contre $(\{(1, a)\}, \{((1, 1), t)\})$ est 1-complet.

Définition 23 (Sous-graphe)
Soient G et H deux graphes sur (A_v, A_e). H est un sous-graphe de G ssi $vl(H) \subseteq vl(G)$ et $el(H) \subseteq el(G)$.

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, et $H = (\{(1, a), (3, b)\}, \{((1, 1), t)\})$. H est un sous-graphe de G.

Définition 24 (Sous-graphe induit)
Soit G un graphe sur (A_v, A_e), et $S \subseteq v(G)$. Le sous-graphe de G induit par S est le graphe H sur (A_v, A_e) défini comme suit :

$$vl(H) = \{(v, vl(G)(v)) | v \in S\},$$
$$el(H) = \{(e, el(G)(e)) | e \in S^2 \cap e(G)\}.$$

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, et $S = \{1, 3\}$. Le sous-graphe de G induit par S est égal à $(\{(1, a), (3, b)\}, \{((1, 1), t)\})$.

Définition 25 (Isomorphisme de graphe)
Deux graphes G et H sur (A_v, A_e) sont dits isomorphes, noté $G \sim H$, ssi il existe une bijection i: $v(G) \to v(H)$ qui préserve leur structure respective :

$$(v_{src}, v_{dest}) \in e(G) \iff (i(v_{src}), i(v_{dest})) \in e(H).$$

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, et $H = (\{(1, a), (4, b), (9, b)\}, \{((1, 1), t), ((4, 9), f)\})$. La bijection i: $\{1, 2, 3\} \to \{1, 4, 9\} = \{(1, 1), (2, 4), (3, 9)\}$ assure que G et H sont isomorphes.

Définition 26 (Isoétiquetage de graphe)
Deux graphes G et H sur (A_v, A_e) sont dits isoétiquetés, noté $G \approx H$, ssi il existe une bijection i: $v(G) \to v(H)$ qui préserve leur structure et étiquetage respectifs :
- préservation de la structure :

$$(v_{src}, v_{dest}) \in e(G) \iff (i(v_{src}), i(v_{dest})) \in e(H);$$

- préservation de l'étiquetage des sommets :

$$v \in v(G) \implies vl(G)(v) = vl(H)(i(v));$$

– préservation de l'étiquetage des arêtes :

$$(v_{src}, v_{dest}) \in e(G) \Longrightarrow el(G)(v_{src}, v_{dest}) = el(H)(i(v_{src}), i(v_{dest})).$$

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, et $H = (\{(1, a), (4, b), (9, b)\}, \{((1, 1), t), ((4, 9), f)\})$. La bijection $i\colon \{1, 2, 3\} \to \{1, 4, 9\} = \{(1, 1), (2, 4), (3, 9)\}$ assure que G et H sont isoétiquetés.

Remarque
Il est évident que l'isoétiquetage de deux graphes implique leur isomorphisme, par la même bijection.

Définition 27 (Sous-graphe commun)
Soient G et H deux graphes sur (A_v, A_e), et I un sous-graphe de G. I est un sous-graphe commun à G et H ssi I est isoétiqueté à un sous-graphe de H.

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, $H = (\{(4, b), (9, b)\}, \{((4, 9), f)\})$, et $I = (\{(2, b), (3, b)\}, \{((2, 3), f)\})$. La bijection $i\colon \{2, 3\} \to \{4, 9\} = \{(2, 4), (3, 9)\}$ assure que I et H sont isoétiquetés, et donc communs à G et H.

Définition 28 (Plus grand sous-graphe commun)
Soient G et H deux graphes sur (A_v, A_e), et I un sous-graphe commun à G et H. I est un plus grand sous-graphe commun à G et H ssi aucun sous-graphe commun à G et H a plus de sommets que I.

Exemple
Soient $A_v = \{a, b, c\}$, $A_e = \{t, f\}$, $G = (\{(1, a), (2, b), (3, b)\}, \{((1, 1), t), ((2, 3), f)\})$, $H = (\{(4, b), (9, b)\}, \{((4, 9), f)\})$, et $I = (\{(2, b), (3, b)\}, \{((2, 3), f)\})$. H est un plus grand sous-graphe commun à G et H, car il est commun à G et H et qu'il n'existe aucun sous-graphe de H qui ait plus de sommets que H.

Soit $n \in \mathbb{N}$. Nous définissons enfin l'opération d'agrandissement, qui produit un nouveau graphe H, identifié sur $\{1, \ldots, n+1\}$ et étiqueté sur (A_v, A_e), à partir d'un graphe G, identifié sur $\{1, \ldots, n\}$ et étiqueté sur (A_v, A_e), dans lequel y sont éventuellement insérés le nouveau sommet $(n + 1)$, et de nouvelles arêtes reliant $(n + 1)$ aux sommets de G. L'agrandissement est neutre dans le cas où $(n + 1)$ est étiqueté par la lettre vide, et de même toute nouvelle arête potentielle étiquetée par la lettre vide ne contribue pas à l'agrandissement de $e(G)$.

Définition 29 (Agrandissement de graphe)
L'agrandissement sur (A_v, A_e) est l'opération binaire suivante :

$$\oplus^n : \mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n} \times E \to \mathbb{G}(A_v, A_e, \{1, \ldots, n + 1\})^{\leqslant n+1},$$

avec :

$$E = (A_v \cup \{\lambda\}) \times (A_e \cup \{\lambda\}) \times (A_e \cup \{\lambda\})^n \times (A_e \cup \{\lambda\})^n,$$

et telle que : $\forall G \in \mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}, \forall (l_v, l_e, l_{in}, l_{out}) \in E$:

$$G \oplus^n (l_v, l_e, l_{in}, l_{out}) = \begin{cases} G & \text{si } l_v = \lambda, \\ H & \text{sinon, avec :} \\ & vl(H) = vl(G) \cup \{(n + 1, l_v)\}, \\ & el(H) = el(G) \\ & \qquad \cup \begin{cases} \{((n + 1, n + 1), l_e)\} & \text{si } l_e \neq \lambda, \\ \{\} & \text{sinon.} \end{cases} \\ & \qquad \cup \bigcup_{i \in v(G) | (l_{in})_i \neq \lambda} ((n + 1, i), (l_{in})_i) \\ & \qquad \cup \bigcup_{i \in v(G) | (l_{out})_i \neq \lambda} ((i, n + 1), (l_{out})_i)\,. \end{cases}$$

2.6 Édition

Nous allons maintenant avoir un aperçu du cadre d'édition structurelle le plus utilisé en reconnaissance de formes ou autres champs applicatifs concernés par des données structurées, comme par exemple la bio-informatique. De ce cadre découle la définition de la *distance d'édition* permettant de mesurer la dissimilarité de deux structures X et Y en tant que coût minimal d'une *structure d'édition* permettant d'obtenir Y en transformant X via l'application d'opérations de base.

2.6.1 Structure d'édition

Définition 30 (Opération d'édition)
Une opération d'édition sur A est un couple (l, m) de lettres de $A \cup \{\lambda\}$, noté $l \to m$. Si $l = \lambda$ et $m \neq \lambda$ (resp. $l \neq \lambda$ et $m = \lambda$), alors il s'agit de l'insertion de m (resp. la suppression, ou délétion, de l) ; si $l \neq \lambda$ et $m \neq \lambda$, alors il s'agit de la substitution de l par m, et si de plus $l = m$, alors il s'agit de la substitution identique de l ; enfin, $\lambda \to \lambda$ est l'opération d'édition vide.

Nous disons qu'une telle opération *transforme* l en m, ou *consomme* l pour *produire* m.

Exemple
Soit $A = \{a, b\}$. $a \to b$ est la substitution de a par b, $a \to a$ la substitution identique de a, $a \to \lambda$ la délétion de a, $\lambda \to a$ l'insertion de a, etc.

Définition 31 (Alphabet d'édition)
L'alphabet d'édition sur A est l'alphabet des opérations d'édition sur A :

$$\text{edit}(A) = (A \cup \{\lambda\})^2 .$$

Exemple
Soit $A = \{a, b\}$. L'alphabet d'édition sur A est $\{(a, a), (a, b), (a, \lambda), (b, a), (b, b), (b, \lambda), (\lambda, a), (\lambda, b), (\lambda, \lambda)\}$.

Définition 32 (Structure d'édition)
Une chaîne (resp. Un multiensemble ; Un arbre) d'édition sur A est une chaîne (resp. un multiensemble ; un arbre) sur $\text{edit}(A)$.

Un graphe d'édition sur (A_v, A_e) est un graphe sur $(\text{edit}(A_v), \text{edit}(A_e))$.

Soit Z une chaîne d'édition sur A. Étendant le vocabulaire utilisé pour les opérations d'édition, nous disons que Z *transforme* X en Y, ou *consomme* X pour *produire* Y, avec X (resp. Y) la chaîne formée par la concaténation des lettres consommées (resp. produites) par les opérations d'édition composant Z :

$$X = \cdot_{i=1}^{|Z|} Z(i)_1,$$
$$Y = \cdot_{i=1}^{|Z|} Z(i)_2,$$

où $Z(i)_1$ (resp. $Z(i)_2$) désigne la lettre consommée (resp. produite) par Z_i (en se souvenant qu'une opération d'édition est un couple).

Pour ce qui est des multiensembles, la construction est similaire en changeant simplement l'opération de concaténation par celle d'union-addition. Le cas des graphes est également relativement simple, en utilisant l'opération d'agrandissement, et en y ajoutant certaines contraintes, comme par exemple le fait d'ignorer une arête dont la destination est un sommet qui est supprimé. Par contre, il existe deux paradigmes dans la littérature pour le cas des arbres [Bil05], l'adhésion à l'un ou l'autre dépendant de leur pertinence vis-à-vis de l'application considérée. La question est : que faire des arbres enfants d'un arbre lorsque leur parent P est supprimé ? Et les deux types de réponses sont : il faut les ignorer (utilisation de l'opération d'enracinement), ou alors il faut les rattacher à leur « grand-parent », s'il existe, c'est-à-dire le nœud parent de P, et cela récursivement jusqu'à trouver un de leurs « ancêtres » qui ne soit pas supprimé. Dans le cas d'édition d'arbres qui modélisent la structure d'un document HTML par exemple, il est clair que la première solution doit

être adoptée (une balise <li> n'est pas valide sans une balise parente <ul> ou <ol>) ; un exemple opposé est celui de la modélisation d'une hiérarchie dans une entreprise, où l'on peut tout à fait rattacher certains employés directement à leur chef plutôt qu'à leur sous-chef, et donc supprimer un poste.

Exemple
Soit $A = \{a, b\}$. $(a \to a)(b \to a)(\lambda \to b)$ est une chaîne d'édition sur A, consommant $a.b.\lambda = ab$, et produisant $a.a.b = aab$.

2.6.2 Distance d'édition

Définition 33 (Fonction de coût)
Une fonction de coût c sur A est une fonction $\text{edit}(A) \to \mathbb{R}^+$ satisfaisant les deux conditions suivantes : $\forall l, m \in (A \cup \{\lambda\}) | l \neq m$:
- ne rien faire est gratuit : $c(l \to l) = 0$;
- faire quelque chose est payant : $c(l \to m) > 0$.

c peut être utilisée pour traduire une connaissance experte ou apprise pour une application particulière, par exemple relative à la dissimilarité entre l et m, ou à la probabilité de transformation de l en m.

Le coût selon c d'une chaîne d'édition Z, noté $c(Z)$ pour faire simple, est égal à la somme des coûts des opérations d'édition composant Z :

$$c(Z) = \sum_{i=1}^{|Z|} c(Z(i)).$$

Le coût d'un multiensemble d'édition M est calculé de manière similaire, simplement en ajoutant autant de fois le coût d'une opération d'édition que sa multiplicité dans M :

$$c(M) = \sum_{e \in M} \left(M(e) \times c(e) \right).$$

Pour le cas des arbres, plusieurs approches sont possibles, en fonction du traitement spécifique des nœuds dont le nœud parent est supprimé : si l'on décide d'ignorer ces nœuds, alors il faut choisir si l'on doit considérer cette ignorance comme la suppression de ces nœuds, et donc s'il faut ajouter le coût de ces suppressions au coût total de l'arbre d'édition ; et si l'on décide de rattacher ces nœuds à un de leurs ancêtres, alors il faut ajouter le coût des opérations transformant ces nœuds au coût total de l'arbre d'édition.

Pour le cas des graphes, notons $(c_v, c_e)(G)$ le coût selon (c_v, c_e) du graphe d'édition G sur (A_v, A_e), avec c_v et c_e deux fonctions de coût sur A_v et A_e, respectivement. Grossièrement, $(c_v, c_e)(G)$ est égal à la somme des coûts selon c_v des opérations d'édition transformant un sommet, auquel y est ajoutée la somme des coûts selon c_e des opérations d'édition transformant une arête. Mais comme pour le cas des arbres, il faut faire un choix sur le traitement des opérations d'édition ignorées, par exemple les opérations transformant une arête ayant pour source ou destination un sommet supprimé : doit-on considérer qu'il faut supprimer les composantes incriminées et donc ajouter les coûts de ces suppressions au coût total du graphe d'édition ?

Exemple
Soient $A = \{a, b\}$. La fonction de coût c sur A la plus simple est celle associant la même valeur 1 à toute opération d'édition qui n'est ni une substitution identique ni l'opération vide. Dès lors, le coût selon c d'une chaîne d'édition Z sur A est égale au nombre d'opérations d'édition composant Z qui ne sont ni une substitution identique ni l'opération vide ; en d'autres termes $c(Z)$ est égal au nombre de modifications que subit la chaîne transformée par Z.

Définition 34 (Distance d'édition)
Soit c une fonction de coût sur A. La distance d'édition de chaîne (resp. multiensemble ; arbre) sur A selon c est la fonction d associant à un couple (X, Y) de chaînes (resp. multiensembles ;

arbres) sur A le coût minimal selon c d'une chaîne (resp. multiensemble ; arbre) d'édition sur A transformant X en Y :

$$d(X,Y) = \min_{Z \in X \to Y} c(Z),$$

où $X \to Y$ désigne l'ensemble des chaînes (resp. multiensembles ; arbres) d'édition sur A transformant X en Y.

Soient c_v et c_e deux fonctions de coût sur A_v et A_e, respectivement. La distance d'édition de graphe sur (A_v, A_e) selon (c_v, c_e) est la fonction d associant à un couple (G, H) de graphes sur (A_v, A_e) le coût minimal selon (c_v, c_e) d'un graphe d'édition sur (A_v, A_e) transformant G en H :

$$d(X,Y) = \min_{I \in G \to H} (c_v, c_e)(I),$$

où $G \to H$ désigne l'ensemble des graphes d'édition sur (A_v, A_e) transformant G en H.

Exemple
Soient $A = \{a, b\}$, c la fonction de coût simple sur A définie dans l'exemple suivant la définition 33, d distance d'édition de chaîne selon c, $X = aba$, et $Y = aa$. Nous avons $d(X,Y) = c((a \to a)(b \to \lambda)(a \to a)) = 0 + 1 + 0 = 1$.

Le concept de distance d'édition a été initialement formalisé par Levenshtein [Lev66] pour le cas des chaînes, et Wagner et Fischer [WF74] furent les premiers à proposer un algorithme efficace pour son calcul. Cet algorithme repose sur une méthode de programmation dynamique de complexité temporelle et spatiale $O(|X| \times |Y|)$, avec X et Y les deux chaînes incriminées. Soient c une fonction de coût sur A, et d la distance d'édition de chaîne selon c. Le déroulement de l'algorithme peut être décrit par la fonction récursive suivante : $\forall i \in \{1, \ldots, |X|\}, \forall j \in \{1, \ldots, |Y|\}$:

$$\begin{aligned}
&\text{matrix}(0,0) = 0,\\
&\text{matrix}(i,0) = \text{matrix}(i-1,0) + c(X(i) \to \lambda),\\
&\text{matrix}(0,j) = \text{matrix}(0,j-1) + c(\lambda \to Y(j)),\\
&\text{matrix}(i,j) = \min \left\{ \begin{array}{c} \text{matrix}(i-1,j) + c(X(i) \to \lambda),\\ \text{matrix}(i,j-1) + c(\lambda \to Y(j)),\\ \text{matrix}(i-1,j-1) + c(X(i) \to Y(j)) \end{array} \right\},
\end{aligned}$$

et Wagner et Fischer ont prouvé que $\text{matrix}(|X|, |Y|) = d(X,Y)$.

Un grand nombre d'algorithmes ont été proposés dans la littérature pour le calcul efficace d'une distance d'édition d'arbre [Bil05]. Ils s'appuient tous plus ou moins sur l'utilisation d'un schéma récursif du même genre que celui proposé par Wagner et Fischer pour les chaînes. Pour le cas des graphes, le problème est NP-difficile. Des approches ont été proposées dans la littérature pour approximer ce calcul [RKH05, NRB06], c'est-à-dire trouver un graphe d'édition (ou tout autre formalisme équivalent pour modéliser la transformation d'un graphe en un autre) proche de l'optimalité.

Définition 35 (Structure d'édition optimale)
Soit c une fonction de coût sur A, d la distance d'édition de chaîne (resp. multiensemble ; arbre) selon c, X et Y deux chaînes (resp. multiensembles ; arbres) sur A, et Z une chaîne (resp. un multiensemble ; un arbre) d'édition sur A transformant X en Y. Z est dite optimale (resp. dit optimal ; dit optimal) selon c (ou d) ssi $c(Z) = d(X,Y)$.

Soient c_v et c_e deux fonctions de coût sur A_v et A_e, respectivement, d la distance d'édition de graphe selon (c_v, c_e), G et H deux graphes sur (A_v, A_e), et I un graphe d'édition sur (A_v, A_e) transformant G en H. I est dit optimal selon (c_v, c_e) (ou d) ssi $(c_v, c_e)(I) = d(G,H)$.

Exemple
Soient $A = \{a, b\}$, c la fonction de coût simple sur A définie dans l'exemple suivant la définition 33, $X = aba$, $Y = aa$, et $Z = (a \to a)(b \to \lambda)(a \to a)$. Z est optimale selon c.

Il existe généralement, selon une même distance d'édition, plusieurs structures d'édition optimales transformant une structure en une autre.

Enfin, notons que de nombreuses variantes de la distance d'édition ont été formalisées dans la littérature, surtout pour le cas des chaînes, avec par exemple la distance d'édition *normalisée* [MV93], tenant en compte la longueur des chaînes, et bien d'autres encore [PM02], [CB08], [SS07]...

Chapitre 3

Statistiques

Nous allons voir dans ce chapitre un ensemble d'approches définies dans la littérature pour caractériser statistiquement les ensembles de structures discrètes.

La littérature se divise sur ce sujet en deux grandes catégories de travaux communs ou qui partagent la même philosophie.

3.1 Statistiques basées sur une distance

Dans cette section, nous allons voir de quelle manière une partie de la communauté travaillant sur la définition de statistiques d'ensembles de structures tente d'imiter le modèle numérique classique par le biais du concept de distance (cf. définition 94).

3.1.1 Médiane

Soit S une séquence de taille finie et non nulle de nombres réels, et T la séquence obtenue en triant les éléments de S selon l'ordre croissant. La médiane de S est définie comme le réel qui partage T en deux parties de même taille :

Définition 36 (Médiane réelle)

$$\text{médiane}(S) = \begin{cases} T_{(|S|+1)/2} & \text{si } |S| \text{ est impair,} \\ \dfrac{T_{|S|/2} + T_{(|S|/2)+1}}{2} & \text{sinon.} \end{cases}$$

Exemple
Soit $S = (2, 1, 3, 1)$. Nous avons $T = (1, 1, 2, 3)$, et médiane$(S) = (T_2 + T_3)/2 = (1 + 2)/2 = 1{,}5$.

Une caractéristique principale la médiane réelle est qu'elle minimise l'*erreur absolue* à S, c'est-à-dire la somme des valeurs absolues de la différence entre elle et une valeur de S :

$$\text{médiane}(S) = \arg\min\{x \in \mathbb{R}\} \sum_{i=1}^{|S|} |x - S_i| \,.$$

Cette notion d'erreur absolue, traduite dans le cas réel par la valeur absolue, peut aisément être généralisée à tout ensemble par le concept de distance, comme Weiszfeld l'a proposé avec la distance euclidienne (cf. exemple de la définition 94) sur le produit cartésien de copies de $\mathbb{R}$ [Wei37]. En effet, la valeur absolue est une fonction vérifiant l'axiomatique de distance, d'où la volonté de définir la médiane selon un espace métrique via la généralisation de la propriété précitée, qui serait le critère à vérifier pour qu'un élément de l'ensemble considéré soit qualifié d'élément médian.

Définition 37 (Médiane selon une distance)
Soient (E, d) un espace métrique, S une séquence de taille finie et non nulle d'éléments de E, et $y \in E$. y est une médiane de S selon d ssi :

$$\sum_{i=1}^{|S|} d(y, S_i) = \min\{x \in E\} \sum_{i=1}^{|S|} d(x, S_i).$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, et $S = ((2,1),(3,1))$. L'ensemble des médianes de S selon d est le suivant :

$$\{(x,1)|x \in [2,3]\}.$$

Il se peut en effet qu'il y ait plusieurs éléments de E qui minimisent cette somme de distances (SOD pour *Sum Of Distances*). Notons médiane(S, d) l'ensemble des éléments médians de S selon d.

Soit A un alphabet. Pour les données qui nous intéressent, Kohonen [Koh85] a été le premier à proposer d'utiliser cette dernière définition pour le cas de chaînes sur A, et avec une distance d'édition. Le problème de cette approche est sa complexité : calculer la (une) chaîne médiane selon une distance d'édition est un problème NP-difficile [dlHC00, SP03], c'est-à-dire que le seul moyen de trouver cette chaîne dans le cas général est d'énumérer l'ensemble des chaînes sur A, ou au moins celles de longueur au plus n, avec n la longueur maximale d'une chaîne de S. Or ce procédé requiert un temps de calcul exponentiel en n.

Quand l'informaticien doit résoudre un problème combinatoire qu'il sait ou au moins suppose NP-difficile, s'offre à lui deux possibilités :

1. soit il doit se résoudre à considérer seulement les éventuelles instances de son problème pour lesquelles la solution exacte est obtenable en temps polynomial ;
2. soit il a recours à des techniques d'optimisation combinatoire (métaheuristiques) afin d'arriver à un résultat qui a de fortes chances d'être approximé et donc n'être que sous-optimal.

Il existe bien entendu des classes d'instances de ce problème de chaîne médiane dont la résolution requiert un temps de calcul seulement polynomial, par exemple avec certaines fonctions de coût ayant des caractéristiques particulières.

Pour ce qui est du deuxième point de l'énumération ci-dessus, il existe un grand nombre de métaheuristiques (génétique, locale, colonies de fourmis...), chacune étant plus ou moins performante en fonction des paramètres particuliers à chaque instance. Le lecteur peut se référer à l'article de Blum et Roli [BR03] pour avoir un aperçu assez complet de l'outil métaheuristique dans un cadre général.

Un nombre important de métaheuristiques ont été appliquées à la résolution de ce problème. Une comparaison de l'efficacité de certaines d'entre elles est disponible dans un article de l'équipe de recherche de Bunke [JBC04], qui a proposé de plus une étude approfondie de cette tâche et montré les bénéfices que peut apporter la chaîne médiane à des problèmes de classification classiques en reconnaissance de formes structurelle, comme par exemple la classification de chiffres dessinés à la main, dont le codage sous forme de chaîne est obtenu par calcul de différences de directions dans le contour discret. L'efficacité d'une approche se mesure en fonction d'un compromis entre minimisation du temps de calcul et minimisation de la SOD.

Notons que lorsque la recherche de la médiane est limitée à l'ensemble des chaînes appartenant à la séquence S elle-même, nous obtenons ce qui est couramment nommé *set median* dans la littérature internationale. Dans cette situation, la médiane est souvent renommée *generalized median*, de manière à bien distinguer ces deux concepts. Enfin, le lecteur doit noter que la définition de chaîne médiane est souvent liée à un *ensemble* plutôt qu'à une *séquence*, ce qui n'est pas la reproduction exacte dans le domaine des chaînes de la propriété de la médiane réelle, mais cela ne change rien au résultat lorsque la séquence ne possède aucun élément répété. Or les bases de données de chaînes utilisées dans la littérature ne possèdent que rarement des répétitions du fait de la longueur des chaînes et de la taille de l'alphabet A utilisé. Cependant, la définition devrait toujours être respectée puisqu'elle est plus fidèle à la propriété que nous avons énoncé précédemment, et qu'elle permet

de raisonner de manière exacte même dans les situations où la base de chaînes considérée contient effectivement des répétitions. Pour faire simple dans la suite, nous n'allons plus considérer ce genre de détails, et allons de fait utiliser les termes médiane, *set median...* sans se préoccuper de savoir s'ils sont liés à un ensemble ou bien à une séquence dans un cas particulier considéré.

Nous renvoyons le lecteur sur des ensemble d'articles complémentaires pour avoir une vision globale des méthodes d'optimisation combinatoire ou autre utilisées pour ce problème : recherche gloutonne [CdA97, Kru99], locale [MHJC00], fonction linéaire par parties pour alphabet numérique [SLT02], algorithme dynamique [JABC03], génétique [JBC04], approche stochastique [ORO08].

Pour le cas d'arbres, la difficulté est d'autant plus élevée que la distance d'édition est plus complexe à calculer, et le constat est pire pour les graphes où cette distance est un problème NP-difficile. Il existe cependant des études de cas dans la littérature où il est montré que des solutions sous-optimales de bonne qualité peuvent être obtenues avec des métaheuristiques telles que celles précitées (recherche génétique [BMJ99, JMB01]), ou autres heuristiques ad hoc, c'est-à-dire mises au point spécifiquement pour ce problème (regroupement de sommets [HW03], implantation dans un espace vectoriel [FVS+08]), même avec des calculs de distance d'édition de graphe nécessairement approximés.

3.1.2 Moyenne

Dans le domaine numérique, le terme « moyenne » est souvent ambigu de par son association à plusieurs concepts. Il peut être utilisé pour désigner, entre autres, celui de moyenne arithmétique ou géométrique d'une séquence de données, ou celui de moyenne théorique d'une variable aléatoire réelle (définition 119). Nous allons voir deux concepts différents de moyenne proposés dans la littérature pour le cas de structures.

Moyenne arithmétique d'une séquence

Soit S une séquence de taille finie et non nulle de nombres réels. La moyenne arithmétique de S est définie comme suit :

Définition 38 (Moyenne arithmétique réelle)

$$\text{moyenne}(S) = \frac{\sum_{i=1}^{|S|} S_i}{|S|}.$$

Exemple
$\text{moyenne}(2,1,3,1) = (2+1+3+1)/4 = 7/4 = 1{,}75$.

Cette moyenne possède une propriété très similaire à celle possédée par la médiane réelle vue précédemment (section 3.1.1) : tout comme la médiane minimise l'erreur absolue à S, la moyenne arithmétique minimise l'*erreur carrée* à S, c'est-à-dire la somme des carrés de la différence entre elle et une valeur de S :

$$\text{moyenne}(S) = \arg\min\{x \in \mathbb{R}\} \sum_{i=1}^{|S|} (x - S_i)^2 .$$

Il est donc tout à fait possible de raisonner de la même manière que pour la médiane pour généraliser aux espaces métriques la définition de la moyenne arithmétique :

Définition 39 (Moyenne arithmétique selon une distance)
Soient (E, d) un espace métrique, S une séquence de taille finie et non nulle d'éléments de E, et $y \in E$. y est une moyenne arithmétique de S selon d ssi :

$$\sum_{i=1}^{|S|} d(y, S_i)^2 = \min\{x \in E\} \sum_{i=1}^{|S|} d(x, S_i)^2.$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, et $S = ((2,1),(3,1))$. L'unique moyenne arithmétique de S selon d est $(2{,}5, 1)$.

Il se peut en effet qu'il y ait plusieurs éléments de E qui minimisent cette somme de carrés de distances (SOSD pour *Sum Of Squared Distances*). Notons moyenne(S, d) l'ensemble des éléments moyens de S selon d.

Notons enfin qu'il est faux de dire qu'une moyenne de S selon une distance d est une médiane de S selon la distance égale au carré de d, pour la simple et bonne raison que le carré d'une distance n'est pas une distance puisque cette fonction ne vérifie pas l'inégalité triangulaire (cf. définition 94).

Pour le cas structurel avec une distance d'édition, il est fortement probable, même si ce fait n'a pas encore été prouvé ni réfuté, que ce problème de structure moyenne soit NP-difficile comme l'est le problème de structure médiane. Même cause mêmes effets, toute métaheuristique appliquée au calcul d'une structure médiane peut être appliquée au calcul d'une structure moyenne, simplement en changeant la distance par le carré de cette même distance, pour passer d'une minimisation de SOD à une minimisation de la SOSD.

Ce concept de structure moyenne ne semble pas aussi attrayant que celui de médiane pour la communauté. La raison est qu'une médiane doit être un meilleur représentant « central » pour une séquence de données issues du monde réel que ne peut l'être une moyenne, car en minimisant un carré de distances, une moyenne tient plus compte des éléments qui lui sont éloignés que ne le fait une médiane, cela ayant pour effet une moins grande robustesse aux données aberrantes (*outliers* en anglais) présentes dans un échantillon de données d'une application particulière. Martínez et al. [MJC01] ont pourtant revendiqué des résultats aussi bons voire légèrement meilleurs en termes de classification en se basant sur le concept de moyenne sur une base de chromosomes codés sous forme de chaînes. À un chromosome de l'échantillon de test est affectée la classe pour laquelle l'élément central (moyenne, médiane) est le moins distant. Ces résultats pratiques ne vont pas dans le sens de la théorie mais cela est dû aux paramètres et au déroulement spécifiques de l'application, qui font que les éléments centraux utilisés pour chaque classe ne sont pas vraiment leur médiane ou moyenne (chaque classe est préalablement partitionnée en plusieurs sous-classes selon l'algorithme des k-centroïdes [Ste56, JMF99] avec la *set median* pour centroïde, et les éléments centraux de chacune de ces sous-classes sont ensuite recalculés avec des algorithmes nécessairement approximatifs, pour cause de NP-difficulté de la tâche).

Moyenne pondérée de deux

Une moyenne pondérée de deux réels est clairement définie de la manière suivante :

Définition 40 (Moyenne pondérée de deux réels)
Soient $x, y \in \mathbb{R}$, et $w_x \in [0; 1]$.

$$\text{moyenne}(x, y, w_x) = [w_x \times x] + [(1 - w_x) \times y]\,.$$

Exemple

$$\text{moyenne}(2, 1, 0{,}4) = (2 \times 0{,}4) + (1 \times 0{,}6) = 0{,}8 + 0{,}6 = 1{,}4.$$

Soit $m = (x, y, w_x)$. Le poids de x dans m est égal à w_x, et celui de y est égal à $(1 - w_x)$. Or cette moyenne peut être définie alternativement de la manière suivante :

$$[|m - x| = (1 - w_x) \times |x - y|] \wedge [|m - y| = w_x \times |x - y|]\,,$$

et ce constat est généralisé quelle que soit la dimension de notre espace numérique, c'est-à-dire plus précisément en considérant des éléments d'un espace vectoriel classique (avec les opérations coordonnée par coordonnée) de dimension $n \in \mathbb{N} \setminus \{0\}$, et en remplaçant le concept d'erreur absolue, instanciée par la fonction valeur absolue dans le cas unidimensionnel, par celui de n-distance (la valeur absolue est égale à la 1-distance). Nous obtenons ensuite la propriété que toute moyenne pondérée m de 2 vecteurs x et y de taille n se situe nécessairement sur une « ligne directe » entre x et y selon la n-distance, c'est-à-dire que la n-distance entre x et y est égale à la somme des

n-distances entre x et m et entre m et y : il n'est pas plus « coûteux » en termes de distance de « passer par m pour aller de x à y ».

Nous pouvons donc nous abstenir du concept d'espace vectoriel pour généraliser la définition de moyenne pondérée selon une distance, de la manière suivante :

Définition 41 (Moyenne pondérée de deux éléments selon une distance)
Soient (E, d) un espace métrique, et $x, y, z \in E$. z est une moyenne pondérée de x et y selon d ssi l'équation suivante est vérifiée :

$$d(x, y) = d(x, z) + d(z, y).$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, $x = (2, 1)$, et $y = (3, 1)$. L'ensemble des moyennes de S selon d est le suivant :

$$\{(x, 1) | x \in [2, 3]\}.$$

Soit m une moyenne pondérée de x et y. Le poids de x dans m est égal à $d(m, y)/d(x, y)$, et celui de y est égal à $d(x, m)/d(x, y)$. Notons qu'il existe au moins deux moyennes pondérées de x et y, à savoir x et y. De plus, une moyenne pondérée de x et y est nécessairement une médiane de (x, y) (ou (y, x)), et vice versa, cela d'après l'inégalité triangulaire (cf. définition 94). Nous obtenons donc la définition alternative suivante :

Définition 42 (Moyenne pondérée de deux éléments selon une distance)
Soient (E, d) un espace métrique, et $x, y, z \in E$. z est une moyenne pondérée de x et y selon d ssi z est une médiane de (x, y) selon d.

Notons moyenne(x, y, d) l'ensemble des moyennes pondérées de x et y selon d.

Soit A un alphabet. Bunke et al. [BJAK02] ont proposé une procédure pour calculer une moyenne pondérée M de deux chaînes X et Y sur A selon une distance d'édition d, et qui peut être résumée de la manière suivante :

1. calculer une chaîne d'édition optimale O selon d transformant X en Y ;
2. choisir une sous-séquence O' de O ;
3. appliquer O' à X (en veillant à respecter les positions des opérations de O' dans O), pour produire M.

Le poids de X dans M est donc égal à $[d(X, Y) - c(O')]/d(X, Y)$, et celui de Y est égal à $c(O')/d(X, Y)$, où c désigne la fonction de coût sous-jacente à d. Cet algorithme est particulièrement intéressant car il permet de générer un ensemble de moyennes pondérées de deux formes représentées par des chaînes, cela en contrôlant le poids de chacune, et l'interprétation visuelle est très parlante dans le cas de formes à 2 dimensions, comme montré dans l'article de Bunke et al. [BJAK02] avec des chiffres dessinés à la main, où l'on peut voir l'évolution visuelle d'un 2 vers un 1 par exemple. Cet algorithme permet ainsi de produire de nouvelles formes bruitées à partir de deux références. Ce résultat n'aurait pas été possible en considérant la deuxième définition de moyenne pondérée, car les algorithmes de calcul de chaîne médiane ne permettent pas de spécifier les distances de M à X et Y.

Cet algorithme est clairement adaptable aux structures plus complexes telles que les arbres ou les graphes. La seule différence est que la structure d'édition O étant de type différent, son application est différente, mais la philosophie est la même, à savoir l'application partielle d'une structure d'édition optimale, selon la distance d'édition correspondante, transformant X en Y. Bunke et Günter [BG01] se sont attaqués au cas des graphes avec une représentation de O non pas sous forme de graphe d'édition mais de ce qu'il nomment *isomorphisme de graphe correcteur d'erreur* (ECGI pour *Error-Correcting Graph Isomorphism*), formalisme légèrement différent du graphe d'édition, mais dont les objectifs et la puissance de représentation sont identiques.

Enfin, le concept de moyenne pondérée de deux graphes peut être relié à certains concepts de la théorie des graphes sous certaines conditions. Par exemple, Bunke et Kandel [Bun99, BK00] ont montré que sous certaines classes de fonctions de coût, un plus grand sous-graphe commun C

de deux graphes G et H était également une moyenne pondérée de G et H, et qu'il en était de même pour tout sous-graphe de G ou de H pour lequel C est un sous-graphe. Des résultats d'un tel genre augurent, dans la situation où les fonctions de coût associées aux alphabets des sommets et des arêtes de la classe de graphes considérée satisfont les conditions nécessaires, le fait de pouvoir utiliser des algorithmes de graphe moyen pour résoudre certains problèmes de théorie des graphes. Dans tous les cas, parce qu'ils sont basés sur le concept de distance d'édition, ces résultats ne pourraient s'abstenir d'utiliser bien souvent des solutions seulement sous-optimales.

3.1.3 Variance, écart-type

Soit S une séquence de taille finie et non nulle de nombres réels, et C la séquence des carrés des écarts des éléments de S à la moyenne arithmétique de S :

$$(|C| = |S|) \wedge \left(\forall i \in \{1, \ldots, |S|\}, C_i = [S_i - \text{moyenne}(S)]^2\right).$$

La variance de S est définie comme la moyenne arithmétique de C, et l'écart-type de S est égale à la racine carrée de sa variance :

Définition 43 (Variance et écart-type réels)

$$\text{variance}(S) = \text{moyenne}(C) = \frac{\sum_{i=1}^{|S|} [S_i - \text{moyenne}(S)]^2}{|S|},$$
$$\text{écart-type}(S) = \sqrt{\text{variance}(S)}.$$

Exemple
Soit $S = (2, 1, 3, 1)$. Nous avons moyenne$(S) = 1{,}75$, $C = ((2 - 1{,}75)^2, (1 - 1{,}75)^2, (3 - 1{,}75)^2, (1 - 1{,}75)^2) = (0{,}0625, 0{,}5625, 1{,}5625, 0{,}5625)$, variance$(S) = 0{,}6875$, et écart-type$(S) \simeq 0{,}83$.

Ces statistiques donnent une idée de la « dispersion » de S autour de sa moyenne. Ces concepts étant basés sur ceux de moyenne et d'écart (ou erreur), nous pouvons obtenir directement les définitions de variance et écart-type selon une distance de la manière suivante :

Définition 44 (Variance et écart-type selon une distance)
Soient (E, d) un espace métrique, S une séquence de taille finie et non nulle d'éléments de E, m une moyenne de S selon d. La variance de S selon (m, d) est définie de la manière suivante :

$$\text{variance}(S, m, d) = \frac{\sum_{i=1}^{|S|} d(S_i, m)^2}{|S|},$$

et l'écart-type de S selon (m, d), de la manière suivante :

$$\text{écart-type}(S, m, d) = \sqrt{\text{variance}(S, m, d)}.$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, et $S = ((2, 1), (3, 1))$. L'unique moyenne arithmétique de S selon d est $m = (2{,}5, 1)$, et la variance de S selon (m, d) est égale à $(0{,}25+0{,}25)/2 = 0{,}25$.

Une variance (ou écart-type) est nécessairement liée à une moyenne particulière et il y a dans le pire des cas autant de variances de S qu'il y a de moyennes de S, et le problème est que ce nombre peut être très grand.

Pour le cas de structures, Jolion [Jol03a] a proposé une définition alternative d'écart-type spécifiquement pour la distance d'édition, appliquée initialement au type chaîne. Selon son approche, cet écart n'est pas représenté par une distance à une chaîne moyenne mais par une chaîne d'édition D transformant une chaîne médiane M. De fait, D contient à la fois l'information de la médiane (M), de la distance-type à cette médiane (le coût de D selon la fonction de coût considérée), et de la transformation-type de cette médiane (D). Le choix de baser cette définition sur le concept de médiane plutôt que celui de moyenne est justifié par des considérations applicatives, étant donné que la médiane est théoriquement plus robuste que la moyenne, comme nous l'avons vu précédemment. La définition de cet écart-type est la suivante :

Définition 45 (Écart-type selon des distances d'édition de chaîne)
Soient A un alphabet, S une séquence de taille finie et non nulle de chaînes sur A, d_1 une distance d'édition sur A, d_2 une distance d'édition sur edit(A), et M une médiane de S selon d_1. Un écart-type D de S selon (M, d_1, d_2) est une médiane selon d_2 d'une séquence E de taille $|S|$ telle que E_i est une chaîne d'édition optimale selon d_1 transformant M en S_i ($i \in \{1, \ldots, n\}$) :

$$\begin{aligned} &(|E| = |S|) \\ \wedge\, &(\forall i \in \{1, \ldots, |S|\}, E_i \in M \rightarrow_{d_1} S_i) \\ \wedge\, &(D \in \text{médiane}(E, d_2))\,, \end{aligned}$$

où $M \rightarrow_{d_1} S_i$ désigne l'ensemble des chaînes d'éditions optimales selon d_1 transformant M en S_i.

Notons écart-type(S, M, d_1, d_2) l'ensemble des écarts-type de S selon (M, d_1, d_2).
Le problème de cette définition est qu'elle hérite doublement des inconvénients de la médiane :
- il peut exister plusieurs médianes de S selon d_1 ;
- quelle que soit l'une de ces médianes M choisie, et quel que soit $i \in \{1, \ldots, |S|\}$, il peut exister plusieurs chaînes d'éditions optimales selon d_1 transformant M en S_i ;
- quelle que soit la séquence E de ces chaînes d'édition choisies, il peut exister plusieurs chaînes médianes D de E selon d_2 ;
- le calcul d'une chaîne médiane de S selon d_1 est NP-difficile, et il en de même pour le calcul d'une chaîne médiane de E selon d_2.

Pour résumer, le nombre de chaînes écart-type de S selon d_1 et d_2 peut être très grand, et n'en trouver ne serait-ce qu'une est une tâche très difficile. Pour ce qui est du double problème de calculabilité, il peut être limité en approximant les médianes par des solutions sous-optimales, comme nous l'avons vu précédemment. Jolion [Jol03a] a même proposé d'en rester deux fois à la *set median*, ce qui définirait le concept de *set deviation*.

Cette définition possède en plus un autre inconvénient : d_2 étant une distance d'édition sur l'alphabet des opérations d'éditions sur A, elle nécessite une fonction de coût sur ce même alphabet, c'est-à-dire que nous devons spécifier quel est le coût d'insérer telle suppression (sur A), et celui de substituer telle insertion par telle substitution, etc. Cette spécification est dépendante de l'application considérée et doit être guidée par de la connaissance experte.

Cet écart-type est aisément adaptable aux structures plus complexes que les chaînes, avec cependant les problèmes de complexités soulevés par la distance d'édition, comme nous en avons discuté pour les concepts de structure médiane et moyenne.

Ce concept a été appliqué avec des résultats honorables pour le traitement de mots bruités [Jol03a] résultant d'extraction de texte dans des séquences vidéo. Un texte est extrait plusieurs fois, obtenant ainsi un ensemble de représentants R pour chaque mot du texte. Ensuite, une *set median* M et une *set deviation* D de R sont calculées, et cette procédure est ré-itérée jusqu'à ce qu'un critère de convergence soit vérifié. À chaque itération, les mises à jour $\widehat{M}$ et $\widehat{D}$ de M et D, respectivement, sont fonction du nouveau poids affecté à chaque représentant $X \in R$, ce poids étant quant à lui fonction de la distance de X à M et de la distance-type selon D. $\widehat{M}$ et $\widehat{D}$ sont donc plus précisément une *set median pondérée* et une *set deviation pondérée*, respectivement, définies comme suit :

Définition 46 (*Set median* pondérée selon une distance)
Soit (E, d) un espace métrique, F un sous-ensemble de cardinal fini et non nul de E, w une fonction $F \rightarrow \mathbb{R}^+$, et $z \in F$. z est une *set median* pondérée de F selon (w, d) ssi :

$$\sum_{x \in F} [w(x) \times d(x, z)] = \min\{y \in F\} \sum_{x \in F} [w(x) \times d(x, y)]\,.$$

Notons qu'une *set median* d'une séquence S est une *set median* pondérée de l'ensemble des éléments appartenant à S, en fixant le poids de chacun de ces éléments comme étant égal au nombre de leur(s) copie(s) dans S. Notons set_median(F, w, d) l'ensemble des *set median* pondérées de F selon (w, d).

Tout comme la définition d'écart-type selon l'approche de Jolion, la *set deviation* pondérée est liée à la distance d'édition, car elle utilise le concept de chaîne d'édition.

Définition 47 (*Set deviation* pondérée selon des distances d'édition de chaîne)
Soient A un alphabet, $n \in \mathbb{N} \setminus \{0\}$, $E = \{X_1, \ldots, X_n\}$ un ensemble de cardinal n de chaînes sur A, d_1 une distance d'édition sur A, d_2 une distance d'édition sur edit(A), w_1 une fonction $E \to \mathbb{R}^+$, et M une *set median* pondérée de E selon (w_1, d_1). Une *set deviation* pondérée D de E selon (M, w_1, d_1, d_2) est une *set median* pondérée selon (w_2, d_2) d'un ensemble $F = \{Y_1, \ldots, Y_n\}$ de chaînes d'édition optimales selon d_1, tel que Y_i transforme M en X_i $(i \in \{1, \ldots n\})$, et tel que w_2 est une fonction $F \to \mathbb{R}^+$ associant à Y_i le poids selon w_1 de X_i $(i \in \{1, \ldots, n\})$:

$$\begin{aligned} &(\forall i \in \{1, \ldots n\}, Y_i \in M \to_{d_1} X_i) \\ \wedge\, &(\forall i \in \{1, \ldots n\}, w_2(Y_i) = w_1(X_i)) \\ \wedge\, &(D \in \text{set\_median}(F, w_2, d_2)), \end{aligned}$$

où $M \to_{d_1} X_i$ désigne l'ensemble des chaînes d'éditions optimales selon d_1 transformant M en X_i.

Notons qu'une *set deviation* d'une séquence S est une *set deviation* pondérée de l'ensemble des éléments appartenant à S, en fixant le poids de chacun de ces éléments comme étant égal au nombre de leur(s) copie(s) dans S. Notons set_deviation(E, M, w_1, d_1, d_2) l'ensemble des *set deviation* pondérées de E selon (M, w_1, d_1, d_2).

Enfin, ce formalisme a également été appliqué au partitionnement (en anglais *clustering* [JMF99]) d'ensembles de dessins à 2 dimensions, dont le codage sous forme de chaîne est obtenu par séquencement des directions du trait dans le contour discret [Jol03b]. L'algorithme de *clustering* utilisé est une adaptation de l'estimateur des ellipsoïdes de volume minimum (estimateur MVE pour *Minimum Volume Ellipsoid*), qui nécessite le calcul de moyennes et d'écarts-type. Pour son adaptation au type chaîne, Jolion [Jol03b] utilise les concepts de *set median* et *set deviation*, respectivement. Cet article montre les apports bénéfiques de la *set deviation* pour le partitionnement de données structurées, en comparaison avec un partitionnement qui bénéficie uniquement des apports de la *set median*.

3.1.4 Généralisation

L'avantage de baser ces statistiques sur une distance est que nous pouvons aisément généraliser l'approche à des ordres supérieurs. Par exemple, tout comme une médiane minimise une somme de distances, et tout comme une moyenne arithmétique minimise une somme de carrés de distances, nous définissons une statistique nommée *élément central d'ordre k* qui minimise une somme de distances élevées à la puissance k, avec k un entier naturel strictement positif :

Définition 48 (Élément central d'ordre k selon une distance)
Soient (E, d) un espace métrique, $k \in \mathbb{N} \setminus \{0\}$, S une séquence de taille finie et non nulle d'éléments de E, et $y \in E$. y est un élément central d'ordre k de S selon d ssi :

$$\sum_{i=1}^{|S|} d(y, S_i)^k = \min\{x \in E\} \sum_{i=1}^{|S|} d(x, S_i)^k.$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, $k = 2$, et $S = ((2,1),(3,1))$. L'unique élément central d'ordre k (moyenne arithmétique) de S selon d est $(2{,}5, 1)$.

Tout comme nous l'avons vu avec la médiane $(k = 1)$ et la moyenne arithmétique $(k = 2)$, il se peut en effet qu'il y ait plusieurs éléments de E qui minimisent cette somme de distances élevées à la puissance k.

Les problèmes restent cependant ceux de la complexité pour le cas de structures. De plus, tout comme une moyenne arithmétique n'est pas aussi attrayante qu'une médiane pour la communauté appliquant ses travaux sur des données issues du monde réel, l'attractivité d'un élément central diminue avec l'augmentation de son ordre, car sa robustesse aux données aberrantes diminue.

Pour ce qui est des concepts de variance et écart-type, nous pouvons également les généraliser aux ordres supérieurs dans un espace métrique de la manière suivante :

Définition 49 (Distance-type d'ordre (k, l) selon une distance)
Soient (E, d) un espace métrique, $k, l \in \mathbb{N} \setminus \{0\}$, S une séquence de taille finie et non nulle d'éléments de E, et c un élément central d'ordre k de S selon d. La distance-type d'ordre (k, l) de S selon (c, d) est définie de la manière suivante :

$$\text{distance\_type}(S, c, d, k, l) = \frac{\sum_{i=1}^{|S|} d(S_i, c)^l}{|S|}.$$

Exemple
Soient $E = \mathbb{R}^2$, d la distance euclidienne sur E, $k = 2$, $l = 2$, et $S = ((2,1),(3,1))$. L'unique élément central d'ordre k (moyenne arithmétique) de S selon d est $m = (2{,}5, 1)$, et la distance-type d'ordre (k, l) (variance) de S selon (m, d) est égale à $(0{,}25 + 0{,}25)/2 = 0{,}25$.

Une distance-type d'ordre (k, l) est nécessairement liée a un élément central d'ordre k particulier et il y a dans le pire des cas autant de distances-type d'ordre (k, l) de S qu'il y a d'éléments centraux d'ordre k de S, et le problème est que ce nombre peut être très grand.

Jolion [Jol03a] a généralisé son approche d'écart-type selon une distance d'édition pour définir un ensemble de statistiques de dispersion d'ordre supérieur représentées par une chaîne d'édition. L'ordre d'une telle chaîne T est égal à celui de son alphabet d'opérations d'édition, et T joue le rôle de transformation-type d'une transformation-type d'ordre directement inférieur. La définition récursive peut être reformulée de la manière suivante :

Définition 50 (Transformation-type d'ordre k selon des distances d'édition de chaîne)
Soient A un alphabet, S une séquence de taille finie et non nulle de chaînes sur A, $k \in \mathbb{N} \setminus \{0\}$, d une distance d'édition sur A, D une séquence de taille k telle que k_i est une distance d'édition sur $\text{edit}(A, k)$ ($i \in \{1, \ldots, k\}$), et X une chaîne sur $\text{edit}(A, k)$. X est une transformation-type d'ordre k de S selon (d, D) ssi :

- si $k = 1$:

$$\begin{aligned} &[\exists M \in \text{médiane}(S, d)] \\ \wedge\, &[\exists E \in \text{édition}(M, S, d)] \\ \text{tels que : } &X \in \text{médiane}(E, D_1); \end{aligned}$$

- sinon ($k > 1$) :

$$\begin{aligned} &[\exists T \in \text{transformation\_type}\,(S, d, (D_1, \ldots, D_{k-1}))] \\ \wedge\, &[\exists P \in \text{édition\_couples}\,(S, d, (D_1, \ldots, D_{k-1}))] \\ \wedge\, &[\exists E \in \text{édition}(T, P, D_{k-1})] \\ \text{tels que : } &X \in \text{médiane}(E, D_k), \end{aligned}$$

avec :

- $\text{edit}(A, 0) = A$;
- $\text{edit}(A, k) = \text{edit}\,(\text{edit}(A, k-1))$, c'est-à-dire l'alphabet des opérations d'édition sur $\text{edit}(A, k-1)$;
- $\text{transformation\_type}\,(S, d, (D_1, \ldots, D_{k-1}))$ l'ensemble des transformations-type d'ordre $k-1$ de S selon $(d, (D_1, \ldots, D_{k-1}))$;
- $Q \in \text{édition\_couples}\,(S, d, ())$ ssi Q est une séquence de chaînes d'édition optimales selon d transformant une chaîne de S en une autre :

$$\begin{aligned} &\left[|Q| = |S|^2\right] \\ \wedge\, &\left[\forall i, j \in \{1, \ldots, |S|\}, Q_{[(i-1)\times|S|]+j} \in S_i \rightarrow_d S_j\right], \end{aligned}$$

  où $S_i \rightarrow_d S_j$ désigne l'ensemble des chaînes d'éditions optimales selon d transformant S_i en S_j ;
- $P \in \text{édition\_couples}\,(S, d, (D_1, \ldots, D_{k-1}))$ ssi P est une séquence des chaînes d'édition optimales selon D_{k-1} transformant une chaîne de $\text{édition\_couples}\,(S, d, (D_1, \ldots, D_{k-2}))$ en une autre : soit $R = \text{édition\_couples}\,(S, d, (D_1, \ldots, D_{k-2}))$:

$$\left[|P| = |R|^2\right]$$

$$\wedge \left[\forall i, j \in \{1, \ldots, |R|\}, P_{[(i-1)\times|R|]+j} \in R_i \rightarrow_{D_{k-1}} R_j\right],$$

où $R_i \rightarrow_{D_{k-1}} R_j$ désigne l'ensemble des chaînes d'éditions optimales selon D_{k-1} transformant R_i en R_j ;

– $E \in$ édition(M, S, d) ssi E est une séquence des chaînes d'édition optimales selon d transformant M en une chaîne de S :

$$[|E| = |S|]$$
$$\wedge [\forall i \in \{1, \ldots, |S|\}, E_i \in M \rightarrow_d S_i],$$

où $M \rightarrow_d S_i$ désigne l'ensemble des chaînes d'éditions optimales selon d transformant M en S_i ;

– $E \in$ édition(T, P, D_{k-1}) ssi E est une séquence des chaînes d'édition optimales selon D_{k-1} transformant T en une chaîne de P :

$$[|E| = |P|]$$
$$\wedge \left[\forall i \in \{1, \ldots, |P|\}, E_i \in T \rightarrow_{D_{k-1}} P_i\right],$$

où $T \rightarrow_{D_{k-1}} P_i$ désigne l'ensemble des chaînes d'éditions optimales selon D_{k-1} transformant T en P_i.

Encore une fois, le plus gros problème de cette approche, outre sa complexité et le potentiel grand nombre de transformations-type d'ordre k de S selon (d, D), est la nécessité de spécifier une fonction de coût pour chacun des ordres, c'est-à-dire que nous devons spécifier, par exemple, pour l'ordre 2, quel est le coût d'insérer telle suppression de telle substitution (sur A), et celui de substituer telle insertion de telle substitution par telle suppression de telle suppression, etc. Cette spécification est quasiment impossible pour $k > 1$, même avec de la connaissance experte donnée pour une application particulière.

3.2 Représentations vectorielles

Nous allons maintenant voir qu'une partie de la communauté travaillant principalement sur la reconnaissance de formes structurelle s'est penché sur le problème d'implantation de données structurées dans des espaces vectoriels numériques classiques. L'idée est de se détacher en partie des problèmes de complexité soulevés lors de manipulation de structures, et de pouvoir profiter d'algorithmes traditionnellement utilisés en reconnaissance de formes statistique pour la classification de données numériques, et réputés pour leurs performances.

3.2.1 Vecteurs de distance

Soit A un alphabet. Spillmann et al. [SNB+06] proposent de représenter un ensemble fini E de chaînes sur A dans un espace de distances de la manière suivante :

1. soit $n \in \mathbb{N}$ la dimension de l'espace ;
2. sélectionner un sous-ensemble $P = \{p_1, \ldots, p_n\}$ de E, l'ensemble des *prototypes* ;
3. une chaîne X de E est transformée en un vecteur de taille n en fonction de ses distances à chaque prototype : soit d une distance d'édition de chaînes sur A :

$$t(X) = (d(X, p_1), \ldots, d(X, p_n)).$$

Les variables critiques de cette méthode sont le nombre et la sélection des prototypes. Les auteurs présentent plusieurs stratégies possibles :

– éléments centraux : processus itératif qui sélectionne, à chaque itération, un nouveau prototype parmi les éléments les moins distants (médianes) de E privé de l'ensemble des prototypes déjà sélectionnés ;

- éléments en bordure : processus itératif qui sélectionne, à chaque itération, un nouveau prototype parmi les éléments les plus distants de E privé de l'ensemble des prototypes déjà sélectionnés ;
- éléments recouvrants : processus itératif qui sélectionne, à chaque itération, un nouveau prototype parmi les éléments les plus distants de l'ensemble des prototypes déjà sélectionnés, le premier prototype sélectionné étant une médiane de E ;
- k-médianes : les prototypes sont les centroïdes finaux calculés par une adaptation de l'algorithme de partitionnement *k-means* [Ste56, JMF99], où les centroïdes sont des médianes.

Les auteurs ont évalué leurs transformations sur une base de chaînes extraites du codage d'écriture manuscrite de chiffres. Ils ont comparé les performances en classification des algorithmes suivants :

- l'algorithme de recherche des k plus proches voisins (en anglais k-NN pour *k Nearest Neighbors*), très utilisé à la fois en reconnaissance de formes structurelle et statistique, car nécessitant uniquement la spécification d'une fonction de distance ;
- l'algorithme du SVM [Vap98] (de l'anglais *Support Vector Machine*), surtout utilisé en reconnaissance de formes statistique.

L'article montre que de meilleurs résultats peuvent être obtenus après transformation des chaînes dans un espace de distances.

Notons enfin que cette méthode de transformation a ensuite été directement portée aux ensembles de graphes [RNB07], pour des conclusions relativement similaires.

3.2.2 Méthode spectrale

Une approche notable pour la représentation de graphes sous forme de vecteurs de caractéristiques numériques est celle proposée par Wilson et al. [WHL05], basée sur la théorie spectrale des graphes.

La théorie spectrale des graphes est le champ d'étude s'intéressant à la caractérisation des propriétés structurelles d'un graphe par les *valeurs propres* et *vecteurs propres* de sa *matrice d'adjacence* ou de sa *matrice laplacienne*.

Soit $n \in \mathbb{N}$. La matrice d'adjacence A d'un graphe G à n sommets $v_1, \ldots, v_n$ est la matrice de taille $(n \times n)$ telle que : $\forall i, j \in \{1, \ldots, n\}$:

$$A_{i,j} = \begin{cases} 1 & \text{si } (v_i, v_j) \in e(G), \\ 0 & \text{sinon,} \end{cases}$$

et la matrice laplacienne L de G est la matrice diagonale de taille $(n \times n)$ telle que : $\forall i, j \in \{1, \ldots, n\} | i \neq j$:

$$L_{i,i} = d_{out}(v_i) - A_{i,i},$$
$$L_{i,j} = -A_{i,j}.$$

L possède n valeurs propres $\lambda_1, \ldots, \lambda_n$, et n vecteurs propres $e_1, \ldots, e_n$ de taille n, vérifiant l'équation suivante : $\forall i \in \{1, \ldots, n\}$:

$$L \times e_i = \lambda_i \times e_i,$$

et la *matrice spectrale* Φ de L est finalement définie comme la matrice de taille $(n \times n)$ telle que : $\forall i, j \in \{1, \ldots, n\}$:

$$\Phi_{i,j} = \sqrt{\lambda_j} \times (e_j)_i.$$

Wilson et al. [WHL05] ont proposé de définir un ensemble de valeurs numériques caractérisant G à partir d'un ensemble de polynômes symétriques dérivés de sa matrice spectrale Φ, c'est-à-dire des polynômes possédant des propriétés d'invariance à la permutation de ses variables. Il en résulte que ces polynômes ont la même valeur quelle que soit l'indexation des sommets de G considérée $(v_1, \ldots, v_n)$. Ces polynômes peuvent exister en grand nombre, cela étant fonction du nombre de sommets de G, et Wilson et al. ont utilisé diverses méthodes de réduction de dimension (PCA, MDS [Fod02], LPP [HN03]) pour implanter des ensembles de graphes dans des espaces vectoriels

numériques de dimension raisonnable à partir des caractéristiques extraites des polynômes symétriques de chacun de ces graphes. Il est montré dans cet article que leur approche de caractérisation d'ensembles de graphes se montre assez cohérente, en ce sens où les rapports de distance entre deux graphes sont assez bien respectés après transformation sous forme de vecteurs de caractéristiques (plus la distance d'édition entre deux graphes est grande et plus la distance euclidienne entre les vecteurs représentant ces deux graphes est grande). De plus, les auteurs nous montrent que cette cohérence pourrait ainsi permettre d'utiliser des algorithmes de classification et de partitionnement de vecteurs numériques pour le traitement des bases de données de graphes, après une phase initiale de transformation de ces graphes. Les auteurs ont obtenus leurs résultats sur des bases de données d'images de voitures, chiens, mains, et maisons.

3.2.3 Méthodes à noyau

Les machines à noyau [HSS08] représentent une famille de séparateurs linéaires très utilisés en reconnaissance de formes ou autres domaines applicatifs quand on cherche à classer en deux catégories des ensembles de données qui n'ont pas nécessairement de représentation naturelle sous forme de vecteurs de caractéristiques numériques, comme les données structurées par exemple.

Un noyau est défini comme un produit scalaire sur un espace intermédiaire :

Définition 51 (Noyau)
Soit E un ensemble. Un noyau k sur E est une fonction $E^2 \to \mathbb{R}$ satisfaisant la condition suivante : $\forall x, y \in E$:

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle,$$

avec $\langle \cdot, \cdot \rangle$ le produit scalaire d'un espace préhilbertien (cf. définitions 100 et 101) sur un ensemble I, et Φ une fonction $E \to I$.

Soit F un sous-ensemble fini de E. Les machines à noyau, dont la plus connue est le SVM [Vap98], ne nécessitent que les valeurs de k pour tous les couples d'éléments de F, en plus des étiquettes de classe (0 ou 1) des éléments de F, pour pouvoir, si possible, séparer linéairement F dans l'espace préhilbertien intermédiaire, c'est-à-dire trouver une fonction $f : E \to \mathbb{R}$ de la forme suivante :

$$[\exists a \in E, \exists b \in \mathbb{R} | \forall x \in E, f(x) = k(x, a) + b] \qquad \text{(linéarité)}$$
$$\wedge \left[\forall y \in F, f(y) \begin{cases} \leqslant 0 & \text{si } y \text{ est étiqueté 0,} \\ > 0 & \text{sinon.} \end{cases} \right] \qquad \text{(séparation)}$$

La fonction intermédiaire Φ peut être implicite, à savoir que k peut avoir une expression analytique sous une forme plus directe qu'un produit scalaire dans l'espace intermédiaire, et les méthodes à noyau sont également capables de trouver une expression directe de f, en général comme une combinaison linéaire de valeurs de k en F : $\forall x \in E$:

$$f(x) = \sum_{y \in F} \left(\alpha(y) \times k(x, y) \right),$$

avec $\alpha : F \to \mathbb{R}$.

Les méthodes à noyau sont donc très attractives pour les données structurées complexes, car dans certaines situations le fait de pouvoir calculer k puis f sans explicitement recourir au passage par Φ peut apporter un gain de temps considérable. Le lecteur peut se référer à l'étude de Gärtner [Gär03] pour avoir un aperçu de certaines fonctions noyau qui peuvent être utilisées pour la classification de structures.

Neuhaus et Bunke [NB06] ont défini un ensemble de noyaux pour données structurées basés sur la distance d'édition. Dans ce cas, E est un ensemble de chaînes ou de graphes, et la valeur de k est fonction de la distance entre les deux structures considérées et de leurs distances respectives aux structures de F. Aucun produit scalaire de valeurs de Φ n'est calculé, l'espace intermédiaire reste donc implicite, mais le problème est que même dans sa forme directe, k nécessite le calcul d'un nombre potentiellement grand de distances d'édition, ce qui peut s'avérer assez coûteux en

termes de temps, surtout pour le cas des graphes, même avec un algorithme approximatif. Le point positif est la possibilité d'utiliser l'algorithme performant du SVM, qui nécessite uniquement les valeurs de k pour inférer une séparation linéaire pour chacune des paires de classes considérées, et les auteurs ont montré de bons résultats sur des bases de données très diverses d'objets codés sous forme de chaînes (dessins de chiffres, images de morceaux de poulets, d'outils, et de chromosomes) ou graphes (dessins de lettres, images d'empreintes digitales et de diatomées).

Enfin, l'utilisation de machines à noyau n'est pas l'unique possibilité intéressante offerte par k pour la classification de données structurées. Un autre usage intermédiaire de k peut être dans le but de définir une distance entre structures, et donc un espace métrique de structures. En effet, k induit la distance d sur E suivante : $\forall x, y \in E$:

$$d(x, y) = \sqrt{k(x, x) + k(y, y) - 2 \times k(x, y)}.$$

D'autres distances plus ou moins sophistiquées peuvent être définies sous certaines conditions à vérifier par le noyau k. Un article de Schölkopf [Sch00] fournit un aperçu de ce genre de conditions, et des distances pouvant alors être définies à partir de k.

Mais alors, si E est un ensemble de structures, nous pouvons utiliser les définitions exposées en section 3.1 pour caractériser statistiquement F selon d. Cependant, ce choix ne semble pas pertinent si l'on dispose déjà d'une distance satisfaisante entre structures, l'utilisation de cette dernière étant alors certainement plus appropriée que celle induite par k. En effet, le passage par un espace vectoriel intermédiaire, et donc une transformation, explicite ou implicite, des structures en vecteurs par Φ, induit souvent une perte d'une partie de l'information structurelle, ce qui est rarement désirable. De même, d étant toujours au moins aussi complexe à calculer que k, son utilisation entraîne les mêmes problèmes de complexités que ceux pouvant être soulevés par k. Un exemple de mauvaise utilisation de la définition de d ci-dessus serait par exemple faite dans le cas où k est le noyau défini dans l'article de Neuhaus et Bunke [NB06] cité précédemment dans cette même section. En effet, se cumuleraient alors les deux effets négatifs dont nous venons de parler :

- d est définie selon k, k est un produit scalaire de vecteurs résultant de la transformation par Φ de structures, et k est défini selon une distance d'édition ; il est alors plus judicieux d'utiliser directement la distance d'édition qui tient bien plus en compte l'information structurelle que ne le fait d, cette dernière étant soumise à une perte d'information due à Φ ;
- k est soumis à des calculs de distance d'édition, et d est soumise à des calculs de k ; le problème de complexité de la distance d'édition n'est donc pas résolu par d.

Pour résumer, si E est un ensemble de structures, alors l'utilisation de d peut être un choix pertinent uniquement si nous ne disposons pas déjà d'une distance sur E moins ou aussi complexe que k.

3.3 Conclusion

Nous avons vu différentes approches pour caractériser statistiquement des ensembles de structures discrètes.

Ces approches peuvent êtres rangées dans deux catégories : certaines tentent d'imiter directement le modèle numérique classique par le biais du concept de distance, et d'autres se basent sur un changement de représentation, plus précisément sur une transformation, implicite ou explicite, sous forme de vecteurs numériques. Les méthodes de la première catégorie sont limitées par la complexité intrinsèque des structures de données, alors que celles de la deuxième catégorie sont limitées à une utilisation intermédiaire pour des applications telles que la classification, et souffrent d'un manque de pouvoir caractéristique dans l'espace initial des structures. Une approche hybride, c'est-à-dire une représentation des structures dans un espace vectoriel numérique directement dérivé de l'espace originel, pourrait certainement posséder les avantages des deux catégories. Nous reparlerons de ce sujet plus tard (section 6.6), et discuterons de ses liens fondamentaux avec le théorème central limite.

Chapitre 4

Modèles probabilistes

Dans ce chapitre, nous allons passer en revue les principales approches utilisées en reconnaissance de formes pour modéliser des distributions de structures.

4.1 HMM

Le *modèle de Markov caché* (HMM de l'anglais *Hidden Markov Model*) est un objet permettant de représenter une classe particulière de distributions de séquences, qui peuvent être par exemple des chaînes sur un alphabet.

Le HMM a été initialement introduit dans les années 1960 par Baum et Petrie [BP66], et cette présentation s'appuie en partie sur le tutoriel de Rabiner [Rab89].

4.1.1 Chaîne de Markov

Définition 52 (Chaîne de Markov)
Soient S un ensemble fini, et $n \in \mathbb{N} \setminus \{0\}$. Une chaîne de Markov sur S est une séquence $X_1, \ldots, X_n$ de variables aléatoires à valeurs dans l'espace d'états $(S, 2^S)$, et telle que : $\forall i \in \{1, \ldots, n-1\}, \forall s_1, \ldots, s_{i+1} \in S$:

$$\mathrm{P}(X_{i+1} = s_{i+1} | X_1, \ldots, X_i = s_1, \ldots, s_i) = \mathrm{P}(X_{i+1} = s_{i+1} | X_i = s_i),$$

avec :

$$\mathrm{P}(X_{i+1} = s_{i+1} | X_1, \ldots, X_i = s_1, \ldots, s_i) = \frac{\mathrm{P}(X_1, \ldots, X_i, X_{i+1} = s_1, \ldots, s_i, s_{i+1})}{\mathrm{P}(X_1, \ldots, X_i = s_1, \ldots, s_i)},$$
$$\mathrm{P}(X_{i+1} = s_{i+1} | X_i = s_i) = \frac{\mathrm{P}(X_i, X_{i+1} = s_i, s_{i+1})}{\mathrm{P}(X_i = s_i)}.$$

Selon une chaîne de Markov, la réalisation de la prochaine variable aléatoire dépend donc uniquement de la valeur de la variable aléatoire courante. Par analogie avec le vocabulaire temporel, on dit que l'état du « futur proche » est uniquement dépendant de celui du « présent », et pas du tout des états « passés ».

4.1.2 Modèle de Markov caché

Le modèle de Markov caché est une structure mathématique couramment utilisée pour représenter un ensemble de chaînes de Markov.

Définition 53 (Modèle de Markov caché)
Soient $n, m \in \mathbb{N}$. Un HMM est un quintuplet (S, O, A, B, π), avec :
- $S = \{s_1, \ldots, s_n\}$ l'ensemble des états ;

- $O = \{o_1, \ldots, o_m\}$ l'ensemble des observations ;
- A une matrice de taille $(n \times n)$ représentant la probabilité des transitions entre états : $\forall i \in \{1, \ldots, n\}$:
$$(\forall j \in \{1, \ldots, n\}, A_{i,j} \in [0, 1]) \wedge \left(\sum_{j=1}^{n} A_{i,j} = 1 \right) ;$$
- B une matrice de taille $(n \times m)$ représentant la probabilité des observations à chaque état : $\forall i \in \{1, \ldots, n\}$:
$$(\forall j \in \{1, \ldots, m\}, B_{i,j} \in [0, 1]) \wedge \left(\sum_{j=1}^{m} B_{i,j} = 1 \right) ;$$
- π un n-uple représentant la probabilité d'initialiser le processus à un état particulier :
$$(\forall i \in \{1, \ldots, n\}, \pi_i \in [0, 1]) \wedge \left(\sum_{i=1}^{n} \pi_i = 1 \right).$$

Génération

Soient $n, m \in \mathbb{N}$, $T \in \mathbb{N} \setminus \{0\}$. Un HMM $H = (S = \{s_1, \ldots, s_n\}, O = \{o_1, \ldots, o_m\}, A, B, \pi)$ peut être utilisé pour générer une séquence d'observations $V_1, \ldots, V_T$, de la manière suivante :

1. soit $t = 1$;
2. choisir un état initial s_i selon π $(i \in \{1, \ldots, n\})$;
3. choisir V_t selon B_i ;
4. choisir l'état s_j vers lequel transiter selon A_i $(j \in \{1, \ldots, n\})$;
5. soient $t = t + 1$ et $i = j$;
6. si $t \leqslant T$, alors retourner à l'étape 3.

La séquence générée est sous-jacente à une séquence d'états Q de longueur T, distribuée selon la chaîne de Markov de même longueur représentée par H. A chaque position t, V_t est distribuée selon la probabilité des observations associée à l'état Q_t, mais on ne peut pas connaître Q_t à partir de V_t, et c'est pour cette raison que l'on dit que les états sont *cachés*.

Probabilité

Étant donnés une séquence V d'observations et un HMM H, comment calculer la probabilité de V selon la chaîne de Markov de longueur $|V|$ représentée par H ?

Cette probabilité est égale à la somme, sur l'ensemble des séquences Q de longueur $|V|$ d'états de H, des probabilités de V selon Q, c'est-à-dire les probabilités de générer V à partir des états de Q et des transitions entre ces états. Elle peut être calculée de manière efficace grâce à une procédure dynamique nommée *forward*, qui s'appuie sur un schéma récursif. Le lecteur peut se référer à l'article de Rabiner [Rab89] pour plus de détails.

États optimaux

Étant donnés une séquence V d'observations et un HMM H, quelle est la séquence Q d'états de H (de longueur $|V|$) qui maximise la probabilité de V selon Q ?

L'algorithme généralement utilisé pour résoudre ce problème est l'algorithme dynamique de *Viterbi* [Vit67, For73], qui s'appuie sur un schéma récursif. Le lecteur peut se référer à l'article de Rabiner [Rab89] pour plus de détails.

Optimisation du modèle

Comment ajuster les probabilités d'états initiaux, de transitions entre états, et d'observations à chaque état d'un HMM H, pour maximiser la probabilité d'une séquence d'observations V selon la chaîne de Markov de longueur $|V|$ représentée par H ?

Les valeurs de ces probabilités sont le plus souvent mises à jour avec des proportions calculées empiriquement en fonction de H et de V, ce à l'aide de procédures nommées *forward* et *backward*, qui s'appuient sur des schémas récursifs. Le lecteur peut se référer à l'article de Rabiner [Rab89] pour plus de détails.

4.1.3 Champs d'application

Le HMM est devenu particulièrement populaire pour des applications telles que la reconnaissance de la parole [Rab89], la reconnaissance de caractères dans des images [VK92], la prédiction de séquences génétiques en bio-informatique [KKS05] etc. La raison est qu'il est riche en structures mathématiques, et donne de bons résultats en pratique, grâce aux nombreux algorithmes existant permettant de résoudre de manière efficace les principaux problèmes liés à son utilisation (probabilité, génération, mise à jour des paramètres).

4.2 Grammaire stochastique

La grammaire stochastique est un autre modèle générateur de chaînes, dédié à la modélisation de la syntaxe dans la formation des phrases selon un langage formel, en tentant d'imiter les caractéristiques propres à des langages naturels plus ou moins évolués.

4.2.1 Langage stochastique

Dans le vocabulaire de la théorie des langages formels, les termes « chaîne » et « mot » sont synonymes, et il en est de même pour les termes « ensemble de chaînes » et « langage », ou encore pour les termes « distribution de chaînes » et « langage stochastique ».

Définition 54 (Langage)
Soit A un alphabet. Un langage (formel) sur A est un sous-ensemble de $\mathbb{S}(A)^*$.

Définition 55 (Langage stochastique)
Soit A un alphabet. Un langage stochastique sur A est une distribution sur l'ensemble puissance de $\mathbb{S}(A)^*$.

Soit P un langage stochastique sur A. Nous avons :

$$\sum_{X \in \mathbb{S}(A)^*} \mathrm{P}(X) = 1.$$

4.2.2 Grammaire et automate stochastique

Une grammaire stochastique est une structure mathématique représentant un langage stochastique.

Définition 56 (Grammaire stochastique)
Soit $n \in \mathbb{N} \setminus \{0\}$. Une grammaire stochastique est un quadruplet (A, V, P, s), avec :
- A l'alphabet des terminaux ;
- $V = \{v_1, \ldots, v_n\}$ l'alphabet des non-terminaux, tel que $A \cap V = \{\}$;
- $P = \{\mathrm{P}_1, \ldots, \mathrm{P}_n\}$ l'ensemble des probabilités des règles de production : $\forall i \in \{1, \ldots, n\}$, P_i est un langage stochastique sur $\mathbb{S}(A \cup V)^*$:

$$\sum_{X \in \mathbb{S}(A \cup V)^*} \mathrm{P}_i(X) = 1,$$

  et $\mathrm{P}_i(X)$ est la probabilité de produire la chaîne X de terminaux et non-terminaux en consommant le non-terminal v_i ;
- s le non-terminal initial.

Un cas particulier de grammaire stochastique est la grammaire stochastique *régulière*, ou *automate stochastique*, pour laquelle il est improbable de produire, à partir d'un non-terminal, des chaînes qui ne soient pas de l'une des trois formes suivantes : vide, composée uniquement d'un terminal, composée d'un terminal suivi d'un non-terminal.

Définition 57 (Grammaire stochastique régulière)
Soit $n \in \mathbb{N} \setminus \{0\}$. Une grammaire stochastique $(A, V = \{v_1, \ldots, v_n\}, P = \{\mathrm{P}_1, \ldots, \mathrm{P}_n\}, s)$ est dite régulière ssi : $\forall i \in \{1, \ldots, n\}$:

$$\mathrm{P}_i(\lambda) + \sum_{l \in A} \mathrm{P}_i(l) + \sum_{l \in A} \sum_{j=1}^{n} \mathrm{P}_i(l.v_j) = 1.$$

Dès lors, un langage stochastique est dit *régulier* ssi il peut être représenté par une grammaire stochastique régulière.

Génération

Soit $n \in \mathbb{N} \setminus \{0\}$. Une grammaire stochastique $G = (A, V = \{v_1, \ldots, v_n\}, P = \{\mathrm{P}_1, \ldots, \mathrm{P}_n\}, s)$ peut être utilisée pour générer une chaîne X sur A, de la manière suivante :

1. soit $X = s$.
2. soient $i \in \{1, \ldots, n\}$, $Y \in \mathbb{S}(A)^*$, et $Z \in \mathbb{S}(A \cup V)^*$, tels que $X = Y.v_i.Z$;
3. choisir W selon P_i ;
4. soit $X = Y.W.Z$;
5. si X n'est pas composée uniquement de terminaux, alors retourner à l'étape 2.

La chaîne générée est sous-jacente au langage stochastique sur A représenté par G. La procédure ci-dessus est un *parsing* de G, c'est-à-dire une production d'une chaîne de terminaux à partir de son non-terminal initial et de ses règles de production. La probabilité d'un tel *parsing* est égale à la multiplication des probabilités des règles de production utilisées (les $\mathrm{P}_i(W)$ dans la procédure ci-dessus).

Probabilité

Étant données une grammaire stochastique $G = (A, V, P, s)$ et une chaîne X sur A, comment calculer la probabilité de X selon le langage stochastique représenté par G ?

Cette probabilité est égale à la somme des probabilités de tous les *parsing*s de G produisant X. Elle peut être obtenue grâce à une procédure dynamique *forward* [Laf99] similaire à celle utilisée pour le calcul d'une probabilité selon un HMM (cf. section 4.1.2).

Parsing optimal

Étant données une grammaire stochastique $G = (A, V, P, s)$ et une chaîne X sur A, quel est le *parsing* le plus probable de G produisant X ?

Ce résultat peut être obtenu grâce à une adaptation de l'algorithme de Viterbi [Vit67, For73, Laf99], utilisé pour calculer la séquence d'états la plus probable produisant une séquence d'observations selon un HMM (cf. section 4.1.2).

Apprentissage

Comment apprendre une grammaire $G = (A, V, P, s)$ représentant une estimation d'un langage stochastique L sur A, à partir d'une séquence S de chaînes supposées distribuées selon L ?

Pour le cas des automates, l'algorithme pionnier est *Alergia*, développé par Carrasco et Oncina [CO94], et de nombreuses améliorations et/ou optimisations de cet algorithme furent proposées par la suite, comme par exemple l'algorithme MDI [TDdlH00]. Ces algorithmes fonctionnent selon le même principe : un automate est codé sous forme d'une *machine à états* [VTdlH$^+$05a, VTdlH$^+$05b], c'est-à-dire un graphe où les sommets sont des états, et les arêtes des transitions entre états. Chaque

état est étiqueté par un non-terminal, et chaque arête par un terminal, cela étant cohérent au vu du fait que toutes les productions à probabilité non nulle d'un automate ne génèrent qu'au plus un seul terminal, et que celles en générant effectivement un génèrent également un seul non-terminal, à la suite. Ces algorithmes initialisent l'automate avec une machine représentant exactement la probabilité représentée par S, puis itèrent une série de fusions successives d'états de l'automate, en fonction d'un compromis entre généralisation (pouvoir affecter une probabilité non nulle à des chaînes n'appartenant pas à S) et consistance (ne pas trop diverger de la probabilité représentée par S).

Pour ce qui est des grammaires qui ne sont pas des automates, le problème est plus difficile et ce procédé de codage sous forme de machine à états n'est pas possible. Si l'on connaît l'ensemble des règles de production à probabilité non nulle, l'algorithme *inside-outside* [LY90] peut être utilisé pour fixer ces probabilités. Sinon, une approche génétique [KB96] permet d'inférer directement les distributions des règles.

4.2.3 Champs d'application

L'inférence grammaticale stochastique (champ d'étude visant à l'apprentissage de grammaires stochastiques pour la classification) a été porteuse de succès sur de nombreuses applications en reconnaissance de formes, linguistique, bio-informatique etc. Le lecteur peut se référer à l'étude de de la Higuera [dlH05] pour de plus amples détails et pour avoir une séries de pointeurs sur ce sujet.

De manière générale, il semble logique que la grammaire stochastique doit s'avérer performante dès lors qu'il est pertinent de modéliser la formation des données considérées par des règles syntaxiques, comme cela est le cas en reconnaissance de la parole [Jel98] ou en classification de genres musicaux [CAV98]. Le plus gros problème est la difficulté générale d'apprentissage d'une grammaire stochastique, d'où le recours récurrent à des modèles moins puissants mais moins complexes, tels que l'automate stochastique ou le HMM.

4.3 Graphe aléatoire

Le graphe aléatoire [WCY90] est une structure représentant une distribution de graphes. Elle est similaire à celle de graphe, mais les sommets et les arêtes ne sont pas étiquetées par une valeur prise dans un alphabet mais par une variable aléatoire à valeurs dans un alphabet.

4.3.1 Graphe aléatoire et dérivés

Définition 58 (Graphe aléatoire)
Soient A_v et A_e deux alphabets, auxquels nous associons λ en qualité de lettre vide ($\lambda \notin (A_v \cup A_e)$), et Ω_v et Ω_e deux ensembles de variables aléatoires à valeurs dans les espaces d'états $\left(A_v \cup \{\lambda\}, 2^{A_v \cup \{\lambda\}}\right)$ et $\left(A_e \cup \{\lambda\}, 2^{A_e \cup \{\lambda\}}\right)$, respectivement. Un graphe aléatoire RG (étiqueté) sur (Ω_v, Ω_e) est un couple $(rvl(G), rel(G))$, avec $rvl(G)$ une fonction $rv(G) \to \Omega_v$, et $rel(G)$ une fonction $re(G) \to \Omega_e$, telles que $rv(G)$ est un ensemble fini d'éléments de $\mathbb{N}$ nommés sommets, et $re(G)$ un ensemble d'éléments de $rv(G)^2$ nommés arêtes.

Probabilités, appariement

Étant donné un graphe G réalisation d'un graphe aléatoire RG, la probabilité par RG d'obtenir un graphe H isoétiqueté à G est fonction d'une somme de probabilités jointes des variables aléatoires étiquetant les sommets et arêtes de RG, cela sur l'ensemble des possibilités de réaliser G à partir de RG [SAS02a].

Ce genre de constat est un exemple tendant à conclure que le graphe aléatoire est un modèle trop complexe, inutilisable en pratique. Le graphe aléatoire *de premier ordre* [WCY90] et le graphe *décrit par une fonction* [SAS03] sont deux cas spéciaux de graphe aléatoire. Ils satisfont certaines contraintes d'indépendance, entre autres, entre deux variables aléatoires étiquetant un sommet, deux variables aléatoires étiquetant une arête, ou encore une variable aléatoire étiquetant un sommet et une variable aléatoire étiquetant une arête. Ces conditions permettent de simplifier considérablement les calculs de probabilités, que ce soit les probabilités jointes de deux graphes,

ou la probabilité d'un graphe, selon la distribution représentée par un graphe aléatoire. Le graphe aléatoire *de second ordre* [SAS02a] est une autre approximation de graphe aléatoire, proposée comme une généralisation des deux approches précitées, tout en préservant certaines exigences de simplicité.

Une autre manière de mesurer l'appariement d'un graphe à un modèle aléatoire, plutôt que de calculer la probabilité du graphe selon le modèle, ce qui est souvent trop coûteux en termes de temps, est de calculer une dissimilarité entre le graphe et le modèle, basée sur les variables aléatoires composant le modèle, et ne prenant donc en compte qu'une partie de l'information probabiliste. Le tout est d'établir un bon compromis entre d'une part, augmenter la rapidité du calcul de l'appariement, et d'autre part, ne pas trop augmenter son incertitude. Des exemples typiques de tels appariements approximatifs, pour d'autres types de structures, sont ceux que nous avons vu précédemment pour les HMM (resp. grammaires stochastiques) où la probabilité d'une séquence d'observations (resp. chaîne) est souvent approximée en pratique par la probabilité de la séquence d'états la (resp. du parsing le) plus probable, pour un gain de temps non négligeable et une faible augmentation de l'incertitude.

Quelques appariements approximatifs entre graphe et graphe aléatoire, basés sur l'information locale fournie par les variables aléatoires étiquetant les sommets et arêtes du graphe aléatoire, ont été proposés dans la littérature. Alquézar et al. [ASS00] ont défini une telle dissimilarité pour le cas de graphe décrit par une fonction, puis Sanfeliu et al. [SSA04] se sont attaqués au cas plus général de graphe aléatoire de second ordre.

Apprentissage

Comment construire un graphe aléatoire RG représentant une estimation de la distribution supposée sous-jacente à une séquence S de taille finie et non vide de graphes ?

Sanfeliu et al. [SSA04] ont mis au point un algorithme incrémental d'apprentissage de graphe aléatoire de second ordre, qui est une généralisation d'un algorithme initialement proposé pour le cas plus spécial de fonction décrit par un graphe [SAS02b]. L'algorithme est paramétré par une fonction d'appariement entre un graphe et un graphe aléatoire de type concerné, appariement forcément approximatif pour les même raisons de complexité que celles citées précédemment. Le graphe aléatoire RG à retourner est initialisé à partir du premier graphe S_1 de la séquence en entrée, puis s'en suit une série d'itérations, où chacune consiste en l'« intégration » à RG d'un graphe de S, jusqu'à ce que tous les graphes de S aient été intégrés. L'intégration d'un graphe G à RG consiste en la modification des variables aléatoires étiquetant les sommets et arêtes de RG, de manière à tenir compte de la nouvelle information probabiliste fournie par les valeurs des sommets et arêtes étiquetant G. C'est à ce moment que les auteurs utilisent la fonction d'appariement. G est apparié à RG, en est déduit une mise en correspondance entre les composantes aléatoires de RG et les composantes de G, et finalement entre les composantes aléatoires de RG et celles du graphe aléatoire R déduit directement de G. Grossièrement, il reste à « fusionner » les variables aléatoires de RG et de R correspondant par l'appariement, pour en déduire le nouveau graphe aléatoire. Cette fusion est estimée de manière dépendante du type de modèle considéré, c'est-à-dire en fonction des suppositions d'indépendance faites entre les variables aléatoires composant le modèle.

4.3.2 Champs d'application

Le graphe décrit par une fonction a été utilisé pour une application de partitionnement d'ensemble de graphes générés aléatoirement [SAS02b]. Le partitionnement est effectué de manière hiérarchique agglomérative [JMF99], un groupe de graphes étant représenté par un graphe aléatoire, et les deux groupes les plus proches étant déterminés par une fonction de dissimilarité entre graphes aléatoires. La fusion de deux groupes résulte en la création d'un nouveau graphe aléatoire à partir des deux graphes aléatoires représentant les groupes.

Le graphe décrit par une fonction a également été utilisé en reconnaissance de formes pour la classification d'images d'objets divers ou de visages [SAS03], ainsi que pour la classification d'images de lettres majuscules de l'alphabet dessinées à la main [SSA08]. Dans ces articles, les auteurs extraient un graphe par image, et apprennent un graphe aléatoire pour chaque classe, une image de classe inconnue étant ensuite affectée à la classe pour laquelle la dissimilarité entre son

graphe extrait G et le modèle aléatoire de la classe est la plus faible. Les auteurs montrent que passer par la synthèse d'un graphe aléatoire pour chaque classe permet d'améliorer les résultats en comparaison avec une classification au plus proche voisin, c'est-à-dire affecter directement une image à la même classe que celle du graphe avec lequel G est le moins dissimilaire.

Enfin, le même type d'expérimentation de classification a été mené pour évaluer la performance de la modélisation d'ensembles de graphes par un graphe aléatoire de second ordre [SSA04]. Les auteurs utilisent des bases de données, générées synthétiquement, d'images de vues en 2 dimensions d'objets à 3 dimensions, et comparent les résultats obtenus par le graphe aléatoire de second ordre, le graphe aléatoire de premier ordre, le graphe décrit par une fonction , et l'approche directe, c'est-à-dire sans modélisation probabiliste mais en classifiant en fonction du graphe le moins dissimilaire. Les auteurs montrent que le graphe aléatoire de second ordre s'avère le plus performant, et que son utilisation reste raisonnable au vu des temps de calcul.

4.4 Conclusion

Nous avons eu un aperçu des méthodes les plus utilisées en reconnaissance de formes structurelle pour la modélisation de distributions de structures discrètes.

Ces modèles ont été mis au point pour une utilisation générique, c'est-à-dire pour tenter de résoudre des problèmes de classification, entre autre, par l'intermédiaire de l'estimation ou apprentissage d'une distribution, sans supposition aucune d'éventuelles caractéristiques ou propriétés que pourrait ou devrait posséder cette distribution, autre que la caractéristique de pouvoir être représentée ou au moins correctement approximée par le modèle considéré.

Dans les deux chapitres suivants de cette thèse, nous n'allons pas nous atteler à définir de nouveaux modèles, mais plutôt à définir de nouvelles classes de distributions de structures, à savoir les distributions uniformes et normales. Le lecteur pourra se rendre compte que ces distributions peuvent très bien être représentées par les modèles précités (même si cela est inutile), mais dans ce cas les paramètres des modèles sont fixés par ceux des distributions, et il ne nous faut pas utiliser les algorithmes génériques d'estimation des paramètres de ces modèles, mais des algorithmes spécifiques d'estimation de ceux des distributions, si l'on veut apprendre une distribution supposée appartenir à une certaine classe, c'est-à-dire respectant les caractéristiques spécifiques que doit respecter une distribution pour être membre de ladite classe.

Chapitre 5

Distribution uniforme

Dans ce chapitre, nous définissons le concept de distribution uniforme de structures.

La distribution uniforme est spécifiée dans un ensemble S, en fonction d'une mesure μ sur S, par la probabilité P respectant comme critère le fait que tout sous-ensemble E de S, mesurable par μ, a une probabilité proportionnelle à sa mesure : $\mathrm{P}(E) = Z \times \mu(E)$. Z est donc l'inverse de la mesure totale de S, et P n'est rien d'autre qu'une simple normalisation de μ. Deux exemples bien connus d'une telle spécification sont donnés dans les cas numériques unidimensionnels discrets et continus. Pour le cas discret, la mesure est la cardinalité, ce qui implique que tout ensemble de même cardinal a la même probabilité. Pour le cas continu, la mesure est celle de Lebesgue, et cela implique que tout intervalle de même longueur à la même probabilité.

Nous nous intéressons à la spécification de cette distribution pour le cas d'ensembles de structures. Pour chaque type structurel considéré, nous proposons une définition de mesure respectant sa combinatoire inhérente, afin de définir une distribution uniforme pour ce type. Nous nous intéressons aux cas des chaînes, multiensembles, arbres et graphes, dans les sections 5.2, 5.3, 5.4 et 5.5, respectivement.

5.1 Uniformité

Une définition générale de probabilité uniforme est la suivante :

Définition 59 (Distribution uniforme)
Soit (S, σ, μ) un espace mesuré, tel que $0 < \mu(S) < \infty$. La distribution uniforme selon μ est la probabilité P sur σ définie comme suit : $\forall E \in \sigma$:

$$\mathrm{P}(E) = \mu(E) \times \mu(S)^{-1}.$$

Exemple
Soient S un ensemble fini de nombres réels, $\sigma = 2^S$, et μ la cardinalité (qui est une mesure sur toute tribu) sur σ. La distribution uniforme selon μ est classiquement connue sous le nom de *loi uniforme discrète sur S*.

5.2 Chaînes

Soient A un alphabet, auquel nous associons λ en qualité de lettre vide, et $n \in \mathbb{N}$. Dans cette section, nous supposons la donnée d'une mesure a sur l'ensemble puissance de $A \cup \{\lambda\}$, et proposons la définition de la mesure s^n sur l'ensemble puissance de $\mathbb{S}(A)^{\leqslant n}$, et sa normalisation résultant en la distribution uniforme selon s^n, notée S^n. Nous allons définir s^n en fonction de a et de la combinatoire inhérente à la nature du type chaîne.

5.2.1 Mesure

Soit X une chaîne sur A de longueur au plus n. X peut être décrite par une chaîne λ-X sur $A \cup \{\lambda\}$ de longueur n, simplement en préservant l'ordre relatif des lettres dans la composition de X, et en complétant avec le nombre approprié de λ. Nous considérons donc X comme étant la représentation canonique de l'ensemble de ces telles chaînes λ-X par lequel elle peut être décrite grâce à une telle procédure. Ce raisonnement, ajouté à la propriété d'additivité des mesures, nous permet de définir s^n de la manière suivante :

$$\mathrm{s}^n(X) = \sum_{\lambda\text{-X} \in \lambda\text{-chaînes}^n(X)} \lambda\text{-s}^n(\lambda\text{-X}),$$

où λ-chaînes$^n(X)$ désigne l'ensemble des chaînes de $\mathbb{S}(A \cup \{\lambda\})^n$ que représente X, et λ-s^n une mesure définie sur l'ensemble puissance de $\mathbb{S}(A \cup \{\lambda\})^n$.

Soit $\lambda\text{-X} \in \lambda$-chaînes$^n(X)$. Nous considérons chaque lettre de λ-X comme son expression dans une certaine dimension, et calculons donc la mesure de λ-X simplement par multiplication des mesures par a de chacune de ses lettres, obtenant la formule suivante :

$$\lambda\text{-s}^n(\lambda\text{-X}) = \prod_{i=1}^{n} \mathrm{a}(\lambda\text{-X}(i)) = \prod_{i=1}^{|X|} \mathrm{a}(X(i)) \times \mathrm{a}(\lambda)^{n-|X|}.$$

La valeur de λ-s^n est donc la même en toute chaîne de λ-chaînes$^n(X)$. De plus, le cardinal de λ-chaînes$^n(X)$ est égal au nombre de combinaisons de taille $|X|$ d'un ensemble de cardinal n, ce qui nous permet de préciser la définition de s^n de la manière suivante :

Définition 60 (Mesure de chaînes)
$\forall X \in \mathbb{S}(A)^{\leqslant n}$:

$$\mathrm{s}^n(X) = \mathrm{C}_n^{|X|} \times \prod_{i=1}^{|X|} \mathrm{a}(X(i)) \times \mathrm{a}(\lambda)^{n-|X|}.$$

La mesure d'un ensemble E qui n'est pas un singleton est déduite de la propriété d'additivité des mesures : somme des mesures de tous les singletons qui sont sous-ensembles de E (et donc 0 pour l'ensemble vide). Il est donc inutile de prouver que s^n *est* une mesure sur l'ensemble puissance de $\mathbb{S}(A)^{\leqslant n}$.

Notons enfin que s^n peut être calculée de manière récursive, grâce à la formule ci-dessous, utile en 5.2.2 :

Proposition 1 (Récursivité de la mesure de chaînes)
$n > 0 \Longrightarrow \forall X \in \mathbb{S}(A)^{\leqslant n-1}, \forall l \in A$:

$$\mathrm{s}^n(X.l) = \mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{|X|+1}}{\mathrm{C}_n^{|X|}} \times \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)}.$$

Preuve (Proposition 1)

$$\begin{aligned}
\mathrm{s}^n(X.l) &= \mathrm{C}_n^{|X.l|} \times \prod_{i=1}^{|X.l|} \mathrm{a}((X.l)(i)) \times \mathrm{a}(\lambda)^{n-|X.l|} \\
&= \mathrm{C}_n^{|X|+1} \times \prod_{i=1}^{|X|} \mathrm{a}(X(i)) \times \mathrm{a}(l) \times \mathrm{a}(\lambda)^{n-|X|-1} \\
&= \left(\mathrm{C}_n^{|X|} \times \prod_{i=1}^{|X|} \mathrm{a}(X(i)) \times \mathrm{a}(\lambda)^{n-|X|} \right) \times \frac{\mathrm{C}_n^{|X|+1}}{\mathrm{C}_n^{|X|}} \times \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)} \\
&= \mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{|X|+1}}{\mathrm{C}_n^{|X|}} \times \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)}.
\end{aligned}$$

■

Nous imposons $n > 0$ car l'ensemble $\mathbb{S}(A)^{-1}$ n'est pas défini ($-1 \notin \mathbb{N}$).

5.2.2 Probabilité

D'après le critère général d'uniformité (cf. définition 59) selon la mesure s^n, la probabilité d'une chaîne est donnée par la formule suivante : $\forall X \in \mathbb{S}(A)^{\leqslant n}$:

$$\mathrm{S}^n(X) = \mathrm{s}^n(X) \times \mathrm{s}^n\left(\mathbb{S}(A)^{\leqslant n}\right)^{-1}.$$

Il serait bien trop complexe de calculer la mesure totale de $\mathbb{S}(A)^{\leqslant n}$ par une naïve énumération récursive de toutes les chaînes de cet ensemble, procédure exigeant un temps de calcul exponentiel en n. Bien heureusement, nous pouvons obtenir cette valeur plus simplement, grâce à l'égalité suivante :

Proposition 2 (Mesure totale de chaînes)

$$\mathrm{s}^n\left(\mathbb{S}(A)^{\leqslant n}\right) = \mathrm{a}(A \cup \{\lambda\})^n.$$

La formule suivante va nous être utile pour prouver cette dernière proposition :

Lemme 1
$\forall i \in \{0, \ldots, n-1\}$:

$$\mathrm{s}^n\left(\mathbb{S}(A)^{i+1}\right) = \mathrm{s}^n\left(\mathbb{S}(A)^i\right) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^i} \times \frac{\mathrm{a}(A)}{\mathrm{a}(\lambda)}.$$

Dès lors, nous avons :

Preuve (Proposition 2)

$$\text{Lemme 1} \wedge \left[\mathrm{s}^n\left(\mathbb{S}(A)^0\right) = \mathrm{s}^n(\{\}) = \mathrm{a}(\lambda)^n\right] \Longrightarrow \forall i \in \{0, \ldots, n\} :$$

$$\begin{aligned}
\mathrm{s}^n\left(\mathbb{S}(A)^i\right) &= \mathrm{a}(\lambda)^n \times \prod_{j=1}^{i} \left(\frac{\mathrm{C}_n^j}{\mathrm{C}_n^{j-1}} \times \frac{\mathrm{a}(A)}{\mathrm{a}(\lambda)}\right) \\
&= \mathrm{a}(\lambda)^n \times \frac{\mathrm{a}(A)^i}{\mathrm{a}(\lambda)^i} \times \prod_{j=1}^{i} \frac{\mathrm{C}_n^j}{\mathrm{C}_n^{j-1}} \\
&= \mathrm{a}(\lambda)^{n-i} \times \mathrm{a}(A)^i \times \frac{\mathrm{C}_n^i}{\mathrm{C}_n^0} \\
&= \mathrm{a}(\lambda)^{n-i} \times \mathrm{a}(A)^i \times \mathrm{C}_n^i.
\end{aligned}$$

Nous avons alors :

$$\begin{aligned}
\mathrm{s}^n\left(\mathbb{S}(A)^{\leqslant n}\right) &= \sum_{i=0}^{n} \mathrm{s}^n\left(\mathbb{S}(A)^i\right) \\
&= \sum_{i=0}^{n} \left[\mathrm{a}(\lambda)^{n-i} \times \mathrm{a}(A)^i \times \mathrm{C}_n^i\right] \\
&= {}^{\mathrm{i}}\left[\mathrm{a}(\lambda) + \mathrm{a}(A)\right]^n \\
&= \mathrm{a}(A \cup \{\lambda\})^n.
\end{aligned}$$

■

Il nous reste à prouver le lemme précédent :

i. Formule du binôme (théorème 2).

Preuve (Lemme 1)
Si $n = 0$, alors $\{0, \ldots, n-1\} = \{\}$, et par conséquent le lemme est vérifié par vacuité. Sinon ($n > 0$), nous avons :

$$\begin{aligned}
\mathrm{s}^n\left(\mathbb{S}(A)^{i+1}\right) &= \sum_{Y \in \mathbb{S}(A)^{i+1}} \mathrm{s}^n(Y) \\
&= \sum_{X \in \mathbb{S}(A)^i} \sum_{l \in A} \mathrm{s}^n(X.l) \\
&= \sum_{X \in \mathbb{S}(A)^i} \sum_{l \in A} \left(\mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{|X|+1}}{\mathrm{C}_n^{|X|}} \times \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)} \right) \\
&= \sum_{X \in \mathbb{S}(A)^i} \sum_{l \in A} \left(\mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^{i}} \times \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)} \right) \\
&= \sum_{X \in \mathbb{S}(A)^i} \left(\mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^{i}} \times \sum_{l \in A} \frac{\mathrm{a}(l)}{\mathrm{a}(\lambda)} \right) \\
&= \sum_{X \in \mathbb{S}(A)^i} \left(\mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^{i}} \times \frac{\mathrm{a}(A)}{\mathrm{a}(\lambda)} \right) \\
&= \sum_{X \in \mathbb{S}(A)^i} \mathrm{s}^n(X) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^{i}} \times \frac{\mathrm{a}(A)}{\mathrm{a}(\lambda)} \\
&= \mathrm{s}^n\left(\mathbb{S}(A)^i\right) \times \frac{\mathrm{C}_n^{i+1}}{\mathrm{C}_n^{i}} \times \frac{\mathrm{a}(A)}{\mathrm{a}(\lambda)}.
\end{aligned}$$

■

D'après la définition 60 et la proposition 2, nous pouvons calculer la probabilité d'une chaîne en $O(n \times \alpha)$, avec $O(\alpha)$ la complexité d'une mesure selon a :

Définition 61 (Distribution uniforme de chaînes)
$\forall X \in \mathbb{S}(A)^{\leqslant n}$:

$$\mathrm{S}^n(X) = \mathrm{C}_n^{|X|} \times \prod_{i=1}^{|X|} \mathrm{a}(X(i)) \times \mathrm{a}(\lambda)^{n-|X|} \times \mathrm{a}(A \cup \{\lambda\})^{-n}.$$

5.2.3 Préservation

Une propriété intéressante de notre distribution uniforme de chaînes est la préservation du critère d'uniformité par concaténation : la concaténation de deux chaînes uniformes résulte en une chaîne uniforme. Nous utilisons les termes « chaîne uniforme » ou « chaîne uniformément distribuée » pour désigner une chaîne dont la probabilité est assignée par une distribution uniforme.

Proposition 3 (Préservation d'uniformité de chaînes)
$\forall i \in \{0, \ldots, n\}, \forall Y \in \mathbb{S}(A)^{\leqslant i}, \forall Z \in \mathbb{S}(A)^{\leqslant n-i}$, si Y est uniformément distribuée selon s^i, et Z uniformément distribuée selon s^{n-i}, alors $X = Y.Z$ est uniformément distribuée selon s^n.

Preuve (Proposition 3)
Tout d'abord, notons l'égalité suivante : $\forall i, k \in \{0, \ldots, n\}$:

$$\mathrm{C}_n^k = \sum_{j=0}^{k} \left(\mathrm{C}_i^j \times \mathrm{C}_{n-i}^{k-j} \right).$$

Dès lors, nous avons : $\forall X \in \mathbb{S}(A)^{\leqslant n}$:

$$\mathrm{C}_n^{|X|} = \sum_{j=0}^{|X|} \left(\mathrm{C}_i^j \times \mathrm{C}_{n-i}^{|X|-j} \right)$$

$$= \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} | Y.Z = X} \left(\mathrm{C}_i^{|Y|} \times \mathrm{C}_{n-i}^{|Z|} \right).$$

De plus, nous avons : $\forall Y \in \mathbb{S}(A)^{\leqslant i}, \forall Z \in \mathbb{S}(A)^{\leqslant n-i} | Y.Z = X$:

$$\begin{aligned}
\mathrm{S}^i(Y) \times \mathrm{S}^{n-i}(Z) &= \left(\mathrm{C}_i^{|Y|} \times \prod_{j=1}^{|Y|} \mathrm{a}(Y(j)) \times \mathrm{a}(\lambda)^{i-|Y|} \times \mathrm{a}(A \cup \{\lambda\})^{-i} \right) \\
&\times \left(\mathrm{C}_{n-i}^{|Z|} \times \prod_{j=1}^{|Z|} \mathrm{a}(Z(j)) \times \mathrm{a}(\lambda)^{(n-i)-|Z|} \times \mathrm{a}(A \cup \{\lambda\})^{-(n-i)} \right) \\
&= \mathrm{C}_i^{|Y|} \times \mathrm{C}_{n-i}^{|Z|} \times \prod_{j=1}^{|Y.Z|} \mathrm{a}((Y.Z)(j)) \times \mathrm{a}(\lambda)^{n-|Y.Z|} \times \mathrm{a}(A \cup \{\lambda\})^{-n} \\
&= \mathrm{C}_i^{|Y|} \times \mathrm{C}_{n-i}^{|Z|} \times \prod_{j=1}^{|X|} \mathrm{a}(X(j)) \times \mathrm{a}(\lambda)^{n-|X|} \times \mathrm{a}(A \cup \{\lambda\})^{-n}.
\end{aligned}$$

Nous obtenons finalement l'égalité suivante :

$$\begin{aligned}
\mathrm{S}^n(X) &= \mathrm{C}_n^{|X|} \times \prod_{j=1}^{|X|} \mathrm{a}(X(j)) \times \mathrm{a}(\lambda)^{n-|X|} \times \mathrm{a}(A \cup \{\lambda\})^{-n} \\
&= \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} | Y.Z = X} \left(\mathrm{C}_i^{|Y|} \times \mathrm{C}_{n-i}^{|Z|} \right) \times \prod_{j=1}^{|X|} \mathrm{a}(X(j)) \times \mathrm{a}(\lambda)^{n-|X|} \times \mathrm{a}(A \cup \{\lambda\})^{-n} \\
&= \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} | Y.Z = X} \left[\mathrm{C}_i^{|Y|} \times \mathrm{C}_{n-i}^{|Z|} \times \prod_{j=1}^{|X|} \mathrm{a}(X(j)) \times \mathrm{a}(\lambda)^{n-|X|} \times \mathrm{a}(A \cup \{\lambda\})^{-n} \right] \\
&= \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} | Y.Z = X} \left[\mathrm{S}^i(Y) \times \mathrm{S}^{n-i}(Z) \right].
\end{aligned}$$

Nous en déduisons que quel que soit $i \in \{0, \ldots, n\}$, la probabilité d'une chaîne uniforme X selon s^n est égale à la somme des probabilités de toutes les concaténations possibles d'une chaîne uniforme Y selon s^i avec une chaîne uniforme Z selon s^{n-i}, telles que $Y.Z = X$. Il s'agit exactement de la signification de la dernière proposition. ■

Le raisonnement général ci-dessus nous conduit au résultat spécifique suivant, utile en 5.2.4 :

Corollaire 1 (Préservation d'uniformité de chaînes)
$\forall X \in \mathbb{S}(A)^{\leqslant n}, \forall l \in A \cup \{\lambda\}$, si X est uniformément distribuée selon s^n, et l uniformément distribuée selon a, alors $X.l$ est uniformément distribuée selon s^{n+1}.

La preuve du corollaire ci-dessus découle du lemme suivant :

Lemme 2
$\forall l \in A \cup \{\lambda\}$, l est uniformément distribuée selon a ssi $\{\}.l$ est uniformément distribuée selon s^1.

Dès lors, nous avons :

Preuve (Corollaire 1)
D'après le lemme 2, $\{\}.l$ est uniformément distribuée selon s^1. Alors, d'après la proposition 3, $X.(\{\}.l) = X.l$ est uniformément distribuée selon s^{n+1}. ■

Il nous reste à prouver le lemme précédent :

Preuve (Lemme 2)
Notons $\mathrm{A}(l)$ la probabilité de l selon une distribution uniforme selon a :

$$\mathrm{A}(l) = \mathrm{a}(l) \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Si $l = \lambda$, nous avons :

$$\mathrm{A}(l) = \mathrm{C}_1^0 \times 1 \times \mathrm{a}(\lambda)^{1-0} \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Sinon ($l \in A$), nous avons :

$$\mathrm{A}(l) = \mathrm{C}_1^1 \times \mathrm{a}(l) \times \mathrm{a}(\lambda)^{1-1} \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Donc dans tous les cas, nous avons :

$$\begin{aligned}\mathrm{A}(l) &= \mathrm{C}_1^{|\{\}.l|} \times \prod_{i=1}^{|\{\}.l|} \mathrm{a}((\{\}.l)(i)) \times \mathrm{a}(\lambda)^{1-|\{\}.l|} \times \mathrm{a}(A \cup \{\lambda\})^{-1} \\ &= \mathrm{S}^1(\{\}.l).\end{aligned}$$

Ce résultat prouve le lemme, car la promotion en chaîne est une bijection (cf. définition 10). ■

5.2.4 Génération

Nous souhaitons générer des chaînes selon notre distribution uniforme.

Le cas $n = 0$ est évident, car l'unique chaîne concernée est la chaîne vide. Sinon ($n > 0$), d'après une utilisation récursive de la propriété exposée par le corollaire 1, il est suffisant de concaténer n lettres uniformes selon a à la chaîne vide pour obtenir une chaîne uniforme selon s^n.

Une implémentation simple d'une telle procédure est donnée par l'algorithme 1, de complexité $O(n \times \alpha)$, avec $O(\alpha)$ la complexité d'un choix aléatoire selon A.

Données : $n \in \mathbb{N}$.
Résultat : Une chaîne uniforme selon s^n.
début
  A $\longleftarrow$ distribution uniforme selon a : $\forall l \in A \cup \{\lambda\}$:

$$\mathrm{A}(l) = \mathrm{a}(l) \times \mathrm{a}(A \cup \{\lambda\})^{-1};$$

  $X \longleftarrow \{\}$;
  pour $i \longleftarrow 1$ **à** n **faire**
    $l \longleftarrow$ choix aléatoire selon A;
    $X \longleftarrow X.l$;
  fin
  retourner X;
fin

Algorithme 1 : Génération d'une chaîne uniforme.

Pour information, il existe un grand nombre de techniques permettant de faire le choix aléatoire d'un élément d'un ensemble fini E selon une probabilité définie sur 2^E. Par exemple, le lecteur peut s'informer sur la méthode de la *roulette* (*roulette wheel selection* [Whi94]).

5.2.5 Conclusion

Nous avons vu dans cette section comment définir une mesure pour ensembles de chaînes de taille limitée, et sa normalisation en distribution uniforme.

Notre distribution uniforme possède la propriété intéressante de préservation par concaténation, ce qui nous a permis de mettre au point un algorithme simple de génération de chaîne uniforme,

de complexité linéaire en la taille maximale des chaînes considérées, et qui peut se résumer de la manière suivante : il est possible de générer une chaîne uniforme en concaténant des lettres uniformes. Cette simplicité est bien entendu due à la mesure particulière de chaînes que nous avons définie, mais comme nous l'avons vu, cette mesure ne satisfait pas seulement notre souhait de simplicité mais est en plus pertinente par rapport au type de structures considéré, notamment à la combinatoire qui lui est inhérente.

Nous allons voir dans la section suivante comment étendre ce formalisme aux multiensembles, simplement en introduisant, dans la définition de la mesure, le facteur combinatoire supplémentaire inhérent à la *non* prise en compte de l'ordre dans la composition de telles structures. Ensuite, la section 5.4 est dédiée à l'extension naturelle de ces mesures pour les cas d'arbres ordonnés et non ordonnés, simplement en introduisant la combinatoire supplémentaire inhérente cette fois-ci à l'aspect *hiérarchique* de tels types structurels.

5.3 Multiensembles

Soient A un alphabet, auquel nous associons λ en qualité de lettre vide, et $n \in \mathbb{N}$. Dans cette section, nous supposons la donnée d'une mesure a sur l'ensemble puissance de $A \cup \{\lambda\}$, et proposons la définition de la mesure m^n sur l'ensemble puissance de $\mathbb{M}(A)^{\leqslant n}$, et sa normalisation résultant en la distribution uniforme selon m^n, notée M^n. Nous allons définir m^n en fonction de a et de la combinatoire inhérente à la nature du type multiensemble.

5.3.1 Mesure

Un multiensemble M sur A peut être décrit par une chaîne sur A de longueur $|M|$, simplement en composant cette chaîne d'autant de copies de chaque lettre l de A que la multiplicité de l dans M, et en tenant compte d'un ordre sur cette composition. Nous considérons donc M comme étant la représentation canonique de l'ensemble des chaînes par lequel il peut être décrit grâce à une telle procédure. Ce raisonnement, ajouté à la propriété d'additivité des mesures, nous permet de définir m^n de la manière suivante :

$$\mathrm{m}^n(M) = \sum_{X \in \text{chaînes}(M)} \mathrm{s}^n(X),$$

où chaînes(M) désigne l'ensemble des chaînes que représente M, et s^n la mesure définie sur l'ensemble puissance de $\mathbb{S}(A)^{\leqslant n}$ (cf. définition 60, section précédente).

Le cardinal de chaînes(M) est égal au nombre de r-permutations (cf. section B.4) de A avec $M(l)$ répétitions de chaque lettre l de A. De plus, nous avons vu dans la section précédente que toutes les chaînes composées des mêmes lettres sont d'égale mesure. Ce raisonnement nous permet de définir m^n de la manière suivante :

Définition 62 (Mesure de multiensembles)
$\forall M \in \mathbb{M}(A)^{\leqslant n}$:

$$\begin{aligned}
\mathrm{m}^n(M) &= \text{r-P}(M) \times \mathrm{C}_n^{|M|} \times \prod_{l \in A} \mathrm{a}(l)^{M(l)} \times \mathrm{a}(\lambda)^{n-|M|} \\
&= \frac{|M|!}{\prod_{l \in A} M(l)!} \times \frac{n!}{|M|! \times (n-|M|)!} \times \prod_{l \in A} \mathrm{a}(l)^{M(l)} \times \mathrm{a}(\lambda)^{n-|M|} \\
&= \frac{n!}{\prod_{l \in A} M(l)! \times (n-|M|)!} \times \prod_{l \in A} \mathrm{a}(l)^{M(l)} \times \mathrm{a}(\lambda)^{n-|M|} \\
&= \text{r-P}(M, n-|M|) \times \prod_{l \in A} \mathrm{a}(l)^{M(l)} \times \mathrm{a}(\lambda)^{n-|M|}.
\end{aligned}$$

La mesure d'un ensemble E qui n'est pas un singleton est déduite de la propriété d'additivité des mesures : somme des mesures de tous les singletons qui sont sous-ensembles de E (et donc 0 pour l'ensemble vide). Il est donc inutile de prouver que m^n *est* une mesure sur l'ensemble puissance de $\mathbb{M}(A)^{\leqslant n}$.

5.3.2 Probabilité

D'après le critère général d'uniformité (cf. définition 59) selon la mesure m^n, la probabilité d'un multiensemble est donnée par la formule suivante : $\forall M \in \mathbb{M}(A)^{\leqslant n}$:

$$\mathrm{M}^n(M) = \mathrm{m}^n(M) \times \mathrm{m}^n\left(\mathbb{M}(A)^{\leqslant n}\right)^{-1}.$$

Il serait bien trop complexe de calculer la mesure totale de $\mathbb{M}(A)^{\leqslant n}$ par une naïve énumération récursive de tous les multiensembles de cet ensemble, procédure exigeant un temps de calcul exponentiel en n. Bien heureusement, nous pouvons obtenir cette valeur plus simplement, grâce à l'égalité suivante :

Proposition 4 (Mesure totale de multiensembles)

$$\mathrm{m}^n\left(\mathbb{M}(A)^{\leqslant n}\right) = \mathrm{a}(A \cup \{\lambda\})^n.$$

Preuve (Proposition 4)

$$\begin{aligned}\mathrm{m}^n\left(\mathbb{M}(A)^{\leqslant n}\right) &= \sum_{N \in \mathbb{M}(A)^{\leqslant n}} \mathrm{m}^n(N) \\ &= \sum_{N \in \mathbb{M}(A)^{\leqslant n}} \sum_{X \in \text{chaînes}(N)} \mathrm{s}^n(X),\end{aligned}$$

où chaînes(N) désigne l'ensemble des chaînes que représente le multiensemble N, comme nous l'avons vu dans la section précédente.

Nous pouvons aisément remarquer que la fonction définie par chaînes possède ces deux caractéristiques :

- *toute* chaîne est associée à un *unique* multiensemble ;
- *aucun* multiensemble n'est associé à l'ensemble vide.

En d'autres termes, chaînes définit une bijection de départ $\mathbb{M}(A)^{\leqslant n}$ et d'arrivée une *partition* de $\mathbb{S}(A)^{\leqslant n}$. Nous en concluons donc que la mesure totale de $\mathbb{M}(A)^{\leqslant n}$ est égale à la mesure totale de $\mathbb{S}(A)^{\leqslant n}$, ce qui nous donne l'égalité suivante, d'après la proposition 2 :

$$\mathrm{m}^n\left(\mathbb{M}(A)^{\leqslant n}\right) = \mathrm{s}^n\left(\mathbb{S}(A)^{\leqslant n}\right) = \mathrm{a}(A \cup \{\lambda\})^n. \qquad \blacksquare$$

D'après la définition 62 et la proposition 4, nous pouvons calculer la probabilité d'un multiensemble en $O(n \times \alpha)$, avec $O(\alpha)$ la complexité d'une mesure selon a :

Définition 63 (Distribution uniforme de multiensembles)
$\forall M \in \mathbb{M}(A)^{\leqslant n}$:

$$\mathrm{M}^n(M) = \text{r-P}(M, n - |M|) \times \prod_{l \in A} \mathrm{a}(l)^{M(l)} \times \mathrm{a}(\lambda)^{n-|M|} \times \mathrm{a}(A \cup \{\lambda\})^{-n}.$$

5.3.3 Préservation

Une propriété intéressante de notre distribution uniforme de multiensembles est la préservation du critère d'uniformité par union-addition : l'union-addition de deux multiensembles uniformes résulte en un multiensemble uniforme. Nous utilisons les termes « multiensemble uniforme » ou « multiensemble uniformément distribué » pour désigner un multiensemble dont la probabilité est assignée par une distribution uniforme.

Proposition 5 (Préservation d'uniformité de multiensembles)
$\forall i \in \{0, \ldots, n\}, \forall N \in \mathbb{M}(A)^{\leqslant i}, \forall P \in \mathbb{M}(A)^{\leqslant n-i}$, si N est uniformément distribué selon m^i, et P uniformément distribué selon m^{n-i}, alors $M = N \uplus P$ est uniformément distribué selon m^n.

Preuve (Proposition 5)
Tout d'abord, notons l'égalité suivante : $\forall n_1, \ldots, n_k \in \mathbb{N}, \forall i \in \left\{0, \ldots, \sum_{j=1}^{k} n_j\right\}$:

$$\text{r-P}(n_1, \ldots, n_k) = \sum_{0 \leqslant i_j \leqslant n_j, \forall j \in \{1,\ldots,k\} | \sum_{j=1}^{k} i_j = i} \left[\text{r-P}(i_1, \ldots, i_k) \times \text{r-P}(n_1 - i_1, \ldots, n_k - i_k)\right].$$

Dès lors, nous avons : $\forall M \in \mathbb{M}(A)^{\leqslant n}$:

$$\begin{aligned}\text{r-P}(M, n - |M|) = &\sum_{0 \leqslant i(l) \leqslant M(l), \forall l \in A | \sum_{l \in A} i(l) \leqslant i} \left[\text{r-P}\left(i(l), \forall l \in A; i - \sum_{l \in A} i(l)\right)\right. \\ &\left. \times \text{r-P}\left(M(l) - i(l), \forall l \in A; (n-i) - \sum_{l \in A}(M(l) - i(l))\right)\right] \\ = &\sum_{N \in \mathbb{M}(A)^{\leqslant i}, P \in \mathbb{M}(A)^{\leqslant n-i} | N \uplus P = M} \left[\text{r-P}(N, i - |N|) \times \text{r-P}(P, (n-i) - |P|)\right].\end{aligned}$$

De plus, nous avons : $\forall N \in \mathbb{M}(A)^{\leqslant i}, \forall P \in \mathbb{M}(A)^{\leqslant n-i} | N \uplus P = M$:

$$\begin{aligned}&\text{M}^i(N) \times \text{M}^{n-i}(P) \\ &= \left(\text{r-P}(N, i - |N|) \times \prod_{l \in A} \text{a}(l)^{N(l)} \times \text{a}(\lambda)^{i-|N|} \times \text{a}(A \cup \{\lambda\})^{-i}\right) \\ &\times \left(\text{r-P}(P, (n-i) - |P|) \times \prod_{l \in A} \text{a}(l)^{P(l)} \times \text{a}(\lambda)^{(n-i)-|P|} \times \text{a}(A \cup \{\lambda\})^{-(n-i)}\right) \\ &= \text{r-P}(N, i - |N|) \times \text{r-P}(P, (n-i) - |P|) \times \prod_{l \in A} \text{a}(l)^{(N \uplus P)(l)} \times \text{a}(\lambda)^{n-|N \uplus P|} \times \text{a}(A \cup \{\lambda\})^{-n} \\ &= \text{r-P}(N, i - |N|) \times \text{r-P}(P, (n-i) - |P|) \times \prod_{l \in A} \text{a}(l)^{M(l)} \times \text{a}(\lambda)^{n-|M|} \times \text{a}(A \cup \{\lambda\})^{-n}.\end{aligned}$$

Nous obtenons finalement l'égalité suivante :

$$\begin{aligned}\text{M}^n(M) &= \text{r-P}(M, n - |M|) \times \prod_{l \in A} \text{a}(l)^{M(l)} \times \text{a}(\lambda)^{n-|M|} \times \text{a}(A \cup \{\lambda\})^{-n} \\ &= \sum_{N \in \mathbb{M}(A)^{\leqslant i}, P \in \mathbb{M}(A)^{\leqslant n-i} | N \uplus P = M} \left[\text{r-P}(N, i - |N|) \times \text{r-P}(P, (n-i) - |P|)\right] \times \prod_{l \in A} \text{a}(l)^{M(l)} \\ &\times \text{a}(\lambda)^{n-|M|} \times \text{a}(A \cup \{\lambda\})^{-n} \\ &= \sum_{N \in \mathbb{M}(A)^{\leqslant i}, P \in \mathbb{M}(A)^{\leqslant n-i} | N \uplus P = M} \left[\text{r-P}(N, i - |N|) \times \text{r-P}(P, (n-i) - |P|) \times \prod_{l \in A} \text{a}(l)^{M(l)}\right. \\ &\left. \times \text{a}(\lambda)^{n-|M|} \times \text{a}(A \cup \{\lambda\})^{-n}\right] \\ &= \sum_{N \in \mathbb{M}(A)^{\leqslant i}, P \in \mathbb{M}(A)^{\leqslant n-i} | N \uplus P = M} \left[\text{M}^i(N) \times \text{M}^{n-i}(P)\right].\end{aligned}$$

Nous en déduisons que quel que soit $i \in \{0, \ldots, n\}$, la probabilité d'un multiensemble uniforme M selon m^n est égale à la somme des probabilités de toutes les unions-additions possibles d'un multiensemble uniforme N selon m^i avec un multiensemble uniforme P selon m^{n-i}, tels que $N \uplus P = M$. Il s'agit exactement de la signification de la dernière proposition. ■

Le raisonnement général ci-dessus nous conduit au résultat spécifique suivant, utile en 5.3.4 :

Corollaire 2 (Préservation d'uniformité de multiensembles)
$\forall M \in \mathbb{M}(A)^{\leqslant n}, \forall l \in A \cup \{\lambda\}$, si M est uniformément distribué selon m^n, et l uniformément distribuée selon a, alors $M \uplus l$ est uniformément distribué selon m^{n+1}.

La preuve du corollaire ci-dessus découle du lemme suivant :

Lemme 3
$\forall l \in A \cup \{\lambda\}$, l est uniformément distribuée selon a ssi $[] \uplus l$ est uniformément distribué selon m^1.

Dès lors, nous avons :

Preuve (Corollaire 2)
D'après le lemme 3, $[] \uplus l$ est uniformément distribué selon m^1. Alors, d'après la proposition 5, $M \uplus ([] \uplus l) = M \uplus l$ est uniformément distribuée selon m^{n+1}. ■

Il nous reste à prouver le lemme précédent :

Preuve (Lemme 3)
Notons $\mathrm{A}(l)$ la probabilité de l selon une distribution uniforme selon a :

$$\mathrm{A}(l) = \mathrm{a}(l) \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Si $l = \lambda$, nous avons :

$$\mathrm{A}(l) = \text{r-P}([] \uplus l, 1 - 0) \times 1 \times \mathrm{a}(\lambda)^{1-0} \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Sinon ($l \in A$), nous avons :

$$\mathrm{A}(l) = \text{r-P}([] \uplus l, 1 - 1) \times \mathrm{a}(l) \times \mathrm{a}(\lambda)^{1-1} \times \mathrm{a}(A \cup \{\lambda\})^{-1}.$$

Donc dans tous les cas, nous avons :

$$\begin{aligned}\mathrm{A}(l) &= \text{r-P}([] \uplus l, 1 - |[] \uplus l|) \times \prod_{l \in A} \mathrm{a}(l)^{([] \uplus l)(l)} \times \mathrm{a}(\lambda)^{1-|[] \uplus l|} \times \mathrm{a}(A \cup \{\lambda\})^{-1} \\ &= \mathrm{M}^1([] \uplus l).\end{aligned}$$

Ce résultat prouve le lemme, car la promotion en multiensemble est une bijection (cf. définition 6). ■

5.3.4 Génération

Nous souhaitons générer des multiensembles selon notre distribution uniforme.

Le cas $n = 0$ est évident, car l'unique multiensemble concerné est le multiensemble vide. Sinon ($n > 0$), d'après une utilisation récursive de la propriété exposée par le corollaire 2, il est suffisant d'union-additionner n lettres uniformes selon a au multiensemble vide pour obtenir un multiensemble uniforme selon m^n.

Une implémentation simple d'une telle procédure est donnée par l'algorithme 2, de complexité $O(n \times \alpha)$, avec $O(\alpha)$ la complexité d'un choix aléatoire selon A.

5.3.5 Conclusion

Nous avons vu dans cette section comment définir une mesure pour ensembles de multiensembles de taille limitée, et sa normalisation en distribution uniforme.

Notre distribution uniforme possède la propriété intéressante de préservation par union-addition, ce qui nous a permis de mettre au point un algorithme simple de génération de multiensemble uniforme, de complexité linéaire en la taille maximale des multiensembles considérés, et qui peut se résumer de la manière suivante : il est possible de générer un multiensemble uniforme en union-additionnant des lettres uniformes. Cette simplicité est bien entendu due à la mesure particulière de multiensembles que nous avons définie, mais comme nous l'avons vu, cette mesure ne satisfait pas seulement notre souhait de simplicité mais est en plus pertinente par rapport au type de structures considéré, notamment à la combinatoire qui lui est inhérente.

Nous avons obtenu lesdits résultats en étendant simplement aux multiensembles la mesure de chaînes définie dans la section précédente, par prise en compte du facteur combinatoire supplémentaire inhérent à la *non* considération de l'ordre dans la composition de telles structures.

Données : $n \in \mathbb{N}$.
Résultat : Un multiensemble uniforme selon m^n.
début
  A $\longleftarrow$ distribution uniforme selon a : $\forall l \in A \cup \{\lambda\}$:

$$\mathrm{A}(l) = \mathrm{a}(l) \times \mathrm{a}(A \cup \{\lambda\})^{-1};$$

  $M \longleftarrow []$;
  pour $i \longleftarrow 1$ **à** n **faire**
    $l \longleftarrow$ choix aléatoire selon A;
    $M \longleftarrow M \uplus l$;
  fin
  retourner M;
fin

Algorithme 2 : Génération d'un multiensemble uniforme.

Nous allons voir dans la section suivante comment, en restant dans cette philosophie de simplicité, étendre aux arbres ces deux dernières définitions de mesures, par introduction de la combinatoire supplémentaire inhérente cette fois-ci à l'aspect *hiérarchique* de ce type structurel.

5.4 Arbres

Soient A un alphabet, auquel nous associons λ en qualité de lettre vide, et $a, d \in \mathbb{N}$. Dans cette section, nous supposons la donnée d'une mesure a sur l'ensemble puissance de $A \cup \{\lambda\}$, et proposons les définitions des mesures u_d^a et o_d^a sur les ensembles puissances de $\mathbb{U}(A)_{\leqslant d}^{\leqslant a}$ et $\mathbb{O}(A)_{\leqslant d}^{\leqslant a}$, respectivement, et leur normalisation résultant en les distributions uniformes selon u_d^a et o_d^a, notées U_d^a et O_d^a, respectivement. Nous allons définir lesdites mesures en fonction de a et de la combinatoire inhérente à la nature du type arbre.

5.4.1 Mesures

Si $d > 0$, alors nous pouvons en déduire, d'après les définitions exposées en 2.4, que la liste des enfants d'un u-arbre U de $\mathbb{U}(A)_{\leqslant d}^{\leqslant a}$ est un multiensemble sur l'alphabet $\mathbb{U}(A)_{\leqslant d-1}^{\leqslant a} \smallsetminus \{(\lambda, [])\}$. Ainsi, en fixant à l'u-arbre vide la lettre vide associée à cet alphabet particulier, nous pouvons calculer la contribution de $c(U)$ dans la mesure de U grâce à la mesure de multiensembles définie dans la section 5.3, et à une utilisation récursive de la mesure d'u-arbres pour chacun desdits enfants. Finalement, la contribution de la racine de U est donnée par a, et la mesure de U est simplement obtenue par multiplication des contributions de sa racine et de sa liste d'enfants.

Il y a un cas spécial, celui de l'u-arbre vide. Souvenons nous, grâce à la définition 11, que nous avons imposé, par souci de consistance, qu'un u-arbre ayant une racine vide doit avoir une liste d'enfants vide, et donc *être* l'u-arbre vide. En d'autres termes, l'u-arbre vide est la représentation canonique de l'ensemble des telles « structures hiérarchiques » non ordonnées ayant une racine vide, cela sans aucune considération des enfants. Cette caractéristique du type u-arbre, bien illustrée d'ailleurs par l'opération d'u-enracinement (définition 14), doit se retrouver dans la définition de notre mesure u_d^a : la contribution de la liste des enfants dans la mesure de l'u-arbre vide doit être égale à la mesure (de multiensembles) totale de $\mathbb{M}\left(\mathbb{U}(A)_{\leqslant d-1}^{\leqslant a} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}$ (l'ensemble des multiensembles de taille au plus a sur l'alphabet des u-arbres non vides sur A d'arité a et profondeur au plus $d-1$).

Le cas $d = 0$ est la base de ce schéma récursif, car il n'y a aucun arbre de profondeur négative. Notons que le seul u-arbre concerné est l'u-arbre vide.

Si $d > 0$, notons m_{d-1}^a la mesure de multiensembles de taille au plus a telle que définie dans la section 5.3, mais en considérant l'alphabet $\mathbb{U}(A)_{\leqslant d-1}^{\leqslant a} \smallsetminus \{(\lambda, [])\}$, avec l'u-arbre vide associé en qualité de lettre vide, et u_{d-1}^a la mesure de lettres correspondante. Le raisonnement ci-dessus nous

conduit à la définition suivante :

Définition 64 (Mesure d'u-arbres)
$\forall U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant d}$:
- si $d = 0$ (alors U est vide) :
$$\mathrm{u}^a_0(U = (\lambda, [])) = \mathrm{a}(\lambda);$$
- sinon ($d > 0$) :
  - si U est vide :
$$\mathrm{u}^a_d(U = (\lambda, [])) = \mathrm{a}(\lambda) \times {}^{\text{ii}}\,\mathrm{m}^a_{d-1}\left(\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d-1} \setminus \{(\lambda, [])\}\right)^{\leqslant a}\right),$$
  - sinon (U n'est pas vide) :
$$\mathrm{u}^a_d(U) = \mathrm{a}(r(U)) \times {}^{\text{iii}}\,\mathrm{m}^a_{d-1}(c(U)).$$

La mesure d'un ensemble E qui n'est pas un singleton est déduite de la propriété d'additivité des mesures : somme des mesures de tous les singletons qui sont sous-ensembles de E (et donc 0 pour l'ensemble vide). Il est donc inutile de prouver que u^a_d *est* une mesure sur l'ensemble puissance de $\mathbb{U}(A)^{\leqslant a}_{\leqslant d}$.

La définition de o^a_d est déduite du même type de raisonnement, simplement en remplaçant le terme « u-arbre » (resp. « non ordonné » ; « multiensemble » ...) par le terme « o-arbre » (resp. « ordonné » ; « chaîne » ...), et en corrigeant bien entendu les notations et références correspondantes.

Définition 65 (Mesure d'o-arbres)
$\forall O \in \mathbb{O}(A)^{\leqslant a}_{\leqslant d}$:
- si $d = 0$ (alors O est vide) :
$$\mathrm{o}^a_0(O = (\lambda, \{\})) = \mathrm{a}(\lambda);$$
- sinon ($d > 0$) :
  - si O est vide :
$$\mathrm{o}^a_d(O = (\lambda, \{\})) = \mathrm{a}(\lambda) \times {}^{\text{iv}}\,\mathrm{s}^a_{d-1}\left(\mathbb{S}\left(\mathbb{O}(A)^{\leqslant a}_{\leqslant d-1} \setminus \{(\lambda, \{\})\}\right)^{\leqslant a}\right),$$
  - sinon (O n'est pas vide) :
$$\mathrm{o}^a_d(O) = \mathrm{a}(r(O)) \times {}^{\text{v}}\,\mathrm{s}^a_{d-1}(c(O)).$$

ii. D'après la proposition 4, nous avons :
$$\mathrm{m}^a_{d-1}\left(\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d-1} \setminus \{(\lambda, [])\}\right)^{\leqslant a}\right) = \mathrm{u}^a_{d-1}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d-1}\right)^a.$$

iii. D'après la définition 62, nous avons :
$$\mathrm{m}^a_{d-1}(c(U)) = \text{r-P}(c(U), a - |c(U)|) \times \prod_{C \in c(U)} \mathrm{u}^a_{d-1}(C)^{c(U)(C)} \times \mathrm{u}^a_{d-1}(\lambda, [])^{a-|c(U)|}.$$

iv. D'après la proposition 2, nous avons :
$$\mathrm{s}^a_{d-1}\left(\mathbb{S}\left(\mathbb{O}(A)^{\leqslant a}_{\leqslant d-1} \setminus \{(\lambda, \{\})\}\right)^{\leqslant a}\right) = \mathrm{o}^a_{d-1}\left(\mathbb{O}(A)^{\leqslant a}_{\leqslant d-1}\right)^a.$$

5.4.2 Probabilités

D'après le critère général d'uniformité (cf. définition 59) selon les mesures u_d^a et o_d^a, la probabilité d'un arbre est donnée par l'une des formules suivantes :

Définition 66 (Distribution uniforme d'arbres)
$\forall U \in \mathbb{U}(A)_{\leqslant d}^{\leqslant a}, \forall O \in \mathbb{O}(A)_{\leqslant d}^{\leqslant a}$:

$$\mathrm{U}^n(U) = \mathrm{u}_d^a(U) \times \mathrm{u}_d^a\left(\mathbb{U}(A)_{\leqslant d}^{\leqslant a}\right)^{-1},$$
$$\mathrm{O}^n(O) = \mathrm{o}_d^a(O) \times \mathrm{o}_d^a\left(\mathbb{O}(A)_{\leqslant d}^{\leqslant a}\right)^{-1}.$$

Il serait bien trop complexe de calculer les mesures totales de $\mathbb{U}(A)_{\leqslant d}^{\leqslant a}$ et $\mathbb{O}(A)_{\leqslant d}^{\leqslant a}$ par de naïves énumérations récursives de tous les arbres de ces ensembles, procédures exigeant des temps de calcul exponentiels en a^d. Bien heureusement, nous pouvons obtenir ces valeurs plus simplement, grâce à l'égalité suivante :

Proposition 6 (Mesure totale d'arbres)

$$\mathrm{u}_d^a\left(\mathbb{U}(A)_{\leqslant d}^{\leqslant a}\right) = \mathrm{o}_d^a\left(\mathbb{O}(A)_{\leqslant d}^{\leqslant a}\right) = \mathrm{a}(A \cup \{\lambda\})^{\text{taille\_max\_arbre}(a,d)} \times \mathrm{a}(\lambda)^{a^d},$$

où taille_max_arbre(a, d) = taille_max_u-arbre(a, d) = taille_max_o-arbre(a, d) (cf. définitions 13 et 17).

Nous allons prouver la proposition précédente seulement pour le cas non ordonné, car le raisonnement est le même pour le cas ordonné, et il suffit de corriger les notations et références correspondantes pour obtenir ladite preuve. La formule suivante va nous être utile :

Lemme 4
$\forall i \in \mathbb{N}$:

$$\mathrm{u}_{i+1}^a\left(\mathbb{U}(A)_{\leqslant i+1}^{\leqslant a}\right) = \mathrm{a}(A \cup \{\lambda\}) \times \mathrm{u}_i^a\left(\mathbb{U}(A)_{\leqslant i}^{\leqslant a}\right)^a.$$

Dès lors, nous avons :

Preuve (Proposition 6)

$$\text{Lemme 4} \wedge \left[\mathrm{u}_0^a\left(\mathbb{U}(A)_{\leqslant 0}^{\leqslant a}\right) = \mathrm{u}_0^a(\lambda, []) = \mathrm{a}(\lambda)\right] \Longrightarrow \forall i \in \mathbb{N} :$$

$$\begin{aligned}\mathrm{u}_i^a\left(\mathbb{U}(A)_{\leqslant i}^{\leqslant a}\right) &= \mathrm{a}(A \cup \{\lambda\})^{\sum_{j=1}^{i} a^{j-1}} \times \mathrm{a}(\lambda)^{\prod_{j=1}^{i} a} \\ &= \mathrm{a}(A \cup \{\lambda\})^{\text{taille\_max\_arbre}(a,i)} \times \mathrm{a}(\lambda)^{a^i},\end{aligned}$$

Ce résultat prouve la proposition 6, avec $i = d$. ■

Nous utilisons le deuxième lemme suivant afin de prouver aisément le lemme précédent :

v. D'après la définition 60, nous avons :

$$\mathrm{s}_{d-1}^a(c(O)) = \mathrm{C}_a^{|c(O)|} \times \prod_{i=1}^{|c(O)|} \mathrm{o}_{d-1}^a(c(O)(i)) \times \mathrm{o}_{d-1}^a(\lambda, \{\})^{a-|c(O)|}.$$

Lemme 5
$\forall i \in \mathbb{N}, \forall l \in A \cup \{\lambda\}$:

$$\sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=l} \mathrm{u}^a_{i+1}(U) = \mathrm{a}(l) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a .$$

Nous avons alors :

Preuve (Lemme 4)

$$\begin{aligned}
\mathrm{u}^a_{i+1}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i+1}\right) &= \sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1}} \mathrm{u}^a_{i+1}(U) \\
&= \sum_{l \in A \cup \{\lambda\}} \sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=l} \mathrm{u}^a_{i+1}(U) \\
&= \sum_{l \in A \cup \{\lambda\}} \left[\mathrm{a}(l) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a\right] \\
&= \sum_{l \in A \cup \{\lambda\}} \mathrm{a}(l) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a \\
&= \mathrm{a}(A \cup \{\lambda\}) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a .
\end{aligned}$$

■

Il nous reste à prouver le dernier lemme :

Preuve (Lemme 5)
Si $l = \lambda$, nous avons :

$$\sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=\lambda} \mathrm{u}^a_{i+1}(U) = \mathrm{u}^a_{i+1}(\lambda, []) = \mathrm{a}(\lambda) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a .$$

Sinon ($l \in A$), nous avons :

$$\begin{aligned}
\sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=l} \mathrm{u}^a_{i+1}(U) &= \sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=l} [\mathrm{a}(l) \times \mathrm{m}^a_i(c(U))] \\
&= \mathrm{a}(l) \times \sum_{U \in \mathbb{U}(A)^{\leqslant a}_{\leqslant i+1} \mid r(U)=l} \mathrm{m}^a_i(c(U)) \\
&= \mathrm{a}(l) \times \mathrm{m}^a_i \left(\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i} \setminus \{(\lambda, [])\}\right)^{\leqslant a}\right) \\
&= {}^{\text{vi}}\, \mathrm{a}(l) \times \mathrm{u}^a_i \left(\mathbb{U}(A)^{\leqslant a}_{\leqslant i}\right)^a .
\end{aligned}$$

■

D'après la proposition 6 et les définitions 64 et 13, nous pouvons calculer la probabilité d'un u-arbre U en $O(\text{taille\_max\_u-arbre}(a, d) \times \alpha)$, avec $O(\alpha)$ la complexité d'une mesure selon a. Encore une fois, la conclusion est similaire pour le cas ordonné, grâce à une adaptation du même raisonnement.

5.4.3 Préservation

Une propriété intéressante de notre distribution uniforme d'arbres est la préservation du critère d'uniformité par enracinement : l'u-enracinement (resp. o-enracinement) d'une lettre uniforme et

vi. Proposition 4.

d'un multiensemble uniforme d'u-arbres (resp. une chaîne uniforme d'o-arbres) non vides résulte en un u-arbre (resp. o-arbre) uniforme. Nous utilisons les termes « arbre uniforme » ou « arbre uniformément distribué » pour désigner un arbre dont la probabilité est assignée par une distribution uniforme.

Proposition 7 (Préservation d'uniformité d'arbres)
$\forall l \in A \cup \{\lambda\}$, si l est uniformément distribuée selon a, alors :

- $\forall M \in \mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}$, si M est uniformément distribué selon m^a_d, alors $U = (l \downarrow^a_d M)$ est uniformément distribué selon u^a_{d+1} ;
- $\forall S \in \mathbb{S}\left(\mathbb{O}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, \{\})\}\right)^{\leqslant a}$, si S est uniformément distribuée selon s^a_d, alors $O = (l \downarrow^a_d S)$ est uniformément distribué selon o^a_{d+1}.

Comme précédemment et pour les mêmes raisons, nous prouvons la proposition précédente seulement pour le cas non ordonné.

Preuve (Proposition 7)
$\forall m \in A \cup \{\lambda\}, \forall N \in \mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}$, notons $\mathrm{A}(m)$ la probabilité de m selon une distribution uniforme selon a, et $\mathrm{M}^a_d(N)$ la probabilité de N selon une distribution uniforme selon m^a_d :

$$\begin{aligned}
\mathrm{A}(m) &= \mathrm{a}(m) \times \mathrm{a}(A \cup \{\lambda\})^{-1}, \\
\mathrm{M}^a_d(N) &= \mathrm{m}^a_d(N) \times \mathrm{m}^a_d\left(\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}\right)^{-1} \\
&= {}^{\text{vii}}\, \mathrm{m}^a_d(N) \times \mathrm{u}^a_d\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d}\right)^{-a}.
\end{aligned}$$

Tout d'abord, nous avons :

$$\mathrm{U}^a_{d+1}(U) = \mathrm{u}^a_{d+1}(U) \times \mathrm{u}^a_{d+1}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d+1}\right)^{-1}.$$

Si $l = \lambda$, nous avons :

$$\begin{aligned}
\mathrm{U}^a_{d+1}(U) &= \mathrm{U}^a_{d+1}(\lambda, []) \\
&= \mathrm{a}(\lambda) \times \mathrm{m}^a_d\left(\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}\right) \times \mathrm{u}^a_{d+1}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d+1}\right)^{-1} \\
&= \mathrm{a}(\lambda) \times \sum_{N \in \mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}} \mathrm{m}^a_d(N) \times \mathrm{a}(A \cup \{\lambda\})^{-1} \times \mathrm{u}^a_d\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d}\right)^{-a} \\
&= \sum_{N \in \mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}} \left[\mathrm{a}(\lambda) \times \mathrm{a}(A \cup \{\lambda\})^{-1} \times \mathrm{m}^a_d(N) \times \mathrm{u}^a_d\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d}\right)^{-a}\right] \\
&= \sum_{N \in \mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d} \smallsetminus \{(\lambda, [])\}\right)^{\leqslant a}} \left[\mathrm{A}(\lambda) \times \mathrm{M}^a_d(N)\right].
\end{aligned}$$

Sinon ($l \in A$), nous avons :

$$\begin{aligned}
\mathrm{U}^a_{d+1}(U) &= \mathrm{U}^a_{d+1}(l, M) \\
&= \mathrm{a}(l) \times \mathrm{m}^a_d(M) \times \mathrm{u}^a_{d+1}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d+1}\right)^{-1} \\
&= \mathrm{a}(l) \times \mathrm{m}^a_d(M) \times \mathrm{a}(A \cup \{\lambda\})^{-1} \times \mathrm{u}^a_d\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d}\right)^{-a} \\
&= \mathrm{A}(l) \times \mathrm{M}^a_d(M).
\end{aligned}$$

vii. Proposition 4.

Finalement dans tous les cas, d'après la définition 14, nous avons :

$$\mathrm{U}^a_{d+1}(U) = \sum_{m\in A\cup\{\lambda\}, N\in\mathbb{M}\left(\mathbb{U}(A)^{\leqslant a}_{\leqslant d}\smallsetminus\{(\lambda,[])\}\right)^{\leqslant a}\,\middle|\,(m\downarrow^a_d N)=U} [\mathrm{A}(m)\times\mathrm{M}^a_d(N)]\,.$$

Nous en déduisons que la probabilité d'un u-arbre U uniforme selon u^a_{d+1} est égale à la somme des probabilités de tous les u-enracinements possibles d'une lettre m uniforme selon a avec un multiensemble N uniforme selon m^a_d, tels que $(m\downarrow^a_d N) = U$. Il s'agit exactement de la signification de la dernière proposition. ■

5.4.4 Génération

Nous souhaitons générer des arbres selon nos distributions uniformes.

Le cas $d = 0$ est évident, car l'unique u-arbre (resp. o-arbre) concerné est l'u-arbre vide (resp. o-arbre vide). Sinon ($d > 0$), la proposition 7 nous dit qu'il est suffisant d'u-enraciner (resp. o-enraciner) une lettre uniforme selon a avec un multiensemble (resp. une chaîne) uniforme selon m^a_{d-1} (resp. s^a_{d-1}), pour obtenir un u-arbre (resp. o-arbre) uniforme selon u^a_d (resp. o^a_d). Le multiensemble (resp. La chaîne) uniforme peut être généré (resp. générée) grâce à l'algorithme 2 (resp. 1).

D'après le raisonnement ci-dessus et la complexité des algorithmes 2 et 1, deux implémentations simples de telles procédures sont données par les algorithmes 3 et 4, respectivement, tous deux de complexité $O(\text{taille\_max\_arbre}(a,d)\times\alpha)$, avec $O(\alpha)$ la complexité d'un choix aléatoire selon A.

Données : $a, d \in \mathbb{N}$.
Résultat : Un u-arbre uniforme selon u^a_d.
début
  si $d = 0$ **alors**
    retourner $(\lambda, [])$;
  fin
  A ⟵ distribution uniforme selon a : $\forall l \in A\cup\{\lambda\}$:

$$\mathrm{A}(l) = \mathrm{a}(l)\times\mathrm{a}(A\cup\{\lambda\})^{-1};$$

  l ⟵ choix aléatoire selon A;
  M ⟵ multiensemble uniforme selon m^a_{d-1} ; (cf. section 5.3)
  retourner $l\downarrow^a_{d-1} M$;
fin

Algorithme 3 : Génération d'un u-arbre uniforme.

Données : $a, d \in \mathbb{N}$.
Résultat : Un o-arbre uniforme selon o^a_d.
début
  si $d = 0$ **alors**
    retourner $(\lambda, \{\})$;
  fin
  A ⟵ distribution uniforme selon a : $\forall l \in A\cup\{\lambda\}$:

$$\mathrm{A}(l) = \mathrm{a}(l)\times\mathrm{a}(A\cup\{\lambda\})^{-1};$$

  l ⟵ choix aléatoire selon A;
  S ⟵ chaîne uniforme selon s^a_{d-1} ; (cf. section 5.2)
  retourner $l\downarrow^a_{d-1} S$;
fin

Algorithme 4 : Génération d'un o-arbre uniforme.

5.4.5 Conclusion

Nous avons vu dans cette section comment définir des mesures pour ensembles d'arbres de taille limitée, et leur normalisation en distribution uniforme.

Nos distributions uniformes possèdent la propriété intéressante de préservation par enracinement, ce qui nous a permis de mettre au point des algorithmes simples de génération d'arbre uniforme, de complexité linéaire en la taille maximale des arbres considérés, et qui peuvent se résumer de la manière suivante : il est possible de générer un arbre non ordonné (resp. ordonné) uniforme en enracinant une lettre uniforme avec un multiensemble (resp. une chaîne) uniforme d'arbres non ordonnés (resp. ordonnés). Cette simplicité est bien entendu due aux mesures particulières d'arbres que nous avons définies, mais comme nous l'avons vu, ces mesure ne satisfont pas seulement notre souhait de simplicité mais sont en plus pertinentes par rapport au type de structures considéré, notamment à la combinatoire qui lui est inhérente.

Nous avons obtenu lesdits résultats en étendant simplement aux arbres les mesures de chaînes et de multiensembles définies dans les deux sections précédentes, par prise en compte du facteur combinatoire supplémentaire inhérent à l'aspect *hiérarchique* de ce type structurel. Toujours dans cette philosophie de simplicité, nous allons en terminer dans la section suivante avec les mesures et distributions uniformes de structures, en s'attaquant finalement au type structurel le plus « libre » dans sa composition, et donc le plus combinatoire, à savoir le type graphe. Nous conclurons ensuite ces travaux par une discussion sur les perspectives possibles qu'ils soulèvent.

5.5 Graphes

Soient A_v et A_e deux alphabets, auxquels nous associons λ en qualité de lettre vide, et $n \in \mathbb{N}$. Dans cette section, nous supposons les données de deux mesures $\mathrm{a_v}$ et $\mathrm{a_e}$ sur les ensembles puissances de $A_v \cup \{\lambda\}$ et $A_e \cup \{\lambda\}$, respectivement, et proposons la définition de la mesure g^n sur l'ensemble puissance de $\mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$, et sa normalisation résultant en la distribution uniforme selon g^n, notée G^n. Nous allons définir g^n en fonction de $\mathrm{a_e}$ et $\mathrm{a_v}$, et de la combinatoire inhérente à la nature du type graphe.

Avant de rentrer dans les détails, convainquons nous que le choix de l'ensemble sur lequel nous identifions nos graphes n'est pas du tout important. Tout autre ensemble de cardinal au moins n peut nous permettre d'identifier un graphe possédant au plus n sommets, mais le choix de $\{1, \ldots, n\}$ s'est naturellement imposé, ce par volonté de simplicité, à la fois dans les notations et dans la construction itérative d'un graphe uniforme, le tout sans perte de généralité.

5.5.1 Mesure

Soit G un graphe étiqueté sur (A_v, A_e), identifié sur $\{1, \ldots, n\}$, et possédant au plus n sommets. G peut être décrit par un graphe λ-G étiqueté sur $(A_v \cup \{\lambda\}, A_e)$, identifié sur $\{1, \ldots, n\}$, et possédant n sommets, simplement en étiquetant λ les sommets de λ-G qui ne sont pas sommet de G, c'est-à-dire les sommets de λ-G qui sont élément de l'ensemble suivant :

$$v(\lambda\text{-G}) \setminus v(G) = \{1, \ldots, n\} \setminus v(G).$$

En d'autre termes, λ-G modéliserait explicitement l'éventuelle présence implicite de sommets vides dans G, résultant en le fait que G contienne un nombre de sommets strictement inférieur à n. Nous considérons donc G comme étant la représentation canonique de l'ensemble de ces tels graphes λ-G par lequel il peut être décrit grâce à une telle procédure. Il s'agit précisément de l'ensemble des graphes étiquetés sur $(A_v \cup \{\lambda\}, A_e)$, identifiés sur $\{1, \ldots, n\}$, possédant n sommets, et pour lesquels G est le sous-graphe induit par l'ensemble de ses sommets, cela sans considération pour l'ensemble des arêtes ayant pour source ou destination un sommet de λ-G qui n'est pas sommet de G, ni de leur l'étiquetage. Ce raisonnement, ajouté à la propriété d'additivité des mesures, nous permet de définir g^n de la manière suivante :

$$\mathrm{g}^n(G) = \sum_{\lambda\text{-G} \in \lambda\text{-graphes}^n(G)} \lambda\text{-g}^n(\lambda\text{-G}), \tag{5.1}$$

où λ-graphes$^n(G)$ désigne l'ensemble des graphes de $\mathbb{G}(A_v \cup \{\lambda\}, A_e, \{1,\dots,n\})^n$ que représente G, et λ-g^n une mesure définie sur l'ensemble puissance de $\mathbb{G}(A_v \cup \{\lambda\}, A_e, \{1,\dots,n\})^n$.

Soit λ-G $\in \lambda$-graphes$^n(G)$. Nous calculons la mesure de λ-G simplement par multiplication des contributions de ses fonctions $vl(\lambda\text{-G})$ et $el(\lambda\text{-G})$ d'étiquetage des sommets et arêtes, respectivement :

$$\lambda\text{-g}^n(\lambda\text{-G}) = E \times V. \tag{5.2}$$

La contribution de $vl(\lambda$-G$)$ est obtenue par multiplication des mesures par $\mathrm{a_v}$ des étiquettes des sommets de λ-G :

$$V = \prod_{v \in v(\lambda\text{-G})} \mathrm{a_v}(vl(G)(v)) = \prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|}, \tag{5.3}$$

et le calcul est similaire pour la contribution de $el(\lambda$-G$)$, sur l'ensemble des arêtes de λ-G, avec la mesure correspondante, et en considérant l'absence d'une arête comme étant la présence implicite d'une arête vide, c'est-à-dire étiquetée λ :

$$\begin{aligned} E &= \prod_{e \in e(\lambda\text{-G})} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{n^2-|e(\lambda\text{-G})|} \\ &= \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \\ &\times \prod_{e \in e(\lambda\text{-G}) \setminus e(G)} \mathrm{a_e}(el(\lambda\text{-G})(e)) \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|e(\lambda\text{-G}) \setminus e(G)|}. \end{aligned} \tag{5.4}$$

Des équations 5.2, 5.3 et 5.4, nous obtenons la formule suivante :

$$\lambda\text{-g}^n(\lambda\text{-G}) = K \times \prod_{e \in e(\lambda\text{-G}) \setminus e(G)} \mathrm{a_e}(el(\lambda\text{-G})(e)) \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|e(\lambda\text{-G}) \setminus e(G)|}, \tag{5.5}$$

avec :

$$K = \prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|} \times \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|}.$$

La valeur de K ne dépend pas de λ-G. Nous obtenons donc l'égalité suivante, d'après les équations 5.1 et 5.5 :

$$\mathrm{g}^n(G) = K \times \sum_{\lambda\text{-G} \in \lambda\text{-graphes}^n(G)} \left(\prod_{e \in e(\lambda\text{-G}) \setminus e(G)} \mathrm{a_e}(el(\lambda\text{-G})(e)) \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|e(\lambda\text{-G}) \setminus e(G)|} \right). \tag{5.6}$$

La proposition suivante va nous permettre de préciser la définition de g^n :

Proposition 8
$\forall G \in \mathbb{G}(A_v, A_e, \{1,\dots,n\})^{\leqslant n}$:

$$\begin{aligned} &\sum_{\lambda\text{-G} \in \lambda\text{-graphes}^n(G)} \left(\prod_{e \in e(\lambda\text{-G}) \setminus e(G)} \mathrm{a_e}(el(\lambda\text{-G})(e)) \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|e(\lambda\text{-G}) \setminus e(G)|} \right) \\ &= \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|v(G)|^2}. \end{aligned}$$

Preuve (Proposition 8)
Soient f et graphes les fonctions définies comme suit :

– $\forall\, \lambda$-G $\in \lambda$-graphes$^n(G)$:

$$f(\lambda\text{-G}) = \prod_{e \in e(\lambda\text{-G}) \setminus e(G)} \mathrm{a_e}(el(\lambda\text{-G})(e)) \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|e(\lambda\text{-G}) \setminus e(G)|};$$

– $\forall M \in \mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}$:

$$\text{graphes}(M) = \left\{ \lambda\text{-G} \in \lambda\text{-graphes}^n(G) \,\middle|\, M = \biguplus_{e \in e(\lambda\text{-G}) \setminus e(G)} el(\lambda\text{-G})(e) \right\}.$$

Soit $M \in \mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}$. La valeur de f est la même en tout graphe de graphes(M) : $\forall\, \lambda\text{-G} \in \text{graphes}(M)$:

$$f(\lambda\text{-G}) = \prod_{l_e \in A_e} \mathrm{a_e}(l_e)^{M(l_e)} \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|M|}.$$

De plus, le cardinal de graphes(M) est égal au nombre de r-permutations de $A_e \cup \{\lambda\}$ avec $M(l)$ répétitions de chaque lettre l de A, et $n^2 - |v(G)|^2 - |M|$ répétitions de λ. Nous avons donc :

$$\begin{aligned}\sum_{\lambda\text{-G} \in \text{graphes}(M)} f(\lambda\text{-G}) &= \text{r-P}(M, n^2 - |v(G)|^2 - |M|) \times \prod_{l_e \in A_e} \mathrm{a_e}(l_e)^{M(l_e)} \times \mathrm{a_e}(\lambda)^{n^2-|v(G)|^2-|M|} \\ &= \mathrm{m}^{n^2-|v(G)|^2}(M),\end{aligned}$$

où $\mathrm{m}^{n^2-|v(G)|^2}$ désigne la mesure définie sur l'ensemble puissance de $\mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}$ (cf. définition 62, section 5.3), avec $\mathrm{a_e}$ la mesure de lettres associée.

Nous pouvons aisément remarquer que la fonction graphes possède ces deux caractéristiques :
– *tout* graphe est associée à un *unique* multiensemble ;
– *aucun* multiensemble n'est associé à l'ensemble vide.

En d'autres termes, graphes est une bijection de départ $\mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}$ et d'arrivée une *partition* de $\lambda\text{-graphes}^n(G)$. Nous en concluons donc l'égalité suivante, d'après la proposition 4 :

$$\begin{aligned}\sum_{\lambda\text{-G} \in \lambda\text{-graphes}^n(G)} f(\lambda\text{-G}) &= \sum_{M \in \mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}} \sum_{\lambda\text{-G} \in \text{graphes}(M)} f(\lambda\text{-G}) \\ &= \sum_{M \in \mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}} \mathrm{m}^{n^2-|v(G)|^2}(M) \\ &= \mathrm{m}^{n^2-|v(G)|^2}\left(\mathbb{M}(A_e)^{\leqslant n^2-|v(G)|^2}\right) \\ &= \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|v(G)|^2}.\end{aligned}$$

■

L'équation 5.6 et la proposition 8 nous conduisent finalement à la définition suivante :

Définition 67 (Mesure de graphes)
$\forall G \in \mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$:

$$\begin{aligned}\mathrm{g}^n(G) = &\prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|} \\ &\times \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|v(G)|^2}.\end{aligned}$$

La mesure d'un ensemble E qui n'est pas un singleton est déduite de la propriété d'additivité des mesures : somme des mesures de tous les singletons qui sont sous-ensembles de E (et donc 0 pour l'ensemble vide). Il est donc inutile de prouver que g^n *est* une mesure sur l'ensemble puissance de $\mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$.

5.5.2 Probabilité

D'après le critère général d'uniformité (cf. définition 59) selon la mesure g^n, la probabilité d'un graphe est donnée par la formule suivante : $\forall G \in \mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$:

$$\mathrm{G}^n(G) = \mathrm{g}^n(G) \times \mathrm{g}^n\left(\mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}\right)^{-1}.$$

Il serait bien trop complexe de calculer la mesure totale de $\mathbb{G}(A_v,\, A_e,\, \{1,\ldots,n\})^{\leqslant n}$ par une naïve énumération récursive de tous les graphes de cet ensemble, procédure exigeant un temps de calcul exponentiel en n^2. Bien heureusement, nous pouvons obtenir cette valeur plus simplement, grâce à l'égalité suivante :

Proposition 9 (Mesure totale de graphes)

$$\mathrm{g}^n\left(\mathbb{G}(A_v, A_e, \{1,\ldots,n\})^{\leqslant n}\right) = \mathrm{a_v}(A_v \cup \{\lambda\})^n \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2}.$$

Preuve (Proposition 9)
Soient f_v, f_e, $\mathrm{graphes}_v$, et, $\forall M_v \in \mathbb{M}(A_v)^{\leqslant n}, \mathrm{graphes}_e(M_v)$ les fonctions définies comme suit :

– $\forall G \in \mathbb{G}(A_v, A_e, \{1,\ldots,n\})^{\leqslant n}$:

$$f_v(G) = \prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|},$$
$$f_e(G) = \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|v(G)|^2};$$

– $\forall M_v \in \mathbb{M}(A_v)^{\leqslant n}$:

$$\mathrm{graphes}_v(M_v) = \left\{ G \in \mathbb{G}(A_v, A_e, \{1,\ldots,n\})^{\leqslant n} \,\middle|\, M_v = \biguplus_{v \in v(G)} vl(G)(v) \right\};$$

– $\forall M_e \in \mathbb{M}(A_e)^{\leqslant |M_v|^2}$:

$$\mathrm{graphes}_e(M_v)(M_e) = \left\{ G \in \mathrm{graphes}_v(M_v) \,\middle|\, M_e = \biguplus_{e \in e(G)} el(G)(e) \right\}.$$

Soient $M_v \in \mathbb{M}(A_v)^{\leqslant n}$, et $M_e \in \mathbb{M}(A_e)^{\leqslant |M_v|^2}$. La valeur de f_v (resp. f_e) est la même en tout graphe de $\mathrm{graphes}_v(M_v)$ (resp. $\mathrm{graphes}_e(M_v)(M_e)$) :

– $\forall G \in \mathrm{graphes}_v(M_v)$:

$$f_v(G) = \prod_{l_v \in A_v} \mathrm{a_v}(l_v)^{M_v(l_v)} \times \mathrm{a_v}(\lambda)^{n-|M_v|};$$

– $\forall G \in \mathrm{graphes}_e(M_v)(M_e)$:

$$f_e(G) = \prod_{l_e \in A_e} \mathrm{a_e}(l_e)^{M_e(l_e)} \times \mathrm{a_e}(\lambda)^{|M_v|^2-|M_e|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2}.$$

De plus, le cardinal de $\mathrm{graphes}_e(M_v)(M_e)$ est égal au nombre suivant :

$$\text{r-P}(M_v, n-|M_v|) \times \text{r-P}(M_e, |M_v|^2 - |M_e|).$$

Nous avons donc :

$$\begin{aligned}
\sum_{G \in \mathrm{graphes}_e(M_v)(M_e)} f_e(G) &= \text{r-P}(M_v, n-|M_v|) \times \text{r-P}(M_e, |M_v|^2 - |M_e|) \\
&\quad \times \prod_{l_e \in A_e} \mathrm{a_e}(l_e)^{M_e(l_e)} \times \mathrm{a_e}(\lambda)^{|M_v|^2-|M_e|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2} \\
&= \text{r-P}(M_v, n-|M_v|) \times \mathrm{m}_e^{|M_v|^2}(M_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2}, \\
\sum_{G \in \mathrm{graphes}_e(M_v)(M_e)} [f_v(G) \times f_e(G)] &= \prod_{l_v \in A_v} \mathrm{a_v}(l_v)^{M_v(l_v)} \times \mathrm{a_v}(\lambda)^{n-|M_v|} \times \sum_{G \in \mathrm{graphes}_e(M_v)(M_e)} f_e(G) \\
&= \text{r-P}(M_v, n-|M_v|) \times \prod_{l_v \in A_v} \mathrm{a_v}(l_v)^{M_v(l_v)} \times \mathrm{a_v}(\lambda)^{n-|M_v|}
\end{aligned}$$

$$\times \mathrm{m}_e^{|M_v|^2}(M_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2}$$
$$= \mathrm{m}_v^n(M_v) \times \mathrm{m}_e^{|M_v|^2}(M_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2},$$

où m_v^n (resp. $\mathrm{m}_e^{|M_v|^2}$) désigne la mesure définie sur l'ensemble puissance de $\mathbb{M}(A_v)^{\leqslant n}$ (resp. $\mathbb{M}(A_e)^{\leqslant |M_v|^2}$) (cf. définition 62, section 5.3), avec $\mathrm{a_v}$ (resp. $\mathrm{a_e}$) la mesure de lettres associée.

Nous pouvons aisément remarquer que les fonctions $\mathrm{graphes}_v$ et, $\forall M_v \in \mathbb{M}(A_v)^{\leqslant n}$, $\mathrm{graphes}_e(M_v)$ possèdent toutes ces deux caractéristiques :

- *tout* graphe est associée à un *unique* multiensemble ;
- *aucun* multiensemble n'est associé à l'ensemble vide.

En d'autres termes, $\mathrm{graphes}_v$ (resp. $\forall M_v \in \mathbb{M}(A_v)^{\leqslant n}, \mathrm{graphes}_e(M_v)$) est une bijection de départ $\mathbb{M}(A_v)^{\leqslant n}$ (resp. $\mathbb{M}(A_e)^{\leqslant |M_v|^2}$) et d'arrivée une *partition* de $\mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$ (resp. $\mathrm{graphes}_v(M_v)$). Nous en concluons donc l'égalité suivante, d'après la proposition 4 :

$$\begin{aligned}
&\sum_{G \in \mathbb{G}(A_v, A_e, \{1,\ldots,n\})^{\leqslant n}} [f_v(G) \times f_e(G)] \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \sum_{G \in \mathrm{graphes}_v(M_v)} [f_v(G) \times f_e(G)] \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \sum_{M_e \in \mathbb{M}(A_e)^{\leqslant |M_v|^2}} \sum_{G \in \mathrm{graphes}_e(M_v)(M_e)} [f_v(G) \times f_e(G)] \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \sum_{M_e \in \mathbb{M}(A_e)^{\leqslant |M_v|^2}} \left(\mathrm{m}_v^n(M_v) \times \mathrm{m}_e^{|M_v|^2}(M_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2} \right) \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \left(\mathrm{m}_v^n(M_v) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2} \times \sum_{M_e \in \mathbb{M}(A_e)^{\leqslant |M_v|^2}} \mathrm{m}_e^{|M_v|^2}(M_e) \right) \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \left(\mathrm{m}_v^n(M_v) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2-|M_v|^2} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{|M_v|^2} \right) \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \left(\mathrm{m}_v^n(M_v) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2} \right) \\
&= \sum_{M_v \mathbb{M}(A_v)^{\leqslant n}} \mathrm{m}_v^n(M_v) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2} \\
&= \mathrm{a_v}(A_v \cup \{\lambda\})^n \times \mathrm{a_e}(A_e \cup \{\lambda\})^{n^2}.
\end{aligned}$$

■

D'après la définition 67 et la proposition 9, nous pouvons calculer la probabilité d'un graphe en $O\big((n \times \alpha_v) + (n^2 \times \alpha_e)\big)$, avec $O(\alpha_v)$ et $O(\alpha_e)$ les complexités d'une mesure selon $\mathrm{a_v}$ et $\mathrm{a_e}$, respectivement :

Définition 68 (Distribution uniforme de graphes)
$\forall G \in \mathbb{G}(A_v, A_e, \{1, \ldots, n\})^{\leqslant n}$:

$$\begin{aligned}
\mathrm{G}^n(G) = &\prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|} \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-n} \\
&\times \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{-|v(G)|^2}.
\end{aligned}$$

5.5.3 Préservation

Une propriété intéressante de notre distribution uniforme de graphes est la préservation du critère d'uniformité par agrandissement : l'agrandissement d'un graphe uniforme avec un sommet et des arêtes uniformes résulte en un graphe uniforme. Nous utilisons les termes « graphe uniforme » ou « graphe uniformément distribué » pour désigner un graphe dont la probabilité est assignée par une distribution uniforme.

Proposition 10 (Préservation d'uniformité de graphes)
$\forall G \in \mathbb{G}(A_v, A_e, \{1,\dots,n\})^{\leqslant n}, \forall l_v \in A_v \cup \{\lambda\}, \forall l_e \in A_e \cup \{\lambda\}, \forall l_{in} \in (A_e \cup \{\lambda\})^n, \forall l_{out} \in (A_e \cup \{\lambda\})^n$,
si toutes les conditions suivantes sont satisfaites :
- G est uniformément distribué selon g^n ;
- l_v uniformément distribué selon $\mathrm{a_v}$;
- l_e uniformément distribué selon $\mathrm{a_e}$;
- $\forall i \in \{1,\dots,n\}$, $(l_{in})_i$ est uniformément distribué selon $\mathrm{a_e}$;
- $\forall i \in \{1,\dots,n\}$, $(l_{out})_i$ est uniformément distribué selon $\mathrm{a_e}$,

alors $H = G \oplus^n (l_v, l_e, l_{in}, l_{out})$ est uniformément distribué selon g^{n+1}.

$\forall m_v \in A_v \cup \{\lambda\}$ (resp. $\forall m_e \in A_e \cup \{\lambda\}$), notons $\mathrm{A_v}(m_v)$ (resp. $\mathrm{A_e}(m_e)$) la probabilité de m_v (resp. m_e) selon une distribution uniforme selon $\mathrm{a_v}$ (resp. $\mathrm{a_e}$) :

$$\mathrm{A_v}(m_v) = \mathrm{a_v}(m_v) \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-1},$$
$$\mathrm{A_e}(m_e) = \mathrm{a_e}(m_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{-1}.$$

La formule suivante va nous être utile pour prouver cette dernière proposition :

Lemme 6
$\forall k \in \mathbb{N}$:

$$\sum_{K \in (A_e \cup \{\lambda\})^k} \prod_{i=1}^{k} \mathrm{A_e}(K_i) = 1.$$

Dès lors, nous avons :

Preuve (Proposition 10)
Si $l_v = \lambda$, nous avons :

$$\begin{aligned}
\mathrm{G}^{n+1}(H) &= \mathrm{G}^{n+1}(G) \\
&= \prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{(n+1)-|v(G)|} \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-(n+1)} \\
&\times \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{-|v(G)|^2} \\
&= \mathrm{a_v}(\lambda) \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-1} \\
&\times \prod_{v \in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|} \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-n} \\
&\times \prod_{e \in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e \cup \{\lambda\})^{-|v(G)|^2} \\
&= \mathrm{A_v}(\lambda) \times \mathrm{G}^n(G) \\
&= \mathrm{A_v}(\lambda) \times \mathrm{G}^n(G) \times \sum_{\substack{m_e \in A_e \cup \{\lambda\}, \\ m_{in} \in (A_e \cup \{\lambda\})^n, \\ m_{out} \in (A_e \cup \{\lambda\})^n}} \left(\mathrm{A_e}(m_e) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{in})_i) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{out})_i) \right) \\
&= \sum_{\substack{m_e \in A_e \cup \{\lambda\}, \\ m_{in} \in (A_e \cup \{\lambda\})^n, \\ m_{out} \in (A_e \cup \{\lambda\})^n}} \left(\mathrm{A_v}(\lambda) \times \mathrm{A_e}(m_e) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{in})_i) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{out})_i) \times \mathrm{G}^n(G) \right).
\end{aligned}$$

Sinon ($l_v \in A_v$), nous avons :

$$\begin{aligned}
&\mathrm{G}^{n+1}(H) \\
&= \prod_{v \in v(H)} \mathrm{a_v}(vl(H)(v)) \times \mathrm{a_v}(\lambda)^{(n+1)-|v(H)|} \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-(n+1)}
\end{aligned}$$

$$\times \prod_{e\in e(H)} \mathrm{a_e}(el(H)(e)) \times \mathrm{a_e}(\lambda)^{|v(H)|^2-|e(H)|} \times \mathrm{a_e}(A_e\cup\{\lambda\})^{-|v(H)|^2}$$
$$= vl(l_v) \times \mathrm{a_v}(A_v\cup\{\lambda\})^{-1}$$
$$\times \prod_{e\in e(H)\setminus e(G)} \mathrm{a_e}(el(H)(e)) \times \mathrm{a_e}(\lambda)^{(2\times|v(G)|)+1-|e(H)\setminus e(G)|} \times \mathrm{a_e}(A_e\cup\{\lambda\})^{-[(|2\times|v(G)|)+1]}$$
$$\times \prod_{v\in v(G)} \mathrm{a_v}(vl(G)(v)) \times \mathrm{a_v}(\lambda)^{n-|v(G)|} \times \mathrm{a_v}(A_v\cup\{\lambda\})^{-n}$$
$$\times \prod_{e\in e(G)} \mathrm{a_e}(el(G)(e)) \times \mathrm{a_e}(\lambda)^{|v(G)|^2-|e(G)|} \times \mathrm{a_e}(A_e\cup\{\lambda\})^{-|v(G)|^2}$$
$$= \mathrm{A_v}(l_v) \times \mathrm{G}^n(G)$$
$$\times \prod_{e\in e(H)\setminus e(G)} \left(\mathrm{a_e}(el(H)(e)) \times \mathrm{a_e}(A_e\cup\{\lambda\})^{-1}\right) \times \left(\mathrm{a_e}(\lambda) \times \mathrm{a_e}(A_e\cup\{\lambda\})^{-1}\right)^{(|2\times|v(G)|)+1-|e(H)\setminus e(G)|}$$
$$= \mathrm{A_v}(l_v) \times \prod_{e\in e(H)\setminus e(G)} \mathrm{A_e}(e) \times \mathrm{A_e}(\lambda)^{(|2\times|v(G)|)+1-|e(H)\setminus e(G)|} \times \mathrm{G}^n(G)$$
$$= \mathrm{A_v}(l_v) \times \mathrm{A_e}(l_e) \times \prod_{i\in v(G)} \mathrm{A_e}((l_{in})_i) \times \prod_{i\in v(G)} \mathrm{A_e}((l_{out})_i) \times \mathrm{G}^n(G)$$
$$= \mathrm{A_v}(l_v) \times \mathrm{A_e}(l_e) \times \prod_{i\in v(G)} \mathrm{A_e}((l_{in})_i) \times \prod_{i\in v(G)} \mathrm{A_e}((l_{out})_i) \times \mathrm{G}^n(G)$$
$$\times \sum_{\substack{m_{in}\in(A_e\cup\{\lambda\})^n|\forall i\in v(G),(m_{in})_i=(l_{in})_i,\\ m_{out}\in(A_e\cup\{\lambda\})^n|\forall i\in v(G),(m_{out})_i=(l_{out})_i}} \left(\prod_{i\in\{1,\dots,n\}\setminus v(G)} \mathrm{A_e}((m_{in})_i) \times \prod_{i\in\{1,\dots,n\}\setminus v(G)} \mathrm{A_e}((m_{out})_i)\right)$$
$$= \sum_{\substack{m_{in}\in(A_e\cup\{\lambda\})^n|\forall i\in v(G),(m_{in})_i=(l_{in})_i,\\ m_{out}\in(A_e\cup\{\lambda\})^n|\forall i\in v(G),(m_{out})_i=(l_{out})_i}}$$
$$\left(\mathrm{A_v}(l_v) \times \mathrm{A_e}(l_e) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{in})_i) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{out})_i) \times \mathrm{G}^n(G)\right).$$

Finalement dans tous les cas, d'après la définition 29, nous avons :

$$\mathrm{G}^{n+1}(H) = \sum_{\substack{F\in\mathbb{G}(A_v,A_e,\{1,\dots,n\})^{\leqslant n},\\ m_v\in A_v\cup\{\lambda\},\\ m_e\in A_e\cup\{\lambda\},\\ m_{in}\in(A_e\cup\{\lambda\})^n,\\ m_{out}\in(A_e\cup\{\lambda\})^n \end{subarray}\Big| F\oplus^n(m_v,m_e,m_{in},m_{out})=H}$$
$$\left(\mathrm{A_v}(m_v) \times \mathrm{A_e}(m_e) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{in})_i) \times \prod_{i=1}^{n} \mathrm{A_e}((m_{out})_i) \times \mathrm{G}^n(F)\right).$$

Nous en déduisons que la probabilité d'un graphe H uniforme selon g^{n+1} est égale à la somme des probabilités de tous les agrandissements possibles d'un graphe uniforme selon g^n avec un sommet et des arêtes potentiels uniformes selon $\mathrm{a_v}$ et $\mathrm{a_e}$, respectivement, tels que le résultat de cette opération soit H. Il s'agit exactement de la signification de la dernière proposition. ■

Il nous reste à prouver le lemme précédent :

Preuve (Lemme 6)
Soit f la fonction définie comme suit : $\forall k \in \mathbb{N}$:

$$f(k) = \sum_{K\in(A_e\cup\{\lambda\})^k} \prod_{i=1}^{k} \mathrm{A_e}(K_i).$$

Nous avons $f(0) = 1$, et $\forall k \in \mathbb{N}$:

$$\begin{aligned}
f(k+1) &= \sum_{L \in (A_e \cup \{\lambda\})^{k+1}} \prod_{i=1}^{k+1} \mathrm{A_e}(L_i) \\
&= \sum_{K \in (A_e \cup \{\lambda\})^k} \sum_{l_e \in (A_e \cup \{\lambda\})} \left(\prod_{i=1}^{k} \mathrm{A_e}(K_i) \times \mathrm{A_e}(l_e) \right) \\
&= \sum_{K \in (A_e \cup \{\lambda\})^k} \left(\prod_{i=1}^{k} \mathrm{A_e}(K_i) \times \sum_{l_e \in (A_e \cup \{\lambda\})} \mathrm{A_e}(l_e) \right) \\
&= \sum_{K \in (A_e \cup \{\lambda\})^k} \left(\prod_{i=1}^{k} \mathrm{A_e}(K_i) \times 1 \right) \\
&= \sum_{K \in (A_e \cup \{\lambda\})^k} \prod_{i=1}^{k} \mathrm{A_e}(K_i) \\
&= f(k).
\end{aligned}$$

Le lemme est donc vérifié par récurrence. ■

5.5.4 Génération

Nous souhaitons générer des graphes selon notre distribution uniforme.

Le cas $n = 0$ est évident, car l'unique graphe concerné est le graphe vide. Sinon ($n > 0$), d'après une utilisation récursive de la propriété exposée par la proposition 10, il est suffisant d'agrandir n fois le graphe vide avec des sommets et arêtes étiquetés uniformément selon $\mathrm{a_v}$ et $\mathrm{a_e}$, respectivement, pour obtenir un graphe uniforme selon g^n.

Une implémentation simple d'une telle procédure est donnée par l'algorithme 5, de complexité $O\left((n \times \alpha_v) + (n^2 \times \alpha_e)\right)$, avec $O(\alpha_v)$ et $O(\alpha_e)$ les complexités d'un choix aléatoire selon $\mathrm{A_v}$ et $\mathrm{A_e}$, respectivement.

5.5.5 Conclusion

Nous avons vu dans cette section comment définir une mesure pour ensembles de graphes de nombre de sommets limité, et sa normalisation en distribution uniforme.

Notre distribution uniforme possède la propriété intéressante de préservation par agrandissement, ce qui nous a permis de mettre au point un algorithme simple de génération de graphe uniforme, de complexité linéaire en la taille maximale des graphes considérés, et qui peut se résumer de la manière suivante : il est possible de générer un graphe uniforme en agrandissant des lettres uniformes. Cette simplicité est bien entendu due à la mesure particulière de graphes que nous avons définie, mais comme nous l'avons vu, cette mesure ne satisfait pas seulement notre souhait de simplicité mais est en plus pertinente par rapport au type de structures considéré, notamment à la combinatoire qui lui est inhérente.

Nous en avons maintenant terminé avec les spécifications de distributions uniformes de structures. Nous allons conclure par une discussion sur les perspectives possibles que soulèvent ces travaux.

5.6 Conclusion

Nous avons étudié dans ce chapitre comment mesurer différents ensembles de structures, en fonction de leur type, et donc de la combinatoire qu'elles soulèvent dans un cadre de taille limitée. Ces mesures nous ont permis de définir des distributions respectant un critère d'uniformité.

Ces distributions uniformes partagent comme point commun une intéressante propriété de préservation par une opération dépendant du type considéré. Dans tous les cas, l'opération concernée

Données : $n \in \mathbb{N}$.
Résultat : Un graphe uniforme selon g^n.
début
  $\mathrm{A_v} \longleftarrow$ distribution uniforme selon $\mathrm{a_v}$: $\forall l_v \in A_v \cup \{\lambda\}$:

$$\mathrm{A_v}(l_v) = \mathrm{a_v}(l_v) \times \mathrm{a_v}(A_v \cup \{\lambda\})^{-1};$$

  $\mathrm{A_e} \longleftarrow$ distribution uniforme selon $\mathrm{a_e}$: $\forall l_e \in A_e \cup \{\lambda\}$:

$$\mathrm{A_e}(l_e) = \mathrm{a_e}(l_e) \times \mathrm{a_e}(A_e \cup \{\lambda\})^{-1};$$

  $G \longleftarrow (\{\}, \{\})$;
  pour $i \longleftarrow 1$ **à** n **faire**
    $l_v \longleftarrow$ choix aléatoire selon $\mathrm{A_v}$;
    si $l_v \neq \lambda$ **alors**
      $vl(G) \longleftarrow vl(G) \cup \{(i, l_v)\}$;
      $l_e \longleftarrow$ choix aléatoire selon $\mathrm{A_e}$;
      si $l_e \neq \lambda$ **alors**
        $el(G) \longleftarrow el(G) \cup \{((i, i), l_e)\}$;
      fin
      pour $j \longleftarrow 1$ **à** $i - 1$ **faire**
        si $j \in v(G)$ **alors**
          $l_{in} \longleftarrow$ choix aléatoire selon $\mathrm{A_e}$;
          si $l_{in} \neq \lambda$ **alors**
            $el(G) \longleftarrow el(G) \cup \{((i, j), l_{in})\}$;
          fin
          $l_{out} \longleftarrow$ choix aléatoire selon $\mathrm{A_e}$;
          si $l_{out} \neq \lambda$ **alors**
            $el(G) \longleftarrow el(G) \cup \{((j, i), l_{out})\}$;
          fin
        fin
      fin
    fin
  fin
  retourner G;
fin

Algorithme 5 : Génération d'un graphe uniforme.

est simple et nous avons pu déduire de ladite propriété un algorithme de génération de structure uniforme, de complexité linéaire en la taille maximale des structures considérées. L'algorithme se résume à chaque fois comme une procédure construisant son résultat en « ajoutant » itérativement des composantes uniformes à une structure initialisée vide, c'est-à-dire n'ayant aucune composante.

Au-delà de leur qualité de simplicité qui devient un atout pour les taches génératives, nos mesures sont toujours pertinentes car intimement liées à la combinatoire sous-jacente au type visé, ce dans un cadre de taille limitée mais variable. Nos mesures représentent et tiennent compte explicitement du vide implicite pouvant survenir lors d'opérations basiques de construction de telles structures. Dès lors, l'apparition d'un tel vide devient la cause de la non-maximalité en taille de certaines structures dans un échantillon de données, et justifie la non-équiprobabilité générale de nos distributions uniformes.

D'un point de vue applicatif, il serait intéressant de mettre au point des tests statistiques permettant d'estimer la déviation de l'uniformité d'un échantillon de structures.

Une autre perspective, celle-ci générale au problème de traitement statistique d'ensembles de structures, est la formalisation du théorème central limite pour de tels espaces probabilisés, qui serait le critère à vérifier pour qu'une probabilité soit qualifiée de « normale ». Une précondition est la définition du concept de variable aléatoire de structures à valeurs dans un espace vectoriel probabilisable, espace qui permettrait ainsi de traiter les structures comme des *vecteurs*, c'est-à-dire des éléments assujettis à des *combinaisons linéaires*. Nous pourrions ainsi aisément définir et calculer les moments de telles variables aléatoires structurelles.

Dans le chapitre suivant, nous allons tout d'abord avoir une approche plus locale pour le cas de chaînes, en concaténant des lettres dont la probabilité est contrôlée par une densité de probabilité normale dont l'écart-type et la discrétisation sont paramétrés par une fonction de coût. Nous nous attellerons ensuite à ce problème de vecteurs et proposerons une réflexion quand à une approche théorique générale et globale.

Chapitre 6

Distribution gaussienne

Dans ce chapitre, nous proposons la définition d'une distribution de chaînes dont les valeurs sont contrôlées par une densité de probabilité gaussienne.

Nous optons pour l'approche proposée par Jolion [Jol03a] pour étendre le concept d'*écart* dans l'espace des chaînes, et utilisons un procédé de discrétisation pour passer du domaine numérique continu à celui des lettres d'un alphabet.

6.1 Normalité

Par « distribution normale » ou « gaussienne », nous entendons classiquement la loi de probabilité définie dans le domaine numérique de la manière suivante :

Définition 69 (Loi normale)
Soient $\mu \in \mathbb{R}$ et $\sigma \in \mathbb{R}^+$. La loi normale de moyenne μ et variance σ^2 est la probabilité sur $(\mathbb{R}, \beta)$ (cf. définition 115) de densité suivante : $\forall x \in \mathbb{R}$:

$$\mathcal{N}(\mu, \sigma^2)(x) = \frac{1}{\sigma \times \sqrt{2 \times \pi}} \times \exp\left(-\frac{1}{2} \times \left(\frac{x-\mu}{\sigma}\right)^2\right).$$

Exemple
Soit P la probabilité de densité $\mathcal{N}(\mu, \sigma^2)$. Nous avons :

$$\begin{aligned}
\mathrm{P}\left([\mu - (1 \times \sigma); \mu + (1 \times \sigma)]\right) &\simeq 68\ \%, \\
\mathrm{P}\left([\mu - (2 \times \sigma); \mu + (2 \times \sigma)]\right) &\simeq 95\ \%, \\
\mathrm{P}\left([\mu - (3 \times \sigma); \mu + (3 \times \sigma)]\right) &\simeq 99\ \%.
\end{aligned}$$

Ces inéquations sont bien connues.

Comme nous le verrons plus tard (cf. section 6.6.1), cette loi joue un rôle très important en termes de comportement asymptotique des autres lois de probabilités.

6.2 Probabilité

Soient A un alphabet, auquel nous associons λ en qualité de lettre vide, et c une fonction de coût sur A. Dans cette section, nous proposons tout d'abord la définition d'une distribution gaussienne de lettres, sur l'ensemble puissance de $A \cup \{\lambda\}$, notée $\mathrm{n}(d)$, avec d une opération d'édition sur A, paramètre jouant à la fois les rôles de « moyenne » et d'« écart-type ». Ensuite, nous « étendons » cette définition aux chaînes, résultant en une distribution gaussienne sur l'ensemble puissance de $\mathbb{S}(A)^*$, notée $\mathrm{N}(D)$, avec D une chaîne d'édition sur A, paramètre jouant également à la fois les rôles de moyenne et d'écart-type.

Comme nous pouvons le voir avec la définition 69, la densité de probabilité gaussienne est paramétrée par sa moyenne μ et son écart-type σ. Ces deux données sont nommées ainsi car les statistiques de mêmes noms d'une variable aléatoire réelle distribuée selon une loi normale sont effectivement égales à leurs valeurs respectives. La densité de probabilité associée à une valeur réelle x est uniquement dépendante de l'*écart* de x à μ. Cet écart, égal à la valeur absolue de la différence entre x et μ, $|x-\mu|$, peut être simplement vu de manière générale comme une *distance*, ou du moins toute notion de *dissimilarité*.

Dans le cas de lettres d'un alphabet, nous traduisons ce raisonnement de la manière suivante : un écart à la lettre moyenne est une opération d'édition la transformant. Le paramètre que nous nommons « moyenne » est une lettre de référence, élément de $A \cup \{\lambda\}$, et celui que nous nommons « écart-type » est une opération d'édition de référence, qui transforme la lettre moyenne. De fait, la donnée de l'écart-type est suffisante pour paramétrer entièrement notre distribution.

Soit $m \to d$ une opération d'édition sur A. La probabilité par $\mathrm{n}(m \to d)$ d'une lettre de $A \cup \{\lambda\}$ est donnée par discrétisation d'une densité de probabilité gaussienne de moyenne $c(m \to m) = 0$ et d'écart-type $c(m \to d)$. La fonction de coût c nous permet donc de passer du concept d'écart dans le domaine discret de l'alphabet à celui de dissimilarité dans le domaine numérique continu de la densité de probabilité gaussienne. La densité discrétisée est de moyenne 0, car ne pas transformer la lettre moyenne est de coût nul, et d'écart-type le coût de l'opération d'édition ainsi nommée. Le processus de discrétisation est une simple normalisation, rien de plus classique pour passer d'une densité à une masse de probabilité dans un ensemble fini.

Définition 70 (Loi normale de lettres)
$\forall m \to d \in \mathrm{edit}(A), \forall l \in A \cup \{\lambda\}$:

$$\mathrm{n}(m \to d)(l) = \frac{g\left(c(m \to l)\right)}{\sum_{k \in A \cup \{\lambda\}} g\left(c(m \to k)\right)},$$

avec $g = \mathcal{N}\left(0, c(m \to d)^2\right)$.

L'extension aux chaînes est simple : l'écart-type est une chaîne d'édition transformant la chaîne moyenne. Nous optons donc pour l'approche proposée par Jolion [Jol03a], où la notion d'écart à une chaîne de référence est traduite par une chaîne d'édition transformant cette référence. La probabilité d'une chaîne X selon une loi normale $\mathrm{N}(D)$ est égale à la probabilité d'obtenir X par concaténation de $|D|$ lettres l_i respectivement distribuées selon $\mathrm{n}(D(i))$, $i \in \{1, \ldots, |D|\}$, c'est-à-dire dont la probabilité est assignée par $\mathrm{n}(D(i))$. Nous obtenons donc la formule suivante :

Définition 71 (Loi normale de chaînes)
$\forall D \in \mathbb{S}\left(\mathrm{edit}(A)\right)^*, \forall X \in \mathbb{S}(A)^*$:

$$\mathrm{N}(D)(X) = \sum_{\lambda\text{-X} \in \lambda\text{-chaînes}^{|D|}(X)} \prod_{i=1}^{|D|} \mathrm{n}(D(i))(\lambda\text{-X}(i)),$$

où λ-chaînes$^{|D|}(X)$ désigne l'ensemble des chaînes sur $A \cup \{\lambda\}$ de longueur $|D|$ et dont la concaténation sur A des lettres est égale à X, notation introduite en section 5.2.1.

Afin de calculer cette probabilité sans énumérer λ-chaînes$^{|D|}(X)$, de cardinal exponentiel en $|D|$, nous avons mis au point une méthode de programmation dynamique, détaillée par l'algorithme 6, de complexité temporelle $O(|X| \times |D|^3)$ et spatiale $O(|D|)$, et qui peut se résumer de la manière suivante :

- à chaque itération j de la boucle centrale principale, $\mathrm{PP}(i)$ est égal à la probabilité du préfixe de longueur j de X selon une loi normale d'écart-type le préfixe de longueur i de D ;
- l'initialisation de $\mathrm{PP}(i)$ tient compte du fait qu'aucune chaîne de λ-chaînes$^{|D|}(X)$ n'a de préfixe de longueur supérieure à $|D|-|X|$ entièrement composé de λ, autrement dit la position maximale d'apparition de la première lettre de X dans une chaîne de λ-chaînes$^{|D|}(X)$ est égale à $|D| - |X| + 1$;
- $\mathrm{N}(D)(X)$ est obtenue par addition des valeurs finales de $\mathrm{PP}(i)$, éventuellement complétées par des probabilités de λ selon des loi normales de lettres aux positions de D supérieures à i ;

– le cas où X est vide ne rentre pas dans ce schéma général, et est donc traité de manière spéciale.

6.3 Préservation

Une propriété intéressante de notre distribution gaussienne de chaînes est la préservation du critère de normalité par concaténation : la concaténation de deux chaînes gaussiennes résulte en une chaîne gaussienne. Nous utilisons les termes « chaîne gaussienne », « chaîne normale », ou « chaîne normalement distribuée », pour désigner une chaîne dont la probabilité est assignée par une distribution gaussienne.

Proposition 11 (Préservation de normalité de chaînes)
$\forall n \in \mathbb{N}, \forall i \in \{0, \ldots, n\}, \forall Y \in \mathbb{S}(A)^{\leqslant i}, \forall Z \in \mathbb{S}(A)^{\leqslant n-i}, \forall E \in \mathbb{S}\left(\text{edit}(A)\right)^{i}, \forall F \in \mathbb{S}\left(\text{edit}(A)\right)^{n-i}$, si Y est distribuée selon $\text{N}(E)$, et Z distribuée selon $\text{N}(F)$, alors $X = Y.Z$ est distribuée selon $\text{N}(D = E.F)$.

Preuve (Proposition 11)

$$
\begin{aligned}
& \text{N}(D)(X) \\
= & \sum_{\lambda\text{-X} \in \lambda\text{-chaînes}^n(X)} \prod_{j=1}^{n} \text{n}(D(j))(\lambda\text{-X}(j)) \\
= & \sum_{\lambda\text{-Y} \in \mathbb{S}(A \cup \{\lambda\})^i, \lambda\text{-Z} \in \mathbb{S}(A \cup \{\lambda\})^{n-i} \mid \lambda\text{-Y} \,.\, \lambda\text{-Z} = \lambda\text{-X}} \left[\prod_{j=1}^{i} \text{n}(E(j))(\lambda\text{-Y}(j)) \times \prod_{j=1}^{n-i} \text{n}(F(j))(\lambda\text{-Z}(j)) \right] \\
= & \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} \mid Y.Z = X} \\
& \left[\sum_{\lambda\text{-Y} \in \lambda\text{-chaînes}^i(Y)} \prod_{j=1}^{i} \text{n}(E(j))(\lambda\text{-Y}(j)) \times \sum_{\lambda\text{-Z} \in \lambda\text{-chaînes}^{n-i}(Z)} \prod_{j=1}^{n-i} \text{n}(F(j))(\lambda\text{-Z}(j)) \right] \\
= & \sum_{Y \in \mathbb{S}(A)^{\leqslant i}, Z \in \mathbb{S}(A)^{\leqslant n-i} \mid Y.Z = X} \left[\text{N}(E)(Y) \times \text{N}(F)(Z) \right].
\end{aligned}
$$

Nous en déduisons que la probabilité d'une chaîne X distribuée selon $\text{N}(E.F)$ est égale à la somme des probabilités de toutes les concaténations possibles d'une chaîne Y distribuée selon $\text{N}(E)$ avec une chaîne Z distribuée selon $\text{N}(F)$, telles que $Y.Z = X$. Il s'agit exactement de la signification de la dernière proposition. ■

Le raisonnement général ci-dessus nous conduit au résultat spécifique suivant, utile en 6.4 :

Corollaire 3 (Préservation de normalité de chaînes)
$\forall n \in \mathbb{N}, \forall X \in \mathbb{S}(A)^{\leqslant n}, \forall l \in A \cup \{\lambda\}, \forall D \in \mathbb{S}\left(\text{edit}(A)\right)^{n}, \forall d \in \text{edit}(A)$, si X est distribuée selon $\text{N}(D)$, et l distribuée selon $\text{n}(d)$, alors $X.l$ est distribuée selon $\text{N}(D.d)$.

La preuve du corollaire ci-dessus découle du lemme suivant :

Lemme 7
$\forall l \in A \cup \{\lambda\}, \forall d \in \text{edit}(A)$, l est distribuée selon $\text{n}(d)$ ssi $\{\}.l$ est distribuée selon $\text{N}(\{\}.d)$.

Dès lors, nous avons :

Preuve (Corollaire 3)
D'après le lemme 7, $\{\}.l$ est distribuée selon $\text{N}(\{\}.d)$. Alors, d'après la proposition 11, $X.(\{\}.l) = X.l$ est distribuée selon $\text{N}\left(D.(\{\}.d)\right) = \text{N}(D.d)$. ■

Il nous reste à prouver le lemme précédent :

Données : $X \in \mathbb{S}(A)^*, D \in \mathbb{S}\left(\text{edit}(A)\right)^*$.
Résultat : $\text{N}(D)(X)$.
début
  /* Cas spécial de la chaîne vide. */
  si $X = \{\}$ **alors**
    retourner $\text{N}(D)(\{\})$; (algorithme 7)
  fin
  /* Initialisation : production de la première lettre de X. */
  pour $i \longleftarrow 1$ **à** $|D| - |X| + 1$ **faire**
    // Production de $X(1)$ en position i de D
    $\text{PP}(i) \longleftarrow \text{n}(D(i))(X(1))$;
    // Complétion à gauche avec la production de λ
    pour $l \longleftarrow 1$ **à** $i - 1$ **faire**
      $\text{PP}(i) \longleftarrow \text{PP}(i) \times \text{n}(D(l))(\lambda)$;
    fin
  fin
  pour $i \longleftarrow |D| - |X| + 2$ **à** $|D|$ **faire**
    // Improbable de produire $X(1)$ à cette position i de D
    $\text{PP}(i) \longleftarrow 0$;
  fin
  /* Itérations : production des autres lettres de X. */
  pour $j \longleftarrow 2$ **à** $|X|$ **faire**
    // Possibilités de production de $X(j)$ en position i de D
    pour $i \longleftarrow |D|$ **à** 1 **faire**
      $\text{PP}(i) \longleftarrow 0$;
      // Possibilités de production de $X(j-1)$ en position $k < i$ de D
      pour $k \longleftarrow 1$ **à** $i - 1$ **faire**
        // Production de $X(j-1)$ en position k de D
        $\text{PLUS} \longleftarrow \text{PP}(k)$;
        // Complétion avec la production de λ en positions $(k+1) \leqslant l \leqslant (i-1)$
        pour $l \longleftarrow k + 1$ **à** $i - 1$ **faire**
          $\text{PLUS} \longleftarrow \text{PLUS} \times \text{n}(D(l))(\lambda)$;
        fin
        // Addition des probabilités de production de $X(j-1)$
        $\text{PP}(i) \longleftarrow \text{PP}(i) + \text{PLUS}$;
      fin
      // Production de $X(j)$ en position i de D
      $\text{PP}(i) \longleftarrow \text{PP}(i) \times \text{n}(D(i))(X(j))$;
    fin
  fin
  /* Finalisation : addition des probabilités de production de X. */
  $P \longleftarrow 0$;
  pour $i \longleftarrow 1$ **à** $|D|$ **faire**
    // La production de X s'est stoppée à cette position i de D
    $\text{PLUS} \longleftarrow \text{PP}(i)$;
    // Complétion à droite avec la production de λ
    pour $l \longleftarrow i + 1$ **à** $|D|$ **faire**
      $\text{PLUS} \longleftarrow \text{PLUS} \times \text{n}(D(l))(\lambda)$;
    fin
    // Addition des probabilités de production de X
    $P \longleftarrow P + \text{PLUS}$;
  fin
  retourner P;
fin

Algorithme 6 : Probabilité d'une chaîne selon une loi normale.

Données : $D \in \mathbb{S}\,(\text{edit}(A))^*$.
Résultat : $\text{N}(D)(\{\})$.
début
  $P \longleftarrow 1$;
  pour $i \longleftarrow 1$ **à** $|D|$ **faire**
    $P \longleftarrow P \times \text{n}(D(i))(\lambda)$;
  fin
  retourner P;
fin

Algorithme 7 : Probabilité de la chaîne vide selon une loi normale.

Preuve (Lemme 7)
Nous avons :

$$\lambda\text{-chaînes}^{|\{\}.d|}(\{\}.l) = \lambda\text{-chaînes}^{1}(\{\}.l) = l.$$

Nous avons donc :

$$\text{N}(\{\}.d)(\{\}.l) = \prod_{i=1}^{1} \text{n}((\{\}.d)(i))(l_i) = \text{n}(d)(l).$$

Ce résultat prouve le lemme, car la promotion en chaîne est une bijection (cf. définition 10). ■

6.4 Génération

Nous souhaitons générer des chaînes selon notre distribution gaussienne.

Le cas où la chaîne écart-type D est vide est évident, car l'unique chaîne concernée est la chaîne vide. Sinon ($|D| > 0$), d'après une utilisation récursive de la propriété exposée par le corollaire 3, il est suffisant de concaténer à la chaîne vide n lettres l_i distribuées selon $\text{n}(D(i))$ ($i \in \{1, \ldots, |D|\}$) pour obtenir une chaîne distribuée selon $\text{N}(D)$.

Une implémentation simple d'une telle procédure est donnée par l'algorithme 8, de complexité $O(|D| \times \alpha)$, avec $O(\alpha)$ la complexité d'un choix aléatoire selon n.

Données : $D \in \mathbb{S}\,(\text{edit}(A))^*$.
Résultat : Une chaîne distribuée selon $\text{N}(D)$.
début
  $X \longleftarrow \{\}$;
  pour $i \longleftarrow 1$ **à** $|D|$ **faire**
    $l \longleftarrow$ choix aléatoire selon $\text{n}(D(i))$;
    $X \longleftarrow X.l$;
  fin
  retourner X;
fin

Algorithme 8 : Génération d'une chaîne gaussienne.

Pour information, il existe un grand nombre de techniques permettant de faire le choix aléatoire d'un élément d'un ensemble fini E selon une probabilité définie sur 2^E. Par exemple, le lecteur peut s'informer sur la méthode de la *roulette* (*roulette wheel selection* [Whi94]).

6.5 Estimation

Soit S une séquence (échantillon) de chaînes supposées générées selon un processus simulant une distribution normale (par exemple l'algorithme 8) d'écart-type D inconnu. Notre objectif est d'estimer D à partir de S.

Nous avons recours à la technique de *maximisation de vraisemblance* : nous estimons D par la chaîne d'édition $\widehat{D_{\max}}$ avec laquelle la loi normale maximise la probabilité de S :

$$\begin{aligned}\widehat{D_{\max}} &= \arg\max \left\{ \widehat{D} \in \mathbb{S}\left(\mathrm{edit}(A)\right)^* \right\} \prod_{i=1}^{|S|} \mathrm{N}(\widehat{D})(S_i) \\ &= \arg\max \left\{ \widehat{D} \in \mathbb{S}\left(\mathrm{edit}(A)\right)^* \right\} \log \left(\prod_{i=1}^{|S|} \mathrm{N}(\widehat{D})(S_i) \right) \\ &= \arg\max \left\{ \widehat{D} \in \mathbb{S}\left(\mathrm{edit}(A)\right)^* \right\} \sum_{i=1}^{|S|} \log \left(\mathrm{N}(\widehat{D})(S_i) \right).\end{aligned}$$

Le passage d'un produit à une somme de logarithmes répond à une préoccupation purement implémentationnelle : sans changer le résultat de l'équation ci-dessus, il permet de repousser les dépassements de capacité de stockage des nombres réels (dans la norme IEEE 754 de codage des nombres flottants sur 64 bits, la valeur strictement positive minimale pouvant être représentée est $2^{-1074} \simeq \exp(-744{,}44)$).

Nous intuitons que ce problème d'estimation est NP-difficile, c'est-à-dire que le seul moyen de trouver $\widehat{D_{\max}}$ dans le cas général est d'énumérer l'ensemble des chaînes d'édition sur A, ou au moins celles de longueur au plus n, avec n la longueur maximale d'une chaîne de S. Or ce procédé requiert un temps de calcul exponentiel en n.

Quand l'informaticien doit résoudre un problème combinatoire qu'il sait ou au moins suppose NP-difficile, s'offre à lui deux possibilités :

1. soit il a recours à des techniques d'optimisation combinatoire (métaheuristiques) afin d'arriver à un résultat qui a de fortes chances d'être approximé et donc n'être que sous-optimal ;
2. soit il doit se résoudre à considérer seulement les éventuelles instances de son problème pour lesquelles la solution exacte est obtenable en temps polynomial.

Il existe un grand nombre de métaheuristiques (génétique, locale, colonies de fourmis…), chacune étant plus ou moins performante en fonction de caractéristiques propres au type de problème considéré, et des paramètres particuliers à chaque instance. Le lecteur peut se référer à l'article de Blum et Roli [BR03] pour en avoir un aperçu assez complet.

Pour ce qui est du deuxième point de l'énumération ci-dessus, notons qu'il existe une classe d'instances de notre problème pour lesquelles nous pouvons calculer $\widehat{D_{\max}}$ en temps polynomial. En effet, supposons que la fonction c soit telle que les coûts de toutes les insertions et suppressions sont infinis : $\forall l \in A$:

$$c(\lambda \to l) = c(l \to \lambda) = \infty,$$

alors les propositions suivantes sont nécessairement vérifiées :

- D est composée uniquement de substitutions ;
- toutes les chaînes de S sont de même longueur, égale à la longueur de D ;
- toutes les lettres à la position j dans une chaîne de S sont distribuées selon $\mathrm{n}(D(j))$ ($j \in \{1, \ldots, |D|\}$),

et il nous est donc possible de calculer $\widehat{D_{\max}}$ en $O(|D| \times |A|^2 \times |S|)$, grâce à la formule suivante :

$$\widehat{D_{\max}}(j) = \arg\max \left\{ \widehat{d} \in A^2 \right\} \prod_{i=1}^{|S|} \mathrm{n}(\widehat{d})(S_i(j)),$$

où, pour rappel, A^2 est égal à l'ensemble des substitutions sur A ($\mathrm{edit}(A) = (A \cup \{\lambda\})^2$).

La figure 6.1 nous montre le résultat d'une expérimentation d'estimation dans ce cas précis. Les paramètres sont les suivants :

- A est l'alphabet latin basique (ou ASCII, intervalle [U+0000; U+007F] dans la norme Unicode) privé de ses caractères de contrôle C0 (intervalle [U+0000; U+001F], ainsi que U+007F). Les 95 lettres de type caractère de A sont donc accessibles par n'importe quel clavier utilisant l'alphabet latin ;

		•	•	•		
	•				•	
	•				•	
	•				•	
	•				•	
	•				•	
		•	•	•		

TABLE 6.1 – Codage visuel du caractère O par une matrice binaire de taille (7×7).

- le coût par c d'une substitution $l \to k$ est égal à la distance de *Hamming* [Ham50] entre les codages visuels respectifs de l et k selon une matrice binaire de taille (7×7). Un exemple de tel codage est donné dans le tableau 6.1 ;
- $|D| = 100$,

et le protocole est le suivant :

1. générer D aléatoirement ;
2. générer S selon $\mathrm{N}(D)$;
3. calculer la vraisemblance L de S selon $\mathrm{N}(D)$:

$$L = \prod_{i=1}^{|S|} \mathrm{N}(D)(S_i);$$

4. calculer l'estimation $\widehat{D_{\max}}$ de D ;
5. calculer la vraisemblance $\widehat{L_{\max}}$ de S selon $\mathrm{N}(\widehat{D_{\max}})$:

$$\widehat{L_{\max}} = \prod_{i=1}^{|S|} \mathrm{N}(\widehat{D_{\max}})(S_i);$$

6. calculer le ratio suivant :

$$R = \frac{L^{1/|S|}}{\widehat{L_{\max}}^{1/|S|}}.$$

Une telle valeur R supérieure (resp. inférieure) à 1 signifie qu'en moyenne une chaîne de S est R fois plus (resp. $1/R$ fois moins) vraisemblable selon $\mathrm{N}(D)$ que selon $\mathrm{N}(\widehat{D_{\max}})$.

Nous avons divisé l'expérimentation en deux niveaux de difficulté :

1. estimation d'écart-type avec moyenne connue ;
2. estimation d'écart-type avec moyenne inconnue.

Dans le premier cas, seule la chaîne en sortie de l'écart-type est à estimer, celle en entrée étant bien entendue fixée à la moyenne. Par contre, dans le deuxième cas, la chaîne en entrée de l'écart-type est également à estimer, constituant par là même une estimation de la moyenne.

Les résultats sont positifs, à savoir que nous pouvons noter une convergence très rapide de nos estimations vers une distribution pour laquelle l'échantillon est aussi vraisemblable que pour la distribution l'ayant généré. Par « très rapide », nous entendons qu'un nombre de chaînes de l'ordre de 2000 est ridiculement faible en comparaison avec le nombre 95^{100} de chaînes à probabilité non nulle selon $\mathrm{N}(D)$ (ce qui est le cas au vu des paramètres choisis). Les cas où le ratio R est inférieur à 1 témoignent d'une sur-modélisation de S par nos estimations, due au fait que S est de taille trop faible pour pouvoir assurer une bonne représentativité de $\mathrm{N}(D)$, et qu'il existe donc une ou plusieurs lois normales pour la(les)-quelle(s) S est plus vraisemblable.

Enfin, la qualité de notre processus d'estimation est emphasée par les valeurs aberrantes de R exposées en figure 6.2, qui ont été obtenues par une loi normale dont l'écart-type a été initialisé aléatoirement. Cette courbe témoigne de la difficulté de notre problème, même dans le cas d'instances relativement simples, telles que celle que nous avons étudiée.

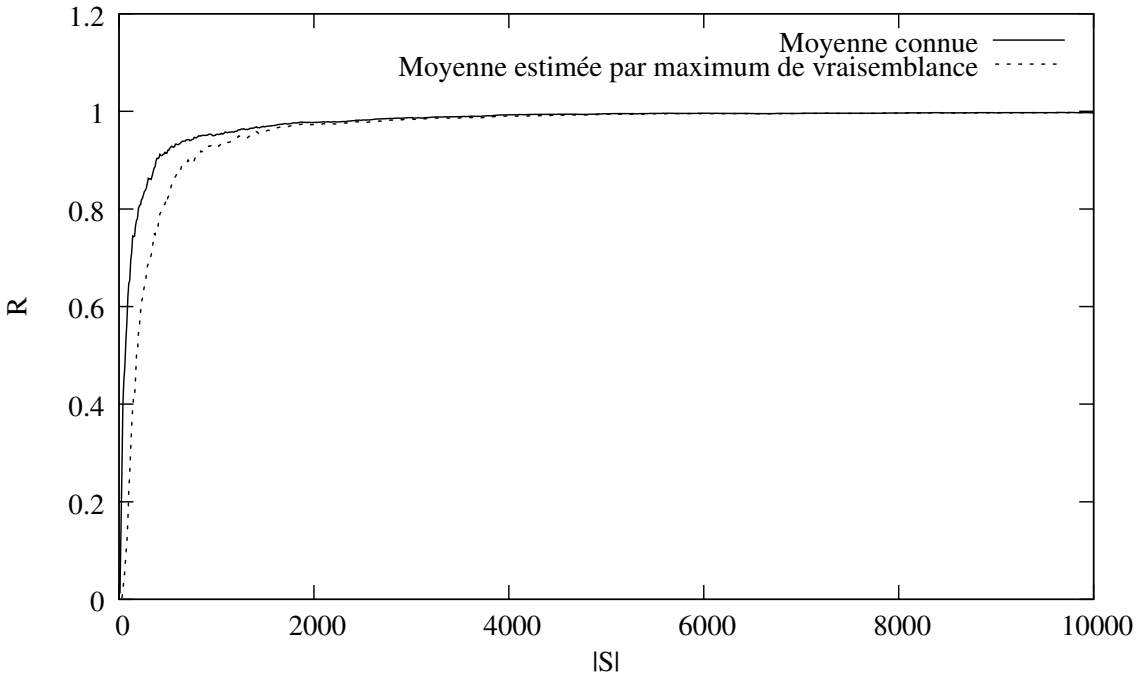

FIGURE 6.1 – Ratios des vraisemblances moyennes de l'échantillon de chaînes selon les lois normales estimées.

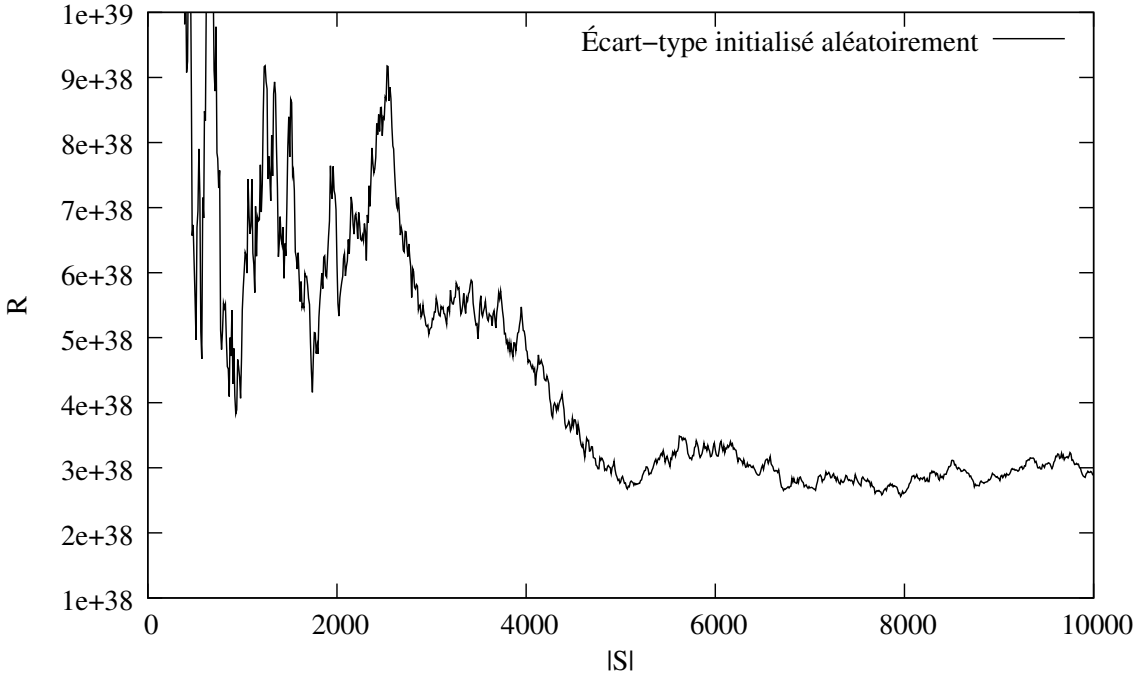

FIGURE 6.2 – Ratios des vraisemblances moyennes de l'échantillon de chaînes selon une loi normale d'écart-type aléatoire.

6.6 Vers une approche plus théorique

L'approche détaillée précédemment est plutôt *locale*, et nous avons tenté de reproduire la *forme* de la loi normale dans un espace de chaînes par le biais de concaténation de distributions dans un espace de lettres, dont le critère de normalité imité se situe bien plus au niveau des écarts à la lettre moyenne qu'au niveau des lettres elles-mêmes.

Une loi normale telle que nous l'entendons classiquement est une distribution qui a pour caractéristique l'important théorème central limite (TCL), dont nous voudrions qu'il soit une propriété à respecter par une distribution pour que celle-ci soit qualifiée de « normale » dans son espace. Le critère de normalité ne serait donc pas analytique, comme l'est l'expression de la densité de probabilité normale, mais théorique.

Nous allons maintenant analyser les qualités générales que doit avoir un espace probabilisable pour que le TCL puisse éventuellement y être exprimé.

6.6.1 Le théorème central limite

Théorème 1 (Théorème central limite (TCL))
Soient $n \in \mathbb{N} \setminus \{0\}$, $X_1, \ldots, X_n$ une séquence de n variables aléatoires i.i.d., chacune de moyenne finie μ et de variance finie σ^2, et S_n la variable aléatoire « somme » définie comme suit :

$$S_n = \sum_{i=1}^{n} X_n.$$

La distribution de S_n tend vers une loi normale de moyenne $(n \times \mu)$ et variance $(n \times \sigma^2)$ quand n tend vers l'infini.

Nous en déduisons que la distribution de la variable aléatoire « moyenne », $M_n = S_n/n$, tend vers une loi normale de moyenne μ et variance (σ^2/n) quand n tend vers l'infini.

Ce théorème signifie que si n'avons pas beaucoup de connaissances sur un processus physique, mais que nous pouvons supposer que ses résultats sont déterminés par un grand nombre de sous-résultats additifs dans leur effet, alors il est raisonnable d'approximer la modélisation de ce processus par une variable aléatoire normalement distribuée.

Le TCL est classiquement appliqué aux variable réelles. Pour pouvoir être utilisée de manière générale, sa formulation requiert que nous puissions au moins additionner les variables et les normaliser, c'est-à-dire les multiplier (diviser) par un *scalaire*. En d'autres termes, ces variables doivent prendre leurs valeurs dans un *espace vectoriel*.

Nous pensons qu'il est possible de définir un tel espace pour variables aléatoires de chaînes qui pourraient s'additionner et être multipliées par un nombre réel, et proposons une première réflexion sur le sujet dans la section suivante.

6.6.2 Un espace vectoriel de chaînes

Soit $\mathcal{R}$ le corps commutatif classique des nombres réels $(\mathbb{R}, +, \times)$. Nous proposons dans cette section la définition d'un espace vectoriel sur $\mathcal{R}$ incorporant une représentation réelle des lettres d'un alphabet, et son extension aux chaînes.

Extension de lettres

De manière à pouvoir incorporer A dans un espace vectoriel sur $\mathcal{R}$, nous étendons tout d'abord les concepts d'alphabet et de lettre.

Définition 72 (Alphabet étendu)
L'alphabet étendu de A est l'ensemble $\mathbb{R}^{|A|}$.

Définition 73 (Lettre étendue)
Une lettre étendue de A est un élément de l'alphabet étendu de A.

Définition 74 (Extension d'alphabet)
Une extension de A est une injection définie comme suit :

$$\begin{aligned} A &\to \mathbb{R}^{|A|} \\ l &\to 1_i^A \,\middle|\, i \in \{1, \ldots, |A|\}, \end{aligned}$$

avec 1_i^A la lettre étendue de A pour laquelle toutes les coordonnées sont nulles, excepté la i-ème qui est égale à 1.

Soit $\text{ext}(A)$ une telle fonction. $\text{ext}(A)$ définit une indexation de A dans son alphabet étendu, c'est-à-dire une mise en correspondance bijective entre A et un sous-ensemble de cardinal $|A|$ de lettres étendues de A. La i-ème coordonnée d'une lettre étendue e de A est interprétée comme le degré (positif ou négatif) dans e de (l'unique élément de) $(\text{ext}(A))^{-1}\left(1_i^A\right)$. Pour faire simple, nous allons parler aussi bien d'une lettre de A que de la lettre étendue de A qui lui est associée par $\text{ext}(A)$, et donc aussi bien de A que de $\text{ext}(A)(A)$.

Nous allons maintenant définir un espace vectoriel sur $\mathcal{R}$ nous permettant de manipuler les lettres de A dans son extension, et pour lequel A est une base.

Représentation vectorielle de lettres

Soit $\mathcal{L} = (\mathbb{R}^{|A|}, +, \times)$, avec $+$ et $\times$ les addition et scaling coordonnée-par-coordonnée des espaces vectoriels classiques sur $\mathbb{R}^n$, $n \in \mathbb{N}$.

$\mathcal{L}$ n'est donc rien d'autre qu'un espace numérique classique dont il est inutile de prouver qu'il est vectoriel sur $\mathcal{R}$ (car ce fait est bien connu), et dont le vecteur identité par l'addition est égal à son origine, notée $0^{|A|}$, c'est-à-dire le vecteur dont les $|A|$ coordonnées sont nulles.

De plus, les propositions suivantes sont vérifiées :
- tout sous-ensemble B de A induit l'espace vectoriel $(\mathbb{R}^{|B|}, +, \times)$ sur $\mathcal{R}$, qui est un sous-espace vectoriel de $\mathcal{L}$;
- tout sur-ensemble C de A induit l'espace vectoriel $(\mathbb{R}^{|C|}, +, \times)$ sur $\mathcal{R}$, pour lequel $\mathcal{L}$ est un sous-espace vectoriel ;
- A est une base de $\mathcal{L}$;
- $\mathcal{L}$ est de dimension $|A|$.

$\mathcal{L}$ « grandit » et « réduit » avec A.

Extension de chaînes

Comme cela a été fait pour le cas de lettres, nous étendons maintenant le concept de chaîne.

Définition 75 (Chaîne étendue)
Soit $n \in \mathbb{N}$. Une chaîne étendue (étiquetée) sur A, de taille n, est une fonction $\{1, \ldots, n\} \to \mathbb{R}^{|A|}$.

Soit X une chaîne étendue sur A de taille n. Nous utilisons le même vocabulaire pour caractériser X que celui que nous utilisons pour les chaînes : X est de *longueur* n, et $\{1, \ldots, n\}$ est l'ensemble des *positions* de X. De plus, notons $\mathbb{XS}(A)^n$ l'ensemble des chaînes étendues sur A de taille n.

Définition 76 (Extension limitée de chaîne)
Soit $n \in \mathbb{N}$. L'extension de chaîne sur A limitée à n est l'injection notée $\text{ext}(A)^n$ et définie comme suit :

$$\begin{aligned} \mathbb{S}(A)^{\leqslant n} &\to \mathbb{XS}(A)^n \\ X \to Y \,\Big|\, &\forall i \in \{1, \ldots, n\}, Y(i) = \begin{cases} \text{ext}(A)(X(i)) & \text{si } i \leqslant |X|, \\ 0^{|A|} & \text{sinon,} \end{cases} \end{aligned}$$

avec $\text{ext}(A)$ une extension de A.

Littéralement parlant, $Y(i)$ est la lettre étendue associée à $X(i)$ si $i \leqslant |X|$, et l'origine de l'alphabet étendu de A sinon. La j-ème coordonnée de $Y(i)$ est interprétée comme le degré (positif ou négatif) dans $Y(i)$ de (l'unique élément de) $(\text{ext}(A))^{-1}\left(1_j^A\right)$.

$\text{ext}(A)^n$ définit une indexation de $\mathbb{S}(A)^{\leqslant n}$ dans $\mathbb{XS}(A)^n$, c'est-à-dire une mise en correspondance bijective entre $\mathbb{S}(A)^{\leqslant n}$ et un sous-ensemble de cardinal $\left|\mathbb{S}(A)^{\leqslant n}\right|$ de chaînes étendues sur A. Pour faire simple, nous allons parler aussi bien d'une chaîne de $\mathbb{S}(A)^{\leqslant n}$ que de la chaîne étendue sur A qui lui est associée par $\text{ext}(A)^n$, et donc aussi bien de $\mathbb{S}(A)^n$ et $\mathbb{S}(A)^{\leqslant n}$ que de $\text{ext}(A)^n\left(\mathbb{S}(A)^n\right)$ et $\text{ext}(A)^n\left(\mathbb{S}(A)^{\leqslant n}\right)$, respectivement.

Nous pouvons maintenant définir un espace vectoriel sur $\mathcal{R}$ nous permettant de manipuler les chaînes de $\mathbb{S}(A)^{\leqslant n}$ dans son extension, et pour lequel $\mathbb{S}(A)^n$ est une base.

Représentation vectorielle de chaînes

Soit $n \in \mathbb{N}$. Nous définissons littéralement notre espace de chaînes $\mathcal{S}^n$ comme l'unique chaîne de longueur n sur l'alphabet de cardinal unitaire $\{\mathcal{L}\}$, en d'autres termes le produit cartésien de n copies de notre espace de lettres $\mathcal{L}$. Plus formellement, nous avons $\mathcal{S}^n = (\mathbb{XS}(A)^n, +, \times)$, avec $+$ et $\times$ les extensions classiques aux produits cartésiens des addition et scaling coordonnée-par-coordonnée des espaces vectoriels classiques sur $\mathbb{R}^m$, $m \in \mathbb{N} : \forall i \in \{1, \ldots, n\}, \forall X, Y \in \mathbb{XS}(A)^n, \forall r \in \mathbb{R}$:

$$(X + Y)(i) = X(i) + Y(i),$$
$$(r \times X)(i) = r \times X(i),$$

en se souvenant que $X(i)$ et $Y(i)$ sont des vecteurs de $\mathcal{L} = (\mathbb{R}^{|A|}, +, \times)$.

$\mathcal{S}^n$ n'est donc rien d'autre que le produit classique de copies d'un espace numérique classique. Il est inutile de prouver que $\mathcal{S}^n$ est vectoriel sur $\mathcal{R}$ (car ce fait est bien connu), et le vecteur identité par l'addition dans $\mathcal{S}^n$ est égal à la chaîne étendue sur A composée de n copies du vecteur identité par l'addition dans $\mathcal{L}$.

De plus, les propositions suivantes sont vérifiées :
- tout sous-ensemble B de A induit l'espace vectoriel $(\mathbb{XS}(B)^n, +, \times)$ sur $\mathcal{R}$, qui est un sous-espace vectoriel de $\mathcal{S}^n$;
- tout sur-ensemble C de A induit l'espace vectoriel $(\mathbb{XS}(C)^n, +, \times)$ sur $\mathcal{R}$, pour lequel $\mathcal{S}^n$ est un sous-espace vectoriel ;
- $\mathbb{S}(A)^n$ est une base de $\mathcal{S}^n$, car $\mathbb{S}(A)^n$ est égal au produit cartésien de n copies de A ;
- $\mathcal{S}^n$ est de dimension $|\mathbb{S}(A)^n| = |A|^n$.

$\mathcal{S}^n$ « grandit » et « réduit » avec A.

Considérations pratiques

Théoriquement, nous pouvons ne supposer aucune limite sur la longueur des chaînes que nous pouvons rencontrer dans une application spécifique. Nous aurions alors tout à fait pu définir un espace vectoriel de chaînes produit cartésien d'un nombre infini dénombrable de copies de notre espace de lettres. Mais cela est inutile, car il suffit, dans le cas précité, de garder à jour une variable mémorisant la taille maximale n d'une chaîne dans la séquence de données courante, et d'utiliser l'espace associé à l'extension de chaînes limitée à n.

Dans les autres cas, où nous avons connaissance de la taille maximale m d'une chaîne que nous pourrions être amené à considérer, $\mathcal{S}^n$ est approprié pour $n \geqslant m$.

6.6.3 Conclusion

Nous avons vu comment incorporer des ensembles des chaînes dans un espace vectoriel, nous permettant de manipuler des variables aléatoires représentant de telles structures via des combinaisons linéaires. Le concept abstrait de chaîne étendue est la clé, et il est possible de représenter le degré d'importance de chaque chaîne dans une chaîne étendue.

Cette approche est d'interprétation fortement probabiliste et pourrait aisément être étendue aux structures plus complexes dans le même esprit de simplicité.

Le problème est qu'il n'est certainement pas d'usage intéressant d'un point de vue applicatif et est de manière générale inadapté pour répondre aux attentes soulevées par le TCL. En effet, l'espace de lettres de base à celui de chaînes est un espace *orthonormal*, qui n'incorpore donc aucune notion ni information sur la dissimilarité entre lettres. Par conséquent, une simulation d'exécution du processus introduit par le TCL, c'est-à-dire une moyenne d'un grand nombre de

variables aléatoires de chaînes i.i.d., étendues pour être à valeurs dans notre espace, résulte, après normalisation des densités de probabilités, en la même distribution de chaînes que celle selon laquelle les variables additionnées étaient distribuées. Sans aucune « déformation » de notre espace induite par une dissimilarité, le degré de convergence vers une valeur limite est nul. Nous pensons que ce cas est spécial, et qu'il est simplement dû au fait qu'un espace orthonormal, le plus régulier des espaces possibles, implicite le fait que tous les éléments deux-à-deux différents de sa base sont de dissimilarité égale, et donc le fait que leur dissimilarité n'apporte globalement aucune information, enfin le fait que l'addition de causes respectant les critères du TCL n'a aucune conséquence notable.

Il serait donc intéressant d'incorporer cette notion de dissimilarité, disons une distance, dans un espace probabilisé, pour pouvoir pleinement, c'est-à-dire généralement, formaliser le TCL dans cet espace. Cet espace devrait donc être à la fois vectoriel et *métrique* pour être un candidat à la réalisation du TCL en son sein. Dans notre cas des chaînes, l'alphabet ne serait plus la base de l'espace de lettres, ce cas étant très spécial, et l'ensemble des chaînes ne serait également plus nécessairement une base à celui des chaînes étendues.

La finalité de ce travail pourrait être l'utilisation de cet espace pour la classification de chaînes via des algorithmes performant dans le traitement de données numériques multidimensionnelles (machine à vecteurs de support, réseau de neurones artificiel...).

6.7 Conclusion

Nous avons proposé dans ce chapitre le moyen de contrôler statistiquement la génération d'ensembles de chaînes discrètes par une distribution liée à une densité de probabilité normale, ainsi qu'une méthode d'estimation de l'écart-type d'une telle loi de probabilité à partir d'un échantillon supposé en être le résultat génératif. Cette méthode de recherche est prometteuse mais reste complexe car elle doit faire face à la combinatoire induite par le type chaîne. Notons que ce travail peut naturellement s'appliquer à toute autre loi de probabilité paramétrée par sa moyenne et son écart-type, ou autre paramètre à partir duquel on peut déduire l'écart-type, ce qui est le cas par exemple de la distribution de Laplace.

Une seconde réflexion fut portée sur la formalisation générale du théorème central limite, justification des suppositions de normalité des échantillons de données numériques traités dans de nombreuses applications de reconnaissance de formes, entre autres, et dont nous voudrions qu'il soit le critère à vérifier pour qu'une distribution puisse être qualifiée de normale dans son espace probabilisé.

Ces deux approches, opposées dans leur paradigme, local et analytique d'une part, et global et théorique d'autre part, soulèvent des questions de fond et promettent des perspectives de recherche plus concrètes dans l'objectif de traitement statistique de données structurées.

Chapitre 7

Expérimentations

Dans ce chapitre, nous présentons une série d'expérimentations de classification de données issues de l'environnement réel, de manière à évaluer les apports de notre approche pour la résolution de problèmes typiques à la reconnaissance de formes.

7.1 Classification de séquences d'ADN

Nous avons réalisé cette tâche sur deux bases de séquences, codées sous forme de chaînes sur l'alphabet $\{\mathrm{a}, \mathrm{c}, \mathrm{g}, \mathrm{t}\}$ des acides nucléiques constituant l'ADN. Ces bases sont accessibles sur le Web à l'adresse suivante : `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/molecular-biology/`.

7.1.1 Séquence promotrice

Présentation du problème

Une séquence promotrice est une partie de l'ADN se situant à proximité ou à l'intérieur d'un gène, et dont le rôle est de rentrer en contact avec une enzyme, l'ARN polymérase, dédiée à la traduction du code génétique en protéines constituantes des atomes.

Nous disposons d'une base de 106 séquences de même longueur, 57, étiquetées promotrice (53) ou non promotrice (53), et notre objectif est de mettre au point un classifieur permettant de décider si une séquence est promotrice.

Classifieur utilisé

Nous utilisons le classifieur de *maximisation de vraisemblance*, dont la construction peut se résumer de la manière suivante : soient $n \in \mathbb{N} \setminus \{0\}$; $\{1, \ldots, n\}$ l'ensemble des classes du problème ; et $\forall i \in \{1, \ldots, n\}$, S_i la séquence des données appartenant à la classe i :

1. $\forall i \in \{1, \ldots, n\}$, apprendre une distribution P_i à partir de S_i ;
2. un objet x de la classe inconnue est affecté à la classe pour laquelle sa probabilité est maximisée :

$$\arg\max \{i \in \{1, \ldots, n\}\} \, \mathrm{P}_i(x),$$

avec bien entendu un choix aléatoire à faire dans le cas où plusieurs classes maximisent cette probabilité (égalité).

Pour notre problème, nous avons $n = 2$, et il nous reste à spécifier comment apprendre P_1 et P_2 à partir des 53 chaînes respectivement concernées. Nous faisons le choix d'estimer une distribution gaussienne telle que présentée en chapitre 6. Cela nécessite que nous ayons préalablement fixé une fonction de coût pour opérations d'édition d'acides nucléiques. Ne bénéficiant d'aucune

connaissance experte relative à la dissimilarité entre ces acides pour ce problème, nous choisissons d'apprendre également cette fonction de coût à partir des données et de leur étiquette de classe (apprentissage supervisé). Comme toutes les chaînes sont de même longueur, nous faisons la simplification de considérer seulement les fonctions associant un coût infini à toute insertion ou suppression, de manière à annuler la vraisemblance de telles opérations, avec pour implication positive le fait que les paramètres de la distribution peuvent être estimés optimalement en temps polynomial (cf. section 6.5). La méthode d'apprentissage d'une fonction de coût de cette forme, décrite par l'algorithme 9, est très simple car basée sur un comptage : pour chaque lettre de l'alphabet considéré, nous comptons le nombre de fois où apparaît chaque lettre de l'alphabet à la même position dans une chaîne appartenant à la même classe. Et le coût d'une opération d'édition diminue avec l'augmentation du nombre de fois où l'on a compté l'apparition de sa lettre produite dans une chaîne sachant l'apparition de sa lettre consommée à la même position dans une chaîne appartenant à la même classe. Il peut arriver que la fonction retournée par l'algorithme ne respecte pas entièrement les conditions d'une fonction de coût (coût non nul d'une substitution non identique), mais, cela ne posant aucun problème en pratique, nous décidons de nous satisfaire de cette « approximation théorique ».

Données : Un alphabet $A = \{a_1, \ldots, a_{|A|}\}$, avec lequel λ est associé en qualité de lettre vide ; $n \in \mathbb{N} \setminus \{0\}$; $l \in \mathbb{N}$; $\forall i \in \{1, \ldots, n\}$, S_i une séquence de chaînes sur A de longueur l.
Résultat : Une fonction de coût sur A associant $+\infty$ à toute insertion ou suppression.
début
  pour $i \longleftarrow 1$ **à** $|A|$ **faire**
    pour $j \longleftarrow 1$ **à** $|A|$ **faire**
      $\text{compte}(a_i, a_j) \longleftarrow 0$;
    fin
  fin
  pour $i \longleftarrow 1$ **à** n **faire**
    pour $j \longleftarrow 1$ **à** $|S_i|$ **faire**
      pour $k \longleftarrow 1$ **à** $|S_i|$ **faire**
        pour $p \longleftarrow 1$ **à** l **faire**
          $\text{compte}(((S_i)_j)_p, ((S_i)_k)_p) \longleftarrow \text{compte}(((S_i)_j)_p, ((S_i)_k)_p) + 1$;
        fin
      fin
    fin
  fin
  pour $i \longleftarrow 1$ **à** $|A|$ **faire**
    $c(a_i \rightarrow \lambda) \longleftarrow +\infty$;
    $c(\lambda \rightarrow a_i) \longleftarrow +\infty$;
    pour $j \longleftarrow 1$ **à** $|A|$ **faire**
      si $\text{compte}(a_i, a_j) > \text{compte}(a_i, a_i)$ **alors**
        $c(a_i \rightarrow a_j) \longleftarrow 0$;
      sinon
        $c(a_i \rightarrow a_j) \longleftarrow \text{compte}(a_i, a_i) - \text{compte}(a_i, a_j)$;
      fin
    fin
  fin
  retourner c;
fin

Algorithme 9 : Apprentissage supervisé d'une fonction de coût interdisant les insertions et suppressions.

Résultat

Le volume des données étant faible (106 séquences), nous évaluons les performances de notre classifieur grâce à la méthode nommée *leave one out* (« en laisser un de côté ») dans la littérature internationale, qui peut se résumer de la manière suivante : soit S la séquence des données du problème :

1. pour chaque donnée S_i, $i \in \{1, \ldots, n\}$, construire le classifieur C_i avec pour données la séquence formée par S privée de S_i, et étiqueter S_i par C_i ;
2. l'estimation de la fiabilité du classifieur est donnée par le pourcentage de classification correcte à l'étape précédente :

$$\frac{c}{|S|},$$

avec c le nombre de classifications correctes à l'étape précédente.

Avec notre méthode, nous obtenons 89 classifications correctes sur un total de 106, soit une fiabilité estimée à environ 84 % sur ces données. Bien entendu, la fonction de coût étant apprise à partir des données, ce processus d'apprentissage est itéré avant la construction de chaque classifieur C_i du premier point de l'énumération ci-dessus, avec les mêmes données que celles à partir desquelles C_i est construit, c'est-à-dire en laissant de côté S_i. Le contraire n'aurait pas été équitable et le résultat du *leave one out* n'aurait donc pas été fiable.

Nous avons comparé ce résultat avec ceux obtenus, selon le même processus d'évaluation *leave one out*, par les deux autres classifieurs détaillés ci-dessous, souvent utilisés en reconnaissance de formes structurelle :

- le classifieur aux k plus proches voisins ;
- le classifieur à la plus proche médiane.

Pour le classifieur aux k plus proches voisins, nous avons essayé avec $k = 1$, 3, et 5, pour un résultat légèrement meilleur obtenu avec $k = 3$. Pour ce qui est du classifieur à la plus proche médiane, il s'agit simplement dans ce cas de calculer, pour chaque classe, la médiane (cf. section 3.1.1) des séquences appartenant à cette classe, et d'affecter une séquence à la même classe que celle de la médiane avec laquelle elle est le moins distante, avec bien entendu un choix aléatoire à faire dans le cas où plusieurs médianes minimisent cette distance (égalité). Dans tous les cas, pour les deux classifieurs cités ci-dessus, la distance choisie est la distance d'édition selon une fonction de coût apprise avec le même algorithme 9. Enfin, nous pouvons également comparer ces résultats avec ceux exposés dans la notice explicative fournie avec la base de données, et récapitulés dans le tableau 7.1.1. De plus amples détails sur ces expérimentations sont donnés dans l'article de Towell et al. [TSN90]. Pour ce qui est des 3 plus proches voisins avec une distance dite « simple », il s'agit simplement d'une distance entre chaînes de même longueur, égale au nombre de positions pour lesquelles les deux chaînes concernées diffèrent. Par exemple, *acc* et *agc* ont une distance simple égale à 1, car elles diffèrent en une seule position, la 2 ($c \neq g$).

Discussion

Des enseignements négatifs et positifs peuvent être extraits de cette expérimentation. D'une part, notre approche ne fournit pas de résultat aussi bon que les meilleurs résultats obtenus dans la littérature. Par contre, on peut souligner qu'ajouter un cadre probabiliste au cadre d'édition structurelle proposé par le concept de fonction de coût peut réduire considérablement les échecs des k plus proches voisins et de la plus proche médiane basés sur la distance d'édition. Notre approche, à la fois probabiliste et d'édition structurelle, nous a permis de nous situer dans l'ensemble des méthodes obtenant des résultats honorables, là où la distance d'édition échoue fortement, ce pour la même fonction de coût. Nous pensons que nos résultats pourraient être améliorés avec un algorithme d'apprentissage de fonction de coût plus sophistiqué, et éventuellement l'addition de la prise en compte d'une connaissance experte relative au domaine d'application particulier, ce qui est le cas des trois méthodes de classification qui obtiennent les meilleurs résultats pour ce problème.

Classifieur	**Résultat**
KBANN (système d'apprentissage automatique hybride)	96,22
BP (réseau de neurones artificiel)	92,45
O'Neill (technique ad hoc de la littérature biologique)	88,67
3 plus proches voisins, distance simple	87,73
Maximisation de la vraisemblance, **distribution gaussienne**, **fonction de coût apprise**	**83,96**
ID3 (arbre de décision)	82,07
Plus proche médiane, distance d'édition, fonction de coût apprise	57,54
3 plus proches voisins, distance d'édition, fonction de coût apprise	55,66

TABLE 7.1 – Évaluation (*leave one out*) de la fiabilité (%) de différents classifieurs pour le problème des séquences promotrices.

7.1.2 Épissage

Présentation du problème

Les points de jonction d'épissage sont des positions dans la séquence d'ADN à partir desquelles une partie de la séquence est soit conservée (partie codante) soit supprimée (partie non codante) pendant le processus de création de protéines chez certains organismes supérieurs. Les morceaux de la séquence conservés après épissage sont nommés « exon » et ceux qui ont été supprimés sont nommés « intron ».

Pour ce problème, nous disposons d'une base de 3190 séquences de même longueur, 60, et notre objectif est de mettre au point un classifieur permettant de décider si une séquence contient un point d'épissage intron/exon (la séquence correspond à la fin d'un intron suivi par le début d'un exon ; classe IE), un point d'épissage exon/intron (classe EI), ou ni l'un ni l'autre (la séquence est un morceau d'intron ou un morceau d'exon ; classe N). Les séquences de cette base sont bien entendu étiquetées par leur classe, et la base est composée de 767, 768, et 1655 séquences de la classe EI, IE, et N, respectivement.

Classifieur utilisé

Nous utilisons le même classifieur que celui utilisé pour l'expérience précédente (section 7.1.1), à savoir, pour rappel, le classifieur de maximisation de vraisemblance pour lequel les distributions des classes sont approximées par une distribution gaussienne (cf. chapitre 6), avec une fonction de coût apprise au moyen de l'algorithme 9.

Résultat

Nous évaluons les performances de notre classifieur grâce à la méthode nommée *cross validation* (validation croisée) dans la littérature internationale, qui peut se résumer de la manière suivante : soit S la séquence des données du problème, et $k \in \mathbb{N} \setminus \{0, 1\}$:

1. diviser S en k sous-séquences de tailles égales (si possible, sinon une de ces sous-séquences doit être de taille différente à celle des autres) ;

2. pour chaque sous-séquence T_i obtenue au point précédent ($i \in \{1, \ldots, k\}$), construire le classifieur C_i avec pour données la séquence formée par S privée de T_i, et étiqueter T_i par C_i ;
3. l'estimation de la fiabilité du classifieur est donnée par le pourcentage de classification correcte à l'étape précédente :

$$\frac{c}{|S|},$$

avec c le nombre de classifications correctes à l'étape précédente.

Le *leave one out* est en fait un cas spécial de la validation croisée, avec $k = |S|$.

Le tableau 7.1.2 montre le résultat obtenu avec notre méthode, ainsi que ceux obtenus par le classifieur aux 3 plus proches voisins et celui à la plus proche médiane, avec la distance d'édition selon une fonction de coût apprise avec le même algorithme 9. Ces chiffres peuvent être comparés avec ceux exposés dans la notice explicative fournie avec la base de données. Dans tous les cas, l'évaluation est faite avec pour données 1000 séquences d'ADN sélectionnées aléatoirement parmi les 3190 disponibles, et les pourcentages donnés pour une classe correspondent à la fiabilité obtenue par un classifieur pour l'étiquetage des données appartenant à cette classe, c'est-à-dire le pourcentage de données affectées à cette classe par le classifieur parmi les données appartenant réellement à cette classe. Pour ce qui est des deux classifieurs que nous avons évalués, les résultats présentés correspondent à une moyenne sur 10 expérimentations, avec à chaque fois une nouvelle sélection aléatoire des données. Par contre, pour ce qui est des résultats présentés dans la notice explicative, il n'est aucunement fait mention d'un tel procédé. Nous ne pouvons donc pas savoir si les chiffres présentés correspondent à une seule sélection des données, ou bien le meilleur, moins bon, ou résultat moyen parmi une série de classifications pour des sélections différentes... Le ou les auteur(s) de cette notice nous parle(nt) simplement d'expérimentations menées dans un laboratoire de son ou leur université. Pour ce qui est des 3 plus proches voisins avec une distance dite « simple », il s'agit simplement d'une distance entre chaînes de même longueur, égale au nombre de positions pour lesquelles les deux chaînes concernées diffèrent. Par exemple, *acc* et *agc* ont une distance simple égale à 1, car elles diffèrent en une seule position, la 2 ($c \neq g$).

Discussion

Globalement, nous pouvons tirer des conclusions relativement similaires à celles tirées lors de l'expérience précédente (section 7.1.1) : l'ajout de la modélisation probabiliste a encore une fois permis de limiter l'échec du cadre d'édition structurelle proposé par la fonction de coût : là où la distance d'édition obtient des résultats de très bas niveau, notre approche, avec la même fonction de coût, se situe de manière honorable, même si, encore une fois, elle devrait pouvoir être améliorée avec un algorithme d'apprentissage de fonction de coût plus sophistiqué, pour éventuellement rivaliser avec les approches dominatrices sur les bases de données de ce genre, notamment celles utilisant de la connaissance experte relative au domaine d'application.

En analysant ensuite plus précisément les résultats pour chaque classe, nous pouvons voir que notre approche a été compétitive pour la classification des séquences d'ADN possédant une structure spéciale, l'échec se situant au niveau de la classification de celles n'étant ni EI ni IE. Or, la classe N est en majorité absolue, et pourrait être composée de plusieurs sous-classes n'ayant structurellement rien en commun. Si tel est le cas, alors la distribution sous-jacente à la classe N pourrait certainement être bien mieux approximée au moyen d'une distribution *multi-modale*, et plus particulièrement d'un *mélange* de plusieurs distributions gaussiennes de chaînes mutuellement différentes, c'est-à-dire une combinaison linéaire convexe de distributions gaussiennes avec écarts-types mutuellement différents, en d'autres termes une distribution de la forme suivante :

$$\sum_{i=1}^{n} (w_i \times \mathrm{P}_i),$$

avec $n \in \mathbb{N} \setminus \{0\}$ et, $\forall i \in \{1, \ldots, n\}$, $w_i \in \mathbb{R}^+$ et P_i une distribution gaussienne de chaînes, tels que $\sum_{i=1}^{n} w_i = 1$ et, $\forall i, j \in \{1, \ldots, n\} | i < j$, $\mathrm{P}_i \neq \mathrm{P}_j$.

La classe N pourrait alors être plus fiablement classifiée à l'aide d'une étape initiale de *décomposition* de mélange, c'est-à-dire une estimation de n, des w_i et des P_i.

Classifieur	Résultat N	EI	IE
KBANN (système d'apprentissage automatique hybride)	95,38	92,44	91,53
BP (réseau de neurones artificiel)	94,71	94,26	89,25
PEBLS	93,14	91,82	92,45
Perceptron (neurone artificiel)	96,01	83,68	82,59
ID3 (arbre de décision)	91,16	89,42	86,01
COBWEB	88,20	84,96	90,54
Maximisation de la vraisemblance, distribution gaussienne, fonction de coût apprise	**68,65**	**86,05**	**91,80**
3 plus proches voisins, distance simple	68,89	88,35	90,91
3 plus proches voisins, distance d'édition, fonction de coût apprise,	67,27	85,07	43,56
Plus proche médiane, distance d'édition, fonction de coût apprise	61,05	68,28	84,75

TABLE 7.2 – Évaluation (10 *cross validation*) de la fiabilité (%) de différents classifieurs pour le problème d'épissage.

7.2 Classification d'images

7.2.1 Chiffres dessinés à la main

Présentation du problème

Une base de données d'images de chiffres décimaux (de 0 à 9) a été construite à partir de leurs dessins par 44 personnes avec un stylet sur un écran d'ordinateur tactile. De chaque dessin est ensuite extrait une séquence de points en coordonnées (x, y) représentant le passage du stylet sur l'écran, puis, après divers processus de discrétisation et de normalisation, il en résulte une séquence de 8 couples (x, y) de coordonnées entières sur un intervalle allant de 0 à 100.

Nous disposons au final d'une base de 10992 chaînes de même longueur, 8, sur l'alphabet A des couples d'entiers allant de $(0, 0)$ à $(100, 100)$ (alphabet de taille $101 \times 101 = 10201$), et étiquetées par un chiffre allant de 0 à 9 (les effectifs des classes sont à peu près similaires, de l'ordre de 1100 par classe). Notre objectif est de mettre au point un classifieur permettant de décider à quelle chiffre décimal correspond l'image à partir de laquelle a été extraite une chaîne.

La base est accessible sur le Web à l'adresse suivante : `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/pendigits/`.

Classifieur utilisé

Comme pour les expériences avec les séquences d'ADN (section 7.1), nous utilisons le classifieur de maximisation de vraisemblance pour lequel les distributions des classes sont approximées par une distribution gaussienne (cf. chapitre 6). La différence est que la fonction de coût c n'a cette fois pas besoin d'être apprise, puisqu'elle s'impose naturellement comme étant la restriction à A de la distance euclidienne sur $\mathbb{R}^2$ (A étant un sous-ensemble de $\mathbb{R}^2$) : $\forall (x_1, y_1), (x_2, y_2) \in A$:

$$c\left((x_1, y_1) \to (x_2, y_2)\right) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

Classifieur	**Résultat**
3 plus proches voisins, distance d'édition, fonction de coût « euclidienne »	97,51
Maximisation de la vraisemblance, **distribution gaussienne,** **fonction de coût « euclidienne »**	**78,47**
Plus proche médiane, distance d'édition, fonction de coût « euclidienne »	76,92

TABLE 7.3 – Évaluation (sur échantillon de test) de la fiabilité (%) de différents classifieurs pour le problème des chiffres dessinés à la main.

Comme toutes les chaînes sont de même longueur, nous faisons la simplification d'associer un coût infini à toute insertion ou suppression, de manière à annuler la vraisemblance de telles opérations : $\forall (x, y) \in A$:

$$c\left((x, y) \to \lambda\right) = c\left(\lambda \to (x, y)\right) = +\infty,$$

avec λ la lettre vide associée à A.

Résultat

Le nombre de chaînes étant élevé, la base de donnée est divisée en deux parties : un *échantillon d'apprentissage* et un *échantillon de test*. L'échantillon d'apprentissage nous sert à construire le classifieur sur ses données, la fiabilité du classifieur étant ensuite estimée par le pourcentage de classification correcte sur les données de l'échantillon de test. La base est divisée de manière à ce que chaque classe ait environ 2 tiers de ses données pour la phase d'apprentissage (de l'ordre de 750 séquences par classe en moyenne) et le tiers restant (de l'ordre de 350) pour la phase de test.

Nous obtenons un peu plus de 78 % de classification correcte, alors que le classifieur aux 3 plus proches voisins avec la distance d'édition selon la même fonction de coût obtient un peu plus de 97 % (tableau 7.2.1).

Discussion

Ces résultats ne sont pas surprenants, et en définitive pas si négatifs que l'on pourrait le croire. En effet, nous pouvons d'une part affirmer que cette expérimentation est typiquement adaptée au classifieur aux plus proches voisins, ce pour deux raisons qui cumulent leur effet sur ce problème :

1. le volume des données est grand alors que les données aberrantes sont en faible proportion (les personnes avaient la possibilité de recommencer leurs dessins s'ils n'en étaient pas content ou avaient fait une erreur). Or il est bien connu que le classifieur aux plus proches voisins, sous condition d'une distance bien choisie, doit en théorie converger vers l'optimalité (classification parfaite) avec l'augmentation du nombre de données ;
2. il n'y a rien de plus naturel et de plus pertinent pour ce problème que la distance euclidienne comme fonction de coût, puisque les lettres de l'alphabet représentent des points en coordonnées (x, y), donc des éléments d'un espace numérique.

D'autre part, nous pouvons voir que notre approche s'en sort tout de même de manière honorable pour ce problème à 10 classes, puisqu'une classification aléatoire (choisir au hasard la classe à affecter aux éléments de l'échantillon de test, sans considération aucune d'informations pouvant être extraites de l'échantillon d'apprentissage) devrait en théorie n'obtenir que 10 % de fiabilité. On voit que là où les probabilités ne sont pas un paradigme pertinent (pourquoi modéliser une distribution par classe puisque les données sont si homogènes ?), notre approche n'est pas aussi médiocre que peut l'être l'approche distance d'édition dans le cas inverse, tel que le montrent les expérimentations sur séquences d'ADN de la section 7.1.

7.3 Conclusion

Nous avons vu dans ce chapitre une série d'expérimentations sur données issues de l'environnement réel, visant à évaluer les apports de notre approche pour la classification de formes.

Nous pouvons dire qu'il est possible d'améliorer les résultats obtenus par l'approche distance d'édition, sans se détacher du cadre d'édition structurelle induit par le concept de fonction de coût, pour les bases de données où cette approche échoue ou ne donne pas de résultat relativement satisfaisant. Dans ce cas, notre approche probabiliste peut s'avérer utile car nous avons vu que, même en apprenant une fonction de coût de manière très et même trop simpliste, notre approche peut obtenir des résultats honorables, là où l'approche distance d'édition, notamment le classifieur à la plus proche médiane selon une telle distance, peut échouer fortement. Le cadre probabiliste augmentant le cadre d'édition permet à notre approche de se détacher un peu de la fonction de coût, c'est-à-dire en limiter les effets, et donc en être moins dépendante au niveau de la classification. Le point négatif de notre méthode est donc déduit de manière similaire à celle selon laquelle nous avons déduit son point positif, à savoir que se détacher de la fonction de coût peut parfois s'avérer non pertinent, dans le cas où cette fonction est quant à elle très appropriée au problème attaqué, comme nous l'avons vu avec l'expérience sur la classification d'images de chiffres dessinés à la main (section 7.2.1). Il paraît judicieux d'affirmer que si la fonction de coût est naturellement adaptée au problème, alors en limiter les effets est un mauvais choix, et donc choisir d'utiliser les probabilités (notre classifieur) ou les statistiques (le classifieur à la plus proche médiane) n'est pas le plus pertinent.

Chapitre 8

Conclusion

8.1 Conclusions

Dans cette thèse, nous voulions contribuer à un rapprochement des approches statistiques et structurelles de la reconnaissance de formes.

Nous avons fait le choix de nous attarder sur le traitement probabiliste d'ensembles de structures discrètes, et nous avons plus particulièrement proposé la traduction, à leurs espaces, de critères permettant de définir des notions d'uniformité et de normalité de distributions. Notre approche se démarque de celles jusqu'à présent développées dans la littérature pour s'attaquer à l'atteinte du même objectif, à savoir, entre autres, les méthodes basées sur des statistiques selon une distance, plus particulièrement la distance d'édition, ou alors les méthodes tentant de représenter des distributions au moyen de modèles à états. Les méthodes statistiques selon la distance d'édition ne permettent pas de se détacher des inconvénients calculatoires soulevés lors de traitements d'ensembles de structures, alors que les machines à états ne supposent aucun a priori statistique quant à la distribution qu'elles modélisent ou approximent (dépendant de la puissance modélisatoire de la machine considérée), et sont donc utilisables de manière générique, c'est-à-dire quand on ne dispose d'aucune connaissance a priori sur la distribution des données du problème que l'on tente de résoudre. Une approche intermédiaire, consistant à transformer les structures en vecteur dans un espace numérique, s'avère particulièrement prometteuse et pourrait concentrer les atouts de plusieurs paradigmes tout en se détachant de leurs faiblesses. Nous avons cependant vu que les méthodes proposées jusqu'à présent dans cette philosophie, même si elles ont indéniablement fait preuve d'une certaine performance pour certaines tâches expérimentales, restent peu satisfaisantes d'un point de vue théorique, car quelques zones d'ombres particulièrement gênantes persistent, comme par exemple le manque de cohérence vis-à-vis de l'espace initial des structures. Nous avons d'ailleurs proposé une réflexion sur le sujet et sur les éventuels apports d'une recherche poussée dans cette direction, notamment en relation avec le besoin d'une généralisation du théorème central limite.

D'un point de vue expérimental, cette thèse a permit de tirer certaines conclusions intéressantes. Nous avons montré que notre approche, même si elle est encore immature et pas indiscutablement performante pour des applications réelles, s'inscrit dans un cadre de recherche qui peut porter ses fruits. Nous avons notamment montré que notre approche de distribution basée sur la loi normale, mixant le cadre d'édition structurelle proposé par le concept de fonction de coût et le cadre probabiliste, peut s'avérer plus performante en termes de classification que l'approche statistique purement basée sur le cadre structurel, dans lequel s'inscrit par exemple le classifieur à la plus proche médiane, très utilisé en reconnaissance de formes structurelle. Ce constat est exacerbé lorsque la fonction de coût utilisée n'est pas très naturelle, le classifieur à la plus proche médiane étant trop « rigidement » lié à cette fonction, ce qui n'est pas le cas de notre classifieur, « assoupli » par les probabilités.

8.2 Perspectives

Tout au long de cette thèse, nous avons proposé quelques perspectives de recherche concernant certains travaux potentiellement intéressants, et nous allons de suite en résumer les principales.

Il est indispensable de mettre au point de nouveaux tests statistiques permettant de décider si la distribution sous-jacente à une séquence de structures peut être fiablement approximée au moyen d'une distribution particulière, ou alors des tests permettant d'estimer l'« écart » probabiliste entre ces deux distributions. De tels tests pourraient par exemple confirmer ou infirmer l'hypothèse selon laquelle une séquence est le résultat d'une distribution gaussienne selon notre définition : après avoir estimé les paramètres de la distribution sous cette hypothèse de normalité, il resterait à tester la distribution, et le résultat de ce test confirmerait ou infirmerait en même temps l'hypothèse de normalité de la séquence.

Les expérimentations ont également mis en avant la nécessité de considérer des distributions multi-modales de structures, avec leur algorithme de décomposition de mélange associé. Là encore pourrait être profitable l'utilisation de nouveaux tests adaptés pour le cas de mélanges.

Un espace permettant de manipuler des structures comme de véritables vecteurs devrait nous permettre de traduire aux structures certains résultats de la théorie statistique numérique, comme le théorème central limite, et devrait de plus faire office de pont entre le cadre structurel et le cadre probabiliste pour la classification de structures au moyen d'algorithmes performants issus de la littérature statistique.

Annexe A

Éléments de théorie des ensembles

Nous supposons que le lecteur est informé des concepts d'*ensemble* et de *séquence*, et notons :
- $|E|$ le cardinal d'un ensemble E ;
- $|S|$ la longueur d'une séquence S ;
- S_i le i-ème élément de S, avec $i \in \{1, \ldots, |S|\}$ si $|S|$ est fini, et $i \in \mathbb{N} \setminus \{0\}$ sinon.

De plus, nous utilisons les termes « tuple » et « n-uple » comme synonymes des termes « séquence » et « séquence de longueur n », respectivement ($n \in \mathbb{N}$) ; si $n = 2$ (resp. 3 ; 4...), alors il s'agit d'un couple (resp. triplet ; quadruplet...).

A.1 Ensemble puissance

Définition 77 (Ensemble puissance)
Soit S un ensemble. L'ensemble puissance de S, noté 2^S, est l'ensemble des sous-ensembles de S :

$$2^S = \{E | E \subseteq S\}.$$

Exemple
Soit $S = \{1, 2, 3\}$. Nous avons $2^S = \{\{\}, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, S\}$.

Remarque
- Quel que soit S, 2^S n'est pas vide, car il contient $\{\}$ et S ;
- $|2^S| = 2^{|S|}$.

A.2 Partition

Définition 78 (Partition)
Soit S un ensemble. Une partition P de S est un ensembles de sous-ensembles de S, non vides, disjoints, et dont l'union est égale à S :
- $P \subseteq 2^S$;
- $\forall E \in P, E \neq \{\}$;
- $\forall E, F \in P | E \neq F, E \cap F = \{\}$;
-
$$\bigcup_{E \in P} E = S.$$

Exemple
Soit $S = \{1, 2, 3\}$. $\{\{1\}, \{2, 3\}\}$ est une partition de S.

A.3 Produit cartésien

Définition 79 (Produit cartésien)
Soient $n \in \mathbb{N}$, et $E_1, \ldots, E_n$ n ensembles. Le produit cartésien de E_1 par E_2 par... par E_n, noté $E_1 \times \cdots \times E_n$, est l'ensemble des n-uples dont la première composante est élément de E_1, la seconde est élément de E_2, ..., la n-ième est élément de E_n :

$$E_1 \times \cdots \times E_n = \{(x_1, \ldots, x_n) | x_i \in E_i \ \forall i \in \{1, \ldots, n\}\}.$$

Le produit cartésien de n copies d'un ensemble E est noté E^n.

Exemple
Soient $E = \{1, 2, 3\}$, et $F = \{a, b, c\}$. Nous avons $E \times F = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c), (3, a), (3, b), (3, c)\}$.

Remarque
$|E_1 \times \cdots \times E_n| = |E_1| \times \cdots \times |E_n|$.

A.4 Fonction

Définition 80 (Fonction)
Une fonction, ou application, f, d'un ensemble E nommé ensemble de départ, et à valeurs dans un ensemble F nommé ensemble d'arrivée, est un sous-ensemble de $E \times F$, noté $f \colon E \to F$, tel que tout élément de E est associé à exactement 1 élément de F :

$$\forall x \in E, |\{y \in F | (x, y) \in f\}| = 1.$$

Notons identité(S) la fonction $S \to S$ associant x à tout élément x de l'ensemble S, et vide(S) la fonction $\{\} \to S$ égale à l'ensemble vide.

Exemple
Soient $E = \{1, 2, 3\}$, et $F = \{a, b, c\}$. $f = \{(1, a), (2, b), (3, b)\}$ est une fonction $E \to F$.

Définition 81 (Valeur d'une fonction, image par une fonction)
Soit f une fonction $E \to F$, $x \in E$, et $X \subseteq E$. La valeur de f en x, notée $f(x)$, est l'élément de F associé à x par f :

$$f(x) = y \iff (x, y) \in f.$$

L'image de X par f, notée $f(X)$, est l'ensemble des valeurs de f en les éléments de X :

$$f(X) = \{f(x) | x \in X\}.$$

Par abus de langage, nous pouvons aussi bien parler d'image de x que de valeur en x, et de valeur en X que d'image de X.

Exemple
Soient $E = \{1, 2, 3\}$, $F = \{a, b, c\}$, et $f = \{(1, a), (2, b), (3, b)\}$. Nous avons $f(1) = a$, $f(2) = b$, $f(3) = b$, et $f(E) = \{a, b\}$.

Définition 82 (Image réciproque par une fonction)
Soit f une fonction $E \to F$, $y \in F$, et $Y \subseteq F$. L'image réciproque de y par f, notée $f^{-1}(y)$, est l'ensemble des éléments de X en lesquels la valeur de f est y :

$$f^{-1}(y) = \{x \in E | f(x) = y\}.$$

L'image réciproque de Y par f, notée $f^{-1}(Y)$, est l'union des images réciproques des éléments de Y par f :

$$f^{-1}(Y) = \bigcup_{y \in Y} f^{-1}(y).$$

L'image réciproque par f de y (resp. Y) est également nommée ensemble des *antécédents* de y (resp. Y).

Exemple
Soient $E = \{1, 2, 3\}$, $F = \{a, b, c\}$, et $f = \{(1, a), (2, b), (3, b)\}$. Nous avons $f^{-1}(a) = \{1\}$, $f^{-1}(b) = \{2, 3\}$, $f^{-1}(c) = \{\}$, et $f^{-1}(F) = f^{-1}(\{a, b\}) = E$.

Définition 83 (Fonction injective, surjective, bijective)
Soit f une fonction $E \to F$. f est dite injective (resp. surjective) ssi tout élément de F possède au plus (resp. au moins) 1 antécédent par f :

$$\forall y \in Y, |f^{-1}(y)| \leqslant 1 \text{ (resp. } \geqslant 1).$$

Enfin, f est dite bijective ssi elle est injective et surjective :

$$\forall y \in Y, |f^{-1}(y)| = 1.$$

Une fonction injective (resp. surjective ; bijective) est également nommée *injection* (resp. *surjection* ; *bijection*). Dès lors, une bijection est aussi bien une injection surjective qu'une surjection injective.

Exemple
Soient $E = \{1, 2, 3\}$, $F = \{a, b, c\}$, et $f = \{(1, a), (2, b), (3, b)\}$. f est n'est ni injective, car $|f^{-1}(b)| = 2$; ni surjective, car $|f^{-1}(c)| = 0$; et donc ni bijective. Par contre, si F avait été limité à $\{a, b\}$, alors F aurait été surjective, mais toujours pas injective.

Remarque
Si f est injective (resp. surjective ; bijective), alors $|E| \leqslant |F|$ (resp. $\geqslant |F|$; $= |F|$).

Définition 84 (Opération binaire)
Une opération, ou opérateur, binaire, est une fonction dont l'ensemble de départ est le produit cartésien de deux ensembles.

Dans ce cas spécifique, nous préférons souvent opter pour une notation infixée.

Exemple
L'addition entière bien connue, $+$, est une opération binaire $\mathbb{N}\times\mathbb{N} \to \mathbb{N}$. Nous avons $+(1, 1) =_{\text{notation}} 1 + 1 = 2\ldots$

A.5 Dénombrabilité

Définition 85 (Équipotence)
Deux ensembles E et F sont dits équipotents ssi il existe une bijection $E \to F$.

Exemple
Soient $E = \{1, 2, 3\}$, et $F = \{a, b, c\}$. La bijection $(f : E \to F) = \{(1, a), (2, b), (3, c)\}$ assure que E et F sont équipotents.

Remarque
E et F sont équipotents ssi $|E| = |F|$.

Définition 86 (Ensemble dénombrable)
Un ensemble S est dit dénombrable ssi il est équipotent à un sous-ensemble de $\mathbb{N}$.

Exemple
$\{a, b, c\}$ est (fini) dénombrable, car équipotent à $\{1, 2, 3\}$. L'ensemble $2\mathbb{N}$ des entiers naturels pairs est (infini) dénombrable car équipotent à lui-même par la fonction identité$(2\mathbb{N})$.

Annexe B

Éléments de combinatoire

La combinatoire est l'étude des divers types de groupements qu'il est possible de faire à partir d'ensembles finis. La fonction nommée *factorielle* apparaît de manière récurrente dans le dénombrement de tels groupements.

B.1 Factorielle

Définition 87 (Factorielle)
La factorielle est la fonction $\mathbb{N} \to \mathbb{N}$ définie comme suit : $\forall n \in \mathbb{N}$:

$$\text{factorielle}(n) =_{\text{notation}} n! = \prod_{i=1}^{n} i.$$

Exemple
$0! = 1$ car, par convention, le résultat d'une multiplication vide est 1, son élément identité ; $3! = 6 \ldots$

B.2 Arrangement

Définition 88 (Arrangement)
Soient S un ensemble fini, et $k \in \{0, \ldots, |S|\}$. Un arrangement de taille k de S est une séquence de k éléments distincts de S.

Exemple
(a, b) est un arrangement de taille 2 de $\{a, b, c\}$.

Proposition 12 (Dénombrement des arrangements)
Soient $n \in \mathbb{N}$, et $k \in \{0, \ldots, n\}$. Le nombre d'arrangements de taille k d'un ensemble de cardinal n est donné par la formule suivante :

$$\mathrm{A}_n^k = \frac{n!}{(n-k)!} = \prod_{i=0}^{k-1} (n-i).$$

Par convention, si k ou n est donné sous forme d'entier négatif (c'est-à-dire élément de $\mathbb{Z}^-$), ou encore si $k > n$, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Exemple
Le nombre d'arrangements de taille 2 de $\{a, b, c\}$ est égal à 6.

Définition 89 (R-arrangement)
Soient S un ensemble fini, et $k \in \mathbb{N}$. Un r-arrangement, ou arrangement avec répétition, de taille k de S, est une séquence de k éléments de S.

Exemple
(a, b, a) est un r-arrangement de taille 3 de $\{a, b, c\}$.

Proposition 13 (Dénombrement des r-arrangements)
Soient $n, k \in \mathbb{N}$. Le nombre de r-arrangements de taille k d'un ensemble de cardinal n est donné par la formule suivante :

$$\text{r-A}_n^k = n^k.$$

Par convention, si k ou n est donné sous forme d'entier négatif, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Exemple
Le nombre de r-arrangements de taille 3 de $\{a, b, c\}$ est égal à 27.

B.3 Combinaison

Définition 90 (Combinaison)
Soient S un ensemble fini, et $k \in \{0, \ldots, |S|\}$. Une combinaison de taille k de S est un sous-ensemble de cardinal k de S.

Exemple
$\{a, b\}$ est une combinaison de taille 2 de $\{a, b, c\}$.

Proposition 14 (Dénombrement des combinaisons)
Soient $n \in \mathbb{N}$, et $k \in \{0, \ldots, n\}$. Le nombre de combinaisons de taille k d'un ensemble de cardinal n est donné par la formule suivante :

$$\text{C}_n^k =_{\text{notation}} \binom{n}{k} = \frac{n!}{k! \times (n-k)!} = \frac{\text{A}_n^k}{k!}.$$

Par convention, si k ou n est donné sous forme d'entier négatif, ou encore si $k > n$, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Exemple
Le nombre de combinaisons de taille 2 de $\{a, b, c\}$ est égal à 3.

Les nombres de combinaisons sont également nommés *coefficients binomiaux*, en raison du résultat suivant :

Théorème 2 (Formule du binôme)
$\forall x, y, n \in \mathbb{N}$:

$$(x+y)^n = \sum_{k=0}^{n} \left(\text{C}_n^k \times x^{n-k} \times y^k \right).$$

Définition 91 (R-combinaison)
Soient S un ensemble fini, et $k \in \mathbb{N}$. Une r-combinaison, ou combinaison avec répétition, de taille k de S, est une fonction $f \colon S \to \mathbb{N}$ telle que :

$$\sum_{x \in S} f(x) = k.$$

Exemple
$\{(a, 2), (b, 1), (c, 0)\}$ est une r-combinaison de taille 3 de $\{a, b, c\}$.

Proposition 15 (Dénombrement des r-combinaisons)
Soient $n, k \in \mathbb{N}$. Le nombre de r-combinaisons de taille k d'un ensemble de cardinal n est donné par la formule suivante :

$$\text{r-C}_n^k = \frac{(n+k-1)!}{k! \times (n-1)!} = \text{C}_{n+k-1}^k \cdot$$

Par convention, si k ou n est donné sous forme d'entier négatif, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Exemple
Le nombre de r-combinaisons de taille 3 de $\{a, b, c\}$ est égal à 10.

B.4 Permutation

Les permutations peuvent être définis comme des arrangements spéciaux.

Définition 92 (Permutation)
Soit S un ensemble fini. Une permutation de S est une séquence de $|S|$ éléments distincts de S.

En d'autres termes, il s'agit d'un arrangement de taille $|S|$ de S.

Exemple
(a, c, b) est une permutation de $\{a, b, c\}$.

La proposition suivante découle de la proposition 12 :

Proposition 16 (Dénombrement des permutations)
Soit $n \in \mathbb{N}$. Le nombre de permutations d'un ensemble de cardinal n est donné par la formule suivante :

$$\mathrm{P}(n) = n!.$$

Par convention, si n est donné sous forme d'entier négatif, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Exemple
Le nombre de permutations de $\{a, b, c\}$ est égal à 6.

Définition 93 (R-permutation)
Soient $S = \{x_1, \ldots, x_k\}$ un ensemble fini, et $n_1, \ldots, n_k \in \mathbb{N}$. Une r-permutation, ou permutation avec répétition, de S, avec n_i répétitions de chaque élément x_i ($i \in \{1, \ldots, k\}$), est une séquence composée de n_i copies de chaque élément x_i ($i \in \{1, \ldots, k\}$).

En d'autres termes, il s'agit d'un r-arrangement de taille $\left(\sum_{i=1}^{k} n_i\right)$ de S composé de n_i copies de chaque élément x_i ($i \in \{1, \ldots, k\}$).

Exemple
(a, c, c, b) est une r-permutation de $\{a, b, c\}$, avec 1 répétition de a, 1 répétition de b, et 2 répétitions de c.

Proposition 17 (Dénombrement des r-permutations)
Soient $S = \{x_1, \ldots, x_k\}$ un ensemble fini, et $n_1, \ldots, n_k \in \mathbb{N}$. Le nombre de r-permutations de S, avec n_i répétitions de chaque élément x_i ($i \in \{1, \ldots, k\}$), est donné par la formule suivante :

$$\text{r-P}(n_1, \ldots, n_k) = \frac{\left(\sum_{i=1}^{k} n_i\right)!}{\prod_{i=1}^{k} n_i!} = \prod_{i=1}^{k} \binom{\sum_{j=i}^{k} n_j}{n_i}.$$

Par convention, si au moins 1 des n_i ($i \in \{1, \ldots, k\}$) est donné sous forme d'entier négatif, alors la notation ci-dessus est acceptée, et ce nombre est égal à 0.

Il est possible que nous soyons amenés dans certaines situations à opter pour une notation légèrement différente de celle exposée ci-dessus, par souci de simplicité. Considérons par exemple la donnée d'un ensemble S et d'un objet o qui n'est pas élément de S, et supposons que nous souhaitions désigner un nombre de r-permutations de $S \cup \{o\}$, dont le nombre de répétitions de chaque élément de S est donné par une fonction $f\colon S \to \mathbb{N}$, et avec m répétitions de o. Il serait alors plus judicieux d'utiliser la notation r-P$(f(x) \forall x \in S, m)$, ou simplement r-P$(f, m)$ s'il n'y a aucune ambiguïté ou confusion possible.

Exemple
Le nombre de r-permutations de $\{a, b, c\}$, avec 1 répétition de a, 1 répétition de b, et 2 répétitions de c, est égal à 12.

Annexe C

Éléments d'algèbre

C.1 Distance

Définition 94 (Distance)
Soit S un ensemble. Une distance sur S est une fonction $d\colon S \times S \to \mathbb{R}^+$ satisfaisant l'ensemble des conditions suivantes : $\forall x, y, z \in S$:

- séparation : $d(x, y) = 0 \iff x = y$;
- symétrie : $d(x, y) = d(y, x)$;
- inégalité triangulaire : $d(x, y) \leqslant d(x, z) + d(z, y)$.

Exemple
Soient $n \in \mathbb{N}, p \in \mathbb{N} \setminus \{0\}$. La p-distance d_p sur $\mathbb{R}^n$ est définie comme suit : $\forall x = (x_1, \ldots, x_n), y = (y_1, \ldots, y_n) \in \mathbb{R}^n$:

$$d_p(x, y) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p}.$$

Cette distance est également nommée *distance de Minkowski* d'ordre p, et d_1 (resp. d_2) est plus connue sous le nom de *distance de Manhattan* (resp. *distance euclidienne*).

Définition 95 (Espace métrique)
Un espace métrique est un couple (S, d), avec S un ensemble, et d une distance sur S.

C.2 Vecteur

Un espace vectoriel est une structure algébrique permettant d'effectuer des *combinaisons linéaires*. Un tel espace est défini sur un corps commutatif.

Définition 96 (Corps commutatif)
Un corps commutatif, ou champ, est un triplet $(S, \text{add}, \text{mult})$, avec S un ensemble, add une opération binaire $S \times S \to S$ nommée addition, et mult une opération binaire $S \times S \to S$ nommée multiplication, satisfaisant l'ensemble des conditions suivantes : $\forall x, y, z \in S$:

- add et mult ont leur propre élément identité : $\exists \text{zero}, \text{un} \in S | (\text{zero} \neq \text{un}) \wedge (x \,\text{add}\, \text{zero} = x \,\text{mult}\, \text{un} = x)$;
- add est inversible : $\exists (-x) \in S | x \,\text{add}(-x) = \text{zero}$;
- mult est $(\setminus\{\text{zero}\})$-inversible : $x \neq \text{zero} \implies \exists x^{-1} \in S | x \,\text{mult}\, x^{-1} = \text{un}$;
- add et mult sont associatives : $[x \,\text{add}(y \,\text{add}\, z) = (x \,\text{add}\, y) \,\text{add}\, z] \wedge [x \,\text{mult}(y \,\text{mult}\, z) = (x \,\text{mult}\, y) \,\text{mult}\, z]$;
- add et mult sont commutatives : $(x \,\text{add}\, y = y \,\text{add}\, x) \wedge (x \,\text{mult}\, y = y \,\text{mult}\, x)$;
- mult est distributive sur add : $x \,\text{mult}(y \,\text{add}\, z) = (x \,\text{mult}\, y) \,\text{add}(x \,\text{mult}\, z)$.

Exemple
Le champ bien connu $(\mathbb{R}, +, \times)$, avec zero $= 0$ et un $= 1$.

Définition 97 (Espace vectoriel)
Soit $\mathcal{F} = (S, \text{add}, \text{mult})$ un champ, avec zero (resp. un) l'identité de add (resp. mult). Un espace vectoriel sur $\mathcal{F}$ est un triplet $\mathcal{V} = (V, \text{ADD}, \text{SCALE})$, avec V un ensemble, ADD une opération binaire $V \times V \to V$ nommée addition, et SCALE une opération binaire $S \times V \to V$ nommée scaling, satisfaisant l'ensemble des conditions suivantes : $\forall u, v, w \in V, \forall a, b \in S$:
- ADD a un élément identité : $\exists\, \text{ZERO} \in V | v \,\text{ADD}\, \text{ZERO} = v$;
- ADD est inversible : $\exists (-v) \in V | v \,\text{ADD}(-v) = \text{ZERO}$;
- ADD est associative : $u \,\text{ADD}(v \,\text{ADD}\, w) = (u \,\text{ADD}\, v) \,\text{ADD}\, w$;
- ADD est commutative : $v \,\text{ADD}\, w = w \,\text{ADD}\, v$;
- un est l'élément de S identité de SCALE : $\text{un} \,\text{SCALE}\, v = v$;
- SCALE est distributive sur ADD : $a \,\text{SCALE}(v \,\text{ADD}\, w) = (a \,\text{SCALE}\, v) \,\text{ADD}(a \,\text{SCALE}\, w)$;
- SCALE est ADD-distributive sur add : $(a \,\text{add}\, b) \,\text{SCALE}\, v = (a \,\text{SCALE}\, v) \,\text{ADD}(b \,\text{SCALE}\, w)$;
- SCALE est compatible avec mult : $a \,\text{SCALE}(b \,\text{SCALE}\, v) = (a \,\text{mult}\, b) \,\text{SCALE}\, v$.

Dans ce contexte, les éléments de S sont nommés *scalaires*, et ceux de V *vecteurs*. En outre, nous parlons d'*addition scalaire* et *addition vectorielle* pour départager respectivement l'addition dans $\mathcal{F}$ (add), et celle dans $\mathcal{V}$ (ADD). *Scaling* est un anglicisme, plus léger à utiliser que les termes « mise à l'échelle », « multiplication scalaire-vecteur », ou encore « agrandissement-réduction ».

Exemple
Soit $\mathcal{F} = (\mathbb{R}, +, \times)$. $\mathcal{F}$ est un espace vectoriel sur $\mathcal{F}$. De même, $\forall n \in \mathbb{N}$, $\mathcal{F}^n = (\mathbb{R}^n, +^n, \times^n)$ est un espace vectoriel sur $\mathcal{F}$, avec $\mathbb{R}^n$ l'ensemble des n-uples à composantes réelles, $+^n$ l'addition composante par composante, et $\times^n$ la multiplication réel-(n-uple réel) composante par composante.

Pour en finir avec l'exemple précédent, il est à noter que $\mathcal{F}^n$ est traditionnellement considéré comme l'extension naturelle de $\mathcal{F}$ à n composantes, et que par conséquent $+^n$ est toujours noté $+$, et $\times^n$ toujours noté $\times$, voire $\cdot$.

Remarque
Tout champ définit un espace vectoriel sur lui-même ($V = S$, ADD = add, et SCALE = mult).

Définition 98 (Sous-espace vectoriel)
Soient $\mathcal{F} = (S, \text{add}, \text{mult})$ un champ, $\mathcal{V} = (V, \text{ADD}, \text{SCALE})$ un espace vectoriel sur $\mathcal{F}$, et $W \subseteq V$. $(W, \text{ADD}, \text{SCALE})$ est un sous-espace vectoriel de $\mathcal{V}$ ssi ces trois conditions sont satisfaites :
- W n'est pas vide : $W \neq \{\}$;
- W est fermé par addition vectorielle : $\forall u, v \in V, u \,\text{ADD}\, v \in W$;
- W est fermé par scaling : $\forall a \in S, \forall v \in V, a \,\text{SCALE}\, v \in W$.

Exemple
Soit $\mathcal{F} = (\mathbb{R}, +, \times)$. $\forall n \in \mathbb{N}, \forall i \in \{1, \ldots, n\}$, $(\mathbb{R}^n_{i=0}, +, \times)$ est un sous-espace vectoriel de $(\mathbb{R}^n, +, \times)$, avec $\mathbb{R}^n_{i=0}$ l'ensemble des n-uples à composantes réelles tels que la i-ème composante est nulle.

Remarque
Un sous-espace vectoriel d'un espace vectoriel sur $\mathcal{F}$ est en lui-même un espace vectoriel sur $\mathcal{F}$.

Définition 99 (Base, dimension d'un espace vectoriel)
Soient $\mathcal{F} = (S, \text{add}, \text{mult})$ un champ, avec zero l'identité de add ; $\mathcal{V} = (V, \text{ADD}, \text{SCALE})$ un espace vectoriel sur $\mathcal{F}$, avec ZERO l'identité de ADD ; et $W \subseteq V$. W est une base de $\mathcal{V}$ ssi ces deux conditions sont satisfaites :
- W est libre dans $\mathcal{V}$:

$$\forall f : W \to S, \text{ADD}\{s \,\text{SCALE}\, w | (w, s) \in f\} = \text{ZERO} \Longrightarrow \forall (w, s) \in f, s = \text{zero};$$

- W est générateur de $\mathcal{V}$:

$$\forall v \in V, \exists f : W \to S | v = \text{ADD}\{s \,\text{SCALE}\, w | (w, s) \in f\}.$$

Toute base de $\mathcal{V}$ est de même cardinal (théorème de la dimension). Ce nombre est nommé dimension de $\mathcal{V}$, et noté $\dim(\mathcal{V})$.

Exemple
Soient $\mathcal{F} = (\mathbb{R}, +, \times)$, $n \in \mathbb{N}$, et $\mathcal{V} = (\mathbb{R}^n, +, \times)$. $\{0_{i=1}^n | i \in \{1, \ldots, n\}\}$ est une base de $\mathcal{V}$, avec $0_{i=1}^n$ le n-uple dont toutes les composantes sont nulles, sauf la i-ème, égale à 1. $\mathcal{V}$ est donc de dimension n.

C.3 Produit scalaire

Un espace de produit scalaire, ou espace *préhilbertien*, est un espace vectoriel sur le champ $(\mathbb{R}, +, \times)$, muni d'une fonction supplémentaire permettant d'associer un scalaire à un couple de vecteurs.

Définition 100 (Produit scalaire)
Soit $\mathcal{V} = (V, \text{ADD}, \text{SCALE})$ un espace vectoriel sur $(\mathbb{R}, +, \times)$, avec ZERO l'identité de ADD. Un produit scalaire sur $\mathcal{V}$ est une fonction $\langle \cdot, \cdot \rangle \colon V \times V \to \mathbb{R}$ satisfaisant l'ensemble des conditions suivantes : $\forall u, v, w \in V, \forall a \in \mathbb{R}$:

- symétrie : $\langle v, w \rangle = \langle w, v \rangle$;
- linéarité :
  - $\langle a \,\text{SCALE}\, v, w \rangle = a \times \langle v, w \rangle$,
  - $\langle u \,\text{ADD}\, v, w \rangle = \langle u, w \rangle + \langle v, w \rangle$;
- définition positive : $v \neq \text{ZERO} \implies \langle v, v \rangle > 0$.

Exemple
$\forall n \in \mathbb{N}$, la fonction suivante est un produit scalaire sur $(\mathbb{R}^n, +, \times)$: $\forall x, y \in \mathbb{R}^n$:

$$\langle x, y \rangle = \sum_{i=1}^{n} (x_i \times y_i).$$

Définition 101 (Espace préhilbertien)
Un espace préhilbertien est un quadruplet $(V, \text{ADD}, \text{SCALE}, \langle \cdot, \cdot \rangle)$, avec $\mathcal{V} = (V, \text{ADD}, \text{SCALE})$ un espace vectoriel sur $(\mathbb{R}, +, \times)$, et $\langle \cdot, \cdot \rangle$ produit scalaire sur $\mathcal{V}$.

Annexe D

Éléments de théorie de la mesure et des probabilités

D.1 Tribu

Définition 102 (Tribu)
Soit S un ensemble. Une tribu, ou σ-algèbre, σ, sur S, est un ensemble de sous-ensembles de S tel que :

- σ contient l'ensemble vide : $\{\} \in \sigma$;
- σ est fermé par complémentaire : $E \in \sigma \Longrightarrow (S \setminus E) \in \sigma$;
- σ est fermé par union dénombrablement infinie :

$$(\forall n \in \mathbb{N}, E_n \in \sigma) \Longrightarrow \bigcup_{n \in \mathbb{N}} E_n \in \sigma.$$

Remarque
Si S est dénombrable, alors l'ensemble puissance de S est l'unique tribu sur S contenant tous les singletons $\{x\}$, $x \in S$.

Définition 103 (Tribu engendrée par un ensemble)
Soit S un ensemble, et E un ensemble de sous-ensembles de S. La tribu sur S engendrée par E est la tribu sur S minimale, au sens de l'inclusion, incluant E.

En d'autres termes, toute tribu sur S qui est sur-ensemble de E est également sur-ensemble de la tribu sur S engendrée par E.

D.2 Mesure

Définition 104 (Espace mesurable)
Un espace mesurable est un couple (S, σ), avec S un ensemble, et σ une tribu sur S.

Les éléments de σ (sous-ensembles de S) sont nommés *éléments mesurables* de (S, σ).

Définition 105 (Mesure)
Soit (S, σ) un espace mesurable. Une mesure μ sur σ est une fonction $\sigma \to \mathbb{R}^+ \cup \{\infty\}$ telle que :

- l'ensemble vide a une mesure nulle : $\mu(\{\}) = 0$;
- μ est additive par union disjointe dénombrablement infinie :

$$(\forall i, j \in \mathbb{N} | i \neq j, E_i, E_j \in \sigma | E_i \cap E_j = \{\}) \Longrightarrow \mu\left(\bigcup_{n \in \mathbb{N}} E_n\right) = \sum_{n \in \mathbb{N}} \mu(E_n).$$

Il est possible que nous simplifions parfois la notation $\mu(\{x\})$ par $\mu(x)$, pour tout singleton $\{x\}$, $x \in S$. Dès lors, nous pouvons parler de mesure de x plutôt que de mesure de $\{x\}$, par abus de langage.

Remarque
Il est facile de démontrer que l'additivité par union disjointe dénombrablement infinie implique l'additivité par union disjointe finie. Il suffit d'ajouter une séquence dénombrablement infinie d'ensembles vides à la suite d'une séquence finie F d'ensembles disjoints pour obtenir une séquence dénombrablement infinie I d'ensembles disjoints, sans modifier le résultat de leur union :

$$\bigcup_{n=1}^{|F|} F_n = \bigcup_{n \in \mathbb{N} \setminus \{0\}} I_n.$$

En effet, l'unique ensemble disjoint avec n'importe quel ensemble, même une copie de lui-même, est l'ensemble vide. De plus, celui-ci est élément de σ (cf. définition 102), et en utilisant les deux propriétés de μ exposées dans la définition précédente, nous obtenons le résultat suivant :

$$\begin{aligned}
\mu\left(\bigcup_{n=1}^{|F|} F_n\right) &= \mu\left(\bigcup_{n \in \mathbb{N} \setminus \{0\}} I_n\right) \\
&= \sum_{n \in \mathbb{N} \setminus \{0\}} \mu(I_n) \\
&= \sum_{n=1}^{|F|} \mu(F_n),
\end{aligned}$$

l'avant dernière égalité étant obtenue par additivité de μ, et la dernière, par le fait que l'ensemble vide a une mesure nulle.

Définition 106 (Espace mesuré)
Un espace mesuré est un triplet (S, σ, μ), avec (S, σ) un espace mesurable, et μ une mesure sur σ.

Les éléments mesurables de (S, σ) sont nommés *éléments mesurés* de (S, σ, μ).

D.3 Probabilité

Définition 107 (Probabilité)
Soit (S, σ) un espace mesurable. Une probabilité P sur σ est une mesure sur σ de valeur totale 1 : $\mathrm{P}(S) = 1$.

Une probabilité est également nommée *loi de probabilité*, *distribution*, ou encore *mesure probabiliste*. De plus, dans le vocabulaire probabiliste, il est d'usage de parler aussi bien d'*espace probabilisable* (resp. *élément probabilisable* ; *espace probabilisé* ; *élément probabilisé*, ou *événement*) que d'espace mesurable (resp. élément mesurable ; espace mesuré ; élément mesuré) dès lors que l'on considère des mesures probabilistes. Enfin, il est possible que nous simplifions parfois la notation $\mathrm{P}(\{x\})$ par $\mathrm{P}(x)$, pour tout singleton $\{x\}$, $x \in S$, et que nous parlions de probabilité de x plutôt que de probabilité de $\{x\}$, par abus de langage.

D.4 Variable aléatoire

Définition 108 (Fonction mesurable)
Soient (E, σ) et (F, τ) deux espaces mesurables. Une fonction $f : E \to F$ est dite (σ, τ)-mesurable ssi l'image réciproque par f de τ est un sous-ensemble de σ :

$$S \in \tau \Longrightarrow f^{-1}(S) \in \sigma.$$

Définition 109 (Variable aléatoire)
Soient (E, σ, P) un espace probabilisé, et (F, τ) un espace probabilisable. Une variable aléatoire sur (E, σ, P) est une fonction (σ, τ)-mesurable.

(F, τ) est nommé *espace état*, et F *ensemble état*, d'une telle variable X. Par abus de langage, nous disons aussi bien que X est à valeurs dans (F, τ) que dans F. De plus, X est dite *discrète* ssi $X(E)$ est dénombrable, et *continue* sinon.

Définition 110 (Loi de probabilité d'une variable aléatoire)
Soit X une variable aléatoire sur (E, σ, P), à valeurs dans (F, τ). La loi de probabilité de X est la probabilité notée P_X et définie comme suit :

$$S \in \tau \Longrightarrow \mathrm{P}_X(S) =_{\text{notation}} \mathrm{P}(X \in S) = \mathrm{P}(X^{-1}(S)).$$

Nous disons que X est *distribuée selon* P_X. La propriété de (σ, τ)-mesurabilité de X nous assure que $X^{-1}(S)$ est élément de σ, et donc élément probabilisé de (E, σ, P). X nous permet donc de définir l'espace probabilisé (F, τ, P_X) à partir de l'espace probabilisé (E, σ, P). L'intérêt de « transporter » ainsi les probabilités est de se donner la possibilité de raisonner dans un espace plus puissant, principalement en termes d'opérations (par exemple, passer du domaine symbolique au domaine numérique, comme illustré par l'exemple exposé en D.8). Enfin, il est possible que nous simplifions parfois la notation $\mathrm{P}(X \in \{x\})$ par $\mathrm{P}(X = x)$, pour tout singleton $\{x\}$, $x \in F$.

D.5 Indépendance

D.5.1 Indépendance d'événements

Définition 111 (Indépendance d'événements)
Soient (S, σ, P) un espace probabilisé, et A et B deux de ses événements. A et B sont dits indépendants selon P ssi :

$$\mathrm{P}(A \cap B) = \mathrm{P}(A) \times \mathrm{P}(B).$$

D.5.2 Indépendance de variables aléatoires

Définition 112 (Tribu produit)
Soient (E, σ) et (F, τ) deux espaces mesurables. La tribu produit de σ et τ, notée $\sigma \otimes \tau$, est la tribu sur $E \times F$ engendrée par l'ensemble suivant :

$$\{A \times B | (A \in \sigma) \wedge (B \in \tau)\}.$$

Définition 113 (Variable aléatoire et loi marginale et conjointe)
Soient (E, σ) et (F, τ) deux espaces mesurables, et X une variable aléatoire à valeurs dans $(E \times F, \sigma \otimes \tau)$. La première (resp. deuxième) variable aléatoire marginale de X est la variable aléatoire notée X_1 (resp. X_2), à valeurs dans (E, σ) (resp. (F, τ)), et définie comme suit : $\forall A \in \sigma$ (resp. $\forall B \in \tau$) :

$$\mathrm{P}(X_1 \in A) = \mathrm{P}(X \in A \times F) \ \left(\text{resp. } \mathrm{P}(X_2 \in B) = \mathrm{P}(X \in E \times B)\right).$$

P_{X_1} (resp. P_{X_2}) est nommée première (resp. deuxième) loi (de probabilité) marginale de X.

De manière réciproque, X (resp. P_X) est nommée variable aléatoire (resp. loi (de probabilité)) conjointe de X_1 et X_2.

Définition 114 (Indépendance de variables aléatoires)
Soient (E, σ) et (F, τ) deux espaces mesurables, et X une variable aléatoire à valeurs dans $(E \times F, \sigma \otimes \tau)$. X_1 et X_2 sont dites indépendantes ssi : $\forall (A \times B) \in (\sigma \otimes \tau)$:

$$\mathrm{P}(X \in A \times B) = \mathrm{P}(X_1 \in A) \times \mathrm{P}(X_2 \in B).$$

Il est d'usage de supposer que deux variables aléatoires soient indépendantes sans préciser leur loi conjointe. Cette dernière est dépendante du contexte, c'est-à-dire des processus incertains modélisés par ses variables marginales, et sa loi ne peut généralement pas être déduite de la donnée de ses lois marginales. Il existe une infinité de variables aléatoires conjointes ayant pour lois marginales un couple de probabilités donné, et dans le contexte précité la loi conjointe ne peut donc qu'être supposée connue.

D.6 Arithmétique de variables aléatoires réelles

Une variable aléatoire est couramment traitée comme un élément de l'ensemble dans lequel elle est à valeurs. Pour le cas de variables réelles, il en résulte l'écriture d'additions, multiplications... de variables aléatoires entres elles ou bien de variables aléatoires avec des nombres réels. Cette écriture implicite effectivement la définition d'une nouvelle variable aléatoire sur la tribu borélienne de $\mathbb{R}$, dont la loi est définie par la probabilité que le résultat des opérations considérées, après *réalisation* des processus modélisés par les variables qui en sont les « causes », soit dans un intervalle donné.

Par exemple, si nous notons $R = X + Y + 3$, avec X et Y des variables réelles, nous implicitons le fait que R soit une variable réelle, modélisant l'addition de 3 au résultat de l'addition de la réalisation des processus modélisés par X et Y.

Tout comme pour le cas de variables aléatoires jointes, il est d'usage de recourir à une telle arithmétique pour formaliser le résultat R d'opérations de la sorte, sans préciser la loi de probabilité de R. Or la donnée la séquence de variables aléatoire cause de R ne permet généralement pas de déduire la loi de probabilité de R, et celle-ci est donc généralement supposée connue.

D.7 Statistiques de variables aléatoires réelles

Typiquement, une variable aléatoire prend ses valeurs dans l'ensemble des réels munis de sa tribu *borélienne*.

Définition 115 (Tribu borélienne de $\mathbb{R}$)
La tribu borélienne de $\mathbb{R}$, notée β, est la tribu sur $\mathbb{R}$ engendrée par l'ensemble des intervalles ouverts de $\mathbb{R}$ (intervalles de la formes $]a; b[$, avec $a, b \in \mathbb{R}$).

Une conséquence de cette définition est que β inclue l'ensemble des intervalles de $\mathbb{R}$. Les éléments d'une telle tribu sont nommés *boréliens*.

Définition 116 (Variable aléatoire borélienne)
Une variable aléatoire borélienne est une variable aléatoire à valeurs dans $(\mathbb{R}, \beta)$.

Par abus de langage, nous parlons aussi bien de variable aléatoire *réelle*. Un grand nombre de statistiques probabilistes ont été historiquement formalisées pour décrire de telles variables. Nous allons en voir un panel non exhaustif, mais cela nécessite que nous ayons connaissance des concepts de *masse* et *densité* de probabilité.

Définition 117 (Masse de probabilité d'une variable aléatoire réelle discrète)
Soit X une variable aléatoire réelle discrète. La masse de probabilité de X, notée p_X, est la fonction $\mathbb{R} \to [0; 1]$ définie comme suit : $\forall x \in \mathbb{R}$:

$$\mathrm{p}_X(x) = \mathrm{P}(X = \{x\}).$$

Il s'agit donc de la probabilité que X ait exactement une valeur donnée.

Définition 118 (Densité de probabilité d'une variable aléatoire réelle continue)
Soit X une variable aléatoire réelle continue. La densité de probabilité de X, notée p_X, est la fonction intégrable $\mathbb{R} \to \mathbb{R}^+$ définie comme suit : $\forall a, b \in \mathbb{R}$:

$$\mathrm{P}(X \in [a; b]) = \int_a^b \mathrm{p}_X(x)\ dx.$$

La probabilité que la valeur de X soit comprise dans un intervalle est obtenue par intégration de p_X.

D.7.1 Espérance mathématique

Définition 119 (Espérance mathématique d'une variable aléatoire réelle)
Soit X une variable aléatoire réelle discrète (resp. continue). L'espérance mathématique, ou simplement espérance, de X, est donnée par la formule suivante :

$$\mathrm{E}(X) = \sum_{x \in \mathbb{R}} \left[x \times \mathrm{p}_X(x) \right] \ \left(\text{resp.} \int_{\mathbb{R}} x \times \mathrm{p}_X(x) \ dx \right).$$

D.7.2 Moments

Définition 120 (Moment d'une variable aléatoire réelle)
Soient X une variable aléatoire réelle discrète (resp. continue), et $n \in \mathbb{N}$. Le moment d'ordre n de X est donné par la formule suivante :

$$m_n(X) = \mathrm{E}(X^n) = \sum_{x \in \mathbb{R}} \left[x^n \times \mathrm{p}_X(x) \right] \ \left(\text{resp.} \int_{\mathbb{R}} x^n \times \mathrm{p}_X(x) \ dx \right).$$

Définition 121 (Moment centré d'une variable aléatoire réelle)
Soient X une variable aléatoire réelle discrète (resp. continue), et $n \in \mathbb{N}$. Le moment centré d'ordre n de X est donné par la formule suivante :

$$\mu_n(X) = \mathrm{E}\left((X - \mathrm{E}(X))^n \right) = \sum_{x \in \mathbb{R}} \left[(x - \mathrm{E}(X))^n \times \mathrm{p}_X(x) \right] \ \left(\text{resp.} \int_{\mathbb{R}} (x - \mathrm{E}(X))^n \times \mathrm{p}_X(x) \ dx \right).$$

Certains moments sont connus sous un nom particulier :
- l'espérance $m_1(X)$ de X est également nommée *moyenne théorique*, ou simplement *moyenne*, de X ;
- $\mu_2(X)$ est nommé *variance* de X, notée $\mathrm{V}(X)$, et sa racine carrée est nommée *écart-type* de X.

D.8 Exemple : le lancé de dé

Supposons l'expérience physique suivante : le lancé d'un dé équilibré à 6 faces. Par « équilibré », nous entendons que le dé est un cube de masse uniformément répartie. Nous pouvons alors supposer l'équiprobabilité d'apparition de chacune des 6 faces, et il est ainsi tout naturel de modéliser mathématiquement l'expérience par l'espace probabilisé (cf. définitions 106 et 107) suivant :

$$(E, 2^E, \mathrm{P}),$$

avec :

$$E = \{\text{face } 1, \text{face } 2, \text{face } 3, \text{face } 4, \text{face } 5, \text{face } 6\},$$
$$\mathrm{P}(\{\text{face } i\}) = \frac{1}{6}, \forall i \in \{1, \ldots, 6\}.$$

La probabilité de tout élément de 2^E qui n'est pas un singleton est déduite de la propriété d'additivité des mesures et probabilités (cf. définition 105), et donc 0 pour l'ensemble vide.

Souhaitant analyser statistiquement notre expérience, nous choisissons d'exprimer notre connaissance probabiliste dans un espace réel, bien moins restrictif mathématiquement que notre espace symbolique initial. Nous optons alors pour l'intermédiaire variable aléatoire réelle (définition 116) suivante :

$$X(\text{face } i) = i, \forall i \in \{1, \ldots, 6\}.$$

Notons que X est discrète, car $X(E) = \{1, \ldots, 6\}$ est un ensemble dénombrable. La loi de probabilité (définition 110) de X est définie comme suit :

$$\mathrm{P}(X \in I) = |I \cap \{1, \ldots, 6\}| \times \frac{1}{6}, \forall I \subseteq \mathbb{R},$$

et X admet alors la masse de probabilité suivante :

$$\mathrm{p}_X(i) = \frac{1}{6}, \forall i \in \{1, \ldots, 6\},$$
$$\mathrm{p}_X(i) = 0, \forall i \in \mathbb{R} \setminus \{1, \ldots, 6\}.$$

Ainsi, la moyenne de X est égale à 3,5, et son écart-type est légèrement inférieur à 1,71 (cf. définitions 120 et 121).

Notons que 3,5 ne représente pas nécessairement l'arrête entre les faces 3 et 4, car nous avons changé d'espace probabilisé. De plus, nous n'« espérons » certainement pas obtenir un dé en équilibre sur cette arrête après un lancé, ce genre de résultat n'étant généralement guère interprétable et utile en soi mais servant plutôt de donnée intermédiaire au calcul et à la déduction d'information de plus haut niveau, comme par exemple la probabilité d'apparition de telle ou telle valeur lorsque l'on additionne des résultats de lancés de dés successifs (cf. théorème 1).

Bibliographie

[ASS00] R. Alquézar, F. Serratosa, and A. Sanfeliu. Distance between attributed graphs and function-described graphs relaxing 2nd order restrictions. In *SSSPR*, pages 277–286, 2000.

[BG01] H. Bunke and S. Günter. Weighted mean of a pair of graphs. *Computing*, 67(3) :209–224, 2001.

[Bil05] P. Bille. A survey on tree edit distance and related problems. *Theoretical computer science*, 337(1-3) :217–239, 2005.

[Bis06] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[BJAK02] H. Bunke, X. Jiang, K. Abegglen, and A. Kandel. On the weighted mean of a pair of strings. *Pattern analysis and applications*, 5(1) :23–30, 2002.

[BK00] H. Bunke and A. Kandel. Mean and maximum common subgraph of two graphs. *Pattern recognition letters*, 21(2) :163–168, 2000.

[BMJ99] H. Bunke, A. Münger, and X. Jiang. Combinatorial search versus genetic algorithms : a case study based on the generalized median graph problem. *Pattern recognition letters*, 20(11-13) :1271–1277, 1999.

[BP66] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37(6) :1554–1563, 1966.

[BR03] C. Blum and A. Roli. Metaheuristics in combinatorial optimization : overview and conceptual comparison. *ACM computing surveys*, 35(3) :268–308, 2003.

[BS90] H. Bunke and A. Sanfeliu, editors. *Syntactic and structural pattern recognition : theory and applications*. World Scientific, 1990.

[Bun99] H. Bunke. Error correcting graph matching : on the influence of the underlying cost function. *IEEE transactions on pattern analysis and machine intelligence*, 21(9) :917–922, 1999.

[CAV98] P. P. Cruz-Alcázar and E. Vidal. Learning regular grammars to model musical style : comparing different coding schemes. In *ICGI*, pages 211–222, 1998.

[CB08] M. Christodoulakis and G. Brey. Edit distance with single-symbol combinations and splits. In *PSC*, pages 208–217, 2008.

[CdA97] F. Casacuberta and M. D. de Antonio. A greedy algorithm for computing approximate median strings. In *SNRFAI*, pages 193–198, 1997.

[CO94] R. Carrasco and J. Oncina. Learning stochastic regular grammars by means of a state merging method. In *ICGI*, pages 139–152, 1994.

[dlH05] C. de la Higuera. A bibliographical study of grammatical inference. *Pattern recognition*, 38(9) :1332–1348, 2005.

[dlHC00] C. de la Higuera and F. Casacuberta. Topology of strings : median string is NP-complete. *Theoretical computer science*, 230(1-2) :39–48, 2000.

[Fod02] I. Fodor. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Laboratory, 2002.

[For73] G. D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3) :268–278, 1973.

[FVS+08] M. Ferrer, E. Valveny, F. Serratosa, K. Riesen, and H. Bunke. An approximate algorithm for median graph computation using graph embedding. In *ICPR*, pages 1–4, 2008.

[Gär03] T. Gärtner. A survey of kernels for structured data. *ACM SIGKDD explorations newsletter*, 5(1) :49–58, 2003.

[Ham50] R. W. Hamming. Error detecting and error correcting codes. *Bell system technical journal*, 26(2) :147–160, 1950.

[Har78] M. A. Harrison. *Introduction to formal language theory.* Addison-Wesley, 1978.

[HN03] X. He and P. Niyogi. Locality preserving projections. In *NIPS*, 2003.

[HSS08] T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *Annals of statistics*, 36(3) :1171–1220, 2008.

[HW03] A. Hlaoui and S. Wang. A new median graph algorithm. In *GbRPR*, pages 225–234, 2003.

[JABC03] X. Jiang, K. Abegglen, H. Bunke, and J. Csirik. Dynamic computation of generalised median strings. *Pattern analysis and applications*, 6(3) :185–193, 2003.

[JBC04] X. Jiang, H. Bunke, and J. Csirik. Median strings : a review. In *Data mining in time series databases*, pages 173–192. World Scientific, 2004.

[JDM00] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition : a review. *IEEE transactions on pattern analysis and machine intelligence*, 22(1) :4–37, 2000.

[Jel98] F. Jelinek. *Statistical methods for speech recognition.* The MIT Press, 1998.

[JMB01] X. Jiang, A. Münger, and H. Bunke. On median graphs : properties, algorithms, and applications. *IEEE transactions on pattern analysis and machine intelligence*, 23(10) :1144–1151, 2001.

[JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering : a review. *ACM computing surveys*, 31(3) :264–323, 1999.

[JMM96] A. K. Jain, J. Mao, and K. M. Mohiuddin. Artificial neural networks : a tutorial. *IEEE transactions on computer*, 29(3) :31–44, 1996.

[Jol03a] J.-M. Jolion. The deviation of a set of strings. *Pattern analysis and applications*, 6(3) :224–231, 2003.

[Jol03b] J.-M. Jolion. Some experiments on clustering a set of strings. In *GbRPR*, pages 214–224, 2003.

[Jol06] J.-M. Jolion. Synthèse et caractérisation statistique d'ensemble de structures symboliques : application à l'estimation d'une densité de probabilités. Technical report, LIRIS, 2006.

[KB96] T. E. Kammeyer and R. K. Belew. Stochastic context-free grammar induction with a genetic algorithm using local search. In *Foundations of genetic algorithms*, pages 409–436, 1996.

[KKS05] L. Käll, A. Krogh, and E. L. L. Sonnhammer. An HMM posterior decoder for sequence feature prediction that includes homology information. *Bioinformatics*, 21(1) :251–257, 2005.

[Koh85] T. Kohonen. Median strings. *Pattern recognition letters*, 3(5) :309–313, 1985.

[Kru99] F. Kruzslicz. Improved greedy algorithm for computing approximate median strings. *Acta cybernetica*, 14(2) :331–339, 1999.

[Laf99] J. D. Lafferty. Notes on probabilistic context-free grammars 11-761 : language and statistics, 1999.

[LBS84] R. A. Olshen L. Breiman, J. H. Friedman and C. J. Stone. *Classification and regression trees.* Wadsworth, 1984.

[Lev66] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8) :707–710, 1966.

[LS92] G. Louchard and W. Szpankowski. A probabilistic analysis of a string edit problem. Technical Report 1814, INRIA, 1992.

[LY90] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer speech and language*, 4 :35–56, 1990.

[MHJC00] C. D. Martínez-Hinarejos, A. Juan, and F. Casacuberta. Use of median string for classification. In *ICPR*, pages 2903–2906, 2000.

[MJC01] C. D. Martínez, A. Juan, and F. Casacuberta. Improving classification using median string and NN rules. In *SNRFAI*, pages 391–395, 2001.

[MV93] A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *IEEE transactions on pattern analysis and machine intelligence*, 15(9) :926–932, 1993.

[NB06] M. Neuhaus and H. Bunke. Edit distance-based kernel functions for structural pattern classification. *Pattern recognition*, 39(10) :1852–1863, 2006.

[NRB06] M. Neuhaus, K. Riesen, and H. Bunke. Fast suboptimal algorithms for the computation of graph edit distance. In *SSPR*, pages 163–172, 2006.

[ORO08] C. Olivares-Rodríguez and J. Oncina. A stochastic approach to median string computation. In *SSPR*, pages 431–440, 2008.

[PM02] G. Peris and A. Marzal. Fast cyclic edit distance computation with weighted edit costs in classification. In *ICPR*, pages 184–187, 2002.

[Qui93] J. R. Quinlan. *C4.5 : programs for machine learning*. Morgan Kaufmann, 1993.

[Rab89] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2) :257–286, 1989.

[RJ08a] S. Rebecchi and J.-M. Jolion. Lois uniformes et normales de chaînes discrètes. In *RFIA*, pages 471–480, 2008.

[RJ08b] S. Rebecchi and J.-M. Jolion. On the Gaussian distribution of strings. In *ICPR*, pages 1–4, 2008.

[RJ08c] S. Rebecchi and J.-M. Jolion. On the uniform distribution of strings. In *PSC*, pages 116–125, 2008.

[RKH05] A. Robles-Kelly and E. R. Hancock. Graph edit distance from spectral seriation. *IEEE transactions on pattern analysis and machine intelligence*, 27(3) :365–378, 2005.

[RNB07] K. Riesen, M. Neuhaus, and H. Bunke. Graph embedding in vector spaces by means of prototype selection. In *GbRPR*, pages 383–393, 2007.

[SAS02a] F. Serratosa, R. Alquézar, and A. Sanfeliu. Estimating the joint probability distribution of random vertices and arcs by means of second-order random graphs. In *SSPR*, pages 252–262, 2002.

[SAS02b] F. Serratosa, R. Alquézar, and A. Sanfeliu. Synthesis of function-described graphs and clustering of attributed graphs. *International journal of pattern recognition and artificial intelligence*, 16(6) :621–656, 2002.

[SAS03] F. Serratosa, R. Alquézar, and A. Sanfeliu. Function-described graphs for modelling objects represented by sets of attributed graphs. *Pattern recognition*, 36(3) :781–798, 2003.

[Sch00] B. Schölkopf. The kernel trick for distances. In *NIPS*, pages 301–307, 2000.

[SJ05] I. Simand and J.-M. Jolion. Représentation d'images par chaînes de symboles : application à la recherche par le contenu. In *GRETSI*, pages 925–928, 2005.

[SLT02] G. Sánchez, J. Lladós, and K. Tombre. A mean string algorithm to compute the average among a set of 2D shapes. *Pattern recognition letters*, 23(1-3) :203–213, 2002.

[SNB+06] B. Spillmann, M. Neuhaus, H. Bunke, E. Pekalska, and R. P. W. Duin. Transforming strings to vector spaces using prototype selection. In *SSPR*, pages 287–296, 2006.

[SP03] J. S. Sim and K. Park. The consensus string problem for a metric is NP-complete. *Journal of discrete algorithms*, 1(1) :111–117, 2003.

[SS07] D. Shapira and J. A. Storer. Edit distance with move operations. *Journal of discrete algorithms*, 5(2) :380–392, 2007.

[SSA04] A. Sanfeliu, F. Serratosa, and R. Alquézar. Second-order random graphs for modeling sets of attributed graphs and their application to object learning and recognition. *International journal of pattern recognition and artificial intelligence*, 18(3) :375–396, 2004.

[SSA08] G. Sanromà, F. Serratosa, and R. Alquézar. Shape learning with function-described graphs. In *ICIAR*, pages 475–484, 2008.

[Ste56] H. Steinhaus. Sur la division des corps matériels en parties. *Bulletin de l'académie polonaise des sciences*, C1. III vol IV :801–804, 1956.

[TDdlH00] F. Thollard, P. Dupont, and C. de la Higuera. Probabilistic DFA inference using Kullback-Leibler divergence and minimality. In *ICML*, pages 975–982, 2000.

[Tsa90] W.-H. Tsai. Combining statistical and structural methods. In *Syntactic and structural pattern recognition : theory and applications*, pages 349–366. World Scientific, 1990.

[TSN90] G. G. Towell, J. W. Shavlik, and M. O. Noordewier. Refinement of approximate domain theories by knowledge-based neural networks. In *AAAI*, pages 861–866, 1990.

[Vap98] V. Vapnik. *Statistical learning theory*. Springer, 1998.

[Vit67] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on information theory*, 13(2) :260–269, 1967.

[VK92] J. A. Vlontzos and S. Y. Kung. Hidden Markov models for character recognition. *IEEE transactions on image processing*, 1(4) :539–543, 1992.

[VTdlH+05a] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. Probabilistic finite-state machines - part I. *IEEE transactions on pattern analysis and machine intelligence*, 27(7) :1013–1025, 2005.

[VTdlH+05b] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. C. Carrasco. Probabilistic finite-state machines - part II. *IEEE transactions on pattern analysis and machine intelligence*, 27(7) :1026–1039, 2005.

[Wat85] S. Watanabe. *Pattern recognition : human and mechanical*. Wiley, 1985.

[WCY90] A. K. C. Wong, J. Costant, and M. L. You. Random graphs. In *Syntactic and structural pattern recognition : theory and applications*, pages 197–236. World Scientific, 1990.

[Wei37] E. Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku mathematical journal*, 43 :355–386, 1937.

[WF74] R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1) :168–173, 1974.

[Whi94] D. Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2) :65–85, 1994.

[WHL05] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE transactions on pattern analysis and machine intelligence*, 27(7) :1112–1124, 2005.