HAL
open science

Visualisation d'information : de la théorie sémiotique à des exemples pratiques basés sur la représentation de graphes et d'hypergraphes

Arnaud Sallaberry

To cite this version:

Arnaud Sallaberry. Visualisation d'information : de la théorie sémiotique à des exemples pratiques basés sur la représentation de graphes et d'hypergraphes. Interface homme-machine [cs.HC]. Université Sciences et Technologies - Bordeaux I, 2011. Français. NNT : . tel-00646397

HAL Id: tel-00646397

https://theses.hal.science/tel-00646397

Submitted on 29 Nov 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N^o d'ordre : 4314

THÈSE

présentée à

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET INFORMATIQUE

par Arnaud SALLABERRY

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : Informatique

Visualisation d'information : de la théorie sémiotique à des exemples pratiques basés sur la représentation de graphes et d'hypergraphes

---

Soutenue le : 18 Octobre 2011

Après avis de :

M.	Alexandru Telea	Professeur	**Rapporteur**
M.	Michel Crampes .	Enseignant-Chercheur (HDR)	**Rapporteur**

Devant la Commission d'Examen composée de :

M.	Christian Retoré	Professeur	**Président**
Mme.	Marie-Luce Viaud	Chargée de recherche	**Examinatrice**
Mme.	Sabine Cornelsen	Assistante de recherche	**Examinatrice**
M.	David Auber	Maître de conférence.....	**Examinateur**
M.	Christophe Douy	Industriel	**Co-directeur de thèse**
M.	Guy Melançon ..	Professeur	**Directeur de thèse**

- 2011 -

N^o d'ordre : 4314

THÈSE

présentée à

L'UNIVERSITÉ BORDEAUX I

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET INFORMATIQUE

par Arnaud SALLABERRY

POUR OBTENIR LE GRADE DE

DOCTEUR

SPÉCIALITÉ : Informatique

Visualisation d'information : de la théorie sémiotique à des exemples pratiques basés sur la représentation de graphes et d'hypergraphes

---

Soutenue le : 18 Octobre 2011

Après avis de :

M.	Alexandru Telea	Professeur	**Rapporteur**
M.	Michel Crampes .	Enseignant-Chercheur (HDR)	**Rapporteur**

Devant la Commission d'Examen composée de :

M.	Christian Retoré	Professeur	**Président**
Mme.	Marie-Luce Viaud	Chargée de recherche	**Examinatrice**
Mme.	Sabine Cornelsen	Assistante de recherche	**Examinatrice**
M.	David Auber	Maître de conférence.....	**Examinateur**
M.	Christophe Douy	Industriel	**Co-directeur de thèse**
M.	Guy Melançon ..	Professeur	**Directeur de thèse**

- 2011 -

Remerciements

Pour commencer, je souhaite remercier Maylis Delest qui m'a introduit auprès de Guy Melançon et sans qui cette thèse n'aurait pas eu lieu. Je voudrais ensuite remercier Guy Melançon et Christophe Douy qui m'ont donné la chance de réaliser ces travaux. Leur soutien aussi bien intellectuel et technique que moral a largement contribué au bon déroulement de ces trois années.

Je tiens aussi à remercier l'ANRT, la société Pikko, l'INRIA Bordeaux Sud-Ouest et l'Agence Nationale de la Recherche (projet RNTL FIVE 06 TLOG 12) pour avoir financé mes travaux de thèse ainsi que l'Université Bordeaux 1 et le LaBRI pour m'avoir permis de les réaliser dans de bonnes conditions.

Je souhaite ensuite remercier tous les membres de mon jury de thèse, Alexandru Telea, Michel Crampes, Christian Retoré, Marie-Luce Viaud, Sabine Cornelsen et David Auber pour avoir accepté de relire mon manuscrit de thèse et pour m'avoir permis de l'améliorer par leurs remarques.

Merci aussi à toutes les personnes avec qui j'ai eu le privilège de travailler durant ces trois années. Je pense, en ce qui concerne le LaBRI, à Faraz Zaidi, David Auber, Antoine Lambert et Charles Huet. Il ne faudrait pas non plus oublier Christian Pich de l'ETH Zurich, Maguelonne Teisseire du CEMAGREF de Montpellier, Mathieu Roche, Nicolas Pecheur et Sandra Bringay du LIRMM à Montpellier, Ulrik Brandes, Sabine Cornelsen et Barbara Pampel de l'Université de Constance. Enfin, n'oublions pas Anaïs Leveuvre du LaBRI. Nos longues discussions sur la sémiotique m'ont permis d'éclaircir de nombreux concepts qui seront abordés au chapitre 1 du présent document.

Je voudrais aussi remercier tous les membres des différentes équipes qui m'ont accueillies durant la thèse. Tout d'abord un grand merci à toute l'équipe GRAVITE pour leur disponibilité et leur bonne humeur. Je n'oublie pas non plus les membres de Pikko, structure dans laquelle j'ai effectué la majorité de mes travaux. Outre Christophe que j'ai déjà mentionné ci-dessus, merci au directeur technique Emmanuel Pellegrin, à Benjamin Combes qui a encadré mon stage de master au sein de la structure, et à Guillaume Aveline pour ses conseils techniques (je l'ai d'ailleurs remercié sur plusieurs des articles

que je présenterai ci-dessous). Enfin, merci à Ulrik Brandes et toute l'équipe d'algorithmique de l'Université de Constance pour m'avoir accueilli pendant presque deux mois avec tant de chaleur. Grâce à eux, ce séjour aura été pour moi aussi enrichissant sur la plan professionnel que personnel.

Je remercie d'autre part tous les membres de ma famille et mes amis pour le soutien qu'ils m'ont apporté. En particulier, un grand merci à ma mère Maryse et mon père Jean-Louis qui ont toujours su me pousser à donner le meilleur de moi-même. Je pense aussi à Jean-Jacques, mon défunt grand-père, qui m'a beaucoup soutenu tout au long de mes études et qui nous a quitté fin 2010.

Pour finir, la dernière mais pas la moindre, je remercie ma compagne Célia. Elle a été mon principal soutien moral durant ces trois années. Outre les efforts qu'elle a fait pour m'aider à améliorer le présent manuscrit, elle a toujours été là pour me supporter dans tous les sens que peut prendre ce terme.

Visualisation d'information : de la théorie sémiotique à des exemples basés sur la représentation de graphes et d'hypergraphes

Résumé :

La visualisation d'information est une discipline récente en pleine expansion et qui a pour objet l'étude des méthodes de représentation visuelle de données abstraites, c'est-à-dire non géolocalisées. La sémiotique est quant à elle une discipline beaucoup plus ancienne (fin du XIXième siècle) qui s'intéresse aux divers systèmes de signes nécessaires aux processus de communication. À ce jour, peu de travaux ont été réalisés pour mettre en parallèle ces deux disciplines. C'est pourquoi le premier chapitre de cette thèse est dédié à l'étude de la visualisation d'information selon les paradigmes élaborés par son ainée tout au long du XXième siècle. Nous montrons en particulier comment l'un des modèles les plus aboutis de validation de visualisations (modèle imbriqué de Tamara Munzner) correspond au processus d'étude sémiotique d'énoncés. Le second chapitre est consacré à la visualisation de graphe, outil de modélisation puissant de divers ensembles de données abstraites. Nous proposons d'une part une application permettant de visualiser et de naviguer à travers les pages Internet retournées par un moteur de recherche et d'autre part un algorithme de visualisation de hiérarchies dynamiques sous forme de « cartes géographiques ». Enfin, nous évoquons dans le troisième chapitre un autre outil de modélisation de données abstraites : les hypergraphes. Nous proposons des résultats théoriques concernant leur représentation et donnons une ébauche de solution permettant de les visualiser.

Mots-clés : Visualisation d'information, Dessin de graphes et d'hypergraphes, Sémiotique, Sémiologie.

Discipline : Informatique.

Information visualization : from semiotic theory to practical examples based on graphs and hypergraphs representation

Abstract :

Information visualization aims at designing visual representations of abstract data, furthermore relying on interaction as a mean to discover knowledge. The first part of this thesis challenges Information Visualization by drawing a parallel with semiotics, a 19th century research field focusing on systems of signs required for communication. We develop a point of view on Information Visualization based on the paradigms developed by semioticians during the 20th century. In particular, we show how the visualization validation model proposed by Tamara Munzner is related to the process used by semioticians for utterance analysis. The second part of the thesis focuses on graph visualization and describes two techniques and system prototypes targeting specific application domains. The first one is an interactive technique to visualize and navigate through Web search results. The second one is an algorithm for the visualization of dynamic hierarchies exploiting the analogy with "geographical maps". Finally, the third chapter is devoted to another model used to structure abstract data : hypergraphs. We propose theoretical results on hypergraph drawing and a preliminary technique to visualize hypergraphs.

Keywords : Information Visualization, Graph and hypergraph drawing, Semiotic, Semiology.

Field : Computer Science.

Table des matières

Remerciements **i**

Remerciements **i**

Resumé **iii**

Abstract **iv**

Table des matières **v**

Table des figures **ix**

Liste des tableaux **xvii**

Introduction **1**

1 Une approche sémiotique de la visualisation d'information **5**

1.1 *SequencesViewer* : un outil de visualisation de séquences de gènes 8

1.1.1 Données manipulées 8

1.1.2 Système 9

1.2 Le signe 17

1.2.1 Le signe comme élément du processus de communication 17

1.2.2 Le signe comme élément du processus de signification 21

1.2.3 Le signe perçu selon trois dimensions : sémantique, syntaxique et pragmatique 24

1.3 La visualisation de l'information 30

1.3.1 Définition(s) 31

1.3.2 Modèle imbriqué pour la production et l'évaluation des cartes . . . 33

1.4 Conclusion 45

2 Visualisation de graphes 47

2.1 Visualisation et navigation interactive dans un ensemble de pages *web* . . 48

2.1.1 Travaux précédents 52

2.1.2 Fragmentation 53

2.1.3 Représentation et interaction 57

2.1.4 Études de cas 64

2.1.5 Conclusion 66

2.2 *Geographical treemaps* 67

2.2.1 Travaux précédents 71

2.2.2 Vue d'ensemble de la méthode 78

2.2.3 Algorithmes 79

2.2.4 Techniques d'interaction 88

2.2.5 Résultats 88

2.3 Conclusion 92

3 Visualisation d'hypergraphes 95

3.1 « Planarité » des hypergraphes basée sur les graphes supports 99

3.1.1 Composantes bi-connexes 102

3.1.2 Supports cactus 106

3.1.3 Hypergraphes à intersections et différences fermées 108

3.1.4 Conclusion 111

3.2 Supports dans lesquels des chemins modélisent les hyperarêtes 112

3.2.1 Les supports basés sur des chemins minimaux et planaires 112

3.2.2 Vers une visualisation 115

3.3 Conclusion 119

4 Articles publiés et communications diverses **121**

4.1 Visualisation de séquences de gènes 121

4.2 Analyse, fragmentation et visualisation de graphes 122

4.3 Dessin d'hypergraphes 123

Conclusion et perspectives **125**

Bibliographie **129**

Table des figures

0.1 Ce que pouvait représenter la carte d'Anaximandre, d'après Marcel Conche [3]. La ville de Milet y est repérable grâce à un M. 2

1.1 Exemple de hiérarchie d'une ensemble de 5 séquences de gènes. 9

1.2 Vue d'ensemble du système. Notre application correspond à la partie *Visualisation and navigation*. 10

1.3 Nuage de points : représentation globale des groupes de séquences. 11

1.4 Nuage de points dont les séquences qui ne sont pas des centres ont été masquées. 11

1.5 Nuage de points : les séquence vertes contiennent un gène recherché par l'utilisateur. 11

1.6 Lorsque l'utilisateur effectue un clic droit sur une séquence, une bulle contenant les informations de la séquences apparaît. 12

1.7 Les ascenseurs permettent de filtrer les séquences selon la valeur de leurs propriétés sain et malade. 12

1.8 Étiquetage excentré des séquences. 13

1.9 Visualisation détaillée d'un groupe. 13

1.10 Visualisation d'une séquence avec les documents qui lui sont associés. 14

1.11 Navigation à travers les groupes de séquences. 14

1.12 *Treemap* : trois niveaux sont affichés, la racine, ses enfants et les propriétés sain et malade de ses enfants. 15

1.13 *Treemap* : trois niveaux sont affichés, la racine, ses enfants et les enfants de ses enfants. 16

1.14 *Treemap* : un chemin permet d'accéder aux sommets plus élevés dans la hiérarchie. 16

1.15 Lorsque l'utilisateur effectue un clic droit sur un sommet, une bulle contenant un bouton d'accès au nuage de points apparaît. 17

1.16 Modèle de Jakobson, les six pôles présent dans un acte de communication : émetteur (producteur d'un objet), récepteur (destinataire de l'objet), référent (ce de quoi parle l'objet), canal (support physique de l'objet, *e.g.* un écran d'ordinateur), code (règles permettant d'attribuer une signification à l'objet), message (lieu d'interaction des cinq facteurs énoncés ci-dessus, signe ou ensemble de signes). 19

1.17 Définition triadique du signe. Signifiant : entité physique du signe. Signifié : concept signifié. Référent : objet auquel le signe se réfère. 22

1.18 Le *pipeline* de la visualisation de l'information (image provenant du site de la librairie de visualisation de l'information *Prefuse*) : « *Visualization can be described as the mapping of data to visual form that supports human interaction in a workspace for visual sense making* » [40]. **Données brutes (*source data*)** : données issues de phénomènes observés, mesurés etc.. **Données structurées (*data tables*)** : données nettoyées et modélisées selon un format pouvant être décodé par l'outil de visualisation. **Abstraction visuelle (*visual abstraction*)** : type de représentation adapté au modèle des données. **Vues (*views*)** : représentation partielle des données modélisées grâce au type de représentation choisi. 33

1.19 Modèle imbriqué pour la création de cartes (image provenant de l'article [131]) : **Caractérisation des problèmes du domaine cible (*domain problem characterization*)** : le cartographe doit comprendre les problèmes du domaine des destinataires. **Conception des données et des opérations (*data/-operation abstraction design*)** : le cartographe doit transformer les données brutes en données abstraites structurées et exprimer en termes informatiques les problèmes du domaine des destinataires que sa carte doit résoudre. **Conception des techniques de représentation et d'interaction (*encoding/interaction technique design*)** : Le cartographe doit sélectionner une abstraction visuelle des données structurées (*e.g.* variables visuelles, métaphore de représentation, *etc.*) qu'il va utiliser, ainsi que les techniques d'interaction dont sa carte sera munie. **Conception des algorithmes (*algorithm design*)** : trouver les algorithmes permettant de mettre en place les techniques conçues à l'étape précédente. 34

1.20 Modèle imbriqué pour la validation de cartes (image provenant de l'article [131]). Les couleurs correspondent aux étapes de production de la figure 1.19. Pour chacune de ces étapes, un type d'erreur est indiqué ainsi que des méthodes permettant de ne pas les commettre (certaines de ces méthodes doivent être mises en place lors de l'étape de production correspondante, d'autres nécessitent la réalisation des étapes suivantes, d'où le nom de modèle imbriqué) : **Mauvais problème (*wrong problem*)** : les problèmes auxquels sont confrontés les destinataires et que la carte doit aider à résoudre ne sont pas compris du cartographe. **Mauvaise abstraction des données et des opérations (*bad data/operation abstraction*)** : le modèle de données structurées choisi ou les problèmes en termes informatiques identifiés ne permettent pas de résoudre le problème des destinataires. **Techniques de représentation et d'interaction non efficaces (*ineffective encoding/interaction technique*)** : les techniques de représentation et d'interaction ne permettent pas la communication des données abstraites à un destinataire. **Algorithme lent (*slow algorithm*)** : les techniques algorithmiques mises en place sont trop lentes ce qui peut rendre le système inutilisable. 36

1.21 Convergence de l'algorithme de *multidimensional scaling* utilisé pour placer les centres dans le nuage de points : le nombre indiqué dans la légende correspond au nombre de centres à placer. 38

1.22 992 séquences sont affichées sur un écran de 15,4 pouces avec une résolution de 1680 x 1050 pixels : on n'observe aucun problème important d'occlusion. . 39

1.23 2726 séquences sont affichées sur un écran de 15,4 pouces avec une résolution de 1680 x 1050 pixels : des problèmes importants d'occlusion apparaissent. . 39

2.1 (a) Réseau social *viadeo*. (b) Mise en évidence des contacts d'un premier individu. (c) Mise en évidence des contacts d'un second individu : ceux-ci se répartissent de façon plus homogène dans l'ensemble du réseau. 48

2.2 Capture d'écran des sept premiers résultats renvoyés par *Google* lorsque l'on cherche le mot *jaguar*. 49

2.3 Visualisation de pages retournées par un moteur de recherche avec la requête *jaguar* et regroupées selon le sujet qu'elles traitent (*clusters* jaunes). 50

2.4 Exemple de graphe de co-occurrence de mots-clés affiché avec l'algorithme *GEM* [72]. 51

2.5 Distribution des degrés des sommets d'un graphe sans échelle : l'axe des abscisses correspond aux degrés, celui des ordonnées au nombre de sommets. . . 52

2.6 (a) Graphe de co-occurrence initial. (b) Graphe après duplication des sommets. (c) Graphe après suppression des ponts. (d) Graphe avec groupes et ponts. . 54

2.7 Duplication du sommet rouge : le sommet rouge appartient à trois cliques (image du haut), il est ensuite dupliqué en trois sommets, chacun d'entre eux étant relié aux sommets de l'une des cliques (image du bas). 55

2.8 La création de deux communautés à partir de ce graphe fera du sommet rouge un sommet de la communauté de gauche. Ce choix est cependant contestable sémantiquement parlant. C'est pour cela que nous avons choisi de considérer ce type de sommets comme des ponts entre les communautés. 56

2.9 Graphe quotient biparti $Q(G) = (B, C, E_{Q(G)})$: (a) après ré-insertion des ponts, (b) après suppression des chevauchements. 61

2.10 Même graphe que celui représenté sur la figure 2.9 avec deux communautés « ouvertes » et la liste des pages internet affichée pour l'une des autres communautés. 63

2.11 Recherche du mot clé *jaguar* : (a) l'infobulle nous montre que la communauté est composée de mots-clés en relation avec le manga *Pyu to Fuku ! Jaguar*, (b) des liens hypertextes vers des pages internet apparaissent lorsque l'utilisateur effectue un clic droit, il est clair ici que la communauté ne contient que des pages sur les voitures. 65

2.12 Recherche du mot clé *CAC40* : (a) Le pont *Paris* relie la communauté *CAC40* et la communauté *crise financière*, (b) *AXA* est un pont entre les communautés *CAC40*, *Euronext/Paris* et *crise financière*. 66

2.13 Recherche du mot clé *Hepburn* : (a) vue d'ensemble du graphe, (b) composante connexe contenant les pages en liaison avec le *cinéma*. 67

2.14 Arbre de Porphyre 68

2.15 Différentes représentations d'arbres : (a) Tilford et Reingold [146], (b) Dendrogramme, (c) Eades [58], (d) Grivet *et al.* [88]. 69

2.16 *Map on temperance* : carte dessinée par Murell en 1846 [132] 70

2.17 *Fisheye* de Sarkar et Brown [158] : (a) graphe initial de villes des États-Unis, (b) même graphe où Saint-Louis a été sélectionné comme point d'intérêt. . . 71

2.18 *Cartogrammes* : (a) l'adjacence des pays est préservée [175], (b) les régions sont représentées par des rectangles [184]. 72

2.19 Des *Self-organizing maps* aux diagrammes de Voronoï : (a) méthode proposée par Skupin [167], (b) méthode proposée par Gansner *et al.* [97]. 72

2.20 *Themescapes* [200] 73

2.21 *Graph splatting* [185] : (a) graphe initial, (b) même graphe après traitement. 73

2.22 Dessin de graphe à l'aide d'une courbe fractale [130] 74

2.23 *Sunburst* [170] 75

2.24 (a) *Slice and Dice* [104], (b) *Squarified treemap* [35], (c) *Mixed treemaps* [191]. 76

2.25 *Strip layout* [18] (figure produite dans [191]) 76

2.26 (a) *Voronoï treemap* [11], (b) *City layout* [197]. 77

2.27 Premières étapes de la production d'une courbe de Hilbert. 78

2.28 Processus de création de la visualisation : (a) nous prenons un arbre en entrée, (b) nous en déduisons un ordre linéaire de ses feuilles, (c) nous plaçons ces feuilles le long d'une courbe fractale en fonction de cet ordre, (d) nous construisons les régions. 79

2.29 Premières étapes de la production d'une courbe de Gosper. 80

2.30 Placement des feuilles ordonnées le long de la courbe : (a) Feuilles positionnées en fonction de la courbe. (b) Régions correspondantes créées grâce à un diagramme de Voronoï. 81

2.31 Arbre initial où les sommets sont représentés par les régions qui leur correspondront après la création du diagramme Voronoï. 81

2.32 Préservation du principe d'imbrication des régions. 81

2.33 Hiérarchie du système de fichiers du logiciel tulip [6]. Les niveaux de l'arbre sont visibles grâce à un changement des tailles des frontières en fonction de la profondeur des sommets correspondants dans l'arbre et grâce à l'attribution d'une transparence aux régions. 82

2.34 (a) Graphe initial contenant les feuilles de l'arbre et les sommets des hexagones issus du diagramme de Voronoï. (b) Même graphe après avoir ajouté un chemin correspondant à la région violette $\{6, 7, 8\}$. 83

2.35 (a) Graphe dans lequel les chemins représentant toutes les frontières ont été ajoutés. (b) Même graphe où les sommets et les arêtes n'appartenant pas aux chemins représentant les frontières ont été supprimés. 83

2.36 Triangulation du graphe de la figure 2.35.a. 84

2.37 (a) Graphe de la figure 2.36 après l'algorithme de Tutte. (b) Même graphe où les sommets et les arêtes n'appartenant pas aux chemins représentant les frontières ont été supprimés. 84

2.38 Exemple du processus de création de la carte : (a) Arbre initial correspondant à la hiérarchie d'un système de fichiers. (b) Graphe triangulé positionné grâce à l'algorithme de Tutte. (c) Visualisation finale dans laquelle une texture a été appliquée le long des frontières. 85

2.39 Texture contenant une étiquette insérée dans un rectangle « courbé ». 86

2.40 Les sommets de l'axe médian peuvent être vus comme les centres de cercles inclus dans une région et étant en contact avec au moins deux points de la frontière de cette région. 86

2.41 Étiquettes des enfants de la région *plugins* : ces enfants appartiennent au troisième niveau de la hiérarchie. L'encadré montre les étiquettes placées sur les frontières des régions imbriquées. 87

2.42 ***Time*** : temps nécessaire à la génération d'une visualisation. ***Aspect ratio*** : ratio $longueur/largeur$ moyen des boîtes englobantes des régions. ***Area*** : combien l'aire d'une région correspond à la somme des tailles des feuilles de son sous-arbre. ***Stability*** : déplacement moyen des régions entre la carte de l'année précédente et la carte de l'année courante. Les écarts types sont notés en rouge. 89

2.43 *Geographical treemaps* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010 91

2.44 *Squarified treemaps* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010 91

2.45 *Sunbursts* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010 92

2.46 Visualisation du contenu de la bibliothèque standart *Python* de *Linux* : (a) Seul le premier niveau de l'arbre est affiché. (b) Les deux premiers niveaux de l'arbre sont affichés. (c) Zoom sur l'une des zones du graphe. 93

3.1 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$: les hyperarêtes sont représentées grâce à des cercles de couleur. 95

3.2 Exemple de représentation d'hypergraphe. 96

3.3 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Graphe biparti : les hyperarêtes sont représentées par les sommets de couleur et elles sont reliées par une arête aux représentations des sommets qu'elles contiennent. (b) Diagramme de Hasse : les hyperarêtes, représentées par les sommets de couleur, sont organisées en hiérarchie de façon à ce qu'une hyperarête contienne tous les sommets de ses hyperarêtes filles. 97

3.4 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Le graphe G représenté par les arêtes noires est un support de H : les sous-graphes induits par les hyperarêtes sont connexes. (b) Le graphe G représenté par les arêtes noires n'est pas un support de H car $G[\{1, 2\}]$ n'est pas connexe. 99

3.5 (a) Cycle. (b) Chemin. (c) Arbre. (d) Cactus. (e) Planaire extérieur. 100

3.6 (a) Planaire extérieur. (b) 2-planaire extérieur. (c) 3-planaire extérieur. . . . 101

3.7 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Hypergraphe induit $H[V \setminus \{4\}]$. (b) Hypergraphe restreint $H|(V \setminus \{4\})$. . 103

3.8 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Le sommet étiqueté 4 est un sommet d'articulation. (b) Les composantes connexes de $H|(V \setminus \{4\})$ sont les parties du sommet 4. 103

3.9 Arbre blocs-articulations, *i.e.* décomposition possible de l'hypergraphe $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. Les blocs sont entourés de cercles en pointillés rouges. 104

3.10 Exemples : les lignes continues noires représentent les supports, les lignes en pointillé représentent les hyperarêtes contenant plus de deux sommets. (a) Support planaire dont les blocs ne sont pas planaires. (b) Support planaire extérieur dont les blocs ne sont pas planaires. 106

3.11 Illustration de la preuve du lemme 3.3. (a) Support cactus de H. Les sommets se trouvant à l'intérieur de l'enveloppe en pointillé appartiennent à la partie W de v. Les sommets u_1 et u_2 sont proches de v. Les sommets x et y sont les sommets terminaux de p_C. (b) Support cactus de $H[W \cup \{v\}]$. 107

3.12 (a) Hypergraphe à intersections non fermées : les sommets 1 et 2 apparaissent dans deux régions séparées alors qu'ils appartiennent aux même hyperarêtes. (b) Hypergraphe à intersections fermées : l'hyperarête $\{1, 2\}$ a été ajoutée, un support de cet hypergraphe contiendra donc l'arête $(1, 2)$ et les deux sommets seront connexes sur le diagramme d'Euler. 108

3.13 (a) Hypergraphe à différences non fermées : les sommets 1 et 2 apparaissent dans deux régions séparées alors qu'ils appartiennent exactement à la même hyperarête. (b) Hypergraphe à différences fermées : l'hyperarête $\{1, 2\}$ a été ajoutée, un support de cet hypergraphe contiendra donc l'arête $(1, 2)$ et les deux sommets seront connexes sur le diagramme d'Euler. 109

3.14 Illustration de la preuve du lemme 3.5. 111

3.15 Plan du métro d'Amsterdam. 113

3.16 $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6, 8\}, \{5, 7\}\}$. (a) Diagramme d'Euler représentant l'hypergraphe. (b) Les arêtes noires représentent un support G non basé sur des chemins ($G[\{4, 5, 6, 8\}]$ ne contient pas de chemin hamiltonien). (c) Les arêtes noires représentent un support G basé sur des chemins. (d) Représentation de l'hypergraphe selon la métaphore des cartes de métro. 114

3.17 $H = (V, A)$ avec $V = \{1, 2, 3\}$ et $A = \{\{1, 2, 3\}, \{1, 3\}\}$. (a) Support non minimal et sa représentation selon la métaphore des cartes de métro. (b) Support minimal et sa représentation selon la métaphore des cartes de métro. 114

3.18 Représentation du graphe biparti associé à notre hypergraphe de départ. Les triangles représentent les films de Jean-Pierre Jeunet. Les cercles de couleur représentent les acteurs ayant joué dans un seul film. Les cercles gris représentent les acteurs ayant joué dans plusieurs films. 117

3.19 Représentation de l'hypergraphe générée par notre algorithme. (a) Ensemble de la représentation. (b) Sous-partie de la représentation. 117

Liste des tableaux

1.1 Ce tableau résume le nombre des participants ainsi que leurs situations respectives. Afin de mesurer le temps d'apprentissage, nous avons vérifié qu'aucun des participants n'avait auparavant utilisé notre système ou un autre système similaire. 37

1.2 Variables dépendantes. 40

1.3 Fonctionnalités évaluées. 40

1.4 Moyennes des notes (sur 10) attribuées au système de visualisation (techniques de représentation et de navigation). 41

1.5 Moyennes des notes (sur 10) attribuées au système de visualisation (abstraction des données et des opérations). 43

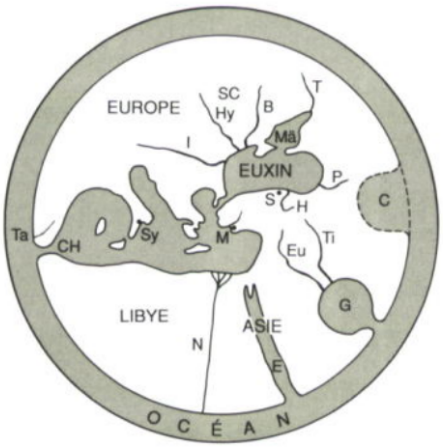
2.1 Récapitulatif des propriétés d'affichage en fonction des différentes techniques mentionnées plus haut : un X signifie que la propriété est complètement respectée, un / signifie qu'elle est partiellement respectée et une case vide signifie qu'elle ne l'est pas du tout. 77

Introduction

La visualisation de l'information est une discipline qui s'est fortement développée ces vingt dernières années. Cependant l'utilisation de graphiques afin de représenter des données est une méthode utilisée depuis des millénaires [1] (le plus vieux système connu est l'écriture cunéiforme développée en Mésopotamie il y a plus de 3000 ans). Rapprochons-nous - un peu - de notre époque et prenons un exemple marquant une évolution significative de la visualisation d'information. Cet exemple nous est fourni par le philosophe Marcel Conche qui a mené un remarquable travail sur l'étude de la pensée d'Anaximandre [3].

Au début du sixième siècle avant notre ère, la cité ionienne de Milet, bien qu'elle soit victime de conflits internes, est la ville grecque d'Asie la plus prospère grâce à ses nombreuses colonies le long de la mer Noire et sa puissance maritime. L'essor de la cité n'est pas seulement marqué par son économie dynamique mais aussi par son école fondée par Thalès, personnage considéré comme le premier philosophe, scientifique et mathématicien grec. La position stratégique de la ville n'est certainement pas étrangère à ce mouvement. En tant que puissance maritime, la cité reçoit non seulement des quantités de produits provenant des villes environnantes mais aussi des quantités d'informations, en particulier des données géographiques. Dans ce contexte, l'école s'applique en partie à réfléchir à la portée que ces informations peuvent avoir sur le développement de la cité. D'une part, elles jouent un rôle clé pour la navigation et le commerce ; d'autre part, elles permettent de mesurer la menace exercée par les Perses sur les cités ioniennes. Il semblerait que Anaximandre de Milet, successeur de Thalès comme maître de l'école milésienne et acteur majeur du développement de la pensée occidentale [189], ait bien saisi la portée et les enjeux de la maîtrise de l'information. Il fut, d'après Thémistios, le premier des Grecs connus à publier un ouvrage écrit sur la nature. On peut donc le considérer comme le premier philosophe à avoir consigné ses travaux par écrit, et l'on sait bien l'importance que peut avoir l'utilisation de la graphique sur le développement de la pensée grâce aux travaux de Goody [84]. Anaximandre, semble t-il, a donc saisi l'intérêt des représentations graphiques : en plus des textes, d'après le géographe antique Strabon, Erothastène affirme que Anaximandre est aussi l'auteur de la première carte du monde à être publiée (cette carte ne nous est pas parvenue mais Conche en propose une reconstitution illustrée ici figure 0.1).

1. Le lecteur intéressé trouvera dans les livres de Tufte [178, 179, 180] un très grand nombre de représentations graphiques d'époques variées.



I = Istros (Danube). — Hy = Hypanis (Boug). — B = Borysthènes (Dniepr). — T = Tanais (Don). — P = Phase Fl. — H = Halys. — Eu = Euphrate. — Ti = Tigre. — N = Nil. — Ta = Tartessos (Guadalquivir). — CH = Colonnes d'Hercule. — C = mer Caspienne. — G = golfe Persique. — Mä = Palus Méotide (mer d'Azov). — E = mer Erythrée (mer Rouge). — S = Sinope. — Sy = Sybaris. — SC = Scythie.

FIGURE 0.1: Ce que pouvait représenter la carte d'Anaximandre, d'après Marcel Conche [3]. La ville de Milet y est repérable grâce à un M.

Bien entendu, des cartes locales ont déjà été produites en Égypte, en Lydie, au Moyen-Orient ou à Babylone. Cependant, l'originalité de celle d'Anaximandre réside dans l'étendue des territoires couverts [2] et les raisons de sa production. Alors que les cartes précédentes permettaient seulement de visualiser un territoire de faible étendue afin de faciliter les déplacements dans celui-ci, la carte d'Anaximandre est issue d'une démarche beaucoup plus complexe. Celui-ci ne se contente pas de retranscrire des informations géographiques à des fins pratiques : la représentation graphique est pour lui un moyen de diffusion de connaissances aussi bien politiques que philosophiques. D'après Marcel Conche, la carte est le fruit d'un travail dont les motivations sont diverses :

- Motivations économiques : la navigation et le commerce constituent l'une des principales ressources de Milet ; dans ce contexte, une carte de toutes les colonies et de toutes les autres cités partenaires présente de nombreux avantages.
- Motivations politiques : la carte permet de visualiser l'isolement des cités ioniennes dans une Anatolie dominée par les Perses. Peut-être Thalès aurait-il eu plus de facilités à convaincre les Grecs de se fédérer afin d'assurer leur indépendance face à ces belliqueux voisins s'il avait utilisé une carte, nous dit le géographe antique Strabon.
- Motivations philosophiques : une carte représentant toutes les caractéristiques géographiques du monde connu de l'époque constitue un apport bénéfique à la connaissance, ce qui aurait justifié sa production même si elle n'avait eu aucun intérêt pratique.

2. La carte d'Anaximandre est connue comme étant la première mappemonde. La fameuse carte du monde babylonienne conservée au *British museum* date aussi de cette époque. Cependant, elle n'était pas destinée à la transmission d'un savoir mais plutôt à une propagande politique comme le montre Marcel Conche.

La carte d'Anaximandre est un exemple frappant de ce que la visualisation d'information peut nous apporter. Elle permet d'ordonner et de transmettre des connaissances sur le monde (exemple des motivations philosophiques), elle permet d'accéder aux données qui y sont représentées (exemple des motivations économiques) et elle permet de découvrir de l'information (exemple des motivations politiques). Ces observations nous permettent de comprendre pourquoi la phrase :

« *A picture is worth ten thousand words.* » (voir [40] qui suggère l'origine de cette expression populaire)

a connu un tel succès dans la communauté de notre domaine : la visualisation d'information sert premièrement à communiquer une grande quantité d'information. Mais ces observations permettent aussi de mettre en relief la capacité que peut avoir une représentation à nous permettre de découvrir de l'information (dans l'exemple, la position catastrophique des cités vis-à-vis des Mèdes, la nécessité de la création d'une fédération basée à Téos, ville située au centre de l'Ionie). Ceci nous amène à un second principe bien connu de notre communauté (*i.e.* la communauté de la visualisation de l'information) :

« *Using Vision to think.* » [40]

Maintenant que nous avons illustré une application de la visualisation de l'information par un exemple et que nous avons, grâce à cet exemple, illustré deux des motivations premières du domaine, nous pouvons donner une première réponse à la question : « Qu'est-ce que la visualisation de l'information ? ». Nous en donnons ici une définition générale, une étude plus détaillée reprenant des définitions trouvées dans la littérature sera ensuite donnée à la fin du chapitre 1 :

Visualisation d'information : présentation visuelle de données abstraites.

Depuis les années 90, grâce aux nouvelles technologies de l'information et de la communication, la quantité de données disponibles n'a cessé d'augmenter de façon exponentielle. Parallèlement, un domaine de recherche s'est développé ayant pour objet l'étude des techniques de représentation de ces données : c'est le domaine de la visualisation d'information qui nous intéresse ici. Le vif intérêt que lui porte le monde de la recherche est d'ailleurs mis en évidence par l'apparition de nombreuses conférences sur le sujet. On en vient donc naturellement à se poser la question : existe-t'il une science de la visualisation de l'information ? Voici ce qu'en pensent Ward *et al.* [193] :

« *We do not yet have a science of visualization. There have been attempts automating the process : given data, automatically generate a visualization.* »

Cette phrase peut être interprétée comme une distinction entre ce qui relève du domaine de l'art [3] et ce qui relève du domaine de la science. Produire une visualisation permettant l'accès à des informations contenues dans un ensemble particulier de données relève de l'art. Une science de la visualisation consisterait quant à elle à découvrir les

3. Le mot art est ici employé selon le sens A « du Lalande » [121] : « En général, ensemble de procédés servant à produire un certain résultat [...]. L'art s'oppose en ce sens : 1. à la science conçue comme pure connaissance indépendante de ses applications ; 2. à la nature, conçue comme puissance produisant sans réflexion. ». Ainsi défini, sa signification est très proche de celle du terme anglais *design*, couramment employé en visualisation d'information.

règles universelles régissant n'importe quelle production de ce type [4]. Pour faire l'analogie avec l'écriture, la science correspondante est connue sous le nom de linguistique, et on voit bien qu'un romancier ne pratique pas la même activité qu'un linguiste. Le premier utilise un système pour transmettre des informations, le second essaie de découvrir comment fonctionne ce système. Une telle approche en visualisation a fait l'objet de quelques études (nous y reviendrons plus tard) mais reste cependant marginale, ce qui peut sembler naturel : la linguistique s'est développée bien après l'apparition des premiers textes.

Comme le précise Saussure [54], le père de la linguistique moderne, la linguistique n'est qu'un sous-ensemble d'une science étudiant l'ensemble des systèmes de signes : la sémiologie ou sémiotique. Une science de la visualisation n'aurait-elle pas quelques analogies avec certains sous-domaines de cette discipline ? En particulier, qu'est donc cette « sémiologie graphique » dont nous parle Jacques Bertin [23, 22] et qui semble être un auteur de référence dans la communauté de la visualisation de l'information ?

Le chapitre 1 de cette thèse s'attachera à poser un cadre formel et traitera d'une approche sémiotique de la visualisation d'information. Des exemples nous aideront à bien ancrer les idées développées par les sémioticiens dans ce contexte. La thèse décrira ensuite nos travaux sur la visualisation de données relationnelles, modélisées sous forme de graphes (chapitre 2), puis d'hypergraphes (chapitre 3).

4. La citation de Ward *et al.* suggère même que ces règles soient assez précises pour pouvoir automatiser le processus de production. Comme nous le verrons au début du premier chapitre, notre position concernant une science de la visualisation d'information est moins ambitieuse.

Chapitre 1

Une approche sémiotique de la visualisation d'information

Le concept de science des signes apparaît simultanément en Europe et aux États-Unis à la fin du siècle dernier, sans qu'aucun des deux pionniers n'ait vraisemblablement connu les travaux de son homologue d'outre-Atlantique. L'européen Ferdinand de Saussure définit la sémiologie comme « la science générale de tous les systèmes de signes (ou de symboles) grâce auxquels les hommes communiquent entre eux » [54]. Parallèlement, Charles Sanders Peirce propose le concept de sémiotique qu'il définit comme « la doctrine quasi-nécessaire ou formelle des signes » [139]. La différence de nos jours entre sémiotique et sémiologie reste ambiguë et souvent différente selon les chercheurs [114]. Nous utiliserons ici le mot sémiotique, choisi de façon arbitraire, pour désigner la science qui étudie les systèmes de signes, c'est-à-dire la science qui recherche les lois communes à tous ces systèmes.

Comme le remarque Anne Hénault [98], la sémiotique est constituée de sémiotiques « réduites » comme la langue française ou le morse, c'est-à-dire des systèmes de signes ayant des lois de construction spécifiques. En suivant Jean-Marie Klinkenberg [114], nous utiliserons la terminologie suivante :

- **La sémiotique générale** a pour sujet d'étude les lois directrices de n'importe quel système de signe.
- **Une sémiotique particulière** étudie un système de signes particulier afin de découvrir les lois qui lui sont propres.
- **Une sémiotique appliquée** consiste, comme son nom l'indique, à appliquer à des objets particuliers les règles d'une sémiotique particulière. Cette branche de la sémiotique relève donc de l'art, et non de la science comme les deux autres.

On se doit de remarquer ici l'ambiguïté du terme « appliqué » dans notre terminologie. En effet, il n'est pas nécessaire de définir d'abord les règles d'une sémiotique particulière pour les appliquer ensuite à la production d'énoncés [1]. Nous pouvons même remarquer que c'est grâce à l'analyse des énoncés issus de la sémiotique appliquée que l'on peut découvrir les lois de la sémiotique particulière correspondante. C'est pour cette raison que des textes sont apparus bien avant la linguistique.

1. Un énoncé correspond ici à un ensemble cohérent de signes porteurs d'un message. Par exemple, un livre peut-être vu comme un énoncé. Cette notion sera précisée dans la section 1.2.3.1

Essayons maintenant d'appliquer la terminologie évoquée ci-dessus au domaine de la visualisation de l'information. Le cartographe [2] se doit de suivre des règles générales, valables pour la production de n'importe quel énoncé. Ces lois appartiennent donc à une sémiotique particulière, et c'est bien elle que nous avons nommée **science de la visualisation de l'information** en introduction. Cette notion doit être complétée par celle d'**art de la visualisation de l'information**, qui est la sémiotique appliquée de la science de la visualisation de l'information et qui consiste à la production de cartes [3] porteuses d'informations spécifiques.

Comme nous l'avons évoqué en introduction, les auteurs de [193] considèrent que les règles énoncées par une science de la visualisation de l'information nous permettraient de générer des cartes automatiquement à partir de données. Nous sommes ici moins ambitieux dans la définition de cette science. En effet, si on la compare avec la linguistique, on s'aperçoit que même si cette dernière n'a pas réussi à définir des processus de génération automatique de textes à partir de données diverses, ce n'est par pour autant qu'on ne la définit pas comme une science [86]. L'intervention humaine d'un écrivain [4] est cependant toujours nécessaire pour appliquer les lois de la linguistique à la production de textes. Donc pour nous, la science de la visualisation de l'information consiste simplement à rechercher des règles applicables à la production de n'importe quelle carte. Cette approche ne rejette pas la possibilité d'inclure aux représentations une dimension artistique.

Un autre argument en faveur de cette conception moins ambitieuse de la science de la visualisation nous est fournie par la sémioticien Umberto Eco [60]. D'après lui, des énoncés possédant une dimension artistique sont ouverts, c'est-à-dire qu'ils peuvent être source d'interprétations différentes et ainsi véhiculer d'autres informations que celles introduites par leurs auteurs. Une visualisation, en plus de communiquer des informations, a aussi pour objectif de permettre d'en découvrir de nouvelles. En d'autres termes, une carte doit montrer à la personne qui la regarde non seulement des informations connues du cartographe, mais aussi d'autres informations qu'elle aura permis de révéler. Ainsi, en suivant Eco, il semblerait que cette objectif ne soit réalisable qu'en faisant appel à une dimension artistique. Dans ce cas, une science de la visualisation décrivant des processus de production entièrement automatisés ne serait pas seulement un objectif trop ambitieux comme nous l'avons suggéré ci-dessus, mais aussi limiterait l'efficacité des représentations qu'elle permettrait de produire.

On ne peut donc pas dire que cette science n'existe pas encore. En effet, certaines règles de construction et de validation ont déjà été proposées comme le modèle imbriqué

2. Le terme cartographe est employé pour signifier un individu producteur d'énoncés non linguistiques et non artistiques, composés de signes graphiques générés grâce au support informatique, et portant sur des données abstraites. Dans ce sens, les écrivains et les producteurs d'art ne sont pas des cartographes (contrairement au sens employé au cours de l'exposé, *cf.* note 3, le mot art est ici employé selon le sens B [121] : L'« Art ou les Arts désignent toute production de la beauté par les œuvres d'un être conscient. »). En revanche, la notion englobe tous les producteurs d'énoncés de visualisation de l'information.

3. Carte doit être compris ici comme énoncé issu d'un processus de visualisation de l'information, donc produit par un cartographe.

4. Le mot écrivain est employé ici comme producteur d'énoncés linguistiques et non comme romancier. Il est donc à la linguistique ce qu'est le cartographe à la visualisation de l'information : il met en application une sémiotique particulière.

de Tamara Munzner [131] sur lequel nous reviendrons plus tard. En revanche, il est clair que la majorité des publications du domaine concernent la production de cartes (art) et non la découverte des lois générales de production (science). Ceci paraît normal au vu de la linguistique. En effet, comme nous l'avons déjà vu plus haut, des textes ont d'abord été produits (art) et c'est plus tard qu'une science a été mise en place. Il est donc possible que la visualisation de l'information suive le même processus.

La sémiotique a pour objet d'étude le signe. C'est pourquoi dans ce chapitre, nous allons commencer par nous pencher plus en détail sur cette notion de signe. Cette étude nous permettra d'extraire certaines règles spécifiques à la science de la visualisation de l'information. Notre objectif étant de trouver les bonnes pratiques qui permettront au cartographe de limiter les erreurs que peuvent contenir ses énoncés, nous discuterons dans un second temps des différents résultats obtenus le long de la description du signe en les mettant en parallèle avec l'un des modèles les plus aboutis dans la production et la validation de cartes : le modèle imbriqué de Tamara Munzner [131] déjà mentionné ci-dessus qui donne un processus permettant de valider *a priori* et *a posteriori* les étapes de la production des cartes.

L'exposé présenté ici a été rédigé dans le cadre d'une thèse CIFRE unissant le LaBRI et la société Pikko, spécialisée dans la création de cartes. Les questions soulevées, outre leur importance théorique et pratique dans le domaine de la recherche, sont aussi primordiales pour une entreprise comme Pikko qui milite pour l'adoption de la cartographie d'info auprès de ses clients : une démarche scientifique dans la production de ses cartes est absolument nécessaire afin qu'elle puisse proposer à ses clients des solutions efficaces.

Concrètement, nous allons commencer ce chapitre en décrivant une application de visualisation de l'information réalisée au cours de la thèse (section 1.1). Cet exemple sera ensuite utilisé comme révélateur de la sémiotique et du signe comme notion fondamentale à la visualisation d'information (section 1.2). Enfin, nous confronterons nos résultats à ceux déjà en pratique dans la domaine de la cartographie d'information, et en particulier au modèle de Munzner (section 1.3).

Tout au long de cette section, nous définirons certains concepts utilisés par les sémioticiens et utiles pour notre exposé. Cependant, si le lecteur désire obtenir des définitions plus complètes, il pourra se reporter au dictionnaire des concepts de sémiotique proposé par Greimas [86]. Des introductions simples et assez complètes de la sémiotique ont aussi été proposées par Eco [61] et Klinkenberg [114]. Ces deux livres pourraient aider à la compréhension des sections suivantes. Pour aller plus loin, le livre « Sémiotique et philosophie du langage » [62], plus difficilement abordable que les précédents, présente une analyse de concepts clés de la sémiotique à travers les débats qui se sont construits autour d'eux. Enfin, une introduction difficile mais exhaustive de la linguistique et de la philosophie du langage, dont beaucoup de concepts sont ou pourraient être appliqués à la sémiotique générale, a été proposée par Auroux *et al.* [9].

1.1 *SequencesViewer* : un outil de visualisation de séquences de gènes

Cette section est dédiée à la présentation d'un système de visualisation d'information réalisé au cours de la thèse en collaboration avec Nicolas Pecheur, Sandra Bringay et Mathieu Roche du LIRMM [5] et Maguelonne Teisseire du CEMAGREF [6] [150]. Les algorithmes ne seront pas détaillés car le système n'est décrit que pour illustrer l'approche sémiotique de ce premier chapitre. Si toutefois le lecteur est intéressé, une description de la visualisation des groupes (*cf.* ci-dessous) est disponible [149]. Une version décrivant l'ensemble du système et des algorithmes implémentés ainsi qu'une évaluation a aussi été publiée [151]. Il est nécessaire de mentionner ici que l'approche descriptive utilisée ci-dessous ne correspond en rien aux étapes du processus de production du modèle de Munzner évoqué précédemment et sur lequel nous reviendrons dans la section 1.3.2.

L'application que nous décrivons a été élaborée en collaboration avec des biologistes du MMDN [7] qui s'intéressent à la maladie d'Alzheimer. Afin de mener correctement leur recherche, ceux-ci doivent analyser les mesures d'expression de milliers de gènes calculées par des puces ADN comme l'Affymetrix U-133 plus 2,0 qui peut mesurer 54 675 valeurs numériques. Inutile de préciser ici les limites d'une représentation de telles quantités de données dans un tableur. C'est pourquoi les biologistes ont besoin, dans un premier temps, de techniques de gestion de connaissances afin de structurer leurs données. Commençons donc par nous intéresser aux données fournies par les puces et les méthodes mises en place pour les structurer. Nous pourrons alors montrer comment un système de visualisation peut aider les biologistes à accéder aux informations ainsi ordonnées.

Dans les parties concernant *SequencesViewer*, certains mots ou expressions seront définis et apparaîtront lors de leur première utilisation en gras. La définition donnée sera valable dès que nous traiterons de ce système dans le présent chapitre. D'autre part, l'application peut être décrite de façon plus générique. Cependant, afin de faciliter la clarté de notre exposé, et compte tenu du fait que la description du système ne constitue qu'un prétexte à l'étude sémiotique qui suivra, nous avons simplifié certaines définitions en les limitant au cas d'un exemple basé sur la maladie d'Alzheimer.

1.1.1 Données manipulées

Grâce à l'algorithme BDSAP [153], les gènes et leur niveau d'expression extraits par une puce ADN peuvent être ordonnés en motifs séquentiels. Une **séquence** se présente sous la forme $< (a, b, c)(d)(e, f) >$ où a, b, c, d, e, f sont des **gènes**. Les gènes sont groupés en **ensembles de gènes** de même expression. Dans notre exemple, a ,b et c ont la même mesure d'expression et sont groupés dans l'ensemble (a, b, c). Une séquence est donc une liste ordonnée d'ensemble de gènes telle que la mesure d'expression est décroissante. Dans notre exemple, la valeur d'expression de a, b ou c est supérieure à celle de d qui est elle-même supérieure à celles de e et f.

5. http ://www.lirmm.fr/
6. http ://tetis.teledetection.fr/
7. http ://www.mmdn.univ-montp2.fr/

Une propriété **sain** (resp. **malade**) est associée à chaque séquence. Elle correspond au nombre d'individus sains (resp. malades) sur l'ensemble des personnes porteuses de la séquence.

L'indice $S2MP$ [155], basé sur le nombre de gènes communs entre deux séquences ainsi que sur l'ordre des ensembles de gènes, permet d'évaluer le degré de similarité entre chaque paire de séquences. Ces valeurs seront appelées **distances** dans la suite de notre exposé. Elles permettent de fragmenter l'ensemble des séquences en groupes grâce à l'algorithme de *clustering k-means*. De plus, cet algorithme permet d'extraire une séquence représentative pour chaque groupe que nous nommerons **centre** de ce groupe.

Les séquences sont ordonnées dans une **hiérarchie** selon la méthode proposée dans [136]. Chaque feuille de l'arbre correspondant à cette hiérarchie est une séquence. Les autres sommets sont quant à eux constitués des ensembles de gènes démarrant la séquence de leurs enfants. La figure 1.1 nous donne un exemple d'une telle hiérarchie.

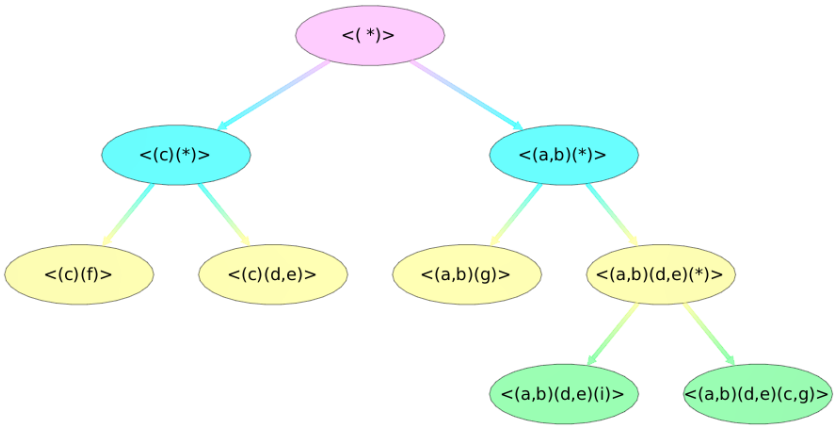


FIGURE 1.1: Exemple de hiérarchie d'une ensemble de 5 séquences de gènes.

Enfin, chaque séquence est associée à des articles obtenus en interrogeant *Pubmed*[8] avec les noms des gènes de la séquence et les synonymes de ces noms lorsque le nombre de documents est limité [153]. Comme pour les paires de séquences, une **distance** est attribuée à chaque paire de séquence-document, de façon à ce qu'un article dont la distance avec une séquence est faible soit plus pertinent qu'un autre dont la distance est plus grande.

1.1.2 Système

Les deux structures créées grâce à l'algorithme décrit ci-dessus (groupes de séquences et hiérarchie) sont difficilement exploitables par les biologistes telles quelles. L'utilisateur a en effet besoin d'interfaces graphiques qui lui permettent de naviguer facilement dans ces structures. C'est maintenant que le visualisation entre en jeu. L'utilisateur doit pouvoir naviguer intuitivement dans les groupes de séquences aussi bien que dans la hiérarchie. Il doit aussi pouvoir rechercher facilement des séquences, des ensembles de gènes, des gènes

8. http ://www.ncbi.nlm.nih.gov/pubmed/

et afficher les valeurs des propriétés sain et malade afin de pouvoir isoler les séquences intéressantes qui pourraient avoir un lien avec la maladie. Enfin, afin d'éviter au biologiste une perte de temps inutile, l'application doit permettre d'accéder aux articles *Pubmed* directement à partir d'une séquence sans avoir à copier cette séquence dans le moteur de recherche de *Pubmed*.

Au vu de ce qui vient d'être mentionné, nous avons décidé de développer notre application en Flash. D'une part, cette technologie permet de créer facilement des applications esthétiquement attractives. D'autre part, il est facile de l'intégrer à une page internet afin de la rendre accessible de n'importe où, ce qui nous a été demandé par les utilisateurs. La figure 1.2 montre une vue d'ensemble du système. Nous allons maintenant décrire la partie visualisation (en rouge dans le schéma).

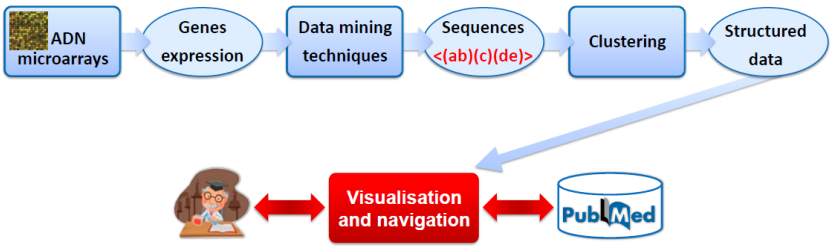


FIGURE 1.2: Vue d'ensemble du système. Notre application correspond à la partie *Visualisation and navigation*.

SequencesViewer est un système de visualisation et de navigation basé sur trois types de représentations (nuage de points, système solaire et *treemap*). Nous allons maintenant voir à quoi servent ces représentations et comment elles s'articulent entre elles.

1.1.2.1 Nuage de points

Le nuage de points (voir figure 1.3) affiche les groupes de séquences. Il permet donc de donner aux biologistes une vision d'ensemble des groupes, des distances entre les centres des groupes et entre les séquences d'un groupe avec leur propre centre.

Les centres sont d'abord placés sur le plan de façon à ce que la distance euclidienne entre chaque paire de centres reflète au mieux la distance calculée grâce à la mesure $S2MP$. Pour cela, nous utilisons une méthode dite de *Multidimensional scaling* [34] présentée dans l'article [79]. Ensuite, nous supprimons les chevauchements des centres grâce à la méthode présentée dans l'article [76]. Enfin, les autres séquences sont placées en cercle autour du centre de leur groupe de façon à ce que leur distance euclidienne avec ce centre représente la distance calculée grâce à la mesure $S2MP$.

L'utilisateur peut choisir de visualiser les centres (figure 1.4) ou les centres avec leurs séquences (figure 1.3) grâce à une case à cocher étiquetée *Sequences*. L'intensité de la couleur des centres reflète le nombre de séquences que contient le groupe correspondant, la légende permet à l'utilisateur d'évaluer ce nombre.

L'utilisateur peut aussi rechercher un ou plusieurs gènes en utilisant le champ *Search Item* de la légende. Dans ce cas, toutes les séquences contenant ce(s) gène(s) sont affichées en vert comme on peut le voir dans la figure 1.5.

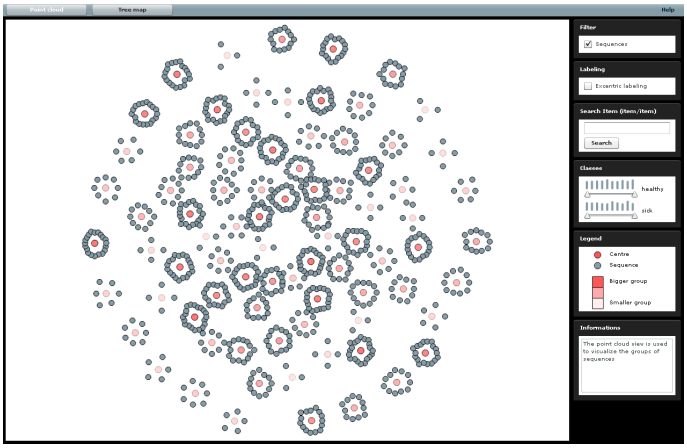


FIGURE 1.3: Nuage de points : représentation globale des groupes de séquences.

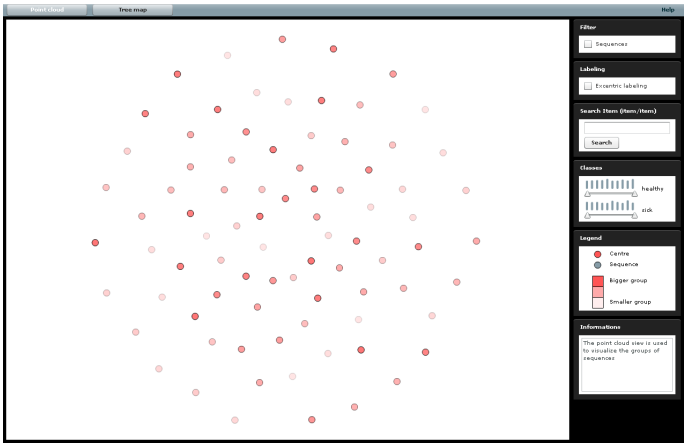


FIGURE 1.4: Nuage de points dont les séquences qui ne sont pas des centres ont été masquées.

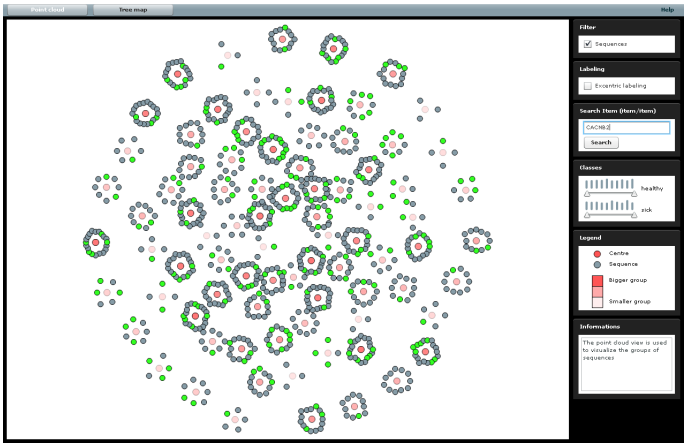


FIGURE 1.5: Nuage de points : les séquence vertes contiennent un gène recherché par l'utilisateur.

Il peut aussi déplacer la carte à l'aide de la souris ou changer le niveau de zoom grâce à la molette. De plus, une bulle accessible grâce au clic droit de la souris permet d'accéder aux informations de la séquence survolée (figure 1.6).

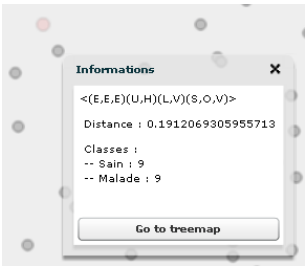


FIGURE 1.6: Lorsque l'utilisateur effectue un clic droit sur une séquence, une bulle contenant les informations de la séquences apparaît.

Des ascenseurs *healthy* et *sick* permettent de sélectionner une fourchette de valeurs des propriétés sain et malade. L'utilisateur peut s'en servir pour filtrer les séquences qui n'appartiennent pas à un certain intervalle (voir figure 1.7). L'histogramme placé au-dessus des ascenseurs permet d'évaluer le nombre de séquences ayant la valeur indiquée par les ascenseurs. Cette idée nous vient de l'article [199].

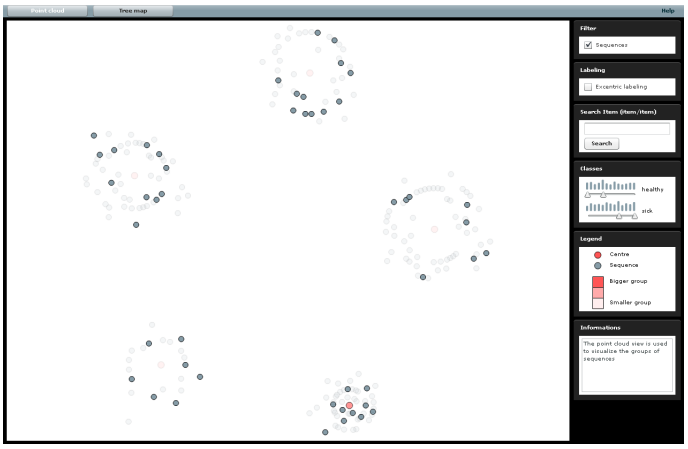


FIGURE 1.7: Les ascenseurs permettent de filtrer les séquences selon la valeur de leurs propriétés sain et malade.

Enfin, nous avons muni notre système d'une technique d'étiquetage excentré [69]. Quand l'utilisateur coche la case *Labelling* et qu'il clique sur un espace vide de la visualisation, les étiquettes des séquences positionnées à l'intérieur d'un cercle entourant le point sur lequel il a cliqué sont affichées (voir figure 1.8). Afin d'empêcher le chevauchement des étiquettes, celles-ci sont placées en périphérie de la carte : des couleurs et des lignes sont utilisées afin que l'utilisateur puisse faire le lien entre un séquence et son étiquette.

1.1.2.2 Systèmes solaires

Lorsque l'utilisateur double-clique sur une séquence dans le nuage de points, il accède à une seconde vue (figure 1.9) basée sur la métaphore du système solaire [135]. Ici, le

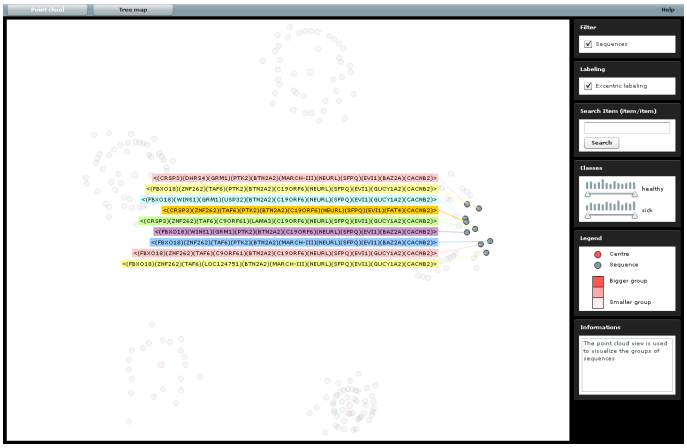


FIGURE 1.8: Étiquetage excentré des séquences.

centre du groupe de la séquence sélectionnée vient se placer au centre de la vue et les séquences de ce même groupe sont disposées autour de lui en fonction de leur distance. Cette vue permet donc de visualiser de façon plus précise un groupe.

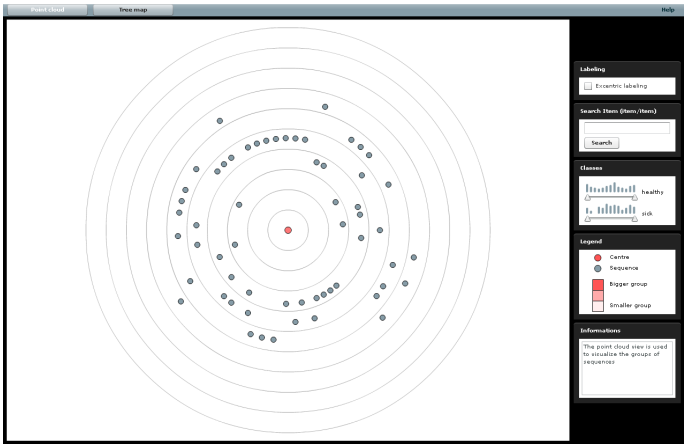


FIGURE 1.9: Visualisation détaillée d'un groupe.

Comme dans la vue précédente, l'utilisateur a la possibilité de déplacer la carte, de changer le niveau de zoom, d'effectuer une recherche de gènes, de filtrer les séquences selon les valeurs de leurs propriétés sain et malade ou d'afficher une bulle d'informations sur une séquence. La légende est aussi toujours disponible, ainsi que l'étiquetage excentré.

Une seconde vue basée sur la métaphore du système solaire est accessible à partir de la première en double-cliquant sur une séquence (figure 1.10). Dans cette nouvelle représentation, la séquence sélectionnée vient se placer au centre de l'écran et des carrés représentant les articles extraits de *Pubmed* la concernant sont disposés autour d'elle en fonction de leur distance par rapport à elle. L'intensité du bleu des documents correspond à leur date de publication. Ici aussi, une bulle d'information peut être ouverte en cliquant sur le bouton droit de la souris. Le zoom et le déplacement de la carte sont aussi disponibles. Enfin, lorsque l'utilisateur double-clique sur l'un des documents, la page

internet correspondant de *Pubmed* est ouverte dans un nouvel onglet.

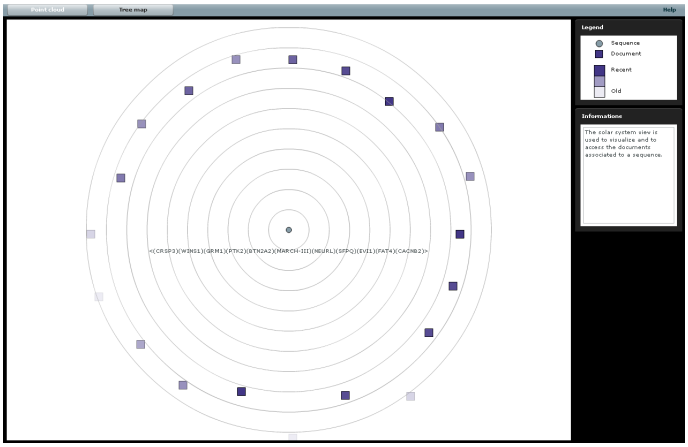


FIGURE 1.10: Visualisation d'une séquence avec les documents qui lui sont associés.

La figure 1.11 montre le système de navigation permettant de passer d'une de ces trois premières vues à une autre. Mise à part les opérations mentionnées ci-dessus, on peut aussi passer du deuxième nuage de point au premier en double-cliquant sur la séquence placée au centre. L'utilisateur peut aussi revenir à tout moment au nuage de point en cliquant sur le bouton *Point Cloud* situé en haut de l'application.

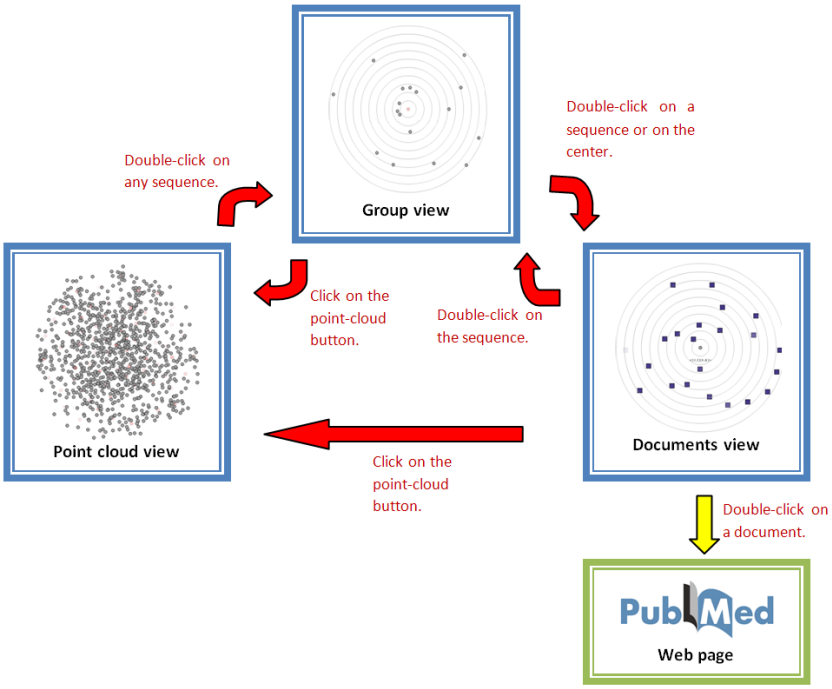


FIGURE 1.11: Navigation à travers les groupes de séquences.

1.1.2.3 *Treemap*

Comme nous l'avons mentionné lorsque nous avons décrit les données, les séquences ont aussi été organisées hiérarchiquement. Or, dans les visualisations décrites ci-dessus, nous n'avons pas représenté cette hiérarchie. C'est pourquoi nous combinons les vues précédentes avec une nouvelle permettant de représenter l'arbre des séquences correspondant (rappelez-vous la figure 1.1).

La manière la plus intuitive de visualiser un arbre est d'utiliser une représentation dite « nœud-lien » très utilisée dans le dessin de graphe (par exemple [146]). Cependant, cette approche nécessite beaucoup d'espace et il est difficile de visualiser de très grands ensembles de données. C'est pourquoi nous avons privilégié une approche *treemap* [104].

Selon cette approche, tous les sommets sont représentés par des rectangles et l'espace est divisé successivement de façon à ce qu'un sommet dans la hiérarchie contienne les rectangles représentant ses enfants. Dans notre implémentation, nous avons utilisé l'algorithme présenté dans l'article [35] car il essaie de limiter l'apparition de longs rectangles difficiles à comparer visuellement.

La figure 1.12 montre un *treemap* issu de notre visualisation. Nous avons décidé de limiter le nombre de niveaux affichés à trois : le niveau courant, ses enfants et les valeurs de leurs propriétés sain et malade (cette valeur correspond à l'aire des rectangles correspondants). Une case à cocher *Classes* permet d'afficher un troisième niveau dans la hiérarchie plutôt que les propriétés sain et malade (figure 1.13).

FIGURE 1.12: *Treemap* : trois niveaux sont affichés, la racine, ses enfants et les propriétés sain et malade de ses enfants.

Lorsque l'utilisateur double-clique sur un sommet, celui-ci vient prendre la place du sommet le plus élevé dans la hiérarchie affichée et ses enfants et leurs enfants ou leurs propriétés sont affichées. On peut ainsi s'enfoncer petit à petit dans la hiérarchie en sélectionnant les séquences qui nous intéressent. A tout moment, on peut aussi remonter dans la hiérarchie grâce au chemin allant de la racine au sommet le plus élevé affiché à l'écran. Il suffit pour cela de cliquer sur le sommet du chemin vers lequel on désire s'élever. Le chemin est affiché en haut de l'écran 1.14.

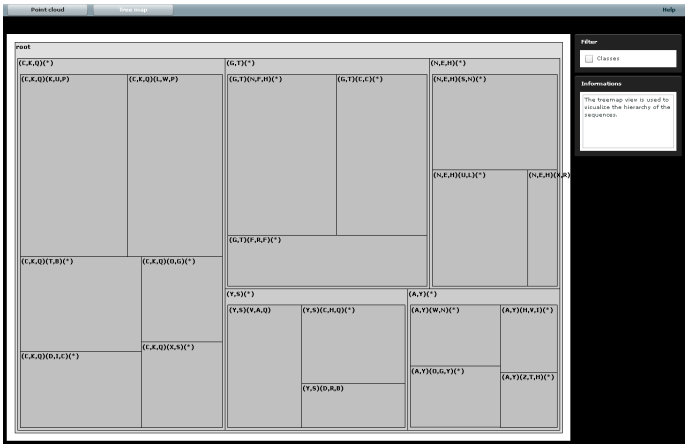


FIGURE 1.13: *Treemap* : trois niveaux sont affichés, la racine, ses enfants et les enfants de ses enfants.

FIGURE 1.14: *Treemap* : un chemin permet d'accéder aux sommets plus élevés dans la hiérarchie.

Lorsque l'utilisateur utilise le bouton droit de la souris sur un sommet du *treemap*, une bulle apparaît avec un bouton *Go to point cloud* (figure 1.15). Comme son nom l'indique, cliquer sur ce bouton permet d'accéder au nuage de points dans lequel toutes les séquences contenant des gènes présents dans la séquence sélectionnée du *treemap* seront sélectionnées (affichées en vert comme si l'on avait fait une recherche, *cf.* figure 1.5). Si l'on effectue cette opération à partir de la figure 1.15, toutes les séquences commençant par $(C, K, Q)(D, I, C)$ seront mises en vert dans le nuage de points.

De la même façon, la bulle obtenue en cliquant avec le bouton droit sur une séquence dans le nuage de points ou le système solaire (figure 1.6) permet d'accéder au sommet correspondant dans le *treemap* grâce au bouton *Go to treemap* qu'elle contient. Dans ce cas, on arrivera bien entendu au niveau d'une feuille de l'arbre (puisque les séquences sont des feuilles) mais il sera aisé de remonter dans la hiérarchie en utilisant le chemin décrit ci-dessus.

Nous allons maintenant utiliser cet exemple afin d'illustrer notre étude sémiotique de la visualisation d'information. Nous reviendrons aussi dessus dans la partie 1.3.2.2 dans

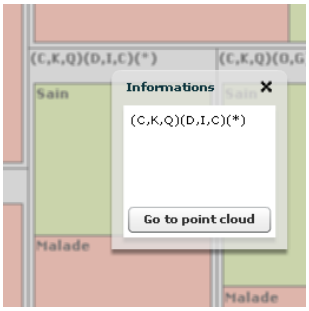


FIGURE 1.15: Lorsque l'utilisateur effectue un clic droit sur un sommet, une bulle contenant un bouton d'accès au nuage de points apparaît.

laquelle nous essaierons de valider notre approche. Mais revenons d'abord sur des questions de sémiotique, et en particulier, puisque celle-ci étudie les signes, essayons de comprendre plus en détail ce concept de signe. C'est en effet grâce à lui que nous pourrons émettre des règles spécifiques à la science de la visualisation de l'information et enfin confronter ces règles aux pratiques existantes.

1.2 Le signe

Le mot **signe** désigne communément « quelque chose qui est là pour représenter autre chose » [86]. Ainsi, chaque disque gris de la figure 1.3 est un signe représentant une séquence de gènes. De même, la distance euclidienne entre un disque gris de la figure 1.9 et le disque rouge du centre est un signe représentant la valeur de la mesure $S2MP$ entre deux séquences de gènes.

Cependant, cette définition intuitive du signe est trop simple pour servir de base à une science capable de rendre compte de l'ensemble des phénomènes sémiotiques. C'est pourquoi différentes approches ont été proposées. Dans cette partie, nous allons passer en revue certaines d'entre elles afin d'en extraire des éléments utiles à la définition d'une science de la visualisation. Pour cela, en se basant sur les travaux de Eco [61], nous allons commencer par définir la notion de signe en tant qu'élément du processus de communication, puis en tant qu'élément du processus de signification et enfin selon ses dimensions sémantique, syntaxique, et pragmatique.

1.2.1 Le signe comme élément du processus de communication

Essayons de préciser la définition ci-dessus tenant compte de la fonction communicationnelle du signe. Par exemple, imaginons que la carte de la figure 1.9 soit dépourvue de légende et que l'utilisateur n'ait pas été averti de la signification de la distance euclidienne entre le centre et une séquence. Dans ce cas, cette distance ne peut plus être considérée comme un signe par lui puisqu'elle ne possède pas de signification pour lui. Un objet, pour être un signe, doit donc permettre la transmission de l'information qu'il représente, autrement dit, il doit être un support pour la communication. Voyons ce que

cela implique. D'après le modèle de la communication de Jakobson [9] [101] schématisé par la figure 1.16, le signe permet la transmission d'une information entre une entité [10] émettrice ou **émetteur** et une entité réceptrice ou **récepteur** grâce à un **code** constitué des règles permettant aux entités de lui attribuer une signification. Sans code, un objet ne peut donc pas être un signe. À noter aussi qu'un signe nécessite un **canal** de transmission. Le **référent** correspond quant à lui à l'objet substitué par le signe dans la définition du premier paragraphe. Enfin, le **message** présenté par Jakobson peut correspondre à un signe ou à un ensemble de signes.

Revenons à l'exemple du signe /distance euclidienne/ [11] entre un disque gris et le disque rouge de la figure 1.9 afin d'illustrer les six pôles du processus de communication. L'émetteur du signe est l'ordinateur, le récepteur est un biologiste. Le canal de transmission est quant à lui composé de l'écran de l'ordinateur, d'une distance entre cet écran et l'œil du biologiste et de ses organes intervenant dans le processus de perception visuelle. Chaque point de ce canal est très important car il peut amener une dégradation du message durant la transmission. Le code nous dit que la mesure $S2MP$ est proportionnelle à la distance euclidienne sur l'écran. Le référent est donc la valeur de la mesure $S2MP$ entre les deux disques étudiés. Enfin, le message composé du signe /distance euclidienne entre un disque gris et le disque rouge/ correspond à « une valeur particulière de la mesure $S2MP$ entre une certaine séquence et son centre ».

Généralisons maintenant ces résultats à un système de visualisation de l'information quelconque : l'émetteur est un ordinateur [12] et le récepteur une personne physique. Une science de la visualisation de l'information, selon l'approche communicationnelle, aurait donc pour but de découvrir une liste de règles (code), permettant à un ordinateur (émetteur) de produire automatiquement des signes (messages) pour un utilisateur (récepteur), par l'intermédiaire d'un écran, d'un espace et du canal de perception visuelle humain (canal), à partir de données quelconques (référent).

9. Le premier modèle de la communication a été proposé par Shannon [162] pour la compagnie téléphonique *Bell*. Selon ce modèle, l'information est transformée en signal par un émetteur, puis transmise à un récepteur qui peut ensuite la décoder. Le signal est soumis aux lois de l'entropie, c'est-à-dire qu'il sera en partie brouillé par du bruit lors de la transmission. Ce modèle ne tient pas compte de la nature de l'émetteur et du récepteur, du sens du message ou de sa signification et n'est donc pas adapté à notre étude sémiotique. C'est pourquoi nous avons choisi d'utiliser le modèle de Jakobson qui inclut ces paramètres. D'autres modèles plus complexes ont aussi été développés mais leur degré de précision est inutile pour les besoins de notre étude. En particulier, celui de Eco [59] est, à notre connaissance, le plus complet (bien qu'il ait été développé exclusivement pour la communication textuelle). Pour une vue d'ensemble des modèles de la communication, nous vous conseillons le livre d'Ollivier [138].

10. Le mot entité est ici utilisé pour désigner une instance théorique telle que la définit Klinkenberg [114], qui peut un être aussi bien une personne physique qu'un autre organisme vivant (*e.g.* l'odeur laissée par un chevreuil est le signe de passage d'un gibier pour le chien de chasse) ou une machine (*e.g.* un compilateur produit un code objet exécutable par un ordinateur).

11. Dans la suite de notre exposé, l'actualisation physique d'un signe sera notée entre deux barres de divisions et sa signification entre guillemets. Par exemple, un /triangle rouge/ dans le code de la route a pour signification « avertissement ».

12. Nous limitons ici la visualisation de l'information aux cartes produites automatiquement grâce à l'outil informatique, nous reviendrons dans la partie 1.3 sur les différentes approches disponibles dans la

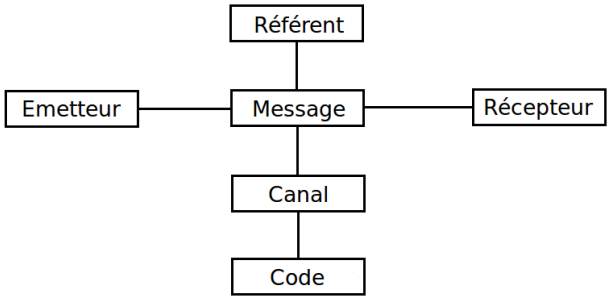


FIGURE 1.16: Modèle de Jakobson, les six pôles présent dans un acte de communication : émetteur (producteur d'un objet), récepteur (destinataire de l'objet), référent (ce de quoi parle l'objet), canal (support physique de l'objet, *e.g.* un écran d'ordinateur), code (règles permettant d'attribuer une signification à l'objet), message (lieu d'interaction des cinq facteurs énoncés ci-dessus, signe ou ensemble de signes).

D'après Klinkenberg [114], trois types de facteurs peuvent faire échouer le processus de communication : le bruit, une erreur sur le signe, une erreur sur le contexte. Il faudra donc que le cartographe en visualisation de l'information soit particulièrement attentif à ne pas commettre ces erreurs :

1. Le **bruit** [13] correspond à une altération du message pendant que celui-ci transite dans le canal. En visualisation de l'information, il peut donc être causé par l'écran, l'espace entre l'écran et l'œil, et le canal perceptif de l'utilisateur. Le cartographe doit donc vérifier que chacune de ces parties du canal ne modifie pas le message qu'il a produit :
   - Le principal problème que l'écran peut poser est induit par sa résolution, autrement dit par la quantité d'information qu'il peut afficher. Il faut donc que le cartographe s'applique à fournir une carte dont la complexité est adaptée à l'écran. Un autre problème relatif à l'écran est celui de l'altération des couleurs. En effet, les paramètres d'un écran d'ordinateur, le fait que la carte soit projetée grâce à un rétroprojecteur *etc.*, peuvent modifier les couleurs utilisées et ainsi être sources de bruit.
   - Deux paramètres de l'espace entre l'écran et l'utilisateur doivent être pris en compte afin de ne pas modifier le message. Tout d'abord, la luminosité de l'environnement peut modifier la façon dont les couleurs utilisées dans la carte sont perçues. Ensuite, la quantité d'information disponible sur la carte doit être calculée en fonction de la distance entre l'écran et l'utilisateur, et de la taille de l'écran.
   - Le canal de perception humain est un appareil complexe qui a déjà été étudié par Colin Ware [194]. Celui-ci propose un modèle du processus cognitif, explique quels en sont les mécanismes et comment optimiser son utilisation dans la création de

littérature.

13. Attention ici à ne pas confondre le mot bruit avec sa définition couramment utilisée en visualisation de l'information. Ce mot correspond ici à une altération du message et non à un ensemble de données non significatives.

cartes (choix des formes, des couleurs, lois de la *gestalt*[14], *etc.*). C'est pourquoi nous ne reprendrons pas ici l'ensemble de ses remarques. À titre d'exemple, une règle typique à respecter lors de la création d'une carte est de ne pas utiliser les couleurs rouges et vertes pour différencier certains éléments. En effet, une personne atteinte de deutéranopie (10% des hommes et 1% des femmes souffrent de daltonisme) ne pourra pas dans ce cas distinguer ces éléments.

2. Une **erreur sur le signe** apparaît lorsque le code utilisé par le cartographe pour transformer ses données en signes est différent de celui utilisé par le récepteur pour passer des signes aux données. En visualisation de l'information, c'est la légende qui permet ces transcriptions, d'où l'importance qu'il faut lui accorder (nous reviendrons plus en détail sur ce point dans la section suivante, après avoir défini certains concepts). De plus, comme Alexandru Telea nous le fait remarquer dans son livre [174], le code utilisé ne peut être efficace que si chaque signe n'est associé qu'à un seul référent[15]. Dans le cas contraire, le message peut être ambigüe.
3. La notion de référent introduite ci-dessus doit être prise dans un sens très large. On ne considère pas la réalité objective de l'objet signifié mais sa réalité subjective, c'est-à-dire relative à un certain **contexte**. Une **erreur sur le contexte** peut arriver lorsque l'on ne prend pas en compte ce contexte, c'est-à-dire les faits sociaux du milieu dans lequel l'utilisateur évolue[16]. D'après la définition de Durkheim [56] un fait est social lorsqu'il « est général dans l'étendue d'une société donnée tout en ayant une existence propre, indépendante de ses manifestations individuelles ». Par exemple, la couleur verte est souvent utilisée en Occident pour signifier une idée de vie, alors que le rouge est utilisé pour signifier une idée de danger. Comme l'a fait remarquer Ware [194], il n'en est pas de même en Chine où le rouge est symbole de vie et le vert symbole de mort. Ainsi, le *treemap* proposé dans la section précédente (voir figure 1.12), dans lequel la couleur rouge symbolise la maladie et la couleur verte le fait d'être sain, peut susciter des problèmes d'interprétation en Chine. En revanche, il est parfaitement compris par les biologistes français pour lesquels nous avons développé *SequencesViewer*. Il est donc primordial pour le cartographe d'identifier

14. Les lois de la *gestalt* (mot allemand signifiant « essence ou forme complète d'une entité ») proviennent de la théorie connue en France sous le nom de psychologie de la forme (vient de l'allemand *Gestaltpsychologie*). Köhler, l'un des fondateurs de cette théorie, nous en donne une vision d'ensemble dans son ouvrage [120]. Les lois qui nous intéressent en visualisation de l'information sont décrites dans le livre de Ware [194]. On peut citer par exemple la loi de proximité (une des plus simples et des plus utilisées) qui nous dit que des éléments spatialement proches dans une carte seront automatiquement perçus comme étant groupés même si une autre variable visuelle, par exemple la forme ou la couleur, induit une autre partition de ces éléments.

15. Dans le livre [174], le passage des données aux signes (*direct mapping*) est considéré comme une fonction injective. Ainsi, on a l'assurance que le passage du signe aux données (*inverse mapping*) ne provoque pas d'ambiguïtés.

16. Une erreur de contexte peut aussi survenir lorsqu'un signe a une signification particulière pour un seul utilisateur, indépendamment des faits sociaux du milieu dans lequel il évolue. Dans ce cas là, c'est le vécu de cet utilisateur qui produit cette erreur. Nous ne traiterons pas ici de ce problème car nous pensons qu'il est impossible, ou tout du moins trop ambitieux, d'établir des lois générales de production de cartes tenant compte des événements particuliers qui feront qu'une personne réagira différemment aux mêmes stimuli que les personnes du groupe social auquel elle appartient.

le groupe social auquel il s'adresse afin de connaître les faits sociaux relatifs à ce groupe et d'adapter sa carte à ces faits s'il veut éviter des erreurs de contexte [17].

Nous venons d'évoquer ci-dessus cinq types de problèmes qui peuvent entraîner une erreur dans la transmission d'un message contenu dans n'importe quel énoncé. Ces remarques appartiennent donc à la sémiotique générale dans la mesure où elles sont valables pour n'importe quelle sémiotique particulière. En revanche, les exemples que nous fournissons sont issus de la visualisation de l'information et les précisions qu'ils impliquent sur les mises en garde générales constituent des règles relatives à une sémiotique particulière : la science de la visualisation de l'information. En effet, elles doivent être mises en pratique pour chaque processus de visualisation d'information (sémiotique appliquée), c'est-à-dire pour la production de chaque carte.

L'approche du signe selon sa fonction communicationnelle que nous venons d'exposer fait donc intervenir les acteurs du processus de communication. Cependant, l'utilisation de ces entités peut paraître hors de propos dans une définition stricte du signe. En effet, on peut aussi considérer qu'un objet est un signe seulement s'il est doté d'une signification, indépendamment de l'utilisation qui en est faite. C'est pourquoi les sémioticiens se sont intéressés à le définir comme un élément du processus de signification. La prochaine section décrit cette approche.

1.2.2 Le signe comme élément du processus de signification

Comme nous l'avons déjà mentionné plus haut, le mot signe désigne communément « quelque chose qui est là pour représenter autre chose » [86]. En partant de cette observation, Ferdinand de Saussure [54], le père de la linguistique moderne, a défini le signe comme étant un objet composé de deux éléments : un **signifiant**, *i.e.* l'objet physique porteur du signe, et un **signifié**, *i.e.* l'idée ou le concept porté(e) par le signe.

Ce système est cependant incomplet car il ne prend pas en compte la chose réelle correspondant au concept porté par le signifié. Revenons sur notre exemple de la section 1.1. Un /disque/ de la figure 1.3 correspond au signifié du signe correspondant. Imaginons qu'il a pour signifiant « la séquence de gènes $< (FBSP3)(GRM1) >$ ». Il ne peut acquérir son statut de signe qu'à partir du moment où cette séquence fait référence à un phénomène, c'est-à-dire qu'il existe deux entités (gènes), qui apparaissent fréquemment ensemble dans des organismes et tels que l'une de ces entités a un niveau d'expression supérieur à l'autre. Cet élément du signe est appelé **référent**. La notion de référent peut paraître ambiguë mais c'est la seule qui a été trouvée pour prendre en compte le fait que nous pensons traiter des choses réelles lorsque nous utilisons un système de signes.

Eco [61] fait remonter cette distinction aux Stoïciens mais c'est Peirce [139] qui est le premier sémioticien à avoir défini la notion de signe comme la réunion de ces trois éléments (representatum pour signifiant, interprétant pour signifié et objet pour référent). La figure 1.17, proposée par Eco, représente cette définition triadique du signe en reprenant les différentes appellations des éléments selon les auteurs classiques. Il existe aussi d'autres modèles, plus complexes, permettant de décrire un signe, *e.g.* le modèle tétradique [114].

17. On pourrait cependant remarquer ici que la légende peut aussi permettre d'éviter certaines erreurs de contexte, comme celle évoquée ci-dessus. Nous reviendrons sur ce point dans la section 1.2.3.3.

Nous n'allons pas les présenter ici car le triadique est amplement suffisant pour la suite de notre exposé. On peut cependant remarquer que chacun d'entre eux repose sur la distinction introduite par Hjemslev [95] entre un **plan de l'expression** (*i.e.* forme physique du signe, signifiant dans le modèle triadique) et un **plan du contenu** (*i.e.* signification du signe, signifié + référent dans le modèle triadique). C'est l'union de ces deux plans, la **semiosis**, qui constitue la forme sémiotique. Cette union est représentée dans la figure 1.17 par un lien discontinu entre le signifiant et le signifié car elle résulte de l'établissement d'un lien entre le signifiant et le signifié d'une part, et d'un lien entre le signifié et le référent d'autre part.

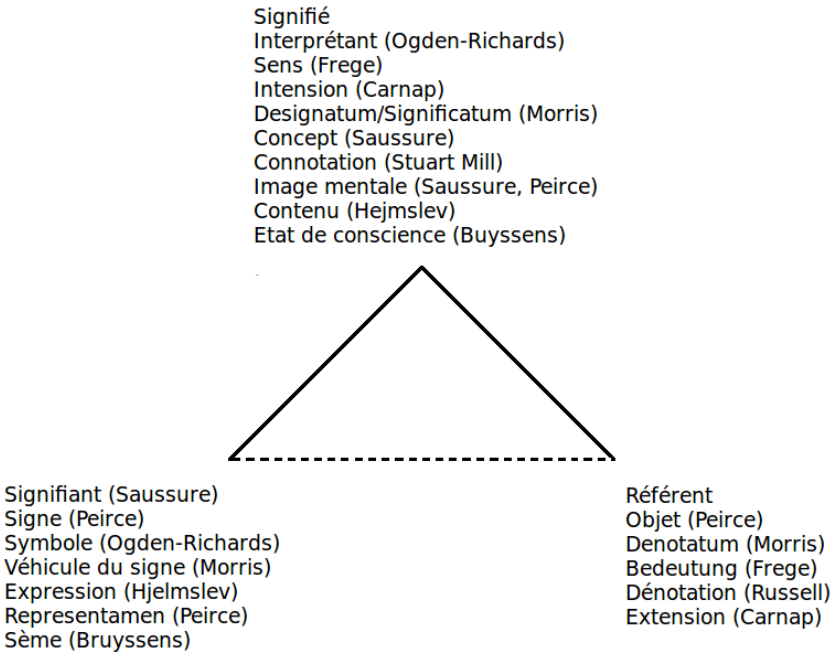


FIGURE 1.17: Définition triadique du signe. Signifiant : entité physique du signe. Signifié : concept signifié. Référent : objet auquel le signe se réfère.

Les notions présentées ci-dessus nous amènent à faire quelques remarques sur certains des points délicats pouvant entraîner l'absence de sémiosis dans une carte produite par un système de visualisation d'information. Ces points correspondent aux erreurs sur le signe évoquées dans la section précédente. Nous pouvons maintenant les analyser grâce aux nouvelles notions que nous avons introduites. Tout d'abord, nous devons insister sur la nécessité de produire une légende claire. En effet, lorsque l'on veut comprendre un texte écrit dans une langue étrangère, un dictionnaire [18] nous permet de transformer les signifiants de cette langue en des signifiants de notre langue natale afin de pouvoir leur associer un signifié. Par exemple, nous allons traduire le signifiant anglais /horse/ en signifiant français /cheval/ par l'intermédiaire duquel nous pourrons associer le signifié « cheval », c'est-à-dire l'idée d'un « animal à quatre pattes permettant aux chevaliers du Moyen-Âge de se livrer bataille ». La prochaine fois que nous rencontrerons ce mot /horse/

18. Le mot dictionnaire est employé ici comme un outil établissant des liens d'équivalence entre les signes issus d'un premier code et les signes issus d'un second code (*e.g.* un dictionnaire anglais-français).

nous pourrons ainsi directement l'associer à son signifié. Il en est de même pour la visualisation de l'information, il nous faut un dictionnaire, c'est-à-dire un moyen permettant de passer du code utilisé dans la carte à un code qui nous est connu, le plus souvent le code linguistique [19]. C'est principalement la légende qui joue ce rôle. Lorsque les signes sont plus compliqués, on peut aussi passer par un manuel utilisateur ou tout simplement transmettre oralement le code.

Un autre problème beaucoup plus difficile à traiter et qui peut limiter l'acceptation d'une méthode de visualisation de l'information concerne la difficulté à établir un lien entre le signifié et le référent. Dans l'exemple de la section 1.1, nous avons construit des groupes de séquences et nous les avons représentés de façon à ce que l'utilisateur puisse les distinguer (voir figures 1.3 et 1.9). Une description détaillée de l'algorithme peut permettre au biologiste de connaître le signifié de tels groupes. En revanche, il peut être compliqué pour lui d'interpréter à quels phénomènes concrets correspond ce découpage. Il existe d'ailleurs énormément de méthodes de fragmentation et énormément de mesures de leurs qualités menant à des résultats contradictoires et difficilement interprétables [20]. Dans un tel contexte, alors que même les spécialistes de la fragmentation n'arrivent pas à définir ce qu'est une fragmentation efficace, comment le biologiste pourrait-il avoir une vision claire des phénomènes concrets que la fragmentation est censée lui présenter.

Un deuxième exemple de ce type de problème est soulevé dans [44]. Ici, les auteurs ont observé que des indices globaux reflétant la pertinence des pages de wikipedia étaient très peu utilisés. La semiosis est ici bloquée car l'utilisateur a certes accès au signifiant (*e.g.* /0.33/) et à son signifié (*e.g.* « l'indice de pertinence de la page est de 0.33 sur 1 ») mais il lui manque le référent, *i.e.* à quoi correspond un indice de 0.33 concrètement. Il ne peut donc pas savoir dans quelle mesure il peut se fier au contenu de cette page et la liaison entre le plan de l'expression et le plan du contenu est rompue. La solution proposée par les auteurs consiste à remplacer l'indice par plusieurs signes reflétant tous une propriété particulière de l'indice et dont le lien entre le signifié et le référent est facilement établi par l'utilisateur.

Pour résumer les deux points que nous venons d'évoquer, le producteur d'une carte de visualisation de l'information doit, pour chaque signe qu'il utilise dans sa carte, d'une part, s'assurer que le lien entre le signifiant et le signifié est correctement fourni à l'utilisateur (grâce à une légende, un texte, oralement ...) et, d'autre part, s'assurer qu'il existe un lien entre le signifié et le référent. Il semble d'ailleurs qu'une étude des réactions suscitées par la carte auprès des utilisateurs soit nécessaire pour éclaircir ce point.

19. Le moyen décrit ici nous permet en fait de repousser le problème de la signification des objets de la carte au problème de la signification des éléments linguistiques : d'où vient le sens que l'on attribue à ces éléments ? Ce problème relève principalement de la philosophie et c'est pourquoi nous n'en parlerons pas ici. L'essai « Le problème de la signification en linguistique » du recueil [144] de Quine, pourra aider le lecteur à en savoir plus. Notons que ce problème a des conséquences importantes pour la conception des dictionnaires établissant des liens d'équivalence entre signes d'un même code (*e.g.* le dictionnaire de la langue française). Pour une complète description de ces questions, voir [62].

20. Pour se faire une idée du nombre de techniques disponibles, nous convions le lecteur à jeter un œil sur l'article de Schaeffer [160] donnant une vue d'ensemble des méthodes de fragmentation de graphe. La difficulté d'interprétation des mesures de qualité de ces méthodes a quant à elle été observée par Boutin et Hascoët [27].

Nous allons maintenant nous intéresser à une troisième approche de définition d'un signe selon ses dimensions sémantique, syntaxique et pragmatique.

1.2.3 Le signe perçu selon trois dimensions : sémantique, syntaxique et pragmatique

Morris [128] distingue trois façons de considérer un signe : les dimensions syntaxique, sémantique et pragmatique. Cette approche s'est largement répandue dans les domaines scientifiques grâce à son efficacité pratique [61]. Les sections suivantes sont dédiées à l'étude de chacune de ces dimensions.

1.2.3.1 Dimension syntaxique

Conformément à la dimension **syntaxique**, le signe est « abordé en ce qu'il peut être inséré dans des séquences d'autres signes, selon certaines règles de combinaisons » [61]. Essayons de comprendre cette définition en analysant la carte 1.10. Cette carte, qui possède une signification en elle-même et qui peut donc être considérée comme un signe, est composée d'éléments eux aussi porteurs d'une signification. Par exemple, le /cercle gris du centre/ est un signe ayant pour signification « la séquence $< (CRSP3)... >$ ». En décomposant une carte récursivement, on finit par trouver un ensemble de signes ne pouvant plus être eux-mêmes décomposés en signes. Par exemple, la couleur /bleu clair/ ne peut pas se diviser en éléments porteurs d'une signification. En revanche, elle signifie « old ». Ces signes « minimaux » sont appelés **unités** sémiotiques de l'énoncé. La combinaison de ces unités permet de produire de nouveaux signes d'ordre 1. Par exemple, sur la carte 1.10, l'unité /bleu clair/ signifiant « *old* » est combinée avec l'unité /carré/ signifiant « l'article intitulé ... » pour former le signe d'ordre 1 /carré bleu clair/ qui a pour signification « l'article intitulé ... a été publié il y a longtemps ». Ensuite, la combinaison de signes d'ordre 1 avec des unités ou avec d'autres signes d'ordre 1 donne des signes d'ordre 2, et ainsi de suite jusqu'à l'obtention d'un signe contenant toutes les informations que l'émetteur veut transmettre au récepteur. C'est ce signe que nous avons appelé **énoncé** ou **carte** dans le cas de la visualisation d'information.

Pour poursuivre notre étude, nous devons maintenant revenir sur une distinction fondamentale entre signes motivés et arbitraires. On appelle signe **arbitraire**, ou **symbole** un signe dont le rapport entre le plan de l'expression et le plan du contenu est purement conventionnel. Par exemple, lorsque je désigne la séquence de gène « $< (FBSP3)(GRM1) >$ » par un /cercle gris/, j'utilise un symbole. De même, le signe linguistique $/< (FBSP3)(GRM1) >/$ est un symbole signifiant « Le gène $FBSP3$ en tant que réalité physique a un niveau d'expression moyen supérieur au gène $GRM1$. Ces deux gènes apparaissent souvent dans le même organisme *etc.* ». En revanche, si j'affiche un réseau social avec des photos de personnes et des liens entre elles si ces personnes sont amies, le signe utilisé pour représenter une personne (*i.e.* sa photo) est **motivé** car il a une certaine ressemblance avec l'objet qu'il a pour référent (*i.e.* la personne correspondante). Un tel signe

est appelé un **icône** [21]. Cette classification peut paraître assez floue, car il est probable que tout signe possède un degré « d'arbitrarité » et un degré de motivation comme le fait remarquer Klinkenberg [114]. Des typologies plus évoluées ont d'ailleurs été proposées, par exemple par Eco [63]. Nous nous en tiendrons cependant à celle-ci car elle est suffisante pour notre exposé et plutôt intuitive.

En sémiotique visuelle [22], les symboles sont appelés des **signes plastiques**. Le groupe μ identifie trois types de signes plastiques jouant le rôle d'unités sémiotiques dans une carte :

1. Les **chromènes** correspondent à des couleurs définies par une teinte, une luminosité et une saturation. Le /bleu clair/ de l'exemple précédent est un chromène.
2. Les **formènes** correspondent à des formes caractérisées par une dimension, une position et une orientation (*e.g.* le /carré/ dans l'exemple).
3. Les **texturèmes** correspondent à des textures.

Dans le domaine de la visualisation d'information, ces types d'unités plastiques sont appelés **variables visuelles**. Ward *et al.* [193] en proposent huit et on va voir que l'on retombe plus ou moins sur celles observées par les sémioticiens :

1. La **luminosité** correspond à la composante luminosité des chromènes.
2. La **couleur** correspond aux composantes teinte et saturation des chromènes. Ward *et al.* ne font pas la distinction entre ces deux composantes. Peut-être considèrent-ils que leur attribuer des significations différentes pourrait provoquer des confusions lors du décodage de la carte par le récepteur ?
3. La **forme** correspond à la forme des formènes.
4. La **taille** correspond à la dimension des formènes.
5. La **position** correspond à la position des formènes.
6. L'**orientation** correspond à l'orientation des formènes.
7. La **texture** correspond au texturème.
8. Le **mouvement** n'apparaît pas dans le modèle sémioticien. Il permet pourtant de coder certaines propriétés des données, auquel cas on ne peut nier son existence en tant que signe.

Il est ici important de remarquer que ces types ne sont pas complètement indépendants les uns des autres. Par exemple, la texture dépend d'une variation de la luminosité.

Revenons maintenant à des questions syntaxiques. Comme nous l'avons indiqué plus haut, la combinaison d'unités (qu'elles soient plastiques ou iconiques) produit des signes. Puis la combinaison de ces signes produit d'autre signes, et ainsi de suite jusqu'à obtention d'une carte. Une question intéressante reste cependant ouverte : la carte est-elle de nature

21. Pour plus de détails sur la notion d'icône et les problèmes qu'elle soulève, voir l'essai intitulé « L'iconicité » de Bordron dans le recueil [99].

22. La sémiotique visuelle étudie les signes se manifestant par l'intermédiaire du canal visuel. Les travaux menés jusqu'ici concernent principalement les objets artistiques ou le cinéma [129]. Cependant, certains résultats peuvent être appliqués à la visualisation de l'information, qui est elle aussi une sémiotique visuelle.

arbitraire ou iconique ? En effet, même une carte composée d'unités plastiques peut être considérée comme iconique. C'est en tout cas ce que soutient Peirce [139] parce qu'il estime que les diagrammes (comme les expressions algébriques *etc.*) possèdent des propriétés « configurationelles » de leurs référents. Nous n'entrerons pas dans ce débat qui relève de la philosophie mais le lecteur intéressé peut se reporter au livre d'Eco [61].

L'étude des règles de combinaison des signes en linguistique constitue un domaine de la recherche très actif. Un objectif est de pouvoir définir, pour chaque code linguistique (*i.e.* langue) une série de règles permettant la génération automatique de n'importe quel énoncé. C'est ce que l'on appelle les grammaires génératives introduites par Chomsky [47]. Une telle approche a aussi suscité de nombreux travaux en visualisation de l'information, dans lesquels les auteurs essaient de définir un ensemble de règles permettant de créer automatiquement, à partir d'un ensemble quelconque de données, les cartes adaptées. Combinées aux règles de la pragmatique (voir ci-dessous) qui seraient chargées de sélectionner la carte la plus utile en fonction du contexte, on obtiendrait ce que Ward *et al.* [193] définissent comme une science de la visualisation.

Le premier de ces travaux a été produit par Jacques Bertin [23, 22]. Bien qu'il l'ait réalisé sans l'aide du support informatique que nous avons défini comme une condition *sine qua non* d'une science de la visualisation, ses ouvrages restent des références incontournables de notre domaine [23]. Nous ne reviendrons pas ici dessus mais nous conseillons vivement au lecteur d'y jeter un œil. Un autre travail qui nous semble particulièrement abouti est la grammaire des graphiques proposée par Wilkinson [198], bien que celle-ci ne couvre que le domaine de la visualisation statistique. Si le lecteur est intéressé par une vue d'ensemble des travaux concernant les grammaires génératives de cartes, nous lui conseillons la section 4.4 du livre [193] déjà mentionné.

Avant de poursuivre notre exposé, nous allons proposer une méthodologie possible pour la découverte d'une grammaire générative de la visualisation de l'information. Elle est basée sur une approche empiriste logique telle qu'elle a été introduite par le cercle de Vienne [41] (pour une introduction simple à l'épistémologie, le lecteur pourra se référer au livre d'Esfeld [66], il y trouvera les principes exposés ci-dessous ainsi que les critiques qui peuvent leur être faites). Cette méthode est suggérée par Chomsky [47] dans le cadre de la production de grammaires génératives en linguistique lorsqu'il écrit : « Toute théorie scientifique s'appuie sur un nombre fini d'observations ; elle cherche à rendre compte des phénomènes observés et à en prédire de nouveaux par la construction de lois générales en termes de concepts hypothétiques, tels que "masse" et "électron" en physique par exemple. De même, une grammaire de l'anglais est fondée sur un corpus fini d'énoncés (les observations) et elle contient certaines règles grammaticales (les lois) [...]. Ces règles expriment des relations structurelles entre les phrases du corpus et le nombre infini de phrases engendrées par la grammaire au-delà du corpus (les prédictions). ». Essayons d'appliquer ce principe à la grammaire d'une visualisation de l'information. En partant d'un ensemble

23. L'étude de Bertin porte sur la sémiotique graphique. Cette sous-branche de la sémiotique visuelle ne peut pas être assimilée à la science de la visualisation de l'information dans la mesure où elle ne s'impose pas l'utilisation de l'outil informatique. La science de la visualisation de l'information est donc une sous-branche de la sémiotique graphique : elle peut utiliser les lois qui définissent cette dernière mais en possède aussi d'autres relatives à la technologie utilisée, *i.e.* l'ordinateur.

fini d'énoncés (les observations), nous devons, par induction, postuler des règles grammaticales (les lois), *i.e.* directrices de la création de n'importe quelle carte. Cette synthèse des phénomènes doit fournir d'une part une taxonomie des types de données et d'autre part une taxonomie des types de cartes. Elle doit aussi fournir le lien entre les deux, c'est-à-dire contenir des règles du type « telles catégories de données peuvent être visualisées grâce à telles catégories de cartes ». Une fois ces lois postulées, il faut essayer de les valider. Pour cela, nous devons les analyser, c'est-à-dire étudier quelles en sont les implications. Nous allons ainsi déduire de ces lois des énoncés et déterminer s'ils sont valides afin de réfuter ou d'attribuer à ces lois un degré de vérisimilitude selon la méthode de Popper [24] [143]. La production d'une grammaire générative de la visualisation d'information doit permettre de répondre à la question : quelles sont les cartes valides qui nous permettraient de visualiser telles données particulières. En revanche, elle n'a pas pour but d'identifier, parmi les cartes valides, celles qui sont le plus adaptées à l'utilisateur. Ce deuxième problème primordial de la visualisation de l'information ne relève pas de la syntaxe mais de la pragmatique. Nous verrons dans la partie 1.3.2 comment il peut être traité.

Pour conclure sur les grammaires génératives, nous pouvons observer que la méthode décrite ci-dessus paraît difficile à mettre en place. En effet, contrairement aux langues qui sont des systèmes de signes aboutis, la visualisation de l'information n'en est encore qu'à ses débuts et les chercheurs continuent de trouver de nouveaux types de représentations. Dans un tel contexte, les observations (*i.e.* les cartes) ne peuvent pas être suffisantes. De plus, leur nombre augmente rapidement ce qui peut rendre caduque toute tentative de grammaire précédemment proposée. Nous pensons cependant que des travaux allant dans ce sens sont utiles à notre domaine. En effet, on peut supposer que certaines règles découvertes seront valables aussi pour des types de cartes non encore proposées. De plus, et c'est certainement le point le plus important, il est possible que de telles grammaires puissent nous aider à découvrir des nouveaux types d'énoncés.

1.2.3.2 Dimension sémantique

D'après Eco [61], selon sa dimension **sémantique**, « le signe [...] est conçu dans sa relation avec ce qu'il signifie ». Cette approche revient à considérer le signe comme un élément du processus de signification, approche que nous avons traitée dans la section précédente. C'est pourquoi nous n'allons pas nous étendre dessus ici.

1.2.3.3 Dimension pragmatique

Stéphane Madelrieux, dans sa préface de la dernière édition française du livre fondateur de la doctrine pragmatique de William James [102], nous livre la phrase suivante :

24. Si nous avions la certitude que le nombre de cartes est fini et non trop grand, nous pourrions toutes les produire selon des lois postulées et déterminer de façon catégorique si ces lois sont valides ou non. Cependant, puisque leur nombre est potentiellement très grand ou peut-être même infini (comme c'est le cas des grammaires en linguistique), nous n'avons pas la certitude de pouvoir toutes les produire même avec un ensemble fini de lois et nous préférons utiliser le principe de réfutabilité introduit par Popper [143]. Il consiste à considérer toute proposition comme réfutable et de lui attribuer une sorte de degré de vérisimilitude si personne ne trouve d'exemples permettant de la réfuter.

« [...] si la carte ne nous permet pas de nous orienter avec succès dans le paysage, si elle ne nous mène pas là où nous voulions aller, si elle nous perd, alors elle sera dite fausse ; et vraie dans le cas contraire. »

Cet exemple, basé sur les cartes, nous permet ici d'introduire la notion de pragmatique en philosophie. Entre autre, elle définit la véracité d'un objet comme sa propriété à remplir la fonction pour laquelle il est utilisé. Appliquée à la sémiotique, elle correspond donc à l'étude des fonctions remplies par un énoncé.

D'après Eco [61], selon sa dimension **pragmatique**, « le signe est [...] perçu en fonction de ses origines, et des effets qu'il a sur les destinataires, les usages que ceux-ci en font, *etc.* ». Cette dimension est particulièrement importante à nos yeux. En effet, l'intérêt grandissant porté à la visualisation de l'information semble essentiellement dû au fait que la carte est envisagée avant tout par les cartographes comme un moyen d'action [142]. La motivation économique qui a poussé Anaximandre à produire sa carte en est un exemple (*cf.* Introduction). De même, l'application que nous avons présentée tout au long de la section 1.1 est avant tout destinée à permettre aux biologistes d'émettre, d'infirmer, de valider des hypothèses sur les combinaisons de gènes qui pourraient être un facteur de la maladie d'Alzheimer. C'est d'ailleurs cette motivation qui a aussi poussé une société comme Pikko [25] à se développer.

La pragmatique se distingue de la sémantique dans la mesure où elle constitue une approche contextuelle du signe. Nous reprenons ici la notion de contexte énoncée par Sperber et Wilson [169] :

« Un contexte est une construction psychologique, un sous-ensemble des hypothèses [du destinataire] sur le monde. Bien entendu, ce sont ces hypothèses, et non l'état réel du monde, qui affectent l'interprétation d'un énoncé. Ainsi défini, un contexte ne contient pas seulement de l'information sur l'environnement immédiat ou sur les énoncés précédents : des prévisions, des hypothèses scientifiques, des croyances religieuses, des souvenirs, des préjugés culturels [...] sont susceptibles de jouer un rôle dans l'interprétation. » [26]

On retrouve donc ici toutes les remarques déjà émises dans la section 1.2.1. En effet, c'est grâce à sa fonction communicationnelle qu'un objet, pris dans un certain contexte, devient un signe et provoque ainsi des effets sur les destinataires [27].

De plus, l'intérêt que porte la pragmatique aux sens implicites qui peuvent être contenus dans des énoncés [114] va nous permettre de mettre en valeur un type particulier d'erreurs sur le signe. Prenons un exemple. Lorsque nous avons créé la carte de la figure 1.9, nous avons placé chaque séquence i selon des coordonnées polaires (ρ_i, θ_i) où ρ_i correspond à la distance euclidienne entre le centre et la séquence et θ_i correspond à l'angle avec l'axe des abscisses. Nous avons calculé la distance euclidienne ρ_i de façon à ce qu'elle

25. http ://www.pikko-software.com/

26. Attention de ne pas confondre avec la notion de contexte faisant référence aux signes qui entourent le signe étudié (*e.g.* co-texte si l'énoncé est de type linguistique). La notion de contexte utilisée ici inclut aussi celle que nous avions proposée dans la partie 1.2.1 mais ne s'y limite pas : certains souvenirs par exemple ne sont pas des faits sociaux et peuvent néanmoins jouer un rôle dans l'interprétation d'un énoncé.

27. Le lecteur intéressé par les processus d'ordre cognitif intervenant dans l'interprétation des énoncés peut se référer au livre [169]. Il serait d'ailleurs intéressant d'étudier la théorie de la pertinence que Sperber et Wilson y proposent afin de voir quels en sont les implications en visualisation de l'information.

soit proportionnelle à la valeur de la mesure $S2MP$ entre le centre et la séquence i. En revanche, l'angle θ_i a juste été attribué en fonction de la position de la séquence dans son groupe, cette position n'ayant aucune signification particulière. Nous avons donc projeté des données unidimensionnelles dans un espace bidimensionnel dont l'une des dimensions n'a pas de signification selon notre code. Cette opération peut être génératrice d'erreur. En effet, le récepteur risque d'interpréter cette deuxième dimension comme porteuse d'un sens implicite : par exemple, il pourrait penser que les séquences sont ordonnées selon leurs proximités dénotées par les valeurs de la mesure $S2MP$. Dans ce cas, son code serait différent du notre (nous sommes donc bien dans une erreur d'expression) et attribuerait à l'ordre angulaire un sens implicite et non désiré par le cartographe [28]. Le cartographe devra donc accorder une attention particulière à ne pas introduire malencontreusement dans ses énoncés des signes implicites de ce type.

On pourrait même étendre cette remarque à n'importe quel signe implicite, c'est-à-dire aux signes implicites introduits délibérément par le cartographe. En effet, ils peuvent être aussi source d'erreurs dans la mesure où l'utilisateur peut mal les interpréter. En plus, comme nous l'avons vu dans la section 1.2.1, il est très facile de les transformer en signes explicites grâce à la légende.

Comme nous l'avons évoqué dans la note 17, on pourrait penser qu'une erreur de ce type puisse être évitée grâce à la légende. Le problème semble cependant plus compliqué. En effet, la légende renvoie à des concepts (par exemple celui de gène dans notre visualisation) qui eux aussi seront interprétés grâce au contexte du destinataire. Il est en effet évident que le mot gène fera appel à un contexte beaucoup plus vaste si le destinataire est un biologiste travaillant sur la maladie d'Alzheimer que si le destinataire est un informaticien testant les valeurs sémantiques de la carte. Katz [108] soutient qu'il est possible de rendre tout énoncé assez précis pour que son interprétation ne fasse pas appel au contexte du destinataire, ce qui reviendrait à créer des cartes « sémantiquement » explicites et ainsi supprimer les effets de la dimension pragmatique. Cependant, comme il l'admet lui-même, cette tâche peut se révéler immense : imaginez le nombre d'informations qu'il faudrait inclure à notre carte afin qu'elle puisse communiquer les mêmes informations à un biologiste et à un informaticien. De plus, en suivant Sperber et Wilson [169], nous ne partageons pas l'avis de Katz. En effet, une telle opération d'explicitation nécessiterait l'ajout de tous les contextes de tous les biologistes (sinon, certaines informations accessibles à certains biologistes resteraient inaccessibles aux autres individus). Or, d'une part il nous parait impossible de déterminer les constructions psychologiques des biologistes qui leur permettent de tirer telles informations d'une carte à tel moment ; d'autre part, rien ne nous permet de dire que l'union de tous ces contextes permettrait de retrouver l'union de toutes les informations que les biologistes pourraient extraire de la carte.

Les maximes de Grice [87] ont été élaborées afin de se prémunir des erreurs liées à l'aspect pragmatique des énoncés et nous conseillons donc aux cartographes de les suivre

28. Le cas particulier d'erreur de signe dû à un sens implicite décrit ici correspond plus concrètement au non respect d'une loi de visualisation de l'information énoncée par Tufte : « *The number of information-carrying (variable) dimensions depicted should not exceed the number of dimensions in the data.* » [178]. Il nous expose donc l'une des règles découlant de la loi que nous avons sous-entendu : « Eviter les erreurs de signe ».

dans la mesure du possible [29]. Elles sont résumées ainsi dans [9] :

- *quantité* : (a) faire en sorte que l'énoncé soit aussi informatif qu'il est requis pour les besoins courants de la transmission ; (b) ne pas le rendre plus informatif qu'il est requis (cette erreur est celle commise dans l'exemple donné ci-dessus) ;
- *qualité* : (a) ne pas donner de fausses informations ; (b) ne pas donner d'informations dont la validité n'est pas établie ;
- *relation* : être pertinent ;
- *modalité* : (a) éviter toute obscurité dans l'énoncé ; (b) éviter toute ambiguïté ou toute prolixité ; (c) être ordonné.

Dans le cadre de la visualisation d'information, ces maximes trouvent d'ailleurs leur pendant dans les livres de Tufte [178, 179, 180] que nous conseillons vivement au lecteur. On peut par exemple y retrouver la maxime de quantité dans les énoncés suivants : « *Above all else, show the data. Maximize the data-ink ratio, within reason. Erase non-data-ink, within reason. Erase redundant data-ink, within reason.* [...] *The number of information carrying (variable) dimensions depicted should not exceed the number of dimensions in the data* ». La maxime de qualité se retrouve dans la question « *Is the representation accurate ?* » ou la phrase « *Forgot chartjunk* ». Celle de relation correspond quant à elle à la question « *Is the diplay revealing the truth ?* ». *Etc.*.

Nous avons maintenant introduit toutes les notions dont nous aurons besoin et défini plusieurs règles relatives à la science de la visualisation de l'information. Nous avons aussi mentionné les ouvrages contenant d'autre règles de cette science. Nous allons maintenant utiliser les résultats obtenus grâce à une analyse des concepts de la sémiotique en les comparant à certains résultats empiriquement démontrés dans le domaine de la visualisation de l'information.

1.3 La visualisation de l'information

La visualisation de l'information est un domaine actif de la recherche comme peuvent en témoigner les nombreux livres portant sur le sujet. Nous allons décrire brièvement certains d'entre eux afin d'orienter le lecteur qui serait intéressé par une vision plus approfondie que celle présentée dans notre document. Plus proche du manuel scolaire que de l'ouvrage scientifique, simple, précis et ludique, le livre de Spence [168] constitue un excellent point de départ pour un néophyte. Une autre introduction simple et donnant un aperçu des méthodes développées par le domaine a été proposée par Mazza [126]. Pour aller plus loin, le livre de Chen nous propose un tour d'horizon de ces méthodes [43]. Le livre de Ward *et al.* [193] nous semble incontournable. Les auteurs y présentent les

29. Ces maximes, malgré leur popularité, ont néanmoins été soumises à des critiques. Le lecteur pourra trouver une introduction aux problèmes soulevés dans l'ouvrage [9]. Les maximes de quantité et de qualité peuvent par exemple être mises en doute si l'on considère qu'un énoncé est efficace à partir du moment où il remplit les objectifs que s'était fixés l'émetteur, objectifs qui peuvent être différents d'une transmission exacte des données. Ainsi, Don Norman [5] s'oppose à Tufte lorsque celui-ci explique que la présentation de la NASA ne rendait pas clairement compte des données parce qu'elle était mal conçue [181]. Il prétend au contraire que celle-ci était parfaitement bien conçue mais que ses auteurs ne voulaient pas que les données soient clairement exposées.

principaux concepts de la visualisation (au sens général du terme, c'est-à-dire non réduit à la visualisation de l'information), de nombreux résultats et un grand nombre d'exemples. Traitant aussi de la visualisation dans sa globalité, le livre de Telea [174] donne une vue d'ensemble des techniques importantes du domaine. À placer entre les mains d'un public averti, ce livre peut servir de référence au programmeur. Pour finir, deux recueils des articles ayant marqué le domaine sont aussi disponibles [40, 17] ; l'incontournable [40] peut être d'ailleurs considéré comme le livre fondateur d'un domaine de la visualisation de l'information.

Dans cette partie, nous allons essayer d'établir un lien entre les différents points évoqués dans la première section de ce chapitre et l'état des recherches en visualisation de l'information. Pour cela, nous allons commencer par définir exactement ce qu'est la visualisation de l'information. Puis, nous étudierons le modèle de Tamara Munzner [131] concernant la production et la validation de cartes en le comparant aux résultats sémiotiques que nous avons exposés précédemment.

1.3.1 Définition(s)

Afin de définir exactement l'objet d'étude de la science de la visualisation de l'information, nous allons tout simplement définir les notions « information » et « visualisation » grâce à un dictionnaire commun [30]. Tout d'abord, regardons le concept d'information. Celui-ci ne pose pas de problèmes car une définition spécifique de son emploi en informatique nous est fournie :

« **Information** : Élément de connaissance susceptible d'être représenté à l'aide de conventions pour être conservé, traité ou communiqué. »

Cette définition nous convient car une carte représentant une certaine quantité de données peut être vue comme le support d'éléments de connaissance. Voyons maintenant ce qu'il en est du terme visualisation.

« **Visualisation** : 1. Action de rendre visible d'une façon matérielle l'action et les effets d'un phénomène. »

Cette définition, très générale, inclut donc n'importe quelle carte porteuse d'information, comme celle d'Anaximandre par exemple. Cette approche, soutenue par Ward *et al.* [193], Mazza [126] ou Spence [168], ne nous semble pas satisfaisante. En effet, un rapide coup d'œil aux actes de n'importe quelle conférence spécialisée en visualisation de l'information prouve à quel point le domaine est dépendant de l'outil informatique. C'est d'ailleurs dans les laboratoires de recherche en informatique qu'elle est pratiquée et les résultats sont exclusivement publiés dans des journaux d'informatique. La définition ci-dessus nous semble plutôt correspondre à celle de la graphique [31] et dont la science (sémiotique particulière) a été initiée par Bertin [23, 22]. Nous préférons considérer la visualisation de l'information comme un sous-domaine de la graphique et la science associée un sous-domaine de la sémiotique graphique. C'est pourquoi nous utiliserons la deuxième définition du dictionnaire, approche suivie par les auteurs des livres [17, 40, 43, 174] :

30. Larousse en ligne : http ://www.larousse.fr/dictionnaires/francais-monolingue

31. D'après « le Lalande », [121] la graphique se définit comme la « méthode qui consiste à représenter des relations abstraites par des figures géométriques ».

« **Visualisation** : 2. Présentation visuelle sur un écran, sous forme d'image alphanumérique ou graphique, d'un ensemble d'informations traitées par des moyens informatiques. »

Un dernier point à considérer ici est le type des données à représenter, *i.e.* le type des informations contenues dans une carte. En effet, la mise en place de visualisations d'informations géographiques et celle de visualisations d'arbres généalogiques font appel à des techniques très différentes. C'est pourquoi la visualisation, en son sens le plus large, peut être divisée en trois sous-domaines [168] : visualisation géographique, scientifique et d'information. Le premier s'intéresse aux représentations de données géographiques, le second aux représentations de données géolocalisées mais non géographiques (un organisme biologique par exemple) et le troisième aux représentations de données abstraites (*e.g.* graphes, matrices...). Cette distinction reste cependant un sujet de discussion : Tory et Möller proposent par exemple une division plus formelle [177] en distinguant la visualisation de données continues (correspondant majoritairement aux techniques des visualisations scientifique et géographique) de la visualisation de données discrètes (correspondant majoritairement à la visualisation d'information). Cette distinction est particulièrement importante dans la mesure où les défis rencontrés dans chacune de ces disciplines et les techniques pour les résoudre sont très différents [174] [32]. Nous définissons donc maintenant la visualisation de l'information de la manière suivante :

Visualisation de l'information : Présentation visuelle d'un ensemble d'informations issues de données abstraites et traitées par des moyens informatiques.

Pour être plus précis, on peut représenter le processus de visualisation grâce à un *pipeline* (voir figure 1.18). Des phénomènes sont d'abord observés et collectés en **données brutes**. Ensuite, ces données brutes sont modélisées en **données structurées** grâce à un code pouvant être signifiant pour le système de visualisation. Ce sont ces structures qu'une science de la visualisation devrait synthétiser en taxonomies (*cf.* section 1.2.3.1 dédiée à la dimension syntaxique). Puis, ces données sont transformées en une **abstraction visuelle** contenant toutes les informations nécessaires à la représentation visuelle des données (*i.e.* variables visuelles). Enfin, une ou plusieurs **vues** dotées de systèmes d'interaction, permettent d'afficher le contenu de l'abstraction visuelle. Les flèches partant de l'utilisateur montrent comment celui-ci peut interagir avec le système. Les interactions sont en effet primordiales en visualisation de l'information. Malheureusement, l'ébauche d'approche sémiotique que nous avons proposée dans cet exposé ne les prend pas en compte. Pour cela, il faudrait se doter d'un autre modèle de communication, ce qui dépasse notre champ d'investigation, mais devra être nécessaire pour des travaux de ce type plus approfondis.

Le *pipeline* présenté ici peut être vu comme un modèle de production de cartes (nous définirons dans la prochaine section ce concept plus en détail). Étant largement répandu dans le domaine de la visualisation (« modulo » certaines variantes comme dans [40]), nous ne pouvions pas ne pas le mentionner ici, ne serait-ce qu'à titre informatif ou en guise d'introduction. Cependant, d'autres modèles plus complets ont plus tard été définis comme celui de Munzner [131]. La section suivante est dédiée à ce modèle.

32. Il serait aussi intéressant de comparer cette taxonomie avec celles proposées par Umberto Eco [63] afin de voir si la distinction trouve son pendant chez les sémioticiens.

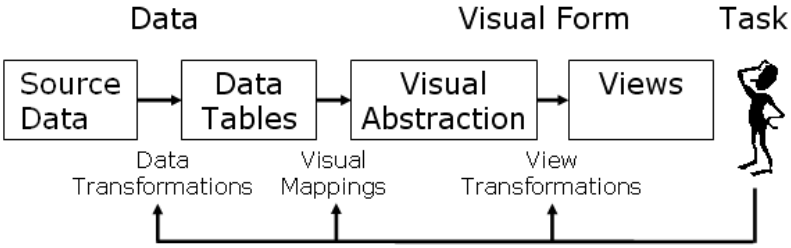


FIGURE 1.18: Le *pipeline* de la visualisation de l'information (image provenant du site de la librairie de visualisation de l'information *Prefuse*) : « *Visualization can be described as the mapping of data to visual form that supports human interaction in a workspace for visual sense making* » [40]. **Données brutes (*source data*)** : données issues de phénomènes observés, mesurés etc.. **Données structurées (*data tables*)** : données nettoyées et modélisées selon un format pouvant être décodé par l'outil de visualisation. **Abstraction visuelle (*visual abstraction*)** : type de représentation adapté au modèle des données. **Vues (*views*)** : représentation partielle des données modélisées grâce au type de représentation choisi.

1.3.2 Modèle imbriqué pour la production et l'évaluation des cartes

Un **modèle de production** de cartes permet de diviser le processus de création d'énoncés en tâches et donne ainsi au cartographe une ligne de conduite à suivre. Un tel modèle est absolument nécessaire pour la création d'énoncés complexes. D'une part, la division d'un problème en tâches distinctes permet d'éviter de nombreuses erreurs de conception, d'autre part, une erreur détectée sur le résultat de l'une des tâches devient plus facile à corriger.

Indépendamment de la démarche suivie lors de la production d'un énoncé, il peut s'avérer que celui-ci ne produise pas les résultats escomptés (*e.g.* les algorithmes sont trop longs à exécuter ou leurs résultats sont mauvais, les destinataires sont réfractaires à la métaphore visuelle employée, *etc.*). C'est pourquoi la mise en place de procédés de validation d'une carte est absolument nécessaire. Un **modèle de validation** décrit la démarche à suivre afin de valider les différentes composantes d'une carte. Comme le modèle de production, il permet de découper en tâches le processus de validation et ainsi d'éviter les erreurs, et en cas d'erreur, de déterminer quelle étape du processus de production est en cause.

Comme nous le fait remarquer Munzner [131] de nombreux modèles ont été proposés pour la production et la validation de cartes. Par exemple, le *pipeline* de la visualisation de l'information étudié dans la section précédente peut être vu comme un modèle de production. Cependant, il n'a pas été couplé avec un modèle de validation et c'est pourquoi nous allons dans cette section en utiliser un autre : celui de Munzner qui est, à notre connaissance, le plus abouti.

Dans cette partie, nous allons commencer par décrire le modèle imbriqué de Munzner, puis nous l'appliquerons au système de visualisation décrit dans la section 1.1. Enfin, nous

aborderons ce modèle sous un angle sémiotique en utilisant les concepts définis dans la partie 1.2.

1.3.2.1 Description du modèle imbriqué

La figure 1.19 représente le modèle de production de cartes de Munzner. Nous allons le décrire brièvement. Le lecteur voulant accéder à plus de détails pourra se référer à l'article [131].

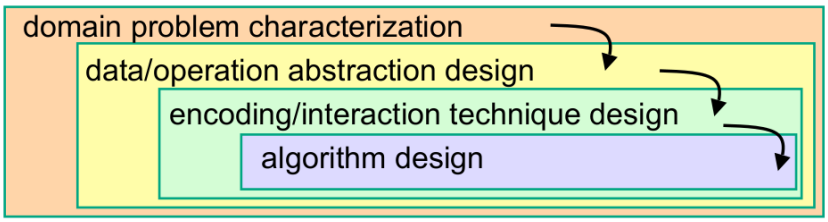


FIGURE 1.19: Modèle imbriqué pour la création de cartes (image provenant de l'article [131]) : **Caractérisation des problèmes du domaine cible (*domain problem characterization*)** : le cartographe doit comprendre les problèmes du domaine des destinataires. **Conception des données et des opérations (*data/operation abstraction design*)** : le cartographe doit transformer les données brutes en données abstraites structurées et exprimer en termes informatiques les problèmes du domaine des destinataires que sa carte doit résoudre. **Conception des techniques de représentation et d'interaction (*encoding/interaction technique design*)** : Le cartographe doit sélectionner une abstraction visuelle des données structurées (*e.g.* variables visuelles, métaphore de représentation, *etc.*) qu'il va utiliser, ainsi que les techniques d'interaction dont sa carte sera munie. **Conception des algorithmes (*algorithm design*)** : trouver les algorithmes permettant de mettre en place les techniques conçues à l'étape précédente.

Tout d'abord, le cartographe doit étudier le domaine de ses destinataires, afin de déterminer quels sont leurs problèmes et comment ces problèmes pourraient être résolus par une carte (*cf.* **caractérisation des problèmes du domaine source** sur la figure 1.19).

Ensuite, comme chaque domaine utilise un vocabulaire particulier, des méthodes d'abstraction particulières, *etc.*, le cartographe doit traduire les problèmes identifiés lors de l'étape précédente en langage informatique et plus particulièrement en langage utilisé en visualisation de l'information (*cf.* **conception des données et des opérations** sur la figure 1.19). Par exemple, imaginons qu'un sociologue demande à un cartographe de lui représenter le réseau social d'un groupe de personnes qu'il étudie afin de déterminer s'il est composé de plusieurs communautés. Ce réseau pourra être modélisé grâce à un graphe, où des sommets représentent des personnes et des arêtes relient des sommets si les personnes correspondantes se connaissent. Le problème de la recherche des communautés pourra quant à lui être assimilé à la recherche de composantes connexes dans le graphe. Nous reviendrons plus en détail sur les graphes comme outil de modélisation au chapitre 2 dans lequel tous ces concepts seront définis formellement.

La conception des techniques de représentation et d'interaction constitue le cœur de l'activité du cartographe. C'est à ce moment qu'il choisit les métaphores visuelles qu'il va utiliser pour représenter les données structurées *i.e.* la façon d'agencer les variables visuelles de façon à ce qu'elles encodent toutes les informations de ces données. C'est aussi lors de cette étape que le cartographe décide des modes d'interaction dont sa carte sera munie. En reprenant l'exemple du réseau social, le cartographe décidera si ce sont des cercles ou des carrés qui représenteront les sommets du graphe, si ce sont des lignes droites ou des lignes courbes qui représenteront les arêtes, si les sommets appartenant à des communautés différentes auront des couleurs différentes, *etc.*. Ils décidera aussi si sa carte sera munie de systèmes de zoom, de filtrage, *etc.*.

Enfin, la **conception des algorithmes** consiste à mettre en place les techniques de génération automatique de la carte en fonction des choix établis lors de l'étape précédente.

Le principal intérêt du modèle de production présenté ci-dessus est qu'il peut être combiné avec un modèle de validation qui explique comment tester chacun des points décris. La figure 1.20 représente ce modèle. A chacune des étapes de production, plusieurs étapes de validation sont associées. Certaines peuvent être menées indépendamment, c'est-à-dire lors de la production associée à l'étape courante. D'autres doivent être menées après la réalisation des productions des étapes suivantes, d'où le nom de modèle imbriqué donné à ce modèle.

La caractérisation des problèmes du domaine cible peut échouer lorsque le cartographe ne comprend pas les problèmes auxquels sont confrontés les destinataires de la carte (**mauvais problème**). Le moyen d'y remédier lors du processus de production consiste à observer et interviewer les destinataires afin d'avoir une idée claire de ces problèmes et ne pas concevoir une carte inutile lors des étapes suivantes (***observe and interview target users***). Une fois le processus de production terminé, le taux d'utilisation de la carte permettra de valider *a posteriori* cette étape (***observe adoption rates***).

La conception des données et des opérations échoue lorsque le cartographe n'arrive pas à transcrire les problèmes des destinataires en problèmes solvables par un système de visualisation de l'information. Cette étape est particulièrement délicate car il n'y a pas de validation possible *a priori*, *i.e.* lors de la production. Seul un test *a posteriori* auprès des acteurs du domaine cible permet de valider l'approche (***test on target users, collect anecdotal evidence of utility***). Afin de mener cela de façon plus rigoureuse, Munzner propose aussi d'observer et de documenter la façon dont les destinataires utilisent le système dans leur travail quotidien (***lab study, document human usage of deployed system***).

La conception des techniques de représentation et d'interaction échoue lorsque les solutions visuelles proposées par le cartographe ne permettent pas de résoudre les problèmes des destinataires formalisés lors de l'étape précédente. Pour éviter ce type d'erreur, la cartographe doit être capable d'argumenter tous les choix qu'il a fait en expliquant ce qu'ils impliquent (***justify encoding/interaction design***). À la fin du processus de production, la validation de cette étape (et donc des arguments du cartographe) devra être menée en vérifiant que les cartes produites par le système permettent effectivement de résoudre les problèmes (***qualitative/quantitative result image analysis***). Pour réaliser cette validation de manière formelle, il faudra tester les temps de réponse aux

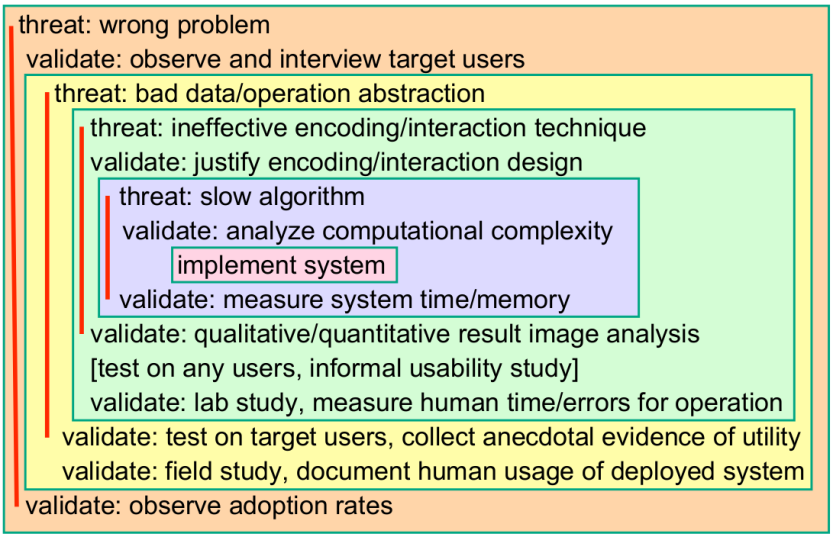


FIGURE 1.20: Modèle imbriqué pour la validation de cartes (image provenant de l'article [131]). Les couleurs correspondent aux étapes de production de la figure 1.19. Pour chacune de ces étapes, un type d'erreur est indiqué ainsi que des méthodes permettant de ne pas les commettre (certaines de ces méthodes doivent être mises en place lors de l'étape de production correspondante, d'autres nécessitent la réalisation des étapes suivantes, d'où le nom de modèle imbriqué) : **Mauvais problème (*wrong problem*)** : les problèmes auxquels sont confrontés les destinataires et que la carte doit aider à résoudre ne sont pas compris du cartographe. **Mauvaise abstraction des données et des opérations (*bad data/operation abstraction*)** : le modèle de données structurées choisi ou les problèmes en termes informatiques identifiés ne permettent pas de résoudre le problème des destinataires. **Techniques de représentation et d'interaction non efficaces (*ineffective encoding/interaction technique*)** : les techniques de représentation et d'interaction ne permettent pas la communication des données abstraites à un destinataire. **Algorithme lent (*slow algorithm*)** : les techniques algorithmiques mises en place sont trop lentes ce qui peut rendre le système inutilisable.

problèmes et les taux d'erreurs auprès d'utilisateurs quelconques (***lab study, measure human time/errors for operation***).

La conception des algorithmes peut échouer si ceux-ci sont trop lents. Il faudra donc évaluer leurs complexités *a priori* (***analyse computational complexity***) et vérifier après implémentation que les temps correspondent et que le système n'est pas trop gourmand en espace mémoire (***measure system time/memory***). Une autre erreur possible lors de cette étape et qui n'apparaît pas sur la figure 1.20 est aussi mentionnée dans le texte de Munzner : le résultat renvoyé par chaque algorithme en fonction de ce que l'on attend de lui peut être faux. C'est pour cela que les algorithmes doivent être analysés de façon minutieuse et que les résultats doivent être regardés avec attention afin de détecter des erreurs éventuelles.

Nous allons maintenant illustrer ce modèle de validation en l'appliquant au système

de visualisation présenté au début de ce chapitre (section 1.1).

1.3.2.2 Application du modèle imbriqué

Cette partie est dédiée à l'application du modèle de validation de Munzner au système présenté dans la section 1.1. À l'exception de la partie de conception des algorithmes, seules les validations *a posteriori* seront menées car les autres sont plutôt des précautions à prendre pendant le processus de production. La validation de la caractérisation des problèmes du domaine cible ne sera pas non plus étudiée car l'observation des taux d'acceptation du produit nécessite un certain recul que nous n'avons pas.

Nous allons commencer par discuter de la conception des algorithmes en analysant leurs complexités en temps et en mesurant les performances du système. Puis, nous évaluerons l'efficacité des techniques de représentation et d'interaction grâce à des tests du système menés par des utilisateurs quelconques (*i.e.* étrangers au domaine des destinataires). Enfin, nous validerons l'abstraction des données et des opérations grâce à des tests menés par des biologistes sur notre système. Les deux catégories de testeurs sont résumées dans le tableau 1.1. Le protocole d'évaluation des deux niveaux intermédiaires du modèle est sommatif (*i.e.* l'évaluation est menée à la fin du processus de production avant le lancement du système), expérimental (*i.e.* l'évaluation est basée sur des connaissances acquises pendant l'utilisation même du système) et non-automatique (*i.e.* les observations sont faites par des observateurs humains).

Catégorie	**Situation des participants**	**Nombre**
Utilisateurs quelconques	Volontaires de la communauté universitaire. Nous nous sommes assurés que chacun d'entre eux avait au moins un master en informatique et n'était pas familiarisé avec la manipulation de puces ADN.	5
Experts du domaine des destinataires	Chercheurs confirmés dans la manipulation et l'analyse de données issues de puces ADN.	2

Table 1.1: Ce tableau résume le nombre des participants ainsi que leurs situations respectives. Afin de mesurer le temps d'apprentissage, nous avons vérifié qu'aucun des participants n'avait auparavant utilisé notre système ou un autre système similaire.

Validation des algorithmes Ce paragraphe étudie les complexités des algorithmes utilisés et les limitations du système.

Complexité du nuage de points Le nuage de points constitue la vue la plus difficile à produire. En effet, une fonction en temps quadratique permettant d'affiner le

placement des centres est itérée jusqu'à convergence. Nous avons testé le temps de cette convergence sur différents ensembles de données aléatoires (voir figure 1.21). Les résultats indiquent qu'aucune amélioration significative ne se fait après 15 itérations. Le temps d'exécution de l'algorithme est donc en $O(n^2)$ avec n le nombre de centres.

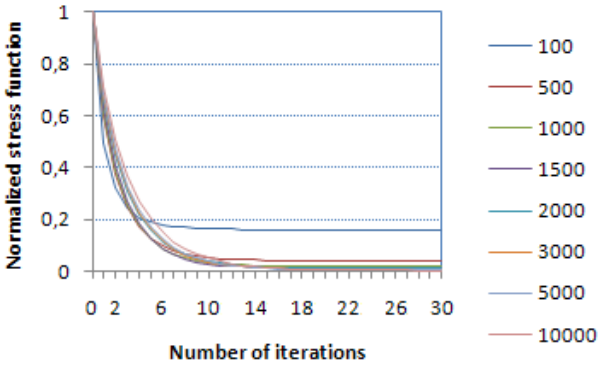


FIGURE 1.21: Convergence de l'algorithme de *multidimensional scaling* utilisé pour placer les centres dans le nuage de points : le nombre indiqué dans la légende correspond au nombre de centres à placer.

Complexités des systèmes solaires et du *treemap* L'algorithme de placement des systèmes solaires est exécuté en temps linéaire. C'est ce même algorithme qui est utilisé pour placer les séquences dans le nuage de points. On est donc dans ce cas en $O(m)$ avec m le nombre total de séquences, ce qui est significatif si $m > n^2$, *i.e.* si le nombre de séquences est supérieur au carré du nombre de centres. En revanche, ce temps est négligeable dans les deux systèmes solaires car le nombre de séquences/documents est plus restreint. L'algorithme du *squarified treemap* s'exécute en temps linéaire en fonction du nombre de séquences affichées. Puisque nous avons limité le nombre de niveaux à 3, ce temps de calcul est faible en pratique.

Limitations de la carte Le système a été développé en *ActionScript 3* pour faciliter sa portabilité et son accès depuis n'importe quelle machine. Dans ces conditions, la complexité du nuage de points empêche l'affichage de plus de 500 groupes. En revanche, il est quand même possible d'afficher jusqu'à 25 000 séquences. Malheureusement, la représentation de plus de 5 000 séquences rend la navigation lente et pénible. Quoiqu'il en soit, des problèmes d'occlusion apparaissent lorsque plus d'un millier d'éléments sont affichés. Par exemple, les figures 1.22 et 1.23 montrent deux jeux de données contenant respectivement 992 et 2726 éléments sur un écran 15,4 pouces avec une résolution de 1680 x 1050 pixels. Le *treemap* permet quant à lui de représenter des arbres contenant plus de 4 000 séquences.

Passons maintenant au paragraphe concernant la validation de l'efficacité des techniques de représentation et d'interaction.

Validation de l'efficacité des techniques de représentation et d'interaction Nous allons tout d'abord présenter le protocole de test puis les résultats obtenus.

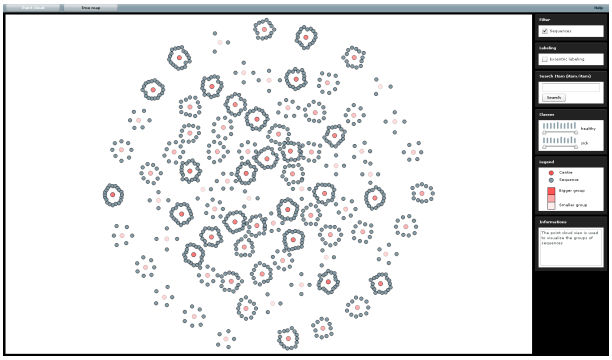


FIGURE 1.22: 992 séquences sont affichées sur un écran de 15,4 pouces avec une résolution de 1680 x 1050 pixels : on n'observe aucun problème important d'occlusion.

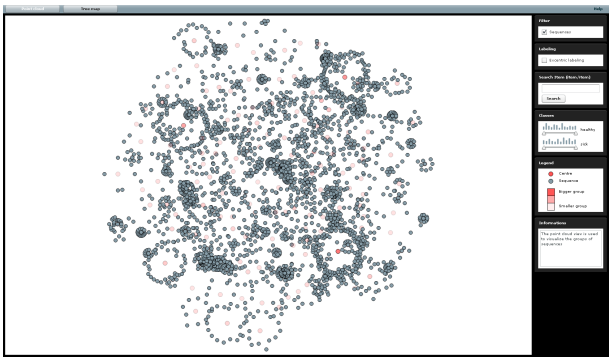


FIGURE 1.23: 2726 séquences sont affichées sur un écran de 15,4 pouces avec une résolution de 1680 x 1050 pixels : des problèmes importants d'occlusion apparaissent.

Protocole Afin d'évaluer les techniques de représentation et d'interaction, un protocole spécifique ainsi qu'un ensemble de mesures inspirés de [157] ont été mis en place. La méthode combine une expérimentation contrôlée avec des techniques de test de l'utilisabilité du système. Elle cherche d'une part à identifier la facilité individuelle que les destinataires ont pour notre système et d'autre part à donner une idée du temps d'apprentissage du système. Nous avons travaillé avec des volontaires universitaires non biologistes en leur donnant seulement quelques indications sur les fonctionnalités principales de l'outil avant le démarrage. Puis ils ont été invités à le manipuler. Ils avaient le droit de poser des questions au sujet du système s'ils n'avaient pas compris le fonctionnement de certains composants. C'était ensuite eux qui stoppaient la manipulation quand ils pensaient avoir fait le tour des fonctionnalités de l'outil. Nous leur donnions à ce moment-là un formulaire à remplir portant sur leur expérience avec l'outil et incluant une description des difficultés et des avantages d'un tel produit. Grâce à ces formulaires, nous avons recueilli 160 résultats portant sur l'utilité et l'utilisabilité des trois visualisations. Les questions sur l'utilité avaient pour objectif d'évaluer la capacité du système à répondre aux besoins de l'utilisateur en fonction de l'effort que celui-ci considérait devoir fournir pour obtenir des résultats. Les questions d'utilisabilité portaient sur la facilité d'utilisation du système. Est-ce que les fonctionnalités sont faciles à utiliser et à mémoriser ? Est-ce que vous y avez vu des erreurs ? Trouvez-vous le système satisfaisant ? *Etc.* Le tableau 1.2 résume

1	Public cible
2	Temps passé à manipuler le système
3	Familiarité avec des techniques de visualisation

Table 1.2: Variables dépendantes.

1	**Nuage de points et premier système solaire (groupes)**
1.1	Aspect général
1.2	Placement des groupes
1.3	Zoom
1.4	Code couleur sur la taille des groupes
1.5	Recherche de séquences contenant des gènes particuliers
1.6	Recherche de séquences en fonction des valeurs de leur support
1.7	Informations sur les séquences (info bulle)
1.8	Efficacité de la représentation sur de gros ensembles de séquences
2	**Deuxième système solaire (documents)**
2.1	Aspect général
2.2	Placement des documents
2.3	Code couleur associé à l'année de publication
2.4	Informations sur les documents (info bulle)
3	***Treemap***
3.1	Aspect général
3.2	Placement des rectangles représentant les séquences
3.3	Code couleur des classes
3.4	Efficacité de l'approche hiérarchique

Table 1.3: Fonctionnalités évaluées.

les variables dépendantes et le tableau 1.3 résume les fonctionnalités étudiées, chacune correspondant à un champ du formulaire rempli par les utilisateurs. Ceux-ci devaient attribuer une note entre 0 et 10 pour chacune de ces fonctionnalités et donner des remarques textuelles afin d'obtenir des détails supplémentaires.

Résultats Le tableau 1.4 montre les moyennes des notes obtenues pour chaque fonctionnalité en terme d'utilité et d'utilisabilité. Étant donné que la méthode d'évaluation est plus qualitative et subjective que quantitative et comme le nombre de participants est limité, une comparaison du niveau général des fonctionnalités est plus appropriée qu'une comparaison du niveau inférieur. Les commentaires des utilisateurs nous permettent de

Fonctionnalités	Moyenne des notes concernant l'utilité (/10)	Moyenne des notes concernant l'utilisabilité (/10)
Nuage de points	6,68	7,46
Système solaire	7,60	7,50
Treemap	6,39	6,53

TABLE 1.4: Moyennes des notes (sur 10) attribuées au système de visualisation (techniques de représentation et de navigation).

compléter ces résultats. En effet, les deux premières fonctionnalités (nuage de points et systèmes solaires) ont obtenu des notes plus élevées car elles sont, d'après ces remarques, plus simples à utiliser et faciles à interpréter. Les utilisateurs nous ont précisé que les informations qu'elles permettaient de visualiser étaient basiques mais utiles. Ils ont moins apprécié le *treemap*, en particulier parce que l'intérêt de la visualisation d'une telle hiérarchie n'était pas claire pour des non biologistes. Cependant, les notes restent positives et le fait que cette dernière représentation donne une vue d'ensemble de la distribution des séquences dans les classes a été souligné comme étant une qualité importante par la plupart des utilisateurs.

Passons maintenant au paragraphe concernant la validation de l'abstraction des données et des opérations.

Validation de l'abstraction des données et des opérations Comme pour l'étape précédente, nous allons tout d'abord présenter le protocole de test puis les résultats obtenus.

Protocole L'évaluation d'un système de visualisation est un procédé complexe. En particulier, lorsque les destinataires de ce système y participent, elle ne doit pas simplement s'intéresser aux aspects humains et techniques mais aussi à l'impact du nouveau système sur leurs pratiques [2]. En ce qui nous concerne, et en accord avec le processus de validation de la conception des données et des opérations du modèle imbriqué, cela consiste à évaluer dans quelles mesures notre outil répond aux attentes des biologistes. Nous avons ainsi mené une évaluation semi-réaliste en collaboration avec des utilisateurs potentiels. Ces experts biologistes ont manipulé des séquences issues de leurs propres jeux de données. En revanche, il est utile de préciser qu'ils avaient l'habitude de mener leurs recherches sur des gènes et non sur des séquences de gènes, d'où le qualificatif de semi-réaliste que nous avons attribué à l'évaluation. Nous avons collaboré avec deux laboratoires afin de sélectionner un ensemble de données pertinent. Le protocole a été défini avec une équipe travaillant sur la maladie d'Alzheimer. Il est basé sur la méthode de la pensée à voix haute (*think aloud*), méthode durant laquelle un observateur demande aux participants d'utiliser le système en expliquant à voix haute le cheminement de son investigation. La variante utilisée ici est aussi coopérative car l'observateur guide, explique

et pose des questions à l'utilisateur. Ceci permet de faciliter la récolte de données importantes pour l'évaluation de la perception que l'utilisateur a des différentes fonctionnalités. En plus des informations collectées oralement, nous avons aussi distribué des questionnaires aux utilisateurs afin qu'ils puissent quantifier leurs impressions après l'utilisation du système (satisfaction, anxiété, *etc.*). Concrètement, l'interview a duré environ trois heures pour chaque biologiste. Avant de commencer, nous leur avons demandé de remplir un questionnaire portant sur leur profil. Puis, comme lors de la précédente évaluation, nous leur avons expliqué brièvement les fonctionnalités du système afin de pouvoir évaluer son « intuitivité ». Ensuite, nous leur avons demandé de mener des tâches basées sur des scénarios réalistes. Pendant le test, un premier observateur était chargé de guider l'utilisateur et d'observer la façon dont il utilisait le système. Un second observateur était quant à lui chargé de noter les réactions (orales ou gestuelles) du biologiste. À la fin du test, les utilisateurs ont rempli un second formulaire centré sur les fonctionnalités déjà évoquées lors de la première évaluation (voir tableau 1.3).

Résultats Les résultats obtenus lors de cette évaluation sont présentés dans le tableau 1.5. Ils révèlent que contrairement aux utilisateurs quelconques, les biologistes ont particulièrement apprécié le *treemap* car la hiérarchie des séquences combinée avec les valeurs des supports de classes leur fournit un outil puissant dans la validation de leurs hypothèses. Le système leur a aussi permis de formuler de nouvelles idées. Par exemple, grâce au nuage de points, les experts ont pu visualiser le gène $A2M$ qui est reconnu comme étant impliqué dans la maladie d'Alzheimer. Ils se sont aussi intéressés à la séquence $S75 =< (MRVI1)(PGAP1)(PLA2R1)(A2M)(GSK3B) >$ qui correspond à des protéines impliquées dans le processus métabolique, sachant que certaines protéines interfèrent avec des événements cellulaires sur les patients atteints par la maladie. Le système s'est donc révélé efficace lors du processus de découverte car il a permis aux experts d'identifier certaines combinaisons de gènes dans les séquences. Il leur a aussi permis de tester leurs hypothèses grâce à l'interface d'accès aux documents (second système solaire). En effet, des séquences regroupant des gènes liés dans la littérature à la maladie d'Alzheimer sont particulièrement intéressantes car elles représentent des configurations qui pourront potentiellement induire de nouveaux axes de recherche. Leurs remarques nous ont aussi permis d'envisager deux axes d'amélioration du système. 1) L'organisation des séquences en groupes selon leur similarité n'a pas suscité de leur part un grand intérêt et d'autres types d'organisation basés sur des mesures de discrimination de séquences par exemple, pourraient leur être plus utiles. 2) Nous devrions inclure d'autres critères pour l'identification des documents les plus importants associés aux séquences (*e.g.* les espèces impliquées dans le document, le type du document, *etc.*). Nous sommes actuellement en train de travailler avec une seconde équipe de biologistes étudiant le cancer du sein afin d'essayer de valider la généralisation de notre approche.

Maintenant que nous avons décrit en détail le modèle imbriqué de production et de validation de cartes et que nous avons étudié un exemple de sa mise en pratique lors du processus de validation, nous allons pouvoir en discuter selon l'approche sémiotique introduite en début de chapitre.

Fonctionnalités	Moyenne des notes concernant l'utilité (/10)	Moyenne des notes concernant l'utilisabilité (/10)
Nuage de points	6.00	6.34
Système solaire	7,5	7.40
Treemap	7.76	7.00

TABLE 1.5: Moyennes des notes (sur 10) attribuées au système de visualisation (abstraction des données et des opérations).

1.3.2.3 Approche sémiotique du modèle imbriqué

Nous allons maintenant confronter le modèle de Munzner aux résultats sémiotiques présentés plus haut. Pour cela, nous allons reprendre chacune des dimensions évoquées dans la partie 1.2.3 car, comme nous l'avons mentionné, elles impliquent les résultats obtenus dans les parties 1.2.1 et 1.2.2. Ainsi, nous allons montrer comment chacune des étapes du modèle imbriqué correspond principalement [33] à une de ces dimensions. Bien que les résultats ci-après ne soient qu'une ébauche certainement discutable de la mise en correspondance des modèles (modèle sémiotique tridimensionnel et modèle imbriqué), ils permettent néanmoins de voir que les résultats sémiotiques sont plus ou moins corrélés avec ceux obtenus de façon indépendante en visualisation de l'information, ce qui peut être vu comme un argument en faveur de ces deux modèles.

Avant de commencer, nous pouvons déjà remarquer que le modèle sémiotique n'inclut pas la dimension algorithmique présentée par Munzner (dernier niveau du processus de production). Ceci est tout à fait normal dans la mesure où il est défini pour la sémiotique générale (*i.e.* applicable à n'importe quelle sémiotique particulière) et que l'utilisation de l'outil informatique est spécifique à la visualisation de l'information. Dans le cadre de la sémiotique particulière « visualisation de l'information », il faudrait donc rajouter l'étude d'une dimension algorithmique. Celle-ci devrait étudier la faisabilité des cartes proposées par la dimension syntaxique. Une autre alternative pourrait être d'inclure les problèmes algorithmiques dans cette partie syntaxique. Ce n'est cependant pas l'approche que nous préconisons car nous estimons que les types de problèmes soulevés par la syntaxe telle qu'elle a été décrite plus haut sont trop différents de ceux soulevés par la faisabilité des cartes.

Dimension syntaxique Comme nous l'avons déjà mentionné plus haut (section 1.2.3.1), étudier les signes selon leur dimension syntaxique revient à s'intéresser aux unités sémiotiques et la façon dont elles se combinent pour former des cartes. Lors du processus de production tel que Munzner le propose, le choix des variables visuelles (nom donné aux signes plastiques en visualisation de l'information, *i.e.* les unités de la sémiotique visuelle

33. Nous employons ici le terme « principalement » car, comme le modèle de Munzner est imbriqué, les parties ne sont pas indépendantes les unes des autres, *i.e.* chaque étape fait indirectement appel aux dimensions sémiotiques traitées lors des étapes précédentes.

symbolique) et de leur mode d'articulation (type de carte que l'on veut produire) est réalisé lors de l'étape de conception des techniques de représentation et d'interaction. C'est pour cela que le processus de validation de cette étape ne nécessite pas l'intervention d'experts du domaine des destinataires. En effet, si l'on fait l'analogie avec la linguistique, il est inutile, par exemple, de faire appel à un expert en informatique afin de rechercher les fautes grammaticales de la thèse que vous êtes en train de lire. Il en va de même en visualisation de l'information : n'importe quelle personne connaissant le code utilisé peut valider la syntaxe de la carte. Cependant, étant donné que la grammaire de la visualisation n'est pas définie par des règles aussi précises que celles de la linguistique, l'utilisateur ne peut pas lui même corriger les erreurs syntaxiques. Le seul moyen que nous avons est donc de lui poser des problèmes relatifs à la syntaxe et d'évaluer les taux d'erreurs. Par exemple, sur le système de visualisation présenté dans la partie 1.1, on peut lui demander de retrouver le groupe des séquences dans lequel un gène particulier apparaît le plus de fois grâce à une carte du type de celle montrée par la figure 1.5. Cette question relève bien de la syntaxe car c'est grâce à la combinaison des éléments que l'utilisateur peut distinguer les groupes et c'est grâce au système de mise en surbrillance de gènes recherchés (chromène /vert/) qu'il peut sélectionner le groupe contenant le plus de fois un certain gène. En revanche, on peut considérer que l'opération ne relève pas de la sémantique car à aucun moment l'utilisateur n'a besoin de savoir ce que signifie le mot /gène/ ni à quel référent il renvoie. Il a juste besoin de savoir qu'un ensemble de /gènes/ constitue une /séquence de gènes/ qui est traduite par un /rond gris/ dans la carte, *etc.*.

Dimension sémantique L'étape concernant l'abstraction des données et des opérations peut être considérée comme celle relevant majoritairement de la dimension sémantique. En effet, c'est la phase durant laquelle on attribue une signification à des modèles de données abstraits et structurés ainsi qu'à des problèmes relevant de la visualisation de l'information en les mettant en correspondance avec des données et des problèmes du domaine des destinataires : on réalise donc une simiosis, *i.e.* on unit le plan de l'expression avec celui du contenu. La validation de cette étape nécessite l'intervention d'experts du domaine des destinataires connaissant le contenu et auxquels on va enseigner le code de l'expression afin de voir si la semiosis se réalise. Pour être plus précis, on vérifie que le code permet bien d'établir un lien entre le signifié et le signifiant auprès des experts capables d'établir le lien entre le signifiant et le référent. Cette vérification relève bien de la sémantique et non de la pragmatique si le destinataire est éduqué correctement, *i.e.* s'il sait bien manipuler le code et arrive ainsi à écarter toute ambiguïté contextuelle. Dans l'exemple que nous avons fourni dans la section 1.3.2.2, nous n'avons pas mis l'accent sur cette éducation afin de prendre aussi en compte une partie de la dimension pragmatique mais nous reconnaissons qu'il y a dans ce choix la possibilité d'avoir fait une erreur conceptuelle. En effet, tout d'abord, la validation de la signification non contextuelle (sémantique) nous semble être nécessaire pour mener à bien la validation de la signification contextuelle (pragmatique) [34]. Ensuite, le fait de mélanger les deux approches pose des problèmes de rigueur scientifique : pourquoi 10 minutes d'éducation et non pas 15 ? Enfin, un petit échantillon de personnes n'est pas suffisant pour la validation de la pragmatique d'une carte dans un certain milieu

34. Ceci est d'ailleurs induit par l'imbrication des niveaux dans le modèle de Munzner.

car les contextes personnels peuvent prendre le pas sur le contexte général de l'ensemble des destinataires.

Dimension pragmatique Nous en arrivons donc à la dimension pragmatique, qui tient compte des compétences cognitives communes à n'importe quel individu ainsi que des faits sociaux relatifs au milieu des destinataires (contexte) afin de valider la capacité de la carte à résoudre les problèmes réels de ces destinataires, *i.e.* le potentiel de la carte à être source d'action. Munzner propose pour cela d'observer le taux d'adoption de la carte. Bien que cette méthode soit la plus efficace, on pourrait néanmoins en proposer une alternative basée sur des méthodes statistiques. Le principe serait de fournir la carte à un échantillon représentatif de destinataires et d'observer le taux d'utilisation sur cet échantillon. La sélection d'un tel échantillon n'est cependant pas simple, de nombreuses techniques ont été proposées [156] et une étude plus approfondie de celles-ci serait nécessaire afin de les adapter au domaine de la visualisation et ainsi pouvoir déterminer lesquelles sont les plus efficaces pour celui-ci.

1.4 Conclusion

L'étude menée dans ce premier chapitre est une première ébauche d'une approche sémiotique de la visualisation de l'information. Afin de pouvoir la réaliser, nous avons dû, d'une part, éviter les polémiques qui agitent le monde de la sémiotique, voire de la philosophie et, d'autre part, se contenter de certains concepts fondamentaux sans trop entrer dans les détails. Une approche aussi simple, voire simpliste, nous a quand même permis d'obtenir quelques résultats intéressants, ce qui pourrait justifier la mise en place de travaux futurs plus ambitieux dans cette voie. Entre autres, nous avons pu établir une distinction nette entre l'art et la science de la visualisation de l'information et trouver une place à ce domaine de recherche parmi les sciences sémiotiques. Ceci nous permettra peut-être d'emprunter des lois utiles aux domaines de rang supérieur [35], d'en découvrir de nouvelles en généralisant les nôtres, d'en trouver des contre-exemples ou d'en confirmer. Par exemple, le fait que l'on retrouve des caractéristiques proches entre le modèle sémiotique tridimensionnel et le modèle imbriqué de Munzner (tous d'eux ayant été vraisemblablement élaborés de manière indépendante) semble être un argument pour leur adoption aussi bien en sémiotique générale qu'en visualisation de l'information.

Le principal point faible de l'étude telle qu'elle a été faite est que nous n'avons pas abordé le problème de l'interaction, élément si crucial de la recherche en visualisation de l'information. Ceci est principalement dû au fait que ce genre de question a été très peu étudié par les sémioticiens car la plupart des énoncés sur lesquels ils se penchent ne sont

35. On peut voir la sémiotique comme une hiérarchie dont l'élément le plus général correspond à la sémiotique générale et un élément quelconque est une sémiotique particulière qui hérite des lois de ses prédécesseurs et en énonce d'autres qui lui sont particulières. Ainsi, la visualisation de l'information hérite des lois de la graphique, qui elle-même hérite des lois de la sémiotique visuelle, qui elle-même hérite des lois de la sémiotique générale. Les lois de Bertin s'appliquent à notre domaine (graphique). Les symboles plastiques évoqués dans la section 1.2.3.1 nous viennent quand à eux de la sémiotique visuelle. Enfin, le découpage tridimensionnel remonte à la sémiotique générale.

pas interactifs. Il semblerait d'ailleurs que ce type de problèmes soit spécifique à notre domaine. Une approche qui nous semblerait correcte serait de considérer un système de visualisation comme un ensemble de cartes (potentielles) et un ensemble d'« interacteurs » permettant de naviguer dans ses cartes grâce à la modification, la suppression ou l'ajout d'unités. Une question reste cependant ouverte : est-ce que cette partie relève de la syntaxe comme nous l'avons suggéré en rapprochant le troisième niveau du modèle de Munzner à la dimension syntaxique ou est-ce une nouvelle dimension ?

En espérant que ce premier chapitre aura amené le lecteur à aborder la visualisation de l'information sous un angle différent, nous allons maintenant passer au second chapitre de cette thèse qui porte sur divers travaux plus conventionnels réalisés au cours du doctorat. Ils concernent la visualisation de graphes, sous-domaine de la visualisation de l'information dont nous donnerons une définition.

Chapitre 2
Visualisation de graphes

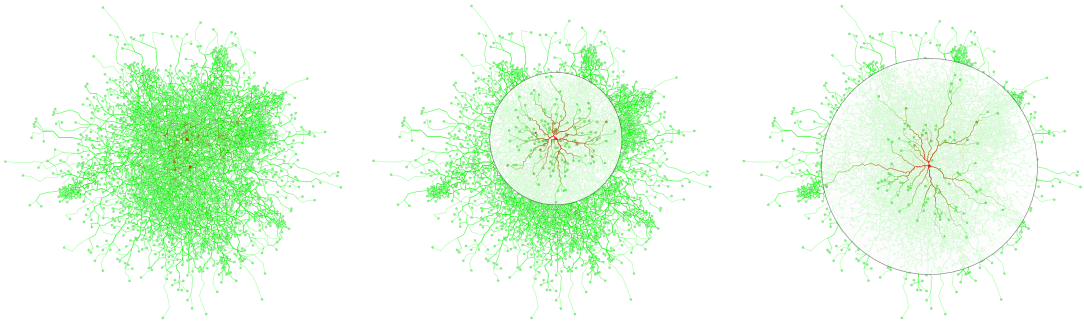
Intuitivement, un graphe correspond à un ensemble de nœuds reliés entre eux par des liens. Plus formellement, un **graphe** G est une paire (V, E) où V est un ensemble d'éléments appelés **sommets** et E est un ensemble de paires de V. Ces paires peuvent être ordonnées, ou non. Dans le premier cas, on parle d'**arcs** et le graphe est dit **orienté**, dans le second cas on parle d'**arêtes** et le graphe est dit **non orienté**. D'un point de vue pratique, les graphes sont des outils puissants permettant de modéliser et de résoudre de nombreux problèmes de la vie courante. Le lecteur curieux pourra se référer à l'introduction simple et pédagogique de Chartrand [42] pour s'en faire une idée. Le domaine de la théorie des graphes connaît ainsi un énorme succès (voir [24] pour une vue d'ensemble assez complète des résultats du domaine).

Voyons maintenant comment un tel outil pourrait être combiné avec la visualisation de l'information afin de résoudre (visuellement) certains problèmes. L'exemple qui suit est tiré d'une étude menée avec la société Pikko. Les sommets du graphe non orienté manipulé correspondent aux membres d'une communauté du réseau social professionnel *viadeo*. Deux sommets sont liés par une arête si les individus correspondants sont en contact. Afin de préserver l'anonymat, nous ne divulgueront ni les noms des personnes, ni le nom de la communauté étudiée. Le graphe de la figure 2.1.a représente cette communauté.

Un problème auquel nos clients sont confrontés consiste à identifier les personnes influentes d'un réseau. Par exemple, il peut être utile de connaître les individus les plus susceptibles de les introduire dans une communauté, de diffuser leurs nouvelles offres, *etc.*. Ce problème est difficile à traiter car il faudrait répondre à de nombreuses questions : Qu'est-ce qu'une personne influente ? Comment une information se propage-t-elle dans un réseau ? Est-ce que cette propagation est la même pour tout type d'information ? *Etc.*. Nous allons cependant voir maintenant à l'aide d'un petit exemple comment la visualisation de l'information peut aider à trouver des éléments de réponse. Dans le graphe 2.1.a, nous avons calculé un indice bien connu de l'étude des réseaux sociaux [71] permettant d'identifier les éléments centraux d'un graphe [1]. Les sommets rouges représentent les individus dont l'indice est le plus élevé. La figure 2.1.b représente le même graphe dans lequel nous avons mis en surbrillance les contacts d'un sommet dont l'indice est très élevé. Comme nous pouvions nous y attendre, le nombre de ses contacts est élevé. L'approche visuelle nous permet cependant de tirer une seconde conclusion. Les contacts

1. Plus d'indications sur cet indice seront données ultérieurement (section 2.1.2.2).

de l'individu sont aussi très « ramassés », on observe clairement que l'individu a peu de contacts avec les personnes se trouvant en périphérie du réseau. Au contraire, dans la figure 2.1.c, qui représente aussi les contacts d'un individu dont l'indice est élevé, on observe que ces contacts se répartissent mieux sur l'ensemble du réseau. Cette approche visuelle nous permet donc d'émettre l'hypothèse que ce second individu est peut-être plus influent sur le réseau pris dans sa totalité.



(a) (b) (c)

FIGURE 2.1: (a) Réseau social *viadeo*. (b) Mise en évidence des contacts d'un premier individu. (c) Mise en évidence des contacts d'un second individu : ceux-ci se répartissent de façon plus homogène dans l'ensemble du réseau.

Ce cas d'école, exemple simplifié issu de notre expérience dans le milieu de l'industrie, nous montre donc en quoi la visualisation de graphe peut nous aider à résoudre certains problèmes (voir [94] pour une introduction à ce type de visualisation). Dans ce qui suit, nous allons donner deux exemples de visualisation de l'information basés sur les graphes. Le premier s'attaque au problème de la navigation dans un ensemble de pages *web* retournées par un moteur de recherche, le second propose une alternative aux visualisations standards de données hiérarchisées permettant d'observer leur évolution dans le temps.

2.1 Visualisation et navigation interactive dans un ensemble de pages *web*

Le premier exemple de méthode de visualisation de l'information que nous allons présenter ici a été réalisé dans le cadre du projet ANR RNTL FIVE 06 TLOG 12 regroupant la société Pikko, la société AMI Software, le LaBRI et l'INRIA Bordeaux Sud-Ouest. Ce projet avait pour objet la conception et la réalisation de composants de visualisation et de navigation pour la veille concurrentielle, *i.e.* la mise en place de moyens cartographiques permettant d'aider la surveillance des informations publiées, diffusées ou disponibles sur des concurrents. La veille concurrentielle est en effet un outil indispensable au bon développement d'une entreprise. Comme le précise une récente publication du MEDEF, la maîtrise de l'information est devenue un enjeu stratégique :

« Crises, accélération des mutations économiques et sociales, globalisation. Le monde est de plus en plus complexe. Le dirigeant d'entreprise rencontre des difficultés croissantes pour garder le cap. Ses traditionnels instruments de navigation deviennent obsolètes. Entre

tempête et brouillard, la maîtrise de l'information devient un enjeu stratégique pour éviter les écueils. » [50]

Or, comme nous le verrons à travers notre exemple, la visualisation peut être utilisée afin d'aider les veilleurs. En effet, le développement des nouvelles technologies de l'information a rendu leur tâche immense, ils sont tous les jours confrontés à de grandes quantités de documents et l'accès aux informations utiles qu'ils contiennent devient difficile. Dans ce contexte, des méthodes de classification et de visualisation peuvent les aider à faire la part entre ce qui leur sera utile et le reste en accélérant l'accès aux informations [111].

Dans notre travail, nous nous sommes intéressés aux données provenant du *web*. Les moteurs de recherche tels que *Google*, qui permettent d'accéder aux pages internet traitant d'un sujet particulier, nous renvoient une longue liste de résultats. L'utilisateur doit alors ouvrir chacun d'entre eux afin de déterminer s'il contient les informations qu'il recherche, ce qui peut devenir long et fastidieux. L'organisation et la visualisation de ces résultats devient donc un enjeu primordial afin d'accélérer l'accès aux informations [25].

Par exemple cherchons le mot *jaguar* sur *Google*. La figure 2.2 montre une capture d'écran des sept premiers résultats renvoyés. Comme on peut le voir, les pages 1, 2, 3 et 6 parlent de voitures, les pages 5 et 7 de l'animal et la page 4 de l'ordinateur *jaguar*. Plus loin, nous pourrons trouver aussi des pages sur un groupe musical, une marque de guitare, *etc.*.

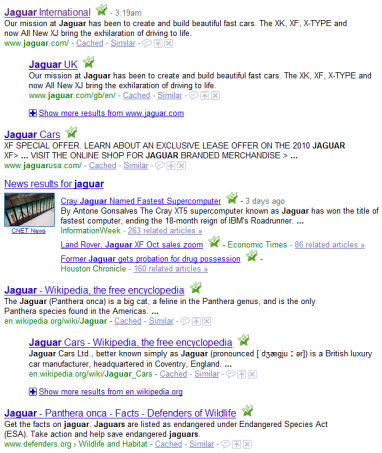


FIGURE 2.2: Capture d'écran des sept premiers résultats renvoyés par *Google* lorsque l'on cherche le mot *jaguar*.

Idéalement, nous voudrions que le moteur regroupe les pages traitant des mêmes sujets afin de pouvoir directement orienter l'utilisateur vers les informations qu'il recherche, comme nous l'avons fait sur la figure 2.3. Un tel système doit donc utiliser les mots-clés des pages afin de les regrouper et d'aider l'utilisateur à s'orienter sans avoir à les parcourir. C'est pourquoi nous sommes partis du graphe de co-occurrences des mots-clés des pages renvoyées par un moteur de recherche.

Un graphe de co-occurrence de mots-clés est un graphe non orienté $G = (V, E)$ où chaque élément de V représente un mot-clé et où il existe une arête (u, v) entre un mot-clé u et un mot-clé v s'ils apparaissent ensemble dans au moins une page internet. La

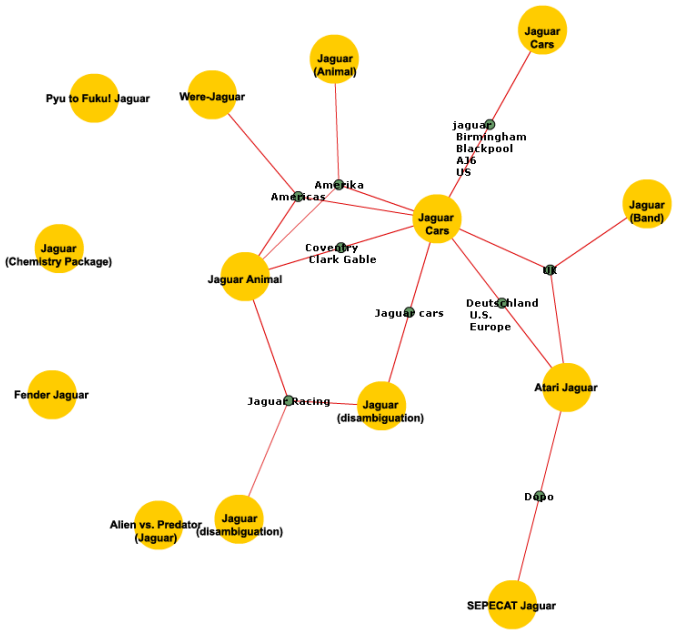


FIGURE 2.3: Visualisation de pages retournées par un moteur de recherche avec la requête *jaguar* et regroupées selon le sujet qu'elles traitent (*clusters* jaunes).

figure 2.4 montre un tel graphe et comme vous pouvez l'observer, un algorithme simple de dessin de graphe comme *GEM* [72] n'est pas suffisant pour l'afficher de façon à ce que l'utilisateur puisse y retrouver les zones traitant des sujets qu'il recherche [2] cette façon est complexe, *i.e.* il partage deux types de propriétés connues sous le nom de **petit-monde** [196] et de **sans-échelle** [12] comme le montrent Cancho et Solé [100].

Afin de bien comprendre quels sont les problèmes relatifs au dessin de tels graphes, nous allons définir ces deux notions plus en détail. La notion de petit-monde nécessite d'abord quelques définitions préliminaires. Le **coefficient de *clustering* d'un sommet** est une valeur attribuée à chaque sommet du graphe et qui représente le nombre de cycles de longueur 3 auquel le sommet appartient divisé par le nombre de cycles de longueurs 3 possibles. Plus formellement, la formule suivante permet de la calculer :

$$C(v) = \frac{r(N(v))}{|N(v)|(|N(v)| - 1)/2}$$

Ici, $N(v)$ dénote l'ensemble des sommets **adjacents** au sommet v, *i.e.* l'ensemble des sommets u tels que $(v, u) \in E$. $r(N(v))$ est quant à lui le nombre d'arêtes reliant deux sommets du voisinage de v, *i.e.* l'ensemble $\{(u, w)|u, w \in N(v)\}$. On peut maintenant définir le **coefficient de *clustering* d'un graphe** $G = (V, E)$ comme la moyenne des coefficients de *clustering* de ses sommets :

2. Il en est de même pour tous les algorithmes de placement basés sur des analogies avec le monde physique et couramment utilisés pour le positionnement des sommets d'un graphe quelconque (*e.g.* celui de Eades [57], celui de Fruchterman et Reingold [73], celui de Kamada et Kawai [107], FM^3 [90] ou encore GRIP [75] pour ne citer que les plus connus).

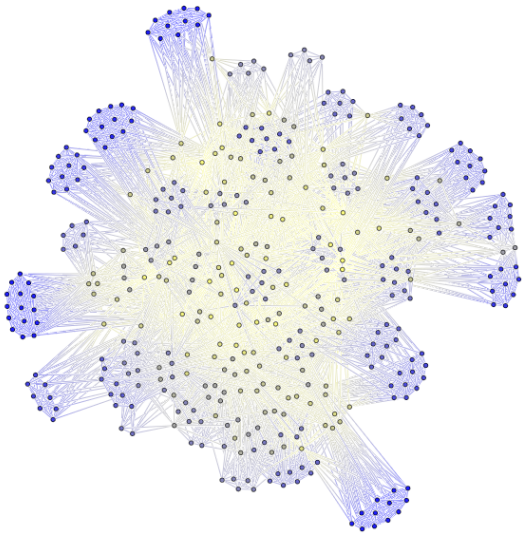


FIGURE 2.4: Exemple de graphe de co-occurrence de mots-clés affiché avec l'algorithme *GEM* [72].

$$C_G = \frac{\sum_{v \in V} C(v)}{|V|}$$

Soit SMG un graphe petit-monde et RG un graphe aléatoire, *i.e.* un graphe contenant n sommets et m arêtes choisies aléatoirement parmi les paires de sommets (modèle de génération proposé par [64]). Alors, d'après la définition de Watts et Strotgatz, une première propriété du graphe petit-monde est d'avoir un coefficient de *clustering* très supérieur à celui d'un graphe aléatoire de même taille (i.e. ayant le même nombre de sommets et le même nombre d'arêtes) : $C_{SMG} \gg C_{RG}$.

Les graphes petits mondes possèdent aussi une seconde propriété basée sur la notion de distance. Commençons par définir un **chemin** entre deux sommets u et v d'un graphe : c'est une suite $P = (u, w_1, w_2, ..., w_k, v)$ telle que $(u, w_1) \in E$, $(w_k, v) \in E$ et $\forall i \in [1, k-1], (w_i, w_{i+1}) \in E$. Parmi tous les chemins possibles $\mathcal{P}_{uv}$ entre deux sommets u et v , on peut en extraire un dont la taille est minimale, *i.e.* le nombre de sommets qu'il contient est inférieur ou égal au nombre de sommets que contiennent les autres chemins. Un tel chemin est appelé **plus court chemin** entre u et v, nous le nommerons PCC_{uv}. Prenons d_{uv} le nombre de sommets de PCC_{uv}. Cette valeur est la **distance** entre u et v. Elle permet de calculer la **distance moyenne** des paires de sommets d'un graphe $G = (V, E)$:

$$D_G = \frac{\sum_{u,v \in V, u \neq v} d_{uv}}{n(n-1)}$$

En plus de la propriété concernant le coefficient de *clustering* énoncée ci-dessus, un graphe petit-monde est aussi caractérisé par cette moyenne des distances. En effet, celle-ci doit être proche de celle d'un graphe aléatoire de même taille : $D_{SMG} \approx D_{RG}$.

Comme nous l'avons dit ci-dessus, le graphe de co-occurrence est aussi sans échelle. Soit $deg(v) = |N(v)|$ le **degré** d'un sommet, *i.e.* la taille de l'ensemble du voisinage

du sommet. Un graphe est appelé sans échelle lorsque la distribution des degrés de ses sommets suit une loi de puissance (voir figure 2.5). De façon moins formelle, la plupart des sommets de ce type de graphes a peu de voisins, mais quelques sommets ont aussi un très grand nombre de voisins.

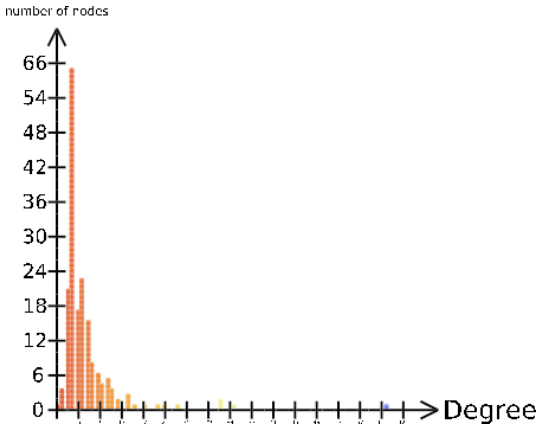


FIGURE 2.5: Distribution des degrés des sommets d'un graphe sans échelle : l'axe des abscisses correspond aux degrés, celui des ordonnées au nombre de sommets.

La visualisation des graphes sans échelle est difficile [103]. La méthode que nous allons présenter permet de trouver une solution à ce problème. Elle est basée sur la combinaison d'une méthode de fragmentation (nous définirons en détail cette notion dans la section 2.1.2), d'une méthode de visualisation des groupes de mots-clés ainsi formés et d'un système de navigation dans ces groupes et dans les pages internet contenant les mots-clés.

Ce travail a été réalisé en collaboration avec Faraz Zaidi et Guy Melançon du LaBRI ainsi qu'avec Christian Pich de l'*ETH* Zurich [3]. Il a fait l'objet de deux publications. Le premier article présente la méthode de fragmentation [205] et le second les techniques de visualisation et de navigation mises en place [152]. Le lecteur plus intéressé sur l'étude de ce type de réseaux pourra aussi se référer à la thèse de doctorat de Faraz Zaidi [204].

Nous allons commencer par revenir sur les méthodes de navigation à travers des pages internet déjà disponibles dans la littérature (section 2.1.1). Puis nous donnerons en détail l'algorithme de fragmentation (section 2.1.2) et les techniques de visualisation (section 2.1.3). Pour finir, nous donnerons des justifications concernant la démarche adoptée en se basant sur des exemples de résultats obtenus (section 2.1.4).

2.1.1 Travaux précédents

Les outils de navigation internet faisant appel à des moteurs de recherche peuvent être groupés en deux catégories : les systèmes basés sur des listes et ceux basés sur des méthodes de représentation graphique. Les premiers consistent à afficher des listes ordonnées de pages web avec des artifices visuels tels que la mise en gras de mots-clés [113] ou avec une hiérarchie permettant de filtrer certains éléments de la liste regroupés selon leur contenu [206, 202]. Les systèmes basés sur des environnements graphiques représentent quant à

3. http ://www.ethz.ch/

eux les résultats renvoyés par un moteur de recherche sous forme de cartes en deux [135] ou trois [25] dimensions. Des comparaisons entre ces deux approches ont déjà été menées mais aucune preuve de l'efficacité de chacune d'entre elles n'a été proposée et le problème reste donc ouvert [8]. Le système que nous allons présenter dans cette section est basé sur des représentations graphiques et c'est pourquoi nous allons maintenant voir plus en détail les différentes solutions entrant dans cette catégorie et déjà disponibles dans la littérature.

WebSearch Viz [135] est un système graphique basé sur la métaphore du système solaire. La requête lancée au moteur de recherche est placée au centre de la visualisation et chaque page i renvoyée est placée selon des coordonnées polaires (ρ_i, θ_i). La distance ρ_i dépend de la pertinence avec le sujet. Cette pertinence est calculée grâce à une mesure de similarité basée sur des vecteurs de mots-clés. L'angle θ_i dépend quant à lui de la proximité de la page avec certains mots-clés placés à l'extérieur du système solaire. Ainsi, les secteurs de ce système contiennent des pages proches sémantiquement les unes des autres. Malheureusement, cette méthode ne permet pas d'afficher un très grand nombre de mots-clés et ne permet donc pas de résoudre les problèmes liés aux propriétés des réseaux de co-occurrence.

LightHouse [122] est un système de navigation intégrant à la fois une représentation sous forme de liste des résultats et une visualisation graphique. Cette dernière utilise des sphères pour représenter les pages et la distance qui les sépare dénote leur proximité sémantique (elles peuvent aller jusqu'à se chevaucher si les thèmes qu'elles abordent sont vraiment très proches). La principale limitation de cette approche est la faible quantité de pages pouvant être affichées. En effet, des problèmes d'occlusions rendant la carte illisible apparaissent rapidement lorsque le nombre de résultats augmente. De plus, contrairement au système que nous allons vous présenter, l'utilisateur ne peut pas savoir quels sont les thèmes articulant les pages entre elles.

Il existe aussi de nombreux moteurs de recherche utilisant la visualisation afin d'aider l'utilisateur à naviguer à travers les pages internet. On pourrait citer par exemple *Kartoo* ou *WebBrain*. A notre connaissance, aucun d'entre eux ne propose de système efficace muni à la fois d'une technique de fragmentation performante et d'un système de navigation clair sans ambiguïté ni occlusion. Pour en savoir plus, l'article [135] propose une vue d'ensemble assez complète de ces sites.

2.1.2 Fragmentation

Une méthode de **fragmentation** d'un graphe $G = (V, E)$ permet de trouver une **partition** de l'ensemble des sommets de ce graphe, *i.e.* un ensemble $P = \{p_0, p_1, ..., p_k\}$ tel que $\bigcup_{i=1}^{k} p_i = V$ et $\forall p_i, p_j \in P, p_i \cap p_j = \varnothing$. Une telle partition permet de construire un **graphe quotient** $Q(G) = (V_{Q(G)}, E_{Q(G)})$ où $V_{Q(G)} = P$ et $(p_i, p_j) \in E_{Q(G)}$ si et seulement si $\exists (u, v) \in E$ tel que $u \in p_i, v \in p_j$. Un graphe ainsi construit est particulièrement intéressant en visualisation de l'information car il permet de simplifier le graphe d'origine et grâce à cela dessiner de façon lisible les informations qu'il contient. Si le lecteur désire avoir une vue d'ensemble des méthodes de fragmentation de graphe, l'article de Schaeffer est un excellent point de départ [160].

La mise en place d'une telle méthode sur le graphe de co-occurrence de mots-clés pourrait donc nous permettre de faciliter la visualisation des résultats retournés par un

moteur de recherche. Comme nous l'avons précisé plus haut, les graphes que nous traitons ici sont petit-monde et sans échelle, ce qui est d'ailleurs le cas de nombreux graphes issus du monde réel [4] comme le réseau des acteurs extrait de l'*Internet Movie Data Base* [5] ou le réseau des co-auteurs de *DBLP* [6]. À cause de la structure sans échelle, on sait que ces réseaux sont difficiles à fragmenter [7] et à visualiser [103] car certains sommets ont un degré très grand (voir figure 2.5). D'un autre côté, la propriété petit-monde nous permet de formuler l'hypothèse selon laquelle il existe des zones dans le graphe pour lesquelles la densité des arêtes est très forte. Ceci laisse supposer la présence de **communautés** d'éléments, *i.e.* d'ensembles de mots-clés apparaissant souvent dans les mêmes documents et rarement les uns sans les autres. Ce sont ces ensembles qui sont intéressants pour nous car ils permettent de caractériser des groupes de documents portant sur des sujets communs. C'est donc eux que nous cherchons à découvrir avec notre méthode de fragmentation.

Cette méthode, qui fera l'objet des deux prochaines sous-sections, combine deux techniques : la première, basée sur la duplication de sommets, permet de modifier le réseau afin de supprimer la propriété sans échelle (section 2.1.2.1) ; la seconde consiste à fragmenter le graphe ainsi modifié afin de déterminer des sous-groupes plus denses et en extraire des sommets « ponts », notion que nous définirons plus tard (section 2.1.2.2). L'image 2.6 donne une vue d'ensemble du processus, nous préciserons au fur et à mesure comment chacun de ses éléments a été construit.

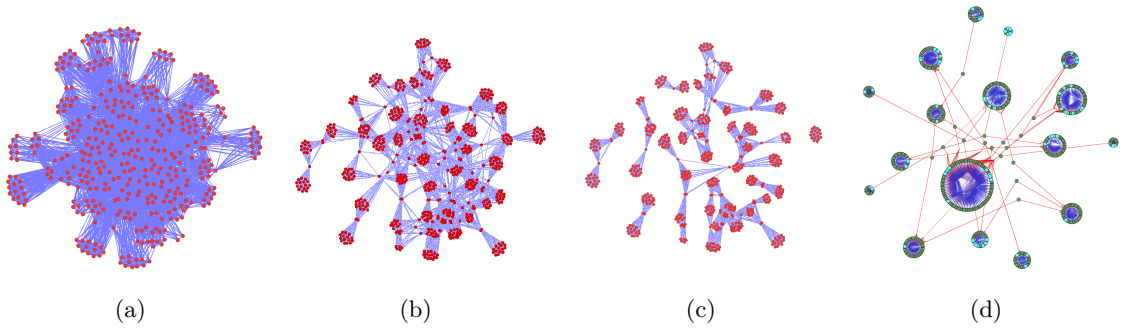


FIGURE 2.6: (a) Graphe de co-occurrence initial. (b) Graphe après duplication des sommets. (c) Graphe après suppression des ponts. (d) Graphe avec groupes et ponts.

4. Plus précisément, les graphes de ce type sont extraits de graphes bi-partis ce qui peut expliquer leur topologie particulière. Une description plus détaillée serait hors du périmètre d'étude de cette thèse mais le lecteur intéressé pourra se référer à l'article [89] de Guillaume et Latapy.

5. http ://www.imdb.com/

6. http ://www.informatik.uni-trier.de/ ley/db/

7. Une méthode de fragmentation très utilisée sur les graphes sans échelle est connue sous le nom de décomposition *k-core* [161, 14]. En simplifiant, elle permet de répartir les sommets en groupes en fonction de leur voisinage. On obtient ainsi une hiérarchie de groupes ou les sommets de même importance appartiennent aux même groupes. Nous n'utilisons pas ici cette méthode car elle ne tient pas compte du caractère petit-monde du réseau, *i.e.* les communautés sous-jacentes.

2.1.2.1 Duplication des sommets

Le graphe de co-occurrence initial est représenté par la figure 2.6.a. Comme on peut l'observer, quelques communautés périphériques peuvent apparaître visuellement en périphérie avec un simple algorithme de placement (*GEM* [72] dans notre exemple) mais les communautés sous-jacentes au cœur du réseau sont indiscernables. C'est pourquoi nous proposons ici de dupliquer les sommets représentant des mots-clés présents dans un grand nombre de documents (un seuil peut être déterminé empiriquement pour des jeux de données issus d'un même moteur de recherche et de même taille). Notre démarche est motivée par le fait que ces mots-clés apparaissant dans la plupart des pages ne sont, au final, pas très pertinents pour notre fragmentation car ils ne sont pas spécifiques à une communauté particulière. Nous pourrions d'ailleurs tout aussi bien les supprimer mais nous préférons garder toutes les informations dans la carte finale.

L'image 2.7 illustre ce que nous entendons par **duplication**. Imaginons qu'un sommet soit présent dans trois documents. Alors ce sommet appartiendra à trois **cliques**[8] du graphe de co-occurrence. Nous dupliquons donc ce sommet en trois sommets, chacun étant relié à des mots-clés apparaissant dans le même document.

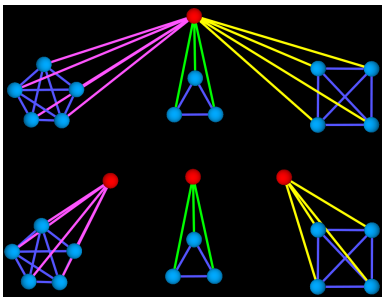


FIGURE 2.7: Duplication du sommet rouge : le sommet rouge appartient à trois cliques (image du haut), il est ensuite dupliqué en trois sommets, chacun d'entre eux étant relié aux sommets de l'une des cliques (image du bas).

La figure 2.6.b nous montre le graphe de la figure 2.6.a après la duplication des sommets et placé grâce au même algorithme. On peut voir qu'il est maintenant beaucoup plus lisible. Ceci est dû au fait qu'il n'est plus sans échelle (il reste cependant petit-monde). Nous allons donc pouvoir faire une partition de graphe.

Avant de passer à cette seconde étape du processus de fragmentation, il est nécessaire de préciser qu'il existe d'autres techniques permettant de supprimer la propriété sans échelle basées sur des systèmes de filtrage des sommets ou des arêtes (par exemple [172]). Nous avons cependant préféré introduire celle-ci car elle nous permet de garder toutes les informations contenues dans le réseau initial.

8. Une clique d'un graphe G est un **sous-graphe complet** de G, *i.e.* un graphe composé d'un sous-ensemble des sommets et des arêtes de G tel que chaque paire de sommet est une arête.

2.1.2.2 Identification des groupes et des ponts

La figure 2.8 nous montre un graphe composé de deux cliques ayant le sommet rouge en commun. Imaginons que nous voulions le fragmenter. Il est clair qu'il est composé de deux communautés (la clique de gauche et la clique de droite). Ce qui est moins clair, c'est de savoir à quelle communauté appartient le sommet rouge. On pourrait l'ajouter à la communauté ayant le plus grand nombre d'éléments (clique de gauche). Cependant, quelle est la valeur sémantique de ce choix ? En effet, si chaque clique représente les mots-clés d'un document, pourquoi le mot-clé associé au sommet rouge serait-il considéré comme étant plus pertinent dans l'un des documents seulement parce que ce document contient plus de mots-clés ? C'est pourquoi nous avons décidé d'identifier ces sommets et de les placer dans des communautés dont ils seront le seul élément. Ainsi, ils pourront apparaître dans la carte comme des **ponts** reliant des communautés.

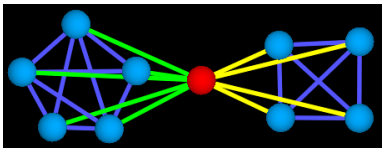


FIGURE 2.8: La création de deux communautés à partir de ce graphe fera du sommet rouge un sommet de la communauté de gauche. Ce choix est cependant contestable sémantiquement parlant. C'est pour cela que nous avons choisi de considérer ce type de sommets comme des ponts entre les communautés.

Nous allons maintenant décrire notre algorithme inspiré de [82] et adapté à la détection des ponts. Nous devons commencer par identifier ces derniers. Étant donné qu'ils se trouvent entre (*between*) les communautés, nous utilisons un indice appelé *betweenness centrality* introduit par Freeman [71]. Cet indice calcule le nombre pondéré de plus courts chemins passant par un sommet dans le graphe. Plus cet indice est grand, plus le sommet correspondant joue donc le rôle de pont dans le graphe. De façon plus formelle, voici la formule de cet indice :

$$BC(v) = \sum_{u \neq v \neq w \in V} \frac{\sigma_{uw}(v)}{\sigma_{uw}}$$

où $\sigma_{uw}(v)$ correspond au nombre de plus courts chemins entre deux sommets u et $w \in V$ passant par un sommet v et σ_{uw} correspond au nombre de plus courts chemins entre deux sommets u et $w \in V$. Pour calculer BC, nous utilisons l'algorithme proposé par Brandes [29].

L'algorithme permettant de trouver les communautés consiste à itérer une procédure qui calcule le *betweenness centrality* pour chaque sommet et supprime le sommet dont la valeur est la plus élevée. À la fin, on se retrouve avec un graphe **non connexe**, *i.e.* il n'existe pas de chemin pour tous les couples de sommets du graphe (voir figure 2.6.c). On peut maintenant considérer que chaque *composante connexe* (sous-graphe connexe) est une communauté.

Cette procédure pose cependant deux problèmes. D'abord, le temps de calcul de l'indice est long : $O(nm)$. Un travail est actuellement mené avec Guy Melançon afin de comparer les résultats du *betweenness centrality* avec ceux obtenus à l'aide d'autres

indices plus rapides à calculer (voir [32] pour une vue d'ensemble de ces indices). Une partie des résultats concernant des indices calculés sur les arêtes d'un graphe a d'ailleurs déjà été publiée [127]. Le second problème concerne le nombre de sommets à détecter en tant que pont. Nous préconisons ici de laisser le choix à l'utilisateur pour la simple et bonne raison qu'il doit déterminer ce nombre (et donc le nombre de communautés) en fonction du degré de précision qu'il désire [9]. Par exemple, s'il recherche le mot *jaguar*, peut-être voudra-t-il simplement créer une communauté pour les pages concernant les voitures, mais peut-être aussi voudra-t-il une fragmentation plus fine séparant aussi ces pages en communautés plus précises.

Grâce à la liste de communautés obtenue, nous pouvons maintenant créer le graphe quotient. Cependant, celui-ci sera légèrement différent d'un graphe quotient tel que nous l'avons défini en début de section car nous devons prendre en compte les ponts. Le nôtre se présente donc sous la forme $Q(G) = (B, C, E_{Q(G)})$ avec B l'ensemble des ponts, C l'ensemble des communautés et $E_{Q(G)}$ l'ensemble des arêtes (b, c) telles que $b \in B$, $c \in C$ et $\exists w \in c | (b, w) \in E$. Un graphe ainsi construit est appelé **bi-parti** : il contient deux ensembles de sommets tels que chaque arête est composée d'un sommet du premier ensemble et d'un sommet du second. La figure 2.6.d montre le graphe quotient obtenu.

Maintenant que nous avons structuré nos données, nous pouvons proposer une méthode de visualisation ainsi que des techniques de navigation.

2.1.3 Représentation et interaction

Dans la section précédente, nous avons proposé une méthode permettant d'identifier des communautés de mots-clés apparaissant souvent ensemble dans les pages *web* et des ponts entre ces communautés. Nous avons maintenant besoin de produire un système permettant de visualiser ces résultats afin d'orienter l'utilisateur vers les pages qu'il recherche. Ce système doit être muni d'une technique de navigation efficace non seulement pour donner la possibilité à l'utilisateur de passer aisément du graphe quotient aux mots-clés que des communautés choisies contiennent, mais aussi pour lui permettre d'accéder aux pages internet contenant ces mots-clés. La solution que nous proposons est décrite dans les sous-sections qui suivent. Nous étudierons d'abord les méthodes que nous utilisons afin de placer les objets sur la visualisation, puis nous décrirons les moyens d'interaction et de navigation.

2.1.3.1 Pré-traitements

Avant de commencer à placer concrètement les objets sur la carte, certaines opérations sont nécessaires.

9. Il n'est pas évident que l'utilisateur sache à l'avance le degré de précision nécessaire à ces recherches et il est encore moins évident qu'un éventuel degré de précision lui permette de déterminer le nombre de ponts à extraire. C'est pourquoi l'interface du système doit être réalisée de façon à ce que l'utilisateur puisse faire varier le nombre de ponts jusqu'à qu'il ait obtenu un résultat satisfaisant en fonction des taches qu'il doit accomplir.

Calcul des plus courts chemins Afin de refléter la topologie du graphe, nous souhaitons placer les sommets de façon à ce que la **distance euclidienne** entre chaque paire de sommets corresponde à la distance entre ces mêmes sommets dans le graphe, *i.e.* la longueur du plus court chemin les reliant. Cette **distance cible** est donc égale à la distance d_{ij} entre un sommet i et un sommet j telle que nous l'avons définie précédemment. Idéalement, les positions des sommets $p(1) = (x(1), y(1)), \ldots, p(n) = (x(n), y(n)) \in \mathbb{R}^2$ doivent être trouvées telles que les distances euclidiennes $\|p(i) - p(j)\| = \sqrt{(x(j) - x(j))^2 + (y(i) - y(j))^2}$ correspondent aux distances cibles.

Nous commençons donc notre méthode en construisant la matrice $D = (d_{ij})_{1 \leq i \leq n, 1 \leq j \leq n}$ contenant les distances cibles entre chaque paire d'éléments de $Q(G)$. Pour cela, nous utilisons un classique parcours en largeur (voir [24] par exemple pour la description d'un tel algorithme).

Suppression des ponts Avant le placement des communautés (ensemble C), nous supprimons temporairement les ponts (ensemble B). Le but de cette opération est de placer ces ponts après les communautés afin qu'ils apparaissent entre celles auxquelles ils sont reliés. Ceci rend la compréhension des relations entre les éléments plus simple pour l'utilisateur.

2.1.3.2 Positionnement des éléments du graphe

Dans la section précédente, nous avons construit une matrice D contenant des distances cibles. Idéalement, nous voudrions donc avoir :

$$\|p(i) - p(j)\| \approx d_{ij}$$

pour toutes les paires (i, j) avec $i, j \in C$. Ce problème peut être résolu grâce à un algorithme de *Multidimensional Scaling (MDS)* [34] (voir [141] pour obtenir plus d'informations sur l'adaptation de ces méthodes au dessin de graphe). Une approche efficace de *MDS* consiste à introduire une pénalité quadratique lorsque l'on s'éloigne de l'équation ci-dessus. Pour cela, on utilise une fonction de *stress* que l'algorithme de placement devra minimiser :

$$\sigma(p) = \sum_{i<j} w_{ij} \big(d_{ij} - \|p(i) - p(j)\|\big)^2$$

Celle-ci mesure l'éloignement entre la **configuration** courante $p = p(1), \ldots, p(n)$ et les distances cibles $\{d_{ij}\}$. Ici, on utilise la fonction standard pour calculer le poids $w_{ij} = d_{ij}^{-2}$ [107] qui permet de donner plus d'influence à la représentation des distances courtes et améliore ainsi le placement des structures locales.

Malheureusement, il n'existe pas de méthodes connues permettant la minimisation de la fonction *stress*. Il existe cependant des techniques permettant d'obtenir des résultats satisfaisants comme celle connue sous le nom de *stress majorization*.

Stress majorization Cette technique consiste à itérer une fonction calculant des configurations successives telles que la fonction *stress* n'augmente pas. Elle a été introduite par de Leeuw [52] et est devenue très populaire pour le dessin de graphe [107, 79, 141]. Plus concrètement, à partir d'une configuration $p^{[t]}$ à un temps t, elle permet de trouver une configuration $p^{[t+1]}$ en améliorant localement les positions des éléments. En voici la formule pour un sommet i :

$$p^{[t+1]}(i) \leftarrow \frac{\sum_{j:j\neq i} w_{ij} \left(p^{[t]}(j) + s_{ij} \cdot (p^{[t]}(i) - p^{[t]}(j))\right)}{\sum_{j:j\neq i} w_{ij}}$$

où

$$s_{ij} = \begin{cases} \frac{d_{ij}}{\|p^{[t]}(i) - p^{[t]}(j)\|} & \text{si } \|p^{[t]}(i) - p^{[t]}(j)\| > 0; \\ 0 & \text{sinon.} \end{cases}$$

En utilisant cette fonction sur tous les sommets, on obtient donc une nouvelle configuration. Cette opération est ensuite itérée jusqu'à ce qu'une configuration stable soit trouvée, *i.e.* jusqu'à ce que la fonction *stress* ne puisse plus être réduite significativement :

$$\frac{\sigma(p^{[t]}) - \sigma(p^{[t+1]})}{\sigma(p^{[t]})} < \epsilon$$

avec $\epsilon > 0$, *e.g.* $\epsilon = 10^{-4}$.

Les fonctions *stress* des configurations successives n'augmentent pas, *i.e.*

$$\sigma(p^{[0]}) \geq \sigma(p^{[1]}) \geq \sigma(p^{[2]}) \geq \cdots \geq \sigma(p^{[t]})$$

et elles convergent vers un minimum local [53]. L'avantage de l'utilisation d'une méthode de *stress majorization* réside dans sa faible complexité en temps : $O(n)$ (nous l'avons déjà expérimentée pour le placement des centres de *SequencesViewer* section 1.3.2.2). Cet algorithme est plus rapide qu'un algorithme de force tel que [72] qui est en $O(n^3)$ (voir [28] pour une vue d'ensemble de ce type de méthodes basées sur l'analogie avec des forces physiques et [91] pour une comparaison de leurs résultats).

En revanche, un problème de cette méthode est dû au fait que le résultat final dépend fortement de la configuration initiale choisie ($p^{[0]}$). Nous allons maintenant décrire la solution à ce problème.

Placement initial Comme le montrent Pich et Brandes [33], la méthode présentée ci-dessus a la fâcheuse tendance de faire tomber la fonction *stress* dans des minima locaux. Pour éviter cela, il faut trouver une configuration initiale globale respectant la topologie du graphe et utiliser ensuite le processus de *stress majorization* pour affiner localement le positionnement.

Une méthode efficace de placement global initial est connue sous le nom de *Classical MDS* [176]. Basée sur de l'algèbre linéaire, elle permet elle aussi de trouver une configuration dont la la valeur de la fonction *stress* est peu élevée. Elle utilise une matrice $B = (b_{ij})$ définie ainsi :

$$b_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{is}^2 - \frac{1}{n}\sum_{r=1}^{n} d_{rj}^2 + \frac{1}{n^2}\sum_{r,s=1}^{n} d_{rs}^2\right).$$

Soient $\lambda_1, \lambda_2 \in \mathbb{R}$ les deux plus grandes valeurs propres de B et $u_1, u_2 \in \mathbb{R}^n$ les vecteurs propres correspondants (ce sont des réels car la matrice est symétrique). Ces valeurs peuvent être calculées, par exemple, grâce à un algorithme nommé *power iteration* (voir [83]). Deux vecteurs de coordonnées x et y peuvent être ensuite déduits :

$$x = \sqrt{\lambda_1}u_1, \quad y = \sqrt{\lambda_2}u_2$$

La configuration initiale correspond aux valeurs de ces vecteurs :

$$p(i) = (x(i), y(i))$$

2.1.3.3 Post-traitements

Après avoir trouvé un positionnement des communautés grâce aux méthodes de *Multidimensional scaling* il faut maintenant réintroduire les ponts.

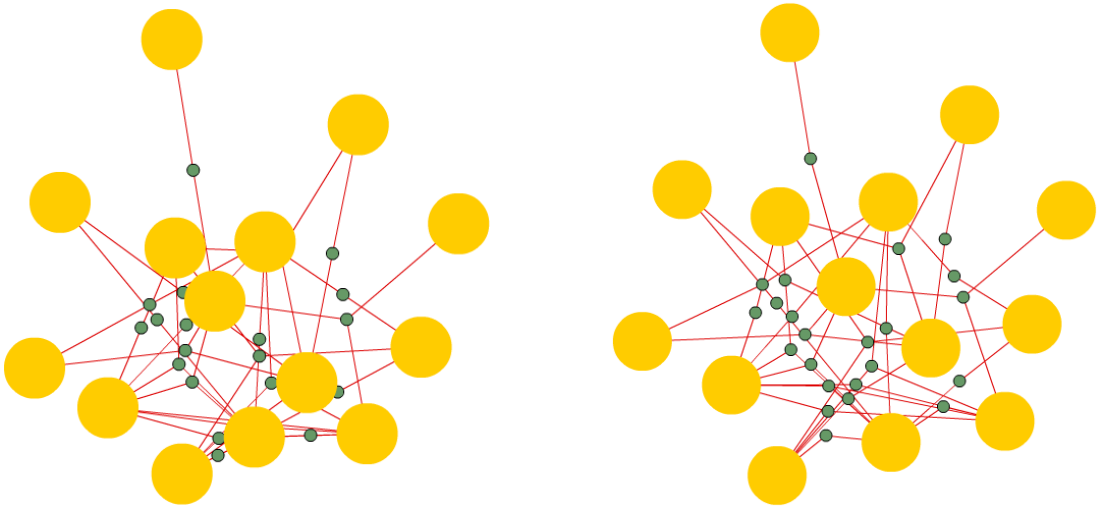
Ré-insertion des ponts Nous considérons ici que la manière la plus intuitive et la plus efficace visuellement consiste à placer les ponts (ensemble B) au niveau du barycentre des communautés de C auxquelles ils sont reliés dans le graphe biparti $Q(G) = (B, C, E_{Q(G)})$. Soit un pont $b \in B$ relié à un ensemble de communautés $N(b) = \{c \in C \colon (b, c) \in E_{Q(G)}\}$, la position de b est donné par la formule :

$$p(b) = \frac{1}{|N(b)|}\sum_{c \in N(b)} p(c)$$

La figure 2.9.a montre le résultat d'un tel positionnement. Nous aurions pu aussi placer les ponts en même temps que les communautés en utilisant les algorithmes de *Multidimensional scaling*. Après avoir testé cette approche, nous avons préféré utiliser la méthode des barycentres afin que les ponts soient positionnés entre les communautés qu'ils relient. Cette solution permet en effet d'introduire visuellement la notion de *betweenness* qui a été utilisée pour trouver les ponts [10].

Comme nous pouvons le constater sur la figure 2.9.a, les sommets ainsi placés se chevauchent beaucoup ce qui rend la carte illisible. C'est pourquoi nous allons maintenant voir quelle technique peut être mise en place afin de supprimer ces chevauchements.

10. Dans ce cas, la syntaxe est basée sur des propriétés cognitives qui nous feront attribuer un sens au signe /pont placé entre les communautés auxquelles il est relié/. Dans le première partie, nous avons mis le cartographe en garde contre l'utilisation de tels procédés. Dans les faits, nous pensons qu'une telle approche peut être utilisée si une légende ou une explication quelconque vient confirmer l'intuition cognitive.



(a) (b)

FIGURE 2.9: Graphe quotient biparti $Q(G) = (B, C, E_{Q(G)})$: (a) après ré-insertion des ponts, (b) après suppression des chevauchements.

Suppression des chevauchements La méthode utilisée afin de supprimer les chevauchements des sommets est présentée dans [76]. C'est une adaptation originale du processus de *strees majorization* décrit ci-dessus au problème qui nous concerne ici. Selon cette approche, les chevauchements sont éliminés graduellement en déplaçant petit à petit les sommets qui se chevauchent de façon à modifier le moins possible la configuration issue de l'étape de placement.

L'algorithme repose sur la création d'un **graphe de voisinage** dérivé de la configuration courante des éléments. Plus concrètement, nous calculons une **triangulation de Delaunay** sur les éléments de $Q(G)$ ce qui nous permet d'obtenir un graphe planaire $DT(Q) = (V_{DT(Q)} = B \cup C, E_{DT(Q)})$. Dans ce graphe, la paire de sommets (i, j) appartient à $E_{DT(Q)})$ si et seulement si les régions du **diagramme de Voronoï**[11] correspondantes sont adjacentes. Il est important de noter ici que $DT(Q)$ est seulement construit à partir du positionnement sur le plan des sommets de $Q(G)$, il ne tient donc pas compte des arêtes de E ou de $E_{Q(G)}$.

La suppression de chevauchements est effectuée en itérant le processus suivant :

1. Tout d'abord, le graphe $DT(Q)$ est construit grâce à une triangulation de Delaunay des sommets de $Q(G)$ préalablement positionnés.
2. Ensuite, pour chaque arête de la triangulation $(i, j) \in E_{DT(Q)}$ un **facteur de chevauchement**
$$t_{ij} = \max\left(\frac{a_i + a_j}{\|p(i) - p(j)\|}, 1\right)$$
est calculé, où a_i et a_j représentent les rayons des sommets i et j. Il correspond à la valeur minimale par laquelle la distance euclidienne doit être multipliée afin de

11. Par définition, la région du diagramme de Voronoï correspondant à un sommet i est la région formée par les points du plan plus proches de i que de n'importe quel autre sommet.

supprimer le chevauchement (il est donc égal à 1 si les sommets ne se chevauchent pas). La nouvelle distance cible doit être

$$d_{ij}^{\mathrm{DT}} = s_{ij}^{\mathrm{DT}} \|p(i)^0 - p(j)^0\|$$

où, s_{ij}^{DT} est un facteur d'amortissement défini par $s_{ij}^{\mathrm{DT}} = \min\{s_{max}, t_{ij}\}$. s_{max} est une constante d'amortissement, elle doit être supérieure à 1 et représente la quantité maximale de chevauchement devant être éliminée en une itération. Elle permet de ne pas trop s'éloigner de la configuration initiale.

3. Enfin, grâce aux nouvelles distances cibles, on obtient une nouvelle fonction *stress*

$$\sigma^{\mathrm{DT}}(p) = \sum_{(i,j)\in E_{DT(Q)}} w_{ij} \left(d_{ij}^{\mathrm{DT}} - \|p(i) - p(j)\|\right)^2$$

que nous allons pouvoir minimiser en utilisant la méthode de *stress majorization* définie dans la sous-section 2.1.3.2. On substitue pour cela d_{ij}^{DT} et s_{ij}^{DT} à d_{ij} et s_{ij}.

Ces trois opérations sont répétées jusqu'à ce que les chevauchements soient supprimés, *i.e.* jusqu'à ce que $t_{ij} = 1$ pour tout $(i, j) \in E_{DT(Q)}$. Le figure 2.9.b représente le même graphe que celui de la figure 2.9.a après avoir utilisé cet algorithme de suppression des chevauchements.

Interaction et navigation La visualisation créée lors des étapes précédentes (figure 2.9.b) permet à l'utilisateur d'avoir une vue d'ensemble des mots-clés des pages internet correspondant à sa recherche. Afin qu'il puisse comprendre la signification des communautés et explorer les pages correspondantes, nous avons mis en place plusieurs types d'interaction.

Tout d'abord, lorsque l'utilisateur survole une communauté, une infobulle contenant la liste des mots-clés appartenant à cette communauté est affichée, comme on peut le voir sur la figure 2.11.a. Ceci lui permet de sélectionner rapidement les communautés qui l'intéressent. De plus, les ponts sont étiquetés par leur mot-clé ce qui est nécessaire à la compréhension des relations entre les communautés. Lorsque l'on clique sur une communauté, le cercle jaune la représentant disparaît et les sommets qu'elle contient viennent se placer sur un cercle dont le rayon dépend de leur nombre [12] (voir figure 2.10 dans laquelle deux communautés sont ainsi « ouvertes »). L'utilisateur peut ainsi avoir une information plus précise sur cette communauté et les liens qui existent entre les mots-clés qui la composent. Il peut aussi la refermer en cliquant au centre du cercle. Enfin, un clic droit sur une communauté permet d'afficher une liste de liens hypertextes menant aux pages internet contenant les mots-clés de cette communauté (un exemple est montré sur la figure 2.10). L'utilisateur peut donc naviguer dans les pages retournées par un moteur de recherche grâce à notre système.

Mises à part les différentes interactions disponibles sur les communautés, nous en avons aussi développé d'autres sur les sommets représentant des mots-clés. Lorsque la souris

12. Compte tenu de la densité des arêtes présentes à l'intérieur des communautés, une alternative intéressante mais que nous n'avons pas testée consisterait à utiliser des matrices pour les représenter, comme Henry *et al.* le proposent dans l'article [93].

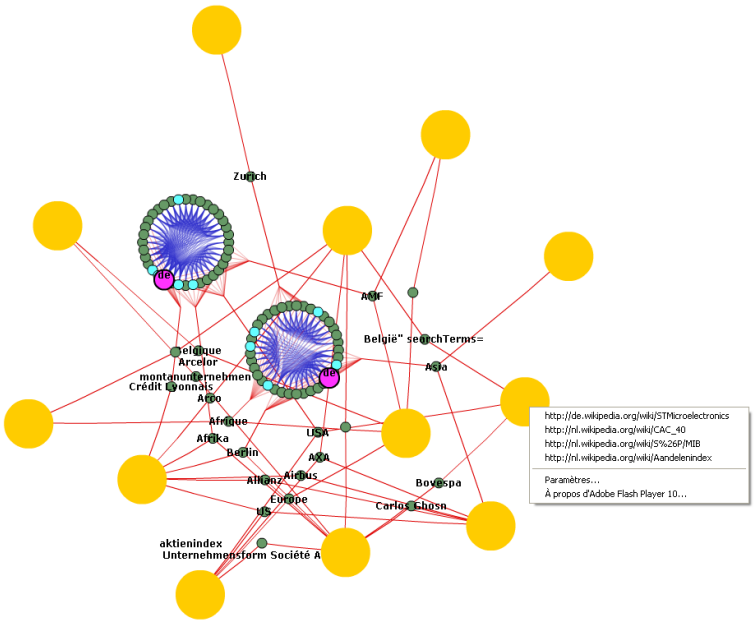


FIGURE 2.10: Même graphe que celui représenté sur la figure 2.9 avec deux communautés « ouvertes » et la liste des pages internet affichée pour l'une des autres communautés.

survole l'un de ces sommets, une infobulle indique quelle est le mot-clé correspondant. Les sommets dupliqués apparaissent en bleu alors que les autres sont dessinés en vert. Lorsque l'on clique sur un des sommets dupliqués (voir section 2.1.2.1), celui-ci, ainsi que toutes les autres instances du même mot-clé, sont affichés en rose, étiquetés avec ce mot-clé et leurs tailles sont augmentées (voir figure 2.10). L'utilisateur peut ainsi identifier rapidement les communautés dans lesquelles ils apparaissent.

L'expansion d'une communauté afin d'afficher les sommets qu'elle contient peut entraîner de nouveaux problèmes de chevauchement. En effet, le diamètre du cercle le long duquel ces sommets viennent se placer est déterminé en fonction de leur nombre. Si ce nombre est supérieur à un certain seuil, le cercle est plus grand que le disque jaune qui représentait précédemment la communauté. Nous sommes donc obligés de dérouler de nouveau l'étape de suppression des chevauchements décrite ci-dessus en modifiant les rayons a_i des sommets « ouverts » de C. L'ordre des sommets positionnés le long du cercle a aussi son importance. En effet, comme ceux-ci ont des arêtes partant vers les autres communautés, il est important que ceux reliés aux même communautés soient placés consécutivement. Ceci permet de limiter les croisements d'arêtes et de les regrouper plus facilement. Pour cela, nous utilisons l'heuristique du barycentre qui consiste à ordonner les sommets en fonction de l'angle des coordonnées polaires de leur barycentre [10]. Lorsque toutes les communautés sont « ouvertes », cela revient à calculer un positionnement connu sous le nom de *micro/macro graph layout* [16].

2.1.4 Études de cas

Cette section est dédiée à l'étude d'exemples illustrant les bénéfices que notre approche apporte à la navigation internet. Les ensembles de données sont composés de pages de *wikipedia* et des réseaux de mots-clés correspondants. Ces pages sont sélectionnées grâce à une requête lancée sur le moteur de recherche *Exalead*[13].

2.1.4.1 *Jaguar*

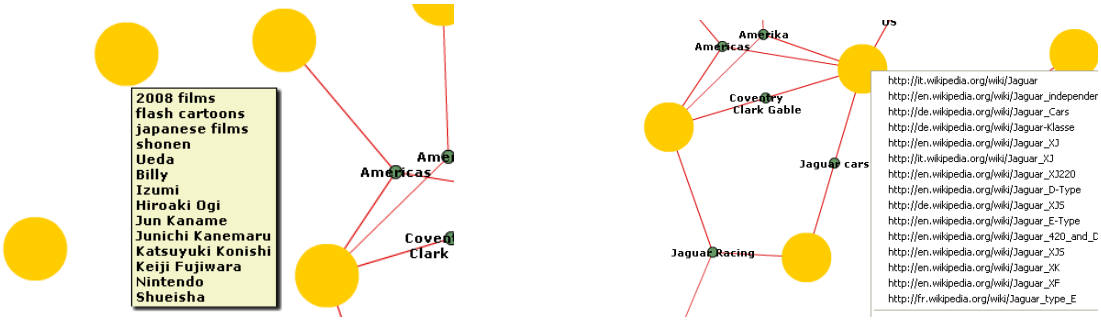
Ce premier exemple montre les résultats obtenus en recherchant le mot *jaguar*. Les 50 premières pages retournées nous ont fourni un graphe contenant 462 mots-clés et 4458 arêtes. Le coefficient de *clustering* moyen est de 0,9 et le diamètre de 2,42. On est donc bien en présence d'un graphe petit-monde.

Comme nous l'avons évoqué plus haut, ce mot est très ambigüe. En effet, il appartient à plusieurs champs sémantiques et un moteur de recherche classique ne fait pas de distinctions entre les pages de ces différents champs. C'est ainsi que les pages concernant la marque de voitures seront mélangés avec les pages concernant les animaux, *etc.*. Ceci est aussi vrai pour le réseau de co-occurrence, *e.g.* le mot *jaguar*, qui est évidemment un mot-clé de toutes les pages retournées, est du coup connecté à tous les autres mots-clés. L'étape de duplication permet de repérer des sommets comme *jaguar* (*i.e.* les sommets ayant les degrés les plus élevés) et de les dupliquer. Ainsi, les mots-clés concernant les animaux et ceux concernant les voitures sont « moins connectés » et les communautés correspondantes peuvent être extraites. Ceci justifie notre approche de dupliquer les sommets dont les voisinages sont les plus importants : ce sont des termes très génériques qui apparaissent dans la plupart des pages et ne sont donc pas utiles à la formation des communautés.

Grâce à ce procédé, notre algorithme de fragmentation peut identifier les communautés (voitures, animaux, jeux vidéos, *etc.*). La visualisation permet ensuite de les séparer en entités visuelles distinctes comme le montre la figure 2.11.a. Ainsi, les différents thèmes en liaison avec le mot recherché sont organisés en groupes. Grâce aux infobulles contenant tous les mots-clés de ces groupes, l'utilisateur peut identifier rapidement quels sont les principaux thèmes abordés par les pages internet correspondantes. Par exemple, la figure 2.11.a nous montre l'infobulle de l'un des groupes. Celui-ci ne contient que des mots-clés en rapport avec le manga japonais *Pyu to Fuku ! Jaguar*.

Lorsque l'utilisateur effectue un clic droit sur l'un des groupes, il accède à la liste des liens hypertextes vers les pages internet ayant pour mot-clé au moins l'un des éléments du groupe. Comme le montre la figure 2.11.b, ceci permet non seulement de naviguer sur internet mais aussi d'aider à l'identification des thèmes des communautés sans passer par la liste des mots-clés.

13. http ://www.exalead.com/search/wikipedia/



(a) (b)

FIGURE 2.11: Recherche du mot clé *jaguar* : (a) l'infobulle nous montre que la communauté est composée de mots-clés en relation avec le manga *Pyu to Fuku ! Jaguar*, (b) des liens hypertextes vers des pages internet apparaissent lorsque l'utilisateur effectue un clic droit, il est clair ici que la communauté ne contient que des pages sur les voitures.

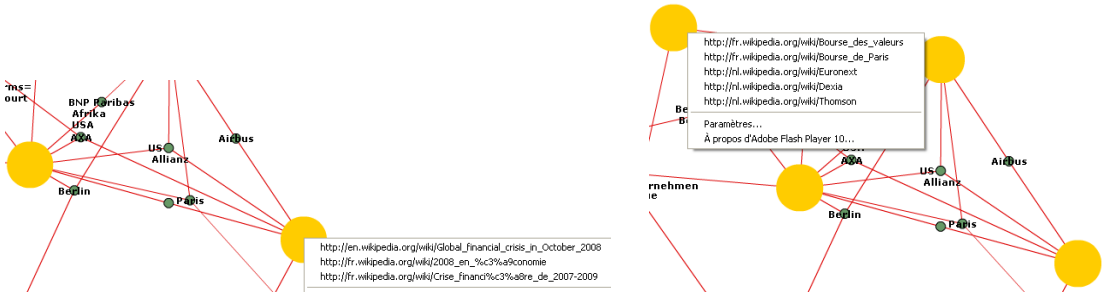
2.1.4.2 *CAC40*

L'algorithme de fragmentation présenté ci-dessus permet d'identifier certains mots-clés comme des ponts. En effet, même si la plupart des mots-clés appartient à un champ sémantique particulier et peut du coup être organisé en communautés distinctes, d'autres mots-clés peuvent appartenir à plusieurs [14] champs sémantiques et ainsi constituer des ponts entre les communautés correspondantes.

Ce second exemple illustre l'importance qu'ont ces ponts pour l'exploration des résultats retournés par un moteur de recherche. Le mot recherché ici est *CAC40*, indice boursier associé aux 40 premières entreprises françaises. Nous avons considéré les 50 premières pages retournées par *Exalead*, ce qui nous donne un graphe de 556 mots-clés et 4852 arêtes. Le coefficient de *clustering* est de 0,91 et le diamètre de 2,56. Il est donc aussi petit-monde.

La liste des hyperliens affichée dans la figure 2.12.a nous indique que les pages en relation avec la communauté concernent la *crise financière*. Cette communauté est reliée avec une communauté de mots-clés représentant des concepts généraux du *CAC40* grâce aux ponts *Paris*, *AXA* et *US Allianz*. La *crise financière* concerne donc, entre autre, *Paris*, place boursière caractérisée par le *CAC40*. Ceci suggère que la crise affecte cet indice boursier. *AXA* est une compagnie d'assurance française et d'investissement financier. La figure 2.12.b montre qu'elle relie non seulement les communautés *CAC40* et *crise financière* mais aussi la communauté *Euronext/Paris*, qui est une partie d'un groupe mondial de places boursières. On peut donc penser que la société *AXA* est implantée dans plusieurs pays, qu'elle est affectée par la *crise financière* et qu'elle fait partie des entreprises côtées au *CAC40*.

14. « Plusieurs » doit être ici compris comme « quelques ». En effet, un mot-clé appartenant à tous les champs sémantiques serait dupliqué et ne pourrait donc pas être considéré comme un pont.



(a) (b)

FIGURE 2.12: Recherche du mot clé *CAC40* : (a) Le pont *Paris* relie la communauté *CAC40* et la communauté *crise financière*, (b) *AXA* est un pont entre les communautés *CAC40*, *Euronext/Paris* et *crise financière*.

2.1.4.3 *Hepburn*

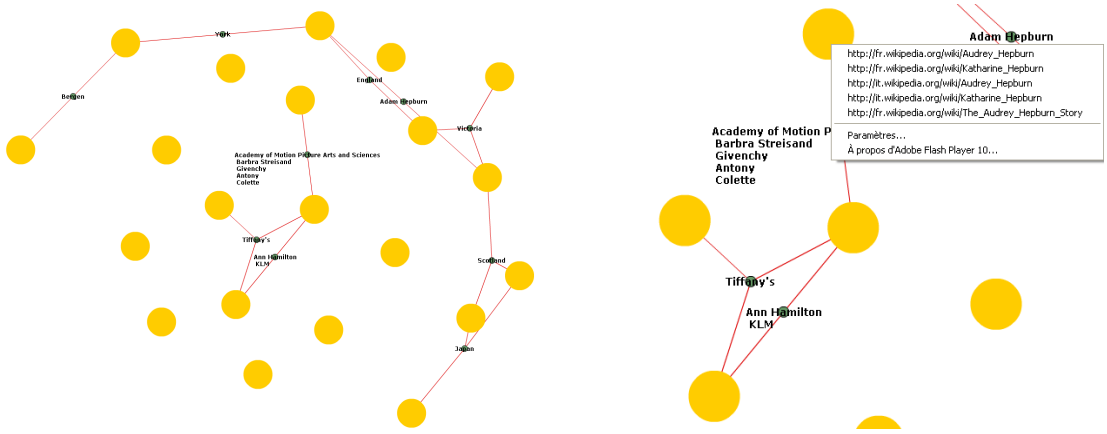
Pour finir, le dernier exemple a été produit en recherchant le mot *Hepburn* qui correspond à une famille écossaise connue. Ce nom est aussi très répandu dans d'autres régions d'Europe. Le graphe obtenu en gardant les 50 premières pages retournées par le moteur de recherche est composé de 554 mots-clés et de 4636 arêtes avec un coefficient de *clustering* de 0,92 et un diamètre de 2,51.

La figure 2.13.a nous montre l'ensemble du réseau produit grâce à notre technique. On observe ici l'efficacité des méthodes de duplication des sommets et d'identification des ponts qui permettent de ne pas déconnecter les communautés sémantiquement proches. Intéressons-nous maintenant aux quatre communautés connexes du centre de la visualisation (voir 2.13.b). Elles contiennent des pages concernant les deux actrices Audrey et Katherine Hepburn. Elles sont complètement déconnectées des autres communautés car elles sont les seules à avoir le *cinéma* comme thème commun. En effet, ces autres communautés contiennent des pages concernant des politiciens ou des écrivains et il est donc bénéfique que notre algorithme les ait déconnectées.

2.1.5 Conclusion

La première application de visualisation de graphes que nous venons de présenter permet donc, d'une part, de regrouper des pages retournées par un moteur de recherche en identifiant des mots-clés « ponts » et, d'autre part, de visualiser et d'explorer ces résultats. Notre système permet de faciliter l'accès aux informations contenues sur internet en accélérant l'identification des pages ayant un intérêt en fonction du thème recherché.

Nous allons maintenant étudier un second exemple d'utilisation de graphes en visualisation de l'information. Plus particulièrement, nous allons nous intéresser aux arbres, catégorie particulière de graphes permettant de modéliser des hiérarchies.



(a) (b)

FIGURE 2.13: Recherche du mot clé *Hepburn* : (a) vue d'ensemble du graphe, (b) composante connexe contenant les pages en liaison avec le *cinéma*.

2.2 *Geographical treemaps*

Cette section est consacrée à la présentation d'une technique de visualisation de hiérarchies sous la forme de « cartes géographiques ». Avant de rentrer dans la description de l'algorithme, nous allons d'abord étudier les raisons qui nous ont poussé à proposer une telle solution. Commençons par présenter la notion d'arbre, graphe particulier permettant de modéliser certaines hiérarchies.

Un **arbre** $T(V, A)$ est un graphe orienté avec V l'ensemble de ses sommets et A l'ensemble de ses arcs *i.e.* un ensemble de paires ordonnées de V. De plus, il n'existe qu'un seul chemin P (indépendamment de l'orientation des arcs) qui aille de n'importe quel sommet u à n'importe quel sommet v sans passer deux fois par le même sommet : $\exists! P = (u, w_1, w_2, ..., w_k, v)$ avec (u, w_1) ou $(w_1, u) \in A$, (w_k, v) ou $(v, w_k) \in A$, $\forall i \in [1, k-1], (w_i, w_{i+1})$ ou $(w_{i+1}, w_i) \in A$ et $\forall a, b \in P, a \neq b$. Enfin, un arbre est **connexe** : il existe un chemin entre chaque paire de sommets [15]. Le sommet r de l'arbre qui n'a que des arcs sortants (r tel que $\nexists v \in V$ avec $(v, r) \in A$) est appelé **racine** de l'arbre. A l'opposé, les sommets l_i qui n'ont pas d'arcs sortants (les l_i tels que $\nexists v \in V$ avec $(l_i, v) \in A$) sont les **feuilles** de l'arbre.

Ce type de graphe est particulièrement utile. Il permet en effet de modéliser un grand nombre de données réelles, en particulier les données hiérarchisées. Prenons un exemple. Le philosophe Porphyre de Tyr essaie d'expliquer à l'un de ses élèves sa classification des objets du monde. De façon textuelle, cela donnerait : « La *substance* peut être *corporelle* ou *non corporelle*. La substance corporelle est elle-même constituée d'êtres *vivants* et d'êtres *non vivants*. Enfin, les êtres vivants peuvent être doués de *raison* ou non. » Ce texte est court et convient pour une hiérarchie peu fournie. Cependant, Porphyre n'en est pas satisfait, il pense que l'on pourrait représenter ces données de façon différente

15. Nous pourrions aussi définir un arbre comme un graphe non orienté, acyclique et connexe (voir [24]) Nous préférons cependant le voir comme un graphe orienté car cela permet de comprendre de façon plus intuitive la modélisation de hiérarchies en arbres.

afin d'accélérer leur compréhension et d'aider son élève à s'en rappeler. Il décide donc de « traduire » ce modèle hiérarchique sous forme d'un arbre $T(V, A)$ avec :

- $V = \{substance, corporelle, non\ corporelle, vivante, non\ vivante, ...\}$
- $A = \{(substance, corporelle), (substance, non\ corporelle), (corporelle, vivante), ...)\}$

Enfin, il peut dessiner cet arbre pour donner à son élève une référence visuelle (voir figure 2.14). Ainsi, les données sont facilement compréhensibles et il sera plus facile à son élève de garder une image mentale de leur contenu.

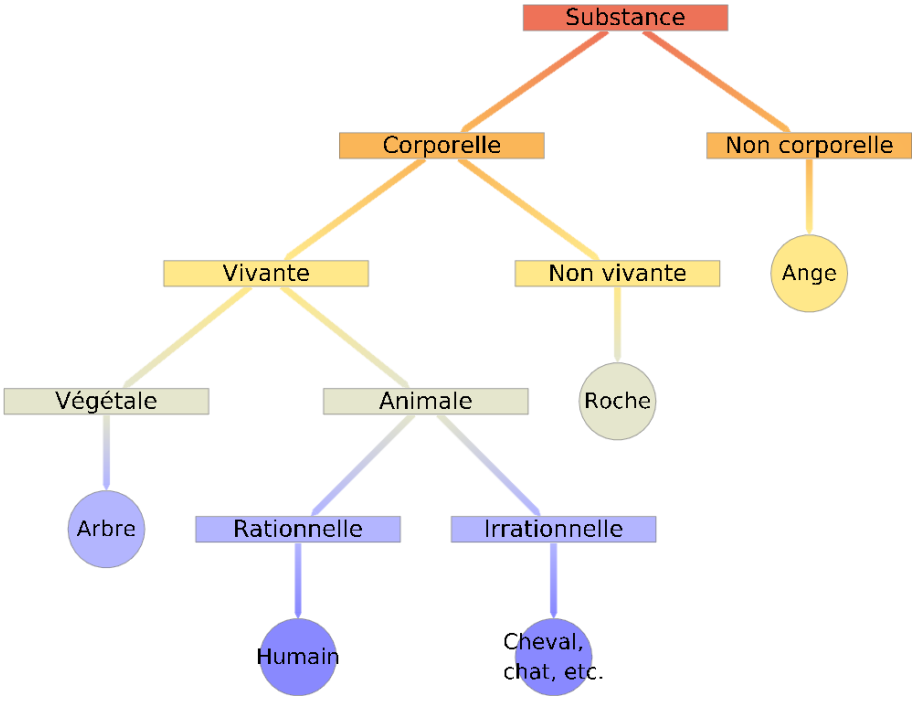
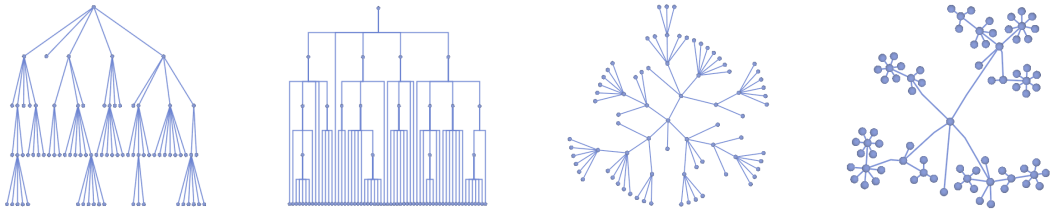


FIGURE 2.14: Arbre de Porphyre

De la classification des sciences d'Aristote à la hiérarchie des dossiers d'un ordinateur, de la taxonomie des espèces de Linné à la classification décimale des livres proposée par Dewey, on voit aisément combien la hiérarchisation des connaissances nous est familière. De nos jours, la forte croissance des données disponibles entraîne la création de taxonomies toujours plus importantes et il devient de plus en plus nécessaire d'avoir recours à des systèmes de visualisation afin de pouvoir les manipuler.

De nombreuses méthodes ont déjà été proposées. Tout d'abord, les chercheurs ont basé leurs représentations sur le dessin de nœuds reliés par des liens (type de représentation que nous avons utilisé pour les graphes de la section précédente et pour l'arbre de Porphyre figure 2.14). Reingold et Tilford ont par exemple proposé une technique permettant d'afficher un sommet u au-dessus de ses **enfants** [146], *i.e.* l'ensemble des sommets $\{v|(u, v) \in A\}$ (voir figure 2.15.a). Aussi basés sur cette métaphore, les dendrogrammes ont connu un grand succès (figure 2.15.b). Plus tard, Eades a introduit le dessin radial consistant à afficher les sommets sur des cercles concentriques en fonction de leur place dans la hiérarchie [58] (figure 2.15.c). La méthode *bubble tree drawing* place quant à elle les sommets en cercle autour de leurs parents [88] (figure 2.15.d). D'autres méthodes ont aussi été proposées sur lesquelles nous ne reviendrons pas. Le lecteur intéressé pourra se

référer à [15, 110]. Le problème de ces méthodes est qu'il est difficile de modifier la taille des sommets sans perdre leur lisibilité et qu'elles n'utilisent pas efficacement l'espace de visualisation.



(a) (b) (c) (d)

FIGURE 2.15: Différentes représentations d'arbres : (a) Tilford et Reingold [146], (b) Dendrogramme, (c) Eades [58], (d) Grivet *et al.* [88].

Une autre métaphore visuelle consiste à représenter les sommets d'un arbre sous forme de régions. Celles-ci peuvent être adjacentes, comme c'est le cas pour les *icicle plots* [119], ou imbriquées (*i.e.* les régions dénotant les enfants d'un sommet sont incluses à l'intérieur de la région représentant ce sommet), comme les *treemaps* [104] dont nous nous sommes déjà servis pour représenter la hiérarchie des séquences de gènes (voir section 1.1.2.3). L'article [85] donne une vue d'ensemble de ces techniques. Comme les méthodes nœuds-liens, les *icicle plots* ont tendance à entraîner la perte de beaucoup d'espace. En ce qui concerne les *treemap*, il a été démontré qu'ils sont moins intuitifs que les représentations nœuds-liens, *i.e.* la structure de l'arbre n'est pas aussi facile à discerner pour un novice [13]. Nous allons maintenant proposer une nouvelle métaphore qui ne présente pas les inconvénients de ces méthodes tout en préservant leurs avantages.

La représentation d'arbres sous forme de visualisations ressemblant a des cartes géographiques a été beaucoup utilisée, ce qui semble indiquer que c'est une solution naturelle et intuitive à notre problème. On peut trouver de nombreux exemples de l'emploi de cette métaphore, comme la célèbre *map on temperance* de Murrell [132] en 1846 (voir figure 2.16). De nos jours, on peut citer en exemple la carte du *European Economic and Social Committee*[16] ou sur un ton plus léger, la carte des communautés du *web*[17].

Nous allons donc maintenant présenter une visualisation basée sur la métaphore géographique [166]. Ce travail est motivé par une hypothèse selon laquelle les gens qui ont l'habitude de manipuler des cartes géographiques ont développé des capacités cognitives spécifiques qui pourraient être utilisées en visualisation de l'information (idée suggérée par Fabrikant et Skupin [68]). Ainsi, nous espérons qu'une telle représentation pourrait améliorer la reconnaissance des régions selon leur forme et leur taille (*e.g.* lorsque nous parlons du pays en forme de botte, n'importe qui comprendra que nous faisons référence à l'Italie). De plus, les cartes géographiques contiennent aussi souvent une hiérarchie que les gens ont l'habitude d'utiliser, par exemple, une carte de France est divisée en régions, elles-mêmes divisées en départements, eux-mêmes divisés en communes.

En plus d'avoir l'aspect d'une carte géographique, nous voulons aussi que notre représentation reflète certaines propriétés de l'arbre. Tout d'abord, les régions doivent avoir une

16. http ://www.eesc.europa.eu/ ?i=portal.en.self-and-co-regulation-cartography
17. http ://xkcd.com/802_large/

FIGURE 2.16: *Map on temperance* : carte dessinée par Murell en 1846 [132]

aire proportionnelle à celle de leurs enfants afin de pouvoir comparer leurs tailles. Ensuite, comme la plupart des ensembles de données évoluent dans le temps, nous souhaitons que la visualisation soit stable afin de préserver la carte mentale de l'utilisateur (voir Liu et Stasko [124] pour une discussion sur la notion de modèle mental en visualisation de l'information).

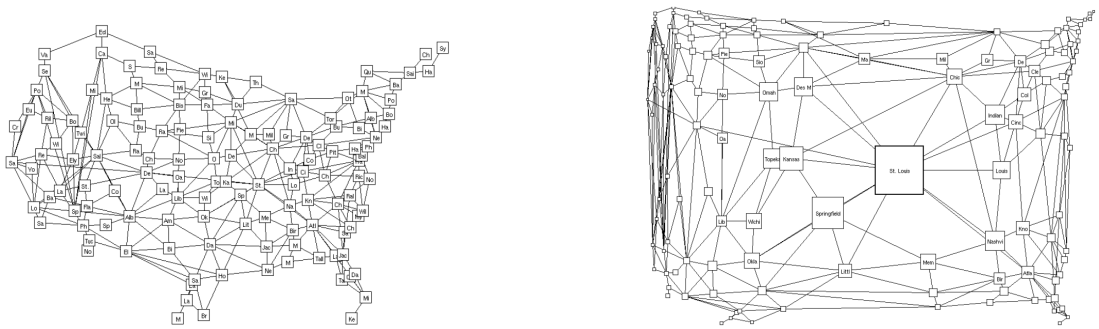
La visualisation proposée dans les sections suivantes suit les contraintes que nous venons de décrire. Elle a été réalisée avec David Auber, Charles Huet et Antoine Lambert et un article est en cours de production [7]. Tout d'abord, nous passerons en revue les méthodes liées à notre problème (section 2.2.1). Ensuite, nous donnerons une vue d'ensemble de notre technique (section 2.2.2). La section 2.2.3 sera dédiée à l'approfondissement de certains points. Puis nous décrirons les méthodes d'interaction mises en place (section 2.2.5). Enfin, nous discuterons des résultats obtenus par notre visualisation (section 2.2.5).

2.2.1 Travaux précédents

Nous allons commencer par décrire les techniques de visualisation basées sur la métaphore géographique afin de montrer qu'elles ne satisfont pas les contraintes mentionnées ci-dessus. Puis nous reviendrons sur les techniques représentant les sommets des arbres sous forme de régions afin d'étudier leurs limites vis-à-vis des propriétés vues dans l'introduction.

2.2.1.1 Méthodes basées sur la métaphore géographique

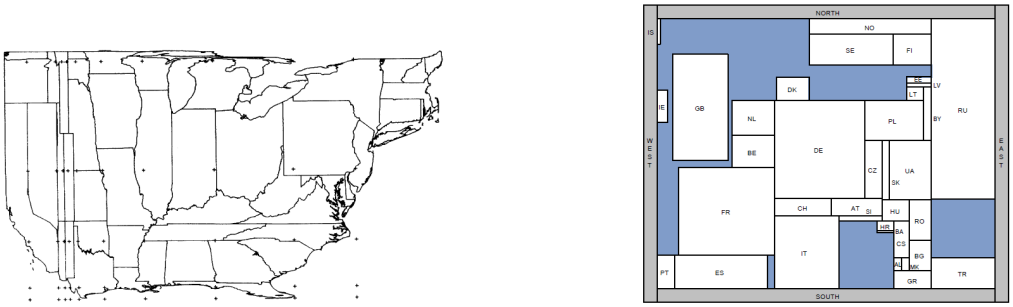
Cartes géographiques en entrée La modification de cartes géographiques en fonction de certains attributs a fait l'objet de nombreux articles. Par exemple, Sarkar et Brown ont introduit une méthode de distorsion permettant de mettre en évidence un point d'intérêt sans perdre le contexte (voir figure 2.17), *i.e.* en gardant tous les éléments de la carte [158] (ce type de méthode est connu sous le nom de *focus+context* et a été introduit par Furnas [74]). Les *cartogrammes* sont des cartes dont les formes et les tailles des régions sont modifiées afin de mettre en évidence certaines de leurs propriétés (voir figure 2.18) . L'adjacence des régions de la carte initiale peut être préservée [175, 112] ou les régions peuvent être représentées par des rectangles [145, 184], ce qui ne préserve pas l'adjacence mais facilite la comparaison des tailles. Toutes ces méthodes sont basées sur l'utilisation d'attributs spatiaux ce qui n'est pas le cas de notre arbre. Elles ne peuvent donc pas nous aider à résoudre notre problème.



(a) (b)

FIGURE 2.17: *Fisheye* de Sarkar et Brown [158] : (a) graphe initial de villes des États-Unis, (b) même graphe où Saint-Louis a été sélectionné comme point d'intérêt.

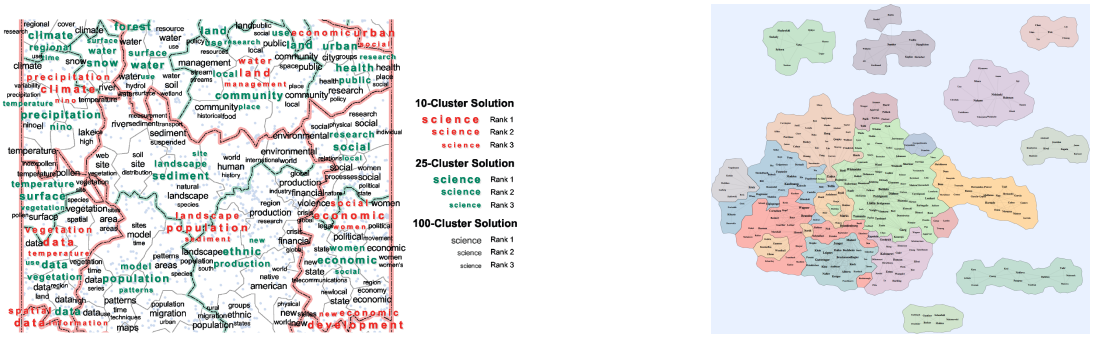
Des *Self-organizing maps* aux diagrammes de Voronoï Les *self-organizing maps* (*SOM*) [115] permettent de produire des cartes bidimensionnelles où les objets similaires sont proches les uns des autres. Elles sont basées sur un algorithme d'apprentissage non supervisé. Skupin [167] propose une méthode basée sur les *SOMs*, les diagrammes de Voronoï, et une méthode de fragmentation afin de produire des images ressemblant à des cartes géographiques. Les entités affichées correspondent à des textes et leur proximité dénote le nombre de concepts qu'ils ont en commun (voir figure 2.19.a). Nous ne connaissons pas de méthode permettant d'adapter cette technique à un arbre. Gansner *et*



(a) (b)

FIGURE 2.18: *Cartogrammes* : (a) l'adjacence des pays est préservée [175], (b) les régions sont représentées par des rectangles [184].

al. [78, 77, 97] représentent des graphes selon la métaphore géographique. Pour cela, ils utilisent un algorithme de dessin de graphe, puis un algorithme de fragmentation et enfin un diagramme de Voronoï (voir figure 2.19.b). Le problème de cette méthode est qu'elle n'assure pas la connexité des régions.



(a) (b)

FIGURE 2.19: Des *Self-organizing maps* aux diagrammes de Voronoï : (a) méthode proposée par Skupin [167], (b) méthode proposée par Gansner *et al.* [97].

Représentations sous forme de cartes topologiques Une autre approche pour produire des visualisations ressemblant à des cartes géographiques consiste à représenter les données sous forme de cartes topologiques. Les *Themescapes* [200] sont des cartes topologiques abstraites tridimensionnelles d'information (voir figure 2.20). Elles ont inspiré une visualisation de graphe connue sous le nom *GraphSplatting* [185] (voir figure 2.21). Celle-ci consiste à transformer un graphe dont les sommets sont positionnés dans un champ scalaire, rendu grâce à une coloration adaptée, une troisième dimension et des lignes de contour. Bien que ces méthodes produisent des visualisations ressemblant à des cartes géographiques, elles ne peuvent pas être adaptées aux arbres sans créer des régions non connexes.

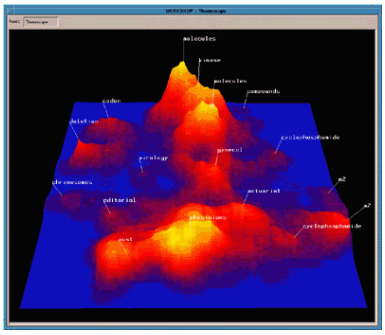
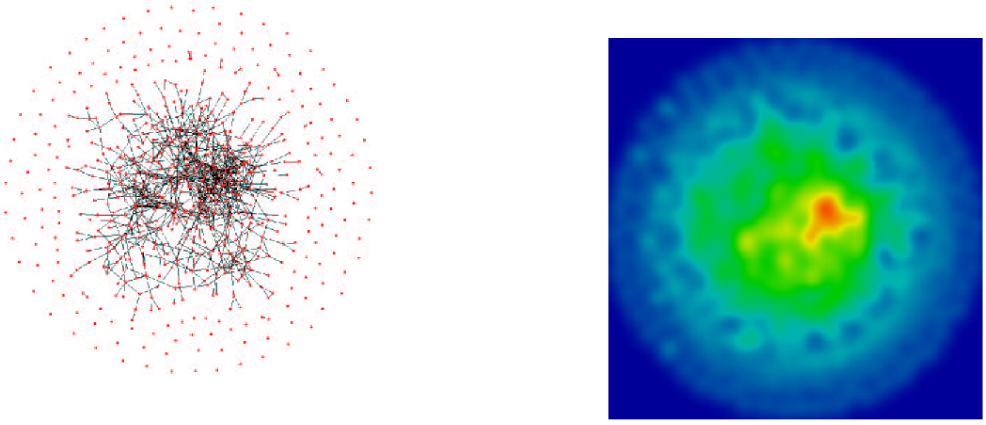


FIGURE 2.20: *Themescapes* [200]



(a) (b)

FIGURE 2.21: *Graph splatting* [185] : (a) graphe initial, (b) même graphe après traitement.

Représentations à l'aide de courbes fractales La génération de cartes virtuelles basées sur des modèles fractals est très souvent utilisée dans le domaine des jeux vidéos [123]. Jusqu'à présent, aucune solution n'a été proposée pour adapter ce type de méthodes à la visualisation de données structurées sous forme de cartes géographiques. Cependant, l'utilisation de courbes fractales a déjà été proposée par Wattenberg [195]. Une méthode basée sur ses remarques a été développée par Muelder et Ma [130] (voir 2.22). Ils commencent par fragmenter de façon hiérarchique un graphe ce qui leur permet de trouver un ordre sur les sommets. Ensuite, ils placent ces sommets le long d'une courbe fractale de façon à ce que les sommets appartenant à une même communauté soient plus proches les uns des autres. Cette méthode a été conçue pour visualiser de grands graphes. Les auteurs n'ont donc pas essayé de mettre en avant la hiérarchie induite par la fragmentation ni de donner à leur représentation un aspect géographique. La méthode que nous allons proposer permet de remédier à cela.

Avant de la décrire, nous allons donner une vue d'ensemble des techniques représentant les arbres sous forme de régions. Bien que celles-ci n'aient pas été conçues de façon à ressembler à des cartes géographiques, nous pensons qu'elles ont quand même un grand

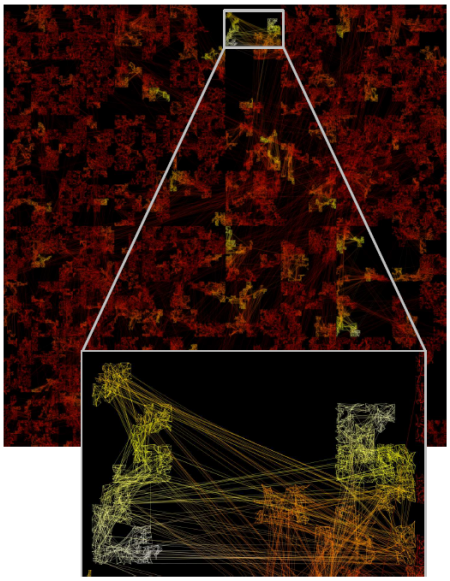


FIGURE 2.22: Dessin de graphe à l'aide d'une courbe fractale [130]

intérêt pour notre exposé.

2.2.1.2 Méthodes basées sur l'utilisation de régions pour représenter un arbre

La visualisation des sommets d'un arbre sous forme de régions a fait l'objet de nombreux travaux. Pour une description plus complète de ces techniques, nous convions le lecteur à se référer à l'article [85]. Dans notre exposé, nous allons considérer deux grandes catégories : les cartes dont les régions sont adjacentes et celles dont les régions sont imbriquées. Ensuite, nous verrons les propriétés induites par les méthodes décrites.

Régions adjacentes À notre connaissance, la première méthode de cette catégorie date de 1983 et est connue sous le nom d'*Icicle plots* [119]. Elle consiste à placer les enfants d'un sommet à côté de ce sommet de façon à ce que l'axe des abscisses représente la profondeur de la hiérarchie et l'axe des ordonnées les fratries.

Une alternative consiste à placer les régions selon des coordonnées polaires. Dans ce cas, ce sont les distances par rapport au centre qui représentent la profondeur dans l'arbre et les angles qui représentent les fratries. Cette méthode a été introduite sous le nom de *Information slices* [4]. Les *Sunbursts* [170] en sont des variantes où la totalité du cercle est utilisée ce qui permet d'avoir plus de niveaux affichés sur une seule image (voir figure 2.23). En contrepartie, l'aire des régions ne reflète plus leur taille (*i.e.* la somme des tailles de leurs enfants).

Un avantage de ces méthodes est que les étiquettes des régions peuvent être placées à l'intérieur de celles-ci sans gêner la visibilité de leurs enfants. De plus, elles permettent une visualisation plus rapide de la profondeur d'un sommet. Ceci n'est pas le cas des techniques produisant des régions imbriquées que nous allons étudier maintenant.

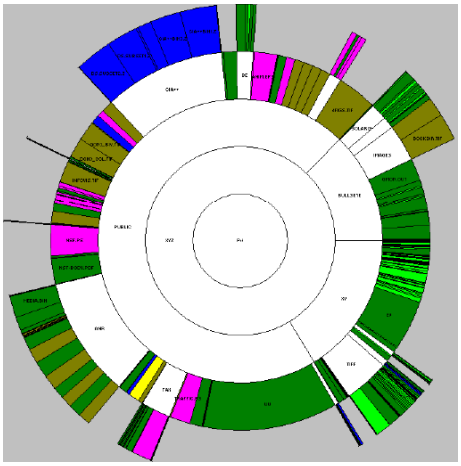


FIGURE 2.23: *Sunburst* [170]

Régions imbriquées Les *treemaps* sont les représentations les plus courantes d'arbres avec des régions imbriquées. Le principe en est de diviser le plan récursivement en allant de la racine aux feuilles de l'arbre. Une bonne vue d'ensemble de Ben Shneiderman mise à jour par Catherine Plaisant est disponible en ligne [18].

La première méthode proposée permettant de produire un *treemap* est connue sous le nom de *Slice and Dice* [104] (voir figure 2.24.a). Elle consiste à diviser le plan en alternant un découpage horizontal et vertical en fonction des niveaux successifs de l'arbre. Le principal défaut de cette technique réside dans le fait qu'elle produit de longs rectangles (ratio $longueur/largeur$ élevé) ce qui rend difficile la comparaison des tailles des sommets. Les *Squarified treemap* [35] ont été introduits pour pallier ce problème. Cette approche permet de construire des régions dont le ratio $longueur/largeur$ est aussi proche de 1 que possible. En revanche, les sommets sont ordonnés selon leurs tailles. Ainsi, leur ordre initial n'est pas pris en compte ce qui rend la visualisation peu stable dans le cas d'arbres dynamiques (voir figure 2.24.b). Le *Strip layout* [18] est un compromis entre ces deux méthodes : les ratios $longueur/largeur$ sont en moyenne plus éloignés de 1 que ceux du *Squarified treemap* mais l'ordre des sommets est pris en compte (voir figure 2.25). Les *Quantum treemaps* [18] utilisent des rectangles dont la hauteur et la largeur sont des multiples d'un même nombre, ce qui aide à la comparaison de leurs tailles. Les *Mixed treemaps* [191] utilisent la méthode *Slice and Dice* pour les plus hauts niveaux de l'arbre et la méthode *Squarified* pour les plus bas (voir figure 2.24.c).

Les *Voronoï treemaps* [11] divisent récursivement le plan, tout comme les méthodes précédentes, mais en utilisant un pavage de Voronoï, ce qui produit des polygones complexes à la place des rectangles (voir figure 2.26.a).

Enfin, le *City layout* [197] est basé sur la métaphore d'une carte citadine dans laquelle les bâtiments représentent les feuilles de l'arbre. Ils sont ensuite regroupés en quartiers, puis ces quartiers sont eux aussi regroupés en quartiers plus grands, *etc.* en accord avec la hiérarchie induite par l'arbre (voir figure 2.26.b) [19].

18. http ://www.cs.umd.edu/hcil/treemap-history/index.shtml
19. Ce type de technique permet donc de produire une sorte de *treemap* dans lequel une troisième

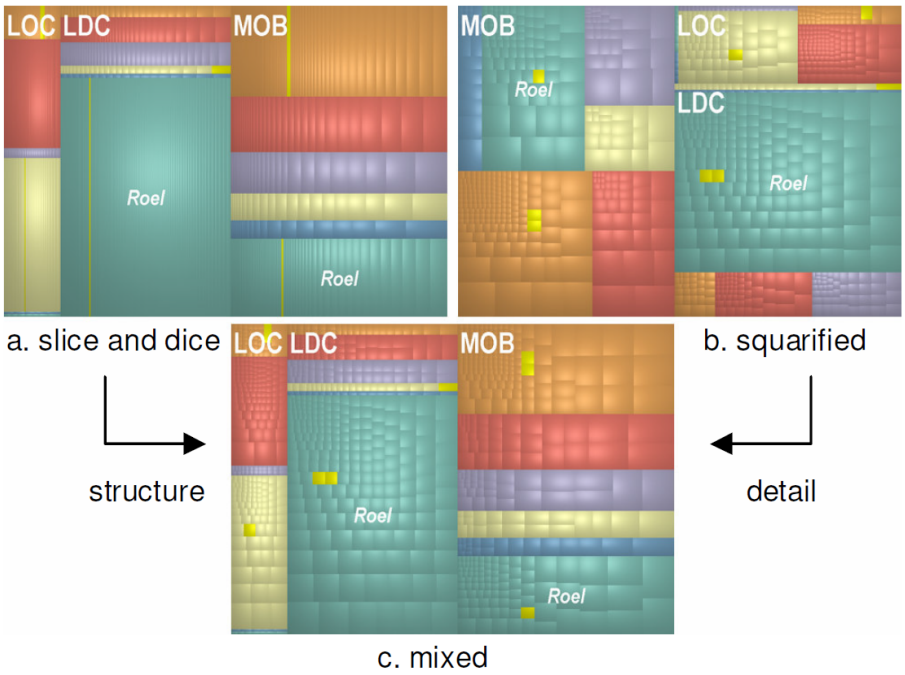


FIGURE 2.24: (a) *Slice and Dice* [104], (b) *Squarified treemap* [35], (c) *Mixed treemaps* [191].

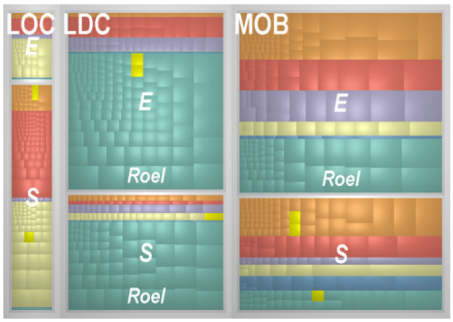
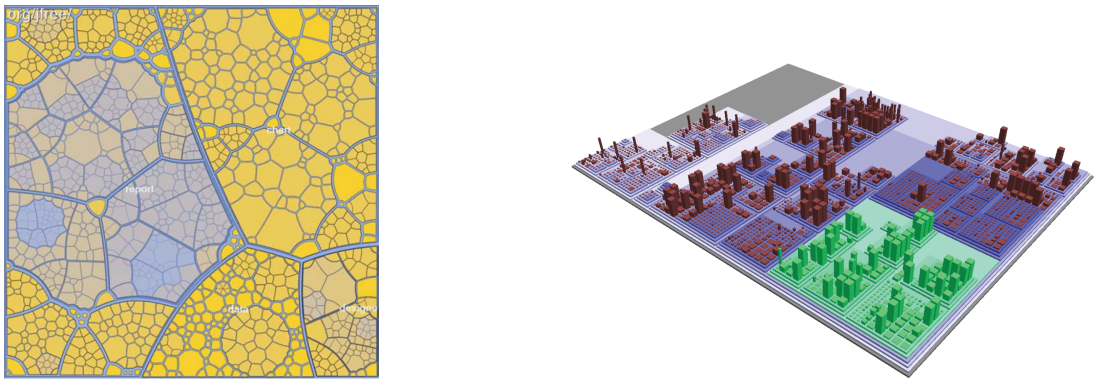


FIGURE 2.25: *Strip layout* [18] (figure produite dans [191])

Propriétés Parmi les nombreuses propriétés qui ont un impact sur la visualisation dynamique des arbres, nous en avons sélectionné quelques-unes qui nous semblent particulièrement importantes pour les représentations des sommets sous forme de régions :

- Imbrication des régions : les enfants d'un sommet sont placés à l'intérieur de la région correspondant à ce sommet.
- Ratio *longueur*/*largeur* : 1 est la valeur idéale parce qu'elle facilite la comparaison des tailles des régions.
- Corrélation des aires des régions avec les tailles des sommets : idéalement, l'aire d'une région est proportionnelle au nombre de ses enfants.
- Stabilité : une visualisation est stable si le déplacement d'une région est faible lorsque

dimension a été ajoutée (hauteur des bâtiments) afin de représenter un attribut des feuilles de l'arbre.



(a) (b)

FIGURE 2.26: (a) *Voronoï treemap* [11], (b) *City layout* [197].

l'arbre évolue [20].

Le tableau 2.1 résume le respect de ces propriétés pour chacune des visualisations décrites ci-dessus.

	Imbrication des régions	Ratio *longueur*/*largeur* proche de 1	Corrélation des aires avec les tailles	Stabilité
Icicle plots			X	X
Informations Slices			X	X
Starburst				X
City Layout	X			/
Slice and Dice Treemap	X		X	X
Squarified Treemap	X	X	X	
Strip Treemap	X	/	X	/
Quantum Treemap	X	/	X	/
Mixed Layout Treemap	X	/	X	/
Voronoï Treemap	X	/	X	X

TABLE 2.1: Récapitulatif des propriétés d'affichage en fonction des différentes techniques mentionnées plus haut : un X signifie que la propriété est complètement respectée, un / signifie qu'elle est partiellement respectée et une case vide signifie qu'elle ne l'est pas du tout.

20. La variation de la forme des régions pourrait aussi être prise en compte dans une définition plus exhaustive de la stabilité. Pour une méthode produisant des formes rectangulaires, cela reviendrait à comparer les variations des ratios *longueur*/*largeur* de chaque région. Cependant, comme la méthode que nous allons proposer produit des formes concaves, l'évaluation d'une telle propriété nous semble délicate. Ce problème reste donc ouvert.

La visualisation que nous allons décrire maintenant satisfait, au moins partiellement, toutes ces propriétés.

2.2.2 Vue d'ensemble de la méthode

Il y a quelques années, Benoît Mandelbrot [125] a étudié la longueur de la côte de la Bretagne. Il s'est très vite aperçu qu'en donner un approximation n'était pas aussi trivial qu'il y paraît. En effet, imaginons que quelqu'un mette bout à bout des bâtons d'un mètre le long de cette côte, il va obtenir une première mesure. Si cette même personne réitère l'opération avec des bâtons de cinquante centimètre, elle obtiendra une nouvelle mesure, plus grande et plus précise. Et cette mesure sera d'autant plus grande et plus précise que les bâtons seront petits. Fort de cette observation, Mandelbrot a pu concevoir les fondations mathématiques d'une modélisation de ce phénomène : les fractales.

Une **fractale** est un objet tel que chaque sous-partie de cet objet est aussi une fractale (une partie de la côte bretonne ressemble à l'ensemble de la côte bretonne). Un tel modèle peut donc être utilisé pour créer des formes ressemblant à des régions géographiques. Le problème qui se pose maintenant est de trouver un moyen de projeter notre arbre dessus. Nous proposons une solution basée sur des courbes (fractales) permettant de remplir le plan (*2D space filling curves*). La figure 2.27 montre les premières étapes de la production d'une telle courbe, la courbe de Hilbert.

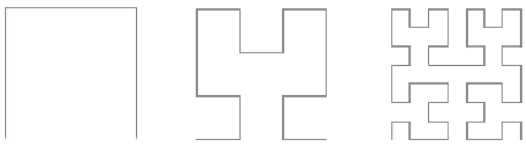
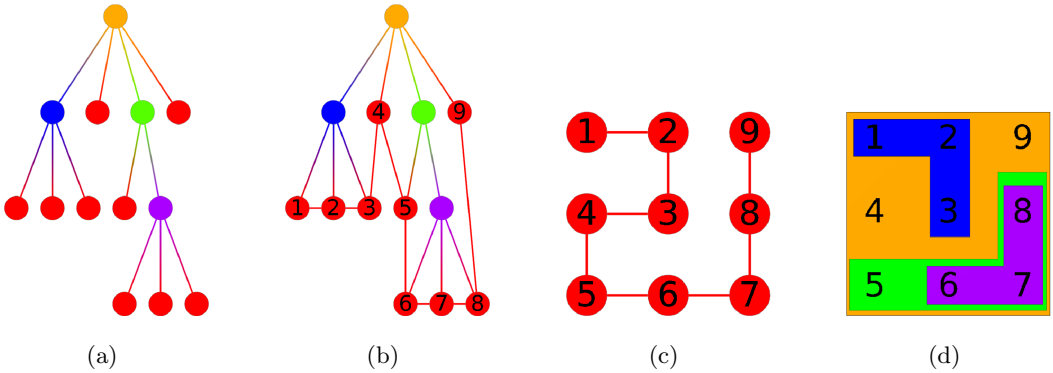


FIGURE 2.27: Premières étapes de la production d'une courbe de Hilbert.

L'utilisation de ce type de courbe va nous permettre de donner un aspect géographique à notre visualisation. Nous devons maintenant trouver un moyen de projeter notre arbre sur une telle courbe. La figure 2.28 montre l'ensemble du processus adopté. L'arbre (a) pris en entrée nous fournit un ordre sur ses feuilles (b). Cette ordre nous permet de construire une ligne rouge que nous projetons le long de la courbe fractale (c). Cette ligne est particulièrement intéressante car elle nous assure que les régions correspondant à des sommets autres que les feuilles seront connexes si elle est représentée de façon planaire (sans croisement). Elle nous permet donc d'obtenir des régions imbriquées. Enfin, nous pouvons créer les régions (d).

Maintenant que nous avons vu l'idée sous-jacente à la solution proposée, nous allons revenir sur les points délicats de la production de la visualisation.



(a) (b) (c) (d)

FIGURE 2.28: Processus de création de la visualisation : (a) nous prenons un arbre en entrée, (b) nous en déduisons un ordre linéaire de ses feuilles, (c) nous plaçons ces feuilles le long d'une courbe fractale en fonction de cet ordre, (d) nous construisons les régions.

2.2.3 Algorithmes

Le choix de la courbe fractale est un point particulièrement important de notre méthode. C'est pourquoi nous allons lui consacrer la sous-section suivante. Ensuite, nous verrons comment positionner les feuilles de l'arbre le long de celle-ci et ainsi créer des régions (sous-section 2.2.3.2). Puis nous expliquerons comment créer les frontières (sous-section 2.2.3.3). Pour finir, nous présenterons la méthode d'étiquetage adoptée (sous-section 2.2.3.4).

2.2.3.1 Sélection de la courbe et création des régions

Le choix de la courbe est un point fondamental de notre algorithme car c'est elle qui permet de donner un aspect géographique aux régions. Tout d'abord, elle doit être simple : un croisement pourrait en effet induire une déconnexion de certaines région. Ensuite, elle doit pouvoir s'étendre indéfiniment afin que l'on ne soit pas restreint sur la taille de l'arbre pris en entrée. Les courbes respectant ces deux contraintes sont connues sous le nom de *2D space filling curves*. Plus formellement, elles sont définies comme une fonction $h : \mathbb{N} \to \mathbb{R}^2$ telle que la distance euclidienne entre $h(i)$ et $h(i+1)$ est 1.

Grâce à cette définition, on peut définir la mesure *Worst-Case Locality* [92] par :

$$WL = \lim_{k \to \infty} \sup_{i,j \in \mathbb{N}_{k^2}} \frac{d(h(i), h(j))^2}{|i-j|}$$

où $d(h(i), h(j))$ représente la distance euclidienne dans $\mathbb{R}^2$. Le carré vient du fait que si la distance maximale dans $\mathbb{N}$ est $O(k^2)$, alors la distance maximale dans $\mathbb{R}^2$ est $O(k)$. WL indique combien les sommets proches dans l'espace unidimensionnel de départ seront proches dans l'espace bidimensionnel sur lequel ils sont projetés. Si cette valeur est finie, on dit que la courbe respecte la *locality property* définie ainsi :

$$d(h(i), h(j)) < c|i-j|^{1/2}$$

où $c = WL^{1/2}$. Cette propriété nous garantie que les enfants d'un même sommet seront proches dans le placement final, ce qui n'est pas la cas si l'on utilise une spirale par exemple (dans le cas d'une grande fratrie, deux frères peuvent se retrouver au pire à une distance correspondant au diamètre de la spirale). Ainsi, on respecte la première loi géographique de Skupin [166] : « *Everything is related to everything else, but closer things are more closely related* ». La courbe de Hilbert rentre dans la catégorie des courbes ayant cette propriété ($c = \sqrt{6}$).

Malheureusement, cette courbe ne nous convient pas car elle ne permet pas de créer des régions ayant un aspect géographique. La prise en compte de ce paramètre n'est pas aussi facile que le choix d'une *space fillling curve* respectant la *locality property*. En effet, le critère correspondant dépend seulement de la perception humaine et il n'existe pas à notre connaissance de méthode formelle capable de l'évaluer. Nous avons étudié de nombreuses courbes fractales et celle qui semble la plus adaptée à notre problématique est la courbe « flocon de neige » de Gosper. Elle vérifie la *locality property* avec $c = \sqrt{6.35}$. Les première étapes de sa génération sont décrites par la figure 2.29.

FIGURE 2.29: Premières étapes de la production d'une courbe de Gosper.

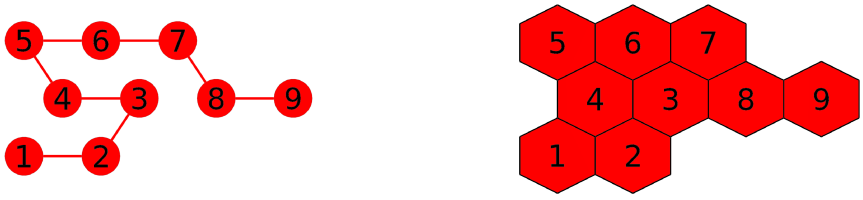
2.2.3.2 Positionnement initial des feuilles de l'arbre

Un simple parcours en profondeur nous permet de déterminer l'ordre dans lequel les feuilles vont être placées le long de la courbe. Le lecteur pourra se référer par exemple au livre [24] pour trouver cet algorithme.

Une fois l'ordre des feuilles trouvées, nous plaçons celles-ci le long de la courbe de Gosper comme le montre la figure 2.30.a. Ensuite, le plan est divisé grâce à un diagramme de Voronoï afin d'obtenir une région pour chacune des feuilles. Dans le cas particulier de la courbe de Gosper, cette étape produit des hexagones comme le montre la figure 2.30.b. Afin d'accélérer cette étape, nous pouvons directement calculer ces hexagones sans passer par le diagramme de Voronoï. Cependant, cette solution est moins générique car toutes les courbes ne divisent pas le plan en hexagones. On peut remarquer qu'il est facile d'adapter la taille des feuilles à certaines valeurs prises en entrée en leur attribuant un nombre de cases correspondant à ces valeurs.

Nous pouvons maintenant créer les régions correspondant aux autres sommets de l'arbre. Nous procédons du bas vers le haut de l'arbre en fusionnant les régions des enfants d'un sommet pour créer leur propre région comme le montre la figure 2.31. La figure 2.32 nous indique quant à elle que cette méthode permet bien de préserver le principe d'imbrication des régions que nous avons énoncé comme une propriété nécessaire à la visualisation que nous voulons mettre en place.

En utilisant cette méthode, nous pouvons déjà produire de jolies cartes comme celle de la figure 2.33 très rapidement (moins d'une seconde pour l'arbre du système de fichiers



(a) (b)

FIGURE 2.30: Placement des feuilles ordonnées le long de la courbe : (a) Feuilles positionnées en fonction de la courbe. (b) Régions correspondantes créées grâce à un diagramme de Voronoï.

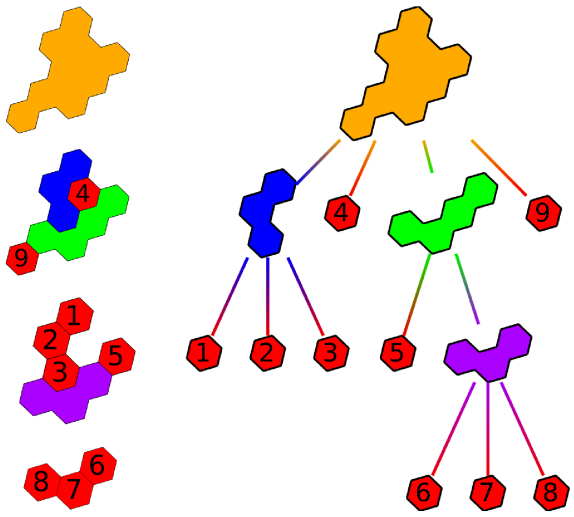


FIGURE 2.31: Arbre initial où les sommets sont représentés par les régions qui leur correspondront après la création du diagramme Voronoï.

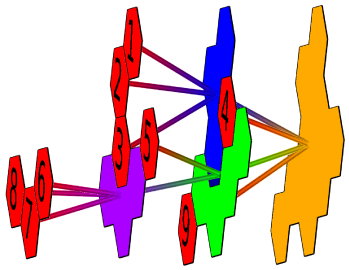


FIGURE 2.32: Préservation du principe d'imbrication des régions.

de tulip [6] composé de 6 000 sommets). Les régions des parents sont affichées par-dessus celles de leurs enfants avec une certaine transparence ce qui permet de distinguer la hiérarchie. La taille des frontières est déterminée en fonction de la profondeur du sommet dans l'arbre ce qui facilite aussi la visualisation de l'arbre. Malheureusement, comme on peut facilement le remarquer sur la figure, l'ajout de ces artifices ne permet pas une représentation claire de plus de trois niveaux de la hiérarchie. Afin de remédier à cela, nous pouvons mettre en place un système de filtre des régions correspondant aux niveaux

les plus élevés de la hiérarchie en fonction d'un niveau de zoom par exemple. Cependant, cette solution n'est pas satisfaisante. Non seulement elle entraîne des problèmes difficiles à résoudre à cause de la forme des régions, mais en plus elle ne permet pas d'avoir une carte à la fois globale et détaillée [21]. C'est pourquoi nous allons maintenant présenter une méthode permettant de redéfinir les régions de façon à ce que les frontières de différents niveaux ne se chevauchent plus.

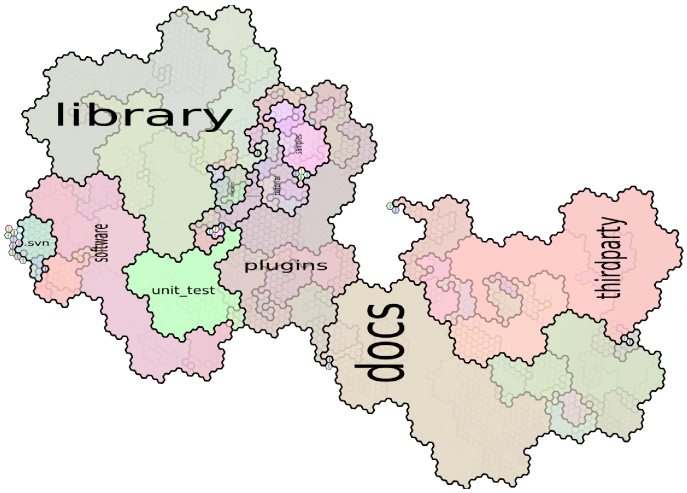


FIGURE 2.33: Hiérarchie du système de fichiers du logiciel tulip [6]. Les niveaux de l'arbre sont visibles grâce à un changement des tailles des frontières en fonction de la profondeur des sommets correspondants dans l'arbre et grâce à l'attribution d'une transparence aux régions.

2.2.3.3 Création des frontières

Une façon efficace de représenter l'imbrication des régions consiste à séparer les frontières délimitant les régions de différents niveaux de la hiérarchie. Par exemple, sur la figure 2.28.d, la frontière de la région $\{6, 7, 8\}$ ne chevauche pas celle de son parent $\{5, 6, 7, 8\}$. Ainsi, il est plus facile de déterminer visuellement la hiérarchie des régions, *e.g.* le chemin allant de la racine de l'arbre à n'importe quel sommet courant. Pour rester sur la métaphore géographique, on peut voir les frontières comme des lignes de contour démarquant des régions de même altitude, *i.e.* des sommets de même profondeur dans l'arbre.

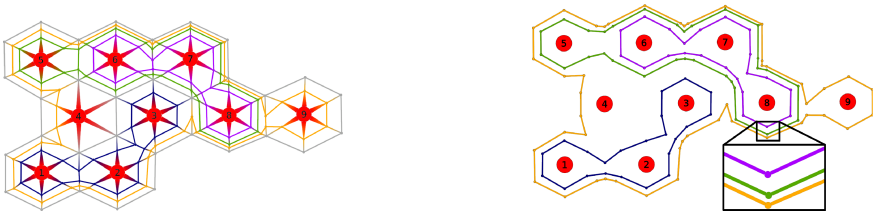
Pour arriver à produire de telles frontières, nous produisons un graphe $G = (L \cup V', E)$ à partir des hexagones produits grâce au diagramme de Voronoï. L'ensemble L correspond aux feuilles de l'arbre $T = (V, A)$ initial. V' et E sont respectivement un ensemble de sommets et un ensemble d'arêtes qui vont être insérés dans G. Les premiers sommets ajoutés à V' correspondent aux sommets des hexagones issus du diagramme de Voronoï. Ils sont liés par des arêtes (ajoutées à l'ensemble E) aux éléments de L qui leur sont les plus proches. Le graphe G ainsi obtenu est représenté par la figure 2.34.a.

21. Il semblerait aussi que le problème puisse être en partie résolu grâce à l'ajout d'ombres. Une telle technique a déjà été employée afin d'améliorer les visualisations de type *treemap* (*cushion treemaps*) [187]. Il serait intéressant de l'adapter a nos cartes afin de voir si le résultat est aussi efficace.

(a) (b)

FIGURE 2.34: (a) Graphe initial contenant les feuilles de l'arbre et les sommets des hexagones issus du diagramme de Voronoï. (b) Même graphe après avoir ajouté un chemin correspondant à la région violette $\{6, 7, 8\}$.

Ensuite, nous ajoutons successivement des sommets et des arêtes afin de former des chemins correspondant aux frontières des régions. Pour cela, nous procédons du bas vers le haut de l'arbre. Dans notre exemple, c'est donc le chemin violet qui est ajouté en premier. Chaque arête déjà présente dans G et qui entoure les sommets devant être inclus dans la région violette $(6, 7, 8)$ est divisée en deux et un nouveau sommet est ajouté au milieu, *i.e.* l'arête (u, v) est remplacée par les arêtes (u, w) et (w, v) où w est un nouveau sommet de G. Nous procédons à cette division en tournant dans le sens des aiguilles d'une montre autour des sommets de la région violette, ce qui nous donne le graphe G de la figure 2.34.b. Si l'on applique ce processus récursivement du bas au haut de l'arbre, on obtient le graphe représenté par la figure 2.35.a. Contrairement à la méthode décrite dans la partie précédente, les frontières ne se chevauchent plus.



(a) (b)

FIGURE 2.35: (a) Graphe dans lequel les chemins représentant toutes les frontières ont été ajoutés. (b) Même graphe où les sommets et les arêtes n'appartenant pas aux chemins représentant les frontières ont été supprimés.

Nous pourrions dès lors nous contenter d'utiliser le placement obtenu et supprimer les sommets qui n'appartiennent pas aux chemins des frontières pour créer notre visualisation. La représentation ainsi obtenue est affichée sur la figure 2.35.b. Cependant, comme nous avons successivement divisé les arêtes en deux, les frontières se retrouvent de plus en plus proches les unes des autres au fur et à mesure qu'elles représentent des sommets plus élevés dans la hiérarchie. L'encadré de la figure 2.35.b illustre ce phénomène : les frontières des régions verte et jaune sont plus proches l'une de l'autre que celle représentant la région violette. Il devient ainsi d'autant plus difficile de distinguer l'imbrication des régions que l'arbre est profond. Afin de pallier ce problème, nous pourrions diviser les arêtes uniformément en fonction de la profondeur de la feuille dans l'arbre. Cependant, cette solution ne serait pas satisfaisante car la taille des régions correspondant aux feuilles

dont la profondeur est élevée serait considérablement réduite.

Afin de résoudre ce problème, nous commençons par transformer le graphe G de la figure 2.35.a en un graphe dont les faces internes sont triangulées. Pour cela, nous ajoutons des arêtes. Le graphe ainsi créé est représenté par la figure 2.36.

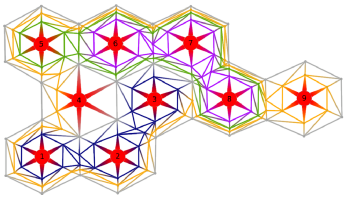


FIGURE 2.36: Triangulation du graphe de la figure 2.35.a.

Ensuite, nous appliquons l'algorithme de Tutte [182] afin de modifier la position des sommets de façon à ce que les arêtes délimitant les régions soient approximativement équidistantes. Cet algorithme consiste à itérer une fonction plaçant tous les sommets du graphe au barycentre de ses voisins. Il permet de représenter de façon planaire les graphes planaires et selon Tutte, il converge vers une configuration stable après un nombre linéaire d'itérations. Dans notre cas, le graphe est déjà représenté de façon planaire et les positions finales sont très proches des positions initiales ce qui raccourcit considérablement le temps d'exécution de l'algorithme. Grâce à une implémentation parallèle, nous avons pu placer le graphe de 100 000 sommets en quelques secondes [22]. La figure 2.38.a représente le graphe obtenu avec l'aide de cette méthode et la figure 2.38.b montre le même graphe où les sommets et les arêtes ne faisant pas partie des chemins représentant les frontières ont été supprimés. Il est important de remarquer ici que l'on peut pondérer les barycentres afin, par exemple, de laisser plus d'espace autour des frontières correspondant aux niveaux les plus élevés de la hiérarchie.

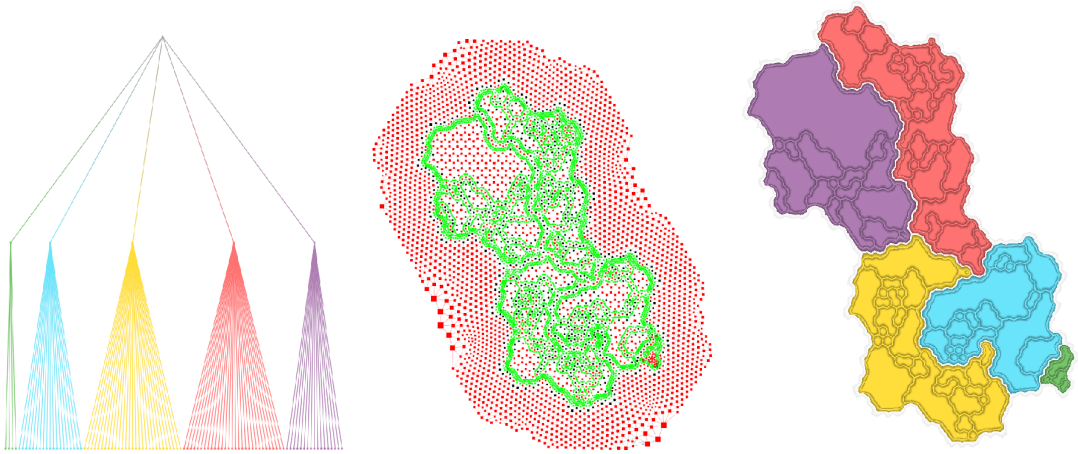
(a) (b)

FIGURE 2.37: (a) Graphe de la figure 2.36 après l'algorithme de Tutte. (b) Même graphe où les sommets et les arêtes n'appartenant pas aux chemins représentant les frontières ont été supprimés.

Nous obtenons ainsi des frontières lisses et ne se chevauchant pas pour chaque sommet de l'arbre. Nous pouvons créer des polygones correspondant à ces sommets et appliquer une texture aux contours afin de faciliter la visualisation des régions imbriquées. La figure 2.38 donne une vue d'ensemble du processus sur un système de fichiers composé de 105 sommets. Comme on peut le remarquer, les feuilles sont représentées par des régions

22. Sur un Intel Core Q9300 2.53Ghz

de différentes tailles. En fait, nous leur avons ajouté dans l'arbre un nombre d'enfants proportionnel aux tailles des fichiers correspondants. Puis nous avons supprimé les régions correspondant à ces enfants fictifs. Cette technique permet donc d'attribuer une taille aux régions en fonction d'une certaine propriété des feuilles des données initiales.



(a) (b) (c)

FIGURE 2.38: Exemple du processus de création de la carte : (a) Arbre initial correspondant à la hiérarchie d'un système de fichiers. (b) Graphe triangulé positionné grâce à l'algorithme de Tutte. (c) Visualisation finale dans laquelle une texture a été appliquée le long des frontières.

Maintenant que nous avons trouvé une technique efficace pour créer les régions, il nous manque encore un système performant d'étiquetage, sans quoi notre visualisation ne peut être utilisée. La sous-section suivante décrit la méthode adoptée.

2.2.3.4 Étiquetage des régions

Notre méthode est basée sur la combinaison de deux approches d'étiquetage. La première permet d'afficher les étiquettes sur des régions correspondant aux sommets de l'arbre ayant une profondeur donnée. Le seconde consiste à répéter les étiquettes le long des frontières des régions correspondantes afin de garder une idée du chemin parcouru lorsque l'on augmente le niveau de zoom.

Étiquetage des régions d'une certaine profondeur Cette première approche a pour but d'afficher les étiquettes à l'intérieur des régions correspondantes. Étant donné qu'un tel affichage cause de nombreux problèmes de chevauchements s'il est réalisé pour tous les niveaux de l'arbre, nous nous limitons ici à un seul niveau. Nous verrons ensuite dans la partie interaction (section 2.2.4) comment l'utilisateur de notre système peut modifier le niveau affiché.

Un moyen efficace pourrait être de créer des enfants factices pour chaque sommet de l'arbre et d'afficher ensuite les étiquettes dans les régions correspondant à ces sommets. Ce n'est pas l'approche que nous avons utilisée car nous considérons qu'elle s'éloigne trop de

la métaphore géographique. En effet, elle crée des régions n'ayant aucun sens vis-à-vis de nos données : lors de l'élaboration d'une carte de France, la concepteur ne va pas déformer les régions afin d'en créer une virtuelle qui recevra l'étiquette « France ».

Si l'on se restreint à l'affichage des étiquettes d'un seul sommet de la hiérarchie, une technique intuitive consiste à trouver un rectangle d'aire maximale à l'intérieur de chaque région afin d'y projeter une texture contenant l'étiquette. C'est ce que nous avons fait sur la figure 2.33. Cependant, à cause de la concavité de certaines régions, les étiquettes peuvent être très petites et difficiles à visualiser.

La solution adoptée consiste donc à trouver le plus grand **rectangle « courbé »** inscrit dans une région. Le figure 2.39 montre ce que nous entendons par rectangle « courbé » : une forme géométrique à 4 côtés dont les deux plus longs sont des courbes parallèles. Une fois cette figure déterminée, nous pouvons appliquer une texture contenant le label dessus.

FIGURE 2.39: Texture contenant une étiquette insérée dans un rectangle « courbé ».

Pour cela, nous calculons l'axe médian pour chacune des régions dont nous voulons afficher l'étiquette. Ceci nous donne un arbre dont les sommets sont affichés dans la figure 2.40 (arbre induit par les sommets autres que rouges). Pour réaliser une telle opération, nous utilisons l'algorithme présenté dans [46]. Ainsi, nous pouvons créer un rectangle « courbé » pour chaque chemin de cet arbre reliant deux feuilles. Sa hauteur sera donnée par la plus courte distance entre un sommet du chemin et un sommet de la frontière de la région afin d'assurer son inclusion. Il nous reste maintenant à sélectionner le rectangle dont l'aire est la plus importante afin de maximiser la taille de l'étiquette.

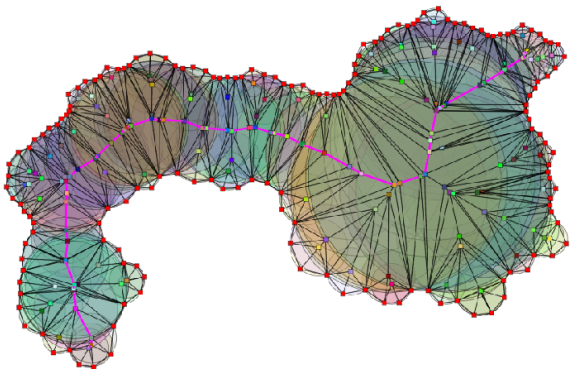


FIGURE 2.40: Les sommets de l'axe médian peuvent être vus comme les centres de cercles inclus dans une région et étant en contact avec au moins deux points de la frontière de cette région.

Les sommets de l'axe médian peuvent être vus comme les centres de cercles inscrits à la région, touchant sa frontière en au moins deux points (voir figure 2.40). Nous calculons ces cercles ainsi que leurs aires. Ensuite, pour chaque sommet v de l'arbre, nous déterminons le chemin $P_i = (v_1, v_2, ..., v_{k-1}, v, v_{k+1}, ..., v_l)$ où v_1 et v_l sont des feuilles de l'arbre. Un sommet v_{i+1} est sélectionné parmi les voisins de v_i différents du sommet v_{i-1} tel que

l'aire de son cercle soit plus élevée que celle des autres voisins de v_i. Ensuite, pour chaque chemin, nous calculons la somme des aires des cercles de ses sommets et nous sélectionnons le chemin ayant la somme la plus élevée.

Nous créons enfin un rectangle « courbé » le long de ce chemin tel que sa hauteur corresponde à deux fois la plus faible distance entre un de ses sommets et une frontière de la région. Nous appliquons alors une texture contenant l'étiquette sur cette forme ce qui nous donne l'étiquetage montré par la figure 2.41 sur le troisième niveau de la hiérarchie.

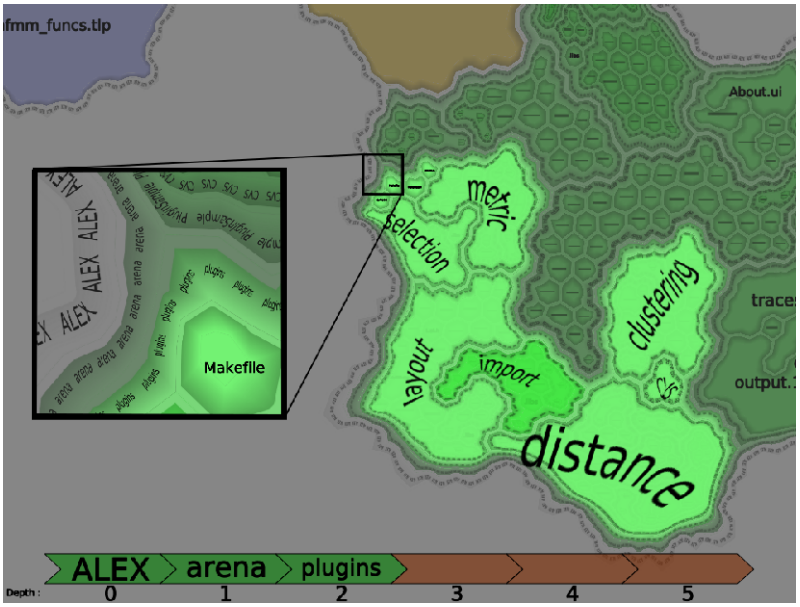


FIGURE 2.41: Étiquettes des enfants de la région *plugins* : ces enfants appartiennent au troisième niveau de la hiérarchie. L'encadré montre les étiquettes placées sur les frontières des régions imbriquées.

Afin de ne pas trop augmenter le temps de calcul des étiquettes de notre carte, les étiquettes des feuilles sont affichées simplement sans utiliser cette technique. Ceci ne perturbe en effet pas vraiment la visualisation de la carte comme le montre la figure 2.46.c.

Étiquetage de l'ensemble de la hiérarchie La seconde méthode d'étiquetage développée permet à l'utilisateur de préserver le contexte lorsque le niveau de zoom est élevé. Elle consiste à appliquer une texture contenant l'étiquette d'une région le long de sa frontière, exactement comme la hauteur est affichée le long des courbes de niveaux dans le cas des cartes géographiques. L'encadré de la figure 2.41 montre le résultat obtenu. La feuille *Makefile* est à l'intérieur de la région *Plugins*, elle-même à l'intérieur de la région *arena* qui représente un fils du sommet *ALEX*. On peut aussi facilement observer que la région *PluginSample* est au même niveau que *plugins*, et que *CVS* est au même niveau que *Makefile*.

2.2.4 Techniques d'interaction

Même si la carte que nous produisons permet de visualiser l'ensemble de l'arbre, nous devons mettre en place un système d'interaction permettant à l'utilisateur de sélectionner le niveau sur lequel les étiquettes seront affichées (*cf.* le premier paragraphe de la sous-section précédente). Pour cela, il suffit à l'utilisateur de cliquer sur la région qu'il veut mettre en évidence. Nous considérons alors cette région comme la région courante et nous affichons les étiquettes de ses enfants. Nous avons aussi ajouté une série de cases en bas de l'écran (voir figure 2.41) symbolisant la profondeur de la hiérarchie. La totalité des cases représente la profondeur maximale de l'arbre. Les cases vertes étiquetées représentent le chemin de la racine au niveau courant. Les cases marrons représentent la profondeur maximale des sous-arbres du niveau courant. Les cases grises en fin de chaîne représentent des niveaux non présents dans un sous-arbre du niveau courant. Dans la figure 2.41, il n'y a pas de cases grises : l'un des sous-arbres du sommet *plugins* atteint donc le niveau 5. Enfin, notre carte est munie d'un système de zoom contrôlé par la molette de la souris. Nous utilisons pour cela l'algorithme de van Wijk et Nuij [186] afin d'assurer sa fluidité.

Le système d'interaction mis en place ici n'est pas innovant mais permet néanmoins de montrer qu'il existe des techniques simples permettant d'utiliser notre visualisation. Pour plus d'informations sur les techniques d'interaction, nous convions le lecteur à se reporter à l'article [203].

2.2.5 Résultats

L'évaluation de notre technique de visualisation est qualitative. Nous avons mesuré différents paramètres sur le jeu de données « *what you pay for* »constitué d'une hiérarchie dynamique (nous l'expliquerons plus en détail dans la prochaine sous-section). Nous avons comparé les résultats de ces mesures avec ceux obtenus grâce à deux autres méthodes de visualisation d'arbre sélectionnées parmi les plus populaires : le *Sunburst* pour sa stabilité et le *squarrified treemap* pour son bon ratio $longueur/largeur$. Pour chaque type de visualisation, nous avons mesuré le temps de génération de la carte, le ratio $longueur/largeur$, la corrélation entre des valeurs projetées sur les feuilles et les tailles des régions correspondantes, et le déplacement des régions sur les cartes consécutives. Les figures 2.43, 2.44 et 2.45 montrent les cartes obtenues selon les différents types de visualisation pour quatre arbres consécutifs. La figure 2.42 résume les valeurs obtenues sur ces cartes.

2.2.5.1 Jeu de données

Les données sont issues du site http ://www.whatwepayfor.com/ et représentent la répartition de l'argent des taxes récoltées aux USA selon les différentes fonctions publiques. La classification comprend des **fonctions** divisées en **sous-fonctions**. Chaque sous-fonction possède différents **comptes** qui seront crédités grâce aux taxes. Nous avons donc un arbre dont le premier niveau représente la totalité des taxes, le deuxième niveau les fonctions, le troisième niveau les sous-fonctions, le quatrième les comptes. À chaque compte est associée une quantité d'argent que nous allons prendre en compte dans la

carte en créant des sommets virtuels comme nous l'avons expliqué précédemment. Nous pourrons ainsi savoir comment se répartit l'argent des taxes en comparant les tailles des régions. Un même compte pouvant appartenir à plusieurs sous-fonctions, nous avons été obligé de dupliquer les sommets correspondant dans l'arbre. Enfin, les données sont dynamiques : nous avons pu extraire les arbres de différentes années ce qui va nous permettre de tester la stabilité de notre algorithme.

2.2.5.2 Mesures

Nous allons maintenant décrire les résultats obtenus sur ces données en commentant les mesures de la figure 2.42.

		2007	2008	2009	2010
time	geomap	6,04	7,5	7,7	7,5
	treemap	0,004	0,005	0,005	0,008
	sunburst	0,03	0,04	0,039	0,04
aspect ratio	geomap	0,85 (0,12)	0,84 (0,12)	0,84 (0,13)	0,83 (0,13)
	treemap	0,76 (0,18)	0,75 (0,18)	0,76 (0,18)	0,75 (0,19)
	sunburst	0,44 (0,26)	0,41 (0,25)	0,32 (0,23)	0,46 (0,26)
area	geomap	0,99	0,99	0,99	0,99
	treemap	0,97	0,97	0,97	0,97
	sunburst	0,91	0,92	0,93	0,92
stability	geomap	0,04 (0,03)	0,09 (0,06)	0,08 (0,05)	0,09 (0,05)
	treemap	0,15 (0,1)	0,08 (0,07)	0,12 (0,11)	0,12 (0,11)
	sunburst	0,01 (0,009)	0,03 (0,02)	0,05 (0,04)	0,05 (0,04)

FIGURE 2.42: ***Time*** : temps nécessaire à la génération d'une visualisation. ***Aspect ratio*** : ratio *longueur/largeur* moyen des boîtes englobantes des régions. ***Area*** : combien l'aire d'une région correspond à la somme des tailles des feuilles de son sous-arbre. ***Stability*** : déplacement moyen des régions entre la carte de l'année précédente et la carte de l'année courante. Les écarts types sont notés en rouge.

Temps de génération de la visualisation : Comme l'indique le tableau de la figure 2.42, le principal défaut de notre méthode est le temps nécessaire au calcul. Ceci est dû au fait que nous travaillons sur un graphe plus important que l'arbre pris en entrée lors de l'étape de création des frontières, ce qui n'est pas le cas des deux autres visualisations. Par exemple, pour générer la figure 2.43, nous avons dû travailler sur un graphe contenant 33 059 sommets et 88 846 arêtes. Même si les algorithmes sont peu coûteux (diagramme de Voronoï, barycentre, fusion des polygones), la grande taille du graphe explique le temps

de calcul. En revanche, si l'on se contente du résultat obtenu sur la figure figure 2.33, alors notre carte est compétitive en temps de calcul avec les deux autres méthodes.

Ratio *longueur/largeur* **:** Afin d'obtenir des ratios compris entre 0 et 1, nous construisons une boîte englobante pour chaque région (rectangle) et nous divisons la taille de ses côtés les plus longs (longueur) par la taille de ses côtés les plus courts (largeur). Ensuite, nous faisons la moyenne des valeurs obtenues pour avoir un ratio sur l'ensemble de la visualisation. Comme le montre le tableau 2.42, notre visualisation surpasse le *squarified treemap (S.T.)* (les ratios moyens sont plus proches de 1) ce qui nous a paru surprenant. En effet, les *S.T.* ont justement été conçus dans le but d'avoir des ratios proches de 1. Ceci semble être dû au fait que bien que la plupart de leurs cases aient un ratio convenable, certaines en ont un qui s'éloigne considérablement de 1. Au contraire, la courbe de Gosper que nous avons utilisée, grâce à sa nature fractale, nous garantit que les régions suffisamment grandes auront toujours un ratio proche de 1. Cependant, cet argument en faveur de notre carte doit être utilisé avec modération. En effet, des régions convexes ayant un tel ratio proche de 1 ont des tailles faciles à comparer. Les régions que nous produisons ont des formes concaves ce qui limite la comparaison de leurs tailles, indépendamment du fait qu'elles possèdent un bon ratio.

Corrélation des aires des régions avec les sommes des tailles des feuilles de leurs sous-arbres : Les aires de la carte produite si l'on ne sépare pas les frontières (voir figure 2.33) sont parfaitement corrélées avec les tailles des sommets de l'arbre. Cependant, l'étape suivante, dans la mesure où elle déplace les sommets, modifie cette propriété. Nous avons donc calculé les aires de tous les polygones formés par les frontières des régions afin de les comparer aux tailles des sommets correspondants. Comme le montre le tableau 2.42, les trois visualisations ont toutes de fortes corrélations entre les tailles attendues et les tailles réelles. La nôtre a un léger avantage mais il faut remarquer que si l'on supprime les larges marges dédiées aux étiquettes du *S.T.*, il obtient des résultats supérieurs (nous pourrions par exemple le doter d'un étiquetage similaire à celui que nous avons mis en place sur notre visualisation géographique). Les plus bas résultats obtenus par les *Sunbursts* sont dûs au fait que la taille est projetée sur l'angle ce qui entraîne une aire plus grande pour les régions basses de la hiérarchie. Nous pourrions cependant le modifier pour pallier ce défaut, comme c'est le cas dans [4].

Stabilité : Les données que nous utilisons nous permettent de créer un arbre par année entre 2006 et 2010. Bien que la plupart des sommets soient les mêmes, on observe cependant de forts changements sur les comptes et les sommes qui leur sont associées. Nous avons évalué la stabilité des trois méthodes en nous basant sur le déplacement du centre de la boîte englobante des régions sur les visualisations successives. Ces déplacements sont normalisés en les divisant par la diagonale de la boîte englobante de la totalité de la carte car sa longueur correspond au déplacement maximal possible. Comme nous l'attendions, c'est le *Sunburst* qui donne les meilleurs résultats et le *S.T.* les moins bons. Ceci est dû au fait que l'algorithme permettant la production des *S.T.* réordonne les enfants d'un sommet en fonction de leurs tailles afin d'optimiser le ratio. Les résultats obtenus grâce à notre carte (*G.T.*) sont entre ceux des deux autres visualisations. En effet, une éventuelle

modification d'un sommet placé au début de la courbe de Gosper entraîne le déplacement de tous les autres sommets. Même si les distances sont courtes, ceci explique que les résultats sont moins bons que ceux du *Sunburst*. Cependant, ces modifications sont quand même assez faibles et n'affectent pas la préservation de la carte mentale sur les visualisations successives. Cette observation est illustrée par les figures 2.43, 2.44 et 2.45 dans lesquelles des flèches mettent l'accent sur l'évolution de la région correspondant au sommet « *Commerce and Housing Credit* ».

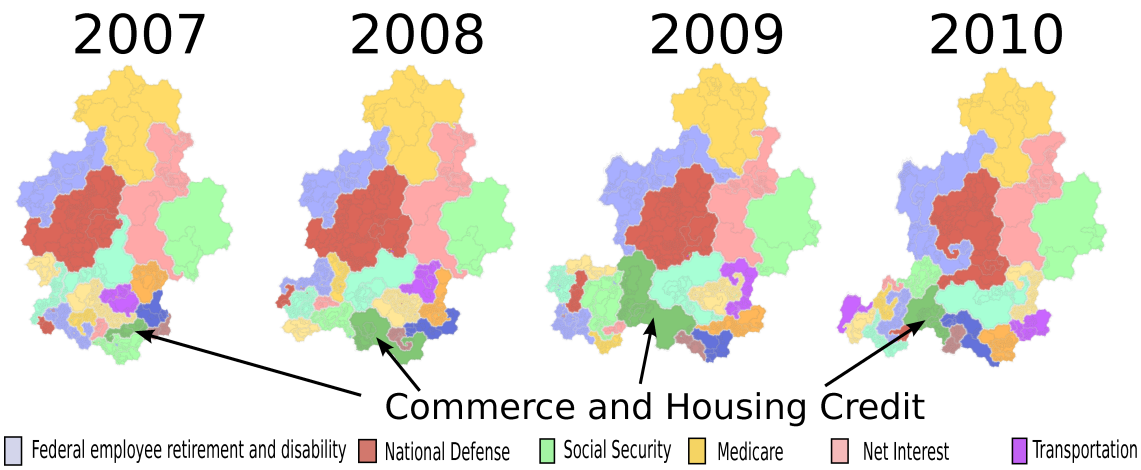


FIGURE 2.43: *Geographical treemaps* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010

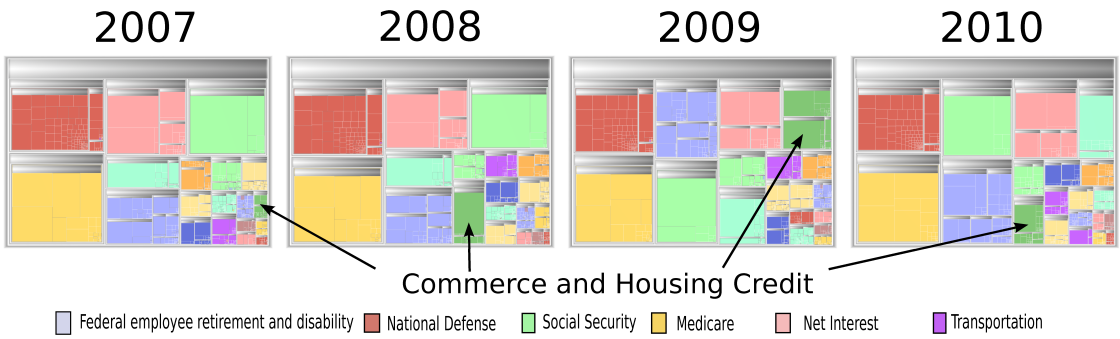


FIGURE 2.44: *Squarified treemaps* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010

2.2.5.3 Discussion

L'image 2.46 montre la carte de la bibliothèque standard du langage *Python*. Le but initial de notre travail était de créer des visualisations ressemblant à des cartes géographiques et ces images semblent valider notre technique sur ce point. De plus, toutes les personnes ayant manipulé notre outil ont effectivement reconnu la métaphore, ce qui nous amène à penser que la solution proposée remplit cette condition. L'enthousiasme manifesté par les utilisateurs nous a poussé à effectuer les mesures que nous avons présentées dans la

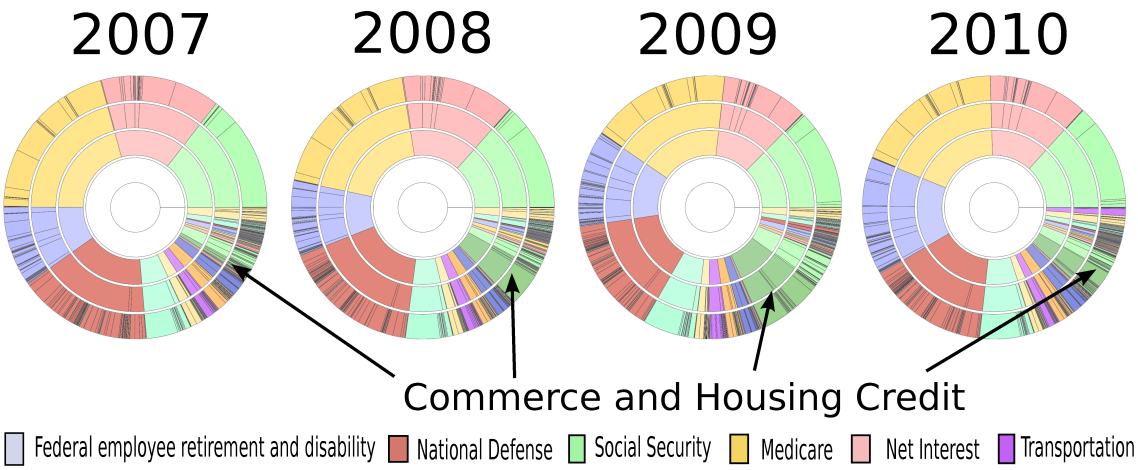


FIGURE 2.45: *Sunbursts* représentant des arbres consécutifs issus du jeu de données *what we pay for* pour les années 2007, 2008, 2009 et 2010

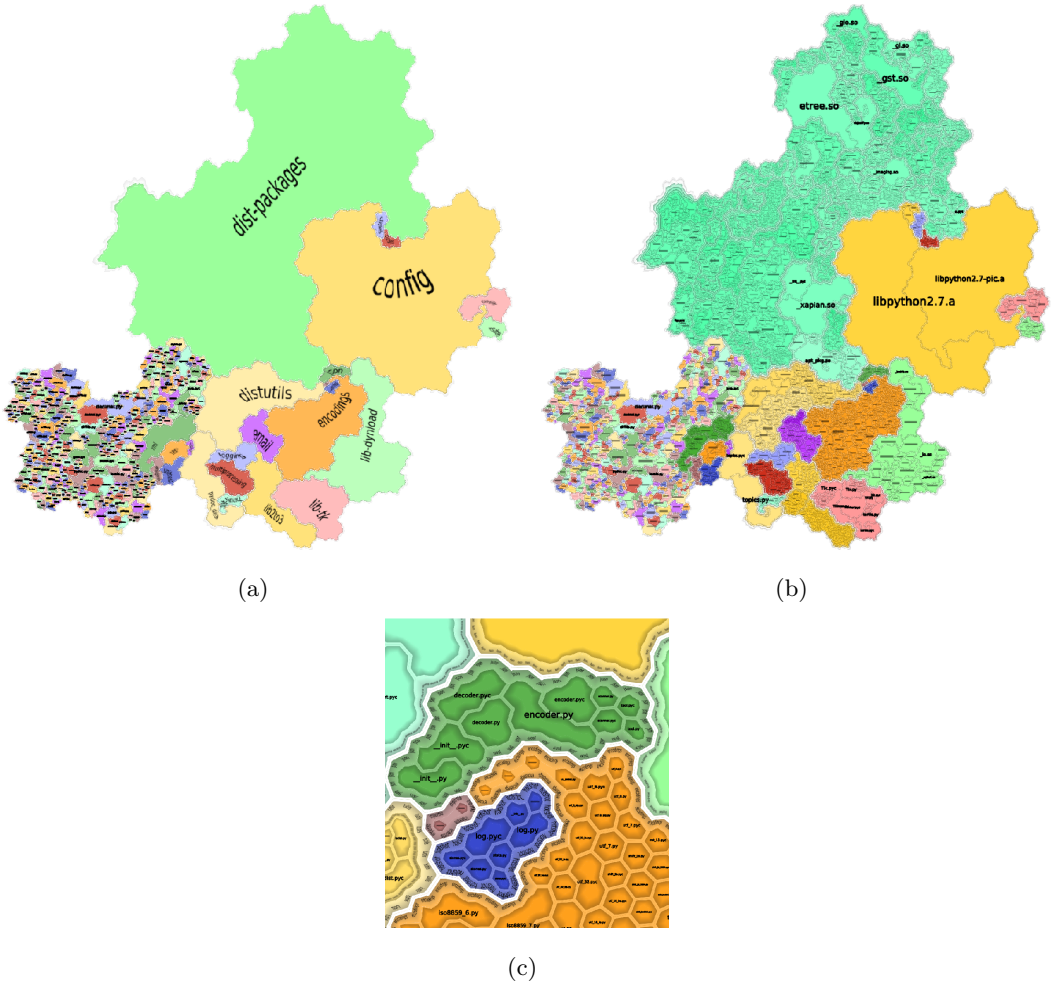
sous-section précédente. Elles nous ont permis de savoir que les *geographical treemaps* rivalisent en qualité avec des algorithmes très utilisés pour la visualisation d'arbres.

L'utilisation de notre technique sur des arbres dynamiques nous montre que même si la somme des déplacements est plus élevée qu'avec un *Sunburst*, la carte mentale est préservée car ses déplacements pris individuellement s'effectuent sur une courte distance. La figure 2.43 illustre ce phénomène, *e.g.* il est facile de suivre l'évolution de la région *Commerce and Housing Credit*. On voit aussi clairement que la quantité d'argent qui lui est attribuée a augmenté entre 2007 et 2009 puis a diminué en 2010.

Postérieurement, nous voudrions mener une étude sur des utilisateurs afin de vérifier la validité de l'approche géographique. Contrairement aux autres méthodes, notre carte est composée de régions aux formes complexes et nous pensons que cette propriété pourrait aider l'utilisateur à garder en mémoire une vue d'ensemble des données. Cette hypothèse nous est suggérée par l'habitude que nous avons de manipuler des cartes géographiques. L'exemple de la figure 2.46 illustre ce phénomène. Les personnes ayant vu la carte ont tout de suite assimilé la région jaune (*config*) à la Chine.

2.3 Conclusion

Les deux exemples de visualisation que nous avons décrits dans ce chapitre peuvent être appliqués à des données « modélisables » sous forme de graphes. Dans le premier cas, les graphes que nous manipulons ont certaines propriétés connues sous le nom de « petit-monde » et « sans échelle », dans le second cas les graphes sont des arbres. Il existe aussi un modèle mathématique généralisant cette notion de graphe : les hypergraphes. Ce type d'outil a été très peu étudié en visualisation de l'information. Il peut être cependant très utile pour modéliser certains types de données. C'est pourquoi nous allons lui dédier le chapitre suivant.



(a) (b)

(c)

FIGURE 2.46: Visualisation du contenu de la bibliothèque standart *Python* de *Linux* : (a) Seul le premier niveau de l'arbre est affiché. (b) Les deux premiers niveaux de l'arbre sont affichés. (c) Zoom sur l'une des zones du graphe.

Chapitre 3
Visualisation d'hypergraphes

La notion d'**hypergraphe** [20, 21, 207] permet de généraliser celle de graphe de façon à ce qu'une « arête » puisse être un sous-ensemble quelconque (*i.e.* non limité à deux éléments) de l'ensemble des sommets. Plus formellement, un hypergraphe H est défini comme une paire (V, A) où V est un ensemble de sommets et A est un ensemble de sous-ensembles de V appelés **hyperarêtes**. La figure 3.1 représente l'hypergraphe $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$.

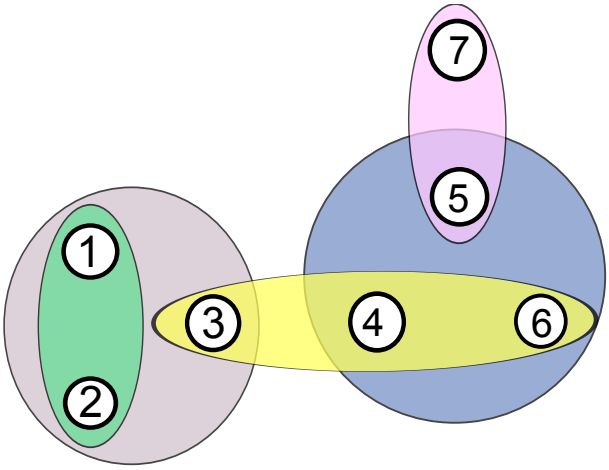


FIGURE 3.1: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$: les hyperarêtes sont représentées grâce à des cercles de couleur.

Un hypergraphe est un outil puissant de modélisation de certains types de données abstraites. Reprenons pour illustrer ce concept l'exemple du réseau social *viadeo* évoqué au début du deuxième chapitre. Nous avons dit qu'un individu pouvait appartenir à des communautés (ou groupes de discussion). Nous pouvons modéliser de telles informations à l'aide d'un hypergraphe où les sommets représentent les personnes et les hyperarêtes ces communautés. Imaginons qu'une telle modélisation nous donne l'hypergraphe représenté sur la figure 3.2. Il est alors possible de déterminer rapidement et visuellement quels sont les individus couvrant un grand nombre de communautés ce qui constitue un indice sur leur influence potentielle dans le réseau. Par exemple, sur la figure 3.2, nous pouvons

observer que les deux personnes représentées par les sommets entourés d'un cercle jaune couvrent à elles seules l'ensemble des communautés.

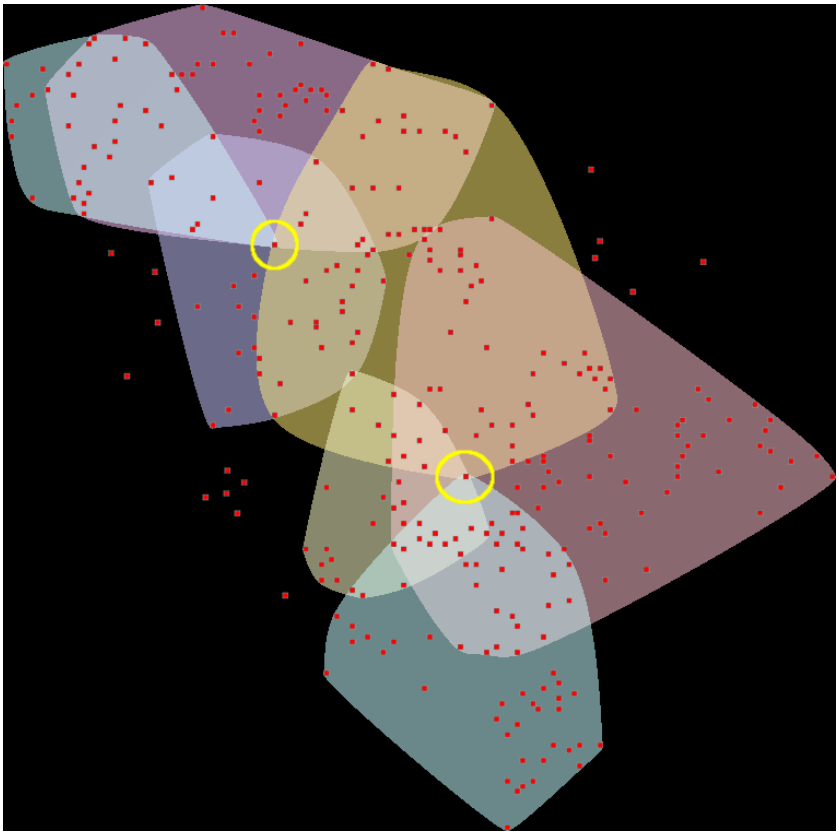


FIGURE 3.2: Exemple de représentation d'hypergraphe.

Intéressons nous donc maintenant aux solutions qui s'offrent à nous pour représenter des hypergraphes. En simplifiant, il existe deux grands types de représentation d'hypergraphes. La première, connue sous le nom de ***edge-standart*** consiste à dessiner les hyperarêtes comme des arbres où les feuilles représentent les sommets qu'elles contiennent. La figure 3.3.a montre le dessin d'un hypergraphe $H = (V, A)$ grâce à l'une de ces méthodes : on dessine ici un graphe biparti $B_H = (V, A, E)$ avec V et A les deux ensembles de sommets et E l'ensemble des arêtes telles que $(v, h) \in E$ si et seulement si $v \in V$, $h \in H$ et $v \in h$. Un autre exemple consiste à dessiner un *diagramme de Hasse*. C'est un graphe $G_H = (\{V\} \cup A \cup \{\{v\}; v \in V\}, E)$ orienté sans cycle dans lequel il existe une arête (h_1, h_2) si et seulement si $h_2 \subsetneq h_1$ et il n'existe pas d'hyperarête h telle que $h_2 \subsetneq h \subsetneq h_1$ (voir figure 3.3.b). Cette approche a fait l'objet de nombreux travaux, certains particulièrement poussés dans le domaine de la visualisation de l'information (voir [51] par exemple). Plus globalement, il existe de nombreux travaux utilisant une approche *edge-standart*. Cependant, comme ce n'est pas celle qui sera étudiée dans le reste de ce chapitre, nous convions le lecteur intéressé à se reporter aux articles [45, 65, 134, 154].

Le second type de méthodes permettant de représenter des hypergraphes est connu sous le nom de ***subset standart*** et consiste à dessiner un **diagramme d'Euler** [67] **ou**

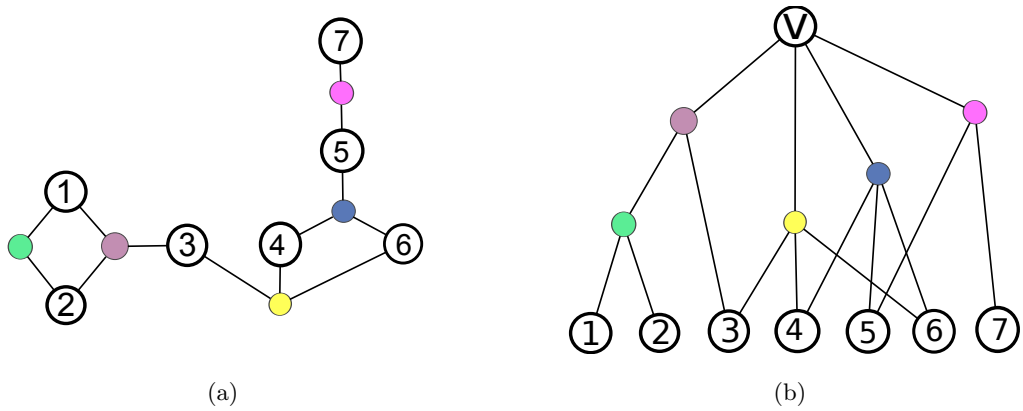


FIGURE 3.3: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Graphe biparti : les hyperarêtes sont représentées par les sommets de couleur et elles sont reliées par une arête aux représentations des sommets qu'elles contiennent. (b) Diagramme de Hasse : les hyperarêtes, représentées par les sommets de couleur, sont organisées en hiérarchie de façon à ce qu'une hyperarête contienne tous les sommets de ses hyperarêtes filles.

de Venn [188] [1]. Les figures 3.1 et 3.2 en montrent des exemples. Obtenir un tel dessin n'est cependant pas aisé. En effet, la plupart des hypergraphes ne peuvent pas être dessinés de façon à ce que chaque paire d'hyperarêtes dont l'intersection est vide ne se chevauche pas [2]. Simonetto *et al.* ont d'ailleurs proposé une méthode permettant d'afficher n'importe quel hypergraphe en créant plusieurs régions pour certaines hyperarêtes [163, 164, 165]. Henry Riche et Dwyer ont proposé quant à eux deux méthodes afin de pallier ce problème [147], la première consistant à représenter certaines hyperarêtes par des enveloppes et les autres par des liens (ComED), la seconde consistant à dupliquer les sommets apparaissant dans plusieurs hyperarêtes en ajoutant des liens pour les identifier (DupED). Nous ne reviendrons pas ici sur ce type de techniques. Au contraire, nous allons maintenant nous concentrer sur les méthodes permettant de découvrir les classes d'hypergraphes pouvant être représentés de façon **planaire**, *i.e.* sans que les régions se chevauchent si les hyperarêtes correspondantes ne partagent pas de sommets. Ces méthodes sont basées sur des graphes associés aux hypergraphes que nous allons maintenant décrire.

1. Ces types de diagrammes ont été introduits par Euler puis améliorés par Venn afin de modéliser les relations logiques d'ensembles d'éléments. Nous ne reviendrons pas ici sur une description plus détaillée de ces objets et du contexte de leur apparition car cela n'a pas un grand intérêt pour nos travaux. Pour une description plus complète, le lecteur pourra se référer à l'article [148].

2. On se doit de remarquer ici que d'autres contraintes peuvent aussi être ajoutées au dessin. Par exemple, lorsque l'hypergraphe est issu d'une fragmentation « chevauchante » de graphe (*i.e.* les communautés correspondent aux hyperarête et un sommet peut appartenir à plusieurs de ces communautés), le dessin est contraint par les arêtes du graphe [55, 133]. Bien que ce problème ne soit pas étudié dans la suite de ce document, nous pensons qu'il est assez proche du nôtre pour le mentionner ici. Un autre problème proche consiste à représenter un hypergraphe dont les sommets sont déjà positionnés. Les techniques mises en œuvre dans ce cas sont radicalement différentes. Des solutions ont été proposées par Collins *et al.* [49] ou Byelas et Telea [38, 39].

La première de ces méthodes est basée sur les graphes bipartis construits à partir des hypergraphes comme nous l'avons décrit précédemment (voir l'exemple du paragraphe concernant les représentations de type *edge-standart* ainsi que la figure 3.3). Selon cette approche, un hypergraphe H est **Zykov-planaire** [207] si et seulement si le graphe biparti B_H qui lui est associé est **planaire** [192] (*i.e.* il existe une représentation dans le plan de ses sommets sous forme de points et de ses arêtes sous forme de lignes telle que ces lignes ne se coupent pas). Il est ainsi possible de tester en temps linéaire si un hypergraphe est Zykov-planaire [96].

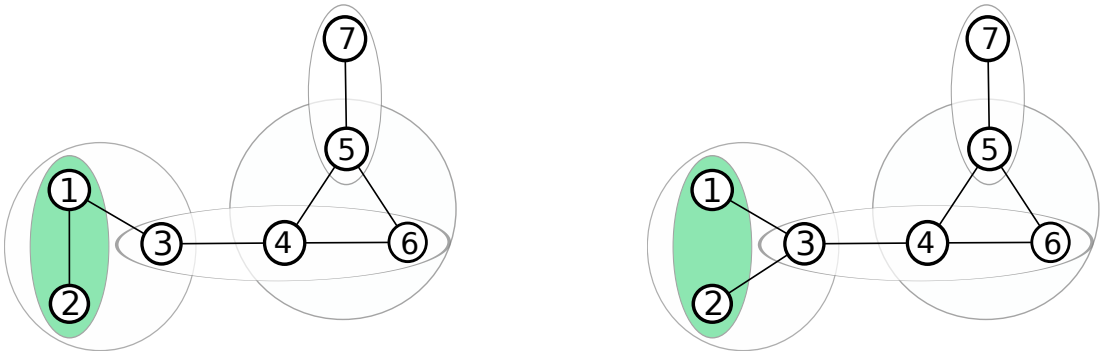
D'autres travaux sur les digrammes d'Euler associés à une définition de ce que pourrait en être une représentation correcte sont résumés dans l'article [70]. Cette définition est basée sur un graphe dit « **superdual** » (ou graphe « **dual combinatoire** »). Ce graphe S_H est construit à partir d'un hypergraphe $H = (V, A)$ de la manière suivante. Tout d'abord, l'ensemble de ses sommets V_S correspond à l'ensemble V auquel un sommet virtuel a été ajouté et où certains sommets ont été supprimés de façon à ce que chaque sommet restant soit l'unique intersection des hyperarêtes auxquelles il appartient. Ensuite, une arête est ajoutée entre u et v si la différence symétrique entre l'ensemble des hyperarêtes contenant u et l'ensemble des hyperarêtes contenant v contient exactement une hyperarête h. L'arête (u, v) est alors étiquetée h. Grâce au superdual, Flower *et al.* [70] proposent des critères pour sélectionner les diagrammes pouvant être bien formés. Dans la même veine, Verroust et Viaud [190] ont étudié les hypergraphes constitués par au plus 8 hyperarêtes. La complexité de la génération des diagrammes d'Euler est étudiée dans [159].

Enfin, d'autres méthodes sont basées sur des graphes dit « supports » [183, 109] (ou graphes « hôtes » [118]). Avant de définir plus précisément cette notion nous devons d'abord présenter le concept de **graphe induit**. Soit $G = (V, E)$ un graphe quelconque et h un sous ensemble de ses sommets ($h \subset V$). Le graphe induit par l'ensemble h, noté $G[h]$, est un graphe dont l'ensemble des sommets est h et l'ensemble des arêtes correspond aux arêtes (u, v) de E telles que $u \in h$ et $v \in h$. Un graphe $G = (V, E)$ est un **support** de l'hypergraphe $H = (V, A)$ si et seulement si les sous-graphes de G induits par chacune des hyperarêtes de H sont connexes, *i.e.* $\forall h \in A$, $G[h]$ est connexe. La figure 3.4 montre un exemple de graphe support.

Selon l'approche basée sur les graphes supports, un hypergraphe est (sommet-)planaire [106] si il possède un support planaire [3]. La « planarité » des hypergraphes ainsi définie peut être vue comme une généralisation de la notion d'hypergraphes Zykov-planaires [183] et d'hypergraphes dont les diagrammes d'Euler peuvent être bien formés [70]. C'est pourquoi nous allons nous y intéresser maintenant de plus près.

Nous allons tout d'abord étudier la « planarité » des hypergraphes basée sur les graphes supports en résumant les résultats déjà publiés et en en apportant des nouveaux (section 3.1). Ensuite, nous introduirons un nouveau type de supports prometteur pour la visualisation des hypergraphes (section 3.2).

3. On peut remarquer ici que les graphes partiellement connexes de Chow [48] sont des supports planaires de versions duales d'hypergraphes.



(a) (b)

FIGURE 3.4: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Le graphe G représenté par les arêtes noires est un support de H : les sous-graphes induits par les hyperarêtes sont connexes. (b) Le graphe G représenté par les arêtes noires n'est pas un support de H car $G[\{1, 2\}]$ n'est pas connexe.

3.1 « Planarité » des hypergraphes basée sur les graphes supports

Comme nous l'avons vu ci-dessus, il est possible de construire un diagramme d'Euler satisfaisant (*i.e.* tel que les régions correspondant à des hyperarêtes ne se chevauchent pas si ces hyperarêtes ne partagent pas de sommets) pour représenter un hypergraphe si et seulement si celui-ci possède un support planaire. On pourrait donc ainsi caractériser les hypergraphes pouvant être représentés par ce type de diagrammes. Cependant, Johnson et Pollack [106] ont prouvé que le problème de décision consistant à déterminer si oui ou non un hypergraphe possède un support planaire est NP-complet.

Il serait néanmoins regrettable de se limiter à ce résultat. En effet, il existe des classe des graphes planaires, comme les arbres que nous avons déjà mentionnés dans la section précédente. Or, Tarjan et Yannakakis [173] ont montré que l'on peut décider en temps polynomial si un hypergraphe a un support qui est un arbre. Nous allons donc maintenant nous intéresser à certaines de ces classes afin de déterminer s'il existe d'autres catégories d'hypergraphes pour lesquels on peut trouver un support planaire. Commençons par les définir.

Un **cycle** est un graphe $G = (V, E)$ tel que $V = \{1, 2, ..., n\}$ et $E = \{(n, 1), (1, 2)(2, 3), ... , (n - 1, n)\}$ (voir figure 3.5.a). Si l'on supprime l'arête $(n, 1)$ d'un cycle, on obtient un **chemin** [4] (voir figure 3.5.b). Contrairement à la section précédente, un **arbre** est défini ici comme un graphe non orienté, acyclique et connexe (voir la note de bas de page 15 et la figure 3.5.c). Le dessin d'un graphe sur le plan permet de construire un certain nombre

4. Dans le chapitre précédent, la notion de chemin avait déjà été employée pour caractériser une suite de sommets d'un graphe G dans laquelle chaque paire de sommets consécutifs était une arête. Si l'on considère l'ensemble de ces sommets et de ces arêtes comme un sous-graphe de G, alors ce sous-graphe est un chemin tel que nous venons de le définir.

de régions, délimitées par les lignes correspondant aux arêtes. Ces régions sont appelées **faces**. La **face extérieure** est celle qui se trouve « à l'extérieur du graphe », donc qui n'est pas entièrement délimitée par les arêtes de celui-ci et dont l'aire est infinie. Un graphe est dit **planaire extérieur** si l'on peut le dessiner dans le plan de façon planaire et de façon à ce que tous ses sommets soient adjacents à sa face extérieure (voir figures 3.5.e et 3.6.a). Enfin, un **cactus** est un graphe planaire extérieur dont toutes les arêtes sont adjacentes à la face extérieure (voir figure 3.5.d).

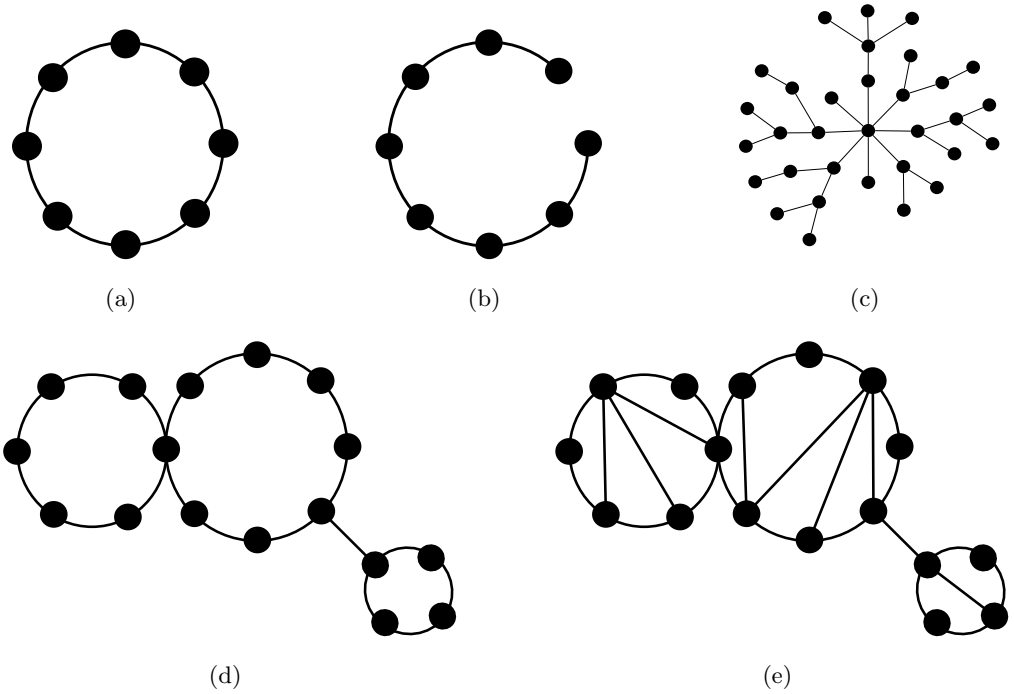


FIGURE 3.5: (a) Cycle. (b) Chemin. (c) Arbre. (d) Cactus. (e) Planaire extérieur.

Le concept de planaire extérieur a aussi été généralisé. On dit qu'un graphe est **2-planaire extérieur** s'il existe un dessin dans le plan tel que la suppression des sommets apparaissant sur la face extérieure nous donne un graphe planaire extérieur (voir figure 3.6.b). De même, un graphe est **3-planaire extérieur** s'il existe un dessin dans le plan tel que la suppression des sommets de sa face extérieure nous donne un graphe 2-planaire extérieur (voir figure 3.6.c).

Avant d'aller plus loin, prenons un peu de temps pour étudier les rapports d'inclusion entre toutes ces classes. En effet, on peut par exemple remarquer qu'un chemin est un arbre particulier. La classe des chemins est donc incluse dans celle des arbres (*i.e.* tout chemin est un arbre, mais tout arbre n'est pas un chemin). Si l'on fait le bilan, on trouve les inclusions suivantes (les classes sont notées $\mathcal{C}(nomdesgraphes)$) :

$$\mathcal{C}(chemins) \subset \mathcal{C}(arbres)$$

$$\mathcal{C}(arbres), \mathcal{C}(cycles) \subset \mathcal{C}(cactus)$$

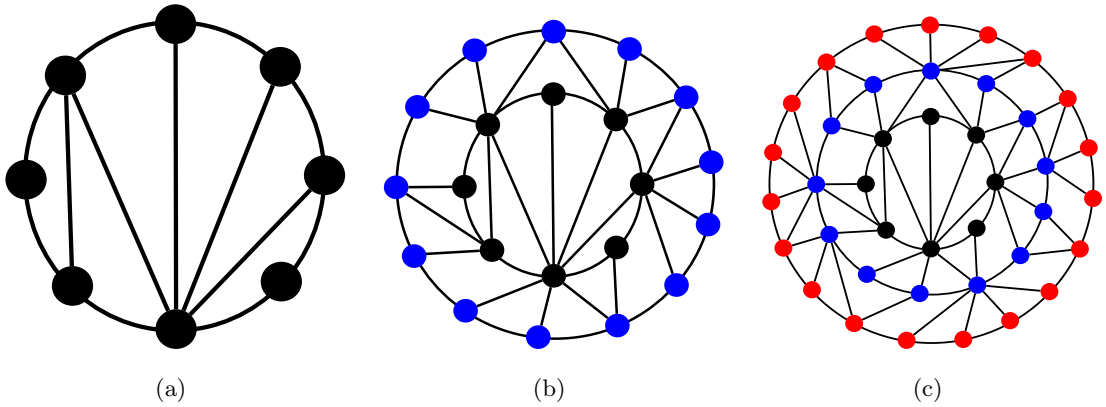


FIGURE 3.6: (a) Planaire extérieur. (b) 2-planaire extérieur. (c) 3-planaire extérieur.

$$\mathcal{C}(cactus) \subset \mathcal{C}(planaires\ extérieurs)$$

$$\mathcal{C}(planaires\ extérieurs) \subset \mathcal{C}(2-planaires\ extérieurs)$$

$$\mathcal{C}(2-planaires\ extérieurs) \subset \mathcal{C}(3-planaires\ extérieurs)$$

$$\mathcal{C}(3-planaires\ extérieurs) \subset \mathcal{C}(planaires)$$

Par transitivité, on sait donc que si l'on arrive à trouver un algorithme polynomial afin de décider si oui ou non un hypergraphe possède un support planaire extérieur, alors on saura trouver un support planaire extérieur pour tous les hypergraphes des sous-classes de $\mathcal{C}(planaires\ extérieurs)$ (en revanche, on ne pourra pas forcément trouver un cactus, un cycle ou un chemin s'il en existe un).

Nous avons listé ci-dessous les complexités des problèmes de décision concernant la nature du support d'un hypergraphe qui ont précédé nos travaux :

- planaire : NP-complet [106]
- 3-planaire extérieur : NP-complet [37]
- 2-planaire extérieur : NP-complet [36]
- planaire extérieur : ?
- cactus : ?
- arbre : temps linéaire [173] [5]
- chemin : temps linéaire [37]
- cycle : temps linéaire [37]

Les éléments notés en rouge correspondent à des problèmes ouverts lors du début de nos travaux.

Dans la section 3.1.2 nous verrons que l'on peut décider en temps polynomial si un hypergraphe possède un support qui est un cactus. Nous étudierons ensuite dans la section 3.1.3 une classe particulière d'hypergraphes : les hypergraphes à intersections et différences fermées. Nous verrons qu'il est possible de décider s'ils possèdent des supports planaires ou non. La section 3.1.4 résumera les résultats obtenus. Mais avant tout cela, nous allons

5. Précisons ici qu'il est aussi possible de trouver en temps linéaire des arbres supports dont le degré est borné [37] ou des arbres supports minimaux lorsque les arêtes possibles sont valuées [116].

définir le concept de composante bi-connexe pour les hypergraphes, ce qui nous permettra de mener à bien nos recherches (section 3.1.1).

Les travaux que nous allons présenter ont été réalisés avec Ulrik Brandes, Sabine Cornelsen et Barbara Pampel. Ils ont fait l'objet d'une publication [30]. Dans la suite de cet exposé, nous allons considérer un hypergraphe $H = (V, A)$ avec $n = |V|$ sommets, $m = |A|$ hyperarêtes et $N = \sum_{h \in A} |h|$ la somme des tailles des hyperarêtes. La **taille de l'hypergraphe** est alors $N + n + m$.

3.1.1 Composantes bi-connexes

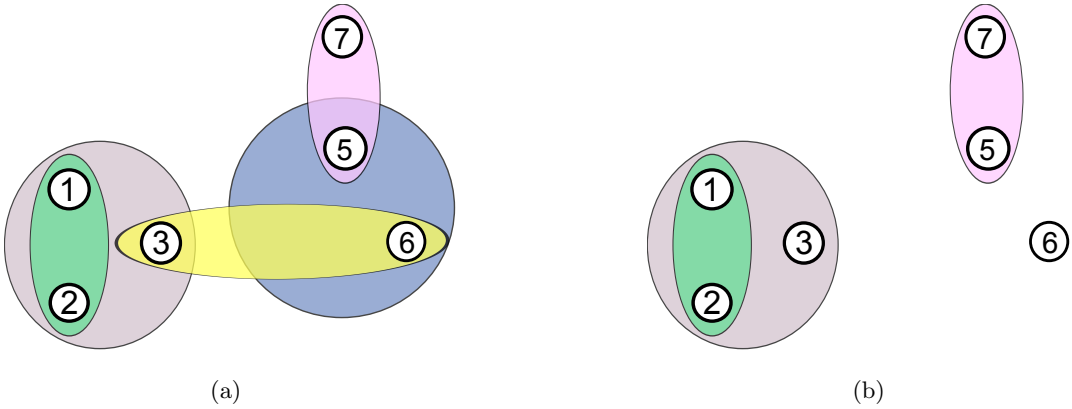
Considérons maintenant un graphe $G = (V, E)$. Ce graphe est dit **connexe** s'il existe un chemin entre chaque paire de ses sommets. Si ce n'est pas le cas, chaque sous-graphe maximal connexe de G est une **composante connexe** de G, *i.e.* il existe un chemin entre chaque paire de sommets d'une même composante connexe mais il n'existe aucun chemin reliant un sommet d'une de ces composantes avec un sommet d'une autre de ces composantes. G est dit **bi-connexe** (ou **2-connexe**) si le graphe induit par l'ensemble de ses sommets V moins un sommet quelconque est connexe : $\forall v \in V, G[V \setminus \{v\}]$ est connexe. Un sommet v dont la suppression déconnecte le graphe est appelé **sommet articulation**. Soit S_A l'ensemble des sommets d'articulation de G. Les composantes connexes de $G[V \setminus S_A]$ sont appelées **composantes bi-connexes** de G.

Dans cette section, nous allons montrer comment on peut étendre le concept de bi-connexité aux hypergraphes. Ceci nous permettra de décomposer récursivement un hypergraphe en composantes bi-connexes. Nous appellerons blocs les composantes connexes du dernier niveau de décomposition. Ceci sera effectué de façon à ce qu'il existe un support tel que les blocs de l'hypergraphe correspondent aux composantes bi-connexes de ce support. Commençons par donner quelques définitions utiles.

Soit un hypergraphe $H = (V, A)$ et un sous-ensemble de ses sommets $V' \subset V$. L'hypergraphe **induit** par V' est $H[V'] = (V', A[V'])$ avec $A[V'] = \{h \cap V'; h \in A\} \setminus \{\emptyset, \{v\}; v \in V\}$. $A[V']$ contient les hyperarêtes de H auxquelles on a enlevé les sommets n'apparaissant pas dans V' (si une hyperarête ainsi construite est vide, ou ne contient qu'un seul sommet, elle est supprimée). La figure 3.7.a montre un exemple d'hypergraphe induit. L'hypergraphe **restreint** par V' est quant à lui défini par $H|V' = (V', A|V')$ avec $A|V' = \{h \in A; h \subseteq V'\}$. La figure 3.7.b en donne un exemple. On peut déjà remarquer que si h est planaire, $H[V']$ ne l'est pas forcément alors que $H|V'$ l'est forcément.

La séquence $p : v_0, h_1, v_1, \ldots, h_k, v_k$ est un v_0v_k**-chemin** de H si $h_1, \ldots, h_k \in A$, $v_0 \in h_1, v_k \in h_k$, et $v_i \in h_i \cap h_{i+1}, i = 1, \ldots k-1$. Les sommets v_0 et v_k sont les **sommets terminaux** de p. Deux sommets v, w d'un hypergraphe $H = (V, A)$ sont **connexes** si il existe un vw-chemin dans H. La connexité est une relation d'équivalence sur l'ensemble des sommets d'un hypergraphe et les hypergraphes induits par les classes d'équivalence sont appelés des **composantes connexes** [207].

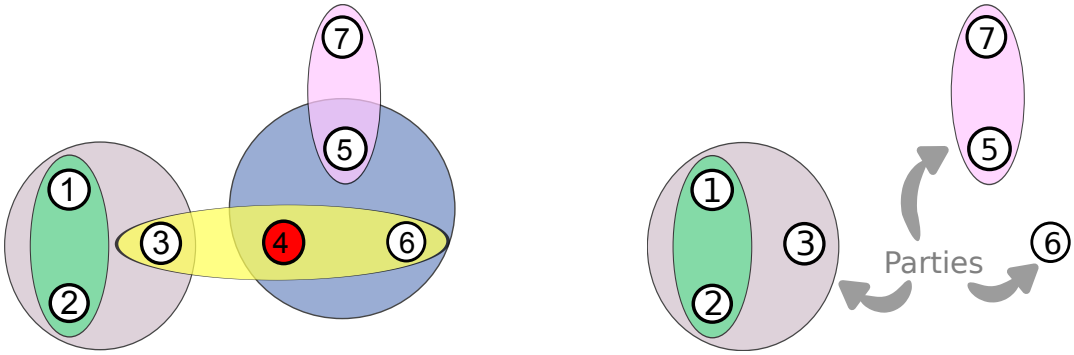
Soit un sommet $v \in V$. Les composantes connexes de $H|(V \setminus \{v\})$ sont les **parties** de v et v est un **sommet d'articulation** de H si v a plus d'une partie. L'image 3.8 montre un exemple de parties d'un sommet. On peut remarque ici que v est un sommet d'articulation de H si et seulement si il existe un support de H dans lequel v est un sommet



(a) (b)

FIGURE 3.7: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Hypergraphe induit $H[V \setminus \{4\}]$. (b) Hypergraphe restreint $H|(V \setminus \{4\})$.

d'articulation tel que nous l'avons défini précédemment sur les graphes. La figure 3.4.a montre que le sommet 4 du graphe support est un sommet d'articulation de ce graphe.



(a) (b)

FIGURE 3.8: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. (a) Le sommet étiqueté 4 est un sommet d'articulation. (b) Les composantes connexes de $H|(V \setminus \{4\})$ sont les parties du sommet 4.

Une **décomposition en blocs** d'un hypergraphe $H = (V, A)$ est définie récursivement. H est un bloc si et seulement si il ne contient pas de sommet d'articulation. Si H n'est pas connexe, alors les blocs de H sont les blocs de ses composantes connexes. Si H est connexe et contient un sommet d'articulation v avec $W_1, \ldots, W_k$ les parties de v, alors les blocs de H sont les blocs de $H[W_1 \cup \{v\}], \ldots, H[W_k \cup \{v\}]$. On peut remarquer ici qu'une telle décomposition n'est pas unique pour un hypergraphe. En effet, un sommet d'articulation est sélectionné à chaque étape de façon arbitraire parmi la liste des sommets d'articulation. La figure 3.9 représente la décomposition d'un hypergraphe.

On se doit de remarquer que la définition des sommets d'articulation et des blocs que nous venons de proposer est proche mais différente de celle donnée dans l'article [19]. Il

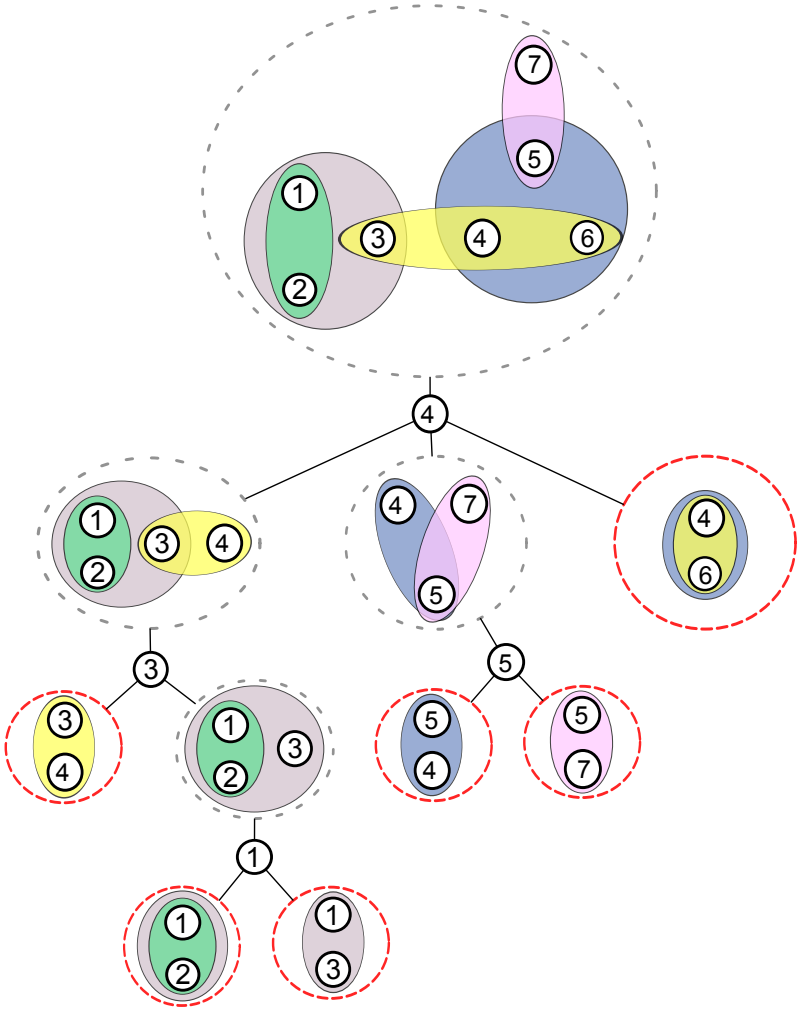


FIGURE 3.9: Arbre blocs-articulations, *i.e.* décomposition possible de l'hypergraphe $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6\}, \{5, 7\}\}$. Les blocs sont entourés de cercles en pointillés rouges.

est aussi important de noter que la taille des blocs est au plus trois fois celle de la taille de l'hypergraphe.

Nous allons maintenant montrer que la décomposition en bloc d'un hypergraphe peut être effectuée en temps polynomial. Pour cela, nous allons utiliser le graphe biparti B_H qui correspond à un hypergraphe H (nous avons déjà défini B_H en introduction de ce chapitre). Il nous est particulièrement utile car ses composantes connexes correspondent aux composantes connexes de l'hypergraphe qui lui est associé. De plus, un sommet v est un sommet d'articulation de H si $B[V \setminus \{v\} \cup A \setminus \{h \in A; v \in h\}]$ contient plus d'une composante connexe (nous appellerons aussi parties de v ces composantes car elles correspondent aux parties de v de l'hypergraphe associé). Les blocs de B_H sont définis comme les graphes bipartis représentant les blocs de H. Ils peuvent être construits en regardant, pour tous les sommets de V, si les sous-graphes $B[V \setminus \{v\} \cup A \setminus \{h \in A; v \in h\}]$ contiennent plusieurs composantes connexes, *i.e.* en déterminant n fois les composantes

connexes d'un sous-graphe de B_H.

Lemme 3.1 *Les blocs d'un hypergraphe H peuvent être déterminés en temps polynomial : $O(nN + n + m)$.*

Preuve : Si H n'est pas connexe, alors les composantes connexes de B_H peuvent être trouvées en $O(N+n+m)$. L'algorithme ci-dessous, en $O(nN)$, peut ensuite être déroulé pour chacune de ces composantes. Considérons donc un hypergraphe H connexe et son graphe biparti B_H. Soit $v_1, \ldots, v_n$ un ordre quelconque des sommets de H. L'algorithme Blockfinder(B, k) prend en paramètres un sous-graphe B de B_H et un indice $k = 0, \ldots, n$ tel que $v_1, \ldots, v_k$ ne sont pas des sommets d'articulation de B. Il retourne un lien vers la liste des blocs de B.

Blockfinder(B, k)
1 - S'il n'existe pas de $k' > k$ tel que $v_{k'}$ appartienne à B, retourner B.
2 - Soit $k' > k$ l'indice minimal tel que $v_{k'}$ appartient à B :
3 - Supprimer $v_{k'}$ et ses voisins $h_1, \ldots, h_j$ de B et calculer ses composantes connexes $B_1, \ldots, B_\ell$.
4 - Pour $i = 1, \ldots, \ell$, ajouter à B_i le sommet $v_{k'}$ ainsi que des arêtes vers les sommets $h_1, \ldots, h_j$ de A contenant $v_{k'}$ dans H.
5 - Retourner Blockfinder(B_1, k'), ..., Blockfinder(B_ℓ, k').

Ainsi Blockfinder(B_H, 0) trouve une partition de H en blocs représentés par des graphes bipartis. En effet, imaginons qu'il retourne un sous-graphe B_i de B_H qui contient un sommet d'articulation $v_{k'}$. Soient P_1 et P_2 les deux parties de $v_{k'}$ dans B_i. Soit B le sous-graphe de B_H tel que l'indice k' a été traité (sans être repéré comme point d'articulation) par l'appel Blockfinder(B,k). Puisqu'à la fin P_1 et P_2 appartiennent à B_i, il existe au moins un chemin dans B reliant P_1 et P_2 qui ne contient pas $v_{k'}$. Soit p le chemin le plus court parmi ces chemins. Alors p est aussi un chemin de B_i et $v_{k'}$ n'est pas un sommet d'articulation. En effet, si p n'était pas dans B_i, il ne serait pas le plus court chemin reliant P_1 et P_2. Regardons cela de plus près. Soit $p : w_0, h_1, \ldots, h_\ell, w_\ell$. On considère w_j comme le premier sommet de p qui n'est pas dans B_i. Soit $j' > j$ le plus petit indice tel que $w_{j'}$ est dans B_i. Il existe alors un sommet d'articulation $v_\ell, \ell > k'$ de B_i avec $v_\ell \in h_j \cap h_{j'}$. Ainsi $w_0, h_1, \ldots, w_{j-1}, h_j, v_\ell, h_{j'}, w_{j'}, \ldots, h_\ell, w_\ell$ est un chemin plus court que p reliant P_1 et P_2. □

La décomposition d'un hypergraphe en blocs induit un **arbre blocs-articulations** (tout comme la décomposition d'un graphe). Soit T cet arbre. T est aussi un graphe biparti composé de sous-hypergraphes de H et des sommets d'articulation identifiés par l'algorithme. Ses feuilles sont les blocs de H. La figure 3.9 en montre un exemple.

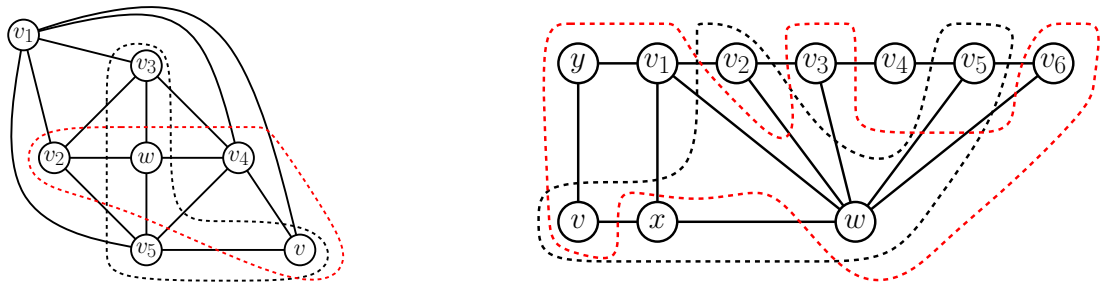
Lemme 3.2 *Une hypergraphe possède un support planaire (extérieur) si tous ses blocs ont un support planaire (extérieur).*

Preuve : Soit $B_1, \ldots, B_k$ les blocs de l'hypergraphe $H = (V, A)$. Soit $G_i = (V_i, E_i)$ un support de B_i pour $i = 1, \ldots, k$. Alors $G = (V, E_1 \cup \cdots \cup E_k)$ est un support de H et $G_1, \ldots, G_k$ sont des composantes bi-connexes de G. En procédant des feuilles à la racine

de l'arbre blocs-articulations, on peut donc positionner les sommets du support de chaque bloc de façon à ce que le sommet d'articulation avec son parent soit sur la face extérieure. Ainsi, si tous les G_i possèdent une représentation planaire (extérieur), alors G en possède une aussi. □

L'inverse du lemme 3.2 n'est pas vrai : il existe des hypergraphes planaires dont les blocs ne sont pas planaires. Prenons par exemple un hypergraphe H composé des hyperarêtes $\{v, v_1\}$, $\{v, v_4\}$, $\{v, v_5\}$, $\{v_2, v_4, v, w\}$, $\{v_3, v_5, v, w\}$, $\{v_1, v_2\}$, $\{v_1, v_3\}$, $\{v_1, v_4\}$, $\{v_1, v_5\}$, $\{v_2, v_3\}$, $\{v_3, v_4\}$, $\{v_4, v_5\}$, $\{v_2, v_5\}$. H est planaire, v est un sommet d'articulation de H et $H[\{v_1, v_2, v_3, v_4, v_5, v\}]$ est un bloc de H non planaire (voir figure 3.10.a).

De même, il existe des hypergraphes planaires extérieurs dont les blocs ne sont pas planaires extérieurs. Considérons par exemple un hypergraphe H composé des hyperarêtes $\{v_1, v_2\}$, $\{v_2, v_3\}$, $\{v_3, v_4\}$, $\{v_4, v_5\}$, $\{v_5, v_6\}$, $\{v, y\}$, $\{y, v_1\}$, $\{v, x\}$, $\{x, v_1\}$, $\{v, x, w, v_2, v_5\}$, et $\{v, y, v_1, w, v_3, v_6\}$ avec le sommet d'articulation v (voir figure 3.10.b).



(a) (b)

FIGURE 3.10: Exemples : les lignes continues noires représentent les supports, les lignes en pointillé représentent les hyperarêtes contenant plus de deux sommets. (a) Support planaire dont les blocs ne sont pas planaires. (b) Support planaire extérieur dont les blocs ne sont pas planaires.

Nous allons maintenant utiliser la décomposition décrite ci-dessus ainsi que ces premiers résultats afin d'étudier la classe des hypergraphes possédant un support qui est un cactus.

3.1.2 Supports cactus

Comme nous l'avons précisé en introduction, un cactus est un graphe pouvant être représenté de façon à ce que tous ses sommets et toutes ses arêtes soient adjacents à sa face extérieure. La classe des cactus est donc strictement incluse dans la classe des graphes planaires extérieurs. Dans cette partie, nous allons montrer que l'on peut décider en temps polynomial si un hypergraphe a un support qui est un cactus. Dans ce cas, nous proposerons aussi une technique pour déterminer ce support. Ce résultat est primordial pour le dessin d'hypergraphe. En effet, il permet de caractériser une classe d'hypergraphes pouvant être dessinés convenablement grâce à un diagramme d'Euler.

Lemme 3.3 *Un hypergraphe possède un support qui est un cactus si et seulement si chacun de ses blocs possède un support qui est un cycle ou une arête.*

Preuve : La preuve de la partie $\Leftarrow$ est analogue à celle du lemme 3.2.

Pour la partie $\Rightarrow$, considérons un hypergraphe $H = (V, A)$ et un cactus $G = (V, E)$ support de H. Soit v un sommet d'articulation de H et W une partie de v. Nous allons montrer que $H[W \cup \{v\}]$ possède aussi un support qui est un cactus (rappelez-vous que les blocs sont obtenus en décomposant l'hypergraphe de cette manière).

Nous disons que $u \in W$ est **proche** de v si et seulement si il existe un chemin dans G de v à u ne contenant aucune arête de $G[W]$ (*e.g.* u_1 et u_2 sont proches de v dans la figure 3.11.a). Remarquons ici que $G[W]$ est un sous-graphe connexe de cactus ne contenant pas v. Il y a donc au plus deux sommets de W qui sont proches de v. Un support cactus $G_W = (V_W, E_W)$ de $H[W \cup \{v\}]$ peut être construit de la manière suivante (voir les figures 3.11.a et 3.11.b pour une illustration) :
1 - $G_W \leftarrow G[W \cup \{v\}]$
2 - Pour chaque $u \in W$ qui est proche de v, ajouter $\{u, v\}$ à E_W.
3 - Pour chaque cycle C de G. Si $E[W] \cap C \neq \emptyset$ et $C \not\subseteq E[W]$, alors $G[W \cap C]$ est un chemin p_C. Si les **sommets terminaux** (*i.e.* les deux sommets de degré 1 d'un chemin) x et y de p_C ne sont pas proches de v, ajouter $\{x, y\}$ à E_W. $\square$

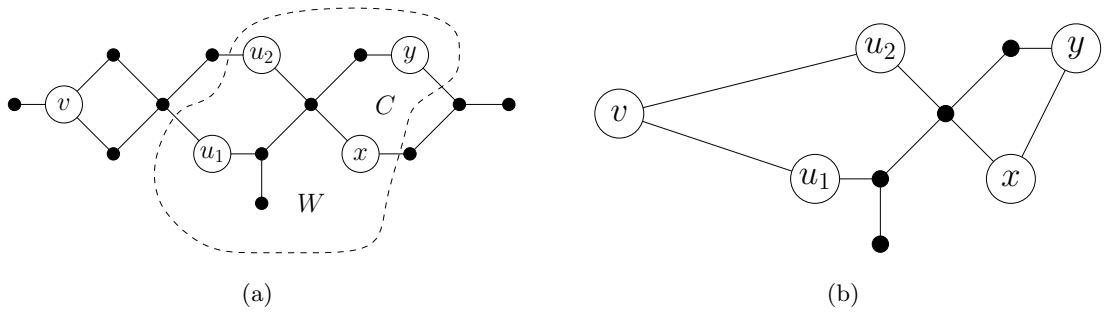


FIGURE 3.11: Illustration de la preuve du lemme 3.3. (a) Support cactus de H. Les sommets se trouvant à l'intérieur de l'enveloppe en pointillé appartiennent à la partie W de v. Les sommets u_1 et u_2 sont proches de v. Les sommets x et y sont les sommets terminaux de p_C. (b) Support cactus de $H[W \cup \{v\}]$.

Un hypergraphe $H = (V, A)$ possède un support qui est un cycle si et seulement si il satisfait la propriété des **uns consécutifs circulairement**, *i.e.* si et seulement si il existe un ordre $v_1, \ldots, v_n$ de ses sommets tel que pour chaque hyperarête $h \in A$, il existe $1 \leq j \leq k \leq n$ tel que $h = \{v_j, \ldots, v_k\}$ ou $V \setminus h = \{v_j, \ldots, v_k\}$. Nous obtenons ainsi le théorème suivant :

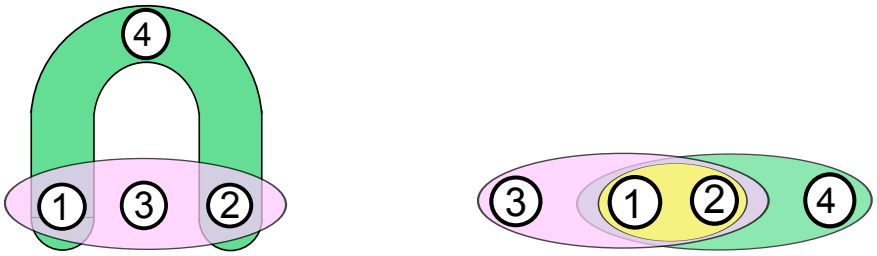
Théorème 3.1 *Il est possible de tester en temps $\mathcal{O}(nN+n+m)$ si un hypergraphe possède un cactus qui est un support.*

Preuve : Calculer tous les blocs en temps $O(nN + n + m)$ (voir lemme 3.1). Tester en temps linéaire la propriété des uns consécutifs circulairement pour chaque bloc [26]. □

Maintenant que nous avons réussi à montrer comment l'on peut déterminer la classe des hypergraphes possédant un support qui est un cactus, nous allons nous intéresser à une nouvelle classe d'hypergraphes que nous nommerons à intersections et à différences fermées.

3.1.3 Hypergraphes à intersections et différences fermées

Un hypergraphe $H = (V, A)$ est **à intersections fermées** si $h_1 \cap h_2 \in A \cup \{\emptyset\} \cup \{\{v\}; v \in V\}$ pour toutes les paires d'hyperarêtes (h_1, h_2) de A. Cette propriété est particulièrement utile pour le dessin d'un hypergraphe. En effet, fermer les intersections d'un hypergraphe permet d'assurer la création de régions connexes pour des sommets appartenant à la même intersection d'hyperarêtes comme le montre la figure 3.12.



(a) (b)

FIGURE 3.12: (a) Hypergraphe à intersections non fermées : les sommets 1 et 2 apparaissent dans deux régions séparées alors qu'ils appartiennent aux même hyperarêtes. (b) Hypergraphe à intersections fermées : l'hyperarête $\{1, 2\}$ a été ajoutée, un support de cet hypergraphe contiendra donc l'arête $(1, 2)$ et les deux sommets seront connexes sur le diagramme d'Euler.

Un premier résultat peut être énoncé pour ce type d'hypergraphes :

Lemme 3.4 *Un hypergraphe à intersections fermées possède un support planaire (extérieur) si et seulement si tous ses blocs possèdent un support planaire (extérieur).*

*Preuve :*La partie $\Leftarrow$ découle directement du lemme 3.2.

Intéressons-nous maintenant à la partie $\Rightarrow$. Soit $H = (V, A)$ un hypergraphe à intersections fermées et $G = (V, E)$ un support planaire (extérieur) de H. Soit $v \in V$ un sommet d'articulation de H et W une partie de v. Nous allons montrer par induction sur le nombre de sommets de $V \setminus W$ que $H[W \cup \{v\}]$ possède un support planaire (extérieur). Il n'y a rien à montrer si $V = W \cup \{v\}$.

Prenons donc $w \in V \setminus (W \cup \{v\})$. Nous construisons un support planaire (extérieur)

G' de $H' = (V \setminus \{w\}, \{h' \in A; w \notin h'\} \cup \{h' \setminus \{w\}; v \in h' \in A\})$. S'il n'existe pas d'hyperarête contenant v et w, G' est le graphe qui résulte de la suppression de w ainsi que de toutes les hyperarêtes qui le contiennent. Si ce n'est pas le cas, soit h l'intersection de toutes les hyperarêtes contenant v et w ($h \in A$ car l'hypergraphe est à intersections fermées). Alors, il existe un chemin allant de w à v dans $G[h]$. Soit w' le voisin de w dans ce chemin. Alors, G' peut être construit à partir de G en fusionnant w et w', *i.e.* pour tous les voisins $u \neq w'$ de w, nous ajoutons $\{u, w'\}$ à l'ensemble des arêtes de G. Ensuite, nous supprimons w ainsi que toutes ses arêtes incidentes pour obtenir G' planaire (extérieur).

En appliquant cette méthode, si $V \setminus \{w\} = W \cup \{v\}$ alors $H' = H[W \cup \{v\}]$. Sinon, puisque v est un sommet d'articulation et W est une partie de v dans H', par hypothèse d'induction, $H'[W \cup \{v\}] = H[W \cup \{v\}]$ possède un support planaire (extérieur). □

On se doit de remarquer ici que ce lemme ne permet pas de prouver que l'on peut déterminer en temps polynomial si un hypergraphe à intersections fermées possède un support planaire. En effet, déterminer si ses blocs possèdent un support planaire reste NP-complet. Ce résultat avait d'ailleurs déjà été démontré dans [106]. C'est pourquoi nous allons maintenant restreindre la classe des hypergraphes étudiés en se limitant à ceux qui sont aussi à différences fermées.

Un hypergraphe $H = (V, A)$ est **à différences fermées** si $h_1 \backslash h_2 \in A \cup \{\{v\}; v \in V\}$ pour toutes les paires d'hyperarêtes (h_1, h_2) de A. Cette propriété est aussi très intéressante pour le dessin d'un hypergraphe. En effet, fermer les différences d'un hypergraphe permet d'assurer la création de régions connexes pour des sommets appartenant à la même différence d'hyperarêtes comme le montre la figure 3.13.

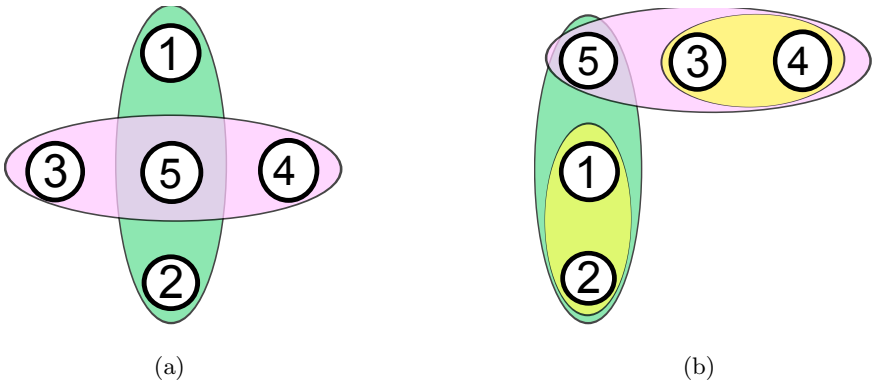


FIGURE 3.13: (a) Hypergraphe à différences non fermées : les sommets 1 et 2 apparaissent dans deux régions séparées alors qu'ils appartiennent exactement à la même hyperarête. (b) Hypergraphe à différences fermées : l'hyperarête $\{1, 2\}$ a été ajoutée, un support de cet hypergraphe contiendra donc l'arête $(1, 2)$ et les deux sommets seront connexes sur le diagramme d'Euler.

Nous allons maintenant voir qu'il est facile de décider si un hypergraphe à intersections et différences fermées possède un support planaire (extérieur). Pour cela, prenons un hypergraphe $H = (V, A)$. Soit $H_2 = (V, \{h \in A;\ |h| = 2\})$ l'hypergraphe composé des

hyperarêtes de H contenant exactement deux sommets. Cet hypergraphe est donc aussi un graphe. Nous allons maintenant montrer qu'il est un support de H si H est un bloc.

Lemme 3.5 *Si un hypergraphe H est à intersections et différences fermées et si il ne contient pas de sommet d'articulation, alors l'hypergraphe H_2 induit par les hyperarêtes de taille 2 de H est un support de H.*

Preuve : Soit $H = (V, A)$ un hypergraphe à intersections et différences fermées ne contenant pas de sommet d'articulation. Soit h une hyperarête de H. Nous allons montrer par induction sur la taille de h que $H_2[h]$ est connexe. Il n'y a rien à montrer pour $|h| \leq 2$. Supposons donc que $|h| > 2$.

Considérons d'abord que $h \neq V$. Puisque H ne contient pas de sommet d'articulation, il existe au moins deux hyperarêtes h_1, h_2 telles que $h_1 \cap h \neq h_2 \cap h$ et dont les intersections avec h sont non vides. Les fermetures du graphe nous disent alors que $h \cap h_i, h \setminus h_i \in A \cup \{\{v\};\ v \in V\}, i = 1, 2$. Par hypothèse d'induction, les quatre hypergraphes $H_2[h \cap h_i]$ et $H_2[h \setminus h_i], i = 1, 2$ sont connexes. Si $h \cap h_1 \neq h \setminus h_2$ alors $H_2[h]$ est connexe.

Supposons au contraire qu'il n'existe pas de paire (h_1, h_2) telle que $h_1 \cap h \neq h_2 \cap h$ et $h \cap h_1 \neq h \setminus h_2$. On a alors $h \cap h_1 = h \setminus h_2$ et il existe donc une partition h^1, h^2 de h telle que pour toute hyperarête h_1 partageant des sommets avec h, on a $h \cap h_1 = h^1$ ou $h \cap h_1 = h^2$. Voir la figure 3.14 pour une illustration de cette preuve. On peut remarquer aussi que selon notre hypothèse d'induction, $H_2[h^1]$ et $H_2[h^2]$ sont connexes. Puisque h contient plus de deux sommets, nous pouvons considérer sans perdre de généralité que h^1 contient au moins deux sommets. Si $|h^2| = 1$, il existe une hyperarête h' qui contient h^2 et dont l'intersection avec h^1 n'est pas vide (dans le cas contraire, chaque sommet de h^1 serait un sommet d'articulation). De même, si $|h^2| > 1$, il existe une hyperarête h' dont les intersections avec h^1 et h^2 ne sont pas vides. Soit h' l'hyperarête de taille minimale ayant cette propriété. Considérons $i = 1$ ou $i = 2$ tel que $|h' \cap h^i| > 1$ (dans ce cas, on a donc $|h'| > 2$). Puisque $H_2[h^i]$ est connexe, il existe deux sommets $v \in h^i \cap h'$ et $w \in h^i \setminus h'$ tels que $\{v, w\}$ est une hyperarête. Cependant, la taille de $h' \setminus \{v, w\} \in A$ est moins élevée que celle de h' ce qui est une contradiction. On a donc $|h'| = 2$. Ainsi, $H_2[h]$ contient les sous-graphes connexes $H_2[h^i], i = 1, 2$ connectés par l'arête h'. $H_2[h]$ est donc connexe.

Revenons maintenant au cas où $h = V$. L'hypergraphe $(V, A \setminus \{V\})$ est nécessairement connexe puisque dans le cas contraire, tous les sommets de H seraient des sommets d'articulation. Puisque $H_2[h']$ est connexe pour toute hyperarête $h' \neq V$, $H_2[V]$ est connexe. □

On peut remarquer ici que les hyperarêtes de taille 2 appartiennent forcément à n'importe quel support d'un hypergraphe. Nous obtenons donc le théorème suivant :

Théorème 3.2 *Il est possible de décider en temps $\mathcal{O}(nN + n + m)$ si un hypergraphe à intersections et différences fermées possède un support planaire (extérieur).*

Preuve : Commencer par décomposer l'hypergraphe en blocs. D'après le lemme 3.4, l'hypergraphe est planaire (extérieur) si et seulement si ses blocs sont planaires (extérieur).

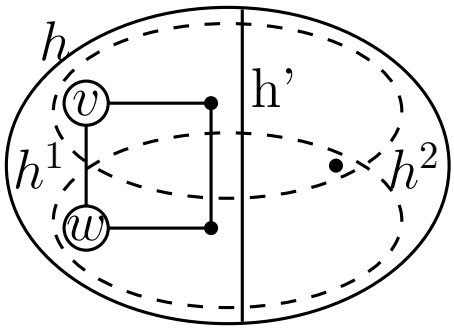


FIGURE 3.14: Illustration de la preuve du lemme 3.5.

Pour le savoir, d'après le lemme 3.5, il suffit de tester pour chacun d'entre eux si le graphe induit par les hyperarêtes de taille 2 est planaire (extérieur). □

Cette partie nous a permis d'identifier une classe d'hypergraphes pour lesquels il est possible de savoir s'ils sont planaires ou planaires extérieurs. Ces hypergraphes sont caractérisés par la fermeture de leurs intersections et de leurs différences. L'avantage d'un tel résultat réside dans le fait qu'il est facile de fermer un hypergraphe quelconque. En revanche, précisons qu'une telle opération a de grandes chances de transformer un hypergraphe planaire en hypergraphe non planaire.

3.1.4 Conclusion

Reprenons pour conclure cette section la liste des résultats existant sur les problèmes de décision concernant la nature du support d'un hypergraphe, en y ajoutant ceux que nous venons de présenter (les problèmes résolus sont notés en vert et les problèmes ouverts en rouge) :

- planaire :
  - NP-complet [106], même si l'hypergraphe est à intersections fermées
  - temps polynomial si l'hypergraphe est à intersections et différences fermées [30]
- 3-planaire extérieur : NP-complet [37]
- 2-planaire extérieur : NP-complet [36]
- planaire extérieur :
  - cas général ?
  - temps polynomial si l'hypergraphe est à intersections et différences fermées [30]
- cactus : temps polynomial [30]
- arbre : temps linéaire [173]
- chemin : temps linéaire [37]
- cycle : temps linéaire [37]

Cette partie nous a donc permis de voir qu'il existe certaines classes d'hypergraphes pour lesquels nous sommes capables de construire des diagrammes d'Euler corrects. Une première remarque concerne les hypergraphes planaires extérieurs : ce problème central reste ouvert. Une deuxième remarque concerne les applications en visualisation que ces

résultats nous apportent. En y regardant de plus près, on peut voir que les classes d'hypergraphes pouvant être représentés grâce à un diagramme d'Euler sont très restrictives. C'est pourquoi nous allons maintenant proposer une ébauche de solution de représentation basée sur un nouveau paradigme.

3.2 Supports dans lesquels des chemins modélisent les hyperarêtes

Les résultats qui vont être présentés dans cette partie ont aussi été réalisés avec Ulrik Brandes, Sabine Cornelsen et Barbara Pampel. Certains ont déjà fait l'objet d'une publication [31]. Nous y introduisons une nouvelle métaphore de représentation d'hypergraphe. Celle-ci consiste à dessiner l'hypergraphe comme une carte de métro. Prenons l'exemple de la carte du métro d'Amsterdam (voir figure 3.15). Chaque ligne peut être vue comme une hyperarête et chaque station comme un sommet. Ainsi, les sommets appartenant à plusieurs hyperarêtes sont traversés par plusieurs lignes. Par rapport aux diagrammes d'Euler classiques, une telle représentation permet de simplifier le dessin : on peut ainsi admettre des croisements de lignes sans perturber la compréhension de la visualisation. Cette idée est motivée par un article d'Alper *et al.* [1] dans lequel une étude menée sur des utilisateurs montre que la visualisation d'ensembles d'éléments géolocalisés représentés par des lignes permet de résoudre certains problèmes plus facilement que lorsqu'ils sont représentés à l'aide d'enveloppes.

Les graphes supports constituent ici aussi des outils adaptés à la représentation des hypergraphes selon cette métaphore. Cependant, leur définition doit être restreinte. C'est pourquoi nous introduisons la notion de **support basé sur des chemins**. Afin de définir cette notion, nous rappelons au lecteur qu'un **graphe hamiltonien** est un graphe contenant un chemin passant par tous ses sommets une seule fois. Prenons un hypergraphe H et un de ses supports G. Celui-ci est basé sur des chemins si il existe un chemin hamiltonien passant par tous les sommets de chaque hyperarête, *i.e.* $\forall h \in A$, $G[h]$ est un graphe hamiltonien. La figure 3.16 illustre ceci. L'hypergraphe de départ est représenté sous forme d'un diagramme d'Euler par la figure 3.16.a. Le support représenté par la figure 3.16.b n'est pas basé sur des chemins : $G[\{4, 5, 6, 8\}]$ n'est pas hamiltonien. En revanche, celui de la figure 3.16.c est bien basé sur des chemins. Il permet ainsi de construire un représentation en accord avec la métaphore des cartes de métro (voir figure 3.16.d).

Nous allons maintenant présenter certains résultats de notre article [31] qui concernent la complexité de problèmes importants sur le calcul de supports basés sur des chemins (sous-section 3.2.1). Ensuite, nous proposerons une ébauche de représentation (sous-section 3.2.2).

3.2.1 Les supports basés sur des chemins minimaux et planaires

Nous allons d'abord nous intéresser aux hypergraphes dont les supports basés sur des chemins sont planaires. Si nous pouvions les caractériser, les dessiner de façon planaire permettrait d'améliorer leur représentation. Malheureusement, ce problème est difficile :

FIGURE 3.15: Plan du métro d'Amsterdam.

Théorème 3.3 *Décider si un hypergraphe (même à intersections fermées) possède un support planaire basé sur des chemins est un problème NP-Complet.*

Preuve : Le support que Johnson et Pollak [106] construisent pour prouver que le problème de décider si un hypergraphe possède un support planaire est NP-complet est déjà un support basé sur des chemins. □

Interrogeons-nous maintenant sur la taille d'un support basé sur des chemins. Selon les chemins choisis pour chaque hyperarête, le nombre des arêtes du support peut varier comme le montre la figure 3.17. Dans cet exemple, en prenant les chemins $\{1, 2, 3\}$ et $\{1, 3\}$, on obtient un support dont le nombre d'arêtes est 3 (figure 3.17.a). Ce nombre est réduit à 2 en prenant les chemins $\{1, 3, 2\}$ et $\{1, 3\}$ (figure 3.17.b). Il serait donc utile, afin de simplifier le dessin final, de trouver les chemins qui minimisent le nombre d'arêtes du support.

Il est déjà intéressant de remarquer que l'on peut facilement construire pour n'importe quel hypergraphe $H = (V, A)$ un support basé sur des chemins $G = (V, E)$ contenant au plus $N - m$ arêtes. Pour cela, il faut commencer par ordonner arbitrairement les sommets. Puis, pour chaque hyperarête $\{v_1, \ldots, v_k\} \in A$ avec $v_1 < \cdots < v_k$, il suffit de mettre dans

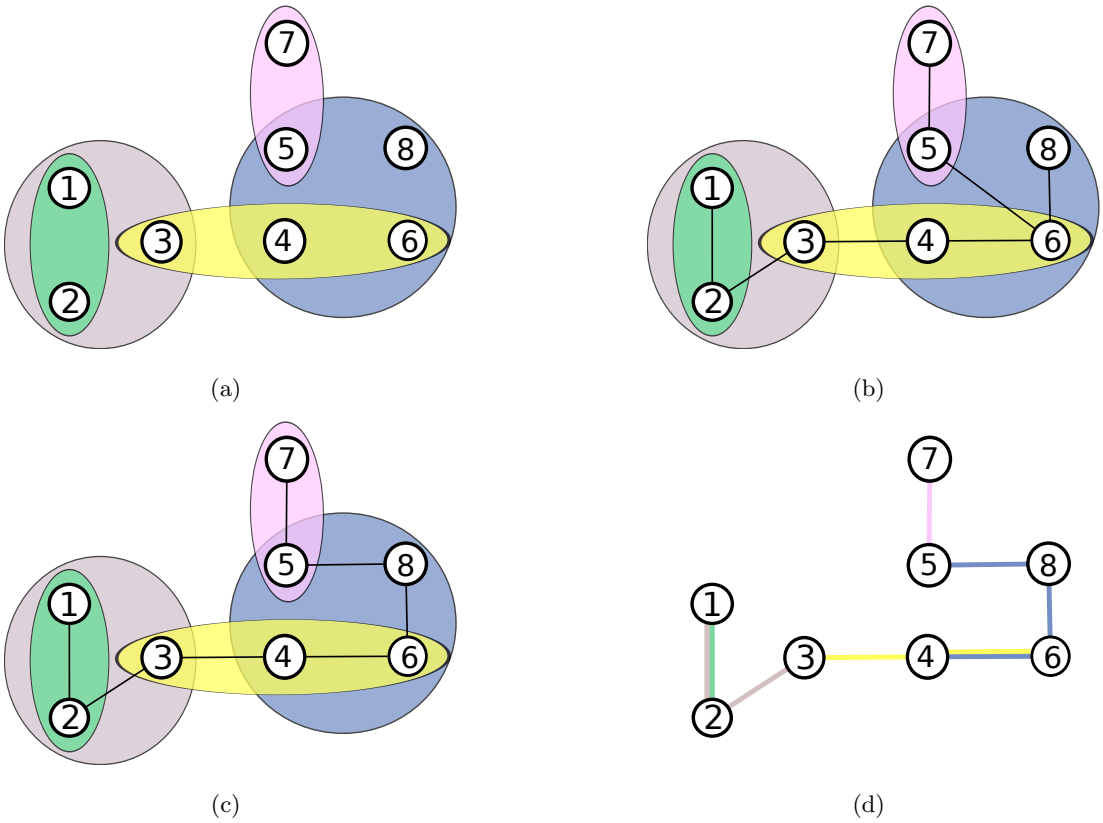


FIGURE 3.16: $H = (V, A)$ avec $V = \{1, 2, 3, 4, 5, 6, 7\}$ et $A = \{\{1, 2\}, \{1, 2, 3\}, \{3, 4, 6\}, \{4, 5, 6, 8\}, \{5, 7\}\}$. (a) Diagramme d'Euler représentant l'hypergraphe. (b) Les arêtes noires représentent un support G non basé sur des chemins ($G[\{4, 5, 6, 8\}]$ ne contient pas de chemin hamiltonien). (c) Les arêtes noires représentent un support G basé sur des chemins. (d) Représentation de l'hypergraphe selon la métaphore des cartes de métro.

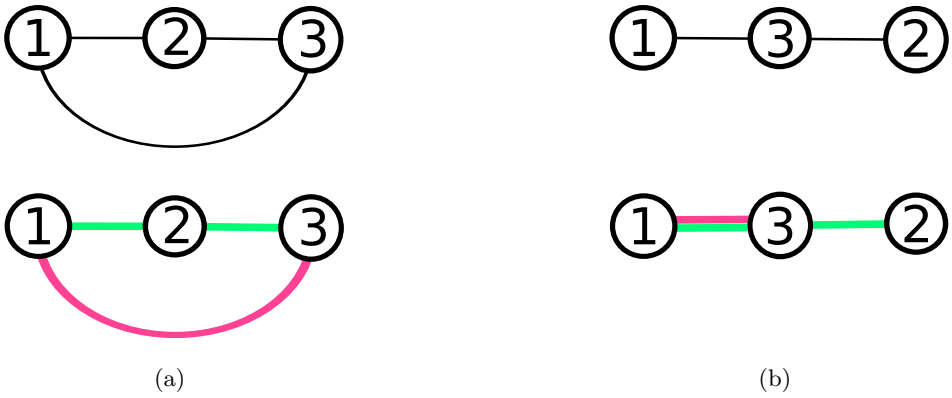


FIGURE 3.17: $H = (V, A)$ avec $V = \{1, 2, 3\}$ et $A = \{\{1, 2, 3\}, \{1, 3\}\}$. (a) Support non minimal et sa représentation selon la métaphore des cartes de métro. (b) Support minimal et sa représentation selon la métaphore des cartes de métro.

E les arêtes $(v_{i-1}, v_i), i = 1, \ldots, k$. Malheureusement, trouver un ordre permettant de minimiser le nombre d'arêtes d'un support construit de cette façon est un problème NP-complet. En effet, prenons une matrice booléenne $M = \{m_{i,j}\}$ où $m_{i,j} = 1$ si $j \in h_i$ avec $j \in V$ et $h_i \in A$ et $m_{i,j} = 0$ sinon. Le problème revient alors à ordonner les colonnes de cette matrice de façon à réduire le nombre de séquences de 1 consécutifs sur les lignes de la matrice, ce qui est NP-complet [105].

Quoiqu'il en soit, même si nous pouvions ordonner les sommets de façon à minimiser le nombre d'arêtes d'un support construit de cette façon, ce support ne serait pas forcément de taille minimale. En effet, prenons comme exemple l'hypergraphe composé des sommets $\{1, 2, 3, 4\}$ et des hyperarêtes $h_1 = \{1, 2, 4\}$, $h_2 = \{1, 3, 4\}$, et $h_3 = \{2, 3, 4\}$. Une matrice ordonnée contenant un nombre minimal de séquences de 1 consécutifs est :

$$\begin{array}{c} \\ h_1 \\ h_2 \\ h_3 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 4 & 3 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Si l'on construit le support $G = (V, E)$ correspondant, son nombre d'arêtes est 4 : $E = \{(1, 2), (2, 4), (1, 4), (4, 3)\}$. Or, il existe un support dont le nombre d'arêtes est 3 : $E = \{(1, 4), (2, 4), (3, 4)\}$. Essayons donc maintenant de nous libérer de la contrainte d'une telle construction. Nous obtenons alors, malheureusement, le résultat suivant :

Théorème 3.4 *Minimiser le nombre d'arêtes d'un support basé sur des chemins est un problème NP-complet, même si l'hypergraphe est à intersections fermées.*

Preuve : Le problème de trouver un chemin hamiltonien dans un graphe est NP-complet [80]. On peut réduire ce problème au nôtre. Prenons un graphe $G = (V, E)$. Soit l'hypergraphe $H = (V, A = E \cup \{V\} \cup \{\{v\};\ v \in V\})$. Alors, G contient un chemin hamiltonien si et seulement si H possède un support basé sur des chemins avec $|E|$ arêtes. En effet, puisque $E \in A$, toutes les arêtes E de G appartiennent à un support de H. De plus, si G est hamiltonien, alors le chemin représentant l'hyperarête $\{V\}$ de H ne nécessite pas l'ajout d'arêtes supplémentaires dans le support et la taille de ce support est minimale. Donc un algorithme polynomial trouvant un support minimal de taille E si celui-ci existe résoudrait, par la même occasion, le problème de décision du chemin hamiltonien. □

Nous allons maintenant voir comment de tels résultats peuvent être utilisés afin de proposer une visualisation d'hypergraphes selon la métaphore des cartes géographiques.

3.2.2 Vers une visualisation

Nous allons proposer dans cette sous-section une idée d'algorithme permettant de représenter les hypergraphes selon la métaphore des lignes de métro. Ce travail étant actuellement en cours, les résultats que nous allons donner doivent être considérés comme une ébauche de solution. L'algorithme imaginé prend en entrée un hypergraphe $H = (V, A)$. $H' = (V', A')$ dénote une copie de H modifiée au cours du traitement. Une **ligne** est un graphe hamiltonien passant par tous les sommets d'une hyperarête. Soit

$\{l_{h_1}, l_{h_2}, ..., l_{h_m}\}$ un ensemble de lignes des hyperarêtes de H, alors $G = (V, l_{h_1} \cup l_{h_2} \cup \cdots \cup l_{h_m})$ est un support basé sur les arêtes de H. Voici l'algorithme imaginé :

1. Pré-calculs
   a. Rendre simple H'.
   b. $\forall h_1, h_2 \in A$, $A' \leftarrow A' \cup h$ avec $h = h_1 \cap h_2$.
2. Création du support
   a. Supprimer de H' les sommets de degré 1.
   b. Calculer les lignes et le support $G = (V', E)$ correspondant.
   c. Réinsérer les sommets de degré 1 supprimés en 2.a.
   d. Construire le support G.
3. Création de la carte
   a. Dans G, remplacer les chemins de sommets de degrés 2 par une arête.
   b. Trouver un placement initial des sommets de G.
   c. Améliorer le placement précédent à l'aide d'un algorithme de dessin de cartes de métro.
   d. Réinsérer les sommets supprimés en 3.1.
   e. Rendu des lignes.

Avant de détailler les différentes étapes évoquées ci-dessus, étudions la représentation que cet algorithme produit grâce à des données extraites de l'*Internet Movie Data Base*[6]. Pour cela, prenons un hypergraphe dont les sommets représentent les acteurs ayant joué dans des films de Jean-Pierre Jeunet. Les hyperarêtes représentent les films. Un acteur ayant joué dans un film h appartiendra donc à l'hyperarête h. La figure 3.18 montre le graphe biparti correspondant à notre hypergraphe de départ.

La figure 3.19 montre une représentation de l'hypergraphe décrit ci-dessus en utilisant notre algorithme de dessin. Comme nous l'avons déjà mentionné, ces travaux sont en cours de développement et nous n'avons pas pu tester l'efficacité de ce type de représentation sur des utilisateurs finaux. De plus, comme vous pourrez le voir ci-dessous, nous ne sommes pas encore complètement satisfaits de l'algorithme. Il reste encore à (ré)implémenter certaines parties avant de le soumettre au jugement des utilisateurs. Nous espérons cependant que de telles cartes les aideront à accomplir certaines tâches, comme nous le laisse supposer l'article [1].

Nous allons maintenant passer à la description des étapes évoquées ci-dessus.

3.2.2.1 Pré-calculs

Un hypergraphe est **simple** si deux hyperarêtes ne contiennent pas exactement les mêmes sommets. Imaginons que nous ayons dans H deux hyperarêtes distinctes $h_1 \in A$ et $h_2 \in A$ telles que $h_1 = h_2$ (l'hypergraphe n'est donc pas simple). Supprimer l'une de ces hyperarêtes permet de faciliter le processus de visualisation. En effet, lorsqu'une ligne est

6. IMDB : http ://www.imdb.com/

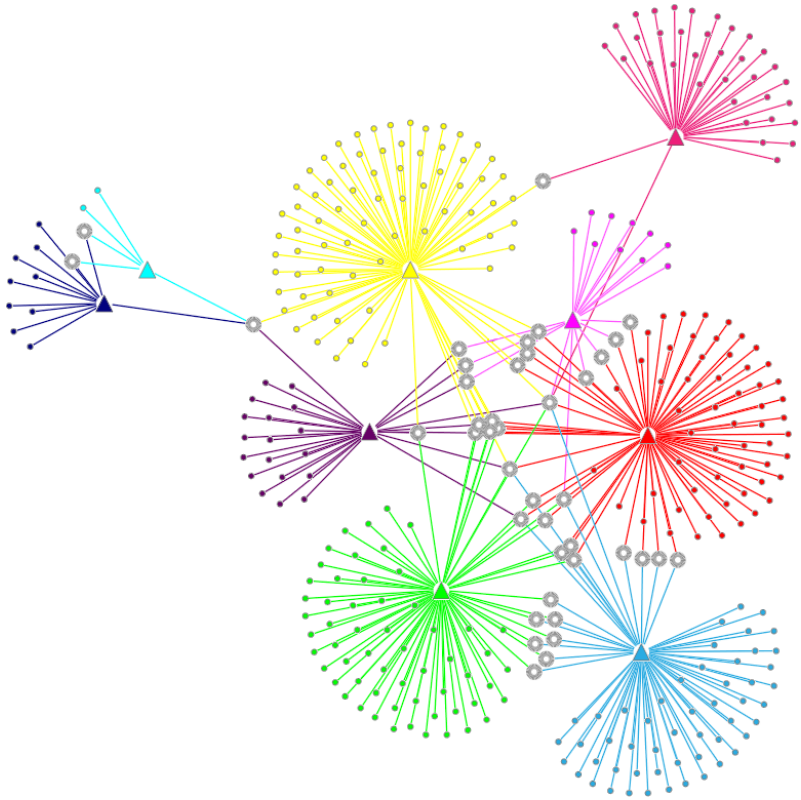
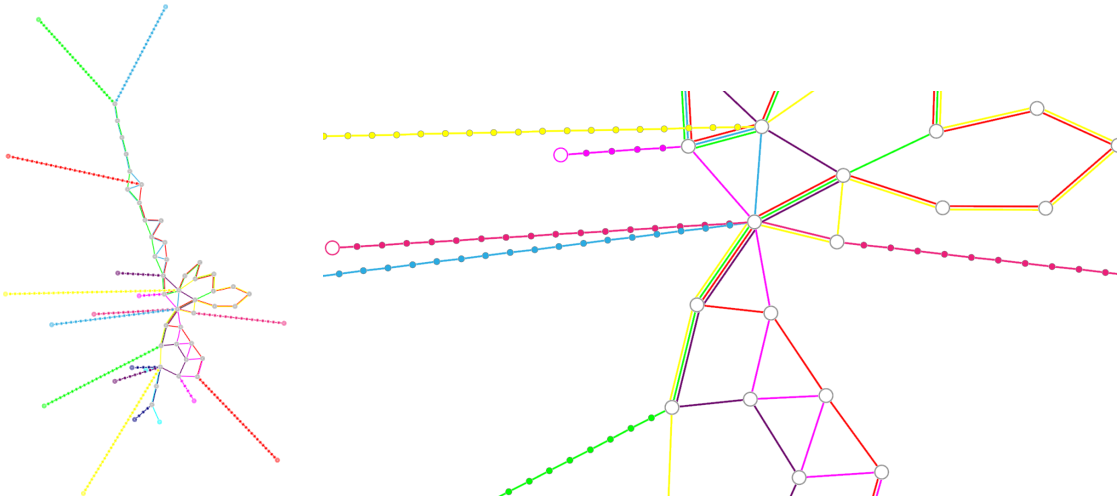


FIGURE 3.18: Représentation du graphe biparti associé à notre hypergraphe de départ. Les triangles représentent les films de Jean-Pierre Jeunet. Les cercles de couleur représentent les acteurs ayant joué dans un seul film. Les cercles gris représentent les acteurs ayant joué dans plusieurs films.



(a) (b)

FIGURE 3.19: Représentation de l'hypergraphe générée par notre algorithme. (a) Ensemble de la représentation. (b) Sous-partie de la représentation.

trouvée pour les sommets de h_1, il est inutile d'en calculer une autre pour les sommets de h_2. La première étape de l'algorithme (1.a) consiste donc à rendre **simple** l'hypergraphe H'.

La seconde étape (1.b) consiste à ajouter à H' les intersections des paires d'hyperarêtes de H. Ainsi, une ligne sera créée pour ces intersections et l'on pourra limiter les représentations telles que celle montrée par la figure 3.17.a. Remarquons cependant qu'afin de ne pas trop augmenter le nombre de lignes à calculer, l'hypergraphe H' n'est pas entièrement fermé par ses intersections. En effet, nous n'ajoutons pas les intersections des hyperarêtes représentant les intersections des hyperarêtes de H. Encore une fois, précisons que ce projet est en cours d'élaboration et qu'il nous a semblé que les dessins ainsi produits étaient de meilleure qualité. Nous ne pouvons cependant rien affirmer avant d'avoir mené plusieurs expériences avec des utilisateurs finaux sur des jeux de données diverses.

3.2.2.2 Création du support

Nous allons maintenant proposer une technique permettant de calculer pour chaque hyperarêtes $h \in A'$ une ligne l_h en essayant de minimiser la taille du support induit par ces lignes (cf. sous-section 3.2.1). Tout d'abord, nous supprimons de H' les sommets n'appartenant qu'à une seule hyperarête (étape 2.a). En effet, les arêtes reliant ces sommets au reste du graphe support seront nécessaires et n'influenceront en aucun cas la taille de ce support. C'est pourquoi nous les rajouterons au début ou à la fin des lignes après avoir calculé celles-ci sans en tenir compte.

Une fois ces sommets supprimés, nous construisons à partir de $H' = (V', A')$ un graphe $G'(V', E', \omega)$ où ω est une fonction de E' dans $\mathbb{N}$. $(u, v) \in E'$ si $\exists h \in A'$ tel que $u \in h$ et $v \in h$. $\omega(u, v)$ correspond au nombre d'hyperarêtes contenant à la fois u et v. Ensuite, pour chaque hyperarête h, nous essayons de trouver un ordre $\pi(h) : h \to \{1, ..., |h|\}$ qui minimise la fonction :

$$LA(\pi, h) = \sum_{(u,v) \in E'} \omega(u, v).|\pi(u) - \pi(v)|$$

Ainsi, deux sommets apparaissant ensemble dans de nombreuses hyperarêtes ont une forte probabilité d'être consécutifs dans l'ordre. Celui-ci permet ensuite de construire une ligne en ajoutant des arêtes entre ses sommets consécutifs. Ce procédé permet enfin de construire un support dont le nombre d'arêtes n'est pas trop élevé.

Trouver l'ordre minimisant la fonction LA est malheureusement un problème NP-complet connu sous le nom de « problème d'arrangement linéaire minimal[7] (MinLA) » [81]. Il existe cependant de nombreuses heuristiques permettant d'en trouver des approximations. Une comparaison de ces techniques nous est donnée par Jordi Petit [140]. Un algorithme performant et rapide a aussi été proposé par Yehuda Koren et David Harel [117].

Après avoir utilisé l'une de ces heuristiques afin de déterminer un ordre $\pi(h_i)$ pour chaque hyperarête $h_i \in A'$ (étape 2.b), nous construisons les lignes $\{l_{h_1}, l_{h_2}, ..., l_{h_m}\}$ en y

7. *Minimum Linear Arrangmement Problem*

ajoutant les sommets supprimés lors de l'étape 2.a (étape 2.c). Enfin, l'étape 2.d consiste à construire le support basé sur des chemins $G = (V, l_{h_1} \cup l_{h_2} \cup \cdots \cup l_{h_m})$.

La solution proposée ici n'est pas forcément la meilleure qui soit : il serait utile d'établir une liste d'heuristique possibles et de mener des évaluations afin de sélectionner la meilleure.

3.2.2.3 Création de la carte

Le positionnement des sommets est effectué en 4 étapes. Tout d'abord (étape 3.a), nous remplaçons les chemins de sommets de degré 2 dans G par une arête dont la taille correspond au nombre de sommets supprimés plus 1 (les autres arêtes ont une taille de 1). Ceci permet d'afficher ces sommets sur une seule ligne lors de leur réintroduction.

Nous voudrions maintenant utiliser un algorithme de dessin de carte de métro. Cependant, ces algorithmes nécessitent un positionnement initial (le positionnement géographique des stations). L'étape 3.b consiste donc à appliquer un algorithme de placement applicable sur un graphe quelconque. Dans notre exemple, nous avons utilisé FM^3 [90].

Une fois ce placement obtenu, un algorithme de disposition des stations de métro peut être utilisé (étape 3.c). Wolff nous livre une vue d'ensemble de ce type de techniques [201]. Nous pouvons aussi envisager les solutions proposées par Nöllenburg et Wolff [137] ou par Stott *et al.* [171]. Ce travail étant en cours de production, nous n'avons pas encore mis en place une telle méthode. Les images de la figure 3.19 ont été générées en la remplaçant par l'algorithme *MDS* présenté dans l'article [79].

Une fois les positions trouvées, les sommets supprimés lors de l'étape 3.a sont réinsérés uniformément le long des arêtes correspondantes (étape 3.d). Enfin, les lignes sont dessinées le long des arêtes du support. Lorsque plusieurs lignes passent par la même arête, elles sont juxtaposées afin de pouvoir les distinguer (étape 3.e).

3.3 Conclusion

Comme le lecteur aura pu s'en persuader tout au long de ce chapitre, la visualisation d'hypergraphe est un domaine peu développé et de nombreux problèmes restent ouverts aussi bien au niveau théorique qu'au niveau pratique. Nous avons présenté dans la première section quelques résultats théoriques, mais le plus important d'entre eux concernant la « planarité » extérieure des hypergraphes reste ouvert. En ce qui concerne la visualisation, nous avons proposé l'ébauche d'une solution qui semble prometteuse. Cependant, il reste beaucoup de travaux à effectuer avant de pouvoir valider cette approche.

Chapitre 4

Articles publiés et communications diverses

Comme son titre l'indique, ce chapitre va nous permettre de résumer les articles publiés au cours de cette thèse ainsi que de mentionner les communications diverses. Les trois lettres qui précèderont les références bibliographiques symboliseront la portée de la communication (I pour internationale, F pour française), le type de la communication (J pour journal, C pour conférence, P pour poster, T pour présentation autre que donnée en conférence) et le numéro permettant de l'identifier dans le texte. Les présentations des articles données en conférences par l'auteur de cette thèse seront indiquées par *. Par exemple, IC1* signifie « premier article publié en conférence internationale et présenté par l'auteur lors de cette conférence ».

4.1 Visualisation de séquences de gènes

Les communications présentées ci-dessous concernent l'outil de visualisation présenté dans le premier chapitre de cette thèse. [IC1*] ne contient que la visualisation des groupes (sans le treemap). [FC1*] et [IC1*] présentent tous les modules sans l'étude auprès des utilisateurs. Enfin, [IJ1] présente l'ensemble des résultats (description de l'outil et étude utilisateur) en détaillant les algorithmes utilisés.

IJ1 Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche and Maguelonne Teisseire. Sequential Patterns Mining and Gene Sequence Visualization to Discover Novelty from Microarray Data. Journal of Biomedical Informatics, 44(5), pages 760-774, 2011.

IC1* Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche and Maguelonne Teisseire. Discovering Novelty in Gene Data : From Sequential Patterns to Visualization. In Proceedings of the 6th International Symposium on Visual Computing (ISVC10), volume 6455 of Lecture Notes in Computer Science, pages 534-543. Part III, Springer-Verlag Berlin Heidelberg, 2010.

IP1 Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche and Maguelonne Teisseire. SequencesViewer : visualization of genes sequences. 13th World Congress on Medical and Health Informatics (Medinfo 2010), 2010.

FC1* Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche et Maguelonne Teisseire. SequencesViewer : Visualisation de séquences ordonnées de gènes ou comment rendre accessible des motifs séquentiels trop nombreux ?. Dans les actes de la 10ième Conférence Internationale Francophone sur l'Extraction des Connaissances (EGC 2010), Revue des Nouvelles Technologies de l'Information (RNTI E-19), pages 387-392, 2010.

4.2 Analyse, fragmentation et visualisation de graphes

[IC4] et [IT1*] présentent les premiers résultats obtenus sur une étude comparative des métriques de graphes en cours de production avec Guy Melançon (nous avons mentionné cette étude dans la section 2.1.2.2).

Innovanews, présenté dans [IP2], est une application de visualisation de communiqués paraissant sur le web, extraits et classifiés par la société *Digimind*. Ce système est constitué d'une compilation de quatre vues permettant d'aborder l'actualité selon différentes perspectives (qualitative, quantitative, géographique, ...). Il a été présenté au public lors de l'exposition « l'observatoire des innovations » [1], organisé par la cité des sciences [2].

[IC3] détaille l'algorithme de fragmentation présenté dans la section 2.1.2.2. [IC2*] et [FT1*] reprennent cet algorithme de façon succincte et décrivent le système de visualisation et de navigation proposé dans la section 2.1.3.

FT1* Arnaud Sallaberry, Faraz Zaidi, Christian Pich et Guy Melançon. Visualisation interactive de pages web retournées par un moteur de recherche de façon à révéler les communautés et les ponts. Séminaire des doctorants (SéminDoc), LIRMM, Décembre 2010.

IC2* Arnaud Sallaberry, Faraz Zaidi, Christian Pich, and Guy Melançon. Interactive visualization and navigation of web search results revealing community structures and bridges. In Proceedings of the 36th Graphics Interface conference (GI'10), pages 105–112, 2010.

IC3 Faraz Zaidi, Arnaud Sallaberry, and Guy Melançon. Revealing hidden community structures and identifying bridges in complex networks : An application to analyzing contents of web pages for browsing. In Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence (WI'09), pages 198–205, 2009.

IP2 Arnaud Sallaberry, Guillaume Aveline and Guy Melançon. InfoVis for the Masses : Easy Navigation and Interactive Browsing of a Digital Catalog. 13th International Conference Information Visualisation (IV'09), 2009.

IT1* Arnaud Sallaberry and Guy Melançon. A comparative study of graph metrics using graph models. Universtät Konstanz. Juin 2009.

IC4 Guy Melançon and Arnaud Sallaberry. Edge Metrics for Visual Graph Analytics : A Comparative Study. Proceedings of the 12th International Conference Information Visualisation (IV'08), pages 610-615, 2008.

1. http ://www.cite-sciences.fr/csmedia/storage/Presse_PJ/DP_Observatoire_Innovations_2.pdf
2. http ://www.cite-sciences.fr/fr/cite-des-sciences/

4.3 Dessin d'hypergraphes

[IC5*] présente les résultats décrits dans la section 3.1 sur la « planarité » des hypergraphes. [IC6] décrit les premiers résultats théoriques sur les supports basés sur des chemins. Certains d'entre eux font l'objet de la section 3.2.1.

IC5* Ulrik Brandes, Sabine Cornelsen, Barbara Pampel, and Arnaud Sallaberry. Blocks of Hypergraphs applied to Hypergraphs and Outerplanarity. In Proceedings of the 21th International Workshop on Combinatorial Algorithms (IWOCA 2010), volume 6460 of Lecture Notes in Computer Science, pages 201-211. Springer, 2010.

IC6 Ulrik Brandes, Sabine Cornelsen, Barbara Pampel, and Arnaud Sallaberry. Path-based supports for hypergraphs. In Proceedings of the 21th International Workshop on Combinatorial Algorithms (IWOCA 2010), volume 6460 of Lecture Notes in Computer Science, pages 20-33. Springer, 2010.

Conclusion et perspectives

La visualisation de l'information est une discipline récente (fin du XXième siècle) en pleine expansion mais dont certains exemples remontent à l'Antiquité comme la carte d'Anaximandre que nous avons évoquée en introduction. D'abord présentée comme l'étude des techniques de représentation visuelle de données abstraites, nous avons limité cette définition dans le premier chapitre aux techniques informatisées afin de la distinguer de la sémiotique graphique introduite par Jacques Bertin [23, 22]. La sémiotique est quant à elle une discipline beaucoup plus ancienne. Elle a été introduite au XIXième siècle par Peirce [139] et a fait l'objet de nombreux travaux tout au long du XXième siècle. Cette science étudie les systèmes de signes utilisés par des entités pour communiquer. Or, un système de visualisation est lui aussi basé sur un système de signes graphiques permettant la transmission de messages. Nous avons donc dans le premier chapitre de cette thèse mis en parallèle ces deux disciplines.

Plus précisément, nous avons mené une étude de la visualisation de l'information selon les principaux paradigmes établis par les sémioticiens. Nous nous sommes limités à des concepts fondamentaux mais cela nous a cependant permis d'obtenir quelques résultats intéressants, qui pourraient justifier la mise en place de travaux futurs plus ambitieux dans cette voie. Entre autres, nous avons établit une distinction nette entre l'art et la science de la visualisation de l'information et trouvé une place à ce domaine de recherche parmi les sciences sémiotiques. Ceci nous permettra peut-être d'emprunter des lois aux domaines plus généraux, d'en découvrir de nouvelles en généralisant les nôtres, d'en trouver des contre-exemples ou d'en confirmer d'autres. Par exemple, nous avons montré comment le modèle de validation de systèmes de visualisation d'information proposé par Tamara Munzner [131] est très proche du processus d'étude d'énoncés utilisé par les sémioticiens : étude syntaxique, puis sémantique et enfin pragmatique. À notre connaissance, ces deux modèles ont été développés indépendamment, ce qui semble être un argument pour leur adoption aussi bien en sémiotique générale qu'en visualisation de l'information.

Le principal point faible de l'étude telle qu'elle a été faite est que nous n'avons pas abordé le problème de l'interaction, élément si crucial de la recherche en visualisation de l'information. Ceci est principalement dû au fait que ce genre de question a été très peu étudié par les sémioticiens car la plupart des énoncés qu'ils manipulent ne sont pas interactifs parce que non informatisés. Il semblerait d'ailleurs que ce type de problèmes soit spécifique à notre domaine. Une approche pourrait être de considérer un système de visualisation comme un ensemble de cartes (potentielles) et un ensemble d'« interacteurs » permettant de naviguer dans ces cartes grâce à la modification, la suppression ou l'ajout d'unités. Une question reste cependant ouverte : est-ce que cette partie relève de la syntaxe

comme nous l'avons suggéré en rapprochant le troisième niveau du modèle de Munzner à la dimension syntaxique, ou est-ce une nouvelle dimension ?

Les graphes sont des outils puissants permettant de modéliser un grand nombre de types de données abstraites. C'est pourquoi nous avons consacré le second chapitre de cette thèse à l'étude de deux systèmes de visualisation de l'information basés sur la représentation de graphes. Le premier permet de naviguer à travers un ensemble de pages *Internet* retournées par un moteur de recherche. L'apport de notre technique est double. Tout d'abord, nous proposons une méthode de fragmentation des mots-clés des pages. Cette fragmentation, basée sur le réseau de co-occurrence des mots, permet de les regrouper selon leur proximité sémantique. De plus, des concepts « ponts » reliant les communautés sont identifiés afin de faciliter la navigation. Nous pouvons dés lors proposer une solution permettant de naviguer à travers ces communautés afin d'accéder rapidement aux pages *Internet* qui les contiennent. Notre méthode est basée sur la combinaison de trois algorithmes de *multidimensional scaling*. Le premier permet de trouver un positionnement global des communautés sur l'écran et le deuxième améliore cette première configuration. Enfin, le troisième supprime les chevauchements des sommets. Tout cela est combiné à un système d'interaction permettant d'ouvrir/fermer des communautés et d'accéder aux pages *Internet* à partir de ces communautés de mots clés.

La principale limite de notre algorithme réside dans la méthode de fragmentation. Celle-ci est basée sur l'indice *betweenness* dont la complexité de calcul est trop élevée pour rendre le système opérant. C'est pourquoi nous avons entrepris une étude comparative des indices de graphes afin de déterminer s'il en existe un plus rapide à calculer et dont les résultats sont corrélés avec ceux obtenus grâce à l'indice *betweenness*. Ce travail est actuellement en cours de réalisation avec Guy Melançon.

Le second travail présenté dans le deuxième chapitre porte sur la visualisation d'une classe de graphes très utilisée : les arbres. Le système proposé est basé sur la métaphore des cartes géographiques. En plaçant les feuilles d'un arbre le long d'une courbe de Gosper, nous arrivons à créer une hiérarchie de régions dont l'aspect est « géographique ». Le principal atout de cette méthode réside dans la stabilité de la carte dans le cas d'arbres dynamiques.

L'idée d'utiliser la métaphore géographique a été motivée par une hypothèse selon laquelle les individus utiliseraient ainsi certaines capacités cognitives développées lors de l'utilisation de cartes classiques afin de mieux aborder la lecture de notre arbre. Cependant, nous n'avons pas mené d'étude utilisateur prouvant cette assertion et ceci constitue donc le principal travail futur à mener concernant nos *Geographical Treemaps*. Une autre perspective de recherche nous est donnée par l'algorithme permettant d'afficher les étiquettes à l'intérieur de régions concaves. Nous avons proposé une solution à ce problème dont les résultats sont plutôt encourageants. Cependant, le calcul est long et il serait intéressant de se pencher plus en avant sur ce problème afin d'améliorer la vitesse de l'algorithme. Il serait aussi intéressant de réfléchir sur une solution permettant d'afficher les étiquettes le long d'axes monotones afin de supprimer les différents points d'inflexion qui peuvent apparaître.

Le troisième chapitre de cette thèse est consacré à la visualisation d'hypergraphes et en particulier à l'utilisation de graphes associés nommés supports qui permettent d'obtenir des diagrammes d'Euler « bien formés ». Dans un premier temps, nous nous sommes

intéressés aux hypergraphes planaires. Nous avons montré que l'on peut déterminer en temps polynomial si un hypergraphe possède un support qui est un cactus. Nous avons aussi prouvé que l'on peut savoir en temps polynomial si un hypergraphe à intersections et différences fermées possède un support planaire. Cependant, le problème de déterminer s'il existe un support planaire pour un hypergraphe quelconque reste ouvert et constitue donc le principal objectif pour de futurs travaux sur le sujet.

Dans un second temps, nous nous sommes intéressés aux supports basés sur des chemins afin de proposer une visualisation basée sur la métaphore des cartes de métro. Nous avons d'abord montré que trouver un tel support de taille minimale d'une part, et planaire d'autre part, sont des problèmes NP-complets. Puis, nous avons proposé l'ébauche d'une solution permettant de trouver et visualiser de tels supports. Cette méthode est cependant encore dans un état embryonnaire. Nous allons donc devoir améliorer le processus puis mener des études auprès d'utilisateurs afin de déterminer si oui ou non la métaphore utilisée permet de résoudre certains problèmes plus facilement que ses concurrentes.

Bibliographie

[1] Basak Alper, Nathalie Henry Riche, Gonzalo Ramos, and Mary Czerwinski. Design study of linesets, a novel set visualization technique. *To appear in IEEE Transactions on Visualization and Computer Graphics*, 2011.

[2] Elske Ammenwerth. Can evaluation studies benefit from triangulation ? a case study. *International Journal of Medical Informatics*, 70(2-3) :237–248, 2003.

[3] Anaximandre. *Fragments et témoignages*. Presses Universitaires de France - PUF, 1991.

[4] Keith Andrews and Helmut Heidegger. Information slices : Visualising and exploring large hierarchies using cascading, semi-circular discs. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'98)*, pages 9–12, 1998.

[5] Cliff Atkinson. Five experts dispute Edward Tufte on powerpoint. *http ://www.sociablemedia.com/PDF/cliff_atkinson_dispute.pdf*, 2004.

[6] David Auber. Tulip - a huge graph visualization framework. In Petra Mutzel and Mickael Jünger, editors, *Graph Drawing Software*, Mathematics and Visualization Series. Springer Verlag, 2003.

[7] David Auber, Charles Huet, Antoine Lambert, and Arnaud Sallaberry. Geographical treemaps. *Not yet published*.

[8] Anne Aula. Enhancing the readability of search result summaries. In *Proceedings of the HCI 2004 : Designing for Life*, volume 2, pages 1–4, 2004.

[9] Sylvain Auroux, Jacques Deschamps, and Djamel Kouloughli. *La philosophie du langage*. Presses Universitaires de France - PUF - Quadriges Manuels, 2004.

[10] Christian Bachmaier. A radial adaptation of the sugiyama framework for visualizing hierarchical information. *IEEE Transactions on Visualization and Computer Graphics*, 13(3) :583–594, 2007.

[11] Michael Balzer, Olivier Deussen, and Claus Lewerentz. Voronoï treemaps for the visualization of software metrics. In *Proceedings of the 2005 ACM symposium on Software visualization (SoftVis'05)*, pages 165–172, 2005.

[12] Albert-László Barabaśi and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439) :509–512, 1999.

[13] Todd Barlow and Padraic Neville. A comparison of 2-d visualizations of hierarchies. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'01)*, pages 131–138, 2001.

[14] Vladimir Batagelj and Matjaz Zaversnik. Generalized cores. *CoRR*, cs.DS/0202039, 2002.

[15] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing : Algorithms for the Visualization of Graphs.* Prentive Hall, 1999.

[16] Michael Baur and Ulrik Brandes. Multi-circular layout of micro/macro graphs. In *Proceedings of the Conference on Graph Drawing (GD'07)*, volume 4875 of *Lecture Notes in Computer Science*, pages 255–267. Springer, 2007.

[17] Benjamin B. Bederson and Ben Shneiderman. *The Craft of Information Visualization : Readings and reflections.* Morgan Kaufmann, 2003.

[18] Benjamin B. Bederson, Ben Shneiderman, and Martin Wattenberg. Ordered and quantum treemaps : Making effective use of 2d space to display hierarchies. *ACM Transactions on Graphics (TOG)*, 21(4) :833–854, 2002.

[19] Catriel Beeri, Ronald Fagin, David Maier, and Mihalis Yannakakis. On the desirability of acyclic database schemes. *Journal of the ACM*, 30(3) :479–513, 1983.

[20] Claude Berge. *Graphes et Hypergraphes.* Monographies Universitaires de Mathématiques, n° 37. Dunod, 1970.

[21] Claude Berge. *Graphs and Hypergraphs*, volume 6. North-Holland Mathematical Library. Translated by Edward Minieka, 1973.

[22] Jacques Bertin. *Semiology of Graphics. Diagrams, Networks, Maps.* University of Wisconsin Press, 1983.

[23] Jacques Bertin. *Sémiologie Graphique. Les diagrammes, les réseaux, les cartes.* EHESS, 3e édition, (1e éd. Gauthier-Villars 1967, 2e éd. 1973) 1999.

[24] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph Theory.* Springer, Graduate Texts in Mathematics, 2008.

[25] Nicolas Bonnel, Vincent Lemaire, Alexandre Cotarmanac'h, and Annie Morin. Effective organization and visualization of web search results. In *Proceedings of the 24th IASTED international conference on Internet and multimedia systems and applications (IMSA'06)*, pages 209–216. ACTA Press, 2006.

[26] Kellogg S. Booth and George S. Lueker. Testing for the consecutives ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13 :335–379, 1976.

[27] François Boutin and Mountaz Hascoët. Cluster validity indices for graph partitioning. In *Proceedings of the 8th International Conference Information Visualisation (IV'04)*, pages 376–381, 2004.

[28] Ulrik Brandes. Drawing on physical analogies. In M. Kaufmann and D. Wagner, editors, *Drawing Graphs : Methods and Models*, volume 2025 of *Lecture Notes in Computer Science*, pages 71–86. Springer edition, 2001.

[29] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2) :163–177, 2001.

[30] Ulrik Brandes, Sabine Cornelsen, Barbara Pampel, and Arnaud Sallaberry. Blocks of hypergraphs applied to hypergraphs and outerplanarity. In *Proceedings of the 21th International Workshop on Combinatorial Algorithms (IWOCA 2010)*, volume 6460 of *Lecture Notes in Computer Science*, pages 201–211. Springer, 2010.

[31] Ulrik Brandes, Sabine Cornelsen, Barbara Pampel, and Arnaud Sallaberry. Path-based supports for hypergraphs. In *Proceedings of the 21th International Workshop on Combinatorial Algorithms (IWOCA 2010)*, volume 6460 of *Lecture Notes in Computer Science*, pages 20–33. Springer, 2010.

[32] Ulrik Brandes and Thomas Erlebach, editors. *Network Analysis*, volume 3418 of *Lecture Notes in Computer Science*. Springer, 2005.

[33] Ulrik Brandes and Christian Pich. An experimental study on distance-based graph drawing. In *Proceedings of the 16th Conference on Graph Drawing (GD'08)*, volume 5417 of *Lecture Notes in Computer Science*, pages 218–229. Springer, 2009.

[34] Ingwer Brog and Patrick J. F. Groenen. *Modern multidimensional scaling : Theory and applications*. Springer-Verlag, 1997.

[35] Mark Bruls, Kees Huizing, and Jarke J. van Wijk. Squarified treemaps. In *Proceedings Joint Eurographics/IEEE TVCG Symposium Visualization, VisSym*, pages 33–42, 2000.

[36] Kevin Buchin, Marc J. van Kreveld, Henk Meijer, Bettina Speckmann, and Kevin Verbeek. On planar supports for hypergraphs. Technical Report UU-CS–009-035, Utrecht University - Department of Information and Computing Sciences, 2009.

[37] Kevin Buchin, Marc J. van Kreveld, Henk Meijer, Bettina Speckmann, and Kevin Verbeek. On planar supports for hypergraphs. In *Proceedings of the 17th Conference on Graph Drawing (GD'09)*, volume 5849 of *Lecture Notes in Computer Science*, pages 218–229. Springer, 2010.

[38] Heorhiy Byelas and Alexandru Telea. Visualization of areas of interest in software architecture diagrams. In *Proceedings of the 2006 ACM symposium on Software visualization (SoftVis'06)*, pages 105–114, 2006.

[39] Heorhiy Byelas and Alexandru Telea. Towards realism in drawing areas of interest on architecture diagrams. *Journal of Visual Languages and Computing*, 20(2) :110–128, 2009.

[40] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in Information Visualization : Using Vision to Think*. Morgan Kaufmann, 1st edition, 1999.

[41] Rudolf Carnap, Hans Hahn, Otto Neurath, Moritz Schlick, and Friedrich Waissman. *Manifeste du Cercle de Vienne et autres écrits*. Vrin, (1e éd. allemande : *Wissenschaftliche Weltauffassung*, Der Wiener Kreis, Artur Wolf Verlag, 1929) 2010.

[42] Gary Chartrand. *Introductory Graph Theory*. Dover Publications Inc., abridged edition, 1985.

[43] Chaomei Chen. *Information Visualization : Beyond the Horizon.* Springer-Verlag, 2nd edition, 2006.

[44] Fanny Chevalier, Stéphane Huot, and Jean-Daniel Fekete. Wikipediaviz : Conveying article quality for casual wikipedia readers. In *Proceedings of the 3rd IEEE Pacific Visualization Symposium (PacificVis 2010)*, pages 49–56, 2010.

[45] Markus Chimani and Carsten Gutwenger. Algorithms for the hypergraph and the minor crossing number problems. In *Proceedings of the 18th International Symposium on Algorithms and Computing (ISAAC 2007)*, volume 4835 of *Lecture Notes in Computer Science*, pages 184–195. Springer, 2007.

[46] Francis Y. L. Chin, Jack Snoeyink, and Cao An Wang. Finding the medial axis of a simple polygon in linear time. *Discrete & Computational Geometry*, 21(3) :405–420, 1999.

[47] Noam Chomsky. *Structures syntaxiques.* Seuil, (1e éd. anglaise : *Syntactic Structures*, Mouton & Co, 1957) 1969.

[48] Stirling C. Chow. *Generating and drawing area-proportional Euler and Venn diagrams.* PhD thesis, University of Victoria, British Colombia Canada, 2007.

[49] Christopher Collins, Gerald Penn, and M. Sheelagh T. Carpendale. Bubble sets : Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6) :1009–1016, 2009.

[50] Valérie Corman and Eric Ingargiola. Guide pratique : intelligence économique et PME. 2005.

[51] Michel Crampes, Jeremy de Oliveira-Kumar, Sylvie Ranwez, and Jean Villerd. Visualizing social photos on a hasse diagram for eliciting relations and indexing new photos. *IEEE Transactions on Visualization and Computer Graphics*, 15(6) :985–992, 2009.

[52] Jan de Leeuw. Applications of convex analysis to multidimensional scaling. In *Recent Developments in Statistics*, pages 133–145, 1977.

[53] Jan de Leeuw. Convergence of the majorization method for multidimensional scaling. *Journal of Classification*, 5(2) :163–180, 1988.

[54] Ferdinand de Saussure. *Cours de linguistique générale.* Payot, (1e éd. 1916) 1995.

[55] Walter Didimo, Francesco Giordano, and Giuseppe Liotta. Overlapping cluster planarity. *Journal of Graph Algorithms and Applications*, 12(3) :267–291, 2008.

[56] Emile Durkheim. *Les règles de la méthode sociologique.* Presses Universitaires de France - PUF - Quadrige Grands textes, (1e éd. 1937) 2007.

[57] Peter Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42 :149–160, 1984.

[58] Peter Eades. Drawing free trees. *Bulletin of the Institute for Combinatorics and its Applications*, 5 :10–36, 1992.

[59] Umberto Eco. *Lector in fabula.* Grasset, (1e éd. italienne : *Lector in fabula*, Bompiani, 1984) 1985.

[60] Umberto Eco. *Œuvre ouverte.* Seuil, (1e éd. italienne : *Opera aperta*, Bompiani, 1962) 1965.

[61] Umberto Eco. *Le Signe, Histoire et analyse d'un concept.* Labor, (1e éd. italienne : *Segno*, Arnoldo Mondadori, 1980) 1988.

[62] Umberto Eco. *Sémiotique et philosophie du langage.* Presses Universitaires de France - PUF - Quadrige Grands textes, (1e éd. italienne : *Semiotica e filosofia del linguaggio*, Guilio Einaudi, 1984. 1e éd. française, 1988) 2006.

[63] Umberto Eco. *La production des signes.* Librairie Générale Française, (Extrait français de l'ouvrage italien : *Trattato di semiotica generale*, Bompiani, 1975) 1992.

[64] Paul Erdös and Alfréd Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6 :290–291, 1959.

[65] Thomas Eschbach, Wolfgang Günther, and Bernd Becker. Orthogonal hypergraph drawing for improved visibility. *Journal on Graph Algorithms and Applications*, 10(2) :141–157, 2006.

[66] Michael Esfeld. *Philosophie des sciences : Une introduction.* Presses polytechniques et universitaires normandes, 2ième édition, 2009.

[67] Leonhard Euler. *Lettres à une Princesse d'Allemagne.* Charpentier, 1843.

[68] Sara I. Fabrikant and André Skupin. Cognitively plausible information visualization. In J. Dykes and A. M. MacEachren nad M.-J. Kraak, editors, *Exploring Geovisualization*, pages 667–682. Elsevier ltd. edition, 2005.

[69] Jean-Daniel Fekete and Catherine Plaisant. Excentric labeling : Dynamic neighborhood labeling for data visualization. In *Proceedings of the ACM International Conference on Human-computer Interaction (CHI 1999)*, pages 512–519, 1999.

[70] Jean Flower, Andrew Fish, and John Howse. Euler diagram generation. *Journal of Visual Languages and Computing*, 19(6) :675–694, 2008.

[71] Linton C. Freeman. A set of measures of centrality based upon betweeness. *Sociometry*, 40 :35–41, 1977.

[72] Arne Frick, Andreas Ludwig, and Heiko Mehldau. A fast adaptive layout algorithm for undirected graphs. In *Proceedings of the DIMACS International Workshop on Graph Drawing (GD '94)*, pages 388–403. Springer-Verlag, 1994.

[73] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software - Practice And Experience*, 21(11) :1129–1164, 1991.

[74] George W. Furnas. Generalized fisheye views. *ACM SIGCHI Bulletin*, 17 :16–23, 2008.

[75] Pawel Gajer and Stephen G. Kobourov. GRIP : Graph dRawing with Intelligent Placement. In *Proceedings of the Conference on Graph Drawing (GD'00)*, volume 1984 of *Lecture Notes in Computer Science*, pages 222–228. Springer, 2000.

[76] Emden R. Gansner and Yifan Hu. Efficient node overlap removal using a proximity stress model. In *Proceedings of the Conference on Graph Drawing (GD'08)*, volume 5417 of *Lecture Notes in Computer Science*, pages 206–217. Springer, 2008.

[77] Emden R. Gansner, Yifan Hu, and Stephen G. Kobourov. Gmap : Drawing graphs as maps. In *Proceedings of the Conference on Graph Drawing (GD'09)*, volume 5849 of *Lecture Notes in Computer Science*, pages 405–407. Springer, 2009.

[78] Emden R. Gansner, Yifan Hu, Stephen G. Kobourov, and Chris Volinsky. Putting recommendations on the map : visualizing clusters and relations. In *Proceedings of the ACM Conference on Recommender Systems (RecSys 2009)*, pages 345–348, 2009.

[79] Emden R. Gansner, Yehuda Koren, and Stephen North. Graph drawing by stress majorization. In *Proceedings of the Conference on Graph Drawing (GD'04)*, volume 3383 of *Lecture Notes in Computer Science*, pages 285–295. Springer, 2004.

[80] Michael R. Garey and David S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-completeness.* A Series of Books in the Mathematical Sciences, W. H. Freeman and Company, 1979.

[81] Michael R. Garey, David S. Johnson, and Larry Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1 :237–267, 1976.

[82] Michelle Girvan and Mark E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 99(12) :7821–7826, 2002.

[83] Gene H. Golub and Charles F. van Loan. *Matrix Computations.* The Johns Hopkins University Press, 3rd edition, 1996.

[84] Jack Goody. *La raison graphique. La domestication de la pensée sauvage.* Minuit - Le sens commun, (1e éd. anglaise : *The domestication of the savage mind*, Cambridge University Press, 1977) 1979.

[85] Martin Graham and Jessie B. Kennedy. A survey of multiple tree visualisation. *Information Visualization (IVS)*, 9(4) :235–252, 2010.

[86] Algirdas-Julien Greimas and Joseph Courtès. *Sémiotique. Dictionnaire raisonné de la théorie du langage.* Hachette université, Série Langage, Linguistique, Communication, 1979.

[87] Paul Grice. Logic and conversation. In P. Cole and J. Morgan, editors, *Syntax and semantics*, volume 3. New york : Academic press edition, 1975.

[88] Sébastien Grivet, David Auber, Jean-Philippe Domenger, and Guy Melancon. Bubble tree drawing algorithm. In Springer Verlag, editor, *International Conference on Computer Vision and Graphics*, pages 633–641, 2004.

[89] Jean-Loup Guillaume and Matthieu Latapy. Bipartite graphs as models of complex networks. In *Workshop on Combinatorial and Algorithmic Aspects of Networking (CAAN)*, volume 1 of *Lecture Notes in Computer Science*, pages 127–139. Springer, 2004.

[90] Stefan Hachul and Michael Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In *Proceedings of the Conference on Graph Drawing (GD'04)*, volume 3383 of *Lecture Notes in Computer Science*, pages 285–295. Springer, 2004.

[91] Stefan Hachul and Michael Jünger. An experimental comparison of fast algorithms for drawing general large graphs. In *Proceedings of the Conference on Graph Drawing (GD'05)*, volume 3843 of *Lecture Notes in Computer Science*, pages 235–250. Springer, 2005.

[92] Herman J. Haverkort and Freek van Walderveen. Locality and bounding-box quality of two-dimensional space-filling curves. *Computational Geometry, Theory and Applications*, 43(2) :131–147, 2010.

[93] Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin. Nodetrix : a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6) :1302–1309, 2007.

[94] Ivan Herman, Guy Melançon, and M. Scott Marshall. Graph visualization and navigation in information visualization : A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1) :24–43, 2000.

[95] Louis Hjelmslev. *Le langage.* Minuit, (1e éd. danoise : *Sproget*, Berlingske Forlag, 1963) 1966.

[96] John Hopcroft and Robert Tarjan. Efficient planarity testing. *Journal of the Association for Computing Mashinery*, 21(4) :549–568, 1974.

[97] Yifan Hu, Emden R. Gansner, and Stephen G. Kobourov. Visualizing graphs and clusters as maps. *IEEE Computer Graphics and Applications*, 30(6) :54–66, 2010.

[98] Anne Hénault. *Histoire de la sémiotique.* Presses Universitaires de France - PUF - Que sais-je ?, 1992.

[99] Anne Hénault and Anne Beyaert. *Ateliers de sémiotique visuelle.* Presses Universitaires de France - PUF - Formes sémiotiques, 2004.

[100] Ramon Ferrer i Cancho and Ricard V. Solé. The small world of human language. *Proceedings of the Royal Society of London*, B268(1482) :2261–2265, 2001.

[101] Roman Ossipovitch Jakobson. *Eléments de linguistique générale (tomes 1 et 2).* Éditions de Minuit, Collection Double, 1981.

[102] William James. *Le pragmatisme.* Flammarion - Champs classiques, (1e éd. anglaise : *Pragmatism : A New Name for Some Old Ways of Thinking*, Longmans, Green & Co, 1907) 2007.

[103] Yuntao Jia, Jared Hoberock, Michael Garland, and John C. Hart. On the visualization of social and other scale-free networks. *IEEE Transactions on Visualization and Computer Graphics*, 14(6) :1285–1292, 2008.

[104] Brian Johnson and Ben Shneiderman. Tree maps : A space-filling approach to the visualization of hierarchical information structures. In *IEEE Visualization*, pages 284–291, 1991.

[105] David S. Johnson, Shankar Krishnan, Jatin Chhugani, Subodh Kumar, and Suresh Venkatasubramanian. Compressing large boolean matrices using reordering techniques. In *Proceedings of the 13th International Conference on Very Large Data Bases (VLDB'04)*, pages 13–23, 2004.

[106] David S. Johnson and Henry O. Pollack. Hypergraph planarity and the complexity of drawing venn diagrams. *Journal of Graph Theory*, 11(3) :309–325, 1987.

[107] Tomihisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31 :7–15, 1989.

[108] Jerrold J. Katz. *Propositional Structure and Illocationary Force : A Study of the Contribution of Sentence Meaning to Speech Acts.* Harvester Press, Hassocks, 1977.

[109] Michael Kaufmann, Marc J. van Kreveld, and Bettina Speckmann. Subdivision drawings of hypergraphs. In *Proceedings of the Conference on Graph Drawing (GD'08)*, volume 5417 of *Lecture Notes in Computer Science*, pages 396–407. Springer, 2009.

[110] Michael Kaufmann and Dorothea Wagner. *Drawing Graph : Methods and Models.* Springer, Lecture Notes in Computer Science 2025, 2001.

[111] Daniel A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1) :1–8, 2002.

[112] Daniel A. Keim, Stephen C. North, and Christian Panse. Cartodraw : A fast algorithm for generating contiguous cartograms. *IEEE Transactions on Visualization and Computer Graphics*, 10(1) :95–110, 2004.

[113] Michael D. Kickmeier and Dietrich Albert. The effects of scannability on information search : An online experiment. In *Proceedings of the HCI 2003 : Designing for Society*, volume 2, pages 33–36, 2003.

[114] Jean-Marie Klinkenberg. *Précis de sémiotique générale.* Le Seuil, (1e éd. Duculot 1997) 2000.

[115] Teuvo Kohonen. *Self-Organizing Maps.* Springer-Verlag, 1995.

[116] Ephraim Korach and Michal Stern. The clustering matroid and the optimal clustering tree. *Mathematical Programming, Series B*, 98(1–3) :385–414, 2003.

[117] Yehuda Koren and David Harel. A multi-scale algorithm for the linear arrangement problem. In *Proceedings of the 28th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'2002)*, volume 2573 of *Lecture Notes in Computer Science*, pages 296–309. Springer, 2002.

[118] Daniel Král, Jan Kratochvíl, and Heinz-Jürgen Voss. Mixed hypercacti. *Discrete Mathematics*, 286(1-2) :99–113, 2004.

[119] Joseph B. Kruskal and James M. Landwehr. Icicle plots : Better displays for hierarchical clustering. *The American Statistician*, 37(2) :162–168, 1983.

[120] Wolfgang Köhler. *Psychologie de la forme.* Gallimard (collection "idées"), (1e éd. allemande : *Gestalt Psychology*, 1929) 1964.

[121] André Lalande. *Vocabulaire technique et critique de la philosophie.* Presses Universitaires de France - PUF - Dicos poche, (1e éd. dans Bulletin de la société française de philosophie, 1902-1923) 2010.

[122] Anton Leuski and James Allan. Lighthouse : showing the way to relevant information. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'00)*, pages 125–129, 2000.

[123] Iman Y. Liao, Maria Petrou, and Rongchun Zhao. A fractal-based relaxation algorithm for shape from terrain image. *Computer Vision Image Understanding*, 109(3) :227–243, 2008.

[124] Zhicheng Liu and John T. Stasko. Mental models, visual reasoning and interaction in information visualization : A top-down perspective. *IEEE Transactions on Visualization and Computer Graphics*, 16(6) :999–1008, 2010.

[125] Benoît B. Mandelbrot. *The Fractal Geometry of Nature.* W. H. Freedman and Co., 1983.

[126] Riccardo Mazza. *Introduction to Information Visualization.* Springer-Verlag, 2009.

[127] Guy Melançon and Arnaud Sallaberry. Edge metrics for visual graph analytics : A comparative study. In *Proceedings of the 12th International Conference Information Visualisation (IV'08)*, pages 610–615, 2008.

[128] Charles Morris. *Foundations of the Theory of signs.* Chicago University Press, 1938.

[129] Groupe μ. *Traité du signe visuel : pour une rhétorique de l'image.* Seuil, 1992.

[130] Chris Muelder and Kwan-Liu Ma. Rapid graph layout using space filling curves. *IEEE Transactions on Visualization and Computer Graphics*, 6(14) :1301–1308, 2008.

[131] Tamara Munzner. A nested process model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6) :921–928, 2009.

[132] William M. Murrell. Map on temperance. *Howe's Sheet Anchor Press*, 1846.

[133] Paul Mutton, Peter Rodgers, and Jean Flower. Drawing graphs in euler diagrams. In *Proceedings of the 3rd International Conference on Diagrammatic Representation and Inference (Diagrams 2004)*, volume 2980 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 2004.

[134] Erkki Mäkinen. How to draw a hypergraph. *International Journal of Computer Mathematics*, 34 :177–185, 1990.

[135] Tien Nguyen and Jun Zhang. A novel visualization model for web search results. *IEEE Transactions on Visualization and Computer Graphics*, 12(5) :981–988, 2006.

[136] Jordi Nin, Paola Salle, Sandra Bringay, Mathieu Roche, and Maguelonne Teisseire. Using owa operators for gene sequential pattern clustering. In *22nd IEEE International Symposium on Computer-Based Medical Systems (CBMS'09)*, page 6, 2009.

[137] Martin Nöllenburg and Alexander Wolff. Drawing and labeling high-quality metro maps by mixed-integer programming. *IEEE Transactions on Visualization and Computer Graphics*, 17(5) :626–641, 2011.

[138] Bruno Ollivier. *Les sciences de la communication : Théories et acquis.* Collection U, Armand Colin, 2007.

[139] Charles Sanders Peirce. *Ecrits sur les signes.* Le Seuil, 1978.

[140] Jordi Petit. Experiments on the minimum linear arrangement problem. *ACM Journal of Experimental Algorithmics*, 8, 2003.

[141] Christian Pich. *Applications of Multidimensional Scaling to Graph Drawing.* PhD thesis, Universität Konstanz, 2009.

[142] Didier Poidevin. *La carte : moyen d'action.* Ellipses, 1999.

[143] Karl Popper. *La logique de la découverte scientifique.* Payot, (1e éd. allemande : *Logik der Forschung*, 1935) 2007.

[144] Willard Van Orman Quine. *Du point de vue logique : Neuf essais logico-philosophiques.* Vrin, (1e éd. anglaise : *From a logical point of view*, Harvard University Press, 1953) 2004.

[145] Erwin Raisz. The rectangular statistical cartogram. *Geographical Review*, 24(2) :292–296, 1934.

[146] Edward M. Reingold and John S. Tilford. Tidier drawings of trees. *IEEE Transactions on Software Engeneering*, 7(2) :223–228, 1981.

[147] Nathalie Henry Riche and Tim Dwyer. Untangling euler diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 16(6) :1090–1099, 2010.

[148] Frank Ruskey and Mark Weston. A survey of venn diagrams. *The electronic journal of combinatorics*, 2005.

[149] Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche, and Maguelonne Teisseire. Discovering novelty in gene data : From sequential patterns to visualization. In *Proceedings of the 6th International Symposium on Visual Computing (ISVC10)*, volume 6455 of *Lecture Notes in Computer Science*, pages 534–543. Part III, Springer-Verlag Berlin Heidelberg, 2010.

[150] Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche, and Maguelonne Teisseire. Sequencesviewer : comment rendre accessible des motifs séquentiels de gènes trop nombreux ? In *10ième Conférence Internationale Francophone sur l'Extraction des Connaissances (EGC 2010)*, volume E-19 of *Revue des Nouvelles Technologies de l'Information*, pages 387–392, 2010.

[151] Arnaud Sallaberry, Nicolas Pecheur, Sandra Bringay, Mathieu Roche, and Maguelonne Teisseire. Sequential patterns mining and gene sequence visualization to discover novelty from microarray data. *Journal of Biomedical Informatics*, 44(5) :760–774, 2011.

[152] Arnaud Sallaberry, Faraz Zaidi, Christian Pich, and Guy Melançon. Interactive visualization and navigation of web search results revealing community structures and bridges. In *Proceedings of the 36th Graphics Interface conference (GI'10)*, pages 105–112, 2010.

[153] Paola Salle, Sandra Bringay, and Maguelonne Teisseire. Mining discriminant sequential patterns for aging brain. In *Proceedings of the 12th conference on Artificial Intelligence in Medicine (AIME'09)*, pages 365–369, 2009.

[154] Georg Sander. Layout of directed hypergraphs with orthogonal hyperedges. In *Proceedings of the 11th International Symposium on Graph Drawing (GD'03)*, volume 2912 of *Lecture Notes in Computer Science*, pages 381–386. Springer, 2004.

[155] Hassan Saneifar, Sandra Bringay, Anne Laurent, and Maguelonne Teisseire. S2mp : Similarity measure for sequential patterns. In *Proceedings of the Australian Data Mining Conference (AusDM08)*, pages 95–104, 2008.

[156] Gilbert Saporta. *Probabilités, analyse des données et Statitique.* Technip, (1e éd. 1990) 2006.

[157] Purvi Saraiya, Chris North, and Karen Duca. An insight-based methodology for evaluating bioinformatics visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 11(4) :443–456, 2005.

[158] Manojit Sarkar and Marc H. Brown. Graphical fisheye views. *Communication ACM*, 37(12) :73–83, 1994.

[159] Marcus Schaefer and Daniel Stefankovic. Decidability of string graphs. *Journal of Computer and System Sciences*, 68(2) :319–334, 2004.

[160] Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, 1(1) :27–64, 2007.

[161] Stephen B. Seidman. Network structure and minimum degree. *Social Networks*, 5 :269–287, 1983.

[162] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27 :379–423, 623–656, 1948.

[163] Paolo Simonetto and David Auber. Visualise undrawable euler diagrams. In *Proceedings of the 12th International Conference on Information Visualisation (IV08)*, pages 594–599, 2008.

[164] Paolo Simonetto and David Auber. An heuristic for the construction of intersection graphs. In *Proceedings of the 13th International Conference on Information Visualisation (IV09)*, pages 673–678, 2009.

[165] Paolo Simonetto, David Auber, and Daniel Archambault. Fully automatic visualisation of overlapping sets. *Computer Graphics Forum*, 28(3) :967–974, 2009.

[166] André Skupin. From metaphor to method : Cartographic perspectives on information visualization. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'00)*, pages 91–98, 2000.

[167] André Skupin. A cartographic approach to visualizing conference abstracts. *IEEE Computer Graphics and Applications (CGA)*, 22(1) :50–58, 2002.

[168] Robert Spence. *Information Visualization : Design for Interaction.* Pearson - Prentice Hall (2nd edition), 2007.

[169] Dan Sperber and Deirde Wilson. *La Pertinance : Communication et Cognition.* Les éditions de minuit, (1e éd. anglaise : *Relevance : Communication and cognition*, Wiley-Blackwell, 1986) 1989.

[170] John T. Stasko and Eugene Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'00)*, pages 57–65, 2000.

[171] Jonathan M. Stott, Peter Rodgers, Juan Carlos Martinez-Ovando, and Stephen G. Walker. Automatic metro map layout using multicriteria optimization. *IEEE Transactions on Visualization and Computer Graphics*, 17(1) :101–114, 2011.

[172] Ilija Subašić and Bettina Berendt. Web mining for understanding stories through graph visualisation. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*, pages 365–369, 2008.

[173] Robert E. Tarjan and Mihalis Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3) :566–579, 1984.

[174] Alexandru C. Telea. *Data Visualization : Principles and Practice.* A K Peters, Ltd., 2008.

[175] Waldo Tobler. Pseudo-cartograms. *The American Cartographer*, 13 :43–50, 1986.

[176] Warren S. Torgerson. Multidimensional scaling : I. theory and method. *Psychometrika*, 17(4) :401–419, 1952.

[177] Melanie Tory and Torsten Moeller. A model-based visualization taxonomy. Technical Report CMPT-TR2002-06, Computer Science Department, Simon Freaser University, 2002.

[178] Edward R. Tufte. *The Visual Display of Quantitive Information.* Graphics Press, 1983.

[179] Edward R. Tufte. *Envisionning Information.* Graphics Press, 1990.

[180] Edward R. Tufte. *Visual Explanations.* Graphics Press, 1997.

[181] Edward R. Tufte. *The Cognitive Style of PowerPoint.* Graphics Press, 2003.

[182] William T. Tutte. How to draw a graph. *Proceedings of the London Mathematical Society*, 13 :743–768, 1963.

[183] William M. van Cleemput. On the planarity of hypergraphs. *Proceedings of the IEEE*, 66(4) :514–515, 1978.

[184] Marc J. van Kreveld and Bettina Speckmann. On rectangular cartograms. *Computational Geometry, Theory and Applications*, 37(3) :175–187, 2007.

[185] Robert van Liere and Wim C. de Leeuw. Graphsplatting : Visualizing graphs as continuous fields. *IEEE Transactions on Visualization and Computer Graphics*, 9(2) :206–212, 2003.

[186] Jark J. van Wijk and Wim A. A. Nuij. Smooth and efficient zooming and panning. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'03)*, pages 15–23, 2003.

[187] Jark J. van Wijk and Huub van de Wetering. Cushion treemaps : Visualization of hierarchical information. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'99)*, pages 73–78, 1999.

[188] John Venn. On the diagrammatic and mechanical representation of propositions and reasonings. *Philosophical Magazine and Journal of Science*, 10(59), 1880.

[189] Jean-Pierre Vernant. *Les origines de la pensée grecque.* Presses Universitaires de France - PUF - Quadrige Grands textes, (1e éd. PUF 1962) 2009.

[190] Anne Verroust and Marie-Luce Viaud. Ensuring the drawability of extended euler diagrams for up to 8 sets. In *Proceedings of the 3rd International Conference of Diagrammatic Representation and Inference (Diagrams 2004)*, volume 2980 of *Lecture Notes in Computer Science*, pages 128–141. Springer, 2004.

[191] Roel Vliegen, Jarke J. van Wijk, , and Erik-Jan van der Linden. Visualizing business data with generalized treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 12(5) :789–796, 2006.

[192] T. R. S. Walsh. Hypermaps versus bipartite maps. *Journal of Combinatorial Theory, Series B*, 18(2) :155–163, 1975.

[193] Matthew Ward, Georges Grinstein, and Daniel Keim. *Interactive Data Visualization : Foundations, Techniques, and Applications.* A K Peters, Ltd., 2010.

[194] Colin Ware. *Information visualizatin : Perception for design.* Morgan Kaufmann, 2000.

[195] Martin Wattenberg. A note on space-filling visualizations and space-filling curves. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'05)*, page 24, 2005.

[196] Duncan J. Watts and Steven Henry Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393 :440–442, 1998.

[197] Richard Wettel and Michele Lanza. Visualizing software systems as cities. In *Proceedings of the 4th IEEE International Workshop on Visualizing Software For Understanding and Analysis (VisSoft'07)*, pages 92–99, 2007.

[198] Leland Wilkinson. *The grammar of Graphics.* Springer-Verlag, Statistics and Computing, 1999.

[199] Wesley Willett, Jeffrey Heer, and Maneesh Agrawala. Scented widgets : Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6) :1129–1136, 2007.

[200] James A. Wise, James J. Thomas, Kelly Pennock, David Lantrip, Marc Pottier, Anne Schur, and Vern Crow. Visualizing the non-visual : spatial analysis and interaction with information from text documents. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'95)*, pages 51–58, 1995.

[201] Alexander Wolff. Drawing subway maps : A survey. *Informatik-Forschung und Entwicklung*, 22(1) :23–44, 2007.

[202] Yi-Fang Brook Wu, Latha Shankar, and Xin Chen. Finding more useful information faster from web search results. In *Proceedings of the twelfth international conference on Information and knowledge management (CIKM'03)*, pages 568–571, 2003.

[203] Jing Yang, Wei Peng, Matthew O. Ward, and Elke A. Rundensteiner. Interactive hierarchical dimension ordering, spacing and filtering for exploration of high dimensional datasets. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'03)*, pages 105–112, 2003.

[204] Faraz Zaidi. *Analysis, Structure and Organization of Complex Networks*. PhD thesis, Université Bordeaux 1, 2010.

[205] Faraz Zaidi, Arnaud Sallaberry, and Guy Melançon. Revealing hidden community structures and identifying bridges in complex networks : An application to analyzing contents of web pages for browsing. In *Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence (WI'09)*, pages 198–205, 2009.

[206] Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, and Jinwen Ma. Learning to cluster web search results. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'04)*, pages 210–217, 2004.

[207] A. A. Zykov. Hypergraphs. *Uspekhi Matematicheskikh Nauk*, 29 :6(180) :89–154, 1974.

Visualisation d'information : de la théorie sémiotique à des exemples basés sur la représentation de graphes et d'hypergraphes

Résumé :

La visualisation d'information est une discipline récente en pleine expansion et qui a pour objet l'étude des méthodes de représentation visuelle de données abstraites, c'est-à-dire non géolocalisées. La sémiotique est quant à elle une discipline beaucoup plus ancienne (fin du XIXième siècle) qui s'intéresse aux divers systèmes de signes nécessaires aux processus de communication. À ce jour, peu de travaux ont été réalisés pour mettre en parallèle ces deux disciplines. C'est pourquoi le premier chapitre de cette thèse est dédié à l'étude de la visualisation d'information selon les paradigmes élaborés par son ainée tout au long du XXième siècle. Nous montrons en particulier comment l'un des modèles les plus aboutis de validation de visualisations (modèle imbriqué de Tamara Munzner) correspond au processus d'étude sémiotique d'énoncés. Le second chapitre est consacré à la visualisation de graphe, outil de modélisation puissant de divers ensembles de données abstraites. Nous proposons d'une part une application permettant de visualiser et de naviguer à travers les pages Internet retournées par un moteur de recherche et d'autre part un algorithme de visualisation de hiérarchies dynamiques sous forme de « cartes géographiques ». Enfin, nous évoquons dans le troisième chapitre un autre outil de modélisation de données abstraites : les hypergraphes. Nous proposons des résultats théoriques concernant leur représentation et donnons une ébauche de solution permettant de les visualiser.

Mots-clés : Visualisation d'information, Dessin de graphes et d'hypergraphes, Sémiotique, Sémiologie.

Discipline : Informatique.

Information visualization : from semiotic theory to practical examples based on graphs and hypergraphs representation

Abstract :

Information visualization aims at designing visual representations of abstract data, furthermore relying on interaction as a mean to discover knowledge. The first part of this thesis challenges Information Visualization by drawing a parallel with semiotics, a 19th century research field focusing on systems of signs required for communication. We develop a point of view on Information Visualization based on the paradigms developed by semioticians during the 20th century. In particular, we show how the visualization validation model proposed by Tamara Munzner is related to the process used by semioticians for utterance analysis. The second part of the thesis focuses on graph visualization and describes two techniques and system prototypes targeting specific application domains. The first one is an interactive technique to visualize and navigate through Web search results. The second one is an algorithm for the visualization of dynamic hierarchies exploiting the analogy with "geographical maps". Finally, the third chapter is devoted to another model used to structure abstract data : hypergraphs. We propose theoretical results on hypergraph drawing and a preliminary technique to visualize hypergraphs.

Keywords : Information Visualization, Graph and hypergraph drawing, Semiotic, Semiology.

Field : Computer Science.