THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITÉ DE MONTPELLIER

En Informatique

École doctorale : Information, Structures, Systèmes

Unité de recherche UMR5506

Réduction de l'encombrement visuel : application à la visualisation et à l'exploration de données prosopographiques

Présentée par Fati CHEN
Le 29 juin 2022

Sous la direction de Pascal PONCELET
et Arnaud SALLABERRY

Devant le jury composé de

Mme. Pascale KUNTZ, Professeure, LS2N, Université de Nantes	Rapportrice
M. David AUBER, Professeur, LaBRI, Université Bordeaux	Rapporteur
Mme. Marianne HUCHARD, Professeure, LIRMM, Université de Montpellier	Présidente
M. Francesco BERETTA, Chargé de Recherche CNRS, LARHRA, Université Lumière Lyon 2	Examinateur
M. Cédric du MOUZA, Maître de Conférences HDR, CEDRIC, CNAM Paris	Examinateur
M. Arnaud SALLABERRY, Maître de Conférences HDR, LIRMM, Université Paul Valéry Montpellier	Co-Directeur
M. Pascal PONCELET, Professeur, LIRMM, Université de Montpellier	Directeur

UNIVERSITÉ DE MONTPELLIER

Résumé

La prosopographie est utilisée par les historiens pour désigner des notices biographiques afin d'étudier des caractéristiques communes d'un groupe d'acteurs de l'histoire au moyen d'une analyse collective de leur vie. La visualisation d'informations présente des perspectives intéressantes pour analyser les données prosopographiques. C'est dans ce contexte que se situe le travail présenté dans ce mémoire. Dans un premier temps, nous présentons la plateforme *ProsoVis* conçue pour analyser et naviguer dans des données prosopographiques. Nous décrivons les différents besoins exprimés et détaillons les choix de conception ainsi que les techniques de visualisation employées. Nous illustrons son utilisation avec la base Siprojuris qui contient les données sur la carrière des enseignants de droit de 1800 à 1950. La visualisation d'autant de données pose des problèmes d'encombrement visuel. Dans ce contexte, nous abordons la problématique des chevauchements des nœuds dans un graphe. Différentes approches existent mais il est difficile de les comparer car leurs évaluations ne sont pas basées sur les mêmes critères de qualité. Nous proposons donc une étude de l'état de l'art et comparons les résultats des algorithmes sur une liste homogène de critères. Enfin, nous abordons une autre problématique d'encombrement visuel au sein d'une carte et proposons une approche de regroupement spatial agglomératif, *F-SAC*, beaucoup plus rapide que les propositions de l'état de l'art tout en garantissant la même qualité de résultats.

Mots-clés – Données prosopographiques, visualisation d'information, encombrement visuel, suppression de chevauchement, regroupement spatial agglomératif

Abstract

Prosopography is used by historians to designate biographical records in order to study common characteristics of a group of historical actors through a collective analysis of their lives. Information visualization presents interesting perspectives for analyzing prosopographical data. It is in this context that the work presented in this thesis is situated. First, we present the *ProsoVis* platform to analyze and navigate through prosopographical data. We describe the different needs expressed and detail the design choices as well as the different views. We illustrate its use with the Siprojuris database which contains data on the careers of law teachers from 1800 to 1950. Visualizing so much data induces visual cluttering problems. In this context, we address the problem of overlapping nodes in a graph. Even if approaches exist, it is difficult to compare them because their respective evaluations are not based on the same quality criteria. We therefore propose a study of the state-of-the-art algorithms by comparing their results on the same criteria. Finally, we address a similar problem of visual cluttering within a map and propose a spatial agglomerative clustering approach, *F-SAC*, which is much faster than the state-of-the-art proposals while guaranteeing the same quality of results.

Keywords – Prosopographical Data, Information Visualization, Visual Cluttering, Overlap Removal, Spatial Agglomerative Clustering

REMERCIEMENTS

Tout d'abord, je tiens à remercier ma compagne Julie Ripoll, pour sa présence, son soutien inconditionnel, son aide et ses précieux conseils tout au long du déroulement de la thèse. Ma famille, mes parents Deyan Chen et Fatima Artemieff, mon frère Yusu Chen et ma sœur Maya Chen pour leur soutien et encouragements. Merci à la famille Abellan–Romita, Raphaël, Nathalie, Francis pour votre support et pour m'avoir poussé vers le haut depuis le lycée. C'est grâce à vous tous que je suis ici aujourd'hui et je vous en remercie profondément.

Je remercie mes deux encadrants, Arnaud Sallaberry et Pascal Poncelet pour m'avoir donné l'opportunité de faire cette thèse, la confiance qu'ils m'ont accordé, le temps qu'ils ont investi pour me guider et m'aider tout le long de cette aventure. Un grand merci à vous.

Je souhaite aussi remercier les membres du jury. Pascale Kuntz et David Auber pour avoir accepté de relire ce manuscript et pour toutes les remarques pertinentes. Marianne Huchard qui m'a également accompagné tout le long de ma thèse. Cédric du Mouza et Francesco Beretta pour avoir accepté d'évaluer mon travail. Dino Ienco pour m'avoir suivi durant cette thèse.

Je tiens aussi à remercier l'Université de Montpellier (UM), le Laboratoire d'Informatique, de Robotique et de Microéléctronique de Montpellier (LIRMM) et l'équipe ADVANSE pour m'avoir accueilli, l'ANR DAPHNE pour avoir financé ces travaux.

Je remercie aussi tous les membres de mon équipe, passé et présent, ceux des équipes WEB^3, MAB, GraphiK, le secrétariat du LIRMM, et plus généralement toutes ces merveilleures personnes, permanents, post-doctorants, doctorants, stagiaires que j'ai eu la chance de rencontrer et qui m'ont beaucoup apporté autant pour les moments qu'on a partagé que les connaissances qu'elles m'ont transmises.

Merci notamment à Arnaud Castelltort, Jérome Azé, Maximilien Servajean, Sandra Bringay, Nancy Rodriguez, Morgan Soulié, Alexis Delaforge, Hugo le Baher, Leonardo Moros, Laëtitia Viau, Samy Benslimane, Vincent Raveneau, Waleed Ragheb, Samiha Fadloun, Erick Cuenca, Constantin Todorov, Clement Jonquet, Andon Tchechmedjiev, Faaiz Hussain Shah, Martin Jedwabny, Bruno Yun, Abdelraouf Hecham, Sylvain Milanesi, Nikolaï Romashchenko, Benjamin Linard, Vincent Lefort, Marie Mille, Sylvain Pulicani, Johannes Wirtz, Raphaël Romero et Virginie Feche pour toutes les discussions intéressantes et moins intéressantes.

Enfin, je remercie tous mes amis pour leur soutien et leur écoute, William Blachère, Anthony Carmona, Maxime Pyz, Adam Garcia, Jessica Vargas Andrande, merci à vous.

TABLE DES MATIÈRES

CHAPITRE 1

INTRODUCTION

Prosopographie signifie d'un point de vue étymologique « description d'une personne » (du grec πρόσωπον : « personnage (de théâtre) ») et apparaît pour la première fois au XVIIIe siècle en désignant alors une liste d'individus. Plus tard, les historiens l'utilisent pour désigner des notices biographiques afin d'étudier des caractéristiques communes d'un groupe d'acteurs de l'histoire, au moyen d'une analyse collective de leur vie (STONE, 1971). Plus précisément, cette étude collective cherche à dégager les caractères communs d'un groupe social (*e.g.* communauté, institution, statut, profession, etc.) en se fondant sur l'observation systématique de leurs vies et de leurs parcours (DELPU, 2015). Dans ce cadre, la visualisation d'informations présente des perspectives intéressantes pour les historiens. Dans cette introduction, nous allons d'abord présenter le contexte de notre approche suivi de nos contributions.

1.1 Contexte

Les avancées en informatique (*e.g.* en bases de données, organisation ou structuration de l'information, reconnaissance d'images, numérisation, etc.) et l'essor des humanités numériques ont favorisé le développement de nombreux projets de bases prosopographiques comme, par exemple, *COL&MON* [1] qui propose des informations sur les collégiales et monastères de la réforme carolingienne au *Concile de Trente* (816–1563), *Biblissima* [2] qui donne accès au patrimoine écrit du Moyen Age conservé ou connu, *PASE* [3] qui enregistre des informations sur des habitants recensés en Angleterre de la fin du VIe à la fin du XIe siècle, *PBW* [4] qui vise à décrire des individus de l'empire byzantin et les régions avoisinantes entre 642 et 1265. AKOKA et al. (2020) proposent un état de l'art récent sur les projets de bases prosopographiques.

Pour les historiens de telles données, souvent très difficiles à obtenir, à structurer et nécessitant un long travail de recherche, sont très importantes car elles permettent d'essayer de répondre à de nombreuses questions ou bien de soulever de nouvelles hypothèses comme par exemple : *Comment et quand a émergé ce courant de pensée dans cette communauté ? Est-il lié au fait que différents individus se sont rencontrés au même moment et*

1. https://colemon.huma-num.fr. Tous les liens de cette thèse ont été accédés le 2022-01.
2. https://projet.biblissima.fr/fr/projet/presentation
3. https://pase.ac.uk
4. https://pbw2016.kdl.kcl.ac.uk

au même lieu ? Pourquoi et comment les fonctions des individus d'un groupe ont évolué au cours du temps et de l'espace ? Est-ce qu'un individu a pu avoir beaucoup d'influence dans son groupe social ? Si oui, pourquoi ? Quand ? Comment ? Existe-t-il une période ou un lieu qui a favorisé l'émergence d'un courant de pensée dans une institution ? etc. Pour avoir des réponses à ces questions, il est possible d'interroger les bases avec des langages de requêtes comme SPARQL. Cependant, en plus de l'expression des requêtes qui est loin d'être triviale pour un historien, les données sont souvent incertaines (*e.g.* différentes variantes de noms), avec des granularités différentes (*e.g.* période de temps ou date), peuvent évoluer au cours du temps (*e.g.* le nom d'un lieu n'est plus le même), etc. Même si des projets s'appuient sur des Systèmes d'Informations Géographiques (*GIS*) pour localiser des évènements, l'interrogation se résume souvent à positionner sur une carte un ou plusieurs individus sans possibilité de voir des évolutions au cours du temps.

D'un autre côté, la visualisation d'informations étudie l'utilisation de l'informatique graphique et interactive pour faciliter l'acquisition et l'emploi des données (Card et al., 1999, p. 6). Elle propose toutes les fonctionnalités pour aider l'utilisateur à naviguer et analyser ces données. Les individus, le temps, l'espace et les évènements sont au cœur des bases de données prosopographiques. Cependant, comme nous le verrons ultérieurement dans le mémoire, il n'existe que peu d'approches de visualisation pour ce type de données. C'est dans ce contexte que se situe cette thèse :

> Quelle visualisation d'information offrir aux experts pour explorer et naviguer dans des données prosopographiques et les aider à répondre à leurs questions ou à tester leurs hypothèses ?

Pour proposer une visualisation adaptée, un concepteur doit répondre à trois questions : **Quoi ? Pourquoi ?** et **Comment ?** (Sallaberry, 2020). Munzner (2014), qui propose une vue d'ensemble de l'état actuel des connaissances sur l'espace de conception de la visualisation, offre des conseils, des recommandations ou des suggestions pour répondre à ces questions et il convient au concepteur de s'en inspirer pour trouver celles qui seront les plus adaptées aux besoins des experts du domaine.

Quoi ?

La première question à laquelle nous devrons répondre est de savoir quelles sont les données manipulées ? Les données prosopographiques sont certes spatio-temporelles et liées à des individus et des évènements mais elles possèdent aussi de nombreuses particularités : des données qui peuvent être incomplètes, des dates qui peuvent être précises, inconnues ou bien correspondant à des périodes, des localisations qui peuvent évoluer au cours du temps (*e.g.* la fusion des régions françaises en 2016), des localisations imprécises (*e.g.* où est située la Faculté de Paris ? existe-t-il/existait-t-il plusieurs bâtiments répartis dans la ville pour la Faculté ?), etc. Cette analyse va nous permettre de mieux structurer ces données brutes pour, par la suite, pouvoir les visualiser ou les explorer via une plateforme. Bien entendu, à ce niveau là, des choix vont être effectués et auront des répercussions sur l'espace de conception, *i.e.* l'ensemble de types d'encodages visuel et des interactions disponibles par rapport à ces choix. Heureusement des taxonomies existent (*e.g.* Munzner, 2014) pour aider le concepteur à trouver parmi les éléments disponibles ceux qui sont les plus adaptés pour structurer un jeu de données.

Cependant, même si, à ce niveau, nous pouvons envisager des structures, ces dernières ne peuvent pas être considérées sans intégrer l'objectif principal de la visualisation et il s'agit là de répondre à la question : pourquoi ?

Pourquoi ?

Pour répondre à cette question, il est indispensable de collaborer et d'échanger avec les experts du domaine car c'est ici que nous définissons les différentes tâches qui devront être réalisées à l'aide de la visualisation. Cette étape conceptuelle consiste pour le concepteur de l'application à lister mais surtout à *traduire* les questions en tâches qui soient réalisables avec les structures de données retenues en réponse à la question précédente.

Comment ?

Les deux étapes précédentes permettent de donner une bonne indication de ce que l'on veut faire (*réponse au pourquoi*) et sur quoi on veut le faire (*réponse au quoi*). Nous avons ainsi transformé les problèmes associés au domaine de l'expert en problèmes de visualisation. Il s'agit maintenant de trouver comment résoudre ces problèmes. Pour cela, en tant que concepteurs, notre objectif est de rechercher les visualisations adaptées mais également (*et surtout ?*) les techniques d'interactions qui vont offrir à l'expert la possibilité de manipuler facilement les données. Bien entendu, des éléments comme la modification, la sélection, la navigation (*e.g.* défilement, zoom, etc.) sont nécessaires et indispensables dans notre contexte. Cependant, liées aux différentes tâches qui pourront être mises en évidence, il se peut que d'autres interactions soient nécessaires entre différentes visualisations. La conséquence évidente est que cela aura aussi un impact sur l'espace de conception de la visualisation et éventuellement … sur les autres questions.

Comme nous l'avons vu, les questions sont extrêmement liées entre elles et la réponse à l'une des questions aura forcément un impact sur les autres. L'objectif principal de cette thèse est donc de répondre à ces trois questions afin de proposer une plateforme de visualisation pour des données prosopographiques adaptée aux besoins des experts. Des choix vont être effectués pour répondre à ces questions. Ces derniers soulèvent d'autres objectifs. Par exemple, intuitivement, nous savons qu'il va falloir représenter un ensemble d'individus. En suivant la taxonomie de Munzner (2014), une structure particulièrement adaptée est celle des graphes. Il est bien connu que l'utilisation d'interactions sur des graphes (*e.g.* zoom) peut engendrer des problèmes de chevauchements (Misue et al., 1995). Dans ce cadre, l'un des autres objectifs est aussi de déterminer quel algorithme est le plus approprié pour gérer ces problèmes. Il existe de nombreuses propositions et pour le concepteur l'essentiel est d'arriver à déterminer, dans son contexte, la solution la plus adaptée en tenant compte de nombreux paramètres (*e.g.* temps, déformation, etc.) toujours avec le souci d'offrir une interface adaptée aux besoins des experts. Enfin, les données sont spatio-temporelles. Pour l'expert, le fait de pouvoir localiser un lieu par rapport à ses recherches est indispensable. Il est donc nécessaire d'offrir une plateforme qui pourra représenter un ou plusieurs évènements, ou bien une personne ou un groupe de personnes (à ce niveau il est trop tôt pour savoir ce qui sera représenté) sur une carte. Là aussi, des interactions sont indispensables pour naviguer sur la carte. Le dernier objectif auquel

nous souhaitons répondre est donc, à partir d'une carte, d'un ensemble de points et d'interactions, d'offrir le plus de fluidité dans la navigation, *e.g.* minimiser le temps d'attente de l'utilisateur lors de regroupement d'objets à certains niveaux de zoom.

Cette thèse a été effectuée dans l'équipe ADVANSE (*ADVanced Analytics for data SciencE*) au LIRMM (*Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier*) de Montpellier. Elle s'inscrit dans le cadre de l'ANR DAPHNE [5]. Les travaux effectués durant la thèse ont été faits en collaboration avec les équipes des laboratoires LARHRA [6], TECHNÉ [7], CÉDRIC [8] et LAMOP [9].

1.2 Contributions

Pour répondre à l'utilisation de la visualisation d'informations pour les données prosopographiques, nous avons proposé la plateforme *ProsoVis* (cf. Figure 1.1) qui offre de nombreuses fonctionnalités pour naviguer au sein des données. Elle permet via des vues à la fois globales (l'ensemble des individus) ou détaillées (un ou plusieurs individus) de pouvoir suivre au cours du temps et de l'espace les évolutions des différentes personnes.

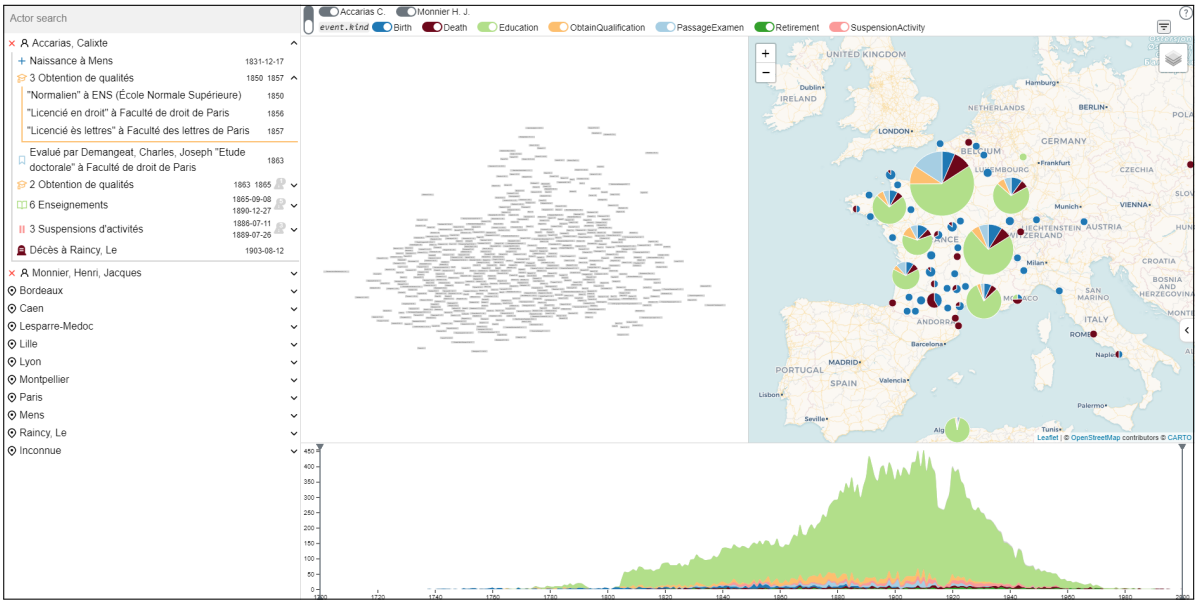


FIGURE 1.1 – ProsoVis *: une plateforme de visualisation et d'exploration de données prosopographiques*

De nombreuses visualisations et interactions ont été proposées afin de faciliter la navigation ou le filtrage d'informations et surtout aider l'expert à réaliser les différentes tâches qui ont été définies. La plateforme utilise actuellement les données de la base

5. Découverte dans les bAses Prosopographiques Historiques de coNnaissancEs (ANR DAPHNE *17-CE28-0013-01*) https://anr.fr/Projet-ANR-17-CE38-0013
6. LAboratoire de Recherche Historique Rhônes-Alpes http://larhra.ish-lyon.cnrs.fr/
7. Laboratoire de recherche en TECHnologies Numériques pour l'Éducation https://techne.labo.univ-poitiers.fr/
8. Centre d'Études et De Recherche en Informatique et Communications https://cedric.cnam.fr/
9. LAboratoire de Médiévistique Occidentale de Paris https://lamop.pantheonsorbonne.fr/

Siprojuris [10] mais a été conçue pour pouvoir intégrer facilement d'autres types de bases prosopographiques.

Une des approches traditionnelles pour représenter un ensemble de personnes en visualisation est d'utiliser le diagramme nœuds-liens du réseau/graphe formé par ces personnes. C'est ce diagramme que nous avons utilisé pour que l'utilisateur puisse avoir une vue globale de tous les individus de la base de données. Une vue statique peut facilement être réalisée via des algorithmes de placement dans un graphe. Cependant, dès que des interactions sont proposées à l'utilisateur (*e.g.* déplacement, zoom), cela pose de nouveaux problèmes dont celui des chevauchements des nœuds du graphe : les nœuds représentant, par exemple, le nom des individus se superposent et il n'est plus possible de lire les informations. Pour résoudre ce problème, il existe différentes approches qui ont été proposées dans la littérature. La question est alors : *quelle est l'approche la plus adaptée à ma problématique ?* Bien que de nombreuses métriques pour évaluer les approches ont été proposées, aucune des propositions de la littérature n'offrent de comparaisons sur les mêmes critères. Par exemple, si une approche propose de conserver la *minimisation de l'expansion*, elle n'utilise pas la même métrique pour l'évaluer qu'une autre. De la même manière, l'utilisation des temps d'exécution pourrait être utile mais comment comparer des temps quand les implémentations sont réalisées dans des langages différents. Pour permettre de choisir et montrer les raisons qui nous ont poussé à sélectionner une approche particulière pour *ProsoVis*, nous avons comparé les différents algorithmes existants. Pour cela, les algorithmes ont été implémentés dans un même langage et nous proposons une classification des différents critères et pour chacun d'entre eux une mesure représentative. Des expérimentations ont été menées sur 854 graphes synthétiques (aléatoires, arbres, à invariant d'échelle, petits-mondes) et réels afin d'analyser le comportement des algorithmes. Nous montrons ainsi, en fonction des critères attendus pour une visualisation, quel est l'algorithme le plus adapté. Ces travaux ont donné lieu à la publication suivante :

Chen, F., Piccinini, L., Poncelet, P., & Sallaberry, A. (2019). Node overlap removal algorithms: A comparative study. In D. Archambault & C. D. Tóth (Éd.), *Proceedings of the international symposium on graph drawing and network visualization (GD)* (p. 179-192). Springer. https://doi.org/10.1007/978-3-030-35802-0_14

Cet article a été sélectionné dans les *Best Papers* de la conférence ce qui nous a permis d'en publier une version étendue en journal :

Chen, F., Piccinini, L., Poncelet, P., & Sallaberry, A. (2020). Node overlap removal algorithms: An extended comparative study. *Journal of Graph Algorithms and Applications*, *24*(4), 683-706. https://doi.org/10.7155/jgaa.00532

La représentation de la dimension spatiale des données de *ProsoVis* a une grande importance car elle permet aux utilisateurs de pouvoir localiser les évènements sur une carte. Cependant, lorsque des fonctionnalités, de zoom par exemple, sont proposées, il devient important de mettre en évidence, non plus les points individuels mais plutôt des regroupements de points proches géographiquement sous une *forme commune*. Ce problème correspond à une problématique bien connue d'encombrement visuel. De

10. http://siprojuris.symogih.org/

nombreuses techniques ont été proposées et, parmi celles-ci, le *Regroupement Spatial Agglomératif* s'est avéré très adapté. L'objectif est le suivant, à partir d'une carte et d'un nombre de points géolocalisés, il s'agit de fusionner les points qui se chevauchent en clusters, la quantité de points qu'ils contiennent affectant directement leurs tailles apparantes. Il suffit ensuite d'afficher à la fois les clusters et les points qui n'ont pas été regroupés sur la carte afin d'obtenir une visualisation sans chevauchements. Différents algorithmes ont été proposés et considèrent deux types d'approches : (1) *en ligne* où le clustering est effectué en temps réel pour un niveau de zoom donné et doit être relancé lorsque l'utilisateur change de niveau et (2) *hors ligne* où l'ensemble de tous les clusters est déterminé au préalable pour chaque niveau de zoom, offrant ainsi plus de fluidité lors de l'interaction. Nous proposons un nouvel algorithme, *F-SAC*, se déclinant en *en ligne* et en *hors ligne*. Son originalité repose sur une nouvelle stratégie de construction des clusters, sur la prise en compte des grands clusters et sur l'utilisation d'une structure de données R-arbres. Les expérimentations montrent que sur 6 jeux de données réelles aux caractéristiques très différentes et 4 000 jeux de données synthétiques générés à partir de distributions différentes, *F-SAC* est significativement plus rapide que les autres approches tout en offrant des résultats similaires. Ces travaux ont donné lieu à la soumission d'un article :

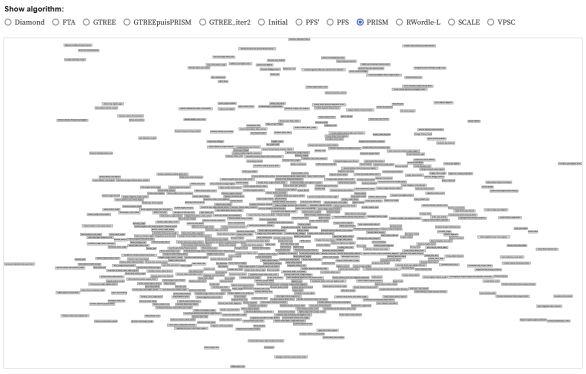
Chen, F., Sallaberry, A., & Poncelet, P. (2022). F-SAC: A fast and efficient algorithm for spatial agglomerative clustering. Soumis.

Outre les publications et la plateforme *ProsoVis*, de nombreux développements ont été réalisés dans le cadre de cette thèse. Motivés par le partage de connaissances et l'*open source*, nous avons souhaité diffuser ces différents travaux. De manière à pouvoir utiliser la plateforme pour les données prosopographiques, une documentation complète a été réalisée afin de pouvoir adapter simplement ses propres données [11]. Dans le cadre des chevauchements de nœuds, outre le fait que les expérimentations et les codes sont mis à disposition [12], nous proposons une plateforme, *AGORA* [13], qui permet à l'utilisateur de pouvoir, pour ses propres graphes, obtenir les graphes sans chevauchements qui résultent de l'application des principaux algorithmes de la littérature ainsi que les valeurs des métriques associées aux critères de qualité de la littérature (cf. Figure 1.2). Enfin tous les jeux de données réels et synthétiques sont mis à disposition [14].

De la même manière, dans le cadre des chevauchements abordées dans les approches de *Regroupement Spatial Agglomératif*, nous mettons à disposition : (1) les codes sources, en *JavaScript*, des algorithmes de l'état de l'art et de *F-SAC* (cf. Figure 1.3(a) [15]) qui ont été utilisés dans ce mémoire, (2) les 6 jeux de données réelles (dont OCS Wrecks, cf. Figure 1.3(b) [16]), (3) les codes de génération des données synthétiques avec différentes distributions, (4) l'ensemble de tous les résultats des expérimentations obtenues sur plus de 4 000 jeux de données synthétiques, l'algorithme pour générer

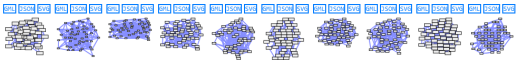
11. https://observablehq.com/@stardisblue/documentation-prosovis
12. https://github.com/agorajs
13. https://agorajs.github.io
14. https://github.com/agorajs/agora-dataset
15. https://observablehq.com/@stardisblue/fsac
16. https://observablehq.com/d/e51fd47b606ba403

ces données et le code utilisé pour obtenir les résultats (*benchmark*) [17].



(**a**) *Visualisation des graphes obtenus par les différentes approches d'AGORA*

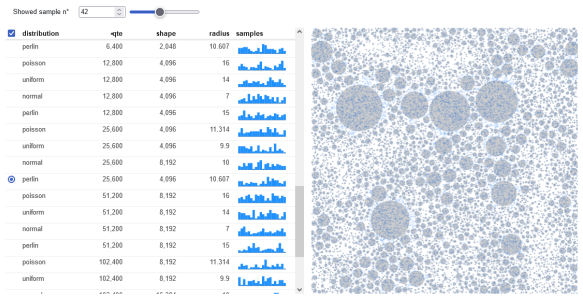
random_50_16.gml overlaps : 87



Criterias	SCALE	PFS	PFS'	FTA	VPSC	PRISM	GTREE	RWordle-L	Diamond
Original	1	0	0	0	0	0	0	0	0
Kendall's Tau Distance	0	0.7298	0.7298	0.1673	0.0416	0.0416	0.0539	0.7518	0.071
Number of Inversions	0	1182	1182	210	51	51	66	1227	87
Normalised Number of Inversions	0	0.4824	0.4824	0.0857	0.0208	0.0208	0.0269	0.5008	0.0355
L1 Length	2.9769	2.8538	1.5846	1.7154	1.1538	1.0077	1.8154	1.1385	2.0462
Bounding Box Area	9.2949	5.0244	2.1936	3.3236	1.4784	2.5645	3.2368	1.2885	4.1549
Bounding Box Normalized Area	0.8924	0.801	0.5441	0.6991	0.3236	0.6101	0.6911	0.2239	0.7593
Convex Hull Area	8.8602	5.4202	2.3426	3.0001	1.4414	2.4866	3.2118	1.307	3.7523
Aspect Ratio	0.9518	1.625	1.1476	0.8854	1.2133	1.2689	1.014	1.0015	1.0115
Improved Aspect Ratio	1.0497	1.6225	1.145	1.1295	1.2091	1.2733	1.0166	1.004	1.0096
Standard Deviation	0.0591	0.7509	0.3911	0.915	1.1388	0.4777	0.4769	0.1048	1.5246
Moved Nodes	1	1	1	1	1	1	1	1	1
Mean Euclidean	181.5973	170.663	80.998	86.6448	24.7342	65.6243	78.5871	58.3096	103.5393
Normalised Euclidean	0.3321	0.3249	0.2777	0.2747	0.1166	0.1979	0.2359	0.2792	0.2747
Hamiltonian	11970	10322.5881	5143.1842	5704	1533.8576	4168.8326	5072.3259	3771.5341	6852
Improved Mean Squared Euclidean	0	1102.7774	116.063	1056.5315	103.204	131.5738	211.0321	228.5632	434.0418
K-Nearest Neighbours	0	2417	2416	436	143	94	76	2350	110
Ratio	1	4.1487	2.7065	2.9675	1.6078	1.2641	1.3126	2.8712	1.3884
Relative Standard Deviation Delaunay	0	0.3915	0.235	0.7081	0.3528	0.2409	0.2579	0.539	0.3369
Relative Standard Deviation	0.0994	0.3449	0.1636	0.3749	0.2372	0.1899	0.1956	0.3063	0.1976

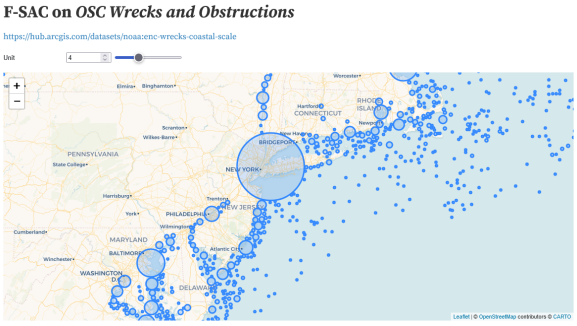
(**b**) *Valeurs des métriques obtenues par les différentes approches*

Figure 1.2 – *Aperçu des outils crées pour* AGORA



(**a**) *Page interactive pour visualiser* F-SAC *sur les jeux de données synthétiques*

F-SAC on *OSC Wrecks and Obstructions*



(**b**) *Jeu de données OCS Wrecks rendu en temps réel avec* F-SAC

Figure 1.3 – *Aperçu des outils crées pour* F-SAC

1.3 Organisation du mémoire

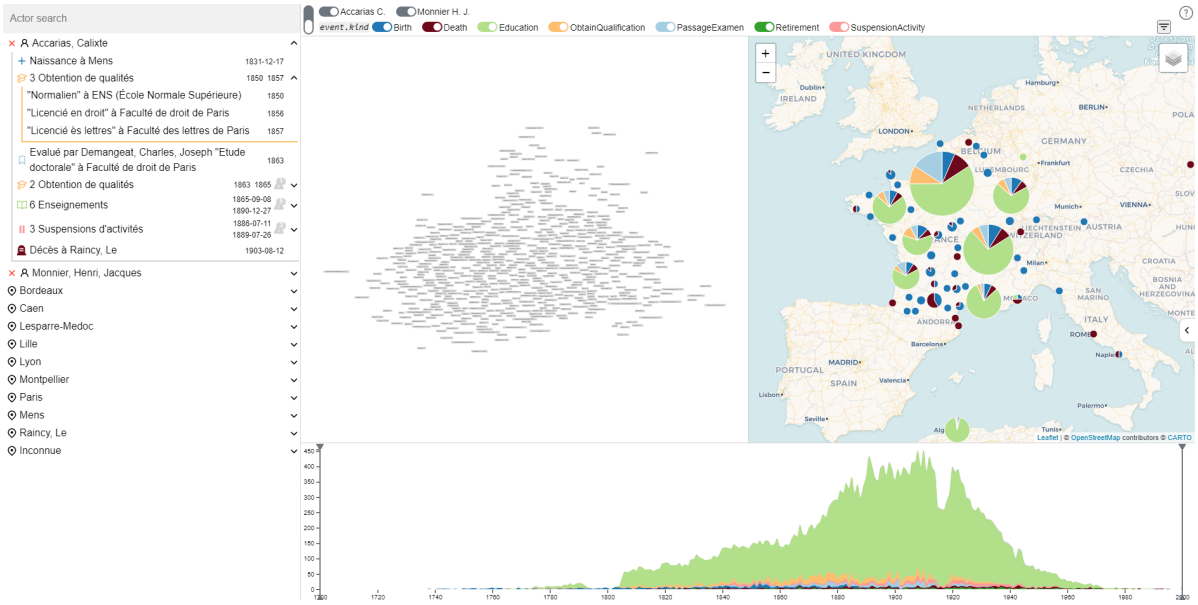
Le mémoire est organisé de la manière suivante. Le Chapitre 2 présente *ProsoVis*, une plateforme de visualisation pour les données prosopographiques. L'étude menée sur la comparaison des différents algorithmes de suppression de chevauchements dans des graphes est décrite dans le Chapitre 3. L'approche *F-SAC* est présentée dans le Chapitre 4. Enfin, nous concluons et présentons les perspectives associées à nos travaux de recherche dans le Chapitre 5.

Les chapitres sont organisés de manière similaire : après un rappel de la problématique et de l'étude de l'état de l'art, nous présentons notre contribution et l'évaluons. Ensuite, nous proposons une discussion avant de conclure le chapitre.

17. l'ensemble des ressources est disponible sur https://github.com/stardisblue/fsac

CHAPITRE 2

PROSOVIS



Sommaire

Nous avons vu, en introduction, que la prosopographie est l'étude de la biographie de personnages historiques, leurs trajectoires et leurs interactions. Ces biographies peuvent ensuite être rassemblées pour servir à l'étude d'un groupe. Une donnée prosopographique peut donc être définie comme la description de la vie d'un individu, l'ensemble des évènements connus décrivant son existence et les différentes relations qu'il a eu cours du temps et de l'espace avec les autres individus. Notre objectif, dans ce chapitre, est de proposer une plateforme de visualisation adaptée à ces données pour aider les historiens à répondre à leurs questions ou tester leurs hypothèses.

2.1 Contexte

Dans l'introduction, nous avons mis en évidence qu'il était indispensable, pour avoir une visualisation adaptée, de répondre à 3 questions : *Quoi ? Pourquoi ?* et *Comment ?* Dans ce chapitre nous montrons comment nous avons répondu à ces questions qui sont très liées entre elles.

Pour répondre au *Quoi ?*, nous disposons d'éléments de réponses. Il faut représenter des individus et des évènements dans l'espace et le temps. Si nous considérons uniquement ces éléments, il peut exister une grande variété de réponses au *Comment ?*. Il faut donc s'intéresser au *Pourquoi ?* afin de réduire l'espace conceptuel (*design space*) et ainsi pouvoir proposer une visualisation adaptée aux utilisateurs. La réponse à cette question est délicate : il est indispensable de bien comprendre les besoins des historiens et de connaître les tâches que la visualisation doit permettre d'accomplir. Au final, les réponses aux trois questions sont fortement entremêlées et un choix de réponse à l'une des questions (*e.g.* *Pourquoi ?*) aura forcément des répercussions sur les réponses aux autres questions (*e.g.* *Comment ?* et *Quoi ?*). Dans ce chapitre, nous faisons donc un état de l'art des approches existantes pour voir si elles sont adaptées à notre problématique puis nous présentons la plateforme de visualisation d'information *ProsoVis* développée et répondant à nos trois questions.

Le reste du chapitre est organisé de la manière suivante. La Section 2.2 présente l'état de l'art des approches en considérant respectivement, la visualisation d'évènements, de personnes, les approches hybrides et les propositions spécifiques aux données prosopographiques. La Section 2.3 analyse les besoins et propose un modèle pour la représentation des données prosopographiques ainsi que son application aux données de la base Siprojuris. Les motivations et les choix lors de la conception de la plateforme sont décrits dans la Section 2.4. Nous y présentons également les différents composants et leur fonctionnement. Un exemple d'utilisation de *ProsoVis* est décrit dans la Section 2.5. Enfin, une discussion est proposée, Section 2.6, avant de conclure le chapitre dans la Section 2.7.

2.2 État de l'art

La visualisation de données prosopographiques comprend notamment la représentation d'évènements et le suivi de la biographie d'un individu ou d'un groupe d'individus. Différentes informations peuvent être visualisées en fonction de l'objet

d'étude. Dans cette section nous présentons, dans un premier temps, les approches de visualisations d'évènements. Nous nous intéressons, par la suite, aux travaux liés à la représentation d'individus de manière détaillée ou en groupe, tout en la généralisant au suivi d'entités au cours du temps. Ensuite, nous décrivons les approches hybrides pour visualiser des individus et des évènements. Enfin, notre contexte étant celui des données prosopographiques, nous étudions les différentes approches spécifiques de visualisation de la littérature.

2.2.1 Visualisation d'évènements

Un évènement fait généralement référence à un fait s'étant produit dans le réel. Il peut être associé à une localisation spatiale et possède une dimension temporelle représentée soit par une date, soit par une période (Aigner et al., 2011, p. 47). Bien entendu, différentes informations complémentaires et/ou propriétés peuvent être associées à un évènement.

Dans *VisGets* (Dörk et al., 2008) (cf. Figure 2.1), des ressources Web, extraites à partir d'un flux RSS, sont considérées comme des évènements et sont représentées avec trois dimensions : temporelle, spatiale et mots-clés (*tags*). *VisGets* permet de filtrer sur ces dimensions à l'aide d'une frise chronologique, d'une carte ou en utilisant un nuage de mots représentatif du corpus. Ces représentations sont synchronisées, ainsi l'action sur une vue entraîne une mise à jour des autres vues. Les ressources filtrées apparaissent dans une vue spécifique.

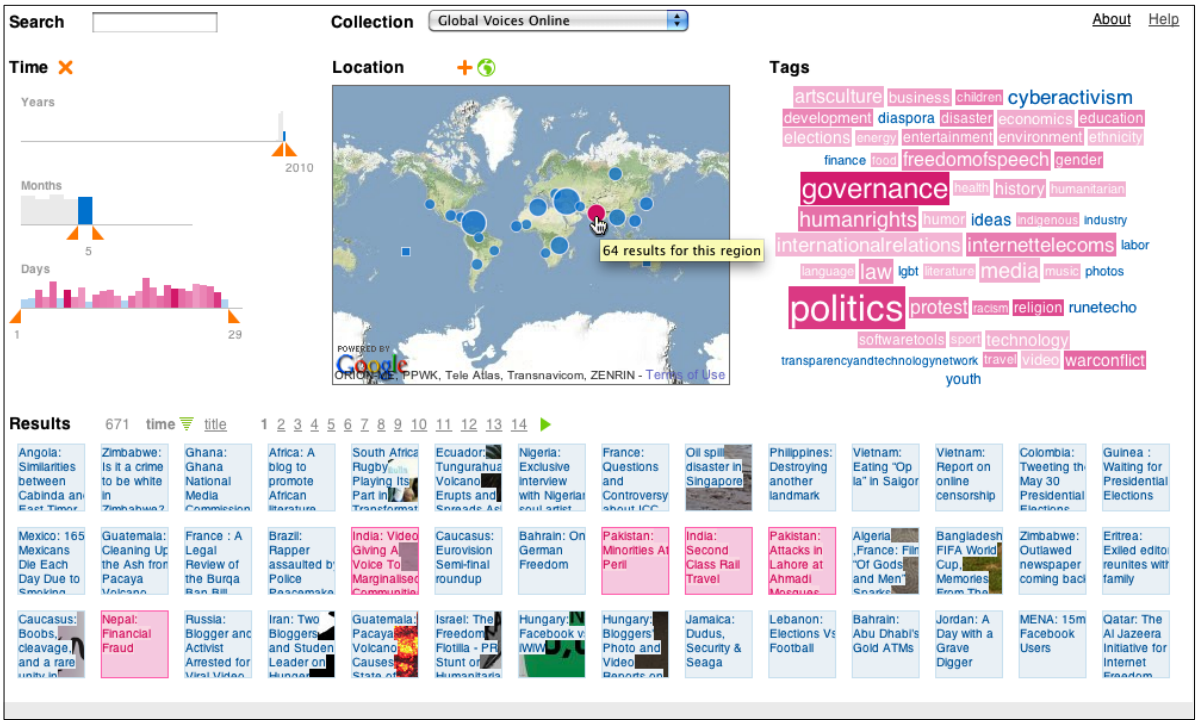


Figure 2.1 – *Aperçu de l'interface de* VisGets (*Dörk et al., 2008*)

Un principe similaire est proposé dans *VAiRoma* (Cho et al., 2016) (cf. Figure 2.2) qui offre une visualisation d'information en extrayant les évènements depuis Wikipedia.

Une partie des mot-clés est récupérée via une extraction des thèmes, basée sur une approche d'apprentissage automatique (*topic modeling*). Ces thèmes sont hiérarchisés et présentés sous la forme d'un graphique en rayon de soleil (*sunburst*, cf. Figure 2.2C). La densité des évènements est représentée sur la carte sous la forme d'une carte de chaleur (*heatmap*, cf. Figure 2.2B). Il est possible de comparer deux périodes temporelles et d'accéder à la page Wikipedia de la ressource concernée. De plus *VAiRoma* intègre un système d'annotation pour sauvegarder la navigation actuelle ou la connaissance acquise lors de l'utilisation de la plateforme.

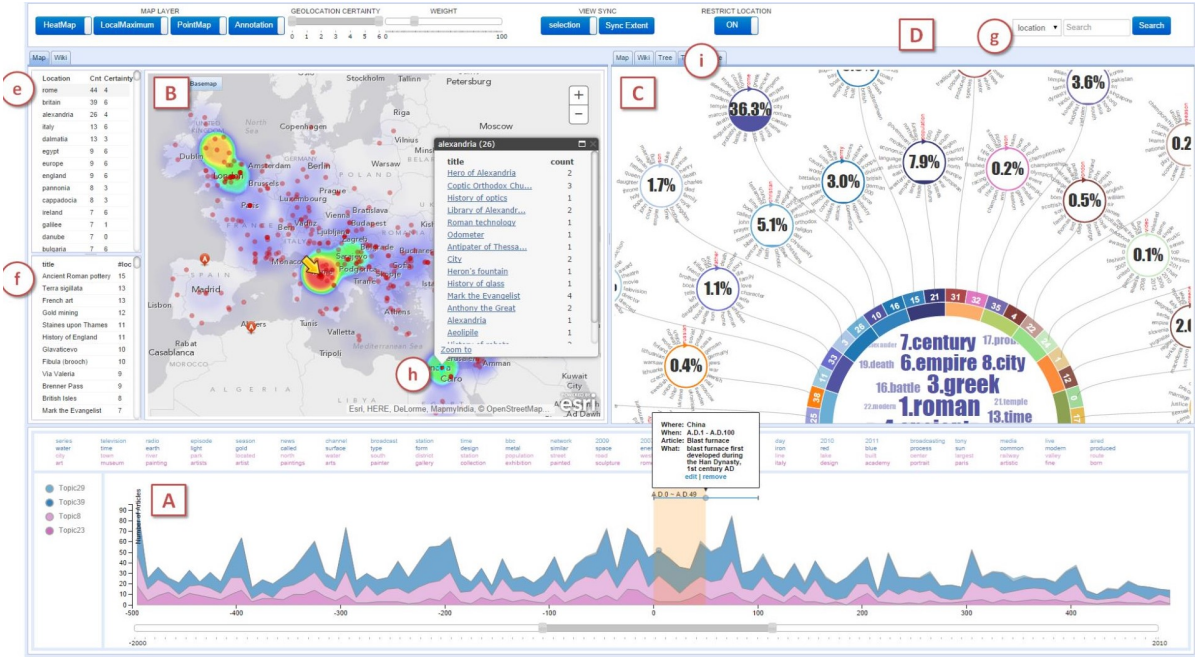


FIGURE 2.2 – *Aperçu de l'interface de* VAiRoma (CHO *et al.*, 2016, *p. 210*)

À l'instar de ses prédécesseurs, *HisVA* (HAN et al., 2021) (cf. Figure 2.3) se situe dans la catégorie « pédagogique » pour l'apprentissage d'évènements historiques. La navigation, effectuée via des mot-clés dans *VAiRoma*, est ici remplacée par une navigation de proche en proche. Une métrique de similarité est calculée entre deux évènements et est ensuite utilisée pour suggérer des évènements similaires à ceux sélectionnés. Cela permet d'enrichir la connaissance sur un domaine de manière ciblée.

L'objectif des approches précédentes est de se focaliser sur des évènements « généraux » intervenant au cours du temps. Elles offrent de nombreuses facilités pour analyser les données : filtres, frises chronologiques, cartes, etc. Cependant, elles ne permettent pas de se focaliser sur l'historique des évènements d'un individu particulier ou de pouvoir facilement comparer le parcours de plusieurs individus.

2.2.2 Visualisation d'individus

Dans cette section, nous nous intéressons aux approches de visualisation d'individus. Nous utilisons ici le terme individu pour désigner une entité qui existe dans le temps, pouvant être décrite par un ou plusieurs évènements, ou par ses relations avec d'autres entités.

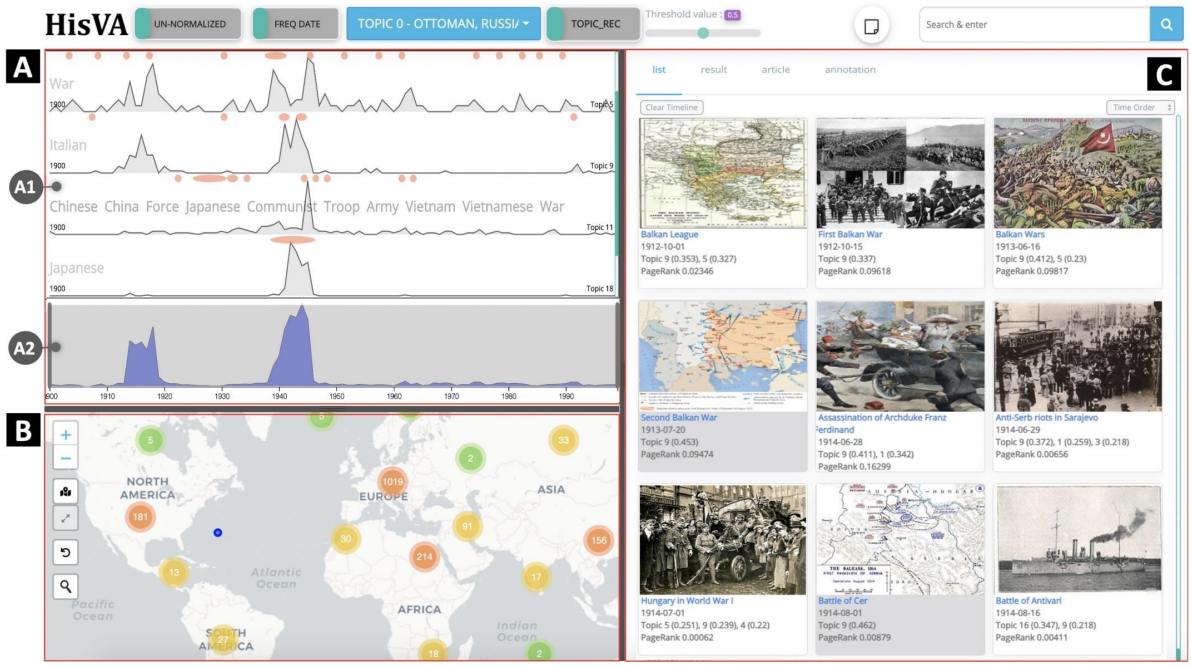


Figure 2.3 – *Aperçu de l'interface de* HisVA (*Han et al., 2021*)

Dans *GeneaQuilts* (cf. Figure 2.4), Bezerianos et al. (2010) abordent le lien de parenté entre les individus et proposent une visualisation interactive inspirée du concept d'arbre généalogique. Différentes interactions permettent de filtrer ou de mettre en avant les descendants, ancêtres et liaisons qu'ont eu les individus. Une fonction de navigation de proche en proche est aussi possible permettant ainsi de naviguer d'un parent à ses enfants de façon fluide. L'interface propose également une vue d'ensemble pour situer la généalogie dans sa période temporelle.

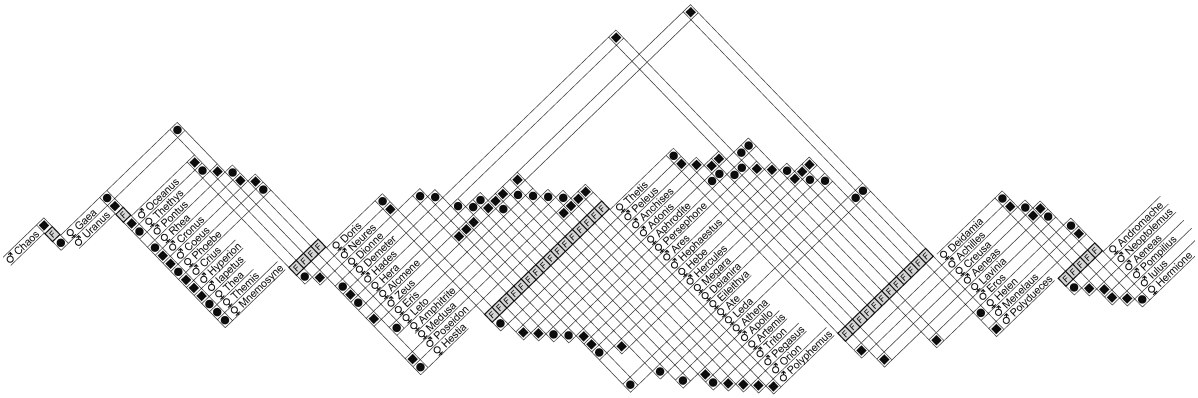


Figure 2.4 – GeneaQuilts (*Bezerianos et al., 2010, p. 1073*)

Pour représenter d'autres types de relations, Itoh et Akaishi (2012) proposent *TimeSlice* (Itoh et al., 2010) (cf. Figure 2.5) qui s'appuie sur un graphe de relations pour mettre en avant l'évolution des interactions entre individus. Ces interactions appartiennent à des catégories qui peuvent être visualisées individuellement. L'évolution temporelle est représentée via des petits multiples (*small multiples*), *i.e.* un ensemble de figures représentant différentes facettes d'un même jeu de données. Chaque facette

correspond, dans ce cas, à une tranche (*slice*) dans le temps. Avec un principe similaire, l'évolution des relations entre individus est modélisée en modifiant la représentation de l'arête entre ces derniers.

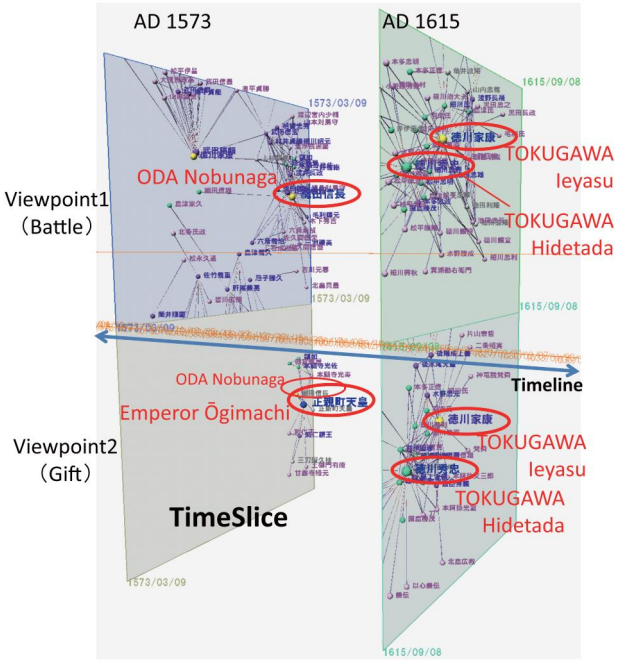


Figure 2.5 – TimeSlice (*Itoh & Akaishi, 2012, p. 286*)

Janicke et al. (2016) (cf. Figure 2.6) proposent un moyen alternatif de visualiser les relations en représentant des musiciens dans un diagramme de flux. Chaque colonne représente une catégorie. Le flux de chaque musicien traverse la catégorie avec la valeur associée mettant ainsi en évidence les corrélations entre les musiciens. La première colonne (cf. Figure 2.6, *Wirkungszeit*) représente la période temporelle du musicien. Différents symboles sur l'intervalle permettent de représenter l'incertitude de la datation. En plus de l'explorateur de colonnes, une carte permet de mettre en évidence les lieux où les musiciens sélectionnés se sont rendus aidant ainsi à identifier de potentiels lieux de collaboration. Enfin, un graphe de relations entre les musiciens permet de rajouter les musiciens d'intérêts dans l'explorateur de colonnes.

S'inspirant du célèbre « A Chart of Biography » (Priestley, 1765), Khulusi et al. (2019) proposent une interface affichant la chronologie d'un ensemble de musiciens (cf. Figure 2.7). L'interface est composée d'une vue détaillée et d'une vue d'ensemble. Les musiciens peuvent être groupés par différents attributs (*e.g.* l'institution). La distinction entre les groupes se fait via la couleur. La sélection d'un individu affiche son détail ainsi qu'un graphe des relations liées à l'individu.

Également dans le domaine musical, l'interface proposée par Kusnick et al. (2020) (cf. Figure 2.8) permet de visualiser la « vie » d'un instrument de musique sous la forme d'une chronologie. Les couleurs des évènements encodent son type, la transparence permet de communiquer l'incertitude. La représentation varie en fonction de la résolution, *i.e.* lorsque l'intervalle de temps est important, les évènements sont agrégés

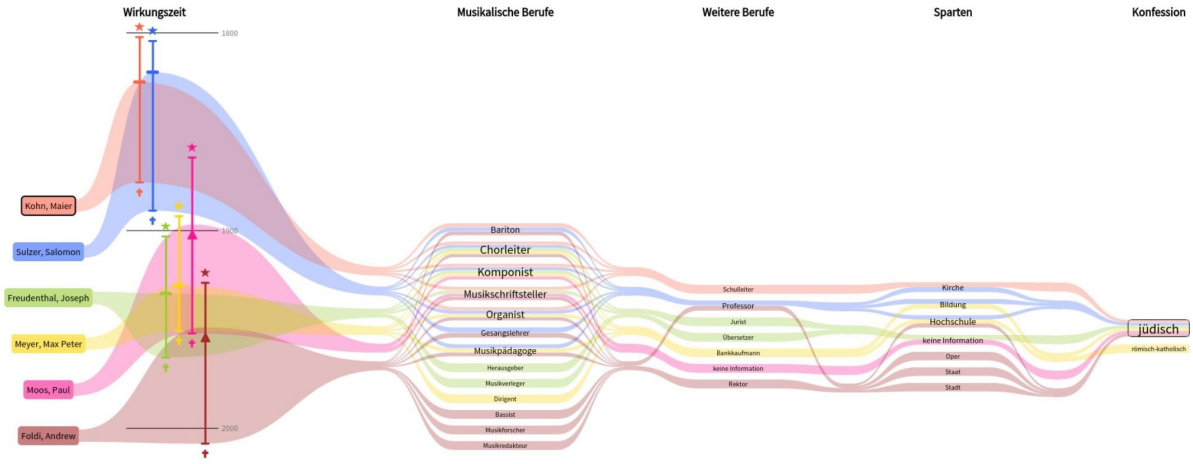


Figure 2.6 – *Explorateur de colonnes* (column explorer) *de* Janicke *et al.* (2016, *p. 200*). *Les colonnes représentent, de gauche à droite, le temps d'activité* (Wirkungszeit), *la profession musicale* (Musikalische Berufe), *autres professions* (Weiteree Berufe), *secteurs* (Sparten) *et confessions* (Konfession).

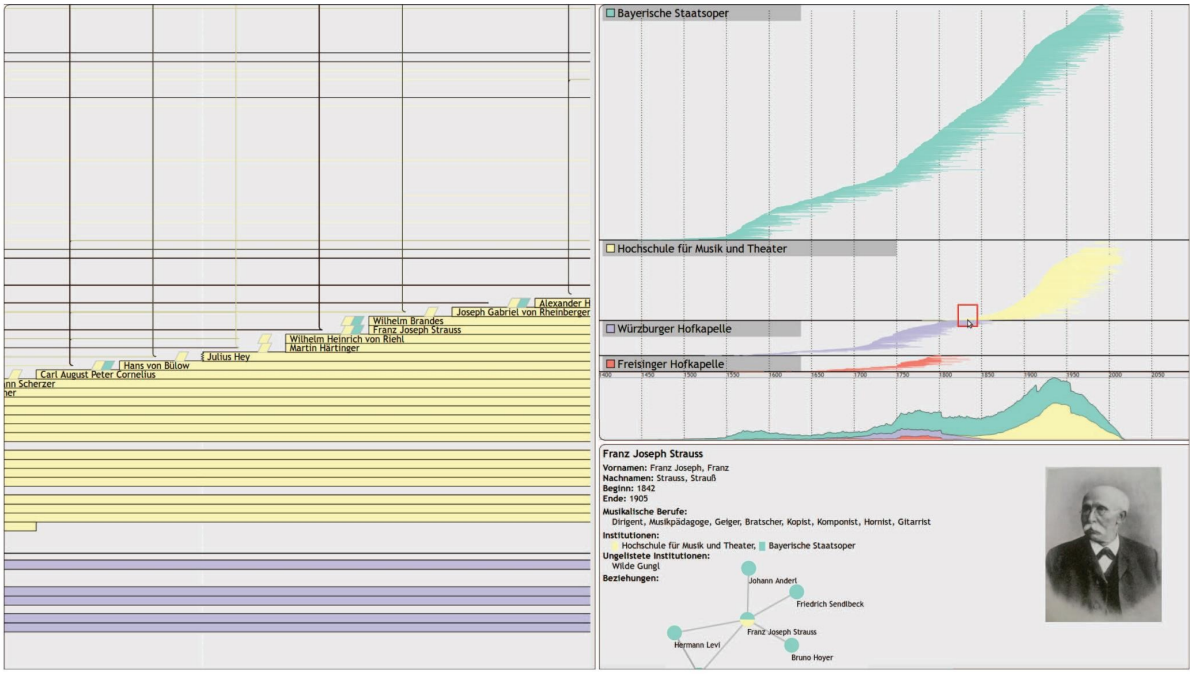


Figure 2.7 – *Aperçu de l'interface de* Khulusi *et al.* (2019, *p. 249*)

en un ovale flouté. Il est possible d'afficher chaque évènement individuellement en zoomant. Enfin, une carte permet de positionner les lieux des évènements visibles sur la chronologie.

Le Tableau 2.1 résume les capacités des approches à offrir une vue globale ou détaillée. Pour chacune de ces vues nous évaluons également leur capacité à gérer le temps, l'espace et les relations entre individus. *GeneaQuilts* (Bezerianos et al., 2010) est limité à des relations familiales. *TimeSlice* (Itoh & Akaishi, 2012) permet de bien représenter les relations entre les individus mais, en ne pouvant sélectionner que des

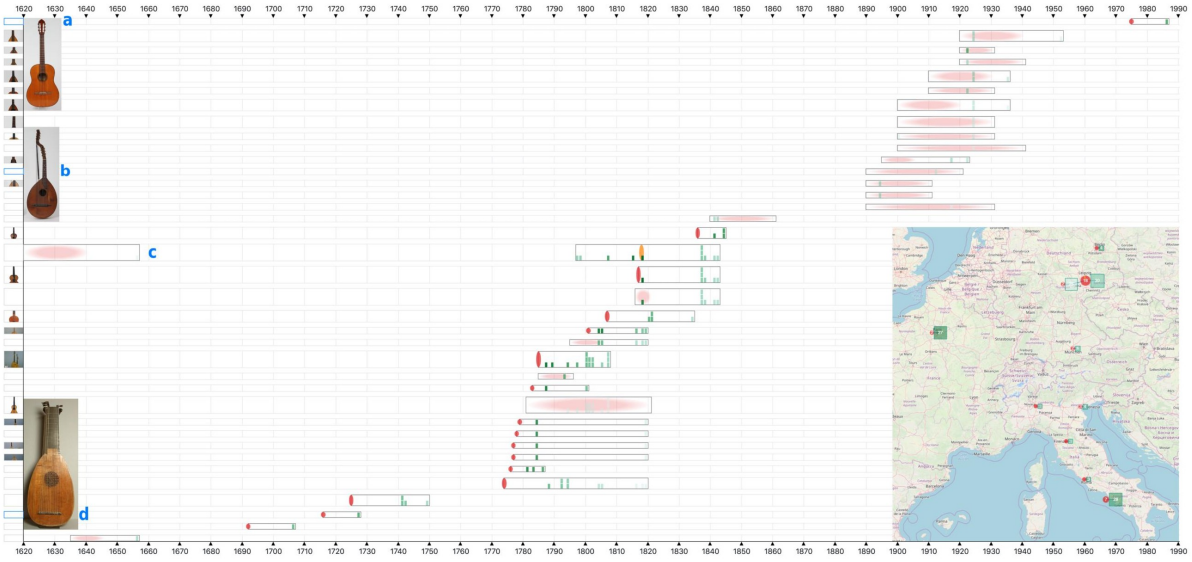


Figure 2.8 – *Aperçu de l'interface de Kusnick et al. (2020, p. 263)*

tranches choisies, ne permet pas d'avoir une vision globale dans le temps. L'approche de Janicke et al. (2016) fournit une bonne vision des interactions entre les individus mais manque d'une vision globale des données sur les dimensions spatiales et temporelles. Khulusi et al. (2019) proposent une fine granularité sur la dimension temporelle mais pas de vision globale des relations et de représentation de la spatialité. Enfin, l'interface proposée par Kusnick et al. (2020) manque d'une représentation globale des données et ne possède aucun moyen de représenter les relations entre individus.

Table 2.1 – *Récapitulatif des approches de visualisation d'individus. Les colonnes correspondent à la capacité de l'approche à représenter les données de façon* Globale *ou* Détaillée. *Chaque groupe se décompose en trois parties, correspondant aux dimensions temporelle, spatiale et relationnelle des données.*

	Globale			Détaillée		
	Temps	Spatiale	Relations	Temps	Spatiale	Relations
Bezerianos et al. (2010)	+		+	+		+
Itoh et Akaishi (2012)				+		++
Janicke et al. (2016)			+	++	++	++
Khulusi et al. (2019)	++			++		+
Kusnick et al. (2020)				++	++	

2.2.3 Visualisations hybrides

D'autres approches ont pris en compte à la fois les évènements et les individus.

Par exemple, *LeadLine* (Dou et al., 2012) (cf. Figure 2.9) définit les évènements comme étant décrit par une temporalité, une position, un individu et des mots-clés.

Les données sont extraites depuis un corpus de texte et représentées sous la forme de plusieurs vues synchronisées. Les mots-clés, les individus et les lieux associés aux évènements peuvent ainsi être mis en avant. Il est aussi possible de visualiser le détail d'un individu sélectionné, *i.e.* les mots-clés, les lieux et les évènements qui lui sont associés. Une approche similaire, *Trading Consequences* (Hinrichs et al., 2015), basée sur l'évolution des commodités (*e.g.* sucre), a aussi été proposée.

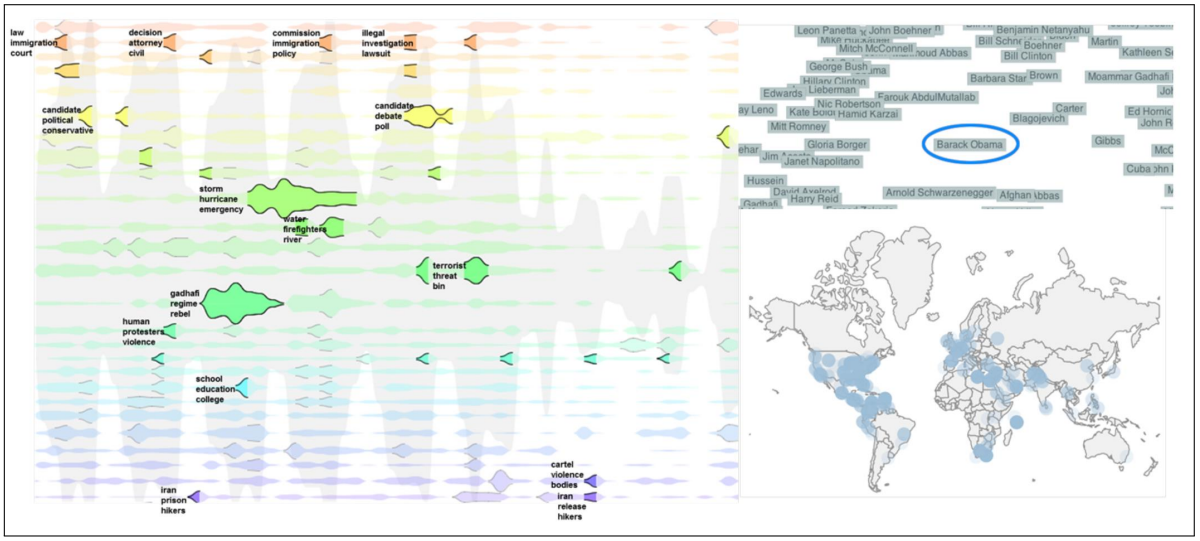


Figure 2.9 – *Aperçu de l'interface de* LeadLine (*Dou et al., 2012, p. 93*)

Latif et al. (2021) (cf. Figure 2.10) proposent une interface qui permet de mettre en évidence les relations entre individus à partir d'un corpus de texte extrait de Wikipedia. Les requêtes (*query*) sont énoncées via une interface semblable à une messagerie instantanée et traitées via des techniques de *traitement du langage naturel* (*NLP*). Les relations entre individus sont ensuite représentées sur quatre vues synchronisées, représentant les évènements de chaque individu ainsi que les évènements communs.

2.2.4 Visualisation de données prosopographiques

Nous avons souligné, en introduction de ce mémoire, que la numérisation des données prosopographiques connaissait un essor assez récent, notamment grâce à la démocratisation du Web Sémantique (illustré par exemple par l'apparition de Workshops comme « Biographical Data in a Digital World »[1]), aux avancées en structuration de données et à la volonté de rendre accessible des connaissances concernant l'héritage culturel (Windhager et al., 2019) de manière générale. Cependant, il n'existe, à notre connaissance, que peu de travaux qui se sont focalisées sur la visualisation de données. Dans cette section, nous présentons les principales approches pour visualiser des données prosopographiques.

Dans *Berkeley Prosopography Services*, Schmitz et Pearce (2013) ont proposé une visualisation se focalisant sur le réseau des relations entre individus représenté sous la forme d'un graphe. Diverses fonctionnalités pour aider à détecter des communautés

1. https://sites.google.com/view/bd2019/

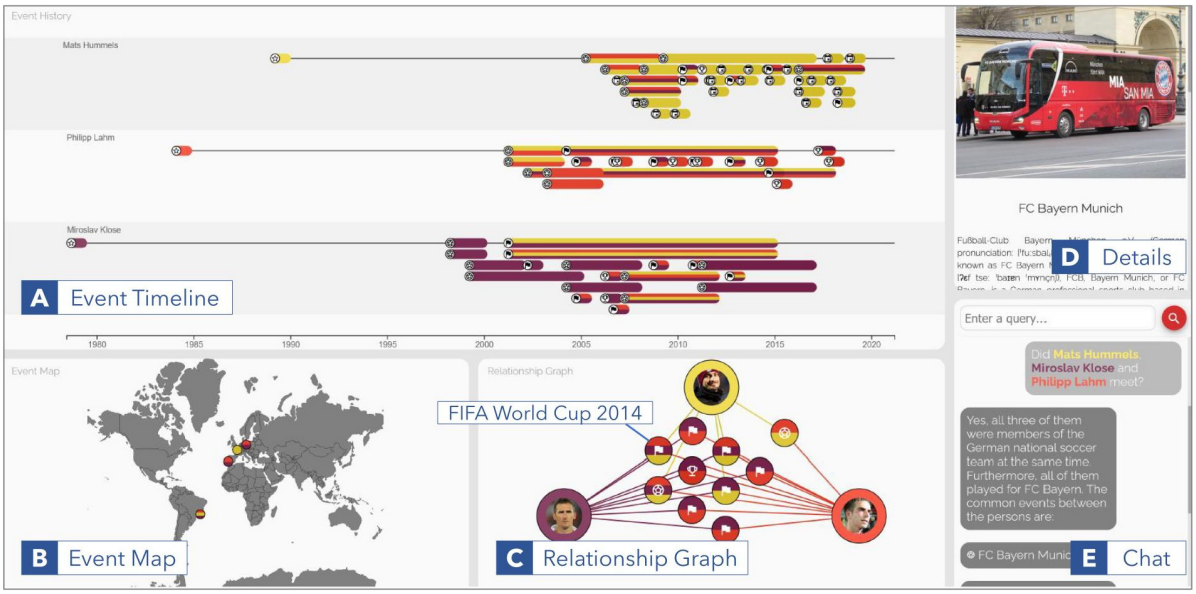


FIGURE 2.10 – *Aperçu de l'interface de* LATIF *et al.* (2021, *p. 156*)

et à annoter des documents sont également proposées. Cependant la visualisation se limite au graphe des relations.

SCHICH et al. (2014) (cf. Figure 2.11) exploitent la visualisation pour mettre en avant les flux et les comportements d'individus notables en fonction de leur lieu et date de naissance (et décès). L'approche permet notamment de représenter l'évolution du flux migratoire de plus de 150 000 individus en Europe et en Amérique du Nord à travers 2 millénaires. Elle permet d'avoir une très bonne vision globale mais ne permet pas d'avoir une vision détaillée des individus et de pouvoir comparer leur parcours au cours du temps et de l'espace.

Avec *Six Degrees of Francis Bacon*, WARREN et al. (2016) proposent une méthode statistique pour inférer des relations entre des individus. Les données sont extraites de la base ODNB [2] en utilisant des techniques d'annotation de corpus de texte et concerne 13 309 individus ayant vécus en Grande Bretagne entre 1500 et 1700. La visualisation, accessible en ligne [3], représente les relations des individus sous la forme d'un graphe de relations (cf. Figure 2.12(a)). Le graphe permet d'indiquer le degré de la relation ainsi que sa nature, *i.e.* si elle est inférée ou indiquée par un utilisateur de la plateforme. Un groupe d'individus peut aussi être présenté sous la forme d'une chronologie (cf. Figure 2.12(b) [4]). Chaque individu est alors représenté sous la forme d'une ligne de vie, de sa naissance à sa mort. Les deux visuels intègrent des interactions afin de faciliter la navigation. Cependant l'interface favorise la visualisation des relations, il est difficile de pouvoir filtrer le temps et aucune visualisation dans l'espace n'est proposée.

Récemment, le portail de données prosopographiques *BiographySampo* (HYVÖNEN et al., 2019), en Finlandais, a été proposé et étend les travaux de LESKINEN et al. (2017) et MIYAKITA et al. (2018). Les données sont stockées en RDF et sont définies suivant une

2. « Oxford Dictionary of National Biography » https://www.oxforddnb.com/
3. http://www.sixdegreesoffrancisbacon.com/
4. http://www.sixdegreesoffrancisbacon.com/?ids=124&type=timeline

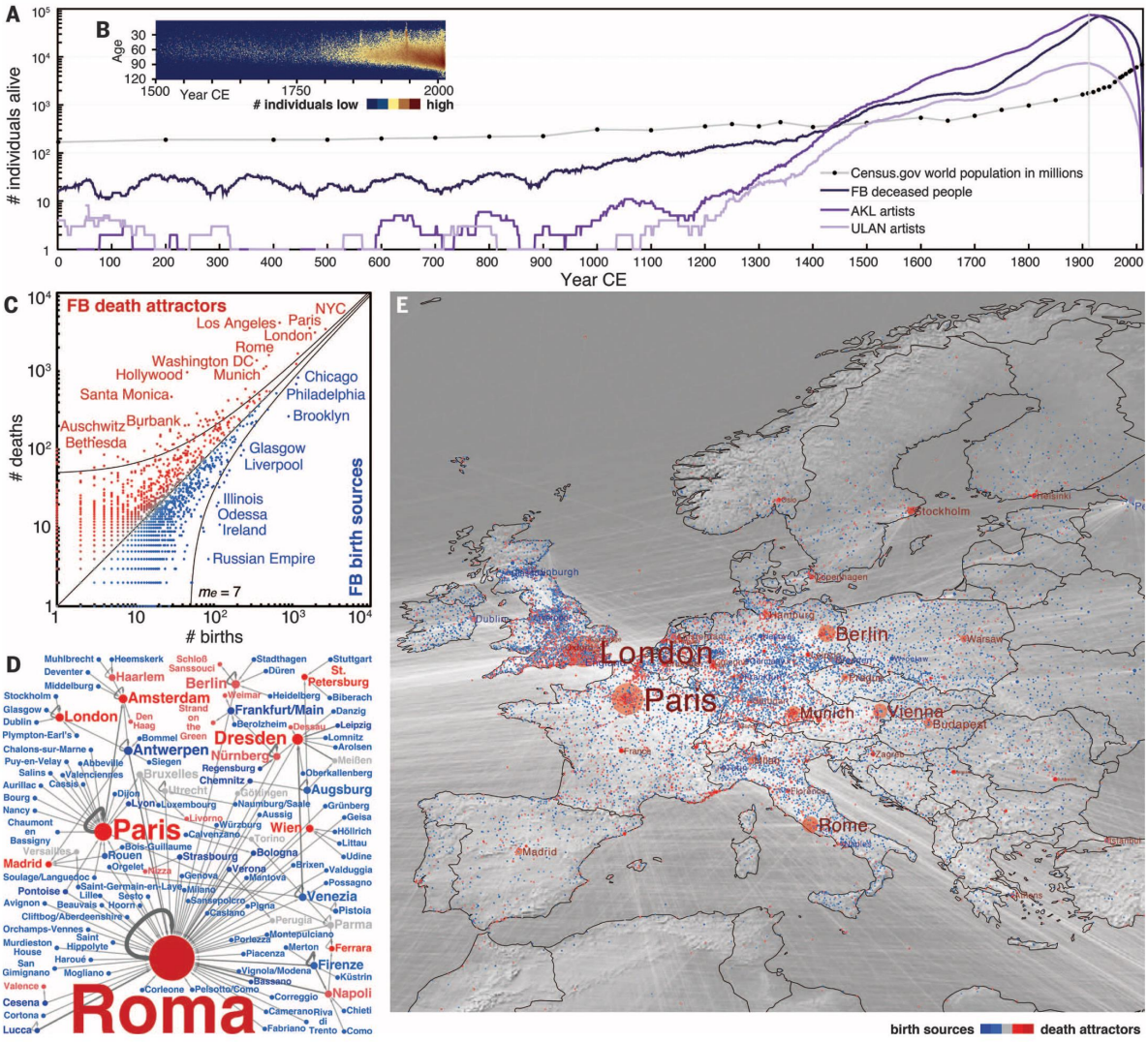
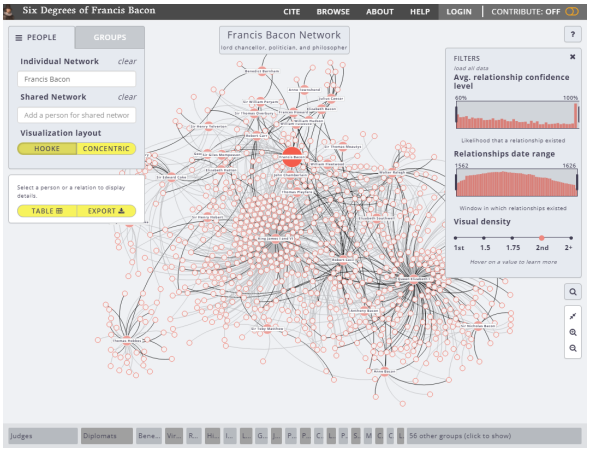


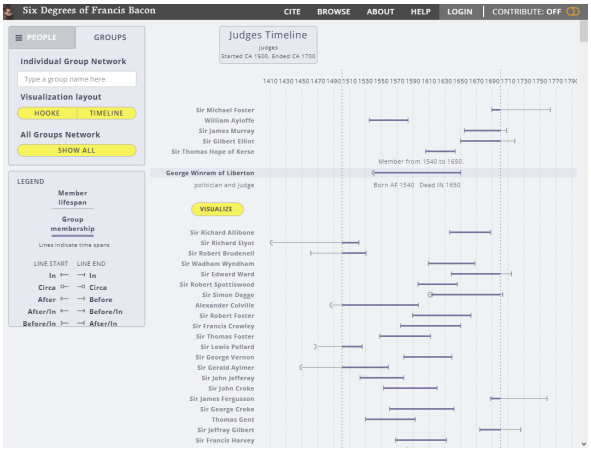
FIGURE 2.11 – *Aperçu des visualisations de SCHICH et al. (2014, p. 559). A désigne le nombre d'individus en fonction du temps des différents jeux de données utilisés. B représente la distribution des âges en fonction du temps. C représente le ratio naissance/décès des différents lieux, en bleu si le lieu à plus de naissance en rouge sinon. D illustre le flux des naissances-décès au 18ième siècle et E le flux migratoire.*

ontologie. L'ontologie est aussi liée à d'autres sources de données tel que Wikipédia. Les données sont accessibles via un point d'accès SPARQL. En plus du modèle et de l'accès aux données, le portail propose une série de visualisations pour les relations entre individus, le parcours des individus de leur naissance à leur décès, des statistiques, une carte avec une frise chronologique, etc. comme l'illustre la Figure 2.13. Les visualisations proposées sont très utiles pour analyser les données prosopographiques. Cependant, les interactions via des onglets spécifiques ne facilitent pas la tâche de l'utilisateur dans la mesure où il est obligé de favoriser un type de visualisation à la fois. De la même manière, la comparaison entre deux cartes pour afficher des vues d'individus ou de groupes d'individus se fait via des barres de défilement (*scrollbar*), ce qui a pour conséquence de masquer les filtres sélectionnés par l'utilisateur. Enfin, l'interface ne propose pas de suggestion pour aider l'expert à étendre ses recherches. Néanmoins, il

convient de noter qu'il s'agit de la première interface permettant d'offrir une navigation prenant en compte les différentes dimensions des données prosopographiques.

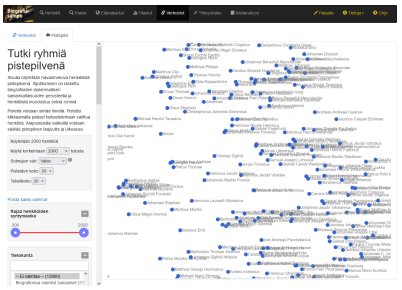


(**a**) *Vue graphe des relations avec* Françis Bacon *au centre en rouge, la couleur dénote le degré de relation avec* Françis Bacon

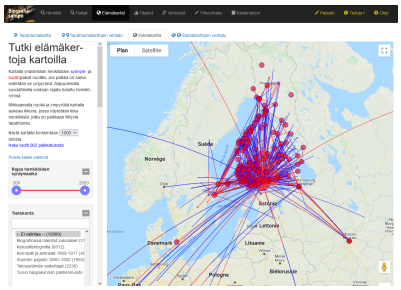


(**b**) *Vue de la chronologie des individus appartenant au groupe « Juges »* (Judges). *Chaque ligne représente les individus, avec l'intervalle commençant par la date de naissance à la date du décès.*

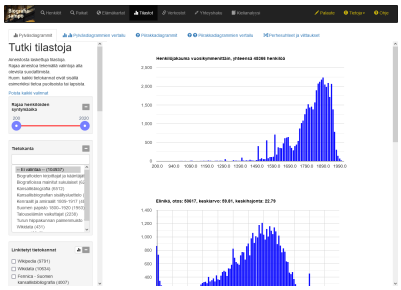
FIGURE 2.12 – *Aperçu des deux vues principales de* Six Degrees of Francis Bacon (*WARREN et al., 2016*)



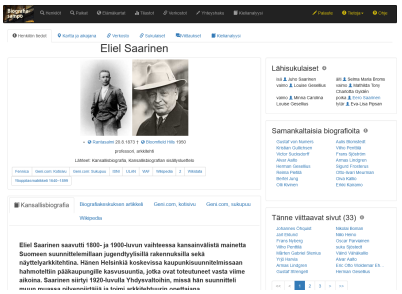
(**a**) *Graphe des individus*



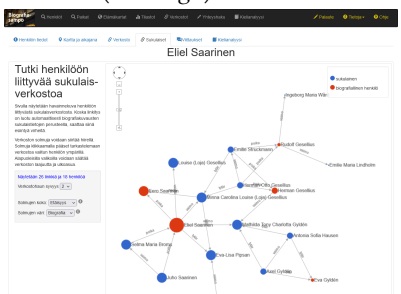
(**b**) *Tracés des naissances (en bleu) → décès (en rouge)*



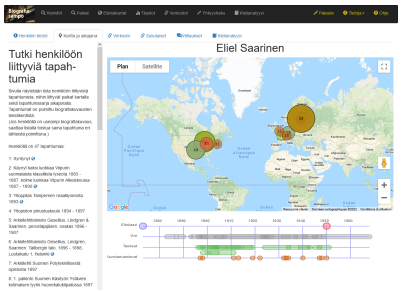
(**c**) *Statistiques des jeux de données*



(**d**) *Fiche détaillée*



(**e**) *Graphe des relations*



(**f**) *Carte et chronologie*

FIGURE 2.13 – *Exemples de visualisations disponibles dans* BiographySampo (*HYVÖNEN et al., 2019*). *(a), (b) et (c) permettent de visualiser le jeu de données de manière globale. (d), (e) et (f) offrent des vues détaillées d'un individu sélectionné, dans cet exemple,* Eliel Saarinen.

2.3 Modélisation des données prosopographiques

Notre objectif est de proposer une plateforme *générique* pour les bases prosopographiques avec des visualisations adaptées aux besoins des historiens. Pour cela, à la lumière des connaissances présentées dans l'état de l'art, il est important de clairement définir conjointement avec les historiens les différents besoins auxquels la plateforme doit répondre. Il faut ensuite proposer un modèle de données qui soit suffisamment expressif et le plus *agnostique* possible en ce qui concerne leur contenu. En d'autres termes, il faut répondre aux deux premières questions que nous avons vu en introduction : *Pourquoi ?* et *Quoi ?*

Dans cette section, nous présentons d'abord les besoins des utilisateurs (*Pourquoi ?*). Le parcours d'un individu étant décrit par un ensemble d'évènements, nous proposons ensuite de représenter les différentes informations à l'aide d'une modélisation des données centrée évènements (*Quoi ?*). Nous montrons également, via les données de Siprojuris, que cette modélisation est parfaitement adaptée à des données prosopographiques.

2.3.1 Définition des besoins

En collaboration avec des historiens et au terme d'un processus itératif, nous avons identifié la liste des besoins suivante :

[B1] Représentation des 3 dimensions des données. Ce besoin se rapporte à la visualisation des données avec leurs différentes dimensions. La première concerne la dimension temporelle **[B1.1]** et nécessite de pouvoir visualiser les données à travers le temps. La seconde concerne la nécessité de pouvoir visualiser les données dans l'espace **[B1.2]**. La dernière est la dimension de l'individu **[B1.3]** et la nécessité de pouvoir visualiser les évènements qui lui sont associés.

[B2] Sélection et filtre des évènements en fonction de la temporalité, de la spatialité ou de l'individu. Ce besoin exprime la possibilité pour mieux analyser les données, de disposer de mécanismes de sélection et/ou de filtrage afin de se focaliser sur des évènements en considérant les dimensions spatiales, temporelles ou associées aux individus.

[B3] Sélection et filtre selon les attributs des évènements. Ce besoin est tout à fait complémentaire du précédent **[B2]** et permet d'exploiter des attributs supplémentaires associées aux données.

[B4] Exploration globale ou détaillée des données. Ce besoin exprime le fait qu'il est indispensable pour l'utilisateur d'avoir d'abord une vue globale de l'ensemble des données et, via cette vue, de pouvoir sélectionner (cf. les besoins précédents) des données particulières et les visualiser avec plus de détails, au sein d'une vue détaillée.

[B5] Navigation de proche en proche. Les individus sont reliés par différents types de relations. Le besoin exprimé nécessite non seulement de pouvoir naviguer de proche en proche mais surtout de mettre en évidence les relations possibles pour la navigation. Par exemple, pour un individu dans la vue globale, il est important de voir les autres individus avec lesquels il a des relations et permettre de naviguer entre eux. Pour la vue détaillée, outre la navigation via leurs relations entre les individus sélectionnés, il est également important de présenter des relations avec des individus non sélectionnés (*suggestions*).

[B6] Mise en évidence des erreurs. Comme nous l'avons vu, les données prosopographiques sont sujettes à de nombreuses erreurs (*e.g.* données manquantes, données incohérentes, données incertaines, erreurs de saisie, etc.). Certaines de ces erreurs peuvent être automatiquement détectées. Mettre en évidence ces erreurs offre une aide à l'amélioration de la qualité des données de la base.

2.3.2 Le modèle de données

Notre objectif est de proposer une modélisation des données qui soit la plus générique possible afin de pouvoir être appliquée à n'importe quelle base de données prosopographiques. Considérant que le parcours d'un individu est décrit par une suite d'évènements intervenus dans différentes lieux à différentes périodes, nous proposons d'utiliser une approche centrée sur les évènements (WINDHAGER et al., 2017).

Un évènement (e) est décrit par un tuple (p, i, l, d) avec p les propriétés de l'évènement, i l'individu concerné par l'évènement, l le lieu et d le moment/la période.

L'individu (i) désigne l'entité concernée par l'évènement. La biographie d'un individu correspond donc à l'ensemble des évènements qui lui sont associés.

Le lieu (l) concerne la dimension spatiale d'un évènement. Il peut être de deux types : (1) une place, un endroit géolocalisable, *e.g.* *Paris* ou (2) une localisation, une entité représentative d'un endroit, *e.g.* *Faculté de Paris*. Plusieurs localisations peuvent occuper une même place. Par exemple, l'actuel bâtiment de l'*Hôtel de la Marine* à Paris a été occupé par différentes institutions en commençant par le *Garde-Meuble de la Couronne* et depuis 2015 par le *Centre des monuments nationaux*. Dans ce cas, le bâtiment (*place*) reste le même mais les deux institutions y sont localisées.

Le moment (d) désigne la dimension temporelle d'un évènement. Nous considérons deux types : (1) une date, dans ce cas l'évènement est dit ponctuel, *e.g.* *une naissance* et (2) un intervalle, on parle alors d'évènement continu, *e.g.* *un règne*.

Dans le cas de données prosopographiques, de nombreuses données peuvent être manquantes, incertaines ou incorrectes. Un évènement peut ne pas avoir de dates, de place ou de localisation. Une place peut ne pas être géolocalisée ou une localisation peut ne pas avoir de place définie, *e.g.* *un lieu de rassemblement dont la position n'a pas été consignée*. Par conséquent, nous n'imposons pas de contraintes sur la présence de date ou de lieu. La seule contrainte imposée est l'association entre l'individu (i) et l'évènement (e).

De manière à considérer les relations qui peuvent exister entre des individus, nous introduisons la notion de relation. Cette dernière est modélisée via un tuple $(i1, i2, l, E)$ avec $i1$ et $i2$ des individus, l un lieu et E des évènements. Tout comme pour le tuple précédent, la présence du lieu (l) et des évènements (E) n'est pas imposée. Un poids et un ordre sont associés à cette relation. Le poids permet d'indiquer la force de la relation ou son importance, alors que l'ordre ordonne les relations lorsqu'elles sont regroupés par rapport à l'individu choisi. Par la suite, on notera que deux individus sont *reliés* quand ils ont au moins une relation en commun.

Siprojuris

Afin de vérifier que notre modèle de données est bien adapté aux données prosopographiques, nous utilisons Siprojuris [5], un jeu de données disponible au format RDF depuis la base de données SYMOGIH [6].

Les données de Siprojuris décrivent la carrière de 566 enseignants de droit d'une période allant de 1800 à 1950. Différents évènements y sont présents, outre la carrière professionnelle, *i.e.* l'obtention de diplômes, les enseignements prodigués, les congés et la retraite. Y figurent aussi des déclarations d'état civil, *i.e.* la naissance et le décès. Pour un total de 7 940 évènements.

Dans la représentation de Siprojuris, on distingue les *Acteurs Collectifs* (*Collective Actors*) qui sont en général des institutions telles que la *Faculté de droit de Paris* et les *Lieux Naturels* (*Natural Place*) correspondant aux lieux géolocalisés. Ainsi, il est possible dans notre modèle de considérer les *Acteurs Collectifs* comme des *localisations* et les *Lieux Naturels* comme des *places*. Chaque *Acteur Collectif* possède un lien décrivant le *Lieu Naturel* correspondant à sa localisation ou son lieu de création, ces liens servent à positionner les localisations.

Les dates sont décrites sous plusieurs types : *date unique*, *date de début*, etc. Cependant, la plupart des *dates de début* d'un évènement continu sont décrites comme *date unique*, il est donc possible de considérer la quantité de dates associées à un évènement pour déterminer si celui-ci est un évènement ponctuel ou continu (respectivement une ou deux dates). Dans le cas où l'évènement est associé à plus de deux dates, l'intervalle le plus englobant est gardé. Enfin, dans le cas où l'évènement concerne plusieurs individus (*e.g.* une remise de doctorat concerne le professeur mais aussi l'élève qui reçoit la qualification), il est dupliqué pour être présent pour chacun de ces individus.

Hormis l'exemple précédent qui représente une très petite quantité de relations, Siprojuris ne possède pas originellement de données relationnelles entre les individus. Il est tout à fait possible, via le modèle de données et à partir des évènements stockés, de créer des relations entre individus en se basant sur la période durant laquelle deux individus ont été à la même *localisation*. Dans ce cas, le poids de la relation peut être mesuré en nombre d'années où deux individus ont enseigné à la même faculté. L'ordre des relations d'un individu est défini par rapport à la date médiane des années communes de chaque relation. Il est bien sûr possible pour deux individus d'être

5. http://siprojuris.symogih.org/
6. http://symogih.org/

connectés par plusieurs relations, chacune correspondant à une localisation différente, *e.g.* une à l'*Université de Paris* et une autre à la *Faculté de Lille*. D'autres types de relations peuvent bien entendu être considérés en fonction des besoins des historiens (*e.g.* même école de pensée).

Comme nous venons de le voir, le modèle de données que nous proposons offre suffisamment de souplesse pour que l'on puisse modéliser les données prosopographiques afin que celles-ci puissent être visualisées. Le modèle offre en plus la possibilité de considérer des relations très différentes. Cela permet, par exemple, pour un même jeu de données et en fonction des domaines d'intérêt des historiens de pouvoir visualiser des relations spécifiques qu'ils souhaitent étudier. Nous verrons, par la suite, que ces choix se font au moment de l'intégration des données dans la plateforme.

2.4 Conception de la plateforme ProsoVis

Dans cette section, nous répondons à la question *Comment ?* en décrivant les choix de conception que nous avons retenus, sur la base des besoins utilisateurs (*Pourquoi ?*) et de notre proposition pour la modélisation des données (*Quoi ?*), tous deux présentés à la section précédente. On peut noter que les réponses au *Comment ?* présentées sont le fruit d'un processus itératif en relation étroite avec les utilisateurs.

Le processus d'exploration que nous proposons pour la visualisation est basé sur le mantra[7] de SHNEIDERMAN (1996) : « *Proposer d'abord une vue d'ensemble, puis à l'aide de filtres et de zoom, offrir une vue détaillée* ». Dans notre contexte, l'objectif est donc d'offrir une vue d'ensemble des individus et/ou de leurs évènements au travers d'une *vue globale* puis de pouvoir, via une *vue détaillée*, visualiser un individu (*resp.* un évènement) ou un petit groupe d'individus (*resp.* d'évènements).

Chaque vue est elle-même composée de vues multiples coordonnées (*CMV – Coordinated Multiple Views*, ROBERTS, 2007), une technique répandue dans la visualisation de données complexes (GRIFFIN & ROBINSON, 2015 ; KEHRER & HAUSER, 2013 ; ROBERTS et al., 2019 ; SUN et al., 2021). L'utilisation de vues multiples coordonnées a été retenue pour les raisons suivantes :

- Profiter de visualisations adaptées mettant en évidence les différentes dimensions associées aux données, *i.e.* carte, frise chronologique, etc.
- Offrir une plus grande flexibilité dans la prise en compte des données d'entrées, *i.e.* connaître les caractéristiques spécifiques d'une visualisation (*e.g.* frise chronologique) permet de faciliter la spécification des besoins en entrée et donc offrir une approche générique pour les données prosopographiques.
- Fournir une visualisation similaire à celle que les utilisateurs ont l'habitude d'utiliser dans d'autres contextes.

Cela implique aussi qu'il est indispensable que les vues soient synchronisées et que les interactions d'une vue soient répercutées sur les autres, *e.g.* l'application d'un filtre

7. « *Overview first, zoom and filter, then details-on-demand* »

sur la frise chronologique doit automatiquement être réperctué sur la carte représentant les lieux associés.

La plateforme est développée sous la forme d'une application web utilisant React [8], D3 (Bostock et al., 2011), Leaflet [9] et vis-timeline [10]. L'interface est composée de trois vues principales : la *vue globale*, la *vue détaillée* et la *vue filtres*.

2.4.1 Vue globale

L'objectif de la vue globale est d'offrir une visualisation de l'ensemble du jeu de données **[B4]**. Elle est conçue pour accepter un certain niveau d'abstraction. Par exemple, elle permet de voir les régions de l'espace ou du temps qui contiennent le plus de données. Elle permet également à l'utilisateur de sélectionner les éléments qu'il souhaite analyser dans la vue détaillée (« *détail "à la demande"* »).

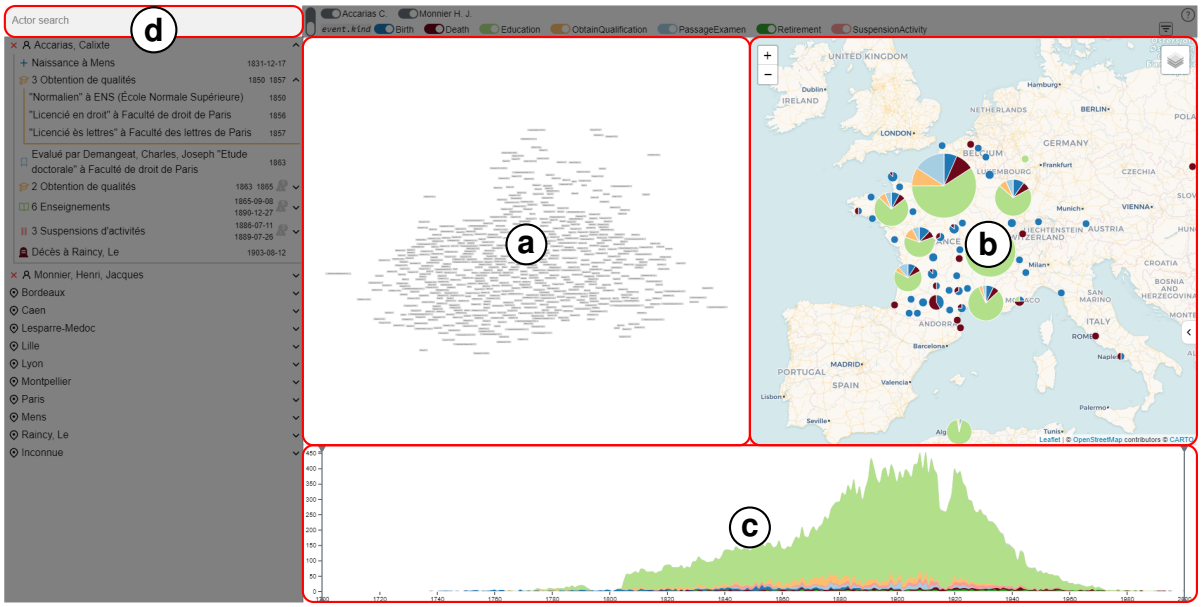


Figure 2.14 – *Aperçu de la vue globale. (a) le graphe des relations, (b) la carte, (c) la frise chronologique et (d) le champ de recherche.*

La vue est pensée pour prendre en charge les 3 dimensions des données prosopographiques : le temps, l'espace et les relations entre individus. Elle est composée de 4 sous-vues :

- La vue *graphe* (cf. Figure 2.14(a)) représente les relations entre les individus.
- La *carte* (cf. Figure 2.14(b)) affiche l'ensemble des évènements dont la localisation est connue.
- La *frise chronologique* (, cf. Figure 2.14(c)) montre la répartition des évènements au cours du temps.

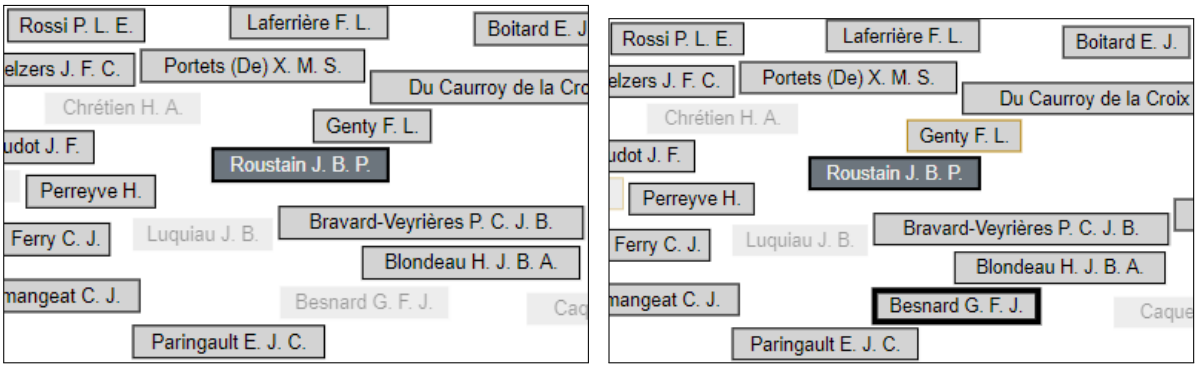
8. https://reactjs.org/
9. https://leafletjs.com
10. https://github.com/visjs/vis-timeline

- Le *champ de recherche* (cf. Figure 2.14(d)) permet de rechercher des individus pour les ajouter à l'exploration détaillée.

Graphe : représentation des individus

Pour représenter les individus **[B1.3]**, nous avons choisi une représentation sous la forme d'un graphe où les nœuds correspondent aux individus et les arêtes à leurs relations. Chaque nœud est représenté par un rectangle qui contient le nom de l'individu. Nous avons vu précédemment (cf. sous-section 2.3.2) que le modèle de données offrait beaucoup de souplesse pour définir, en fonction des besoins des historiens, différents types de relations. Le choix de cette représentation a été guidé par le fait que le graphe permet par sa spatialisation de retranscrire, de façon intuitive pour l'utilisateur, la proximité associée à une relation entre deux individus **[B5]**, *e.g.* deux individus proches en considérant la relation retenue par l'utilisateur seront également proches dans la visualisation du graphe. Nous verrons ultérieurement (cf. Chapitre 3) que cette présentation soulève des problèmes de chevauchements des nœuds liés aux interactions offertes.

L'exploration du graphe peut se faire en se déplaçant ou en zoomant. La sélection d'un individu permet de mettre en évidence l'ensemble des individus auxquels il est relié comme l'illustre la Figure 2.15(a). Dans cette exemple, une fois l'individu sélectionné (ici *Roustain J. B. P.*), ce dernier apparaît en gris foncé, les individus auxquels il est relié apparaissent en gris plus clair et les autres sont estompés. Qu'un individu soit sélectionné ou non, il est toujours possible de le survoler. Par exemple dans la Figure 2.15(b), l'individu survolé, ici *Besnard G. F. J.*, est mis en évidence avec une bordure en gras et les individus qui lui sont reliés avec une bordure jaune (*e.g.* *Genty F. L.*). En combinant les interactions (cf. Figure 2.15(b)) il est donc possible de visualiser pour deux individus, l'ensemble des individus communs, ceux connus par un seul des deux ou bien ceux inconnus des deux.



(a) (b)

FIGURE 2.15 – *Interactions sur le graphe.* (*a*) *Sélection de* Roustain J. B. P. (*au centre en gris*) *et* (*b*) *Survol de* Besnard G. F. J. (*en bas*) *avec* Roustain J. B. P. (*au centre en gris*) *sélectionné précédemment.*

Un clic droit de la souris affiche un aperçu résumé des évènements associés à un individu ainsi que la possibilité de l'ajouter à la vue détaillée via l'icône + (cf. Figure 2.16).

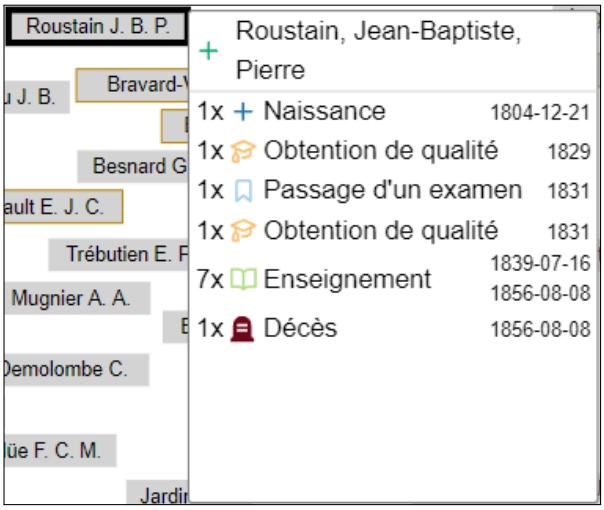
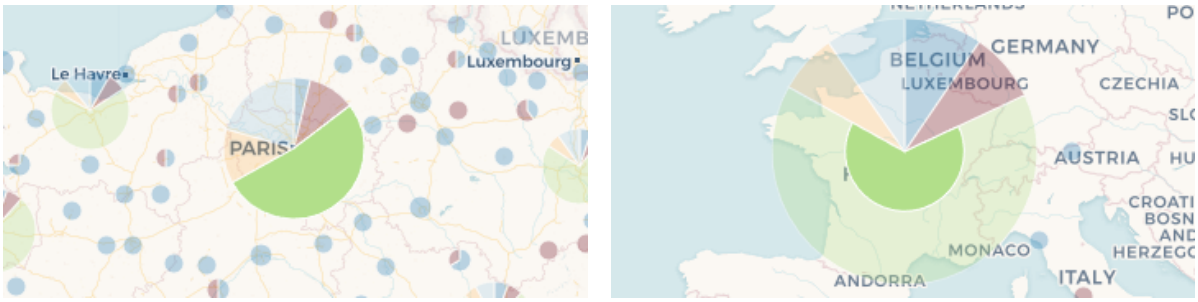


Figure 2.16 – *Un résumé des évènements associés à un individu, ici* Roustain J. B. P.

Carte : représentation spatiale des évènements

Pour représenter la dimension spatiale des évènements **[B1.2]** nous utilisons une carte où chaque évènement est représenté par un disque (cf. Figure 2.14(b)). Bien entendu, seuls les évènements possédant une localisation dans les données sont affichés.

Vouloir représenter l'ensemble des données sur une même carte engendre un problème de surcharge visuelle, étant donné que le nombre d'évènements, pour un lieu, peut être important. Nous utilisons donc un algorithme de *Regroupement Spatial Agglomératif*, *F-SAC* présenté dans le Chapitre 4, dont l'objectif est de proposer des clusters d'évènements proches. Un cluster est représenté sous la forme d'un camembert où chaque part représente une catégorie d'évènements (*e.g.* enseignement, obtention de diplôme, etc.) et où l'aire de chaque part est proportionnelle au nombre d'évènements qui la composent.



(**a**) *Sélection initiale* (**b**) *La même sélection avec vue sur la France*

Figure 2.17 – *Sélection d'une partie des évènements qui se sont produits à Paris et ses environs*

Il est possible de sélectionner les évènements en survolant ou en sélectionnant les parts d'un camembert comme l'illustre la Figure 2.17(a). La part des évènements non sélectionnés est alors rendue semi-transparente. Contrairement au survol, la sélection d'un ensemble persiste aux actions de navigations sur la carte (*e.g.* zoom). Ceci permet, par exemple, de mettre en évidence la part des évènements en France qui se produisent

à Paris (cf. Figure 2.17(b)).

Frise chronologique : représentation temporelle des évènements

Pour représenter la dimension temporelle des évènements **[B1.1]** nous utilisons un graphique en aires empilées (*stacked area chart*) (cf. Figure 2.14(c)) dans lequel chaque aire représente un type d'évènement. Cette visualisation est efficace pour montrer la distribution des évènements dans le temps tout en occupant peu d'espace à l'écran. D'autres types de représentations sont cependant possibles et un lecteur intéressé peut se référer à Brehmer et al. (2017), Cuenca et al. (2018) et Khulusi et al. (2019).

Nous avons vu que, dans les données prosopographiques, les évènements pouvaient correspondre à une date précise ou bien à une période. Pour offrir une vue informative et pour que l'ensemble des évènements soit représenté, ces derniers sont discrétisés, *i.e.* nous considérons qu'un évènement se produit durant au moins une année.

Un filtre peut être appliqué grâce à deux barres glissantes initialement situées de part et d'autre de la vue (cf. Figure 2.14(c)). Ces dernières permettent de spécifier un intervalle de temps. Tous les évènements n'étant pas dans l'intervalle sont masqués. Dans le cas où un évènement est continu, il est masqué seulement si aucune partie de cet évènement ne figure dans l'intervalle conservé. Les évènements masqués sont représentés en gris comme illustré dans la Figure 2.18.

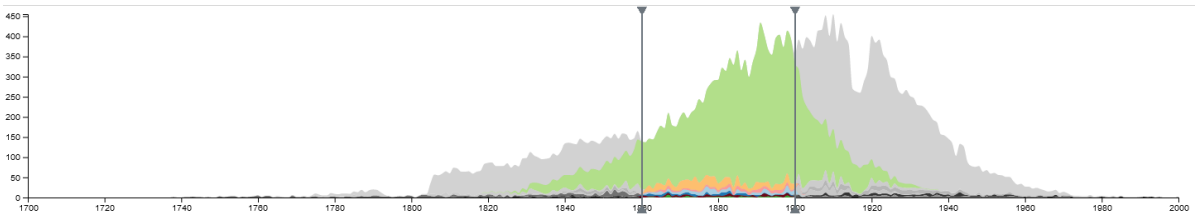


Figure 2.18 – *Aperçu de la frise chronologique de la vue globale avec un filtre temporel de 1860 à 1900*

Champ de recherche : détails « à la demande »

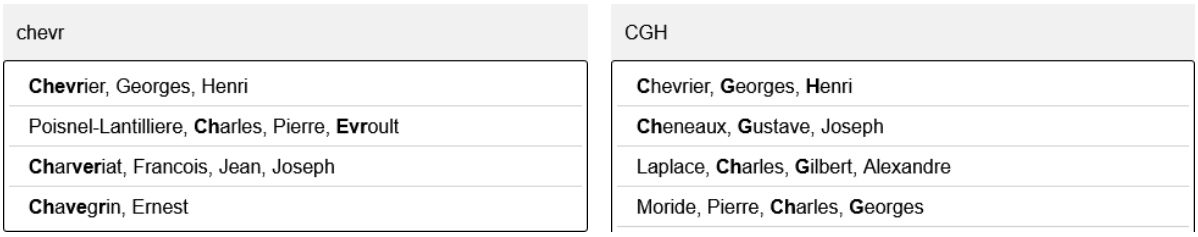
Pour répondre au besoin du « détail "à la demande" » **[B4]**, nous modélisons deux types de comportements.

La sélection d'un petit ensemble d'individus. Elle est obtenue via des filtres successifs appliqués aux données. Cette sélection se fait via le graphe de la vue globale où un individu peut être sélectionné puis ajouté à la vue détaillée.

La sélection d'un individu particulier. Même si le graphe permet, en naviguant, de retrouver les individus, cette approche n'est pas pratique dès lors que le graphe est grand. Aussi, nous proposons d'ajouter un champ de recherche (cf. Figure 2.14(d)) à la vue globale pour sélectionner des individus via leur nom et les ajouter directement dans la vue détaillée.

Le champ de recherche (cf. Figure 2.14(d)) offre la possibilité d'effectuer des recherches via les premières initiales du nom et par complétion en considérant l'ensemble

des individus (cf. Figure 2.19). Lorsqu'un individu est sélectionné, il est alors automatiquement ajouté à la vue détaillée.



(**a**) *Exemple de recherche* (**b**) *Exemple de recherche par initiales*

Figure 2.19 – *Aperçu du champ de recherche*

2.4.2 Vue détaillée

Une fois les individus sélectionnés via la vue globale, l'objectif de la *vue détaillée* est de présenter l'ensemble des informations les concernants disponibles dans la base ainsi que les relations partagées entre ces individus. Globalement la *vue détaillée* est structurée de la même manière que la vue globale (*e.g.* graphe de relations, carte, frise chronologique) toutefois la façon de considérer les données est différente car nous affichons les détails dans chacune des vues.

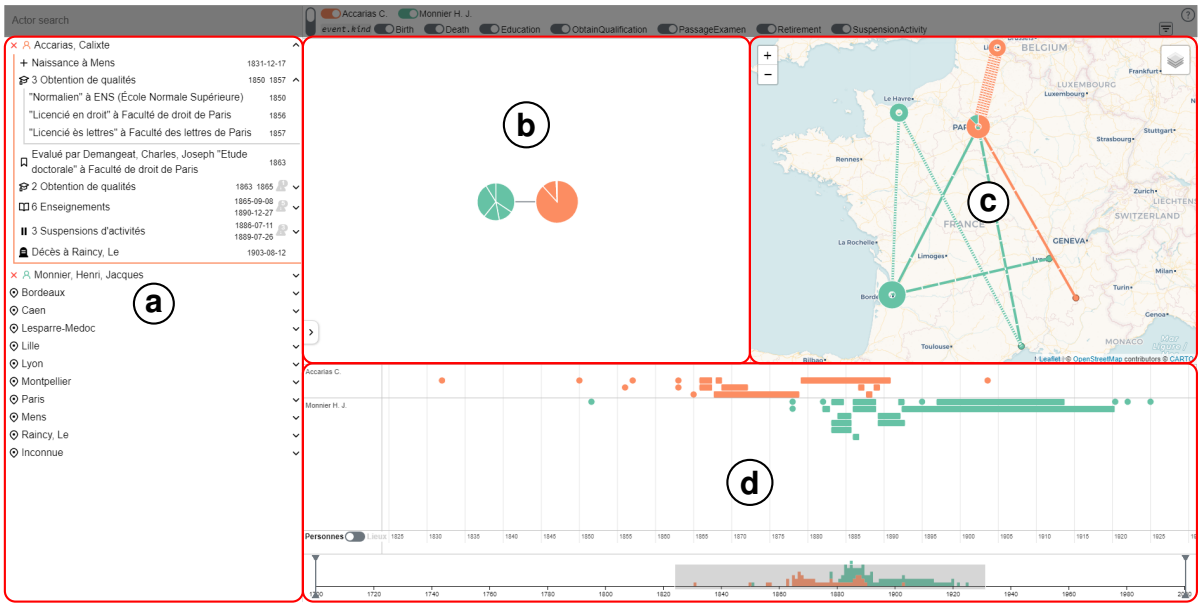


Figure 2.20 – *Aperçu de la vue détaillée avec deux individus* Accarias C. (*en orange*) *et* Monnier H. J. (*en vert*). *Elle est composée de* (*a*) *le panneau d'information,* (*b*) *le graphe des relations,* (*c*) *la carte et* (*d*) *la frise chronologique.*

La *vue détaillée* (cf. Figure 2.20) se décompose en 4 sous-vues :

- La vue *panneau d'information* (cf. Figure 2.20(a)) présente les informations détaillées de chaque individu.

- La vue des *relations* (cf. Figure 2.20(b)) décrit les différents relations que les individus sélectionnés partagent.
- La *carte* (cf. Figure 2.20(c)) affiche les évènements localisés des individus sélectionnés mais également les trajectoires au cours du temps.
- La *frise chronologique* (cf. Figure 2.20(d)) affiche tous les évènements des individus qui possèdent une indication temporelle.

Graphe des relations : représentation des individus et de leurs relations

L'objectif de cette vue est double : (1) afficher les relations entres les individus ajoutés à la vue détaillée **[B1.3]** et (2) étendre la liste des individus explorés à l'aide d'une navigation de proche en proche des relations des individus déjà représentés **[B5]**. Pour cela, nous avons conçu une technique de visualisation basée sur un dessin de graphe où un nœud correspond à un individu et une arête une relation entre deux individus. Pour permettre la navigation de proche en proche, il est possible d'étendre de manière interactive ce graphe en fonction des relations, avec des individus ne faisant pas partie de l'exploration détaillée. Le nombre d'individus à afficher étant beaucoup moins important que dans le graphe de la vue globale, un plus grand nombre d'informations par individu peut être affiché.

Chaque individu est représenté sous la forme d'un camembert avec une couleur associée (cf. Figure 2.21(b)), *i.e.* nous verrons dans la *vue filtres* que lorsque la sélection concerne un individu ou des évènements spécifiques, une couleur est affectée afin de faciliter, pour l'utilisateur, sa reconnaissance au sein des différentes vues. Deux camemberts sont reliés par une arête si les deux individus qu'ils représentent ont une relation en commun. Chaque part du camembert correspond à un lieu. La taille de cette part est proportionelle au nombre d'individus reliés à l'individu représenté par le camembert.

Lorsqu'une part du camembert est sélectionnée (cf. Figure 2.21(a)), les individus reliés sont disposés en cercle autour du graphe des relations. Ce cercle, appelé *cercle des suggestions*, permet d'afficher les individus qui n'ont pas été sélectionnés dans l'exploration détaillée et pour lesquels il existe des relations. La couleur du cercle des suggestions est fonction de l'individu sélectionné. Les suggestions sont ordonnées dans le sens horaire par un ordre défini en amont dans les données. Chaque suggestion, *i.e.* individu suggéré, est représenté par un disque et une barre dont la taille correspond au poids de la relation préalablement défini en amont dans les données. S'il est relié à un autre individu que celui sélectionné, la relation est alors représentée par une arête (cf. Figure 2.21(c)). Le survol d'une suggestion affiche un label contenant le nom de l'individu et le symbole « + » permet de l'ajouter dans l'exploration détaillée. Par exemple, (cf. Figure 2.21(a)), la partie du camembert correspondant à la *Faculté de droit de Lille* pour *Accarias C.* est sélectionnée. Á l'aide du « + », il est possible d'ajouter un nouvel individu qui n'était pas présent dans la vue détaillée, *i.e. Mabire H. C. J. M.*. Celui-ci est alors automatiquement ajouté (cf. Figure 2.21(b)) avec les liens associés dans le graphe de relations.

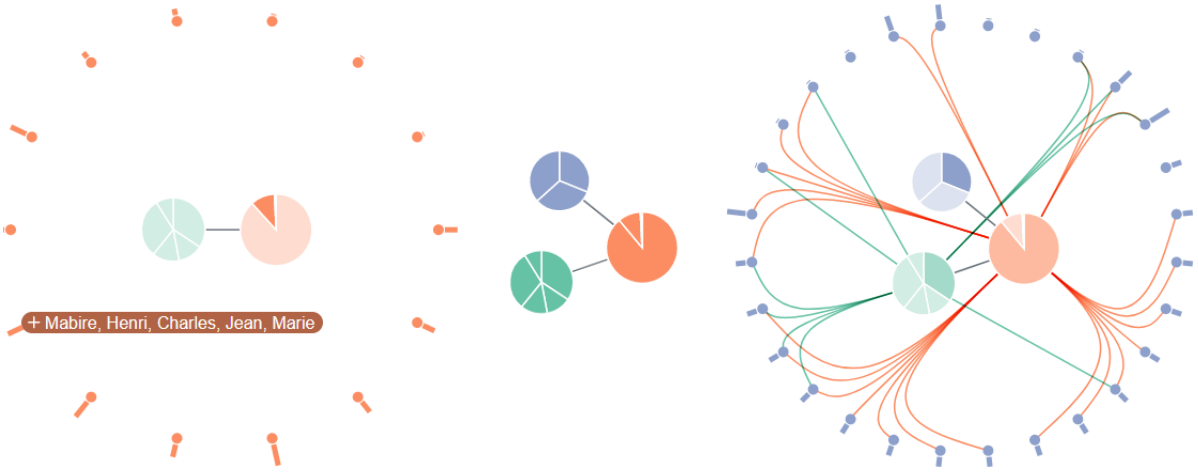


Figure 2.21 – *Exemple d'utilisation du graphe des relations.* (*a*) *Affichage des suggestions pour* Accarias C. (*en orange*) *à la* Faculté de droit de Lille *avec le survol d'une suggestion* (Mabire H. C. J. M., *en bas à gauche de la figure*) (*b*) *Ajout de* Mabire H. C. J. M. (*en violet*) (*c*) *Suggestions pour* Mabire H. C. J. M. *à la* Faculté de droit de Paris.

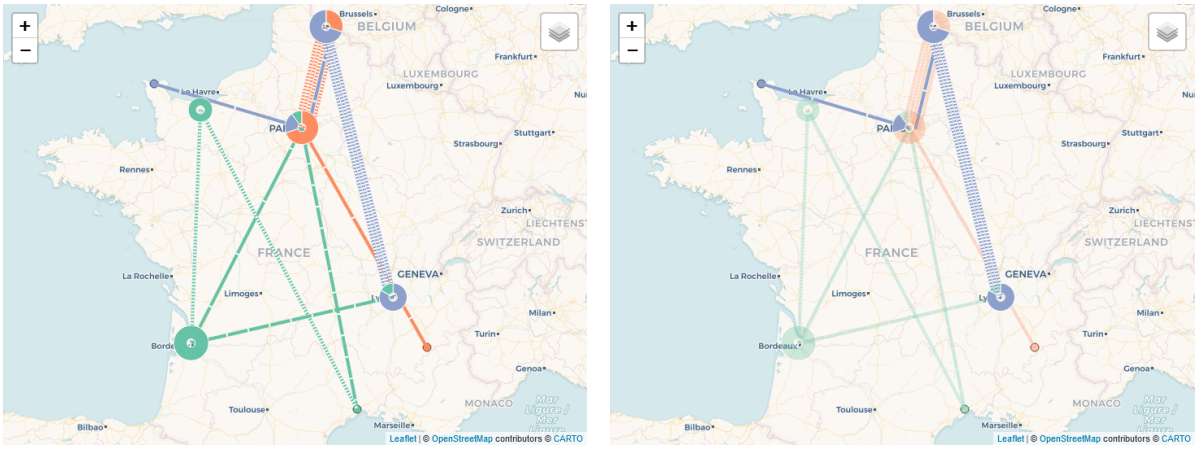
Carte : représentation spatiale des évènements et du mouvement des individus

Comme pour la vue globale, les évènements sont représentés sur une carte **[B1.2]**. Additionnellement, afin de mettre en évidence les déplacements des individus associés à ces évènements, nous affichons leurs tracés au cours du temps et des différents lieux **[B1.3]**. De manière à mettre en évidence des périodes de temps plus ou moins longues, différents types de tracés sont considérés. Outre le fait que ces tracés permettent d'avoir un aperçu général du parcours d'un individu, ils sont aussi utiles pour mettre en évidence le temps écoulé entre deux lieux du parcours d'un individu **[B1.2]**.

La carte détaillée (cf. Figure 2.20(c)) affiche tous les évènements localisés des individus ajoutés à l'exploration détaillée. Les évènements trop proches sont regroupés et le groupe est représenté via un marqueur sous la forme d'un beignet (*donut*) partitionné où chaque part est proportionnelle à la quantité d'évènements. Les évènements non regroupés sont représentés par un marqueur en forme de disque. Cette différence de présentation permet de rapidement distinguer un évènement individuel d'un regroupement.

Les tracés représentent le parcours d'un individu. Ils sont hachurés en fonction de l'écart de temps entre un évènement à un marqueur donné et un autre évènement à un marqueur différent. Plus les hachures sont denses, plus l'écart de temps est faible. Dans le cas où plusieurs tracés sont présents pour une paire de marqueurs, les tracés sont mis côte à côte afin qu'ils soient tous visibles. Cela se produit notamment lorsqu'un individu effectue des allers-retours réguliers entre deux marqueurs. Par exemple, la Figure 2.22(a) montre que *Accarias C.* (en orange) a effectué plusieurs aller-retours entre *Paris* et *Lille*. L'écart entre les dates des évènements est très court (les tracés sont très hachurés) et cela peut signifier que ces allers-retours étaient rapides.

Le survol ou la sélection d'un disque ou d'une part d'un beignet estompe tous



(**a**) *Détail des 3 individus à l'échelle de la France* (**b**) *Sélection du parcours de* Mabire H. C. J. M.

FIGURE 2.22 – *Exemple de carte détaillée avec les 3 individus :* Accarias C. *(en orange)*, Mabire H. C. J. M. *(en violet) et* Monnier H. J. *(en vert)*

les autres marqueurs et tracés de la carte. La même action effectuée sur un tracé met en évidence le parcours de l'individu lié au tracé tout en estompant le reste (cf. Figure 2.22(b)).

Différents fond de cartes peuvent être appliqués et il également possible de masquer les tracés et/ou les marqueurs. Ces options sont accessibles en survolant le bouton calques en haut à droite de la Figure 2.22(a).

Frise chronologique : représentation temporelle des évènements des individus

De manière à permettre de suivre, au cours du temps, les différents évènements associés à un individu **[B1.1]**, nous proposons d'utiliser une structure *focus+contexte* (COCKBURN et al., 2009) qui offre la possibilité d'avoir une vue d'ensemble tout en surmontant la perte du contexte lors de la navigation avec le zoom.

Ainsi la frise (cf. Figure 2.20(d)) est divisée en deux sous-parties (cf. Figures 2.23 et 2.24). La partie haute détaille les évènements individuels (*focus*). La partie basse récapitule la répartition des évènements de tous les individus (*contexte*).

Vue *focus*

Dans la vue *focus* (cf. Figure 2.23), les évènements ponctuels (*e.g.* une naissance) sont représentés par des disques, les évènements continus (*e.g.* un poste occupé par un individu sur une période donnée) par des rectangles dont la longueur encode la durée de l'évènement. Ces deux formes sont simples et facilement distinguables (WARE, 2021). D'autres alternatives ont été explorées pour représenter les évènements ponctuels comme les carrés ou les triangles. Cependant, les carrés peuvent être confondus avec des rectangles et les triangles donnent une impression de direction, *i.e.* vers le haut. Les rectangles ont l'avantage de pouvoir être étirables sur une dimension, les lignes peuvent aussi être utilisées mais sont difficilement sélectionnables car trop fines. Deux rectangles côte à côte peuvent être perçus comme un seul (BERTIN, 2011), par conséquent comme dans HAN et al. (2021) nous proposons que les coins des rectangles soient

légèrement arrondis.

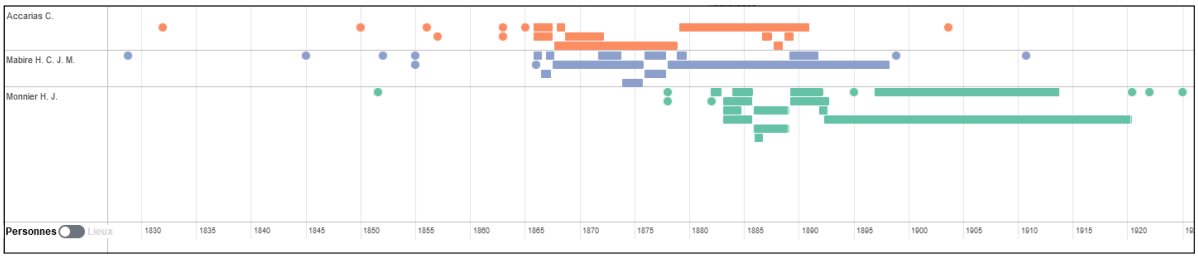


Figure 2.23 – *Partie haute (vue focus) de la frise chronologique détaillée pour les 3 acteurs :* Accarias C. *(en orange)*, Mabire H. C. J. M. *(en violet) et* Monnier H. J. *(en vert)*

Les évènements sont mis l'un en dessous de l'autre lorsqu'ils se chevauchent et sont verticalement regroupés par individu ou par lieu. Il est possible de passer d'un regroupement à l'autre en utilisant l'interrupteur *Personnes – Lieux* situé en bas à gauche de la frise. Par exemple, dans la Figure 2.23, de haut en bas, apparaissent respectivement les évènements de *Accarias C.* (en orange), de *Mabire H. C. J. M.* (en violet) et enfin de *Monnier H. J.* (en vert).

La frise est bien entendu interactive. Il est donc possible de se déplacer et de zoomer dans la fenêtre afin de changer l'intervalle de temps affiché. Ces actions n'affectent que l'axe x qui encode le temps.

Un évènement peut être sélectionné, le reste des évènements est alors estompé. Lors du survol, le label associé à l'évènement est affiché.

L'ensemble des évènements d'un individu ou d'un lieu peut être sélectionné en cliquant sur les cases situées à gauche de la Figure 2.23 et contenant le label du lieu et de l'individu. De la même manière que pour la sélection individuelle, les éléments non concernés sont estompés.

Vue *contexte*

Dans la vue *contexte* (cf. Figure 2.24), un graphe en aires empilées est utilisé, similaire à celui de la frise chronologique de la vue globale. Il illustre l'intervalle affiché par la partie haute sous la forme d'une fenêtre (rectangle gris semi-transparent). Cela permet de préciser le contexte de la partie haute par rapport à l'ensemble des données temporelles de l'exploration détaillée.

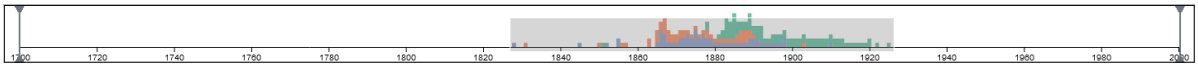


Figure 2.24 – *Partie basse (vue contexte) de la frise chronologique détaillée pour les 3 acteurs :* Accarias C. *(en orange)*, Mabire H. C. J. M. *(en violet) et* Monnier H. J. *(en vert)*

Les deux vues sont synchronisées : le déplacement de la fenêtre dans la partie basse se répercute sur la partie haute en affichant l'intervalle correspondant.

Un filtre peut être appliqué via les deux barres situées de part et d'autre de la frise (cf. Figure 2.24). Similaire à celui dans la vue globale, tous les évènements n'étant pas dans l'intervalle sont masqués (cf. Figure 2.25). Dans le cas où l'évènement est

continu, il est masqué seulement si aucune partie de cet évènement ne figure dans l'intervalle conservé. Lorsqu'un filtre est défini, des barres sont aussi représentées dans la partie haute (cf. Figure 2.25). Elles sont synchronisées avec les barres de la partie basse. Déplacer les barres dans l'une ou l'autre des parties, applique le filtre et propage les mises à jour. En revanche, le graphique en aire ne subit pas l'effet du filtre afin de préserver une vue *contexte* des données temporelles.

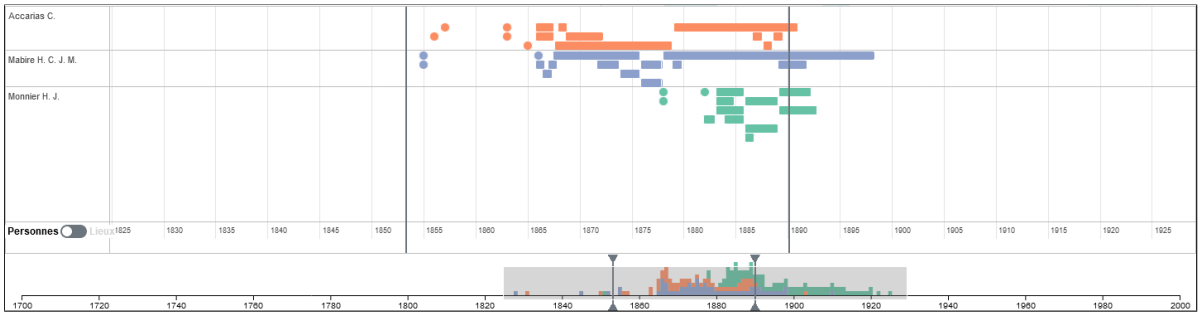


FIGURE 2.25 – *Frise chronologique détaillée avec un filtre temporel de 1854 à 1890 et les 3 acteurs :* Accarias C. (*en orange*), Mabire H. C. J. M. (*en violet*) *et* Monnier H. J. (*en vert*)

Panneau d'information : affichage des ressources

De manière à détailler l'ensemble des informations associées aux individus sélectionnés **[B1]**, nous proposons une vue spécifique (cf. Figure 2.20(a)). Pour visualiser tous les évènements associés à un individu, nous regroupons les informations associées à la fois par évènement mais aussi par lieu. Lors d'une analyse, si des éléments sélectionnés peuvent être filtrés, nous affichons tout de même tous les éléments sélectionnés (même ceux éliminés par le filtre). Dans ce cas, pour rappeler à l'utilisateur qu'il a appliqué un filtre après une sélection, il ne peut pas agir sur les éléments filtrés tant que le filtre n'a pas été désactivé. Nous utilisons également cette vue pour mettre en évidence des erreurs dans les données **[B6]**.

L'ensemble des évènements est ordonné par ordre chronologique et rassemblé pour un individu (symbolisé par un buste) ou un lieu (symbolisé par une épingle). De cette manière, il est possible de voir rapidement tous les évènements qui concernent un individu ou qui se sont produits dans un lieu donné. Si le lieu d'un évènement est inconnu, ce dernier est placé dans un lieu nommé *Inconnue*. Le fait de regrouper les évènements non localisés permet de mettre en évidence pour l'utilisateur des informations manquantes dans les données. Les groupes sont ordonnés par ordre alphabétique, en commençant par les individus puis les lieux. Le lieu inconnu est toujours placé à la fin.

Les évènements ont un niveau de regroupement supplémentaire : leur catégorie. Ceci permet de faciliter la navigation toujours dans une optique de réduction d'encombrement visuel. Ainsi, les évènements de même catégorie et chronologiquement consécutifs sont regroupés. Le groupe est désigné par la catégorie agglomérante et l'échelle de temps correspond à l'intervalle temporel minimal des évènements qu'il contient. Pour les évènements continus, la date déterminant l'agglomération correspond à la date de début. Il se peut donc que la date de fin d'un groupe ne précède pas

nécessairement la date du prochain élément de la liste.

Par exemple, la Figure 2.26 illustre différents individus et lieux. *Accarias C.*, un des individus sélectionné pour l'exploration détaillée, est déroulé pour afficher ses évènements. Nous pouvons constater que parmi ceux-ci, certains sont regroupés dans différentes catégories, dont *Obtention de qualités*, qui est déroulé pour afficher les deux qualités obtenues.

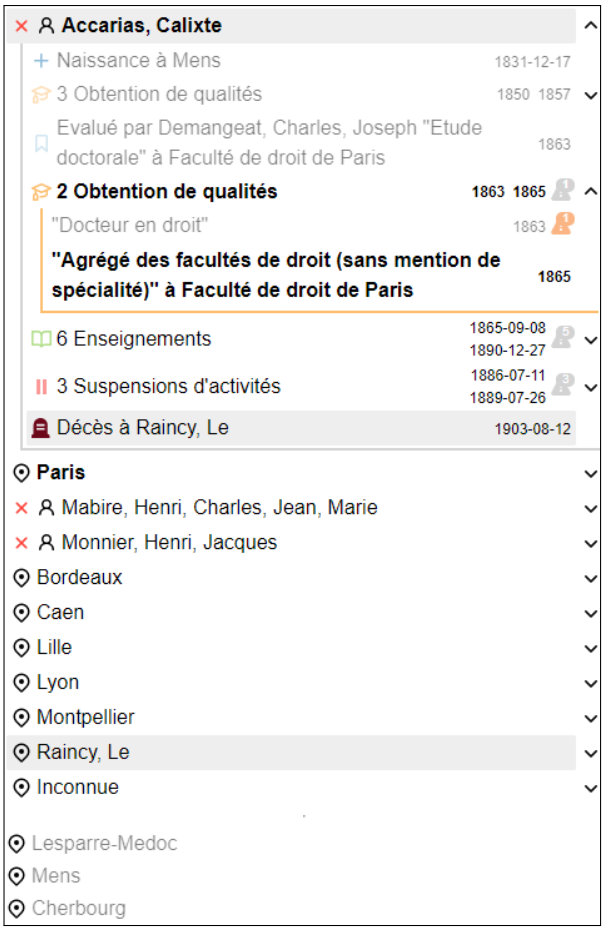


Figure 2.26 – *Aperçu du panneau information. Les évènements d'*Accarias C. *sont affichés. Les 3 lieux en bas apparaissent estompés suite au filtre appliqué au groupe «* 3 Obtention de qualités *» qui apparaît aussi estompé car tous ses éléments sont filtrés.*

Nous pouvons remarquer sur la Figure 2.26, que différents effets peuvent apparaître sur le texte affiché, *i.e.* estompé, gras ou surligné en gris. Ces derniers correspondent aux différents types d'interactions possibles. Les éléments estompés indiquent que ceux-ci ont été éliminés par un filtre et ne sont donc pas affichés dans les autres vues. Un groupe est estompé si l'ensemble des éléments qui le composent le sont. Par exemple, toujours dans la Figure 2.26, les 3 lieux en bas de la figure apparaissent estompés car un filtre a été appliqué. Les éléments en gras correspondent aux éléments sélectionnés. Un groupe apparaît en gras si au moins l'un des éléments qui le compose est sélectionné. Ceci permet, d'un coup d'œil, de connaître les groupes pour lesquels un des éléments est sélectionné sans avoir à le dérouler. Lors d'un survol, les éléments apparaissent sous la forme d'un surligné. La propagation de l'information de survol est organisée

de la même manière que pour la sélection, par exemple dans la Figure 2.26, le survol de l'évènement de décès met en surligné *Accarias C.*, l'individu concerné mais aussi le lieu : *Raincy, Le*.

Les interactions ont un effet sur l'ordre de positionnement des éléments dans le panneau. L'ordre initial est défini de la manière suivante. Les individus et les lieux sélectionnés sont positionnés en premier, puis ceux qui ne sont pas sélectionnés et enfin ceux qui sont filtrés. Un survol n'affecte pas l'ordre.

Un évènement, un individu ou un lieu peuvent être sélectionnés. Le fait de sélectionner un individu ou un lieu engendre la sélection de l'ensemble des évènements qui lui sont associés. Il n'est pas possible de sélectionner les sous-groupes définis par les catégories d'évènements. Le survol peut être effectué sur l'ensemble des éléments, que ce soit les groupes, les sous-groupes ou les évènements. Enfin, il est possible de supprimer les individus de l'exploration détaillée à l'aide de la croix rouge figurant à gauche du groupe correspondant.

Affichage et détection des incohérences

Sur la Figure 2.26, des icônes orange avec un compteur sont présents à droite de certains évènements. Ces dernières permettent d'afficher des informations complémentaires associées aux données liées à l'évènement. Une icône rouge signale une erreur, une orange un avertissement et une bleue un complément d'information. Par exemple, dans la figure Figure 2.27, l'icône rouge associée à l'évènement qui s'est produit à Lyon nous informe qu'il y a une erreur dans les données. Un clic droit de la souris permet de faire apparaître une info-bulle donnant plus de détails sur le problème.

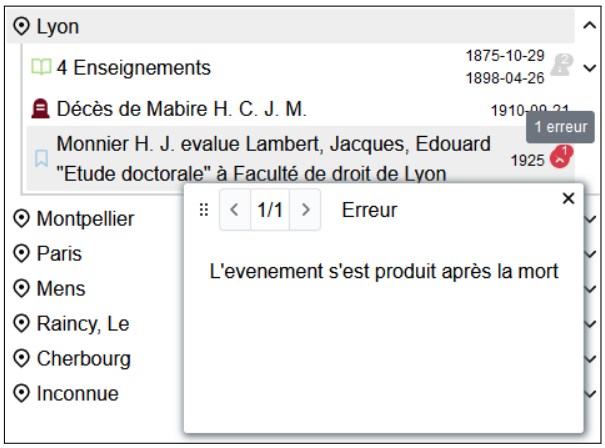


FIGURE 2.27 – *Exemple d'une erreur*

Dans le cas où plusieurs erreurs se produisent pour un même évènement, il est possible d'afficher le message approprié en naviguant avec les flèches en haut à gauche de l'info-bulle. Les sous groupes affichent la somme des signalisations des évènements contenus mais sont estompés car non interagissables. Si des signalisations de différents types apparaissent dans des éléments d'un sous groupe, l'icône retenue est celle de la plus grande sévérité (*i.e.* l'ordre est rouge, orange puis bleu). Le fait de faire apparaître cette information au niveau du sous groupe permet d'informer l'utilisateur qu'il y a des erreurs ou des avertissements dans les éléments qui le composent.

2.4.3 Filtres

Précédemment, nous avons présenté différentes vues pour visualiser les données via leurs différentes dimensions spatio-temporelles ou relationnelles. Cependant pour aider l'utilisateur, il est indispensable de proposer des mécanismes pour affiner sa recherche **[B2, 3]**. Nous proposons pour cela d'utiliser des mécanismes de filtres. Ces derniers permettent de considérer les attributs des évènements ou des individus via notamment les lieux ou les dates. Bien entendu, ces filtres sont propagées sur l'ensemble des vues.

La vue filtres (cf. Figure 2.28) est située en « en-tête » de la plateforme et est commune pour la vue *globale* ou *détaillée*. Son objectif est double. D'une part, elle permet de filtrer les données. D'autre part, elle permet de définir comment les évènements sont regroupés.

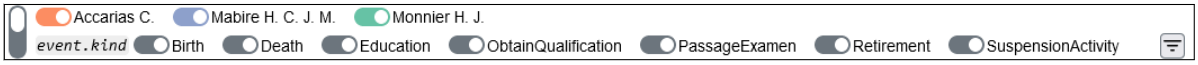


Figure 2.28 – *Aperçu de la vue Filtres. Le regroupement par individu est activé. Il y en a 3,* Accarias C. (*en orange*), Mabire H. C. J. M. (*en violet*) *et* Monnier H. J. (*en vert*).

Dans le cas du regroupement, les groupes sont encodés à l'aide de couleurs, *i.e.* chaque couleur définit un groupe. Par exemple, jusqu'à présent dans les précédentes illustrations de la vue détaillée (cf. Figures 2.20 à 2.26), nous considérions principalement les individus (*Accarias C.*, *Mabire H. C. J. M.* et *Monnier H. J.*) : une couleur était associée à chacun des « groupes », *i.e.* à chaque individu. Cette vue, via l'interrupteur (*switch*) situé à gauche dans la Figure 2.28, permet de définir la stratégie de regroupement : les évènements sont regroupés soit par individu soit par un attribut associé (*e.g.* Birth, Death, Education, etc). Bien entendu les couleurs dans toutes les vues seront associées aux choix du regroupement effectué.

Le Tableau 2.2 illustre comment chaque vue réagit par rapport au regroupement et à la coloration. Ainsi, un regroupement correspond à un ensemble d'éléments mais également au choix d'une couleur associée à ce groupe. Les astérisques précisent un comportement particulier : la carte et la chronologie de la vue globale conservent le partitionnement par attribut. Pour le cas du graphe de la vue globale (*oui**), seuls les individus de l'exploration détaillée sont coloriés afin de les mettre en évidence. Dans le cas de la coloration par attributs de la carte de la vue détaillée, les tracés des individus sont grisés (non coloriés).

Comme nous l'avons précisé précédemment, cette vue permet également de filtrer les évènements. Pour cela un interrupteur est situé à gauche de chaque label. Lorsque l'interrupteur est désactivé, l'interrupteur et le label sont estompés et toutes les vues sont mises à jour soit en éliminant l'évènement associé, soit en le grisant comme nous l'avons explicité précédemment dans la description des vues.

Des filtres supplémentaires peuvent être ajoutés via l'onglet situé en bas à droite (forme d'entonnoir) de la Figure 2.28. Cet onglet (cf. Figure 2.29), via un champ de texte et un bouton de validation, permet d'affiner le filtrage grâce à l'ajout d'attributs et de sélections associées aux valeurs des données.

Table 2.2 – *Application du regroupement* (Groupe) *ou de la coloration* (Couleur)*, pour le regroupement par individu ou par attribut sur les différentes vues de l'interface*

		Individu		Attribut	
		Groupe	Couleur	Groupe	Couleur
Vue globale	Graphe		oui*		
	Carte	*		oui	oui
	Chronologie	*		oui	oui
Vue détaillée	Graphe		oui		
	Carte	oui	oui	oui	sauf tracés
	Chronologie		oui		oui
	Informations		oui		oui

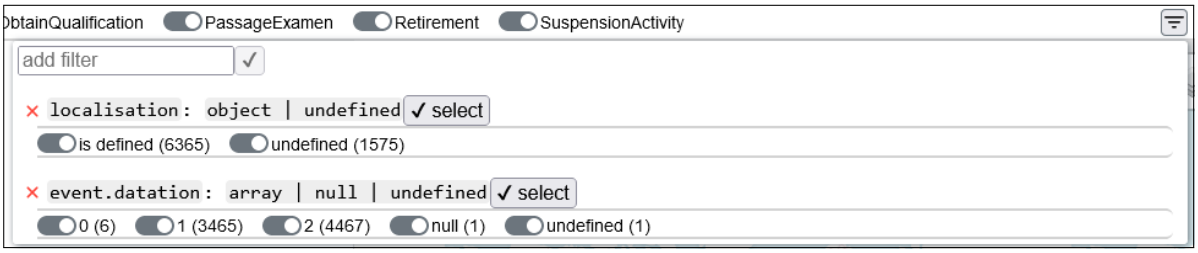


Figure 2.29 – *Listes de filtres supplémentaires. Deux attributs ont été ajoutés comme filtres supplémentaires : la localisation d'un évènement et le nombre de dates d'un évènement.*

Par exemple, dans la Figure 2.29, nous pouvons constater que deux attributs ont été ajoutés, `localisation`, qui correspond aux lieux des évènements, et `event.datation`, qui correspond aux dates des évènements. A côté de chaque attribut apparaît la liste des types [11]. Par exemple pour `localisation`, le type associé est *object*. Par la suite en fonction du type, les différentes valeurs possibles sont proposées afin de pouvoir filtrer les données. Par exemple pour *localisation*, de la même manière que pour un tableau comme *event.datation*, c'est respectivement la présence/absence de l'objet ou l'arité du tableau qui sont affichés. Si l'attribut n'existe pas ou est défini comme nul, le groupe respectif (*undefined* ou *null*) est ajouté à la liste des choix possibles. La sélection de la valeur se fait à l'aide d'interrupteurs et un bouton *select* offre la possibilité de le définir comme l'attribut utilisé pour la coloration et le regroupement. Un filtre est supprimé à l'aide de la croix rouge située à gauche des attributs.

2.4.4 Discussion préliminaire

Dans cette partie nous avons présenté les choix de conceptions et le fonctionnement de chacune des vues. Il est difficile de pouvoir précisément décrire l'impact réel de

11. Nous considérons ici la notion de type dans le sens « informatique » du terme. Plus précisément il s'agit des types récurrents dans la plupart des langages : `object`, `array`, `string`, `number`, `null` et `undefined`.

chacune des vues sur la satisfaction des besoins car elles sont coordonnées et les considérer individuellement est dans certains cas impossible. Par exemple, étant donné que la vue filtres définit la façon de regrouper les éléments sur la carte, elle affecte la façon dont les données sont présentées sur cette dernière et donc les besoins qu'elle satisfait. Néanmoins, le Tableau 2.3 illustre de manière globale la façon dont chacune des vues participent à la satisfaction des besoins identifiés.

Table 2.3 – *Schéma des vues participant à la satisfaction de chaque besoin*

	[B1.1]	[B1.2]	[B1.3]	[B2]	[B3]	[B4]	[B5]	[B6]
Filtres	●	●	●	●	●			
Vue globale						●		
Graphe			●	●			●	
Carte		●		●	●			
Chronologie	●			●	●			
Recherche			●	●				
Vue détaillée						●		
Graphe		●	●	●			●	
Carte	●	●	●	●	●			
Chronologie	●	●	●	●				
Information	●	●	●	●				●

Ainsi, on peut y voir que la vue *Filtres* permet de filtrer les évènements **[B2]**. Suite aux discussions avec les historiens, son utilisation a été généralisée pour pouvoir filtrer sur l'ensemble des attributs du tuple (p, i, l, d), satisfaisant ainsi **[B3]**. Le reste des besoins **[B1, 2]** est satisfait via l'utilisation de filtres.

Concernant les deux vues principales que sont la vue globale et la vue détaillée, c'est sans surprise que ces vues ont été conçues pour satisfaire **[B4]**, chacune d'elle permettant d'afficher de façon globale ou détaillée les données. Cependant, on peut voir, toujours sur le Tableau 2.3 que les besoins satisfaits par leurs sous-vues divergent.

Dans le cas de la vue globale, le graphe, la carte et la frise chronologique représentent respectivement les dimensions individu **[B1.3]**, espace **[B1.2]** et temps **[B1.1]** des données. La vue recherche vient en complément du graphe avec la capacité de parcourir **[B1.3]** et filtrer des individus **[B2]** du jeu de données.

La vue détaillée quant à elle est composée de sous-vues satisfaisant plus de besoins. En effet, en plus de fournir une vision plus détaillée, les représentations sont plus riches pour chacune des sous-vues. Ainsi, la carte, en plus de décrire la dimension spatiale **[B1.2]**, permet aussi de rendre compte de l'ordre chronologique **[B1.1]** des évènements. Dans la même logique, la frise chronologique nous renseigne sur les lieux **[B1.2]** où ils se sont produits ou les individus concernés **[B1.3]**. Le panneau

d'information vient compléter la vue détaillée en offrant l'ensemble des informations des individus sélectionnés.

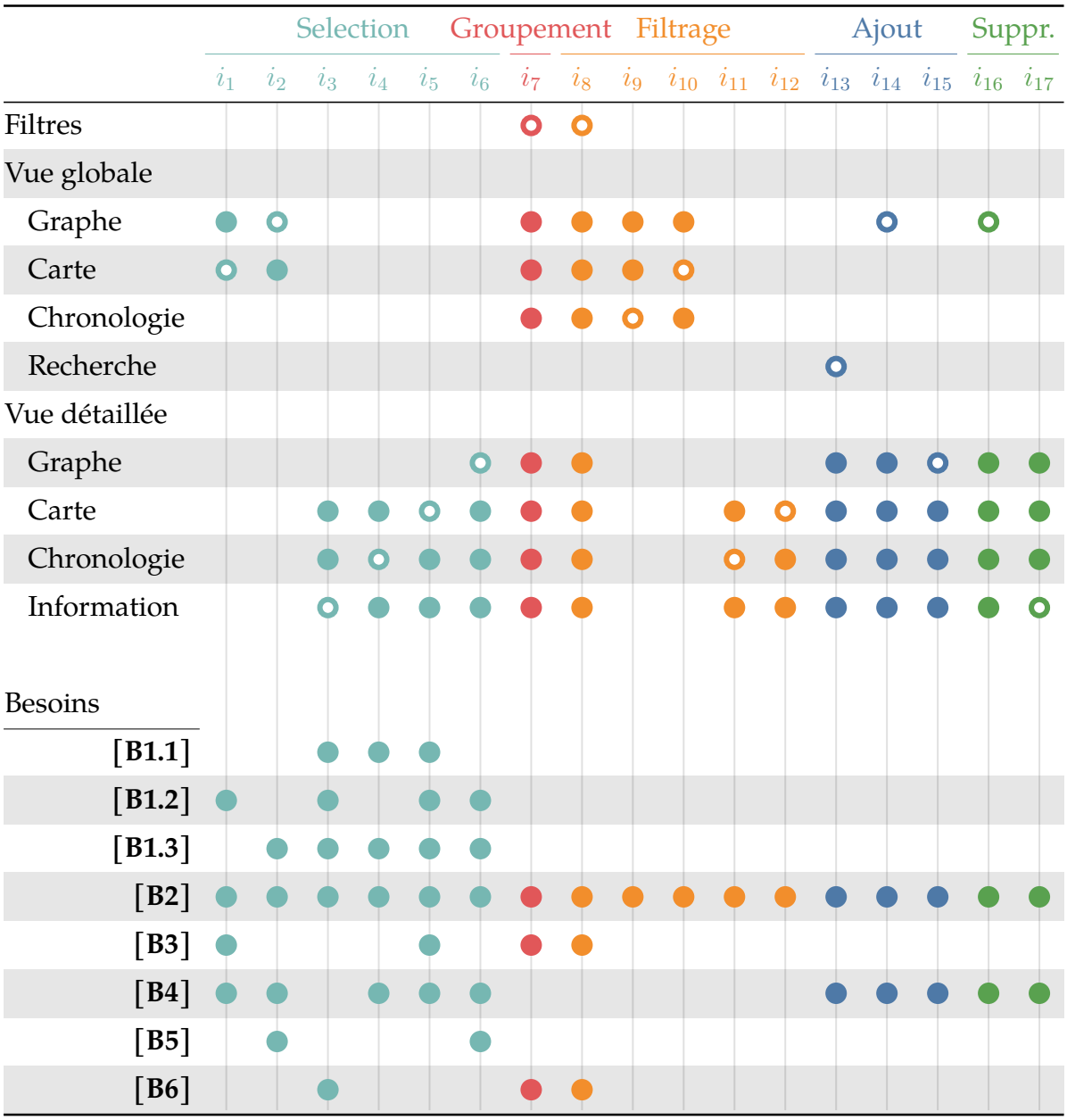
En complément des besoins satisfaits par chaque vue, la façon dont les vues sont coordonnées va aussi affecter les besoins qu'elles permettent de satisfaire. La coordinations des vues se fait via un système de cause à effet. Le fait d'interagir avec une vue propagera cette interaction sur les vues coordonnées à celle-ci. Ainsi, on définit cinq types d'interactions :

1. **Sélection** : met en valeur un ou des éléments (évènements ou individus) **[B2]**.
2. **Groupement** : modifie la façon dont les données sont regroupées **[B2, 3]**.
3. **Filtrage** : masque un ensemble d'éléments, *e.g.* des évènements. Plusieurs filtres successifs peuvent être appliqués **[B2, 3]**.
4. **Ajout** : ajout d'un individu à l'exploration détaillée **[B4]**.
5. **Suppression** : supprime un individu de l'exploration détaillée **[B4]**.

Le Tableau 2.4 illustre la coordination des vues. Les colonnes représentent les interactions possibles (notées i_n) et chaque interaction est indépendante. Les colonnes sont classées en fonction des 5 types présentés précédemment. Il met notamment en évidence la forte séparation entre la vue globale et la vue détaillée. La vue globale a peu d'actions de sélection (i_1, i_2) et repose surtout sur des actions de filtrage (i_{7-10}). La vue détaillée est densément coordonnée (i_{3-6}). La vue *filtres* applique seulement des actions de groupement et de filtrage (i_7, i_8), ceux-ci étant répercutés sur l'ensemble des vues de l'interface. Il y a différentes manières d'ajouter/supprimer des individus à la vue détaillée (i_{13-17}). Les interactions sur les graphes de relations (i_2, i_6) mettent en évidence les relations, permettant une navigation de proche en proche **[B5]**. La mise en évidence d'incohérence **[B6]** est rendu possible grâce à la vue information (i_3) et à la vue filtres (i_7, i_8), *e.g.* le fait de grouper sur la présence ou l'absence d'une géolocalisation aura pour effet de représenter la part des évènements qui n'ont pas de géolocalisation sur la ligne temporelle, satisfaisant ainsi **[B6]**.

La coordination permet notamment de synchroniser l'affichage afin de mettre en avant les éléments d'intérêts sur les différentes vues, pour satisfaire les besoins évoqués précédemment. Enfin, grâce à la partie filtres, elle permet d'altérer de manière contrôlée la représentation de chacune des vues, étendant leurs fonctionnalitées.

Table 2.4 – *Récapitulatif des actions possibles dans* ProsoVis *et leurs effets sur les différentes vues et les besoins qu'ils permettent de satisfaire. Chaque colonne représente une action, un cercle creux représente la vue qui initie l'action et un disque, les vues qui y réagissent. Les actions ont été classées sous plusieurs catégories : sélection, groupement, filtrage, ajout et suppression. Par exemple, une sélection sur la vue Information (i_3) a pour conséquence de mettre à jour la chronologie et la carte de la vue détaillée, cette interaction satisfait les besoins* **[B1]**, **[B2]** *et* **[B6]**.



2.5 Siprojuris

Dans cette partie nous nous attardons sur la façon dont les données de Siprojuris ont été adaptées afin d'être visualisées grâce à *ProsoVis* et nous illustrons via un exemple le fonctionnement de la plateforme.

2.5.1 Intégration des données

Nous avons vu dans la sous-section 2.3.2, que notre modèle de données est adapté pour pouvoir visualiser les données de Siprojuris. Toutefois, des traitements sont nécessaires pour transformer les données initiales. La Figure 2.30 décrit les étapes effectuées afin d'adapter les données de Siprojuris pour la plateforme, les fichiers nécessaires et comment ces fichiers sont exploités par celle-ci. La plateforme *ProsoVis* requiert 5 fichiers : (1) la liste des individus (acteurs) (2) des lieux (3) des évènements (4) des relations, et (5) le placement des nœuds pour le dessin de graphe.

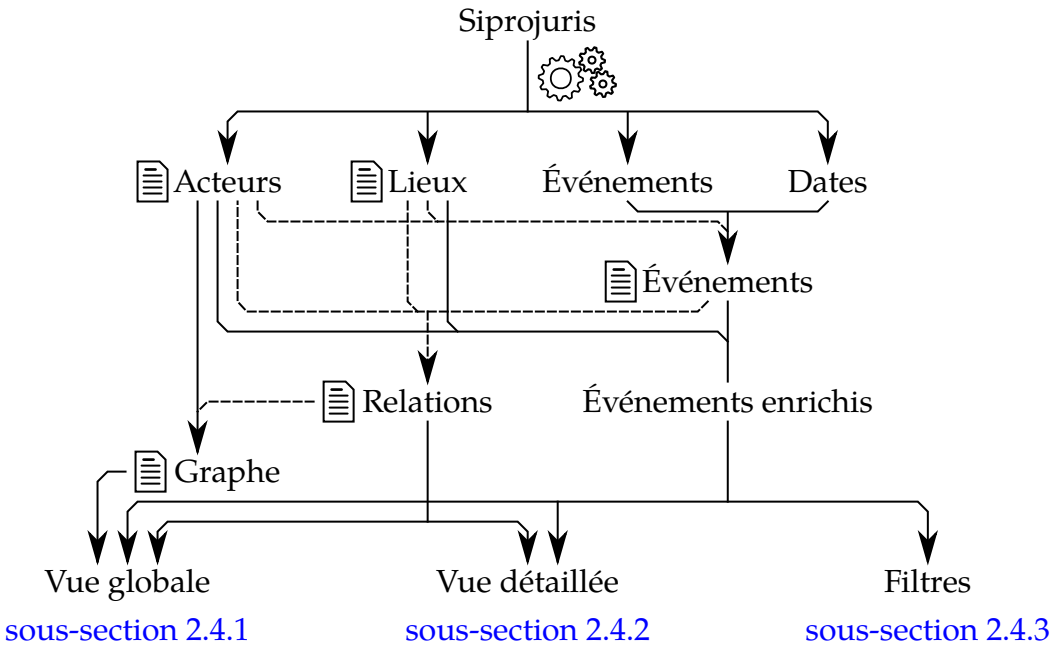


FIGURE 2.30 – *Flux des données et leur emploi. Les fichiers attendus par* ProsoVis *sont représentés avec l'icône de page. Les pointillés représentent les références des données.*

La première étape est l'extraction des données de Siprojuris de la base SYMOGIH, illustrée par les engrenages sur la Figure 2.30. Une fois les données des 4 types entités extraites, les dates sont considérées comme une description de l'évènement. Afin de réduire la redondance, les individus (acteurs) et les lieux sont stockés dans des fichiers indépendants, évitant ainsi de dupliquer l'ensemble de leurs attributs pour chauqe évènement les concernant. Le fichier des évènements contient les références vers les individus et les lieux associés aux évènements. Les *relations* sont ensuite inférées à partir des évènements, des individus et des lieux.

Le dessin de graphe (cf. le fichier *Graphe* sur la Figure 2.30) utilisé par la vue globale (cf. Section 2.4.1) est construit à partir de la liste des individus et de leurs relations.

Chaque individu correspond à un nœud du graphe représenté par un rectangle et contenant le nom de l'individu. Le placement de ces nœuds est déterminé via un algorithme de placement tel qu'un algorithme de placement par forces. L'algorithme ainsi que la métrique utilisée pour calculer la proximité entre deux individus peut être adapté au jeu de données. Nous utilisons *FM³* (Hachul & Jünger, 2004) pour le placement initial en utilisant le poids des relations comme métrique entre deux nœuds. *PRISM* (Gansner & Hu, 2010) est ensuite utilisé pour supprimer les chevauchements occasionnés par la variété de taille des labels. Les motivations et le choix de l'algorithme approprié de suppression des chevauchements est exploré dans le Chapitre 3.

Enfin, l'enrichissement des évènements est effectué par la plateforme, les 3 fichiers *Acteurs*, *Lieux* et *Evènements* sont lus et combinés afin de produire une structure où chaque évènement contient un attribut pour l'individu et le lieu. Les évènements enrichis, le graphe et les relations sont passés aux vues appropriées. Appliquant de manière automatique une série de règles, les évènements enrichis permettent la détection de potentielles incohérences, *e.g.* se produisant avant la naissance ou après le décès d'une personne, ou de données manquantes, *e.g.* le lieu, la localisation ou la date des évènements.

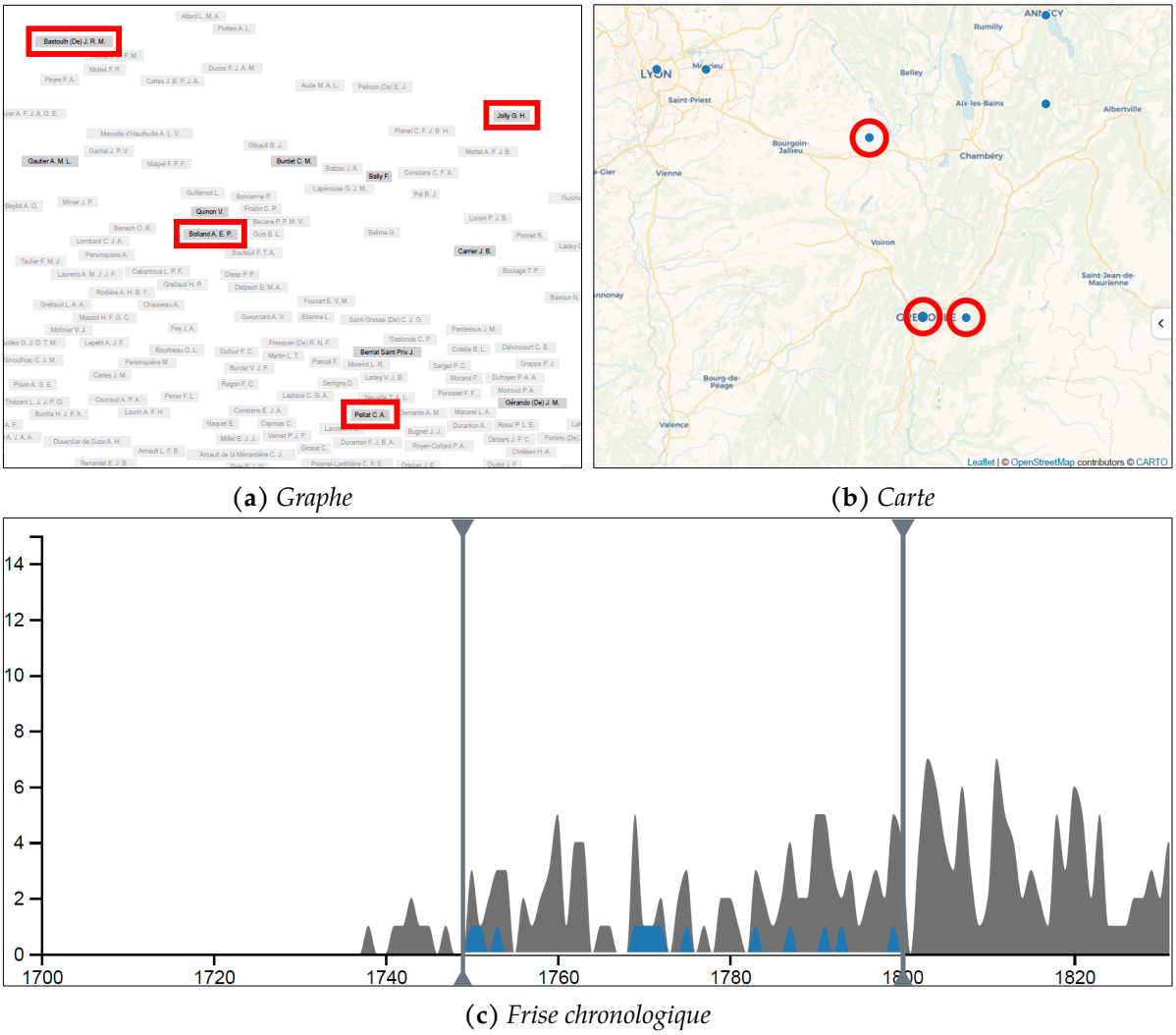
2.5.2 Exemple

Dans cette section nous illustrons le fonctionnement de la plateforme au travers d'un exemple autour de la question suivante :

> *Quelle est la carrière des Professeurs de droits nés dans la région de Grenoble pendant la seconde moitié du XVIII siècle (1750-1800) ?*

Pour répondre à cette question, il est nécessaire dans un premier temps de parcourir la carte de la vue globale (cf. sous-section 2.4.1) afin de ne retenir (via un filtre) que les données concernant la région de *Grenoble* (cf. Figure 2.31(b)). Ensuite, nous appliquons un filtre temporel pour ne retenir que la période d'intérêt, *i.e.* de 1750 à 1800 (cf. Figure 2.31(c)). Comme le montre la vue graphe (cf. Figure 2.31(a)), ces différents filtres permettent de mettre en surbrillance l'ensemble des individus correspondants à la région et à l'intervalle de temps qui nous intéresse. On peut alors sélectionner les individus (dans ce cas, *Pellat C. A.*, *Bolland A. E. P.*, *Jolly G. H.* et *Bastoulh (De) J. R. M.*) pour les visualiser dans la vue détaillée.

En passant à la vue détaillée (cf. Figure 2.32), il est intéressant de mettre en avant le parcours de chaque individu, ainsi nous changeons la coloration pour colorier en fonction des individus. En manipulant la carte pour la centrer sur les évènements concernés (cf. Figure 2.32(a)), nous pouvons voir que *Bastoulh (De) J. R. M.* et *Pellat C. A.* ont quitté *Grenoble* pour aller respectivement à *Noraget* et à *Paris*. Le graphe des relations (cf. Figure 2.32(b)) nous permet de voir que *Pellat C. A.* et *Bolland A. E. P.* se connaissaient certainement. En regardant de plus près ces deux individus, il semble que *Pellat C. A.* ait pu être un élève de *Bolland A. E. P.*, obtenant ainsi un *Baccalauréat* suivie d'une *Licence en Droit*, devenant un collègue de travail pendant un an environ avant de quitter *Grenoble* (cf. Figure 2.32(c)) et continuer en tant qu'enseignant à *Paris* jusqu'à sa mort en 1871.



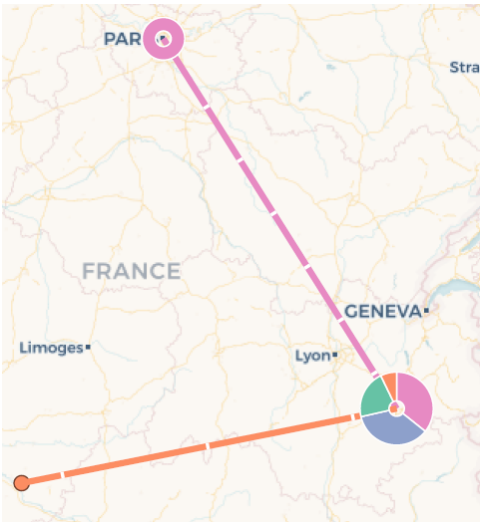
(**a**) *Graphe* (**b**) *Carte*

(**c**) *Frise chronologique*

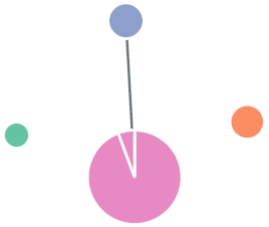
FIGURE 2.31 – *Vue globale affichant les naissances dans la région de Grenoble entre 1750 à 1800. Les éléments marqués en rouge correspondent aux individus sélectionnés pour l'exploration détaillée.*

En utilisant le graphe de la vue détaillée, nous pouvons retrouver des individus liés à ceux déjà présents. En explorant les relations de *Pellat C. A.* à la *Faculté de Grenoble* (cf. Figure 2.32(d)), on observe que la majorité des personnes qu'il a connu sont aussi des relations de *Bolland A. E. P.* et *Jolly G. H.*. Nous rajoutons une de ces personnes *Burdet C. M.* à la vue détaillée : le graphe mis à jour nous montre qu'il est une relation commune à *Bolland A. E. P.*, *Pellat C. A.* et *Jolly G. H.*. En continuant l'exploration de façon similaire, nous rajoutons *Quinon V.* qui vient remplacer *Bastoulh (De) J. R. M.* en raison de la limite fixée à 5 individus dans l'exploration détaillée.

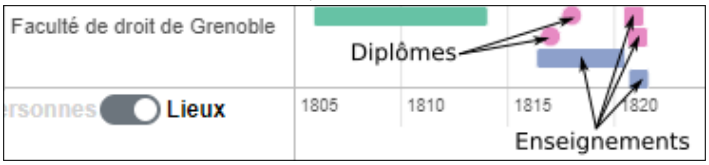
Dans le cas de *Bastoulh (De) J. R. M.*, à l'aide de la vue ressource (cf. Figure 2.32(e)), on note la faible quantité d'informations sur cet individu. De plus, une icône d'alerte signale l'absence de géolocalisation pour la *Faculté de droit de Toulouse* où *Bastoulh (De) J. R. M.* a enseigné le *droit civil*. Cette *localisation* n'as pas été liée au *lieu (Toulouse)*, qui existe pourtant dans le jeu de données. Cela donne ainsi l'impression, en ne regardant que la carte, que *Bastoulh (De) J. R. M.* est né à *Revel* avant de décéder à *Nogaret*, sans être passé par *Toulouse* entre les deux.



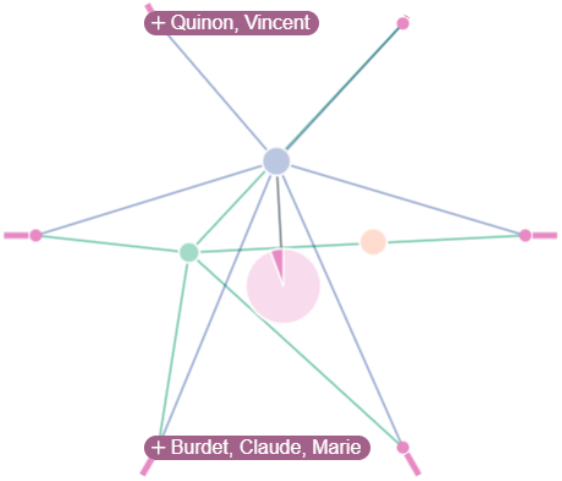
(**a**) *Carte*



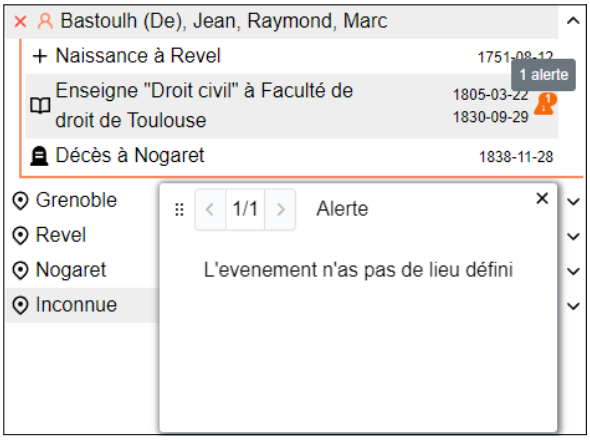
(**b**) *Graphe des relations*



(**c**) *Frise chronologique*



(**d**) *Cercle de suggestions de* Pellat C. A. *à la* Faculté de Grenoble

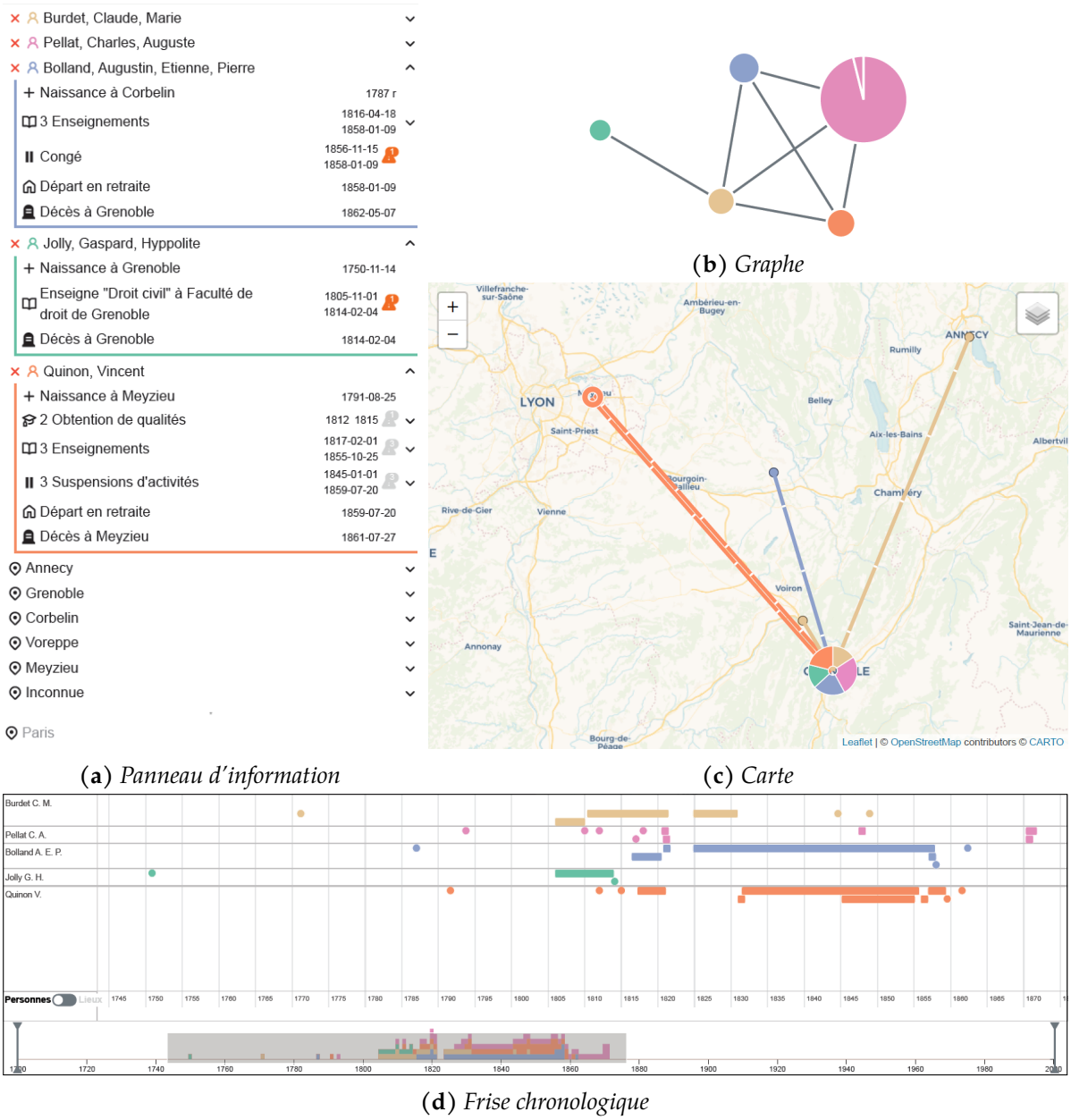


(**e**) *Évènements concernant* Bastoulh (De) J. R. M.

Figure 2.32 – *Vue détaillée décrivant le parcours de* Pellat C. A. (*en rose*), Bolland A. E. P. (*en bleu*), Jolly G. H. (*en vert*) *et* Bastoulh (De) J. R. M. (*en orange*)

Pour conclure (cf. Figure 2.33), à l'exception de *Pellat C. A.*, parti à *Paris* après avoir obtenu ses diplômes à *Grenoble*, *Burdet C. M.*, *Bolland A. E. P.* et *Jolly G. H.* sont nés et ont enseigné dans leur région natale. *Quinon V.* quant à lui, a enseigné à *Grenoble* avant de retourner dans sa ville natale, près de *Lyon*, probablement après son départ en retraite. Il y décède en 1861.

Á travers cet exemple, nous avons mis en avant dans la vue globale la réduction du champ de recherche dans l'ensemble des données, à l'aide des différents filtres. Dans la vue détaillée, nous avons mis en avant la visualisation de l'histoire des individus et l'élargissement de proche en proche de l'espace d'exploration. Cela a permis d'observer le comportement des différents individus nés dans cette période et de nous informer sur l'état des données, *i.e.* la quantité d'informations disponibles ou encore les données inconnues.



(**a**) *Panneau d'information* (**b**) *Graphe*

(**c**) *Carte*

(**d**) *Frise chronologique*

FIGURE 2.33 – *Vue détaillée décrivant le parcours de* Pellat C. A. *(en rose)*, Bolland A. E. P. *(en bleu)*, Jolly G. H. *(en vert)*, Burdet C. M. *(en jaune) et* Quinon V. *(en orange)*

2.6 Discussion

Dans cette partie, nous revenons tout d'abord sur les choix de conception que nous avons proposés. Par la suite, nous revenons sur les approches de visualisation prosopographiques existantes et leur adéquation par rapport aux besoins exprimés par les historiens. Enfin nous revenons sur le caractère générique de l'approche.

Nous avons vu que le processus de conception que nous avons retenu pour développer la plateforme était basée sur les réponses aux trois questions *Quoi ?*, *Pourquoi ?* et *Comment ?* proposée par MUNZNER (2014). En considérant un point de vue développement logiciel, un modèle d'organisation adapté pourrait être dans un premier temps

un *cycle en V* [12] où la partie (*Quoi ?* et *Pourquoi ?*) est faite a priori de l'implémentation (*Comment ?*). Cependant, il est reconnu que ce type d'organisation contient des risques importants liés au fait qu'au cours de la mise en œuvre, les spécifications initiales (*Quoi ?*) ou même les données initiales (*Pourquoi ?*) peuvent être incomplètes, fausses, ou irréalisables. Comme nous l'avons nous-mêmes constaté lors de nos travaux. Les réponses aux *Quoi ?* et *Pourquoi ?* appartiennent « au monde des historiens ». Ces travaux ont montré qu'il était vraiment indispensable d'avoir de nombreux échanges pour comprendre et abstraire leur problématique en besoins. Un autre point important mis en évidence lors de ces travaux est la nécessité d'arriver à définir une langue commune, le lexique utilisé en visualisation d'information etant très différent de celui propre aux historiens. Pour faciliter ces échanges, de nombreuses iterations de l'interface ont été proposées et ont servi de support à la discussion pour permettre aux historiens d'affiner leur problématique en une série de besoins concrets. En d'autres termes, le *Comment ?* permet d'aider, en mettant en évidence tel ou tel type de visualisation ou d'interaction, les experts dans l'expression du *Pourquoi ?*. Au final, notre processus de conception a principalement été basée sur des méthodes AGILE [13] favorisant des cycles courts avec retour d'expérience des utilisateurs.

La conception initiale du système s'inspire fortement du cadre (*framework*) pour les données spatio-temporelles proposé initialement par Peuquet (1994) (cf. Figure 2.34), étendue par Andrienko et al. (2003) et du cadre de travail de Bach et al. (2017). Via une représentation sous la forme d'un schéma triadique composé des questions (*what ?*, *when ?* et *where ?*) pour décrire respectivement les dimensions thématique, temporelle et spatiale des données, il permet de répondre à de nombreuses questions :

- *when* + *where* → *what* : décrire le(s) objet(s) (*what*) présents à un lieux ou un ensemble de lieux (*where*) à un ou un plusieurs temps (*when*).
- *when* + *what* → *where* : décrire le(s) lieu(x) (*where*) occupés par un ou des objets (*what*) à un ou plusieurs temps (*when*).
- *where* + *what* → *when* : décrire le(s) temp(s) (*when*) lors desquels un ou des objets (*what*) ont occupé un ou plusieurs lieux (*where*).

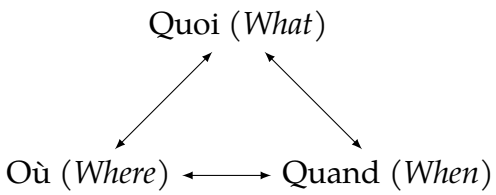


Figure 2.34 – *Schéma triadique de Peuquet (1994)*

Ces deux cadres offrent une partie des réponses pour la modélisation et la manipulation de données prosopographiques. Cependant, ils se révèlent rapidement insuffisants car la notion de relation entre individus ne fait pas intrinsèquement partie des données spatio-temporelles. Par exemple, pour savoir « Qui est le parent de A ? » (Bezerianos et

12. https://fr.wikipedia.org/wiki/Cycle_en_V
13. https://fr.wikipedia.org/wiki/MÃľthode_agile

al., 2010), la question posée implique que la thématique soit autoréférente : *what* → *what*. De la même manière, pour savoir « Qui a rencontré Einstein en Allemagne en 1960 ? » (LATIF et al., 2021), la question peut être très étendue : *when* + *where* + *what* → *what*.

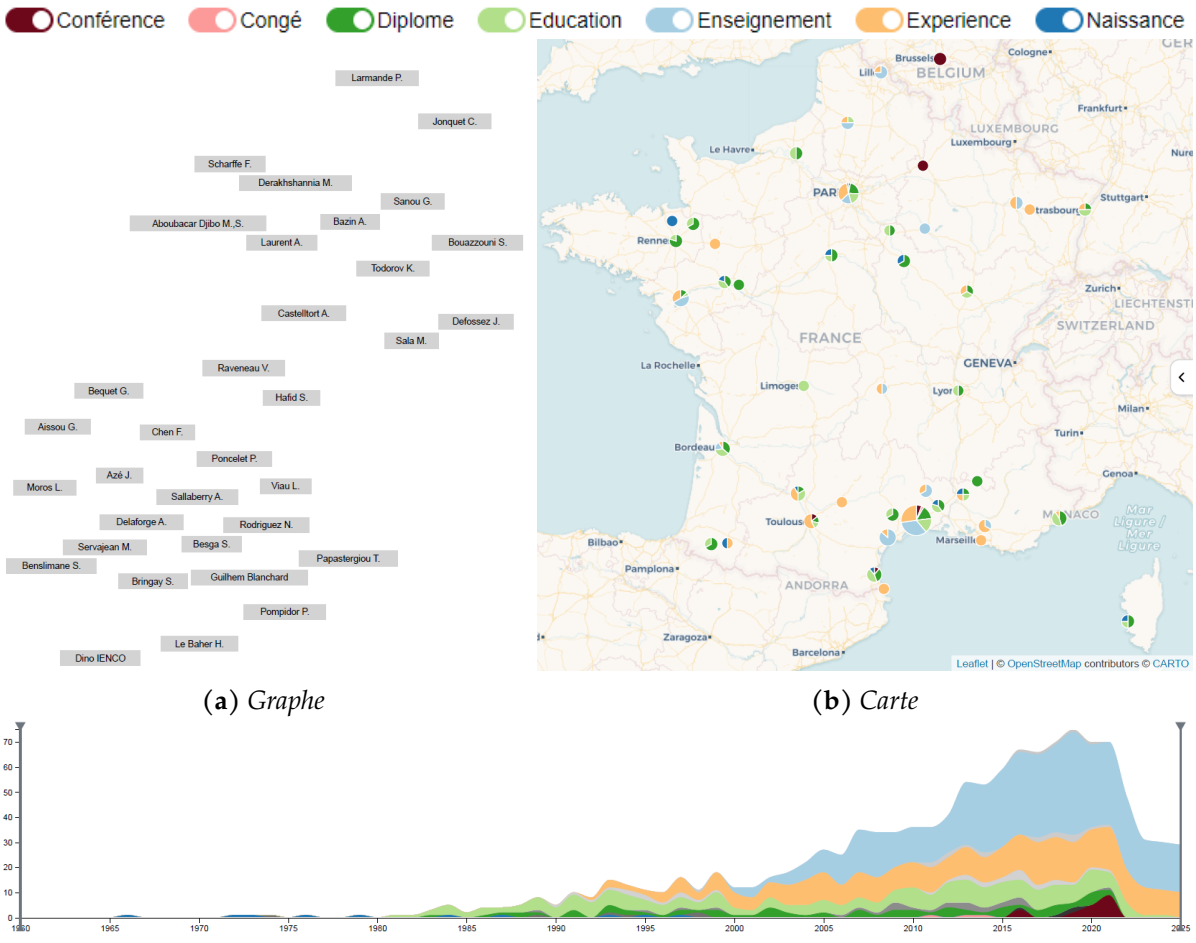
Étant donné les limitations d'une modélisation uniquement basée sur le schéma triadique, nous avons proposé **une modélisation des données centrée évènements**. Une approche alternative pourrait être centrée individu (HYVÖNEN et al., 2019). La raison de ce choix est que nous considérons un individu comme étant une entité décrite par des évènements. Ce qui est assez naturel pour des données prosopographiques où nous disposons des différents évènements intervenus dans la vie des individus. Bien entendu, cette modélisation peut avoir des limites. Par exemple, il n'est pas possible de considérer un individu uniquement par son nom sans aucun évènement, *i.e.* il n'est pas possible d'avoir dans le graphe des relations des individus sans évènements. Il est cependant facile de contourner cette limite en ajoutant simplement un évènement (*e.g.* naissance) ou bien sa participation via une relation avec un autre individu de la base.

Nous avons vu que les données prosopographiques pouvaient être incertaines, partielles, etc. Les échanges avec les experts ont montré qu'il était important de mettre en évidence ces erreurs **[B6]**. Nous avons proposé des solutions afin de faciliter leur représentation. Il est par exemple possible de représenter des données même si une bonne partie des informations sont incomplètes (*e.g.* un individu peut apparaître dans la plateforme même si sa date de naissance ou son lieu de naissance sont approximatifs). Il peut cependant exister des données pour lesquelles nous n'avons aucune connaissance, *i.e.* des données manquantes. Par exemple, il peut exister un grand laps de temps entre deux évènements, au cours duquel nous n'avons pas connaissance d'éventuels évènements intermédiaires ayant pu se produire. Bien entendu, contrairement aux données incertaines, notre plateforme ne peut pas dans ce cas induire un « a priori » sur le parcours d'un individu car il n'y a aucun moyen de mettre en évidence le fait que des évènements seraient potentiellement absents.

Nous revenons, à présent, sur les approches de visualisation proposées pour des données prosopographiques. Nous avons vu dans l'état de l'art, qu'à notre connaissance, il n'existe que peu d'approches. Nous avons mis en évidence que *BibliographySampo* (HYVÖNEN et al., 2019) offre de nombreuses fonctionnalités pour répondre aux besoins des historiens. Cependant la plateforme est pensée comme un portail vers les données plutôt que comme une plateforme de visualisation analytique. Elle offre des fonctionnalités associées aux besoins exprimés par les experts. Par exemple, il est possible de naviguer dans l'ensemble des données **[B1]**. Cependant il est difficile de filtrer sur les différentes dimensions **[B2]**. En particulier, il est difficile de restreindre les données à une zone géographique, de trouver les relations communes entre deux individus à un lieu donné **[B4]** ou encore de comparer le parcours de plusieurs individus. Malgré ces limites, il serait intéressant d'étudier les fonctionnalités complémentaires proposées dans *BibliographySampo* afin de voir comment ces dernières pourraient être intégrées dans notre plateforme. Pour cela, comme nous l'avons indiqué précédemment, des échanges avec les historiens sont nécessaires pour étendre les besoins (*Pourquoi ?*) et réfléchir ensemble au *Comment ?*

Pour conclure cette discussion, nous revenons sur l'aspect générique de notre mo-

dèle et de sa mise en œuvre. De manière à illustrer comment le modèle peut être utilisé pour d'autres bases prosopographiques, nous avons mis en place une plateforme en considérant comme données les membres des équipes WEB3 [14] et ADVANSE [15] du LIRMM (cf. Figure 2.35). Les données ont été récupérées sur la base du volontariat et concernent les principaux éléments d'un curriculum vitæ de doctorant ou d'enseignant chercheur : date et lieu de naissance, études, diplômes, expériences professionnelles, enseignements réalisés, participations à des conférences, encadrements, etc. Bien entendu cette plateforme pourrait être améliorée et offrir un choix plus vaste de type d'évènements. Toutefois son objectif est principalement de montrer que même si *ProsoVis* est particulièrement bien adaptée aux données de Siprojuris, elle permet de facilement considérer d'autres bases de données.



(**a**) *Graphe* (**b**) *Carte*

(**c**) *Frise chronologique*

Figure 2.35 – *Vue globale des évènements des membres des équipes WEB3 et ADVANSE du LIRMM, en France, de 1960 à 2022*

Au total, sur les 34 membres des deux équipes, 17 ont participé, totalisant près de 400 évènements, 90 lieux et 200 localisations. Pour construire le graphe de la vue globale (cf. Figure 2.35(a)) des relations comme l'équipe d'appartenance, l'encadrement dans

14. http://www.lirmm.fr/recherche/equipes/fado
15. http://www.lirmm.fr/recherche/equipes/advanse

le cas des doctorants, stagiaires ou post-doc, ou du co-encadrement dans le cas des chercheurs ont été utilisés. L'ensemble a été placé en utilisant un algorithme basé sur la force et les chevauchements ont été supprimés en utilisant FTA (HUANG et al., 2007). On distingue sur le graphe (cf. Figure 2.35(a)) deux groupes distincts correspondants aux équipes WEB3 (en haut à droite) et ADVANSE (en bas à gauche).

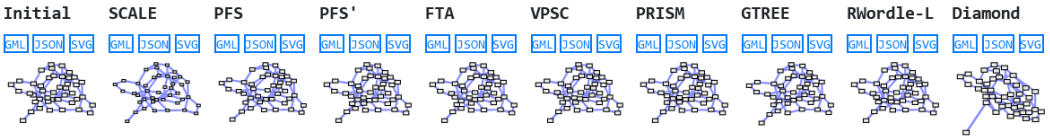
2.7 Conclusion

Dans ce chapitre, nous avons proposé une plateforme de visualisation, *ProsoVis*, pour les données prosopographiques. Ces dernières nécessitent de pouvoir visualiser des individus avec des évènements qui interviennent dans différents lieux au cours du temps. Nous avons étudié les différentes approches spécifiques à une dimension (*e.g*. individus, évènements), les approches hybrides et les travaux proposés spécifiquement pour les données prosopographiques. Nous avons constaté que par rapport aux différentes tâches mais aussi aux spécificités des données ces dernières n'étaient pas forcément adaptées. Nous avons donc proposé *ProsoVis* une plateforme qui offre une vision à la fois générale et très détaillée des données. Á travers différentes vues et de nombreuses interactions, la plateforme permet de naviguer dans les données afin d'aider l'expert à répondre à ses questions ou à vérifier ses hypothèses. Bien entendu, les visualisations proposées dans *ProsoVis* possèdent des caractéristiques qu'il faut prendre en compte. Par exemple, pour la représentation du graphe de la vue globale une visualisation sans chevauchements est indispensable. Même s'il est possible de précalculer un tel placement, le garantir lors d'interactions telles qu'un zoom nécessite un algorithme adapté pour conserver la fluidité nécessaire au processus d'exploration. Il a fallu choisir un algorithme efficace. De le même manière, en voulant reporter les informations utiles sur la carte, nous avons constaté que celles-ci étaient très nombreuses et pouvaient pénaliser les interactions et la lisibilité.

Pour répondre à ces problématiques, nous avons dû, dans le premier cas, trouver, en fonction de nos critères spécifiques associés à la visualisation (*e.g*. pas de perte de la carte mentale, pas trop de déformation, etc.), quel était l'algorithme de suppression de chevauchements le plus approprié. Dans le second cas, il était indispensable, étant donné le nombre d'informations affichées, d'offrir une approche de regroupement spatial qui offre beaucoup de fluidité dans la navigation. Dans les prochains chapitres nous présentons nos réponses à ces deux problématiques.

AGORA

rowe.gml overlaps : 9



Criterias	SCALE	PFS	PFS'	FTA	VPSC	PRISM	GTREE	RWordle-L	Diamond
Normalised Number of Inversions	0	0.5271	0.5271	0.0055	0.0039	0.0039	0.0138	0.5343	0.0936
Convex Hull Area	2.4433	1.2004	1.0003	1.0003	1.0001	1.009	1.1442	0.9925	1.015
Improved Aspect Ratio	1	1.0215	1	1	1	1.0013	1.0377	1	1.2005
Improved Mean Squared Euclidean	0	504.4245	97.2853	51.3027	12.0827	32.0801	382.2989	83.949	5151.6549
Relative Standard Deviation	0	0.1243	0.1061	0.0757	0.0497	0.102	0.1182	0.0791	0.4218

Sommaire

Dans le chapitre précédent, nous avons vu que l'algorithme *PRISM* était utilisé pour visualiser les informations sous la forme de graphes. Dans ce chapitre, nous revenons sur les raisons qui nous ont poussées à retenir cette solution. Comme nous le verrons, même s'il existe des approches très efficaces pour dessiner des graphes, les interactions de l'utilisateur peuvent générer des problèmes de chevauchements. Pour pallier ces problèmes, des algorithmes comme *PRISM* ont été proposés dans la littérature. Cependant savoir quel algorithme est le plus adapté aux besoins d'une visualisation reste une tâche difficile. En effet, il existe de nombreux critères et pour chacun d'entre eux de nombreuses métriques associées, et il n'existe que peu de comparaison des algorithmes sur les mêmes critères et les mêmes métriques. En outre, même si les temps d'exécution font partie des critères que l'on peut considérer, ces derniers restent difficilement comparables car les implémentations sont réalisées dans des langages différents. C'est dans ce contexte que se situent les travaux présentés dans ce chapitre. Nous proposons une étude des principaux critères et des mesures associées afin de pouvoir comparer les algorithmes de suppression de chevauchement. Cette étude offre ainsi à l'utilisateur une solution pour choisir, en fonction des critères qu'il souhaite privilégier, l'algorithme le plus approprié.

3.1 Contexte

Les algorithmes de dessin de graphes (Gibson et al., 2013 ; Kwon & Ma, 2020 ; Tamassia, 2014) ont montré leur efficacité pour représenter des graphes riches et expressifs. Cependant ils considèrent généralement que les nœuds se résument à un point dans l'espace et ne possèdent pas de dimension. Dans le cas d'applications où les nœuds possèdent des informations à afficher (*e.g.* le nom d'une personne dans un réseau social) cela peut engendrer des problèmes de chevauchement entre les différents nœuds et masquer des informations. Pour résoudre ce problème, des algorithmes de post-traitement, appelés *ajustement de la disposition* (*layout adjustment*, Misue et al., 1995), ont été proposés.

L'objectif de ces algorithmes est, compte tenu d'un positionnement initial des nœuds et d'une taille pour chacun d'entre eux, de fournir un nouveau plongement de sorte qu'il n'existe plus de chevauchements entre les nœuds. Le plus trivial de ces algorithme consiste à faire un agrandissement uniforme de l'échelle utilisée tout en conservant la taille des nœuds jusqu'à qu'il n'y ait plus de chevauchements. Cependant, cette technique crée de grandes zones sans objets et étend la surface de la visualisation de façon significative. Par conséquent, il est utile qu'un algorithme de suppression des chevauchements des nœuds essaye de minimiser cette surface. A l'opposé, positionner les nœuds de manière égale sur une grille répond aussi à la problématique mais entraîne la perte de la *carte mentale* de l'utilisateur [1] issue du plongement initial. Pour éviter cet inconvénient, il est important de minimiser les changements de configuration (*e.g.* si un nœud u est à gauche d'un nœud v dans le plongement initial, on veut qu'il soit toujours à gauche dans le plongement issu de l'algorithme de suppression de chevauchements).

1. Une discussion sur le concept de préservation de la carte mentale est proposée dans Archambault et Purchase, 2013.

Depuis un travail préliminaire de Misue et al. (1995), de nombreux algorithmes ont été conçus pour atteindre ces objectifs et de nombreux critères de qualité ont été proposés pour les évaluer. Malheureusement, à notre connaissance, une comparaison complète des algorithmes basée sur les différents critères n'a jamais été proposée et il est donc difficile pour un concepteur de visualisation de choisir l'algorithme qui répond le mieux à ses besoins.

Notre contribution se présente sous trois formes :

1. Nous proposons une classification des 22 métriques utilisées pour évaluer la qualité des algorithmes en les regroupant en fonction de l'objectif qu'elles tentent de capturer. Nous discutons également de leur pertinence et proposons une métrique représentative pour chaque classe de critères.
2. Nous comparons expérimentalement l'état de l'art des approches de suppression des chevauchements des nœuds en fonction des métriques sélectionnées précédemment. Les expériences portent sur 854 graphes synthétiques (aléatoires, arbres, à invariant d'échelle, petits-mondes) et réels.
3. Nous proposons une librairie *JavaScript* [2] contenant tous les algorithmes décrits dans ce document ainsi qu'une plateforme Web, *AGORA* [3] (*Automatic Graph Overlap Removal Algorithms*) dans laquelle il est possible d'importer un ensemble de graphes, d'appliquer les algorithmes de suppression de chevauchements utilisés dans cette étude et de télécharger les résultats avec les valeurs des métriques des critères de qualité associés.

Le chapitre est organisé de la manière suivante : après une introduction des définitions et des notations utilisées dans la Section 3.2, nous présentons et discutons les critères de qualité et les métriques associées dans la Section 3.3. Nous décrivons et analysons ensuite, Section 3.4, les expérimentations menées pour comparer les algorithmes sur des jeux de données synthétiques et réelles. Nous présentons la plate-forme Web *AGORA* dans la Section 3.5. Nous proposons une discussion, Section 3.6, avant de conclure le chapitre.

3.2 Définitions et notations préliminaires

Dans ce chapitre, nous utilisons les définitions et notations suivantes.

$G = (V, E)$ désigne un graphe où V est un ensemble de nœuds et E un ensemble d'arêtes. Le nombre de nœuds $|V|$ est désigné par n et le nombre d'arêtes $|E|$ par m. Contrairement aux algorithmes de dessin de graphes évoqués précédemment, nous considérons chaque nœud comme un rectangle. Ainsi, pour un nœud $v \in V$, sa largeur et sa hauteur sont désignées par le couple (w_v, h_v) qui n'est pas modifié par l'ajustement du plongement.

Le plongement initial est défini comme une injection $\mathcal{E}_G$: $V \to \mathbb{R}^2$ telle que $\forall v \in V$, $\mathcal{E}_G(v) = (x_v, y_v)$ où (x_v, y_v) sont les coordonnées du centre du nœud v. Le plongement

2. https://github.com/agorajs/agorajs.github.io
3. https://agorajs.github.io/

sans chevauchements est désigné par $\mathcal{E}'_G$. Pour simplifier les notations, nous désignons $v = (x_v, y_v)$ au lieu de $\mathcal{E}_G(v)$, et $v' = (x'_v, y'_v)$ au lieu de $\mathcal{E}'_G(v)$. À noter que deux nœuds $(u, v) \in V^2$ se chevauchent lorsque :

$$|x_v - x_u| < \frac{w_v + w_u}{2} \quad \text{et} \quad |y_v - y_u| < \frac{h_v + h_u}{2}$$

La boîte englobante bb d'un plongement $\mathcal{E}_G$ est définie comme le plus petit rectangle contenant tous les nœuds de G, w_{bb} (*resp.* h_{bb}) désigne la largeur (*resp.* la hauteur) du plongement initial et w'_{bb} (*resp.* h'_{bb}) désigne la largeur (*resp.* la hauteur) de celui sans chevauchements. Ils sont définis comme suit :

$$w_{bb} = \left| \max_{v \in V} \left(x_v + \frac{w_v}{2} \right) - \min_{u \in V} \left(x_u - \frac{w_u}{2} \right) \right|$$
$$h_{bb} = \left| \max_{v \in V} \left(y_v + \frac{h_v}{2} \right) - \min_{u \in V} \left(y_u - \frac{h_u}{2} \right) \right|$$

La position du centre de la boîte englobante est désignée par $c_{bb} = (x_{bb}, y_{bb})$ dans le plongement initial et $c'_{bb} = (x'_{bb}, y'_{bb})$ dans le plongement sans chevauchements.

L'enveloppe convexe d'un plongement $\mathcal{E}_G$ est définie comme la plus petite région convexe contenant tous les nœuds de G. Celle-ci peut être calculée en utilisant les 4 coins des nœuds et non pas uniquement leur centre de manière à ce que les rectangles représentant les nœuds y soient entièrement inclus. Dans ce qui suit, ch désigne l'enveloppe convexe du plongement original, ch' l'enveloppe convexe du plongement sans chevauchements, c_{ch} le centre de masse de ch et c'_{ch} le centre de masse de ch'.

3.3 Critères de qualité

De nombreux critères ont été proposés dans la littérature pour évaluer la qualité des plongements résultant des algorithmes d'ajustement. Malheureusement, les évaluations fournies par les auteurs des différentes approches ne sont pas toujours basées sur les mêmes critères ou, s'ils considèrent les mêmes critères, les métriques utilisées ne sont pas les mêmes et les comparaisons restent difficiles. Cette limite se retrouve également sur les temps d'exécution dans la mesure où les implémentations sont souvent réalisés dans des langages très différents (*e.g.* C++, Java, C#, etc.)

Afin de fournir un protocole d'expérimentation uniforme et une comparaison complète des algorithmes, nous devons passer en revue les critères de qualité et les métriques associées afin de les évaluer. De plus, notre objectif est de déterminer si une métrique peut être représentative pour chaque critère.

Table 3.1 – *Métriques classées par critère de qualité : les métriques sélectionnées pour notre étude apparaissent en italique gras. Les abréviations sont basées sur certaines initiales des noms, par exemple sp_bb_a signifie que la métrique est dans la classe* Minimisation de l'expansion, *elle utilise la* boîte englobante *pour quantifier l'expansion de l'aire. La colonne* ***Intervalle*** *contient l'ensemble des valeurs que peut prendre la métrique. La colonne* ***O*** *fait référence à la valeur objectif pour satisfaire le critère correspondant.*

Abrév.	Nom	Intervalle	O
	Préservation de l'ordre orthogonal		
oo_o	Original (Misue et al., 1995)	$\{0, 1\}$	1
oo_kt	Kendall's Tau Distance (Huang et al., 2007)	$[0, 1]$	0
oo_ni	Number of Inversions (Strobelt et al., 2012)	$[0, n(n-1)]$	0
oo_nni	***Normalised Number of Inversions***	$[0, 1]$	0
	Minimisation de l'expansion		
sp_bb_l1ml	Bounding Box L1 Metric Length (Li et al., 2005)	$[1, +\infty[$	1
sp_bb_a	Bounding Box Area (Misue et al., 1995)	$[1, +\infty[$	1
sp_bb_na	Bounding Box Normalised Area (Huang et al., 2007)	$[0, 1[$	0
sp_ch_a	***Convex Hull Area*** (Strobelt et al., 2012)	$[1, +\infty[$	1
	Préservation du rapport de forme		
gs_bb_ar	Bounding Box Aspect Ratio (Li et al., 2005)	$]0, +\infty[$	1
gs_bb_iar	***Bounding Box Improved Aspect Ratio***	$[1, +\infty[$	1
gs_ch_sd	Convex Hull Standard Deviation (Strobelt et al., 2012)	$[0, +\infty[$	0
	Minimisation du mouvement des nœuds		
nm_mn	Moved Nodes (Huang et al., 2007)	$[0, 1]$	0
nm_dm_me	Distance Moved Mean Euclidean (Strobelt et al., 2012)	$[0, +\infty[$	0
nm_dm_ne	Distance Moved Normalised Euclidean (Lyons et al., 1998)	$[0, 1]$	0
nm_dm_h	Distance Moved Hamiltonian (Huang & Lai, 2003 ; Huang et al., 2007)	$[0, +\infty[$	0
nm_dm_se	Distance Moved Squared Euclidean (Marriott et al., 2003)	$[0, +\infty[$	0
nm_dm_imse	***Distance Moved Improved Mean Squared Euclidean***	$[0, +\infty]$	0
nm_d	Displacement (Gansner & Hu, 2010)	$]0, +\infty[$	0
nm_knn	K-Nearest Neighbours (Nachmanson et al., 2016)	$[0, +\infty[$	0

Table 3.1 – (*suite*)

Abrév.	Nom	Intervalle	O
	Préservation des longueurs des arêtes		
el_r	Ratio (Li et al., 2005)	$[1, +\infty[$	1
el_rsdd	Relative Standard Deviation Delaunay (Gansner & Hu, 2010)	$[0, +\infty]$	0
el_rsd	***Relative Standard Deviation***	$[0, +\infty]$	0

Après analyse de l'état de l'art, nous avons identifié 5 classes de critères : *Préservation de l'ordre orthogonal* (noté oo pour *Orthogonal Ordering*), *Minimisation de l'expansion* (noté sp pour *SPread*), *Préservation du rapport de forme* (noté gs pour *Global Shape*), *Minimisation du mouvement des nœuds* (noté nm pour *Node Movement*) et *Préservation des longueurs des arêtes* (noté el pour *Edge Length*).

Chacune d'entre elles représente un critère de qualité. Le Tableau 3.1 montre les métriques associées aux classes de critères ainsi que leur intervalle de valeurs possibles. Les formules sont présentées et explicitées par la suite. Les abréviations des classes sont utilisées comme préfixe pour les métriques.

Les sous-sections suivantes décrivent les métriques associées à chaque classe spécifique. Pour chacune d'entre elles, nous sélectionnons une métrique représentative sur la base du critère de qualité correspondant et des propriétés que les métriques visent à capturer. Notre discussion fait aussi parfois intervenir le coefficient de corrélation de deux métriques obtenues lors des expérimentations menées (cf. Section 3.4).

3.3.1 Préservation de l'ordre orthogonal

La classe de la préservation de l'ordre orthogonal regroupe les métriques qui tentent de quantifier dans quelle mesure un algorithme d'ajustement préserve l'ordre orthogonal initial, *i.e.* les conditions suivantes :

$$\begin{cases} x_u < x_v \Leftrightarrow x'_u < x'_v \\ y_u < y_v \Leftrightarrow y'_u < y'_v \\ x_u = x_v \Leftrightarrow x'_u = x'_v \\ y_u = y_v \Leftrightarrow y'_u = y'_v \end{cases}$$

La première métrique de cette classe présentée dans Misue et al. (1995), ici appelée oo_o, est égale à 1 si le plongement du graphe sans chevauchements préserve l'ordre orthogonal initial et à 0 sinon. Ainsi, si un seul couple de sommets ne satisfait pas les conditions énumérées ci-dessus, la valeur de oo_o est la même que si un grand nombre de couples ne les satisfont pas.

Pour surmonter ce problème, Huang et al. (2007) ont proposé une métrique basée sur la *distance Tau de Kendall* (oo_kt). Pour chaque couple de nœuds, ils calculent

d'abord un nombre d'inversions $inv(u, v)$ correspondant à 0 si l'ordre orthogonal est préservé et à 1 sinon. La métrique est alors définie comme la somme normalisée du nombres d'inversions :

$$\text{oo\_kt} = \frac{\sum_{u \neq v} inv(u, v)}{n(n-1)}$$

Strobelt et al. (2012) ont proposé le nombre d'inversions :

$$\text{oo\_ni} = \sum_{\substack{(u,v) \in V^2 \\ x_u > x_v}} \begin{cases} 1 & \text{if } x'_u < x'_v \\ 0 & \text{otherwise} \end{cases} + \sum_{\substack{(u,v) \in V^2 \\ y_u > y_v}} \begin{cases} 1 & \text{if } y'_u < y'_v \\ 0 & \text{otherwise} \end{cases}$$

Cette métrique présente l'inconvénient de fournir des valeurs non normalisées. Cependant, elle présente l'avantage de pénaliser les inversions se produisant sur chaque axe indépendamment (*i.e.* l'axe x et y) au lieu de pénaliser, de la même manière, une inversion se produisant sur un seul axe et une inversion se produisant sur les deux axes. Ainsi, dans notre étude, nous combinons les deux métriques en utilisant une version normalisée de la seconde :

$$\text{oo\_nni} = \frac{oo\_ni}{n(n-1)}$$

3.3.2 Minimisation de l'expansion

Une fonction de zoom classique conservant la taille des nœuds, *i.e.* une mise à l'échelle uniforme, fournit un plongement sans chevauchements mais étend la visualisation et donne ainsi lieu à de grandes zones sans aucun objet. Pour pallier ce problème, des métriques de qualité ont été introduites pour quantifier l'expansion du plongement. Leur but est de favoriser les algorithmes qui produisent un faible étalement.

La longueur métrique L1 de Li et al. (2005) est le rapport :

$$\text{sp\_bb\_l1ml} = \frac{\max(w'_{bb}, h'_{bb})}{\max(w_{bb}, h_{bb})}$$

L'inconvénient de cette métrique est de ne considérer qu'une seule dimension du plongement, *i.e.* la largeur ou la hauteur. Par exemple, si l'on considère $w_{bb} = 4$, $h_{bb} = 2$, $w'_{bb} = 4$ et $h'_{bb} = 4$, la valeur de la longueur métrique L1 est de 1 (qui est la valeur cible), tandis que la surface du plongement sans chevauchements est deux fois plus grande que dans le plongement initial. Le rapport entre les surfaces des boîtes englobantes des deux plongements proposé par Misue et al. (1995) permet de surmonter ce problème :

$$\text{sp\_bb\_a} = \frac{w'_{bb} \times h'_{bb}}{w_{bb} \times h_{bb}}$$

Bien que le résultat donne une valeur non bornée supérieure à 1, Huang et al. (2007) proposent une version normalisée qui permet d'avoir des valeurs dans l'intervalle $[0, 1[$:

$$\text{sp\_bb\_na} = 1 - \frac{w_{bb} \times h_{bb}}{w'_{bb} \times h'_{bb}}$$

Même si les valeurs sont bornées entre $[0, 1[$, ce critère reste malheureusement peu intuitif car il est difficile de comprendre ce que les valeurs représentent.

Dans notre comparaison, nous avons choisi une autre version du rapport des aires impliquant des enveloppes convexes (Strobelt et al., 2012) dans la mesure où il permet de mieux prendre en compte la surface concrète du plongement :

$$\text{sp\_ch\_a} = \frac{area(ch')}{area(ch)}$$

3.3.3 Préservation du rapport de forme

Cette classe contient des métriques qui tentent de capturer la capacité des algorithmes à préserver la forme globale du plongement initial. La première a été proposée par Li et al. (2005) :

$$\text{gs\_bb\_ar} = \begin{cases} \dfrac{w'_{bb} \times h_{bb}}{h'_{bb} \times w_{bb}} & \text{si } w'_{bb} > h'_{bb} \\ \dfrac{h'_{bb} \times w_{bb}}{w'_{bb} \times h_{bb}} & \text{sinon} \end{cases}$$

L'idée sous-jacente est de capturer la variation du rapport d'aspect (w_{bb}/h_{bb}) entre le plongement initial et le plongement sans chevauchements. Par exemple, considérons $w_{bb} = 3$, $h_{bb} = 2$, $w'_{bb} = 6$ et $h'_{bb} = 4$ (cf. Figure 3.1, rectangles gris et verts). Dans ce cas, le plongement sans chevauchements est deux fois plus grand que le plongement initial mais le rapport de forme reste le même $3/2$. gs_bb_ar vaut 1, qui est la valeur cible. Considérons maintenant un autre exemple où $w_{bb} = 3$, $h_{bb} = 2$, $w'_{bb} = 4$ et $h'_{bb} = 6$ (cf. Figure 3.1, rectangle bleu). Dans ce cas, le rapport de forme initial est de $3/2$ alors que celui sans chevauchements est de $2/3$. gs_bb_ar vaut maintenant 2.25 qui n'est pas la valeur cible ; cela révèle une déformation du plongement initial pendant le processus de suppression des chevauchements.

Le principal inconvénient de cette métrique est qu'elle peut atteindre des valeurs dans l'intervalle $]0, +\infty[$ alors que la valeur cible est 1. Par exemple, $w_{bb} = 3$, $h_{bb} = 2$, $w'_{bb} = 6$ et $h'_{bb} = 5$ induit un gs_bb_ar égal à 0.8 (cf. Figure 3.1, rectangle violet), alors que $w_{bb} = 3$, $h_{bb} = 2$, $w'_{bb} = 4$ et $h'_{bb} = 2$ induit un gs_bb_ar égal à 1.33 (cf. Figure 3.1, rectangle jaune). Dans ce cas, il est difficile de décider quel algorithme est le meilleur entre les deux si le premier obtient la boîte englobante violette et le second la boîte jaune car nous n'avons aucun indice pour comparer $0,8$ et $1,33$ lorsque 1 est la valeur cible. Pour surmonter ce problème, nous proposons de l'affiner comme suit :

$$\text{gs\_bb\_iar} = \max\left(\frac{w'_{bb} \times h_{bb}}{h'_{bb} \times w_{bb}}, \frac{h'_{bb} \times w_{bb}}{w'_{bb} \times h_{bb}}\right)$$

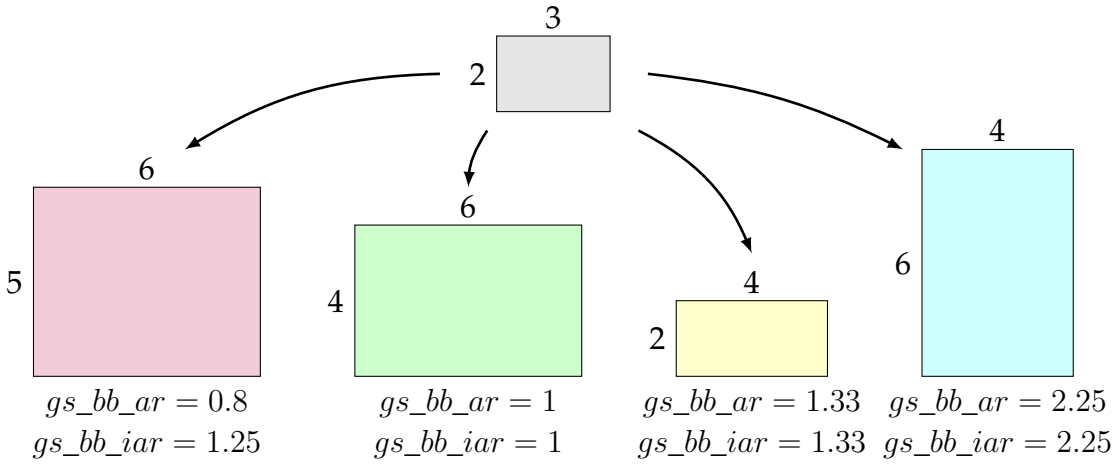


FIGURE 3.1 – *Impact de la variation du rapport de forme sur les métriques gs_bb_ar et gs_bb_iar*

Dans ce cas, la valeur cible est de 1 et la métrique ne peut atteindre des valeurs inférieures à celle-ci (cf. Figure 3.1, les valeurs sous les rectangles). Ce critère est celui que nous avons retenu pour notre étude.

Une alternative à cette approche basée sur l'enveloppe convexe a été proposée par STROBELT et al. (2012). L'idée est d'évaluer la distorsion de l'enveloppe convexe en comparant, entre les deux plongements, les distances des points de l'enveloppe convexe à leur centre. Soit ℓ_θ (*resp.* ℓ'_θ) la distance euclidienne entre le centre de masse c_{ch} (*resp.* c'_{ch}) de l'enveloppe convexe ch (*resp.* ch') et l'intersection de l'enveloppe convexe avec une ligne traversant c_{ch} (*resp.* c'_{ch}) avec un angle θ (θ variant de 0° à 350° avec un pas de 10°. Soit la différence définie comme le rapport $d_\theta = \ell'_\theta / \ell_\theta$. La métrique est l'écart-type des 36 mesures de d_θ :

$$\text{gs\_ch\_sd} = \sqrt{\frac{1}{36} \sum_{\substack{\theta=10k \\ k=0,\cdots,35}} (d_\theta - \overline{d})^2}$$

$$\text{où } \overline{d} = \frac{1}{36} \sum_{\substack{\theta=10k \\ k=0,\cdots,35}} d_\theta \text{ est la valeur moyenne}$$

Sur la base des expérimentations présentées dans la Section 3.4, nous avons observé que gs_bb_iar et gs_ch_sd ont un coefficient de corrélation de 0,77 qui montre qu'ils ont tous deux tendance à capturer des aspects similaires du processus d'ajustement. Nous avons choisi la première pour sa simplicité et sa facilité d'interprétation.

3.3.4 Minimisation du mouvement des nœuds

Cette classe contient les métriques quantifiant les changements de position des nœuds après l'exécution d'un algorithme d'ajustement. L'intuition sous-jacente est qu'un algorithme impliquant de grands mouvements des nœuds fournira une configuration sans chevauchements différente de celle d'origine, et peut donc entraîner une perte substantielle de la carte mentale.

La métrique la plus simple de cette classe a été présentée par Huang et al. (2007) :

$$\text{nm\_mn} = \frac{nb}{n}$$

Ici, nb représente le nombre de nœuds qui ont bougés entre le plongement initial et le plongement sans chevauchements. Le principal inconvénient de cette approche est qu'un algorithme de suppression des chevauchements des nœuds peut induire de très petits changements dans la plupart des nœuds, ce qui n'affecte pas la préservation de la carte mentale, tout en produisant de très mauvais résultat pour cette métrique. Pour pallier ce problème et ajouter plus de granularité à l'évaluation du mouvement des nœuds, une série de métriques, basées sur la même fonction de qualité, a été proposée :

$$\text{nm\_dm} = f(n) \times \sum_{v \in V} \text{dist}(v, v')$$

où f est une fonction de normalisation de $n = |V|$ et $dist$ est la distance entre v et v'. Le Tableau 3.2 résume les fonctions utilisées dans la littérature.

Table 3.2 – *Fonctions utilisées pour ajuster la métrique de mouvement*

$\text{dist}(v, v')$	$f(n)$		
	1	$1/n$	$1/(k\sqrt{2} \times n)$
$\|v' - v\|$		nm_dm_me	nm_dm_ne
$\|v' - v\|^2$	nm_dm_se	nm_dm_imse	
$\|x'_v - x_v| + |y'_v - y_v|$	nm_dm_h		

La fonction f se présente sous trois formes différentes. Marriott et al. (2003) et Huang et Lai (2003) n'en incluent pas, ce qui revient à avoir $f(n) = 1$. L'inconvénient est que la valeur obtenue dépend fortement du nombre de nœuds du graphe. C'est pourquoi Strobelt et al. (2012) ont proposé d'utiliser la moyenne des distances, ce qui revient à définir $f(n) = 1/n$. Finalement, Lyons et al. (1998) ont proposé $f(n) = 1/(k\sqrt{2} \times n)$ où k est le maximum entre w'_{bb} et h'_{bb}. Dans ce cas, $k\sqrt{2}$ est la diagonale du carré contenant le plongement et par conséquence la distance maximale disponible pour un nœud. Malheureusement, cette normalisation génère des valeurs très petites et est plus difficile à interpréter que $f(n) = 1/n$. C'est pourquoi nous avons préféré cette dernière pour notre étude.

Trois fonctions $dist$ ont été proposées dans la littérature. La plus intuitive est la distance euclidienne $\|v' - v\|$ (Lyons et al., 1998 ; Strobelt et al., 2012). Le carré de la distance euclidienne $\|v' - v\|^2$ (Marriott et al., 2003) évite le calcul de la racine carrée et discrimine mieux les grands mouvements. C'est celui que nous avons choisi pour notre étude. La distance de Manhattan $|x'_v - x_v| + |y'_v - y_v|$ a également été utilisée (Huang & Lai, 2003) mais est moins intuitive et produit des résultats proches (nm_dm_se et nm_dm_h ont un coefficient de corrélation de $0,9$).

Considérons un algorithme d'ajustement qui pousse les nœuds sur l'axe des x. La préservation du rapport de forme n'est pas optimale. Cependant la préservation de

la configuration devrait atteindre un bon score car un nœud en haut à droite dans le plongement initial restera en haut à droite dans le plongement sans chevauchements. Afin de mieux capturer le mouvement relatif d'un nœud entre les deux plongements, une fonction $shift$ peut être appliquée pour aligner le centre de la boîte de délimitation initiale avec le centre de la boîte finale et une fonction $scale$ pour aligner la taille de la boîte de délimitation initiale avec la taille de la boîte finale :

$$\begin{aligned} shift(v) =& (x_v + x'_{bb} - x_{bb}, y_v + y'_{bb} - y_{bb}) \\ scale(v) =& (x_v \times \frac{w'_{bb}}{w_{bb}}, y_v \times \frac{h'_{bb}}{h_{bb}}) \end{aligned}$$

Compte tenu de cela, nous avons choisi la métrique suivante pour le mouvement des nœuds :

$$\text{nm\_dm\_imse} = \frac{1}{n} \times \sum_{v \in V} \|v' - scale(shift(v))\|^2$$

nm_d (Gansner & Hu, 2010, la formule complète est disponible dans l'article) repose également sur l'idée que la métrique devrait être basée sur des positions initiales ajustées pour mieux capturer le mouvement relatif des nœuds entre les deux plongements. En plus d'inclure les fonctions $shift$ et $scale$, elle fait également pivoter le plongement initial d'un angle θ qui minimise les distances entre les nœuds du plongement initial et ceux du plongement sans chevauchements :

$$rotation(v) = (x_v \cos\theta - y_v \sin\theta, x_v \sin\theta + y_v \cos\theta)$$

Nous n'avons pas inclus la rotation dans notre expérience car nous considérons qu'elle peut induire une perte de la carte mentale (*e.g.* comment reconnaître une carte si celle-ci est retournée ?).

Une alternative pour quantifier à quel point une configuration sans chevauchements peut entraîner une perte substantielle de la carte mentale est de regarder les voisinages des nœuds et de les comparer avant et après l'ajustement. En se basant sur une approche k-NN, Nachmanson et al. (2016) ont proposé la métrique suivante :

$$\text{nm\_knn}(k) = \sum_{v \in V} (k - |N_k(v) \cap N_k(v')|)^2$$

où $N_k(v)$ (*resp.* $N_k(v')$) désigne les k plus proches voisins de v (*resp.* v') en terme de distance euclidienne, dans le plongement initial (*resp.* le plongement sans chevauchements). Nous n'avons pas retenu cette métrique car, contrairement aux autres métriques de cette classe, elle nécessite de fixer un paramètre (k).

3.3.5 Préservation des longueurs des arêtes

Cette classe contient les deux métriques basées sur la longueur des arêtes. L'ensemble des arêtes peut être E ou un autre ensemble dérivé du graphe.

Les algorithmes standards de plongement basés sur un modèle de force ont tendance à produire des longueurs d'arêtes uniformes. La première métrique de cette classe

calcule donc si les longueurs des arêtes d'un graphe restent uniformes ou non après l'application d'un algorithme d'ajustement (Li et al., 2005) :

$$\text{el\_r} = \frac{\max_{(u,v)\in E} \|u' - v'\|}{\min_{(u,v)\in E} \|u' - v'\|}$$

Comme de nombreux algorithmes de dessin de graphes ne sont pas conçus pour produire des longueurs d'arêtes uniformes, la préservation de la carte mentale n'est pas nécessairement prise en compte par ce type de métrique. Nous avons donc décidé d'envisager des alternatives.

La première alternative est basée sur les arêtes d'un graphe dérivé du plongement initial via une triangulation de Delaunay. Soit E_{dt} l'ensemble des arêtes d'une triangulation de Delaunay effectuée sur les nœuds du plongement initial. La deuxième métrique de cette classe, el_rsdd, est basée sur le calcul du coefficient de variation, également connu sous le nom d'écart-type relatif, du rapport des longueurs d'arêtes défini comme suit (Gansner & Hu, 2010) :

$$r_{uv} = \frac{||u' - v'||}{||u - v||}, \quad (u, v) \in E_{dt}$$

$$\overline{r} = \frac{1}{|E_{dt}|} \sum_{(u,v)\in E_{dt}} r_{uv}$$

$$\text{el\_rsdd} = \frac{\sqrt{\frac{1}{|E_{dt}|} \sum_{(u,v)\in E_{dt}} (r_{uv} - \overline{r})^2}}{\overline{r}}$$

Le principal inconvénient de cette métrique est qu'elle est basée sur un ensemble dérivé d'arêtes au lieu de l'ensemble réel. Par conséquent, elle ne permet de savoir que partiellement si un algorithme préserve ou non la longueur des arêtes. Dans notre étude, nous utilisons le coefficient de variation du rapport des longueurs d'arêtes, el_rsd (mêmes équations que pour el_rsdd en remplaçant E_{dt} par E).

3.4 Comparaison des algorithmes

Dans cette section, nous comparons 9 algorithmes de la littérature en termes de qualité et de temps d'exécution : échelle uniforme (*Scaling*), *PFS* (Misue et al., 1995), *PFS'* (Hayashi et al., 1998), *FTA* (Huang et al., 2007), *VPSC* (Dwyer et al., 2005), *PRISM* (Gansner & Hu, 2010), *RWordle-L* (Strobelt et al., 2012), *GTREE* (Nachmanson et al., 2016) et *Diamond* (Meulemans, 2019). La qualité d'un plongement sans chevauchements est évaluée à l'aide des métriques identifiées dans la section précédente, en suivant une procédure en 3 étapes :

1. **Jeux de données.** Nous générons 840 graphes synthétiques contenant de 10 à 1 000 nœuds. Ces graphes sont fournis par 4 modèles de génération disponibles dans la librairie *OGDF* (Chimani et al., 2013) : graphes aléatoires (Erdös & Rényi, 1959), arbres aléatoires, graphes petits-mondes (Watts & Strogatz, 1998) et graphes à invariant d'échelle (Barabaśi & Albert, 1999). Nous utilisons également 14

graphes du monde réel sélectionnés depuis la suite de tests *Graphviz* [4] (Gansner & North, 2000) et précédemment utilisés par les auteurs de *PRISM* (Gansner & Hu, 2010) et de *GTREE* (Nachmanson et al., 2016). Tous les graphes sont disponibles en ligne [5] sous la forme de fichiers *GML* comprenant le plongement initial.

2. **Calcul du plongement sans chevauchements.** Les graphes synthétiques résultants de la première étape sont initialement positionnés par l'algorithme de placement *FM*³ (Hachul & Jünger, 2004). Ensuite, nous appliquons les 9 algorithmes de suppression des chevauchements des nœuds, ce qui permet d'obtenir un ensemble de 7 560 plongements de graphes sans chevauchements. Les graphes de la suite de tests de *Graphviz* sont initialement positionnés par l'algorithme de positionnement *SFDP* pour suivre le même plongement de base que Gansner et Hu (2010). Nous appliquons ensuite les 9 algorithmes de suppression des chevauchements des nœuds, ce qui permet d'obtenir 126 plongements de graphe sans chevauchements.

3. **Calcul des métriques.** Enfin, nous calculons les valeurs des 5 métriques sélectionnées sur les 7 686 plongements de graphes synthétiques et réels sans chevauchements. Nous mesurons également le temps de calcul des algorithmes.

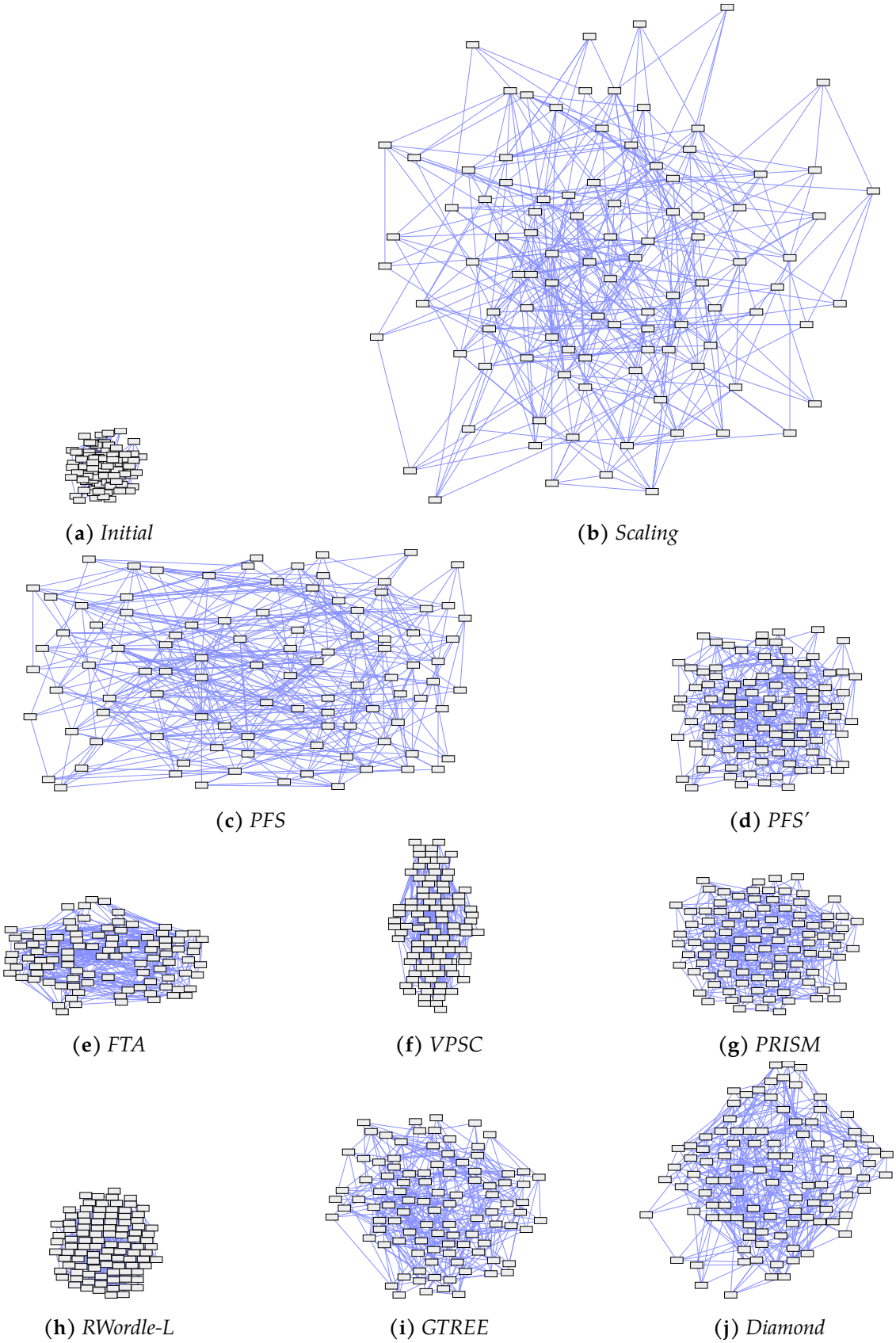
Tous les algorithmes sont implémentés en *JavaScript*. *PFS*, *PFS'*, *FTA* et *Diamond* ont été implémentés à partir des algorithmes fournis par les auteurs dans leurs articles respectifs. Comme *Diamond* est basé sur une optimisation linéaire, nous avons utilisé *jsLPSolver* [6]. Pour *VPSC*, nous avons directement utilisé le programme en *JavaScript* fourni par les auteurs [7]. *PRISM* et *GTREE* ont été adaptés de la librairie *Microsoft Automatic Graph Layout* [8] et convertis en *JavaScript* grâce à *SharpKit* [9]. Enfin, nous nous sommes inspirés du programme *Java* de *RWordle-L* [10] fourni par les auteurs de Strobelt et al. (2012) pour en faire une version *JavaScript*.

3.4.1 Qualité

La Figure 3.2(a) montre un graphe aléatoire contenant 100 nœuds et 400 arêtes, positionnés par l'algorithme de placement FM^3 (Hachul & Jünger, 2004). Ce plongement initial contient 274 chevauchements. Les Figures 3.2(b) à (j) illustrent les plongements sans chevauchements obtenus après l'application des algorithmes mentionnés ci-dessus. Les tailles des figures reflètent l'expansion des plongements.

La Figure 3.3(a) montre un autre exemple avec un graphe réel issu de la suite de tests *Graphviz*, *mode*. Il contient 213 nœuds, 269 arêtes et 1 105 chevauchements. Les Figures 3.3(b) à (j) illustrent les plongements sans chevauchements. Dans ce cas, nous

4. https://gitlab.com/graphviz/graphviz/-/tree/main/rtest/graphs/
5. https://github.com/agorajs/agora-dataset
6. https://github.com/JWally/jsLPSolver
7. https://github.com/tgdwyer/WebCola
8. https://github.com/microsoft/automatic-graph-layout
9. https://github.com/SharpKit
10. https://github.com/HendrikStrobelt/ditop_server/blob/master/src/main/java/de/hs8/graphics/RWordle.java



(**a**) *Initial* (**b**) *Scaling*

(**c**) *PFS* (**d**) *PFS'*

(**e**) *FTA* (**f**) *VPSC* (**g**) *PRISM*

(**h**) *RWordle-L* (**i**) *GTREE* (**j**) *Diamond*

Figure 3.2 – *Plongements sans chevauchements obtenus après application des algorithmes sur le plongement initial (a) d'un graphe aléatoire contenant 274 chevauchements*

n'avons pas conservé la taille relative pour *Scaling* et *PFS* car le dessin était trop grand. Les plongements réels sont deux fois plus grands qu'ils n'apparaissent sur la figure.

Le Tableau 3.3 présente les valeurs agrégées des métriques obtenues sur les graphes synthétiques : pour chacune des cinq métriques sélectionnées et pour chaque algorithme, le premier quartile, la médiane et le troisième quartile des valeurs sont indiqués. Le Tableau 3.4 montre les valeurs des métriques sur les graphes réels. Dans ces tableaux et les suivants, la couleur des cases représente la qualité de l'algorithme sur le critère : vert pour une qualité élevée, orange pour une qualité intermédiaire et rouge pour une qualité médiocre. Les fourchettes sont définies en comparant les valeurs situées sur une seule ligne, *i.e.* les valeurs d'un seul critère obtenues avec les différents algorithmes.

Table 3.3 – *Valeurs agrégées des métriques sélectionnées sur les graphes synthétiques : premier quartile, médiane et troisième quartile*

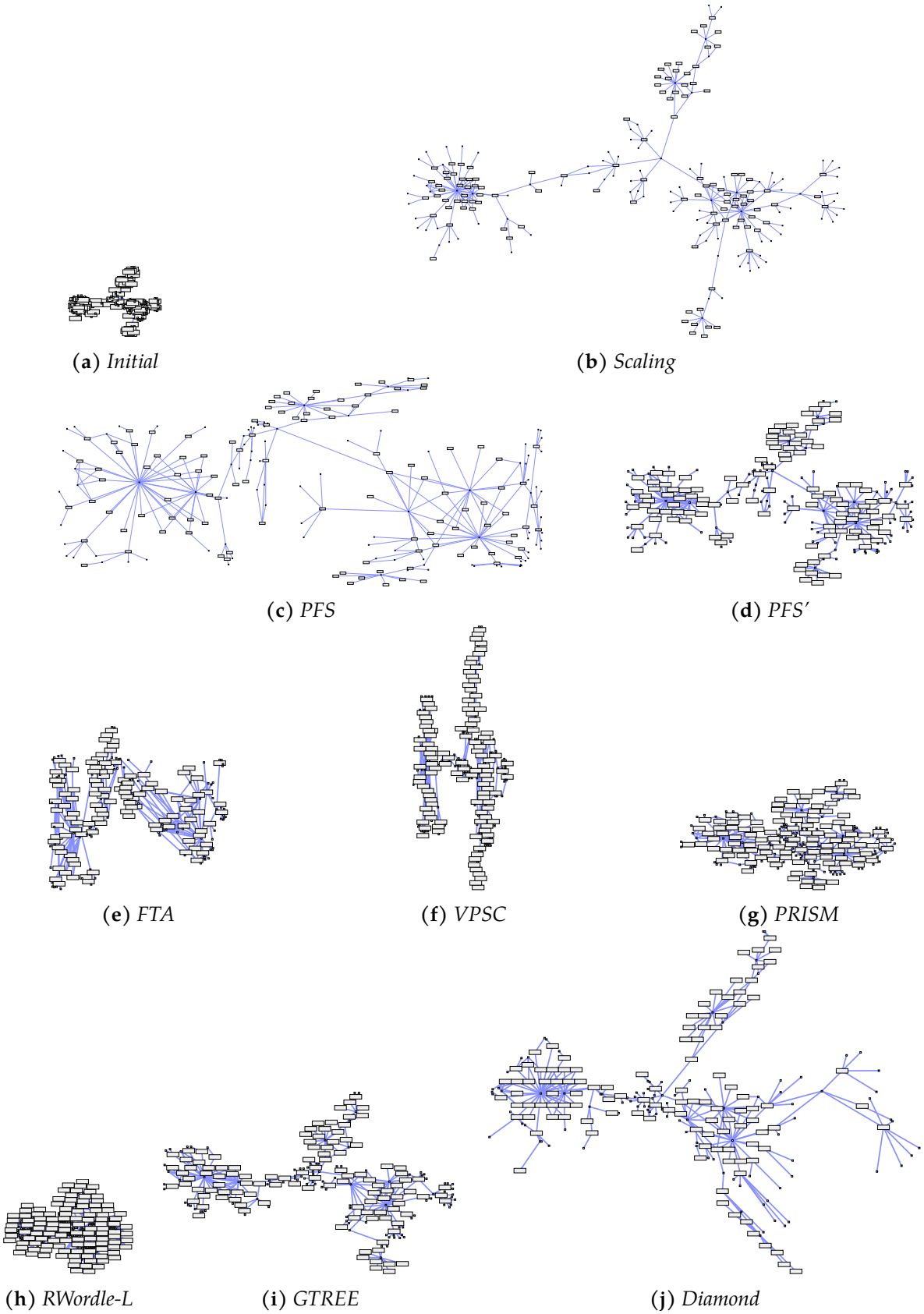
		Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
oo_nni	*Q1*	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,04
	Med.	0,00	0,00	0,00	0,01	0,00	0,01	0,01	0,02	0,06
	Q3	0,00	0,00	0,00	0,04	0,03	0,02	0,10	0,03	0,09
sp_ch_a	*Q1*	1,96	1,04	1,00	1,00	1,00	1,01	1,00	1,12	1,16
	Med.	8,70	1,69	1,22	1,02	1,00	1,12	1,01	1,49	1,96
	Q3	34,29	17,63	5,04	4,21	2,30	4,22	1,99	5,94	6,94
gs_bb_iar	*Q1*	1,00	1,01	1,00	1,00	1,00	1,00	1,00	1,01	1,04
	Med.	1,01	1,19	1,04	1,01	1,00	1,04	1,00	1,04	1,07
	Q3	1,06	1,65	1,14	1,66	1,94	1,23	1,07	1,08	1,12
nm_dm_imse	*Q1*	0,00	6,50	1,65	0,94	0,42	9,82	2,18	28,22	535,65
	Med.	0,00	694,83	116,06	37,87	9,71	131,57	27,60	292,56	1 971,14
	Q3	0,00	7 899,17	1 782,78	2 965,99	283,59	688,11	597,66	2 221,69	16 571,10
el_rsd	*Q1*	0,00	0,03	0,02	0,02	0,01	0,04	0,04	0,05	0,23
	Med.	0,00	0,19	0,11	0,12	0,08	0,16	0,12	0,16	0,66
	Q3	0,00	0,25	0,18	0,36	0,19	0,21	0,25	0,21	0,99

Table 3.4 – *Valeurs moyennes des métriques sélectionnées sur les graphes réels*

	Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
oo_nni	0,00	0,00	0,00	0,07	0,01	0,02	0,04	0,02	0,05
sp_ch_a	217,26	210,28	6,92	6,03	2,36	2,18	1,53	4,02	30,63
gs_bb_iar	1,04	1,97	1,22	1,95	2,20	1,33	1,04	1,17	1,10
nm_dm_imse	0,00	9 768 157,94	63 594,74	4 297 355,84	34 611,40	42 919,66	35 928,91	37 331,83	1 348 426,00
el_rsd	0,02	0,34	0,22	0,54	0,28	0,28	0,36	0,26	0,23

Préservation de l'ordre orthogonal Sans surprise, *Scaling*, *PFS* et *PFS'* obtiennent les meilleurs résultats sur *oo_nni* car il est prouvé qu'ils maintiennent l'ordre orthogonal d'origine. Cependant, tous les algorithmes testés ont obtenu de bons résultats pour ce critère.

Minimisation de l'expansion Comme l'illustre la Figure 3.2(b), *Scaling* augmente fortement la taille du plongement, ce qui induit un mauvais score pour *sp_ch_a*. *PFS* obtient également un mauvais score pour ce critère. *VPSC* et *RWordle-L* produisent



(**a**) *Initial* (**b**) *Scaling*

(**c**) *PFS* (**d**) *PFS'*

(**e**) *FTA* (**f**) *VPSC* (**g**) *PRISM*

(**h**) *RWordle-L* (**i**) *GTREE* (**j**) *Diamond*

Figure 3.3 – *Plongements sans chevauchements obtenus après application des algorithmes sur le plongement initial (a) d'un graphe réel contenant 1 105 chevauchements*

les plongements les plus compacts tandis que les autres algorithmes donnent des résultats intermédiaires. Cependant, en observant les Figures 3.2(f), (h), 3.3(f) et (h), on peut constater que les plongements résultants de ces deux algorithmes sont si compacts qu'ils ne permettent pas de visualiser les arêtes et les structures du graphe (*e.g.* les communautés ou les clusters). Selon la tâche que l'on souhaite effectuer sur le plongement sans chevauchements, cette observation illustre une limite possible du critère lorsqu'il est considéré indépendamment des autres.

Préservation du rapport de forme De manière surprenante, le score de préservation du rapport de forme (gs_bb_iar) n'est pas exactement de 1 pour *Scaling* en raison de la taille des nœuds qui reste la même entre le plongement initial et le plongement sans chevauchements. Néanmoins, elle préserve le rapport de forme initiale. *PFS* est le pire algorithme sur ce critère. Les autres algorithmes ont obtenu de bons scores médians sur les graphes synthétiques mais les scores du troisième quartile montrent que *FTA* et *VPSC* peuvent produire une certaine quantité de plongements déformés. Ceci est confirmé par les tests sur les graphes réels où ils obtiennent de moins bons résultats et sur les Figures 3.2(e), (f) et 3.3(f) où l'on peut observer qu'ils étendent le dessin le long d'un seul des axes (l'axe x pour *FTA* et l'axe y pour *VPSC*).

Minimisation du mouvement des nœuds *Scaling* obtient les meilleurs résultats pour le critère de minimisation du mouvement des nœuds suivi par *VPSC* et *RWordle-L*. *FTA* a également obtenu un bon score médian sur les graphes synthétiques mais sa valeur du troisième quartile montre qu'il peut générer une certaine quantité de plongements avec des changements élevés, comme l'illustre également le mauvais score obtenu sur les graphes réels. *PFS'* et *PRISM* ont obtenu des résultats intermédiaires. *GTREE* a obtenu de mauvais résultats sur les graphes synthétiques alors qu'il en a obtenu d'assez bons sur les graphes réels. Enfin, *PFS* et *Diamond* ont obtenu de mauvais résultats sur les graphes synthétiques et réels.

Préservation des longueurs des arêtes *Scaling* préserve les longueurs relatives des arêtes. *Diamond* obtient les pires scores sur les graphes synthétiques mais ce phénomène ne se confirme pas sur les graphes réels pour lesquels il obtient d'assez bons scores. Tous les autres algorithmes ont obtenu un score médian compris entre 0,08 et 0,36 sur les graphes synthétiques. Le troisième quartile montre que *FTA* génère un certain nombre de chevauchements présentant des variations des longueurs d'arête plus importantes. Cette observation est confirmée par les résultats sur les graphes réels pour lesquels il obtient le plus mauvais score.

3.4.2 Temps de calcul

Les Tableaux 3.5 et 3.6 montrent les valeurs agrégées des temps d'exécution, mesurées avec notre implémentation des algorithmes, en millisecondes sur les graphes synthétiques (premier quartile, médiane et troisième quartile) et les valeurs des temps d'exécution sur les graphes réels.

Nous pouvons observer sur les graphes synthétiques que *Scaling*, *PFS*, *PFS'* et *VPSC*

Table 3.5 – *Temps d'exécution agrégés en millisecondes sur les graphes synthétiques, en fonction du nombre de nœuds (10 à 1 000) : premier quartile, médiane et troisième quartile*

		Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
10	*Q1*	0,00	0,00	0,00	0,00	0,00	3,00	0,00	3,00	1,00
	Med.	0,00	0,00	0,00	0,00	0,00	4,00	0,00	5,00	1,00
	Q3	0,00	0,00	0,00	0,00	0,00	12,00	0,00	12,00	1,00
20	*Q1*	0,00	0,00	0,00	0,00	0,00	4,00	0,00	5,00	3,00
	Med.	0,00	0,00	0,00	0,00	0,00	6,00	0,00	12,00	3,00
	Q3	0,00	0,00	0,00	0,00	1,00	9,00	1,00	16,00	4,00
50	*Q1*	0,00	0,00	0,00	0,00	1,00	7,00	0,00	19,00	15,00
	Med.	0,00	0,00	0,00	1,00	1,00	16,00	3,00	38,00	22,00
	Q3	0,00	0,00	1,00	2,00	2,00	41,25	9,25	50,00	27,00
100	*Q1*	0,00	0,00	1,00	1,00	2,00	25,75	0,00	85,00	145,80
	Med.	0,00	1,00	1,00	4,00	3,00	44,00	36,50	125,00	238,50
	Q3	1,00	1,00	1,00	12,00	4,00	69,00	84,25	156,25	318,50
200	*Q1*	1,00	2,00	4,00	4,00	4,00	40,75	2,00	307,50	3 230,00
	Med.	1,00	2,00	4,00	25,50	10,50	81,50	274,00	422,00	6 358,00
	Q3	2,00	3,00	5,00	89,00	15,00	134,25	589,50	587,50	7 960,00
500	*Q1*	2,00	15,00	25,00	26,00	25,00	196,75	21,00	1 930,00	149 824,00
	Med.	10,00	17,00	29,50	207,50	93,00	308,00	3 410,00	3 126,00	302 230,00
	Q3	13,00	20,00	35,00	1 392,00	140,75	487,25	7 246,25	4 060,00	390 922,00
1 000	*Q1*	4,75	57,00	113,50	87,75	125,50	623,75	112,00	9 187,00	3 765 418,00
	Med.	34,50	67,50	133,00	1 022,00	496,00	1 053,00	26 877,50	16 382,00	7 450 774,00
	Q3	58,25	77,00	176,25	8 694,00	1 065,75	1 440,00	55 173,25	19 892,00	9 775 764,00

Table 3.6 – *Temps d'exécution en millisecondes sur les graphes réels*

	Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
b100	41	82	126	19 823	321	138 528	191 109	32 236	44 786 574
b102	5	16	24	82	15	3 196	96	919	52 913
b124	0	0	0	3	1	308	1	54	199
b143	0	0	1	9	2	351	6	125	982
badvoro	20	43	63	32 553	383	50 360	56 155	26 140	23 981 749
dpd	0	0	0	1	1	146	0	80	47
mode	0	1	2	93	4	1 891	281	422	3 960
NaN	0	1	1	2	0	113	0	38	154
ngk10_4	0	0	0	0	0	36	0	23	55
root	7	24	49	10 866	85	70 724	43 169	17 759	6 093 992
rowe	1	0	0	1	0	223	0	102	132
size	0	0	0	1	0	208	1	88	53
unix	0	0	0	1	0	68	0	44	41
xx	1	3	5	27	7	2 789	60	939	58 866

nécessitent un temps d'exécution plus faible que les autres algorithmes. *FTA* est un peu plus lent, surtout lorsqu'on regarde le troisième quartile, ce qui indique un certain coût en temps lors du calcul du plongement (plus d'une seconde pour les graphes contenant plus de 500 nœuds). Cette observation est confirmée sur les graphes réels.

RWordle-L, *PRISM* et *GTREE* induisent des temps d'exécution intermédiaires sur les graphes synthétiques : moins de 1 seconde pour les graphes contenant jusqu'à 200 nœuds, quelques secondes pour les graphes de 500 nœuds, et des dizaines de secondes pour les graphes de 1 000 nœuds. *PRISM* et *GTREE* sont significativement plus lents

que *RWordle-L* sur les petits graphes (nombre de nœuds inférieur ou égal à 100) mais cela ne semble pas avoir d'importance car les valeurs restent très faibles. Les graphes réels confirment ces observations mais soulignent également que *PRISM* est parfois significativement plus lent que *GTREE*, même si cela ne se produit que sur les graphes nécessitant peu de temps de calcul.

Diamond est souvent l'algorithme qui prend le plus de temps sur les graphes synthétiques et réels. Cependant, il est basée sur une optimisation linéaire et le temps d'exécution dépend donc du solveur utilisé (voir l'introduction de cette section, Section 3.4). Cela pourrait expliquer les différences entre nos résultats et ceux donnés dans l'article de Meulemans (2019).

3.4.3 Synthèse

En conclusion, même si *Scaling* optimise 4 des 5 critères et est très rapide à calculer sur les graphes de nos jeux de données, il ne représente pas une solution satisfaisante car il augmente trop la taille du plongement. *PFS* n'est pas non plus satisfaisant car il a obtenu de mauvais résultats sur 3 critères. En particulier, il augmente aussi considérablement la taille du plongement comme l'illustrent les Figures 3.2(c) et 3.3(c).

FTA a obtenu des résultats intermédiaires sur tous les critères, ce qui est moins bon que tous ses autres concurrents. En particulier, comme nous l'avons déjà mentionné, il allonge le plongement le long d'un seul axe, ce qui déforme fortement la configuration d'origine (voir le critère *Préservation du rapport de forme* gs_bb_iar dans le Tableau 3.3, ainsi que la distorsion illustrée par la Figure 3.2(e)).

VPSC possède également cette propriété. En examinant le Tableau 3.3, nous pouvons observer que *RWordle-L* est meilleur que *VPSC* sur *Préservation du rapport de forme*, tout en obtenant des résultats comparables sur les autres critères. Ceci est dû au fait qu'ils créent les plongements les plus compacts induisant ainsi une faible dispersion et de courts mouvements de nœuds. Nous pouvons également remarquer sur les Tableaux 3.5 et 3.6 que *RWordle-L* peut prendre du temps pour les graphes de plus de 500 nœuds, ce qui n'est pas le cas de *VPSC*. Ainsi, si la compacité du plongement est une priorité, *RWordle-L* devrait être choisi sur les petits graphes et *VPSC* sur les plus grands.

La grande compacité des plongements résultant de *RWordle-L* et *VPSC* ne permet pas de visualiser les arêtes et les structures du graphe. Par conséquent, si la priorité est de fournir un plongement mettant en évidence les chemins et les groupes de nœuds dans le graphe, les options restantes (*PFS'*, *PRISM*, *GTREE* et *Diamond*) doivent être privilégiées. Parmi eux, *Diamond* est le plus lent pour le solveur que nous avons utilisé (voir l'introduction de cette section, Section 3.4). *Diamond* obtient également de mauvais scores pour *Minimisation du mouvement des nœuds* et *Préservation des longueurs des arêtes* sur les graphes synthétiques (cf. Tableau 3.3, nm_dm_imse et eb_rsdd). *GTREE* induit également beaucoup de mouvements de nœuds, mais il surpasse *Diamond* en termes de *Préservation des longueurs des arêtes* et en temps d'exécution. *PFS'* et *PRISM* ont obtenu des résultats comparables, surpassant *GTREE* et *Diamond* sur la *Minimisation du mouvement des nœuds*, même si *PRISM* est légèrement meilleur (cf. Tableaux 3.3 et 3.4, nm_dm_imse). Les Figures 3.2(d) et (g) illustrent leur similarité tandis que les

Figures 3.3(d) et (g) illustrent la capacité de *PRISM* à induire moins de mouvements de nœuds sur un graphe réel. *PFS'* devrait être préféré à *PRISM* pour les grands graphes car son temps de calcul est sensiblement inférieur (cf. Tableaux 3.5 et 3.6) tandis que *PRISM* devrait être préféré pour les petits.

3.5 AGORA

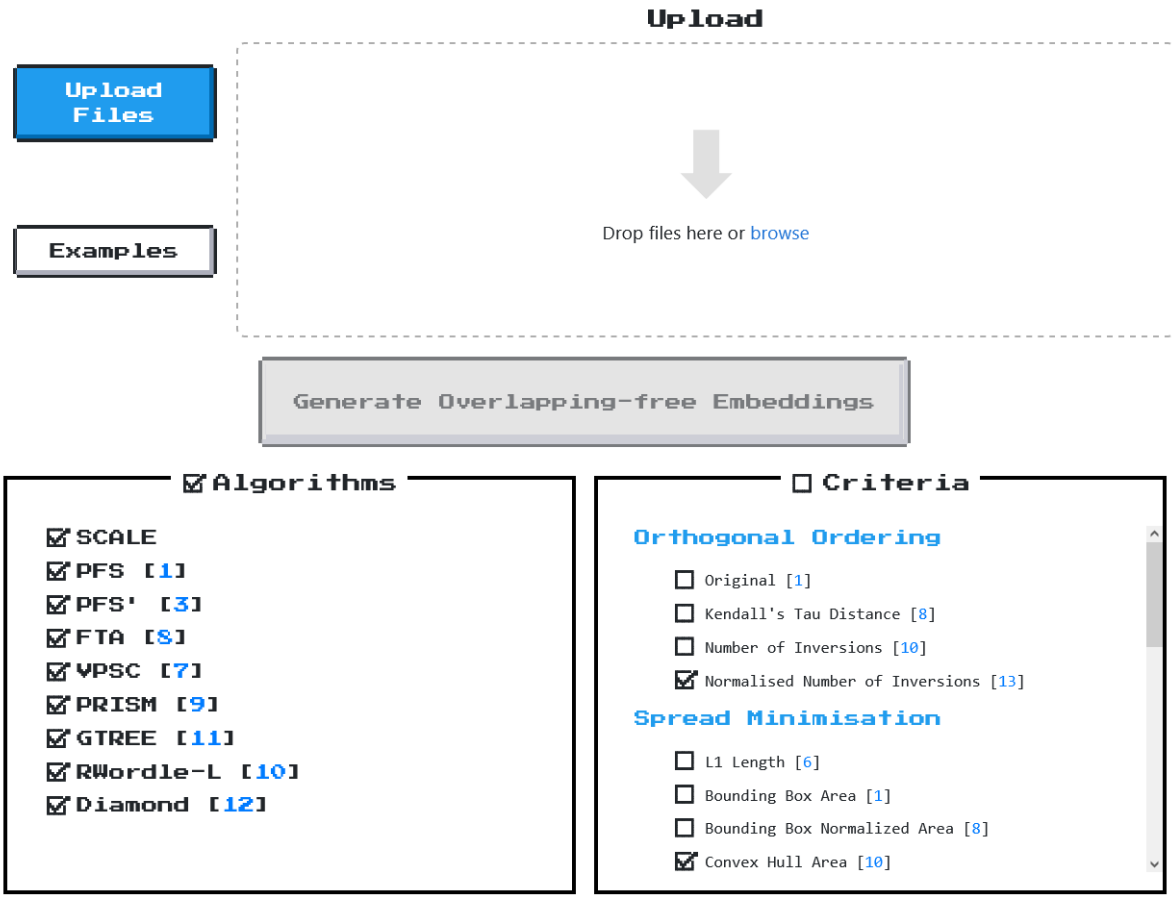


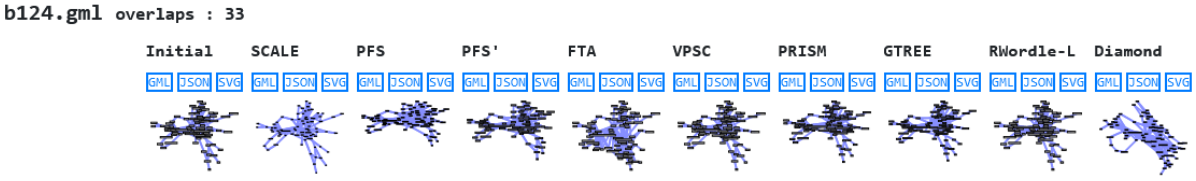
Figure 3.4 – *Aperçu de l'interface du site Web* AGORA. *Dans la partie haute, il est possible de déposer ses fichiers ou d'en sélectionner parmi les exemples fournis via le bouton « Examples ». La partie basse présente les algorithmes et métriques que l'utilisateur peut sélectionner via des cases à cocher.*

Tous les algorithmes de suppression des chevauchements des nœuds ainsi que les critères décrits dans ce chapitre sont disponibles dans une bibliothèque *JavaScript* [11]. Les implémentations sont celles utilisées pour les expériences décrites dans la section précédente.

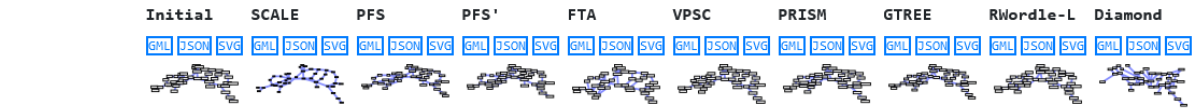
Une plateforme Web, *AGORA* [12] (*Automatic Graph Overlap Removal Algorithms* – algorithmes automatiques de suppression des chevauchements des graphes) est également disponible en ligne (cf. Figure 3.4). L'utilisateur peut sélectionner un ou plusieurs

11. https://github.com/agorajs/agorajs.github.io
12. https://agorajs.github.io/

graphes réels utilisés pour les expériences ou importer ses propres graphes au format GML. Les graphes doivent contenir les coordonnées des nœuds, x et y, ainsi que leur largeur et leur hauteur, w et h. Il peut ensuite sélectionner un ou plusieurs algorithmes de suppression des chevauchements des nœuds parmi les neuf proposés dans l'interface (correspondant à ceux des expérimentations de la section précédente). Enfin, il peut sélectionner les métriques parmi les 22 métriques présentées ci-dessus. Par défaut, les cinq métriques que nous avons retenues précédemment sont sélectionnées. Une fois ces paramètres choisis, l'utilisateur peut générer les plongements sans chevauchements. Un plongement, affiché sous forme de vignette, est fourni pour chaque graphe et chaque algorithme (cf. Figure 3.5). L'utilisateur peut alors télécharger le résultat sous forme de fichier JSON ou GML. Dans ce cas, même si elles n'apparaissent pas dans les vignettes, les propriétés des nœuds et des arêtes des fichiers originaux sont conservées et disponibles dans le fichier de sortie, seules les valeurs de position (x et y) sont modifiées. L'utilisateur peut également télécharger une vignette sous forme de fichier SVG. Enfin, un tableau avec les valeurs des métriques sélectionnées pour chaque plongement est fourni.



Criterias	SCALE	PFS	PFS'	FTA	VPSC	PRISM	GTREE	RWordle-L	Diamond
Normalised Number of Inversions	0	0.4976	0.4976	0.0351	0.005	0.0214	0.0206	0.4568	0.123
Convex Hull Area	9.3295	2.0134	1.2161	1	1	1.1736	1.3511	1	3.388
Improved Aspect Ratio	1.04	1.5418	1.1782	1	1	1.1264	1.153	1	1.0607
Improved Mean Squared Euclidean	0	77766.6946	47049.1919	38207.674	360.8172	11212.9358	9576.4236	780.9736	331684.9676
Relative Standard Deviation	0	0.4252	0.4367	0.5217	0.1585	0.3526	0.2842	0.2499	0.4505



Criterias	SCALE	PFS	PFS'	FTA	VPSC	PRISM	GTREE	RWordle-L	Diamond
Normalised Number of Inversions	0	0.3354	0.3354	0.0445	0.011	0.014	0.0116	0.4701	0.1494
Convex Hull Area	5.4593	1.5508	1.1458	1.1195	1.0166	1.0664	1.288	1.0018	2.0452
Improved Aspect Ratio	1.0159	1.1797	1.1501	1.0183	1.0179	1.0009	1.1055	1.0185	1.0933
Improved Mean Squared Euclidean	0	2687.2703	1397.5281	3802.2203	233.9555	773.1604	1052.4408	466.8422	29291.8465
Relative Standard Deviation	0	0.258	0.222	0.2776	0.1761	0.2425	0.1862	0.1586	0.465

FIGURE 3.5 – *Aperçu des résultats sur le site Web* AGORA. *Deux fichiers ont été sélectionnés :* `b124.gml` *et* `unix.gml`. *Pour chacun des fichiers, un aperçu du plongement initial et des différents plongements issus des approches de suppression de chevauchements sélectionnées sont affichés, ainsi que les liens pour télécharger ces plongements et les scores des métriques sélectionnées.*

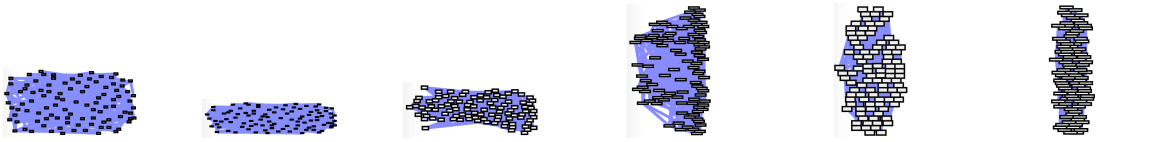
3.6 Discussion

Dans cette section, nous discutons de 2 facteurs qui pourraient influer sur notre étude comparative, le rapport de forme des nœuds et le nombre de chevauchements.

Rapport de forme des nœuds Le rapport d'aspect des graphes synthétiques de l'étude ci-dessus est de 2:1 alors que les rapports d'aspect des graphes réels varient en fonction des données initiales. Le rapport d'aspect fixe des graphes synthétiques pourrait être considéré comme une limite de notre étude : que se passe-t-il en terme de qualité des résultats entre les différents algorithmes lorsque le rapport de forme varie, et en particulier lorsque la largeur des nœuds augmente pour afficher de longues étiquettes de texte ? Un indice pour répondre à cette question est disponible dans le Tableau 3.7, qui montre les résultats pour le graphe de la Figure 3.2 avec un rapport d'aspect de 2:1 et le même graphe avec un rapport d'aspect de 5:1. Dans cet exemple, les métriques mettent principalement en évidence les mêmes propriétés pour les deux rapports de forme. Les principales différences apparaissent sur le rapport de forme global (gs_bb_iar) pour *PFS*, *FTA* et *VPSC*. Comme l'illustre la Figure 3.6, l'inconvénient de l'expansion du plongement sur une dimension est accentuée lorsque la largeur des nœuds augmente pour *PFS* et *VPSC*. À l'inverse, ce phénomène est atténué pour *FTA*. Cependant, comme nous pouvons le voir sur la figure, cela est dû au déplacement important d'un groupe de nœuds le long de l'axe vertical sur la partie droite du plongement, ce qui n'est pas un signe de la qualité du résultat obtenu.

Table 3.7 – *Valeurs des métriques sélectionnées sur le graphe de la Figure 3.2 avec deux rapports de forme des nœuds : 2:1 et 5:1*

		Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
oo_nni	2:1	0,00	0,46	0,46	0,06	0,03	0,02	0,48	0,02	0,03
	5:1	0,00	0,46	0,46	0,02	0,01	0,03	0,48	0,03	0,04
sp_ch_a	2:1	19,52	18,51	4,96	4,28	2,53	4,63	2,04	5,97	6,80
	5:1	61,09	59,08	11,41	11,17	4,42	7,10	4,05	10,80	29,16
gs_bb_iar	2:1	1,06	1,87	1,23	2,48	1,97	1,29	1,03	1,19	1,11
	5:1	1,28	3,26	1,31	2,17	3,91	1,70	1,29	1,33	1,29
nm_dm_imse	2:1	0,00	1 646,75	182,01	1 173,22	210,58	196,85	464,28	302,61	553,71
	5:1	0,00	14 867,75	1 790,54	4 549,93	298,20	630,09	2 350,57	1 723,94	2 390,43
el_rsd	2:1	0,00	0,26	0,13	0,38	0,33	0,22	0,41	0,17	0,17
	5:1	0,00	0,41	0,19	0,36	0,42	0,30	0,62	0,26	0,18



(**a**) PFS *2:1* (**b**) PFS *5:1* (**c**) FTA *2:1* (**d**) FTA *5:1* (**e**) VPSC *2:1* (**f**) VPSC *5:1*

Figure 3.6 – *Plongements sans chevauchements obtenues après l'application de* PFS, FTA *et* VPSC *sur un graphe avec différents rapports de forme des nœuds*

Nombre de chevauchements Un autre facteur qui pourrait limiter les résultats est le nombre de chevauchements du plongement initial : certains algorithmes sont-ils adaptés pour obtenir de meilleures mesures de qualité pour peu ou beaucoup de chevauchements ? Le Tableau 3.8 montre les résultats obtenus sur le graphe de la Figure 3.2 avec différentes tailles de nœuds mais avec le même rapport de forme. La taille initiale, 20×10, crée 274 chevauchements sur le plongement initial, tandis que l'autre, 40×20, crée 1 136 chevauchements. Là encore, les différentes métriques classent les algorithmes dans le même ordre pour les deux graphes. Les principales différences concernent l'effet de *FTA* et *VPSC* sur le rapport de forme mais cette fois-ci *PFS* semble avoir un rapport de forme pas aussi accentué. En effet, l'augmentation du nombre de chevauchements accentue l'étalement le long d'une dimension pour *FTA* mais pas pour *PFS*. *VPSC* présente le même comportement que lorsque nous avons modifié le rapport de forme.

Table 3.8 – *Valeurs des métriques sélectionnées sur le graphe de la Figure 3.2 avec 274 et 1 136 chevauchements*

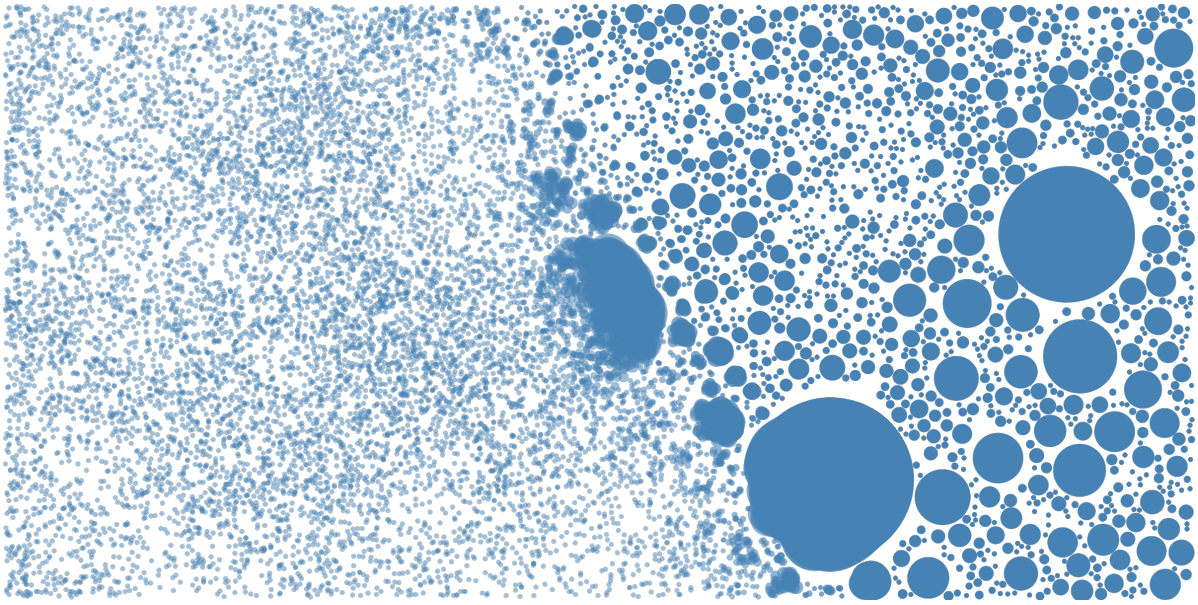
		Scaling	*PFS*	*PFS'*	*FTA*	*VPSC*	*PRISM*	*RWordle-L*	*GTREE*	*Diamond*
oo_nni	274	0,00	0,46	0,46	0,06	0,03	0,02	0,48	0,02	0,03
	1 136	0,00	0,46	0,46	0,06	0,02	0,02	0,47	0,04	0,03
sp_ch_a	274	19,52	18,51	4,96	4,28	2,53	4,63	2,04	5,97	6,80
	1 136	59,96	99,70	15,14	20,36	6,71	10,64	6,38	14,99	19,42
gs_bb_iar	274	1,06	1,87	1,23	2,48	1,97	1,29	1,03	1,19	1,11
	1 136	1,13	1,72	1,17	1,95	4,65	1,33	1,25	1,02	1,10
nm_dm_imse	274	0,00	1 646,75	182,01	1 173,22	210,58	196,85	464,28	302,61	553,71
	1 136	0,00	9 528,27	860,05	27 539,34	938,41	637,20	6 547,33	1 480,85	1 268,12
el_rsd	274	0,00	0,26	0,13	0,38	0,33	0,22	0,41	0,17	0,17
	1 136	0,00	0,25	0,13	0,49	0,46	0,22	0,84	0,18	0,17

3.7 Conclusion

Trouver un algorithme adéquat de suppression des chevauchements des nœuds est difficile pour un concepteur de visualisation car même si de nombreux algorithmes existent, aucune comparaison complète basée sur les mêmes critères n'a été fournie. Dans ce chapitre, nous avons d'abord mis en évidence les cinq classes principales de critères existants et proposé une métrique représentative pour chacune de ces classes. Via un grand nombre d'expérimentations réalisées sur des graphes synthétiques et réels, nous avons comparé 9 algorithmes de l'état de l'art en fonction des métriques retenues et des temps d'exécution. En analysant les résultats, nous avons ensuite montré les avantages, les inconvénients et les limites des algorithmes afin de pouvoir aider le concepteur de visualisation. Enfin, nous avons proposé une bibliothèque *Javascript* contenant tous les algorithmes et critères pour la suppression des chevauchements des nœuds ainsi qu'une plateforme Web, *AGORA*, qui permet à l'utilisateur, pour ses propres graphes, d'obtenir des plongements et les valeurs des métriques selon les algorithmes sélectionnés.

CHAPITRE 4

F-SAC



Sommaire

Nous avons pu constater dans le Chapitre 2 qu'il est parfois indispensable de représenter de nombreuses informations sur une carte. Une solution simple consiste à représenter l'ensemble de tous les éléments sous la forme de points sur la carte. Cependant ce choix de représentation est très limité dans la mesure où un trop grand nombre de points peut entraîner un problème d'encombrement visuel. La question qui se pose alors est : *comment limiter cet encombrement visuel ?* Il existe dans la littérature de nombreuses approches. Dans ce chapitre, nous nous intéressons aux approches basées sur un Regroupement Spatial Aggloméaratif. Elles permettent de créer une « *vue simplifiée* » dans laquelle les points qui se chevauchent sont regroupés dans des clusters dont la taille est proportionelle au nombre de points qu'ils contiennent. Ainsi, une telle vue permet à l'utilisateur d'appréhender facilement une densité d'éléments dans une région tout en gardant un aperçu des points isolés. Il existe dans la littérature différentes approches pour obtenir ces clusters, classées en deux catégories : celles qui extraient les clusters en temps réel dès que l'utilisateur interagit ou bien celles qui pré-calculent les clusters dans une étape de pré-traitement. Dans ce chapitre, nous présentons les approches existantes et leurs limites. Nous proposons également une nouvelle approche, *F-SAC*, utilisée dans *ProsoVis*. Elle est plus rapide que les approches de l'état de l'art, aussi bien en temps réel qu'en pré-traitement, et offre une qualité de cartes produites équivalente.

4.1 Contexte

L'encombrement est un problème majeur lorsque l'on veut visualiser de nombreux objets graphiques au sein d'une seule visualisation. Ce problème se produit souvent dans les diagrammes nœuds-liens, les nuages de points, les cartes avec symboles, etc. Il existe différentes façons de le résoudre, comme par exemple, en déplaçant les objets pour supprimer les chevauchements (Chen et al., 2020), en représentant la densité d'objets plutôt que les objets individuels (Bertin, 2011), en filtrant certains objets par sous-échantillonnage (de Berg et al., 2004) ou en regroupant les objets en clusters.

Cependant l'utilisation de ces méthodes soulève d'autres problèmes. Par exemple, les approches de suppression des chevauchements (cf. Chapitre 3) déforment la vue en modifiant la position des objets, les représentations de la densité (*e.g.* les cartes de chaleurs *heat maps*) ou le sous-échantillonnage produisent une perception biaisée en raison des objets non affichés (Ellis & Dix, 2002 ; Rosenholtz et al., 2007) et les approches par regroupement (*clustering*) induisent une perte de détails.

Dans ce chapitre, nous nous concentrons sur le regroupement (*clustering*), plus précisément sur les algorithmes de Regroupement Spatial Agglomératif (par la suite nous utiliserons l'acronyme SAC pour *Spatial Agglomerative Clustering*). Partant d'un fond de carte et d'une liste de points géolocalisés, l'objectif est de fusionner les points qui se chevauchent en clusters et d'afficher les autres points et ces clusters sur la carte afin d'obtenir une visualisation sans chevauchements. Contrairement aux méthodes basées sur la densité, ces approches permettent de représenter des points/clusters individuels (permettant ainsi une sélection) et ne nécessitent pas de codage couleur pour représenter la densité de l'information (qui peut alors être utilisé pour représenter une autre dimension des données).

Nous envisageons deux variantes du problème : (1) le problème *SAC en ligne* considérant une seule carte, (2) le problème *SAC hors ligne* considérant une liste de cartes représentant la même surface à différents niveaux de zoom. Notre contribution se présente sous cinq formes :

1. Nous formalisons les deux variantes (*en ligne* et *hors ligne*) du problème SAC.
2. Nous proposons un nouvel algorithme, *F-SAC* conçu pour être rapide en temps de calcul.
3. Nous discutons de la façon d'étendre *F-SAC* et les algorithmes de la littérature pour traiter les deux variantes du problème.
4. Nous proposons une liste de métriques pour comparer les plongements fournis par les algorithmes de regroupement spatial.
5. Nous comparons, en termes de qualité et de temps d'exécution, tous les algorithmes au travers d'expérimentations menées sur des jeux de données synthétiques et réelles.

Le reste du chapitre est organisé de la manière suivante. Dans la Section 4.2, nous formalisons les deux variantes du problème. La Section 4.3 présente les travaux connexes. Dans la Section 4.4, nous décrivons notre nouvel algorithme, *F-SAC*. Les évaluations expérimentales sont présentées dans la Section 4.5 et discutées dans la Section 4.6. Enfin, la Section 4.7 conclut le chapitre.

4.2 Problématique

Partant d'un ensemble de points pondérés géolocalisés sur une carte, le problème du Regroupement Spatial Agglomératif (*SAC*) consiste à trouver une partition des points telle que les ensembles pondérés de cette partition ne se chevauchent pas sur le plan de visualisation. Dans ce chapitre, nous considérons deux variantes de ce problème.

Dans la première, *SAC en ligne*, le but est de trouver une partition des points pour un niveau de zoom donné de la carte. Dans la seconde, *SAC hors ligne*, le but est de trouver un ensemble de partitions pour n'importe quel niveau de zoom de la carte. L'avantage de la seconde version est que, les partitions étant pré-calculées, changer le niveau de zoom ne nécessite pas de nouveaux calculs favorisant ainsi la réactivité de la visualisation dans le cas de grands ensembles de données. Néanmoins, son principal inconvénient est que le calcul initial peut être très long et inutile pour une visualisation dans laquelle l'utilisateur ne peut pas changer de manière interactive le niveau de zoom. Dans cette section, nous formalisons les deux variantes du problème SAC.

4.2.1 Problème *SAC en ligne*

Soit M la carte, X un ensemble de points, p: $X \to \mathbb{R}^2$ une fonction associant chaque point de X à une position sur la carte et r: $X \to \mathbb{R}$ une fonction associant chaque point de X à leur rayon (qui peut dépendre des poids des points ou peut être uniforme).

Un *dessin* $\mathscr{L}(M, X, p, r)$ est une visualisation contenant la carte M en arrière-plan et les points de X en premier-plan. Chaque point x de X est positionné à $p(x)$ et est représenté par un cercle de rayon $r(x)$. Un dessin $\mathscr{L}$ est dit *sans chevauchements* s'il n'existe aucun chevauchement entre toute les paires de points de X.

Un *regroupement spatial* de X est une partition C des points de X. Le rayon r d'un cluster c de C est égal à la racine carrée de la somme des rayons au carré de ses points afin que son aire soit égale à la somme des aires de ses points :

$$r(c) = \sqrt{\sum_{x_i \in c} r(x_i)^2}$$

La position p d'un cluster c de C est égale au barycentre des positions de ses points :

$$p(c) = \sum_{p(x_i)|x_i \in c} p(x_i)/|c|$$

Définition 4.1 (Problème *SAC en ligne*)**.** Soient une carte M, un ensemble de points X, une fonction de rayon r et une fonction de position p, le problème de *SAC en ligne* consiste à trouver une partition C telle que $\mathscr{L}(M, C, p, r)$ est sans chevauchements et que la quantité de détails affichés est maximale.

La difficulté dans la conception d'un algorithme pour résoudre le problème SAC consiste à fournir la partition qui maximise la quantité de détails affichés. Sans cette contrainte, une solution simple dans laquelle $C = \{X\}$ serait une solution. Cependant, il est évident que la carte résultante ne serait pas optimale pour une tâche d'analyse visuelle. Décrire formellement ce qu'est une carte maximisant la quantité de détails affichés n'est pas trivial. Par exemple, nous pouvons maximiser le nombre de points/clusters affichés pour éviter les groupes inutiles de points. Nous pouvons également harmoniser les tailles des groupes pour éviter les grands groupes représentant de nombreux points. Une discussion sur les critères d'évaluation de ce point est proposée dans la Section 4.5.4.

4.2.2 Problème *SAC hors ligne*

Lorsque l'on traite de grands ensembles de données et que l'on permet à l'utilisateur de modifier le niveau de zoom de manière interactive, une solution consiste à pré-calculer les partitions pour chaque niveau de zoom avant d'afficher la carte. Dans ce cas, même si l'étape de pré-calcul est coûteuse en temps, il n'y a plus besoin d'effectuer de calculs supplémentaires lorsque l'utilisateur change de niveau de zoom, préservant ainsi davantage l'interactivité de la visualisation.

Soit $\mathscr{M}$ un ensemble ordonné contenant des fonds de cartes encodant les mêmes caractéristiques géographiques à différents niveaux de zoom.

Définition 4.2 (Problème *SAC hors ligne*)**.** Soient un ensemble de cartes $\mathscr{M}$, un ensemble de points X, une fonction de rayon r et une fonction de position p, le problème de *SAC hors ligne* consiste à trouver une liste ordonnée de partitions $\mathscr{C}$ telle que $\mathscr{L}(\mathscr{M}_l, \mathscr{C}_l, p, r)$ est sans chevauchements pour toute paire $(\mathscr{M}_l, \mathscr{C}_l)$ de $(\mathscr{M}, \mathscr{C})$ et que la quantité de

détails affichés est maximale pour tout l. Ici, l'indice l correspond au l-ième niveau de zoom.

Remarque. Ce problème peut être vu comme un problème de regroupement hiérarchique : le but est de fournir un arbre de clusters dans lequel chaque couche l contient la partition C_l pour le l-ième niveau de zoom. La première couche du dendrogramme contient un seul nœud (la racine) qui correspond à l'ensemble $\mathscr{C}_1 = \{X\}$.

4.3 État de l'art

Plusieurs algorithmes ont été conçus pour résoudre le problème SAC. À notre connaissance, Scheepens et al. (2014) ont fait la première proposition. Leur algorithme, que nous appelons par la suite *O-SAC* pour *Original SAC*, est présenté dans la sous-section 4.3.1. Plus tard, Castermans et al. (2019) ont proposé deux algorithmes, *QUAD+* et *QUAD+BIG*, décrits respectivement dans les sous-sections 4.3.2 et 4.3.3.

Tous les algorithmes sont basés sur une approche agglomérative : ils commencent par un ensemble de clusters dans lequel chaque cluster contient exactement un point de l'ensemble des données initiales ($C \leftarrow \{\{p\} \mid p \in X\}$) puis fusionnent récursivement les clusters jusqu'à obtenir un regroupement sans chevauchements.

4.3.1 O-SAC

O-SAC est conçu pour résoudre le problème 4.1 (*SAC en ligne*). Dans cette sous-section, nous présentons d'abord l'algorithme, ensuite nous fournissons une extension pour traiter le problème 4.2 (*SAC hors ligne*).

O-SAC *en ligne*

La sélection des meilleurs candidats à la fusion dans l'algorithme *O-SAC* est basée sur un facteur de chevauchement, ω, qui calcule, pour deux clusters c_i et c_j, la différence entre la somme des rayons et la distance euclidienne :

$$\omega(c_i, c_j) = r(c_i) + r(c_j) - ||p(c_i) - p(c_j)||$$

Lorsque la valeur est supérieure ou égale à 0 les clusters se chevauchent sinon ils ne se chevauchent pas.

Cette métrique est calculée pour chaque paire de clusters et est utilisée pour identifier, pour chaque cluster c_i, son « plus proche voisin », *i.e.* le cluster nnc_i qui se chevauche le plus avec c_i :

$$nnc_i \leftarrow c_j \mid \forall c \in C, \omega(c_i, c_j) \geq \omega(c_i, c)$$

Ensuite, en ordonnant les paires de clusters les plus proches, *i.e.* voisins *par rapport à* ω, l'algorithme sélectionne la paire la plus « chevauchante » (c_i, c) à fusionner. C est finalement mis à jour avec le nouveau cluster :

$$C \leftarrow C \setminus \{c_i, c_j\}$$
$$C \leftarrow C \cup \{\{c_i, c_j\}\}$$

Ce processus est répété jusqu'à ce qu'il n'y ait plus de chevauchements dans le dessin du regroupement.

O-SAC *hors ligne*

L'extension de l'algorithme *O-SAC* pour traiter le problème 4.2 (*SAC hors ligne*) est simple. Il suffit de calculer le regroupement du niveau de zoom le plus élevé $maxl$ de la liste de cartes $\mathscr{M}$, puis, pour chaque niveau de zoom l dans un ordre décroissant, de calculer le regroupement du niveau l à partir des clusters du niveau $l+1$ (cf. Algorithme 1).

Algorithme 1 O-SAC *hors ligne*

Entrée: $\mathscr{M}$, X, p, r
  $C \leftarrow \{\{p\} \mid p \in X\}$
  $\mathscr{C}_{maxl} \leftarrow$ O-SAC$(\mathscr{M}_{maxl}, C, p, r)$
  for l from $maxl - 1$ to 1 **do**
    $\mathscr{C}_l \leftarrow$ O-SAC$(\mathscr{M}_l, \mathscr{C}_{l+1}, p, r)$
  end for
  return $\mathscr{C}$

4.3.2 QUAD+

QUAD+ (Castermans et al., 2019) est conçu à l'origine pour résoudre le problème 4.2 (*SAC hors ligne*). Dans cette sous-section, nous présentons d'abord l'algorithme, puis nous fournissons une extension pour traiter le problème 4.1 (*SAC en ligne*).

Remarque. Dans leur article, Castermans et al. (2019) définissent deux autres algorithmes, un naïf et un plus élaboré, *QUAD*. Cependant, ils fournissent des résultats similaires à *QUAD+* tout en étant plus lents. Nous ne les considérons donc pas dans cette étude.

QUAD+ *hors ligne*

Comme mentionné dans la remarque de la sous-section 4.2.2, le problème *SAC hors ligne* peut être considéré comme un problème de regroupement hiérarchique : le but est de créer un arbre de partition des clusters. Ensuite, lorsqu'un utilisateur lance ou manipule la visualisation, les clusters affichés sont ceux qui se trouvent sur la coupe la plus détaillée de l'arbre de sorte qu'il n'y ait pas de chevauchements. Un avantage de cette approche est que nous n'avons pas besoin de connaître le nombre de niveaux de zoom de la carte puisque la racine de l'arbre est un cluster contenant l'ensemble des données.

L'algorithme *QUAD+* est basé sur ce principe. Il dispose d'abord les clusters initiaux, *i.e.* les clusters ne contenant qu'un seul point, à leurs positions respectives avec un rayon de 0. Ensuite, il augmente le rayon des clusters, fusionnant ceux qui se chevauchent, jusqu'à ce qu'il ne reste qu'un seul cluster contenant tous les points. La vitesse de

croissance des rayons dépend du poids des points. Par conséquent, nous obtenons l'arbre de partition qui est balayé de haut en bas pour trouver la meilleure coupe, *i.e*. celle dans laquelle l'ensemble des clusters ne se chevauchent pas mais dont les descendants direct se chevauchent.

Pour améliorer les performances de l'algorithme, les auteurs utilisent un arbre quaternaire (*QuadTree*) pour stocker les clusters. Ils utilisent également une file de priorité (*priority queue*) qui contient les paires de clusters fusionnés et le moment de leur collision, *i.e*. le moment où ils se chevauchent et doivent être fusionnés au cours du processus de croissance.

QUAD+ *en ligne*

Comme l'algorithme ne considère aucun niveau de zoom, nous ne pouvons pas arrêter le processus de croissance à un niveau de zoom donné pour une version *en ligne* de l'algorithme. Par conséquent, nous devons construire l'arbre entier et ensuite récupérer la coupe qui correspond le mieux au niveau de zoom donné.

4.3.3 QUAD+BIG

Tout comme *QUAD+*, *QUAD+BIG* est conçu pour traiter le problème 4.2 (*SAC hors ligne*). Dans cette sous-section, nous présentons d'abord l'algorithme puis nous fournissons une extension au problème 4.1 (*SAC en ligne*).

QUAD+BIG *hors ligne*

Les performances en temps de l'algorithme *QUAD+* chutent lorsqu'un grand cluster chevauche beaucoup de petits clusters. Pour traiter ce problème, les auteurs ont également proposé *QUAD+BIG*. Dans cette amélioration, les grands clusters sont traités indépendamment. Les plus grands clusters sont stables car leur fusion avec leurs petits voisins n'a pas un grand impact sur leur taille et leur position. Sur la base de cette observation, l'algorithme les fusionne avec leurs petits voisins sans mettre à jour les distances. Une fois les fusions effectuées, les distances sont mises à jour et les autres clusters sont traités. Les résultats obtenus sont assez similaires à ceux de *QUAD+* mais le processus est plus rapide.

QUAD+BIG *en ligne*

Nous pouvons utiliser la même approche que celle utilisée pour *QUAD+* (cf. Section 4.3.2).

4.4 F-SAC

Dans cette section, nous proposons une nouvelle approche, appelée *F-SAC* (pour *Fast SAC*), en versions *en ligne* et *hors ligne*. Tout d'abord, nous présentons les motivations de ces algorithmes, puis nous décrivons respectivement *F-SAC en ligne* et *F-SAC hors ligne*.

4.4.1 Motivations

En termes de temps de calcul, trois aspects des algorithmes SAC sont particulièrement coûteux. Le premier concerne les opérations effectuées pour fusionner les clusters qui se chevauchent. Le deuxième concerne plus spécifiquement les grands clusters chevauchant de nombreux petits clusters. Enfin, le dernier concerne la structure utilisée pour stocker la proximité entre les clusters. Dans cette section, nous décrivons ces différents aspects tout en présentant intuitivement notre proposition.

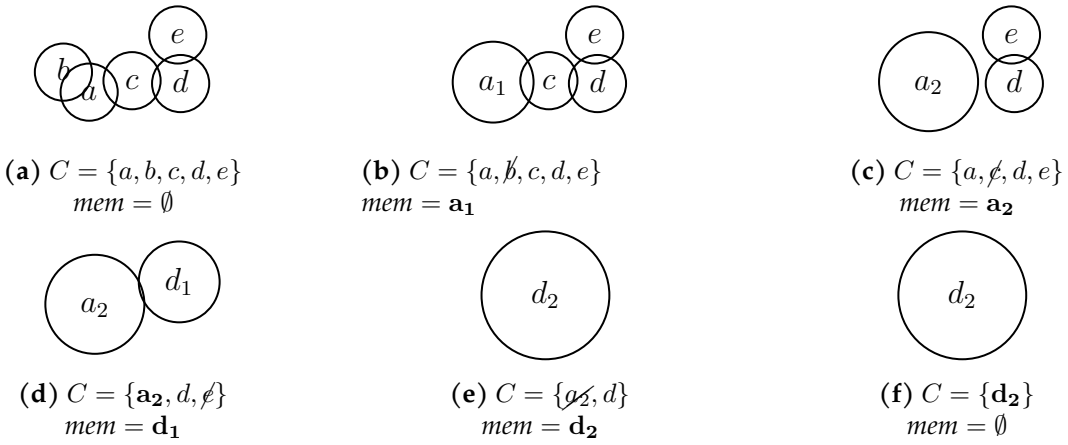
Réduire les opérations coûteuses

Comme nous l'avons vu précédemment, la fusion de clusters qui se chevauchent nécessite de parcourir, en effectuant un retour en arrière (*backtracking*), des structures de données pour trouver les clusters qui peuvent être fusionnés, puis de supprimer les clusters fusionnés et de mettre à jour ces structures. Ces opérations sont nécessairement coûteuses car elles nécessitent de nombreux parcours de la structure. Nous proposons une nouvelle stratégie qui consiste à éliminer progressivement dans la structure les clusters fusionnés, permettant ensuite de ne plus les considérer dans les étapes suivantes, mais surtout de repousser le plus tard possible les mises à jour de la structure, *i.e.* lorsque la dernière fusion possible pour un ensemble de clusters est effectuée.

Plus précisément, le principe est de parcourir la structure pour trouver les clusters qui peuvent être fusionnés. En général, lorsque deux clusters sont fusionnés, ils sont supprimés et un autre cluster est créé et stocké dans la structure. Le processus est répété jusqu'à ce qu'il n'y ait plus de chevauchements. Dans notre cas, lorsque nous fusionnons deux clusters, plutôt que de créer directement un nouveau cluster, nous supprimons un des clusters initiaux de la structure et nous agrandissons le cluster initial restant. Nous répétons ensuite cette étape jusqu'à ce qu'il n'y ait plus de chevauchements éliminant ainsi tous les clusters qui vont finir par être fusionnés.

Afin d'illustrer ce principe, considérons un ensemble de clusters $C = \{a, b, c, d, e\}$ tel que a chevauche b et c, c chevauche d et d chevauche e (cf. Figure 4.1(a)). Commençant par a, nous trouvons qu'il chevauche b et c. Puisque b est le plus proche de a, nous le supprimons de C et nous gardons en mémoire le fait que a et b ont été fusionnés en un cluster a_1 (cf. Figure 4.1(b)). La notation a_1 indique ici que le cluster fusionné est une extension du cluster a fusionné avec un autre cluster. Cette opération est répétée avec a_1 et c, donnant lieu au cluster a_2 (cf. Figure 4.1(c)). Comme a_2 ne peut plus être étendu, *i.e.* il ne chevauche aucun autre cluster, nous remplaçons dans C le cluster a par le nouveau cluster a_2 (cf. Figure 4.1(d)). Les clusters b et c ayant été supprimés de C, ils ne sont pas considérés. Si l'on considère le cluster d, il ne chevauche qu'avec e. Nous fusionnons d et e en d_1 et supprimons le cluster e de C (cf. Figure 4.1(d)). Maintenant en cherchant un autre cluster chevauchant avec d_1 dans C, nous trouvons a_2. Nous pouvons par conséquent fusionner d_1 avec a_2 en d_2 et supprimer a_2 de C (cf. Figure 4.1(e)). Comme il n'y a plus de chevauchement avec d_2, le cluster initial d peut être remplacé dans C par le cluster d_2 qui contient a, b, c, d et e (cf. Figure 4.1(f)).

Cette stratégie présente les principaux avantages suivants : (1) le nombre de clusters se chevauchant est réduit progressivement ; (2) tous les clusters qui ont fini de croître ont atteint leur taille maximale, *i.e.* une fois que l'ensemble C a été itéré, il n'y a plus de



(a) $C = \{a, b, c, d, e\}$ *mem* $= \emptyset$ **(b)** $C = \{a, \not{b}, c, d, e\}$ *mem* $= \mathbf{a_1}$ **(c)** $C = \{a, \not{e}, d, e\}$ *mem* $= \mathbf{a_2}$

(d) $C = \{\mathbf{a_2}, d, \not{e}\}$ *mem* $= \mathbf{d_1}$ **(e)** $C = \{\not{a}, d\}$ *mem* $= \mathbf{d_2}$ **(f)** $C = \{\mathbf{d_2}\}$ *mem* $= \emptyset$

Figure 4.1 – *Un exemple illustrant le comportement de* F-SAC

chevauchements dans C ; (3) le fait de reporter la mise à jour du cluster initial à la fin du processus minimise le nombre d'opérations sur la structure ; (4) tant qu'un cluster n'a pas atteint sa taille maximale, aucun cluster n'a besoin d'être ajouté à la structure de données évitant ainsi des opérations coûteuses.

Gestion des grands clusters

Comme indiqué dans Castermans et al. (2019), un grand cluster chevauchant de nombreux petits clusters est susceptible de fusionner avec tous ces derniers et ce sans significativement changer sa position. Dans ce cas, la création itérative d'un nouveau cluster avec le grand et un petit cluster est coûteuse et n'a que peu d'impact sur la taille et la position du grand cluster. Grâce au processus décrit dans la section précédente, nous fusionnons directement le grand cluster avec tous les petits sans mettre à jour la structure de données et cela permet également d'améliorer le temps de calcul.

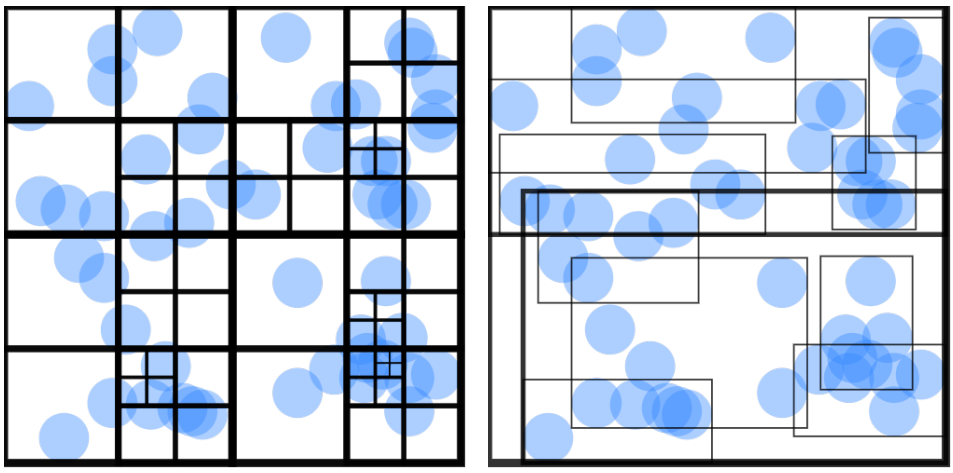
Calcul rapide des requêtes de proximité

Dans notre cas, la recherche de clusters se chevauchant est l'une des étapes les plus importantes. Pour détecter efficacement ces clusters, l'utilisation de structures d'indexation spatiale est très appropriée, comme le soulignent Castermans et al. (2019) avec un arbre quaternaire (*QuadTree*, Finkel et Bentley, 1974). Une telle structure fonctionne en ajoutant des points à une région et une fois qu'un nombre maximum d'éléments dans la région est atteint, la région est divisée en 4 quadrants égaux et les points sont répartis entre eux. Cette étape est répétée de manière récursive, produisant des régions de plus en plus petites.

Si les arbres quaternaires sont très efficaces pour stocker et récupérer des points, ils sont moins efficients lorsqu'il s'agit de manipuler des objets 2D comme les cercles. Dans ce cas, lorsqu'un cercle est ajouté à l'arbre, il est nécessaire de l'ajouter également à toutes les régions feuilles qu'il couvre. Par exemple, la Figure 4.2(a) illustre des régions représentées par des carrés noirs, la largeur de leurs arêtes représentant leur profondeur dans l'arbre. Plus le trait est épais, plus la région est élevée dans l'arbre. Dans cet exemple, une région ne peut contenir que 5 éléments au maximum. Nous

pouvons remarquer que certains cercles recouvrent plusieurs régions adjacentes : ils vont donc apparaître plusieurs fois dans la structure. Cette duplication présente plusieurs inconvénients : (1) des divisions inutiles peuvent se produire comme illustré en bas à droite de la figure où il y a une forte densité de points ; (2) lors de la suppression d'un objet, il est nécessaire de vérifier toutes les zones couvertes par celui-ci pour le supprimer en toute sécurité, un problème similaire se pose lors de l'insertion d'un objet ; (3) lors de la recherche des chevauchements dans l'arbre, certains éléments peuvent être trouvés plusieurs fois et doivent être dédoublés ; (4) enfin, le stockage d'une plus grande quantité d'éléments a nécessairement des conséquences tant sur la mémoire utilisée que sur les performances notamment dans le cas de grands clusters recouvrant de nombreuses régions.

Plusieurs autres limitations de cette structure et diverses améliorations ont été proposées et le lecteur intéressé peut se référer à Har-peled (2011).



(**a**) *Arbre quaternaire* (QuadTree) (**b**) *R-arbre* (R-Tree)

Figure 4.2 – *Aperçu de deux algorithmes d'indexation spatiale. Chaque élément est un cercle. Les rectangles noirs représentent des régions et l'épaisseur du trait représente leur profondeur dans l'arbre.*

Pour surmonter les limites des arbres quaternaires, nous proposons d'utiliser une structure de R-arbres (*R-Tree*, Manolopoulos et al., 2006). Particulièrement adaptés pour stocker des objets multidimensionnels tels que les cercles, les R-arbres fonctionnent de la manière suivante. Les objets sont placés dans des régions qui sont divisées lorsqu'un nombre maximal d'objets est atteint. Cependant, contrairement aux arbres quaternaires, les régions ne divisent pas strictement l'espace mais permettent un certain chevauchement. En effet, la taille d'une région est définie par la boîte englobante minimale de ses éléments (*minimum bounding box*), comme l'illustre la Figure 4.2(b) où chaque région peut contenir jusqu'à 5 éléments. Cependant, comme les régions peuvent se chevaucher, il est important d'assurer une distribution optimale des régions dans l'espace. Alors que les insertions en masse permettent des stratégies de fractionnement optimales, les insertions d'objets individuels peuvent facilement entraîner une dégradation de la distribution des régions dans l'espace, ce qui se traduit par des chevauchements importants de régions et donc par des performances plus faibles. Heureusement, plusieurs techniques (Arge et al., 2008 ; Beckmann et al., 1990 ;

Kamel & Faloutsos, 1994) ont été proposées pour résoudre ce problème.

4.4.2 Fast SAC *en ligne*

Sur la base des motivations de la section précédente, nous proposons un nouvel algorithme appelé *F-SAC* (cf. Algorithme 2). Notre proposition fonctionne en sélectionnant un cluster c_i dans l'ensemble des clusters C, que nous appelons c (cf. lignes 2-3). L'objectif est de supprimer tous les chevauchements possibles avec ce cluster c. Pour cela, nous recherchons tous les clusters chevauchants avec c qui ne sont pas c_i (ligne 5) et les trions suivant ω [1], par ordre décroissant de chevauchement. Nous appelons cet ensemble ordonné T ou *candidats*. Pour chaque *candidat* c_j de T, si c_j et c se chevauchent, nous fusionnons c et c_j en c et nous supprimons c_j de C. Cela se produit dans la Figure 4.1(a)(b)(c), où *mem* représente c. Il est avantageux de fusionner tous les clusters qui se chevauchent pour les grands clusters qui vont probablement peu se déplacer. Cependant, avec des clusters plus petits, il est possible que c ne chevauche plus avec certains des *candidats* de T car il s'est déplacé ou que son rayon a changé. C'est pourquoi nous vérifions s'il chevauche toujours avec c_j avant de le fusionner (ligne 7).

Algorithme 2 F-SAC

Entrée: M, X, p, r

1: $C \leftarrow \{\{c\} \mid c \in X\}$
2: **for** $c_i \in C$ **do**
3: $\quad c \leftarrow c_i$
4: $\quad$ **while** c has overlap **do**
5: $\quad\quad T \leftarrow \{c_j \mid c_j \in C, c_i \neq c_j \text{ and } \omega(c, c_j) > 0\}$
6: $\quad\quad$ **for** $c_j \in T$ in descending order of ω **do**
7: $\quad\quad\quad$ **if** $\omega(c, c_j) > 0$ **then**
8: $\quad\quad\quad\quad C \leftarrow C \setminus \{c_j\}$
9: $\quad\quad\quad\quad c \leftarrow c \cup c_j$
10: $\quad\quad\quad$ **end if**
11: $\quad\quad$ **end for**
12: $\quad$ **end while**
13: $\quad$ **if** $c \neq c_i$ **then** $C \leftarrow C \setminus \{c_i\} \cup \{c\}$
14: **end for**
15: **return** C

Puisque c change, il peut chevaucher d'autres clusters qui n'ont pas été collectés précédemment dans T. Si c'est le cas, l'opération est répétée (ligne 4). Un exemple de l'utilisation de cette boucle est illustré Figure 4.1(d), où d et e sont fusionnés en d_1, qui par conséquent chevauche a_2.

Une fois que c a résolu tous ses chevauchements, nous nous assurons qu'il a changé en le comparant à c_i. Si $c \neq c_i$, nous enlevons c_i de l'ensemble des clusters C et ajoutons c au même ensemble C (ligne 13). Ceci est fait afin d'éviter de supprimer et d'ajouter le même élément dans le cas où c n'a pas changé et est égal à c_i. Dans la Figure 4.1, cela

1. la différence entre la somme des rayons et la distance euclidienne, cf. sous-section 4.3.1

se produit lorsque les variables dans *mem* sont réajoutées à C, Figure 4.1(d) pour a_2 et (f) pour d_2.

Une fois que nous avons itéré sur tous les éléments de la partition C, cet ensemble n'a plus de clusters qui se chevauchent. Pour des raisons de performance, nous utilisons un R-arbre en plus de l'ensemble de clusters C pour améliorer la recherche de l'ensemble de candidats T. Bien entendu, les suppressions ou ajouts à C sont également pris en compte dans le R-arbre.

Complexité

La complexité moyenne de l'ajout, de la suppression ou de la recherche d'objets dans un R-arbre est de $O(\log n)$. Il peut y avoir au maximum $|X|-1$ clusters supplémentaires créés. Avec la pire configuration initiale [2], chaque fois que deux clusters sont fusionnés, le cluster résultant ne chevauche qu'un seul cluster. Dans ce cas, la complexité finale est de $O(|X| \log |X|)$.

La complexité dans le pire des cas de la recherche, de l'ajout ou de la suppression dans un R-arbre est de $O(n)$. Même s'il est très peu probable qu'un jeu de données entier produise le pire cas pour chacune de ces opérations, la complexité du pire des cas de notre algorithme est de $O(|X|^2)$.

4.4.3 F-SAC *hors ligne*

Pour résoudre la problématique de *SAC hors ligne* présentée dans la sous-section 4.2.2, nous proposons une version adaptée de *F-SAC* appelée « Offline *F-SAC* ».

Le processus est le même que celui présenté pour *O-SAC en ligne* (cf. Algorithme 1). Nous appliquons d'abord *F-SAC* au niveau de zoom le plus élevé, *i.e.* celui dans lequel les points sont les plus éloignés les uns des autres. Puis nous dézoomons progressivement en réutilisant la partition créée à partir du niveau de zoom précédent et en résolvant les nouveaux chevauchements créés.

4.5 Expérimentations

Dans cette section, nous présentons les résultats de nos expérimentations sur les performances des algorithmes décrits précédemment. Nous évaluons l'efficacité et le temps d'exécution pour différents jeux de données synthétiques et réelles. Avant de discuter des résultats, nous décrivons d'abord notre dispositif expérimental et les jeux de données. Ensuite, nous analysons les résultats d'un point de vue qualitatif et quantitatif. L'analyse quantitative est basée sur les métriques de qualité et le temps d'exécution. L'ensemble du code, les jeux de données et les résultats complets des expériences sont disponibles dans *Observable* [3].

2. un exemple de cette configuration est illustré dans https://observablehq.com/d/93cb90ee7e0a21fb
3. https://observablehq.com/@stardisblue/sac-overview

4.5.1 Dispositif expérimental

De manière à effectuer des expérimentations dans les mêmes conditions, tous les algorithmes ont été implémentés en JavaScript. *QUAD+* et *QUAD+BIG* (Castermans et al., 2019) sont refactorés en JavaScript [4] à partir de l'implémentation des auteurs [5]. *O-SAC* est implémenté à partir de l'algorithme décrit dans l'article Scheepens et al. (2014). Les auteurs proposent d'améliorer l'approche originale en utilisant une structure d'arbre K-D (*KD-Tree*). Plutôt que d'utiliser cette structure, nous avons développé une approche appelée *IO-SAC* (*Improved O-SAC*) qui utilise une structure de R-arbre (Manolopoulos et al., 2006) particulièrement efficace pour accélérer la détection des chevauchements. *O-SAC* et *IO-SAC* ont bien sûr été étendus pour gérer la version *hors ligne* (cf. Section 4.3.1). *O-SAC* et *IO-SAC* donnant des cartes similaires, nous rapportons les résultats des métriques de qualité obtenus soit par l'un soit par l'autre.

Toutes les expérimentations sont réalisées sur un PC Ryzen 5 3600X avec 16 gigaoctets de RAM, sous Microsoft Windows 10 et Google Chrome v91.0. Les exécutions des algorithmes sont effectuées séparément avec un démarrage à froid pour éviter les effets de mémoire. Enfin, les jeux de données réelles sont cartographiés en utilisant la projection web-mercator (EPSG :3857) [6]. Nous utilisons également Leaflet [7] pour visualiser les jeux de données réelles et leurs clusters.

4.5.2 Jeux de données

Pour la comparaison, nous utilisons des ensembles de données réelles et synthétiques.

En ce qui concerne les jeux de données réelles, les quatre premiers jeux sont sélectionnés à partir de Castermans et al. (2019). **Trove** est une collection d'environ 60 millions de notices bibliographiques publiées dans près de 8 000 lieux différents. **Glottolog** est une collection de données bibliographiques sur un peu plus de 7 000 langues du monde. Les positions des langues correspondent aux emplacements géographiques représentatifs de ces langues. **RISSE** est une collection de notices bibliographiques provenant de la « Bibliographia logica » (Risse, 1979), un ouvrage de référence sur l'histoire de la logique. Il comporte environ 7 000 livres dans 447 lieux différents. **IRA** est une notice de bibliothèque sur l'Architecture de la Renaissance Italienne (notée IRA pour *Italian Renaissance Architecture*) extrait de WorldCat [8]. Il comporte plus de 10 000 livres, distribués dans 582 bibliothèques. A ces jeux de données, nous ajoutons **OCS Wrecks** [9], une collection de plus de 10 000 épaves et obstructions immergées identifiées dans les limites maritimes des États-Unis, et **Siprojuris** [10], un recueil des carrières des professeurs de droit en France entre 1804 et 1950 avec leur géolocalisation. La plupart des 427 villes sont en France.

4. https://github.com/stardisblue/ts-growing-glyphs, plus précisément en TypeScript, une version typée de JavaScript.
5. https://github.com/Caster/growing-glyphs
6. https://github.com/d3/d3-geo#geoMercator
7. https://leafletjs.com/
8. https://www.worldcat.org/
9. https://www.fisheries.noaa.gov/inport/item/39988
10. http://siprojuris.symogih.org

Le Tableau 4.1 donne quelques informations sur les ensembles de données. **Chevauchement/point** représente le nombre moyen de chevauchements par point et **Chevauchement%** est le pourcentage de chevauchements par rapport au nombre total de chevauchements possibles. IRA, RISSE et Siprojuris sont des jeux de données relativement petits avec des pourcentages de chevauchements différents. Bien que Glottolog et Trove aient à peu près la même taille, le **Chevauchement%** de Trove est pratiquement le double de celui de Glottolog. Enfin, OCS Wrecks est le plus grand jeu de données avec le **Chevauchement/point** le plus élevé.

Table 4.1 – *Description des jeux de données réelles.* ***Chevauchement/point*** *représente le nombre moyen de chevauchements par point.* ***Chevauchement%*** *est le pourcentage de chevauchements par rapport au nombre total de chevauchements possibles.*

	Points	Chevauchement/point	Chevauchement%
Trove	7 758	26,868	0,346
Glottolog	7 103	10,794	0,152
IRA	582	4,478	0,771
RISSE	477	2,107	0,473
OCS Wrecks	11 493	214,622	1,868
Siprojuris	427	7,04	1,653

Comme nous voulons également considérer des jeux de données de différentes tailles et de différentes distributions, nous avons généré des jeux de données synthétiques. Nous considérons les distributions suivantes : uniforme, normale, échantillonnage en disques de Poisson (*Poisson disk sampling*, Cook, 1986) et bruit de Perlin (*Perlin noise*, Perlin, 1985). Pour chaque distribution, en commencant par 50 jusqu'à 25 600 points avec un pas en puissance de 2, nous générons 100 échantillons à chaque pas. Au total, 4 000 jeux de données synthétiques sont générés. Les jeux de données sont créés de manière à ce que la densité reste la même pour un nombre variable de points, suivant l'équation suivante :

$$\frac{\text{area}(bounding\ box)}{\sum_{p \in X} \text{area}(p)}$$

Enfin, pour reproduire fidèlement les scénarios *hors ligne*, 19 niveaux de zoom sont définis. Chaque niveau est plus grand que le précédent, partant d'un carré de 256 pixels qui est généralement la valeur par défaut utilisée par la plupart des cartes Web.

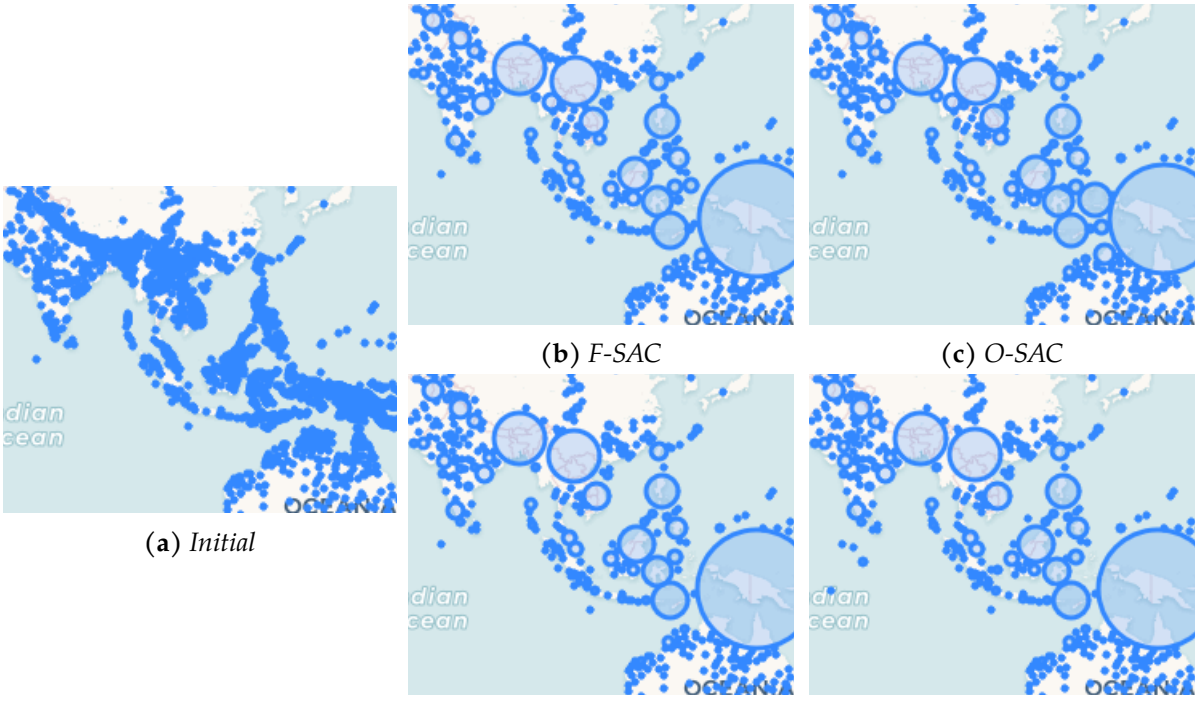
4.5.3 Analyse qualitative

Dans cette section, nous présentons quelques résultats qualitatifs, d'abord sur un jeu de données réelles puis sur un ensemble de jeux de données synthétiques. L'objectif n'est pas de présenter toutes les expériences mais de mettre en évidence les différences qui existent dans les clusters obtenus par les algorithmes. Le lecteur intéressé par l'ensemble des expériences menées sur les jeux de données réelles et synthétiques peut

se référer à l'*Observable* [11].

Jeux de données réelles

La Figure 4.3 illustre une partie des résultats obtenus après l'application des différents algorithmes pour le jeu de données Glottolog. Nous pouvons constater sur la Figure 4.3(a) qu'il y a deux zones très denses. La première est proche de l'Asie du Sud et la seconde de l'Indonésie et de la Papouasie-Nouvelle-Guinée. Ceci est confirmé par les Figures 4.3(b-e) qui montrent que tous les algorithmes génèrent un grand cluster pour la zone de Papouasie-Nouvelle-Guinée ainsi que deux clusters de taille moyenne en Malaisie.



(**a**) *Initial* (**b**) *F-SAC* (**c**) *O-SAC* (**d**) *QUAD+* (**e**) *QUAD+BIG*

Figure 4.3 – *Aperçu d'une zone spécifique de la carte pour mettre en évidence les différences entre les algorithmes*

Nous pouvons voir au centre (à gauche du grand cluster) que *QUAD+* (cf. Figure 4.3(d)) et *QUAD+BIG* (cf. Figure 4.3(e)) génèrent des clusters similaires. En revanche, pour *O-SAC* (cf. Figure 4.3(c)), il est intéressant de constater que les résultats sont assez différents avec un plus grand nombre de petits clusters. Avoir une taille de cluster plus petite signifie que moins de points ont été regroupés et que les centroïdes des clusters sont plus proches de la position respective de leurs points. Cela pourrait montrer une représentation plus précise de la position initiale des points et une moindre distorsion du jeu de données initial. Nous reviendrons sur ce point lors de l'analyse quantitative, en Section 4.5.4. En ce qui concerne *F-SAC* (cf. Figure 4.3(b)), nous constatons qu'il présente également plus de petits clusters par rapport à *QUAD+* et *QUAD+BIG*.

11. https://github.com/stardisblue/fsac#results

Jeux de données synthétiques

Les Figures 4.4(a-d) représentent les jeux de données initiaux. Les résultats des algorithmes sont affichés en dessous, chaque ligne représentant les résultats d'un algorithme, *e.g.* les résultats de *F-SAC* sont présentés dans les Figures 4.4(e-h), etc.

La distribution uniforme (cf. Figure 4.4(a)) place les points de manière aléatoire dans l'espace. Les regroupements effectués par les algorithmes sont similaires. Les différences ne sont pas facilement visibles et il semble y en avoir peu. Nous pouvons observer dans le coin supérieur gauche qu'*O-SAC* (i) produit 3 clusters similaires de taille moyenne alors que *F-SAC* (e), *QUAD+* (m) et *QUAD+BIG* (q) semblent créer un grand cluster et deux plus petits.

La distribution normale (cf. Figure 4.4(b)) crée une zone dense au centre de la carte. Par conséquent, les algorithmes créent un grand cluster au centre. Il est intéressant de noter que *QUAD+* (n) et *QUAD+BIG* (r) présentent une zone plus ou moins vide autour du grand cluster. Au contraire, pour *F-SAC* (f) et *O-SAC* (j), nous pouvons voir que cette zone contient de nombreux clusters de taille moyenne. Dans cet exemple, *F-SAC* et *O-SAC* semblent avoir des résultats similaires et produire une disposition plus détaillée que *QUAD+* et *QUAD+BIG*.

La distribution en disque de Poisson (cf. Figure 4.4(c)) place les points de manière uniforme dans l'espace. Par conséquent, il n'y a pas beaucoup de chevauchements. Nous pouvons observer que *QUAD+* (o) et *QUAD+BIG* (s) ont tendance à créer de plus grands clusters que *F-SAC* (g) et *O-SAC* (k). La différence entre *F-SAC* et *O-SAC* est moins prononcée. En regardant attentivement, on peut voir que, parfois *F-SAC* crée des clusters plus grands que *O-SAC*, parfois c'est le cas de *O-SAC*. En conclusion, *F-SAC* et *O-SAC* donnent des résultats plus détaillés sur cet exemple de la distribution en disque de Poisson car ils ont tendance à agréger moins de points que *QUAD+* et *QUAD+BIG*. *F-SAC* et *O-SAC* ne donnent pas des résultats similaires mais le degré global de détails affichés semble être équivalent.

La distribution en bruit de Perlin (cf. Figure 4.4(d)) génère des zones de points à la fois clairsemées et denses. Encore une fois, *QUAD+* (p) et *QUAD+BIG* (t) donnent des résultats similaires avec des clusters plus grands que *F-SAC* (h) et *O-SAC* (l). Cependant, ce phénomène est moins marqué, comme on peut le constater en bas au centre-droit où *QUAD+* et *QUAD+BIG* créent trois grands clusters alors que *F-SAC* et *O-SAC* n'en créent qu'un seul beaucoup plus grand. Dans l'ensemble, *O-SAC* crée la carte la plus détaillée car les clusters sont généralement moins grands que ceux de *F-SAC*.

En conclusion, nous pouvons observer sur cet exemple que *QUAD+* et *QUAD+BIG* produisent des résultats très similaires et que *O-SAC* et *F-SAC* semblent plus performants pour la plupart des distributions (normale, disque de Poisson et bruit de Perlin). *O-SAC* semble être plus performant dans la plupart des cas mais *F-SAC* semble aussi efficace sur les distributions normales et disques de Poisson. La différence entre les algorithmes ne semble pas visuellement significative pour la distribution uniforme.

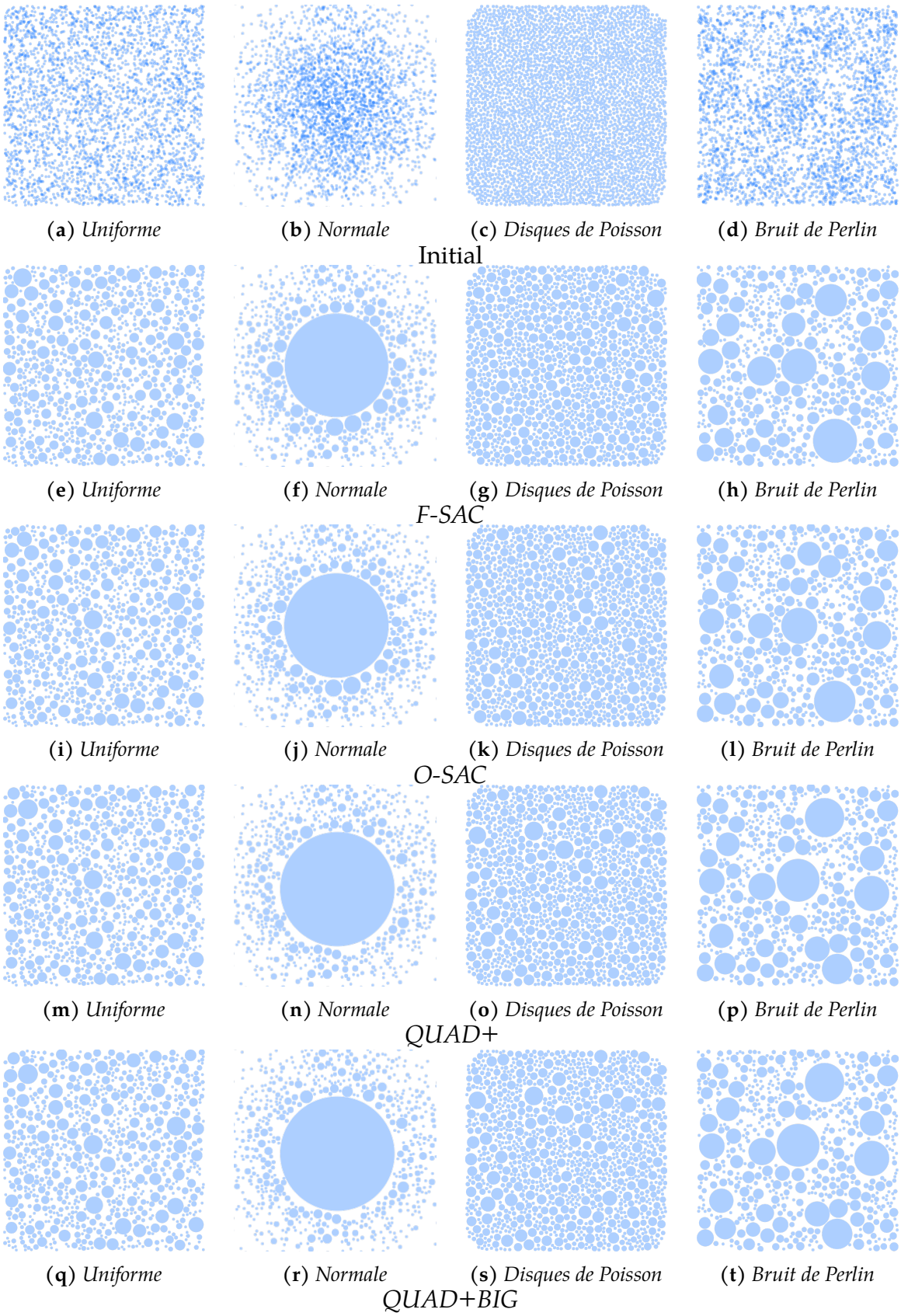


Figure 4.4 – *Un exemple de placement avec 3 200 points pour chaque modèle de distribution. Les points et les clusters sont représentés par des cercles bleus semi-transparents.*

4.5.4 Analyse quantitative

Dans cette section, nous comparons les différents algorithmes en termes de temps d'exécution et de qualité des résultats. Pour mesurer cette dernière, nous présentons trois métriques : le facteur de compression, la distance moyenne des points aux centres des clusters et la variance de la taille des clusters.

Temps d'exécution

Le temps d'exécution des algorithmes sur des jeux de données réelles est indiqué dans le Tableau 4.2 pour les versions *en ligne* et dans le Tableau 4.3 pour les versions *hors ligne*. Chaque expérience a été reproduite 30 fois et la médiane a été utilisée comme valeur représentative (la moyenne a obtenu des valeurs similaires à la médiane).

Table 4.2 – *Temps d'exécution : temps médian (ms) des algorithmes* en ligne *sur des jeux de données réelles*

	O-SAC	IO-SAC	QUAD+	QUAD+BIG	F-SAC
Trove	139 960	6 335	2 054	3 535	**15**
Glottolog	68 942	2 688	1 801	2 870	**14**
IRA	94	22	118	130	**< 1**
RISSE	20	5	100	104	**< 1**
OCS Wrecks	≈4 704 225	570 266	3 114	4 091	**19**
Siprojuris	119	44	89	95	**< 1**

Table 4.3 – *Temps d'exécution : temps médian (ms) des algorithmes* hors ligne *sur des jeux de données réelles*

	O-SAC	IO-SAC	QUAD+	QUAD+BIG	F-SAC
Trove	≈265 314	1 594	2 054	3 535	**730**
Glottolog	≈214 812	1 241	1 801	2 870	**667**
IRA	267	46	118	130	**26**
RISSE	182	42	100	104	**17**
OCS Wrecks	≈339 483	996	3 114	4 091	**801**
Siprojuris	145	36	89	95	**14**

Tout d'abord, nous pouvons remarquer que *F-SAC* surpasse les autres algorithmes sur n'importe quel jeu de données aussi bien pour les versions *en ligne* que *hors ligne*. La différence entre les deux versions est liée au fait que dans la version *hors ligne*, tous les niveaux de zoom possibles sont traités alors que la version *en ligne* ne considère qu'un seul niveau de zoom. En revanche, les temps d'exécution de *QUAD+* et *QUAD+BIG* sont similaires pour les versions *en ligne* et *hors ligne* car, comme précisé dans la Section 4.3.2, ils doivent générer l'arbre entier avant d'effectuer la coupe pour la version *en ligne*.

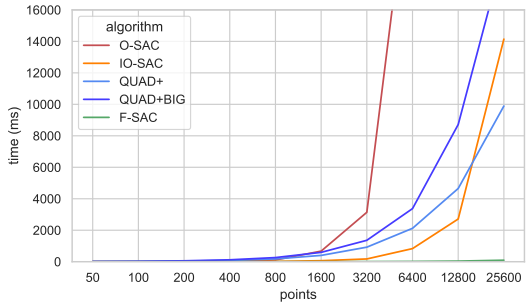
Sans surprise, *IO-SAC* est beaucoup plus rapide que *O-SAC* grâce à sa structure de R-arbre. Quant à *O-SAC*, les temps de réponse étaient vraiment très longs pour les grands ensembles de données (*i.e.* OCS Wrecks) par rapport aux autres algorithmes. Les valeurs approximatives rapportées ont été obtenues via 3 échantillons plutôt que 30.

Pour des jeux de données de taille similaire, tous les algorithmes prennent plus de temps avec les jeux de données possédant un plus grand nombre de chevauchements (*i.e.* Trove et Glottolog, cf. Tableau 4.1). Dans le Tableau 4.2, nous pouvons remarquer que *O-SAC* et *IO-SAC en ligne* prennent 2 à 3 fois plus de temps entre Glottolog et Trove, ce qui indique que ces algorithmes ont du mal lorsqu'il y a beaucoup de chevauchements. Comparé à *QUAD+* qui prend environ 10% de temps en plus et à *QUAD+BIG* qui prend environ 20% de temps en plus entre Glottolog et Trove. L'augmentation du temps d'exécution de *F-SAC* n'est que de 6%. En comparant le Tableau 4.2 et Tableau 4.3, on pourrait s'attendre à ce que les temps d'exécution *hors ligne* soient plus élevés que les temps d'exécution *en ligne* pour *O-SAC*, *IO-SAC* et *F-SAC*, puisque les versions *hors ligne* doivent prendre en compte tous les niveaux de zoom plutôt qu'un seul comme pour les versions *en ligne*. Nous pouvons noter que cela est toujours vrai avec *O-SAC* et *F-SAC*, mais pas toujours avec *IO-SAC*, pour lequel la version *hors ligne* est plus rapide pour les plus grands jeux de données. La différence entre les temps d'exécution *en ligne* et *hors ligne* pour OCS Wrecks est assez impressionnante, *IO-SAC en ligne* prend 570 266 ms (*i.e.* 9,5 min) alors que la version *hors ligne* ne prend que 996 ms (moins d'une seconde). Cela peut s'expliquer par le mode de fonctionnement des algorithmes *hors ligne* (cf. Section 4.3.1). Comme le processus pour chaque niveau de zoom réutilise le précédent, seule une petite quantité de chevauchements est introduite. Par conséquent, *IO-SAC*, qui est très sensible au nombre de chevauchements, donne de meilleurs résultats avec une approche hiérarchique (version *hors ligne*) qu'avec une approche linéaire (version *en ligne*).

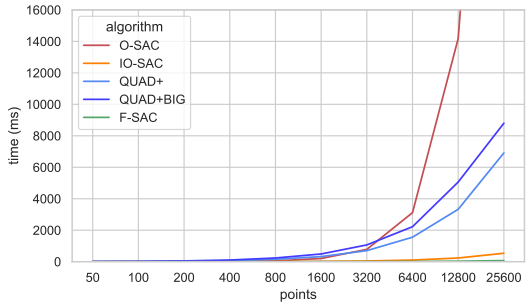
Les Figures 4.5 et 4.6 montre le temps d'exécution par rapport à la taille des jeux de données (*i.e.* nombre de points) des algorithmes *en ligne* et *hors ligne* sur les jeux de données synthétiques. L'échelle de temps est tronquée à 16 000 ms pour mieux distinguer *F-SAC* et *IO-SAC*. Comme 100 échantillons ont été générés pour chaque taille de jeu de données (cf. sous-section 4.5.2), la ligne tracée correspond à la durée moyenne de l'exécution.

Tout d'abord, dans la Figure 4.5, nous pouvons observer que, quel que soit le type de distribution, *F-SAC* surpasse toujours ses concurrents en termes de temps d'exécution (moins d'une seconde pour 25 600 points). Ceci confirme les résultats obtenus pour les données réelles où *F-SAC* offrait déjà de meilleurs temps d'exécution quel que soit les données. Il est intéressant de noter qu'à l'exception de la distribution normale (cf. Figure 4.5(a)) *IO-SAC* est également très performant.

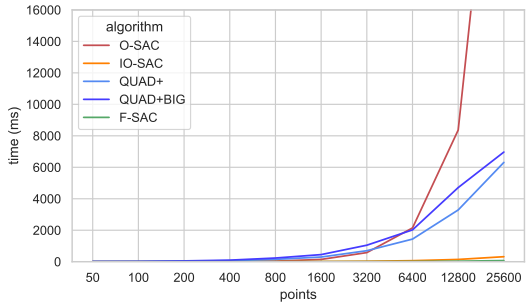
Les résultats pour la distribution bruit de Perlin (cf. Figure 4.5(b)), disque de Poisson (cf. Figure 4.5(c)) et uniforme (cf. Figure 4.5(d)) sont presque similaires : *F-SAC* et *IO-SAC* sont les plus rapides, puis viennent *QUAD+* et *QUAD+BIG* (*QUAD+* étant légèrement plus rapide que *QUAD+BIG*) et enfin *O-SAC*, qui est de loin le plus lent. Cependant, jusqu'à 3 200 points, *O-SAC* est plus rapide que *QUAD+* et



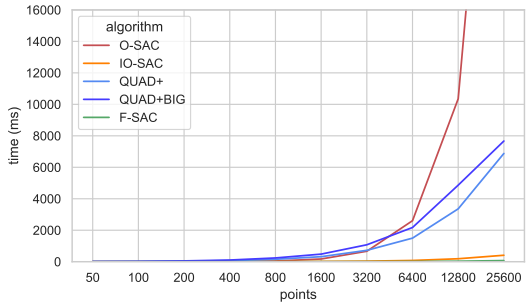
(**a**) *Normal*



(**b**) *Bruit de Perlin*

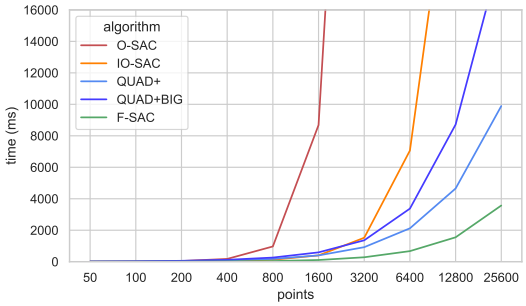


(**c**) *Disques de Poisson*

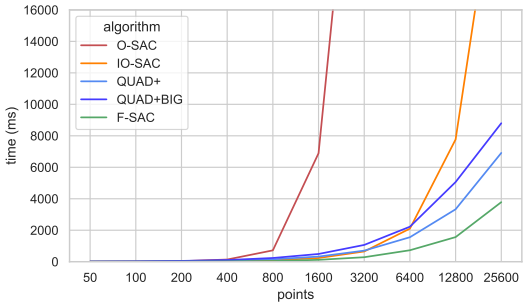


(**d**) *Uniforme*

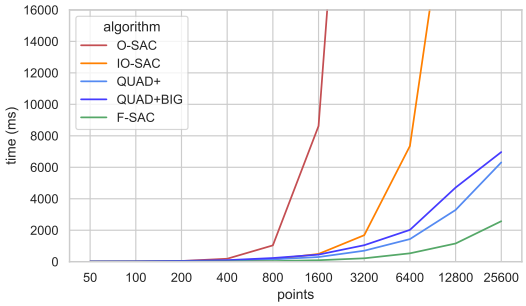
Figure 4.5 – *Temps d'exécution des algorithmes* SAC en ligne *sur des jeux de données synthétiques*



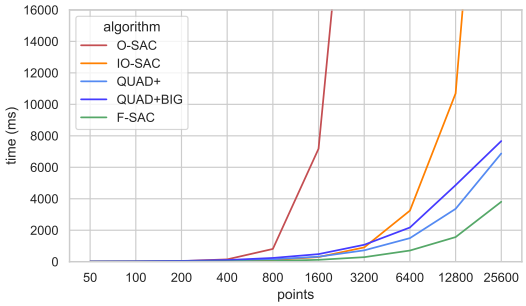
(**a**) *Normal*



(**b**) *Bruit de Perlin*



(**c**) *Disques de Poisson*



(**d**) *Uniforme*

Figure 4.6 – *Temps d'exécution des algorithmes* SAC hors ligne *sur des jeux de données synthétiques*

QUAD+BIG.

Le comportement est différent sur la distribution normale Figure 4.5(a). Même si *F-SAC* reste le plus rapide et *O-SAC* le plus lent, *IO-SAC*, *QUAD+* et *QUAD+BIG* ont des performances différentes : *IO-SAC* est relativement rapide jusqu'à 3 200 points mais son temps d'exécution augmente rapidement et devient supérieur à celui de *QUAD+* par la suite, *QUAD+* est toujours plus rapide que *QUAD+BIG* mais cette différence augmente plus qu'avec les autres distributions.

Les résultats des versions *hors ligne* (cf. Figure 4.6) montrent des comportements similaires pour toutes les distributions : *F-SAC* est le plus rapide, puis viennent *QUAD+* et *QUAD+BIG*, puis *IO-SAC* et enfin *O-SAC*. Nous pouvons noter que *IO-SAC* surpasse *QUAD+* et *QUAD+BIG* sur de petits ensembles de données (moins de 6 400 points) et que les résultats de *QUAD+* et *QUAD+BIG* sont identiques à ceux des versions *en ligne* (cf. Section 4.3.2).

Mesure de la qualité

Dans cette section, nous présentons d'abord les métriques de qualité utilisées pour comparer les algorithmes. Nous détaillons ensuite les résultats obtenus sur les jeux de données synthétiques.

Facteur de compression

En appliquant les algorithmes SAC, le nombre de clusters dans l'ensemble final C est toujours inférieur ou égal au nombre de points de X. L'objectif du facteur de compression est de déterminer le degré de changement introduit. L'intuition sous-jacente est de pénaliser les algorithmes qui proposeraient un seul cluster regroupant tous les points et de favoriser ceux qui proposent un nombre de clusters proche du nombre de points initiaux. En effet, dans ce dernier, le nombre de changements devient minimal. Le facteur de compression est obtenu par :

$$cf = 1 - \frac{|C|}{|X|}$$

Ainsi, plus le facteur est proche de 0, meilleur est le résultat. Par exemple, si le jeu de données initial ne présente aucun chevauchement, le nombre de clusters devrait correspondre au nombre de points ($cf = 0$). A l'inverse, une valeur proche de 1 indique qu'il y a eu beaucoup de changements.

Distance moyenne au centre

Le but de cette métrique est de déterminer dans quelle mesure la zone contenant les points d'un cluster est étendue. Pour cela, on mesure la distance entre le centroïde du cluster et ses différents points. L'intuition est de pénaliser les algorithmes qui ont tendance à regrouper des points situés à de grandes distances, *i.e.* la position représentée par le centroïde du cluster peut être très éloignée de la position réelle d'un point et donc conduire à un biais pour l'utilisateur. Nous utilisons donc la distance

euclidienne au carré pour pénaliser les grandes distances :

$$mdc = \frac{1}{|X|} \times \sum_{c \in C} \sum_{p \in c} ||x(c) - x(p)||^2$$

Bien entendu, plus la valeur est petite, plus l'algorithme est performant.

Variance de la taille des clusters
Nous proposons également de mesurer la variance des tailles des clusters. Une partition avec un grand cluster et beaucoup de petits a une variance plus élevée qu'une partition avec des tailles de cluster presque similaires. Cette mesure est plus subjective car, en fonction des visualisations attendues, certains utilisateurs peuvent être intéressés par de grands clusters avec de petits autour alors que d'autres peuvent être intéressées par des clusters de taille assez similaire. En fait, l'objectif de cette mesure est d'évaluer l'impact du regroupement par rapport au plongement initial. Par exemple, si la distribution est uniforme, nous nous attendons à avoir une variance faible, *i.e.* que tous les clusters soient à peu près de taille identique. En revanche, si les points sont placés selon une distribution normale, nous nous attendons à une forte variance : de grands clusters au centre et de très petits clusters autour. Étant très subjective, nous utilisons cette métrique principalement pour aider l'utilisateur, en fonction de ses choix, à avoir plus d'informations sur le comportement des algorithmes. La variance de la taille des clusters est donc obtenue comme suit :

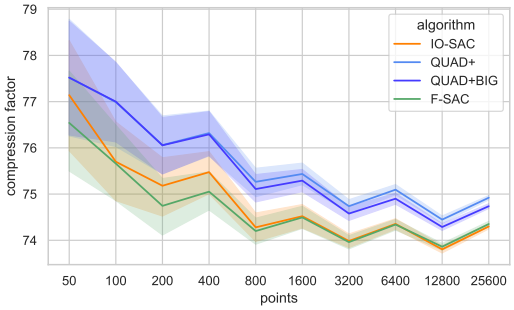
$$csv = \frac{1}{|C|} \sum_{c \in C} (|c| - \bar{|c|})^2$$

où $\bar{|c|}$ est la moyenne des tailles des clusters de C.

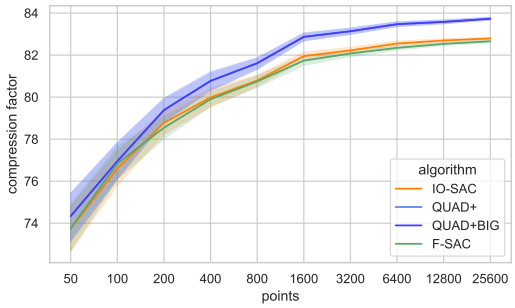
Resultats
La Figure 4.7 illustre les résultats obtenus sur les différentes distributions pour le facteur de compression. Nous pouvons observer que ce facteur augmente lorsque la taille du jeu de données augmente pour toutes les distributions sauf pour la distribution normale. Pour la distribution normale, bruit de Perlin et uniforme (cf. Figures 4.7(a)(b)(d)), *F-SAC* et *IO-SAC* ont un comportement similaire et surpassent *QUAD+* et *QUAD+BIG*. Pour une distribution en disques de Poisson (cf. Figure 4.7(c)), *F-SAC* et *IO-SAC* sont également plus performants que *QUAD+* et *QUAD+BIG* mais *F-SAC* obtient de moins bon résultats que *IO-SAC*.

La Figure 4.8 illustre les résultats obtenus sur les différentes distributions pour la distance moyenne au centre. Comme pour la métrique précédente, nous pouvons observer que *F-SAC* et *IO-SAC* obtiennent de meilleurs résultats que *QUAD+* et *QUAD+BIG*. Nous pouvons également voir que *F-SAC* et *IO-SAC* obtiennent des résultats similaires pour les distributions normales et uniformes mais que *IO-SAC* obtient de meilleurs résultats pour les distributions en bruit de Perlin et disques de Poisson.

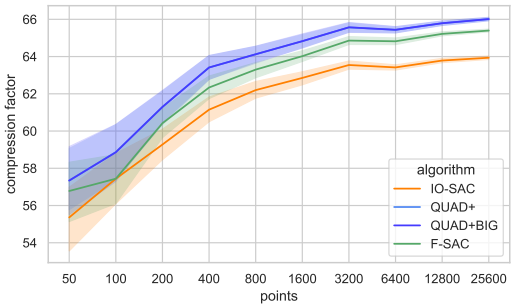
La Figure 4.9 illustre les résultats obtenus sur les différentes distributions pour la variance de la taille des clusters. Les résultats sont presque similaires à ceux obtenus sur la mesure précédente indiquant une forte corrélation entre les valeurs. Ainsi, lorsqu'un



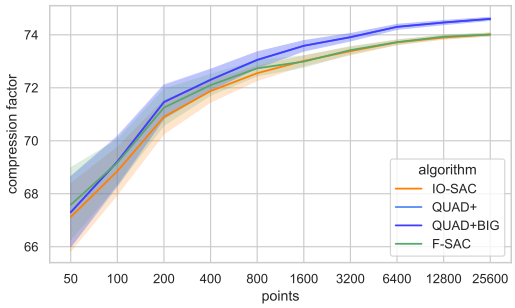
(**a**) *Normale*



(**b**) *Bruit de Perlin*

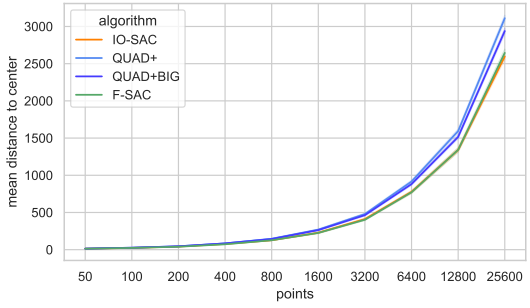


(**c**) *Disques de Poisson*

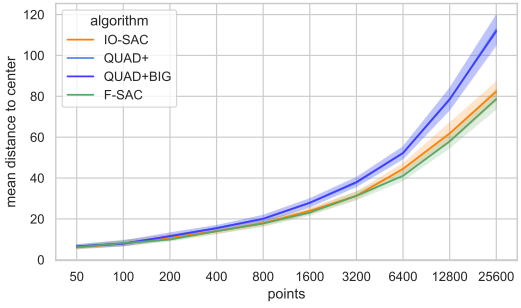


(**d**) *Uniforme*

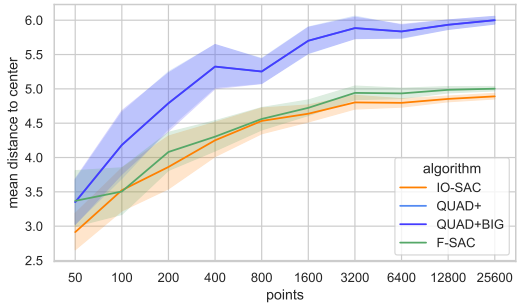
Figure 4.7 – *Facteur de compression pour les algorithmes* SAC en ligne *sur les jeux de données synthétiques. L'axe des y est différent pour chaque figure. La ligne représente la valeur moyenne et la zone l'intervalle de confiance à 95%.*



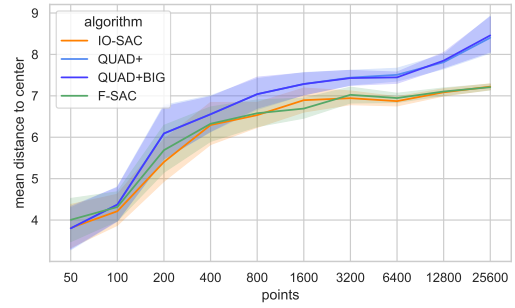
(**a**) *Normale*



(**b**) *Bruit de Perlin*



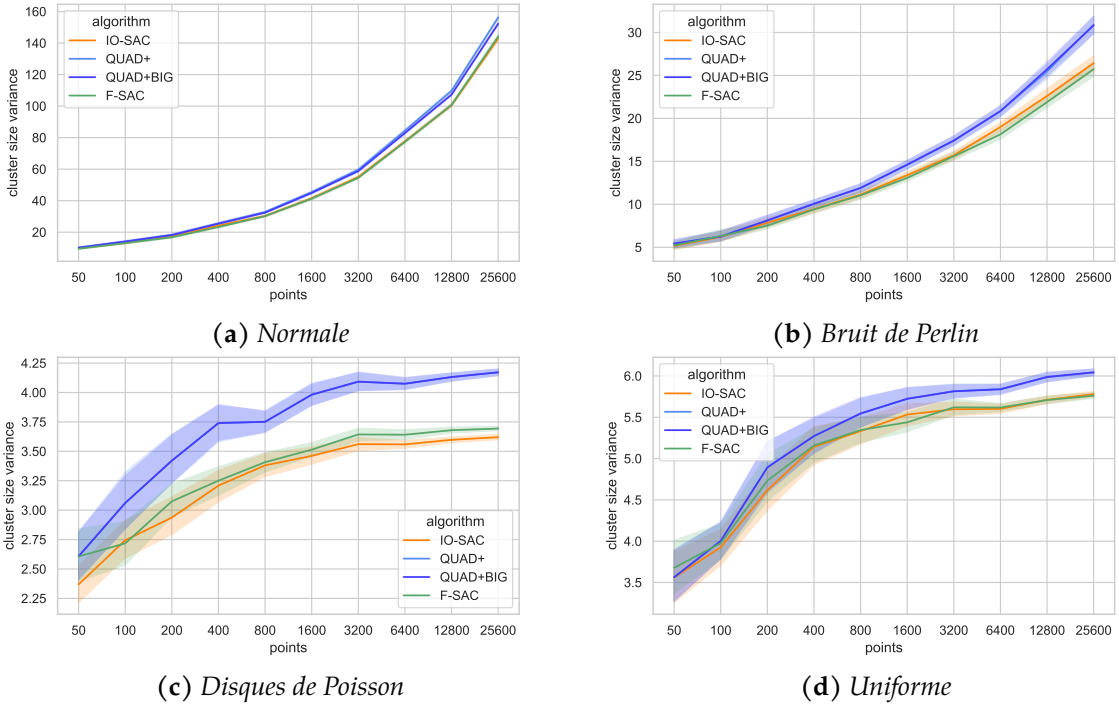
(**c**) *Disques de Poisson*



(**d**) *Uniforme*

Figure 4.8 – *Distance moyenne au centre pour les algorithmes* SAC en ligne *sur les jeux de données synthétiques. L'axe des y est différent pour chaque figure. La ligne représente la valeur moyenne et la zone l'intervalle de confiance à 95%.*

algorithme tend à regrouper des points dans un cluster loin de leur position initiale, il tend également à créer des clusters de taille variée.



(**a**) *Normale* (**b**) *Bruit de Perlin*

(**c**) *Disques de Poisson* (**d**) *Uniforme*

Figure 4.9 – *Variance de la taille des clusters pour les algorithmes* SAC en ligne *sur les jeux de données synthétiques. L'axe des y est différent pour chaque figure. La ligne représente la valeur moyenne et la zone l'intervalle de confiance à 95%.*

En conclusion, nos mesures suggèrent qu'*IO-SAC* fournit toujours les meilleurs résultats, que *F-SAC* fournit souvent des résultats similaires ou légèrement moins bons que *IO-SAC*, et que *QUAD+* et *QUAD+BIG* fournissent les moins bons résultats. Étant donné qu'*O-SAC* et *IO-SAC* fournissent les mêmes résultats, nous ne reportons que l'un d'entre eux sur les figures.

4.6 Discussion

Dans cette partie, nous revenons sur certains choix effectués lors des expérimentations ainsi que sur d'autres expérimentations réalisées afin de mieux comprendre par exemple le niveau d'impact des zooms ou bien pour évaluer l'impact d'implémentation avec d'autres langages.

Les expérimentations sur les algorithmes *hors ligne* ont été réalisées en utilisant 19 niveaux de zoom. Comme attendu, l'augmentation de ce nombre entraîne une augmentation des temps d'exécution de *O-SAC*, *IO-SAC* et *F-SAC*. Nous avons vu que *QUAD+* et *QUAD+BIG* résolvent tous les niveaux possibles (cf. sous-section 4.3.2), aussi leur temps d'exécution reste donc constant quelque soit le nombre de niveaux de zoom. De manière complémentaire, nous avons mené des expérimentations pour savoir à partir de quel niveau de zoom *QUAD+* et *QUAD+BIG* deviennent plus rapides. La Figure 4.10 illustre la variation du zoom sur un jeu de données généré de 1 600 points

(distribution en bruit de Perlin). Nous pouvons voir que *QUAD+* devient plus rapide que *F-SAC* et *IO-SAC* après 50 niveaux et *QUAD+BIG* après 90.

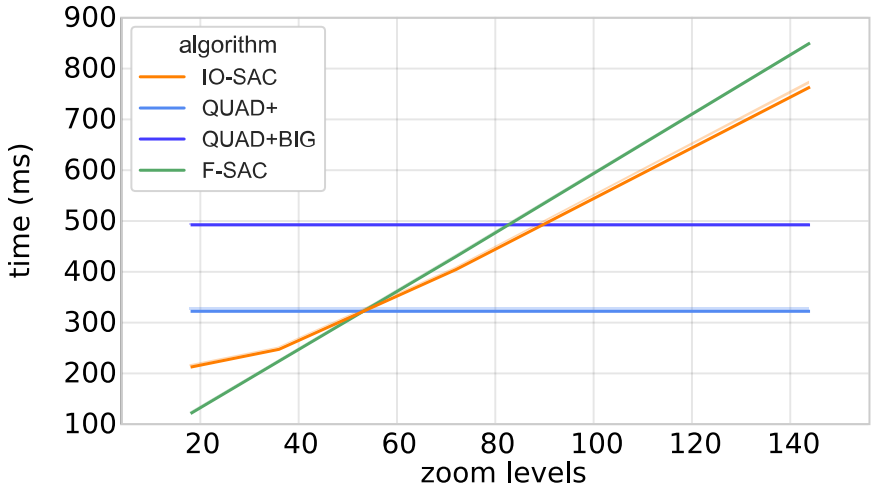


Figure 4.10 – *Temps d'exécution des algorithmes* hors ligne *en fonction du nombre de niveaux de zoom sur un jeux de données de distribution en bruit de Perlin avec 1 600 points*

Pour présenter les résultats de qualité des algorithmes *hors ligne*, nous avons choisi un niveau de zoom de 2. En effet, un niveau de zoom de 0 aura tendance à produire un seul cluster et un niveau de zoom élevé (*e.g.* 4) aura tendance à produire de nombreux petits clusters. Le niveau de zoom 2 a également été choisi car il permet à l'algorithme de regrouper des niveaux de zoom plus élevés et donc de voir l'effet de l'approche pas à pas pour les versions *hors ligne* de *O-SAC*, *IO-SAC* et *F-SAC*.

Pour les jeux de données synthétiques, la valeur de densité utilisée pour générer les ensembles de données a été fixée empiriquement. Nous avons sélectionné celle qui donnait les résultats les plus satisfaisants sur différentes tailles de jeux de données. Par exemple, pour la distribution en disques de Poisson, la densité choisie génère des chevauchements mais les disques se touchent à peine. De même, pour une distribution normale, une valeur de densité trop élevée n'induit qu'un seul gros cluster, et inversement, une valeur trop faible ne permet pas de faire ressortir les propriétés d'une distribution normale dans les résultats. Bien que nous ne rapportions pas tous les résultats, différentes valeurs de densité ont été testées et n'ont pas vraiment d'impacts sur les résultats présentés ci-dessus. De même, les jeux de données réelles ont des densités différentes et cela n'affecte en rien la comparaison de nos algorithmes.

Dans nos expérimentations, nous avons montré que *F-SAC* est nettement plus rapide que les autres approches tout en donnant des résultats finaux comparables. Afin de s'assurer qu'il n'y a pas de biais liés à l'implémentation en JavaScript, nous avons également implémenté *F-SAC* en Java pour le comparer avec les versions originales de *QUAD+* et *QUAD+BIG* [12]. L'implémentation utilise une structure de R-arbres disponible [13] sans optimisation particulière. Les nouvelles expérimentations menées

12. https://github.com/Caster/growing-glyphs
13. https://github.com/Geomatys/geotoolkit

confirment les résultats précédents obtenus en Javascript : les ordres de grandeur restent similaires comme l'illustre le Tableau 4.4 [14].

Table 4.4 – *Temps d'exécution : Temps médian (ms) pour les algorithmes Java* en ligne

	QUAD+	QUAD+BIG	F-SAC
Trove	926,06	935,06	**48,95**
Glottolog	862,18	886,7	**41,27**
IRA	31,67	31,44	**4,79**
RISSE	28,44	23,69	**3,30**
OCS Wrecks	1 949,44	2 227,48	**45,53**
Siprojuris	26,65	26,24	**3,26**

4.7 Conclusion

Dans ce chapitre, nous avons exploré le problème du regroupement spatial agglomératif dans sa version *en ligne* et *hors ligne*. Nous avons proposé un nouvel algorithme, *F-SAC*, qui optimise la construction des clusters et utilise une structure de R-arbre pour un calcul rapide des requêtes de proximité particulièrement bien adapté au traitement de grands clusters. Des expérimentations approfondies ont été menées avec des approches de l'état de l'art qui ont toutes été implémentés dans le même langage de programmation. Six jeux de données réelles présentant des caractéristiques très différentes et 4 000 jeux de données synthétiques générées à partir de différentes distributions ont été utilisés pour comparer les approches. Ces expérimentations ont montré que *F-SAC* est nettement plus rapide que les approches précédentes tout en garantissant des résultats de regroupement au moins similaires en terme de qualité.

14. https://observablehq.com/d/90c52d9fb54ff51c décrit plus en détail les résultats de l'expérience.

CHAPITRE 5

CONCLUSIONS ET PERSPECTIVES

5.1 Résumé des contributions

Dans ce mémoire nous avons abordé la problématique de la visualisation de données prosopographiques. Pour que la visualisation soit adaptée aux besoins des historiens nous avons répondu aux trois questions : *Quoi ?*, *Pourquoi ?* et *Comment ?*

Dans le Chapitre 2, nous avons proposé la plateforme *ProsoVis* pour analyser et naviguer dans des données prosopographiques. L'étude de l'état de l'art nous a montré que de nombreuses approches se focalisaient sur une ou deux dimensions et n'étaient pas adaptées aux besoins des utilisateurs. Elle a également mis en évidence qu'il n'existait que peu de travaux pour visualiser les données prosopographiques. L'analyse des données nous a permis de répondre au *Quoi ?* Nous avons montré dans ce chapitre que les réponses aux *Pourquoi ?* et *Comment ?* ont été trouvées grâce à de nombreux échanges avec les historiens et ont nécessité de nombreuses itérations. *ProsoVis* contient une *vue globale* composée de 4 « sous-vues » mettant en évidence les différentes dimensions des données prosopographiques, *i.e.* le temps, l'espace et les relations entre individus. Une *vue détaillée* représente, pour des individus sélectionnés au sein de la vue globale, l'ensemble des informations disponibles dans la base ainsi que les relations partagées entre les individus. Différentes méthodes de filtrages sont également proposées pour aider l'utilisateur à naviguer et à explorer les données. Nous avons illustré l'utilisation de *ProsoVis* avec la base Siprojuris qui contient les données sur la carrière des enseignants de droit de 1800 à 1950.

En répondant au *Comment ?*, nous avons été confronté à deux problèmes relatifs à l'encombrement visuel. Le premier concerne le graphe des relations où des chevauchements de nœuds peuvent apparaître, le second est lié à un trop grand nombre d'informations à afficher sur la carte.

Pour répondre au problème de chevauchement, nous avons tout d'abord recherché dans l'état de l'art quel algorithme serait le plus approprié à nos besoins. Dans le Chapitre 3, nous avons montré que, même si des approches existaient, il n'y avait aucun moyen de choisir car ces dernières ne sont pas comparées sur les mêmes critères de qualité. Nous avons donc proposé une comparaison, via de très nombreuses expérimentations sur des données synthétiques et réelles, des différents algorithmes sur des critères communs. Cette étude nous a permis de sélectionner l'algorithme

PRISM particulièrement approprié à nos besoins. Elle permet également d'aider le concepteur de visualisation à sélectionner l'approche la plus adaptée en fonction de ses propres critères (*e.g.* temps, déformation spatiale, etc.). En outre, nous avons proposé la plateforme *AGORA* qui permet à l'utilisateur d'évaluer les algorithmes sur ses propres graphes.

Dans le Chapitre 4 nous avons proposé *F-SAC*, un nouvel algorithme de regroupement spatial agglomératif permettant de répondre à la problématique d'encombrement visuel lié au nombre trop important d'informations affichées sur une carte. Deux versions ont été proposées : *F-SAC en ligne* qui effectue en temps réel (lors de zoom) le clustering et *F-SAC hors ligne* qui calcule au préalable les clusters. Nous avons montré, à travers de nombreuses expérimentations sur des données réelles et synthétiques, que notre approche était beaucoup plus rapide que les autres travaux de l'état de l'art tout en garantissant des résultats similaires.

5.2 Perspectives

Les travaux présentés dans ce mémoire offrent de nombreuses perspectives. Certaines d'entre elles ont été proposées lors des discussions dans les chapitres. Dans cette section nous proposons différentes perspectives associées à chacun des travaux que nous avons menés.

5.2.1 ProsoVis

Prise en compte de relations plus complexes Dans *ProsoVis*, les relations sont représentées comme un lien unique entre deux individus dans un lieu donné. De nombreuses relations plus complexes pourraient être considérées. La première pourrait être d'un individu vers un groupe ou de deux groupes entre eux. Il s'agirait dans ce cas de pouvoir définir des agrégations d'individus et de définir la relation qu'ils partagent. Bien entendu, se pose la question de la visualisation : comment faire la différence entre des groupes d'individus reliés entre eux et simplement deux individus comme dans la version actuelle ? quelles variables visuelles seraient les plus appropriées ? comment intégrer ces deux représentations aux sein des vues actuelles ? Les représentations de graphes multicouches, *i.e.* différents types d'arcs peuvent relier les nœuds, ont montré que leur pouvoir d'expression était particulièrement adapté pour manipuler des données complexes possédant de nombreux types de relations entre elles (CUENCA et al., 2022 ; MCGEE et al., 2019). À l'heure actuelle, deux individus peuvent être reliés par deux relations de manière séparées. Représenter les relations via des graphes multicouches pourrait permettre de nouveaux types d'agrégation (*e.g.* les individus qui possèdent tous les mêmes types de liens multiples peuvent être mis en évidence) ou être utilisés pour étendre le graphe des relations et offrir d'autres suggestions. D'autres perspectives, comme mieux définir la relation d'un individu aux évènements, sont intéressantes. Par exemple, il pourrait être utile de pouvoir représenter l'émotion ou l'intérêt d'un individu pour un évènement (*e.g.* spécifier qu'il s'agit d'un évènement marquant).

Vers une plateforme de visualisation analytique Dans Keim et al. (2010), les auteurs définissent la visualisation analytique comme une approche qui « *combine des techniques d'analyse automatisées avec des visualisations interactives pour une compréhension, un raisonnement et une prise de décision efficaces sur la base d'ensembles de données très volumineux et complexes.* » À l'heure actuelle, la plateforme *ProsoVis* permet de visualiser de manière interactive les différentes données afin d'aider l'expert à valider ou tester ses hypothèses. Il s'agit d'une plateforme de visualisation d'information.

Une perspective importante est de l'étendre pour avoir une plateforme de visualisation analytique en ajoutant des connaissances issues de techniques d'analyse automatisées, *e.g.* fouille de données ou apprentissage automatique. De telles connaissances permettraient, par exemple, de mettre en évidence dans la visualisation des parcours communs, des comportements inattendus, etc. En outre les interactions de l'utilisateur permettraient aux techniques d'apprentissage de se focaliser sur les zones d'intérêt de l'utilisateur. La Figure 5.1 illustre ce type d'approche. Bien entendu, cette *boucle de rétroaction* n'est pas simple à mettre en place. Par exemple, elle nécessite que les approches d'apprentissage soient capables de considérer des données imparfaites, de disposer de mesures appropriées pour quantifier ou qualifier ce qu'est « un parcours commun ou inattendu », etc. et soulève donc de nombreux problèmes. Concernant l'interaction entre la visualisation et l'apprentissage, de nombreux problèmes existent également : *comment réduire l'espace de recherche des algorithmes pour tenir compte de l'interaction de l'utilisateur ? quel type d'interaction est la plus appropriée ? quelles sont les conséquences sur les temps de réponses pour ne pas pénaliser la navigation ?*

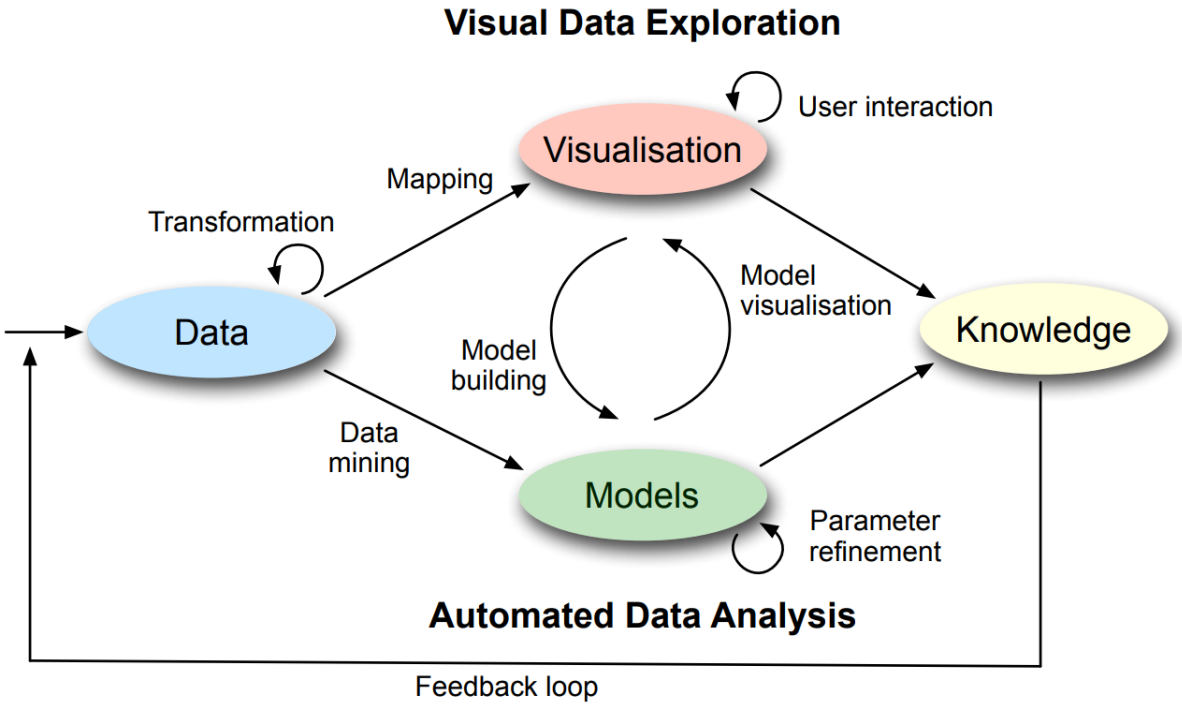


Figure 5.1 – *Intégration de méthodes d'analyse visuelle de données et de méthodes d'analyse automatiques pour une aide à la décision interactive et évolutive* (Keim et al., 2010, *p. 10*)

Richesse des données Dans *ProsoVis*, nous avons vu que le modèle de données permettait de gérer la représentation de l'incertitude sur les données et que, au sein du panneau d'information, pour répondre au besoin **[B6]** (mise en évidence des erreurs), nous proposons d'y répondre à l'aide d'un code couleur : une icône rouge pour signaler une erreur dans les données, une orange pour signaler une erreur et une bleue pour un complément d'information. Différents types de visualisation de l'incertitude peuvent être considérés et, en fonction de l'appartenance à des catégories, modifier la façon dont la donnée est affichée. Par exemple, BENITO-SANTOS et al. (2021) proposent différentes catégories d'incertitude dans les données historiques. Dans GSCHWANDTNER et al. (2016) les auteurs s'intéressent à la visualisation de différentes incertitudes au sein de frises chronologiques. MACEACHREN et al. (2012) proposent une étude sur les types de visualisations les plus adaptées pour représenter des incertitudes pour différents types de données. Bien entendu, des problèmes se posent également sur la manière d'agréger des données incertaines. Par exemple, dans le panneau d'information, si les éléments d'un sous groupe ont des signalisations de types différentes (*e.g.* rouge, orange, bleue), la couleur de l'icône au plus haut niveau est celle exprimant la plus grande sévérité (*i.e.* rouge pour une erreur). La prise en compte d'agrégation au sein des autres types de vues (*e.g.* carte ou frise chronologique) est une perspective intéressante. Enfin, quid des données de mobilités ? Nous considérons les évènements comme ponctuels dans l'espace. Cependant un déplacement est continu dans l'espace. Dans ce cas, comment représenter ce déplacement et ses incertitudes souvent associées ?

5.2.2 AGORA

Autres algorithmes Notre étude se concentre sur les algorithmes explicitement conçus pour supprimer les chevauchements de nœuds des plongements de graphes. Il serait intéressant d'envisager d'autres algorithmes dédiés à des problèmes connexes. Par exemple, van GARDEREN et al. (2017) proposent une heuristique pour supprimer les chevauchements de rectangles géoréférencés. Leur objectif est de minimiser le déplacement des nœuds tout en préservant l'ordre orthogonal. De la même manière, NICKEL et al. (2019) s'intéressent au maintien de la stabilité de cartogrammes de Demers variant dans le temps et proposent un algorithme de suppression des chevauchements des rectangles.

Autres critères Dans la Section 3.3, nous avons décrit 22 métriques, nous les avons groupées en 5 classes selon les propriétés qu'elles visent à capturer et nous en avons sélectionné une représentative pour chaque classe. Parmi les 22 métriques, 18 proviennent de la littérature sur les algorithmes de suppression des chevauchements des nœuds, les autres sont des améliorations mineures des précédentes afin de répondre à des inconvénients soulignés dans les discussions. Cependant, nous n'avons pas proposé de métrique radicalement différente et nous n'avons pas présenté comment des métriques provenant d'autres applications pourraient être adaptées pour répondre au problème traité ici. Il s'agit là d'une piste de recherche intéressante, mais qui dépasse le cadre du présent travail. Par exemple, FADLOUN et al. (2017) ont proposé des critères pour évaluer la qualité des algorithmes de suppression des chevauchements des nœuds en 1D et il serait intéressant d'étudier comment ils pourraient être adaptés au cas de la

2D. Sondag et al. (2018) ont proposé une métrique pour quantifier le changement des positions relatives des rectangles dans le contexte d'algorithmes de positionnement de cartes proportionnelles stables (*stable treemap*). Une autre source d'inspiration pourrait venir de la visualisation de données géographiques. Par exemple, Guo et Gahegan (2006) ont proposé plusieurs approches pour encoder la proximité spatiale entre les éléments et Haunert et Sering (2011) quantifient les distorsions locales des réseaux routiers.

Qualité du plongement Notre étude compare les algorithmes en fonction de leur capacité à préserver certaines propriétés du plongement initial. Nous n'avons considéré que les mesures qui quantifient dans quelle mesure les algorithmes préservent la carte mentale entre un plongement initial et le plongement sans chevauchements correspondant. Il existe cependant de nombreuses approches pour mesurer les différents aspects de la qualité d'un plongement en soi (Purchase, 2002) et une direction de recherche intéressante serait d'évaluer dans quelle mesure un algorithme de suppression des chevauchements des nœuds dégrade la qualité d'un plongement initial. Une telle étude n'est pas triviale car il existe de nombreux algorithmes de positionnement qui visent à optimiser différents critères de qualité, et il devrait être important de tester la dégradation du plongement au regard de ces algorithmes. Par exemple, un algorithme de dessin A_1 peut produire de meilleurs résultats sur un critère c qu'un autre algorithme A_2, mais une plus grande dégradation de c lors de la suppression des chevauchements. Pour sélectionner le meilleur algorithme de suppression des chevauchements des nœuds préservant c, il faut appliquer, pour plusieurs graphes initiaux, les plongements de différents algorithmes de positionnement et évaluer l'application des différents algorithmes de suppression des chevauchements sur ces plongements.

5.2.3 F-SAC

Vers un regroupement étendu Lors de la définition de *F-SAC*, nous avons considéré la proximité spatiale pour effectuer les regroupements, *i.e.* regroupement spatial agglomératif. Cette approche est bien entendue très bien adaptée à des données univariées. Une perspective intéressante serait d'étendre *F-SAC* à des données multivariées, *i.e.* en considérant des catégories sur les données. L'intérêt serait alors de proposer une approche d'agglomération utilisant non seulement l'aspect spatial, mais aussi tenant compte des catégories associées aux données. Ainsi, au niveau de la visualisation, les éléments proches et de la même catégorie apparaîtraient dans le même cluster. Bien entendu, un plus grand nombre de clusters serait susceptible d'apparaître mais via un mécanisme d'interaction ou de filtre offert à l'utilisateur, il serait possible d'offrir des priorités ou de n'afficher que ceux pour lesquels l'utilisateur est intéressé. En outre, il est tout à fait envisageable de prendre en compte des préférences lors du regroupement dans *F-SAC*, *i.e.* plus d'intérêt pour l'aspect spatial ou pour la catégorie, et ainsi faciliter l'analyse. Il s'agirait dans ce cas, lors du calcul des distances d'associer un poids relatif, défini par l'utilisateur, à l'aspect spatial et à l'aspect catégorie avec une fonction de type $\alpha \dots (1-\alpha)$, qui peut être étendue pour prendre en compte des préférences utilisateurs pour les catégories.

Approches hybrides Dans le cadre de cette thèse, nous nous sommes intéressés aux problèmes liés à l'encombrement visuel. Nous avons abordé respectivement la problématique de la suppression des chevauchements des nœuds via *AGORA* et d'agglomération spatiale via *F-SAC*. Une perspective intéressante serait de proposer une approche hybride qui utiliserait la suppression de chevauchements lors de la visualisation sur une carte. Dans *F-SAC*, les objets sont regroupés lorsque la distance qui les sépare est inférieure à un certain seuil. Comme nous l'avons vu l'approche est très efficace. Une approche hybride consisterait à ne pas forcément fusionner tous les points qui se chevauchent dans un cluster mais plutôt de les déplacer légèrement. Bien entendu, de nombreuses expérimentations doivent être réalisées pour mieux analyser les différents seuils à spécifier et déterminer à partir de quel seuil il est préférable de fusionner ou au contraire de déplacer légèrement le point.

BIBLIOGRAPHIE

Aigner, W., Miksch, S., & Schumann, H. (2011). *Visualization of time-oriented data*. Springer. (cf. page 11).

Akoka, J., Comyn-Wattiau, I., & du Mouza, C. (2020). Conception de Bases de Données Prosopographiques en Histoire - Un Etat de l'Art. *Revue ouverte d'ingénierie des systèmes d'information*, *20*(3), 1-19. https://doi.org/10.21494/ISTE.OP.2020.0531 (cf. page 1).

Andrienko, N., Andrienko, G., & Gatalsky, P. (2003). Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, *14*(6), 503-541. https://doi.org/10.1016/S1045-926X(03)00046-6 (cf. page 47).

Archambault, D. W., & Purchase, H. C. (2013). The "Map" in the mental map: Experimental results in dynamic graph drawing. *International Journal of Human-Computer Studies*, *71*(11), 1044-1055. https://doi.org/10.1016/j.ijhcs.2013.08.004 (cf. page 52).

Arge, L., Berg, M. D., Haverkort, H., & Yi, K. (2008). The priority R-tree: a practically efficient and worst-case optimal R-tree. *ACM Transactions on Algorithms*, *4*(1). https://doi.org/10.1145/1328911.1328920 (cf. page 84).

Bach, B., Dragicevic, P., Archambault, D., Hurter, C., & Carpendale, S. (2017). A descriptive framework for temporal data visualizations based on generalized space-time cubes. *Computer Graphics Forum*, *36*(6), 36-61. https://doi.org/10.1111/cgf.12804 (cf. page 47).

Barabaśi, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509-512. https://doi.org/10.1126/science.286.5439.509 (cf. page 62).

Beckmann, N., Kriegel, H.-P., Schneider, R., & Seeger, B. (1990). The R*-tree: An efficient and robust access method for points and rectangles. *SIGMOD Rec.*, *19*(2), 322-331. https://doi.org/10.1145/93605.98741 (cf. page 84).

Benito-Santos, A., Doran, M., Rocha, A., Wandl-Vogt, E., Edmond, J., & Therón, R. (2021). Evaluating a taxonomy of textual uncertainty for collaborative visualisation in the digital humanities. *Information*, *12*(11), 436. https://doi.org/10.3390/info12110436 (cf. page 104).

Bertin, J. (2011). *Semiology of graphics: Diagrams, networks, maps*. ESRI Press. (cf. pages 32, 76).

Bezerianos, A., Dragicevic, P., Fekete, J.-D., Bae, J., & Watson, B. (2010). GeneaQuilts: A system for exploring large genealogies. *IEEE Transactions on Visualization and Computer Graphics*, *16*(6), 1073-1081. https://doi.org/10.1109/TVCG.2010.159 (cf. pages 13, 15, 16, 47).

Bostock, M., Ogievetsky, V., & Heer, J. (2011). D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, *17*(12), 2301-2309. https://doi.org/10.1109/TVCG.2011.185 (cf. page 25).

Brehmer, M., Lee, B., Bach, B., Riche, N. H., & Munzner, T. (2017). Timelines revisited: A design space and considerations for expressive storytelling. *IEEE Transactions on Visualization and Computer Graphics*, *23*(9), 2151-2164. https://doi.org/10.1109/TVCG.2016.2614803 (cf. page 28).

Card, S. K., Mackinlay, J. D., & Shneiderman, B. (1999). *Readings in information visualization: Using vision to think* (S. K. Card, J. D. Mackinlay & B. Shneiderman, Éd.). Morgan Kaufmann Publishers. (cf. page 2).

Castermans, T., Speckmann, B., & Verbeek, K. (2019). A practical algorithm for spatial agglomerative clustering. In S. G. Kobourov & H. Meyerhenke (Éd.), *Workshop on algorithm engineering and experiments* (*ALENEX*) (p. 174-185). SIAM. https://doi.org/10.1137/1.9781611975499.14 (cf. pages 79, 80, 83, 87).

Chen, F., Piccinini, L., Poncelet, P., & Sallaberry, A. (2019). Node overlap removal algorithms: A comparative study. In D. Archambault & C. D. Tóth (Éd.), *Proceedings of the international symposium on graph drawing and network visualization* (*GD*) (p. 179-192). Springer. https://doi.org/10.1007/978-3-030-35802-0_14 (cf. page 5).

Chen, F., Piccinini, L., Poncelet, P., & Sallaberry, A. (2020). Node overlap removal algorithms: An extended comparative study. *Journal of Graph Algorithms and Applications*, *24*(4), 683-706. https://doi.org/10.7155/jgaa.00532 (cf. pages 5, 76).

Chen, F., Sallaberry, A., & Poncelet, P. (2022). F-SAC: A fast and efficient algorithm for spatial agglomerative clustering. Soumis (cf. page 6).

Chimani, M., Gutwenger, C., Jünger, M., Klau, G. W., Klein, K., & Mutzel, P. (2013). The Open Graph Drawing Framework (OGDF). In R. Tamassia (Éd.), *Handbook on graph drawing and visualization* (p. 543-569). Chapman; Hall/CRC. (cf. page 62).

Cho, I., Dou, W., Wang, D. X., Sauda, E., & Ribarsky, W. (2016). VAiRoma: A visual analytics system for making sense of places, times, and events in roman history. *IEEE Transactions on Visualization and Computer Graphics*, *22*(1), 210-219. https://doi.org/10.1109/TVCG.2015.2467971 (cf. pages 11, 12).

Cockburn, A., Karlson, A., & Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, *41*(1), 1-31. https://doi.org/10.1145/1456650.1456652 (cf. page 32).

Cook, R. L. (1986). Stochastic sampling in computer graphics. *ACM Transactions on Graphics*, *5*(1), 51-72. https://doi.org/10.1145/7529.8927 (cf. page 88).

Cuenca, E., Sallaberry, A., Ienco, D., & Poncelet, P. (2022). VERTIGo: A visual platform for querying and exploring large multilayer networks. *IEEE Transactions on Visualization and Computer Graphics*, *28*(3), 1634-1647. https://doi.org/10.1109/TVCG.2021.3067820 (cf. page 102).

Cuenca, E., Sallaberry, A., Wang, F. Y., & Poncelet, P. (2018). MultiStream: A multiresolution streamgraph approach to explore hierarchical time series. *IEEE Transactions on Visualization and Computer Graphics*, *24*(12), 3160-3173. https://doi.org/10.1109/TVCG.2018.2796591 (cf. page 28).

de Berg, M., Bose, P., Cheong, O., & Morin, P. (2004). On simplifying dot maps. *Computational Geometry*, *27*(1), 43-62. https://doi.org/10.1016/j.comgeo.2003.07.005 (cf. page 76).

Delpu, P.-M. (2015). La prosopographie, une ressource pour l'histoire sociale. *Hypothèses*, *18*(1), 263-274. https://doi.org/10.3917/hyp.141.0263 (cf. page 1).

Dörk, M., Carpendale, S., Collins, C., & Williamson, C. (2008). VisGets: Coordinated visualizations for web-based information exploration and discovery. *IEEE Transactions on Visualization and Computer Graphics*, *14*(6), 1205-1212. https://doi.org/10.1109/TVCG.2008.175 (cf. page 11).

Dou, W., Wang, X., Skau, D., Ribarsky, W., & Zhou, M. X. (2012). LeadLine: Interactive visual analysis of text data through event identification and exploration. *2012 IEEE Conference on Visual Analytics Science and Technology* (*VAST*), 93-102. https://doi.org/10.1109/VAST.2012.6400485 (cf. pages 16, 17).

Dwyer, T., Marriott, K., & Stuckey, P. J. (2005). Fast node overlap removal. *Proceedings of the International Symposium on Graph Drawing* (*GD*), 153-164. https://doi.org/10.1007/11618058_15 (cf. page 62).

Ellis, G., & Dix, A. (2002). Density control through random sampling: An architectural perspective. *Proceedings Sixth International Conference on Information Visualisation*, 82-90. https://doi.org/10.1109/IV.2002.1028760 (cf. page 76).

Erdös, P., & Rényi, A. (1959). On random graphs. *Publicationes Mathematicae Debrecen*, *6*, 290-291 (cf. page 62).

Fadloun, S., Poncelet, P., Rabatel, J., Roche, M., & Sallaberry, A. (2017). Node overlap removal for 1d graph layout. *Proceeding of the International Conference on Information Visualisation* (*IV*), 224-229. https://doi.org/10.1109/iV.2017.14 (cf. page 104).

Finkel, R. A., & Bentley, J. L. (1974). Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, *4*(1), 1-9. https://doi.org/10.1007/bf00288933 (cf. page 83).

Gansner, E. R., & Hu, Y. (2010). Efficient, proximity-preserving node overlap removal. *Journal of Graph Algorithms and Applications*, *14*(1), 53-74. https://doi.org/10.7155/jgaa.00198 (cf. pages 43, 55, 56, 61-63).

Gansner, E. R., & North, S. C. (2000). An open graph visualization system and its applications to software engineering. *Software: Practice and experience*, *30*(11), 1203-1233. https://doi.org/10.1002/1097-024X(200009)30:11<1203::AID-SPE338>3.0.CO;2-N (cf. page 63).

Gibson, H., Faith, J., & Vickers, P. (2013). A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization*, *12*(3-4), 324-357. https://doi.org/10.1177/1473871612455749 (cf. page 52).

Griffin, A. L., & Robinson, A. C. (2015). Comparing color and leader line highlighting strategies in coordinated view geovisualizations. *IEEE Transactions on Visualization and Computer Graphics*, *21*(3), 339-349. https://doi.org/10.1109/TVCG.2014.2371858 (cf. page 24).

Gschwandtner, T., Bogl, M., Federico, P., & Miksch, S. (2016). Visual encodings of temporal uncertainty: A comparative user study. *IEEE Transactions on Visualization and Computer Graphics*, *22*(1), 539-548. https://doi.org/10.1109/TVCG.2015.2467752 (cf. page 104).

Guo, D., & Gahegan, M. (2006). Spatial ordering and encoding for geographic data mining and visualization. *Journal of Intelligent Information Systems*, *27*(3), 243-266. https://doi.org/10.1007/s10844-006-9952-8 (cf. page 105).

Hachul, S., & Jünger, M. (2004). Drawing large graphs with a potential-field-based multilevel algorithm. *Proceedings of the International Symposium on Graph Drawing (GD)*, 285-295. https://doi.org/10.1007/978-3-540-31843-9_29 (cf. pages 43, 63).

Han, D., Parsad, G., Kim, H., Shim, J., Kwon, O.-S., Son, K. A., Lee, J., Cho, I., & Ko, S. (2021). HisVA: A visual analytics system for learning history. *IEEE Transactions on Visualization and Computer Graphics, PP*. https://doi.org/10.1109/TVCG.2021.3086414 (cf. pages 12, 13, 32).

Har-peled, S. (2011). Quadtrees - hierarchical grids. *Geometric Approximation Algorithms*, 12-26. https://doi.org/10.1090/surv/173 (cf. page 84).

Haunert, J., & Sering, L. (2011). Drawing road networks with focus regions. *IEEE Transactions on Visualization and Computer Graphics, 17*(12), 2555-2562. https://doi.org/10.1109/TVCG.2011.191 (cf. page 105).

Hayashi, K., Inoue, M., Masuzawa, T., & Fujiwara, H. (1998). A layout adjustment problem for disjoint rectangles preserving orthogonal order. *Proceedings of the International Symposium on Graph Drawing (GD)*, 183-197. https://doi.org/10.1007/3-540-37623-2_14 (cf. page 62).

Hinrichs, U., Alex, B., Clifford, J., Watson, A., Quigley, A., Klein, E., & Coates, C. M. (2015). Trading consequences: A case study of combining text mining and visualization to facilitate document exploration. *Digital Scholarship in the Humanities*, i50-i75. https://doi.org/10.1093/llc/fqv046 (cf. page 17).

Huang, X., & Lai, W. (2003). Force-transfer: A new approach to removing overlapping nodes in graph layout. *Proceedings of the Australasian Computer Science Conference (ACSC)*, 349-358. https://doi.org/10.5555/783106.783146 (cf. pages 55, 60).

Huang, X., Lai, W., Sajeev, A., & Gao, J. (2007). A new algorithm for removing node overlapping in graph visualization. *Information Sciences, 177*(14), 2821-2844. https://doi.org/10.1016/j.ins.2007.02.016 (cf. pages 50, 55, 56, 58, 60, 62).

Hyvönen, E., Leskinen, P., Tamper, M., Rantala, H., Ikkala, E., Tuominen, J., & Keravuori, K. (2019). BiographySampo – publishing and enriching biographies on the semantic web for digital humanities research. In P. Hitzler, M. Fernández, K. Janowicz, A. Zaveri, A. J. Gray, V. Lopez, A. Haller & K. Hammar (Éd.), *The semantic web* (p. 574-589, T. 11503). Springer International Publishing. https://doi.org/10.1007/978-3-030-21348-0_37 (cf. pages 18, 20, 48).

Itoh, M., & Akaishi, M. (2012). Visualization for changes in relationships between historical figures in chronicles. *2012 16th International Conference on Information Visualisation*, 283-290. https://doi.org/10.1109/IV.2012.55 (cf. pages 13-16).

Itoh, M., Toyoda, M., & Kitsuregawa, M. (2010). An interactive visualization framework for time-series of web graphs in a 3D environment. *2010 14th International Conference Information Visualisation*, 54-60. https://doi.org/10.1109/IV.2010.18 (cf. page 13).

Janicke, S., Focht, J., & Scheuermann, G. (2016). Interactive visual profiling of musicians. *IEEE Transactions on Visualization and Computer Graphics, 22*(1), 200-209. https://doi.org/10.1109/TVCG.2015.2467620 (cf. pages 14-16).

Kamel, I., & Faloutsos, C. (1994). Hilbert r-tree: An improved r-tree using fractals. In J. B. Bocca, M. Jarke & C. Zaniolo (Éd.), *VLDB'94, proceedings of 20th international conference on very large data bases, september 12-15, 1994, santiago de chile, chile*

(p. 500-509). Morgan Kaufmann. http://www.vldb.org/conf/1994/P500.PDF (cf. page 84).
Kehrer, J., & Hauser, H. (2013). Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, *19*(3), 495-513. https://doi.org/10.1109/TVCG.2012.110 (cf. page 24).
Keim, D., Kohlhammer, J., Ellis, G., & Mansmann, F. (2010). *Mastering the information age – solving problems with visual analytics*. (cf. page 103).
Khulusi, R., Kusnick, J., Focht, J., & Janicke, S. (2019). An interactive chart of biography. *2019 IEEE Pacific Visualization Symposium (PacificVis)*, 257-266. https://doi.org/10.1109/PacificVis.2019.00038 (cf. pages 14-16, 28).
Kusnick, J., Khulusi, R., Focht, J., & Jänicke, S. (2020). A timeline metaphor for analyzing the relationships between musical instruments and musical pieces. *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 240-251. https://doi.org/10.5220/0008990502400251 (cf. pages 14, 16).
Kwon, O.-H., & Ma, K.-L. (2020). A deep generative model for graph layout. *IEEE Transactions on Visualization and Computer Graphics*, *26*(1), 665-675. https://doi.org/10.1109/TVCG.2019.2934396 (cf. page 52).
Latif, S., Agarwal, S., Gottschalk, S., Chrosch, C., Feit, F., Jahn, J., Braun, T., Tchenko, Y. C., Demidova, E., & Beck, F. (2021). Visually connecting historical figures through event knowledge graphs. *2021 IEEE Visualization Conference (VIS)*, 156-160. https://doi.org/10.1109/VIS49827.2021.9623313 (cf. pages 17, 18, 48).
Leskinen, P., Hyvönen, E., & Tuominen, J. (2017). Analyzing and visualizing prosopographical linked data based on short biographies. *Biographical Data in a Digital World 2017* (cf. page 18).
Li, W., Eades, P., & Nikolov, N. (2005). Using spring algorithms to remove node overlapping, 131-140. https://doi.org/10.5555/1082315.1082334 (cf. pages 55-58, 62).
Lyons, K. A., Meijer, H., & Rappaport, D. (1998). Algorithms for cluster busting in anchored graph drawing. *Journal of Graph Algorithms and Applications*, *2*(1), 1-24. https://doi.org/10.7155/jgaa.00004 (cf. pages 55, 60).
MacEachren, A. M., Roth, R. E., O'Brien, J., Li, B., Swingley, D., & Gahegan, M. (2012). Visual semiotics & uncertainty visualization: An empirical study. *IEEE Transactions on Visualization and Computer Graphics*, *18*(12), 2496-2505. https://doi.org/10.1109/TVCG.2012.279 (cf. page 104).
Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A. N., & Theodoridis, Y. (2006). *R-trees: Aheory and applications*. Springer London. https://doi.org/10.1007/978-1-84628-293-5 (cf. pages 84, 87).
Marriott, K., Stuckey, P., Tam, V., & He, W. (2003). Removing node overlapping in graph layout using constrained optimization. *Constraints*, *8*(2), 143-171. https://doi.org/10.1023/A:1022371615202 (cf. pages 55, 60).
McGee, F., Ghoniem, M., Melançon, G., Otjacques, B., & Pinaud, B. (2019). The state of the art in multilayer network visualization. *Computer Graphics Forum*, *38*(6), 125-149. https://doi.org/10.1111/cgf.13610 (cf. page 102).

Meulemans, W. (2019). Efficient optimal overlap removal: Algorithms and experiments. *Computer Graphics Forum*, *38*(3), 713-723. https://doi.org/10.1111/cgf.13722 (cf. pages 62, 69).
Misue, K., Eades, P., Lai, W., & Sugiyama, K. (1995). Layout adjustment and the mental map. *Journal of Visual Languages & Computing*, *6*(2), 183-210. https://doi.org/10.1006/jvlc.1995.1010 (cf. pages 3, 52, 53, 55-57, 62).
Miyakita, G., Leskinen, P., & Hyvönen, E. (2018). Using linked data for prosopographical research of historical persons: Case u.s. congress legislators. In M. Ioannides, E. Fink, R. Brumana, P. Patias, A. Doulamis, J. Martins & M. Wallace (Éd.), *Digital heritage. Progress in cultural heritage: Documentation, preservation, and protection* (p. 150-162, T. 11197). Springer International Publishing. https://doi.org/10.1007/978-3-030-01765-1_18 (cf. page 18).
Munzner, T. (2014). *Visualization analysis and design*. CRC Press. (cf. pages 2, 3, 46).
Nachmanson, L., Nocaj, A., Bereg, S., Zhang, L., & Holroyd, A. (2016). Node overlap removal by growing a tree. *Proceedings of the International Symposium on Graph Drawing and Network Visualization (GD)*, 33-43. https://doi.org/10.1007/978-3-319-50106-2_3 (cf. pages 55, 61-63).
Nickel, S., Sondag, M., Meulemans, W., Chimani, M., Kobourov, S. G., Peltonen, J., & Nöllenburg, M. (2019). Computing stable demers cartograms. *Proceedings of the International Symposium on Graph Drawing and Network Visualization (GD)*, 46-60. https://doi.org/10.1007/978-3-030-35802-0_4 (cf. page 104).
Perlin, K. (1985). An image synthesizer. *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, 287-296. https://doi.org/10.1145/325334.325247 (cf. page 88).
Peuquet, D. J. (1994). It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers*, *84*(3), 441-461. https://doi.org/10.1111/j.1467-8306.1994.tb01869.x (cf. page 47).
Priestley, J. (1765). A chart of biography. *Lectures on History and General Policy* (cf. page 14).
Purchase, H. C. (2002). Metrics for graph drawing aesthetics. *Journal of Visual Languages & Computing*, *13*(5), 501-516. https://doi.org/10.1006/jvlc.2002.0232 (cf. page 105).
Risse, W. (1979). *Bibliographia logica: Verzeichnis der Zeitschriftenartikel zur Logik*. Olms. (cf. page 87).
Roberts, J. C., Al-maneea, H., Butcher, P. W. S., Lew, R., Rees, G., Sharma, N., & Frankenberg-Garcia, A. (2019). Multiple Views: Different meanings and collocated words. *Computer Graphics Forum*, *38*(3), 79-93. https://doi.org/https://doi.org/10.1111/cgf.13673 (cf. page 24).
Roberts, J. C. (2007). State of the art: coordinated & multiple views in exploratory visualization. *Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV 2007)*, 61-71. https://doi.org/10.1109/CMV.2007.20 (cf. page 24).
Rosenholtz, R., Li, Y., & Nakano, L. (2007). Measuring visual clutter. *Journal of Vision*, *7*(2), 17-17. https://doi.org/10.1167/7.2.17 (cf. page 76).

Sallaberry, A. (2020). *Visualisation d'information : techniques et solutions visuelles pour l'exploration de données relationnelles, temporelles et spatiales* [Habilitation à diriger des recherches]. Université de Montpellier. (cf. page 2).

Scheepens, R., van de Wetering, H., & van Wijk, J. J. (2014). Non-overlapping aggregated multivariate glyphs for moving objects. In I. Fujishiro, U. Brandes, H. Hagen & S. Takahashi (Éd.), *IEEE pacific visualization symposium* (*PacificVis*) (p. 17-24). IEEE Computer Society. https://doi.org/10.1109/PacificVis.2014.13 (cf. pages 79, 87).

Schich, M., Song, C., Ahn, Y.-Y., Mirsky, A., Martino, M., Barabási, A.-L., & Helbing, D. (2014). Quantitative social science. A network framework of cultural history. *Science* (*New York, N.Y.*), *345*(6196), 558-562. https://doi.org/10.1126/science.1240064 (cf. pages 18, 19).

Schmitz, P., & Pearce, L. (2013). Berkeley prosopography services: Ancient families, modern tools. *Proceedings of the 1st International Workshop on Collaborative Annotations in Shared Environment: Metadata, Vocabularies and Techniques in the Digital Humanities*. https://doi.org/10.1145/2517978.2517980 (cf. page 17).

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. *Proceedings 1996 IEEE Symposium on Visual Languages*, 336-343. https://doi.org/10.1109/VL.1996.545307 (cf. page 24).

Sondag, M., Speckmann, B., & Verbeek, K. (2018). Stable treemaps via local moves. *IEEE Transactions on Visualization and Computer Graphics*, *24*(1), 729-738. https://doi.org/10.1109/TVCG.2017.2745140 (cf. page 105).

Stone, L. (1971). Prosopography. *Daedalus*, *100*(1), 46-79 (cf. page 1).

Strobelt, H., Spicker, M., Stoffel, A., Keim, D. A., & Deussen, O. (2012). Rolled-out wordles: A heuristic method for overlap removal of 2D data representatives. *Computer Graphics Forum*, *31*(3), 1135-1144. https://doi.org/10.1111/j.1467-8659.2012.03106.x (cf. pages 55, 57-60, 62, 63).

Sun, M., Namburi, A., Koop, D., Zhao, J., Li, T., & Chung, H. (2021). Towards systematic design considerations for visualizing cross-view data relationships. *IEEE Transactions on Visualization and Computer Graphics*, *PP*. https://doi.org/10.1109/TVCG.2021.3102966 (cf. page 24).

Tamassia, R. (Éd.). (2014). *Handbook of graph drawing and visualization*. CRC Press. (cf. page 52).

van Garderen, M., Pampel, B., Nocaj, A., & Brandes, U. (2017). Minimum-displacement overlap removal for geo-referenced data visualization. *Computer Graphics Forum*, *36*(3), 423-433. https://doi.org/10.1111/cgf.13199 (cf. page 104).

Ware, C. (2021). *Information visualization: Perception for design* (4th edition). Elsevier. (cf. page 32).

Warren, C. N., Shore, D., Otis, J., Wang, L., Finegold, M., & Shalizi, C. R. (2016). Six Degrees of Francis Bacon: A statistical method for reconstructing large historical social networks. *Digital Humanities Quarterly* (*DHQ*), *10*(3). Récupérée janvier 26, 2022, à partir de http://www.digitalhumanities.org/dhq/vol/10/3/000244/000244.html (cf. pages 18, 20).

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*, 440-442. https://doi.org/10.1038/30918 (cf. page 62).

Windhager, F., Federico, P., Salisu, S., Schlögl, M., & Mayr, E. (2017). A synoptic visualization framework for the multi-perspective study of biography and prosopography data. In *2nd IEEE VIS workshop on visualization for the digital humanities (VIS4DH)*. (cf. page 22).

Windhager, F., Federico, P., Schreder, G., Glinka, K., Dork, M., Miksch, S., & Mayr, E. (2019). Visualization of cultural heritage collection data: State of the art and future challenges. *IEEE Transactions on Visualization and Computer Graphics*, *25*(6), 2311-2330. https://doi.org/10.1109/TVCG.2018.2830759 (cf. page 17).