Visualisation de flux géographiques

Rapport pour la DATAR
sous mandat du Laboratoire Chôros

André Ourednik
Communauté d'études pour l'aménagement du territoire
École Polytechnique Fédérale de Lausanne

Juin 2013

Table des matières

1 Introduction

Les cartes de flux communiquent le mouvement d'objets tangibles (personnes, notes bancaires, produits etc.) et d'objets non-tangibles (énergie, idées, phonèmes etc.) à travers l'espace. Les premières cartes de flux apparaissent à partir du 19e siècle dans les travaux de Henry Drury Harness, Alphonse Belpaire (Figure 30) ou de Charles Joseph Minard.

Tout traitement informatique de cartes de flux est tributaire d'outils mathématiques et algorithmiques implémentés dans les logiciels. De ce point de vue, il existe deux manières de concevoir un flux : comme un phénomène *continu* ou comme un phénomène *discret*.

Les flux continus correspondent à des processus de diffusion dans l'espace euclidien et peuvent être présentés sous forme de champs de vecteurs, de zones de potentiel ou d'isolignes (2.3).

La cartographie de flux discrets, sur lesquels nous nous penchons prioritairement dans le présent document, relève de la visualisation de *graphes*[1] .

Un graphe est un ensemble de *nœuds (vertices)*, dont certains sont directement reliées par une ou plusieurs *arrêtes (edges)*. Si le graphe contient une arrête asymétrique (dénotant par exemple une relation du nœud *u* vers le nœud *v*, sans relation réciproque de *v* vers *u*) le graphe est dit *orienté* (*directionnal*). Si toutes les relations sont symétriques, le graphe est *non orienté* (*non-directionnal*). Chaque arrête peut être porteuse d'attributs qualitatifs (type de relation) et quantitatifs (le *poids* de chaque relation, c'est-à-dire son importance). Les nœuds peuvent de même être dotés d'une pluralité d'attributs.

Ce cadre abstrait s'ouvre au traitement d'une multiplicité de types de réseaux, dont ceux des flux de mobilité individuelle, d'information ou de produits matériels. Généralement, les flux sont modélisés par des graphes orientés, à moins que l'on ne traite de flux symétriques. Prenons comme exemple le Tableau 1, qui présente un flux de pendulaires à travers un réseau de transport public fictif composé des stations A, B, C et D. Traduite en graphe, cette matrice de flux équivaut à quatre nœuds et six arrêtes. Le poids de chaque arrête correspond au nombre de pendulaires voyageant des stations de départ dans les colonnes aux stations de destination dans les lignes. La Figure 1 et la Figure 2 donnent une image du graphe en question. On constate qu'il s'agit d'un graphe à une seule *composante*, tous les nœuds faisant partie d'un réseau inter-relié. Il est courant, dans les analyses de flux, de trouver des graphes à plusieurs composantes déconnectées (Figure 3) – cas de figure se présentant par exemple lors d'une analyse de réseaux de transports intra-urbains faite conjointement sur plusieurs villes.

Tableau 1 : Exemple d'une matrice de flux.

	A	B	C	D
A	-	38	0	0
B	10	-	0	31
C	0	28	-	0
D	12	95	0	-

[1] Biggs et al. 1986 ; Even 2011 ; Chartrand et al. 2010. L'origine de la représentation automatique de graphes est également à trouver dans le développement de circuits imprimés.

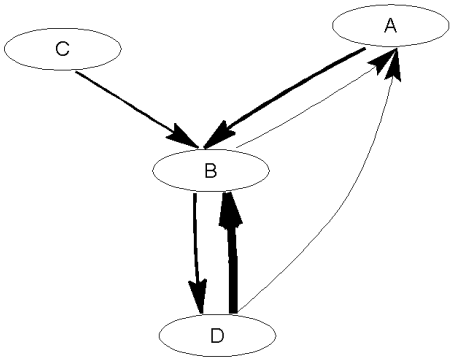


**Figure 1 : Représentation du graphe construit à partir du Tableau 1.
Figure produite à l'aide de Matlab (4.2.4.).**

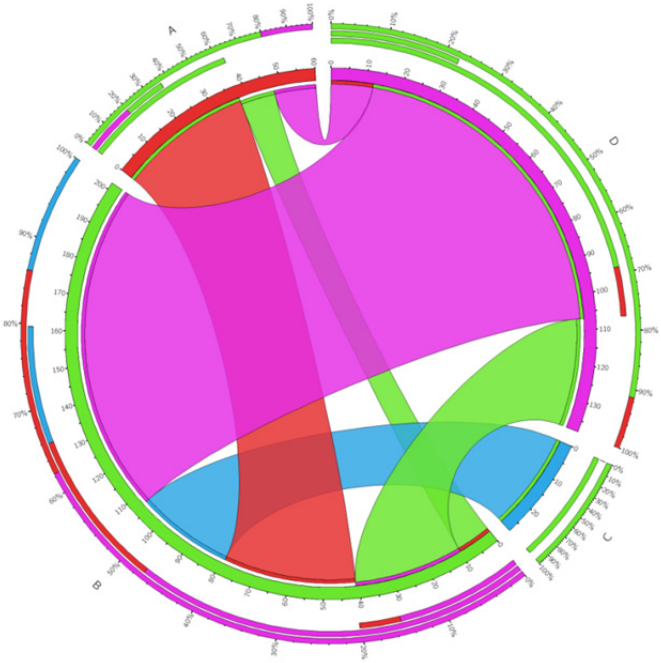


Figure 2 : Représentation de la matrice des flux du Tableau 1 sous forme de *chord diagram* avec Circos (4.2.6.1).

L'usage d'un *algorithme* élaboré s'impose lorsqu'il s'agit de visualiser un graphe possédant un grand nombre de nœuds et d'arrêtes. Des algorithmes identiques sont souvent implémentés dans une pluralité de logiciels.

Le présent document fait état des algorithmes existants sans se pencher spécifiquement sur leur fonctionnement, mais en clarifiant les transformations cartographiques qu'ils permettent d'accomplir. Ces dernières sont discutées dans le §2.

Dans le §3, nous présentons les logiciels majeurs, que nous évaluons en fonction des critères dressés, c'est-à-dire en fonction des algorithmes qu'ils proposent. La capacité d'interaction avec d'autres logiciels, en termes de données, d'éléments graphiques et de séquences programmatiques est aussi évaluée.

Le §4 est consacré à des solutions de cartographie de flux non comprises dans la catégorie de logiciels stricto sensu. Il s'agit, d'une part, de plugins et d'éléments de logiciels SIG et cartomatiques courants (ESRI, Manifold, qGIS etc.), d'autre part de bibliothèques cartographiques pour des langages de programmation majeurs.

Dans une dernière partie (§5), nous évoquons les services cartographiques permettant d'externaliser la production de cartes de flux.

2 Critères d'évaluation

Le but d'un logiciel de cartographie de flux est de faciliter la lisibilité et la mémorisation des informations. Mis à part des procédés cartographiques standard, comme la sélection, la généralisation ou la symbolisation, son cas particulier impose de chercher tout particulièrement à réduire le nombre de croisements d'arrêtes (Purchase 2000).

La préparation d'une carte de réseau de flux comprend généralement un prétraitement statistique de données, un traitement graphique et un post-traitement graphique lié au mode de diffusion de la carte. Certaines cartes peuvent également être présentées sous forme d'objets interactifs. Ces aspects sont strictement inter-reliés. La simplification d'une carte de réseau serait arbitraire si l'on ne calculait pas des métriques-réseau permettant d'identifier les nœuds pouvant être exclus de l'analyse, ou les arrêtes pouvant être regroupées. Penchons-nous d'abord sur le type d'éléments graphiques pouvant être produits.

2.1 Types d'éléments graphiques

Le graphe simple (Figure 3) est fait de points et de lignes. Des variations sont concevables en termes de taille, de forme, de couleurs des nœuds et des arrêtes. Également en termes d'agencement des éléments ; les nœuds par exemple, peuvent être distribués de manière aléatoire, en ligne, en cercle, ou suivant des algorithmes plus complexes cherchant à répondre à des critères spécifiques. Le fond de carte, enfin, peut rester blanc mais il peut aussi être doté d'éléments supplémentaires facilitant l'interprétation géographique, comme un fond de carte territorial (carte topographique, carte des limites administratives etc.) ou un ensemble de polygones servant à identifier des groupes de nœuds.

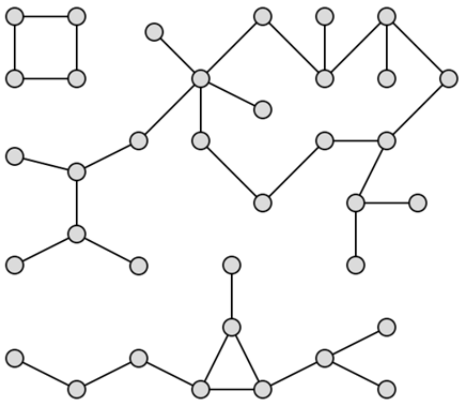


Figure 3 : Exemple de graphe simple à trois composantes. Parce qu'il peut être représenté sans croisement d'arrêtes dans le plan (2D), c'est aussi un « graphe planaire ».

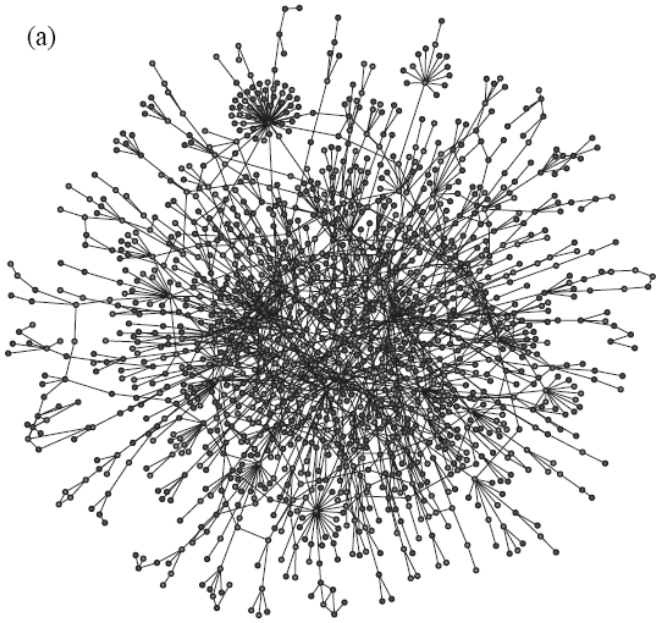


Figure 4 : Exemple de graphe complexe à une seule composante.

2.1.1 Optimisation de l'agencement de nœuds

Le traitement cartographique le plus fréquemment proposé dans les logiciels de visualisation de réseau porte sur le *déplacement* des nœuds[2], en vue de structurer l'information portant sur leurs liens. On fait en sorte, par exemple, que des lieux se rapprochent sur la carte en proportion du nombre de pendulaires qui circulent entre eux, ou encore en fonction de leur ressemblance structurelle en termes de leur position dans le réseau (centralité, participation à un sous-réseau etc.). Il existe plusieurs stratégies pour parvenir à ce but.

2.1.1.1 Approches basées sur la force

Les approches « basées sur la force » (*force-based*) rassemblent les nœuds fortement liés. Les nœuds sont déplacés de manière itérative, selon une métaphore de système de ressorts et d'interaction de particules. Typiquement, cette approche combine des forces d'attraction et de répulsion entre nœuds, en fonction de paramètres définis, de manière à trouver un agencement où la longueur des arrêtes est la plus courte possible, tout en maximisant la séparation entre les nœuds. L'avantage de l'approche est de répartir les nœuds dans l'espace, ce qui permet de les distinguer, tout en garantissant la proximité des nœuds reliés par un flux fort.

La méthode de *spring embedding* (Eades 1984; Fruchterman/Reingold 1991) traite chaque nœud comme une particule de charge identique, obtenant une force répulsive entre chaque paire de nœuds. Les arrêtes, au contraire, sont traitées comme des ressorts attirant les paires de nœuds qu'elles connectent. À partir d'un agencement initial – qui peut être aléatoire ou basé sur les coordonnées topographiques des nœuds – l'algorithme procède par itérations. À chaque pas, il calcule les vecteurs de forces répulsives et attractives agissant sur chaque nœud, et les déplace en fonction. Afin d'empêcher l'éloignement infini des composantes d'un graphe, l'effet de répulsion est décroissant avec la distance. D'autres algorithmes comme le GEM (*Generalized Expectation-Maximization*) ou Kamada/Kawai (1998) évitent tout particulièrement le croisement de nœuds. L'algorithme Yifan-

[2] Les termes anglais employés dans le jargon des logiciels de visualisation sont aussi *layout*, *positioning* ou *spatialization*.

Hu (2005) combine l'approche basée sur la force avec une approche hiérarchique pour réduire la complexité et augmenter la rapidité d'exécution de l'algorithme ; dans cette version, on prend en compte les forces répulsives entre clusters de nœuds que l'on traite comme super-nœuds.
L'approche OpenOrd (Martin et al. 2011)[3], atteint un résultat similaire en combinant l'approche basée force avec une approche de type *recuit simulé*, faisant évoluer les positions des nœuds au fur de cycles de refroidissement et de réchauffage du système – elle est particulièrement utile pour des réseaux excédant le million de nœuds.

La plupart des logiciels implémentent plusieurs de ces variantes, qui se distinguent non seulement par leur résultat graphique mais aussi par leur vitesse d'exécution. Les implémentations les plus connues sont par exemple « *spring-embedded* » dans Cytoscape, « Yifan-Hu » dans Gephi, « Fruchterman-Reingold » dans iGraph, « GEM » dans GUESS, ou encore « *OpenOrd* » dans Gephi. Le résultat dépend bien sûr aussi de la taille et de la structure du réseau analysé. Dans l'idéal, l'application permet de choisir parmi plusieurs algorithmes, et de paramétrer ces derniers, à la recherche de la meilleure solution.

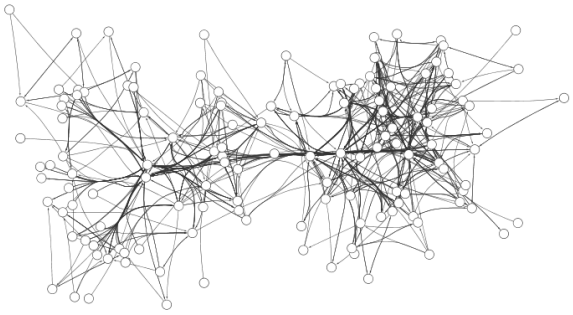


Figure 5 : Exemple d'un agencement *spring-embedded* avec Cytoscape. Les arrêtes ont été rassemblées par *edge bending*.

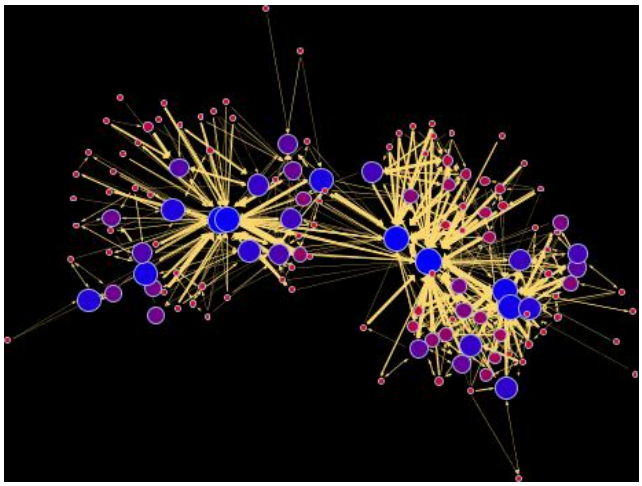


Figure 6 : Agencement *Generalized Expectation-Maximization*, produit à partir de données test avec GUESS.

[3] S. Martin, W. M. Brown, R. Klavans, and K. Boyack, "OpenOrd: An Open-Source Toolbox for Large Graph Layout," SPIE Conference on Visualization and Data Analysis (VDA), 2011

2.1.1.2 Approches matricielles et graphes auto-organisés

Les approches basées sur la force, comme nous l'avons vu, passent par une métaphore physique où les nœuds sont traités comme des particules et les arrêtes comme des ressorts. Une autre classe de méthodes d'agencement prend une approche très différente, en s'appuyant directement sur l'analyse de la matrice de flux d'adjacence (Tableau 1). Tels sont les *agencements spectraux* (Koren 2005) ou les agencements par *multidimensional scaling* (Figure 7a). Leur objectif est d'adapter la proximité des nœuds à l'intensité de leurs liens. Elles y parviennent en trouvant un plan de projection optimal de la matrice d'adjacence. Les deux dimensions du plan (ses coordonnées) correspondent aux vecteurs propres de la matrice d'adjacence ou d'une de ses transformations (*e.g.* laplacien). La méthode convient aux graphes fortement interconnectés, pondérés, et non-directionnels.

Des résultats similaires peuvent être obtenus par la méthodologie très différente des graphes auto-organisés. L'agencement le plus cité est la carte auto-organisée inversée (*inverted self-organised map* – ISOM) (Bernd 1998), produit selon une logique basée sur les réseaux neuronaux similaire aux cartes auto-organisées (Kohonen 1982).

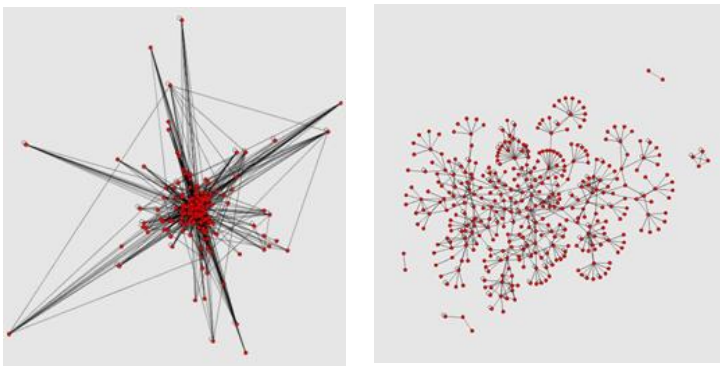


Figure 7 : a) agencement MDS b) agencement basé sur la force *spring embedded*. Produit avec GUESS (3.6).

2.1.1.3 Les coordonnées porteuses de signification

Lorsqu'on a affaire à des réseaux complexes de grande taille, les algorithmes matriciels ou basés sur la force produisent souvent des structures compactes, sphériques, qui demeurent difficiles à lire. Les positions des nœuds, en outre, sont uniquement interprétables en termes relatifs. On distingue les centres et les périphéries, ainsi que les clusters de nœuds voisins, mais la position absolue des nœuds dans l'espace du plan importe peu. La signification de la représentation ne varie guère avec le changement d'orientation, pas même lorsque l'on inverse en miroir l'image du réseau. La position centrale ou périphérique, enfin, n'est pas attribuée a priori : les nœuds les plus reliés se *retrouvent* au centre du réseau par un jeu de forces d'attraction et de répulsion avec d'autres nœuds, ou par le résultat d'une projection de la matrice d'adjacence.

Une série d'approches très différentes consiste à inscrire le réseau dans un espace a priori signifiant, c'est-à-dire dans un espace dont les coordonnées sont à priori dotées d'une interprétation spécifique. On place les nœuds dans ce système de coordonnées en fonction des attributs dont ils sont porteurs. Un nœud possédant une centralité élevée sera par exemple placé en haut de la carte, et en bas s'il a une centralité faible.

2.1.1.3.1 Agencements circulaires

L'exemple le plus simple d'un agencement dans un espace à coordonnées a priori signifiantes est l'agencement en cercle. On ordonne les nœuds sur le pourtour d'un tracé de cercle, de manière croissante ou décroissante, basée sur la valeur d'un attribut.

Ce faisant, on essaie de choisir l'attribut dont la prise en compte permettra de réduire le nombre de croisements et/ou de réduire la distance entre les nœuds fortement liés. Les arrêtes sont dessinées en lignes ou en arcs. Certaines méthodes utilisent plusieurs niveaux de cercles (0). Dans des cercles à un seul niveau, on retient notamment les agencements suivants, dans le sens horaire ou antihoraire :

- Les cercles où les nœuds se suivent en fonction de leur *degré* (Figure 8).
- Les cercles où les nœuds se suivent en fonction de la valeur d'un attribut de nœud.
- Le rassemblement en cercles distincts de nœuds présentant une valeur d'attribut identique ou similaire (Figure 9).

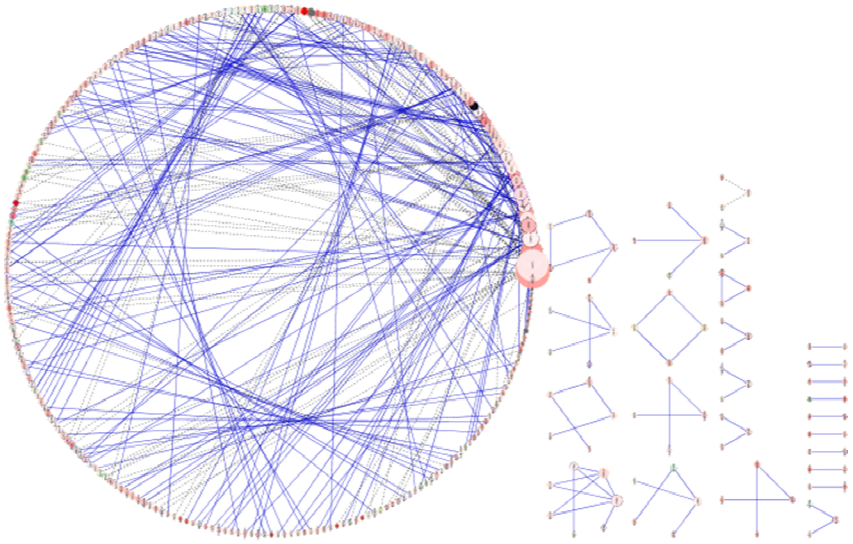


Figure 8 : Agencement circulaire par degré, avec 26 composantes. Généré avec Cytoscape (3.1).

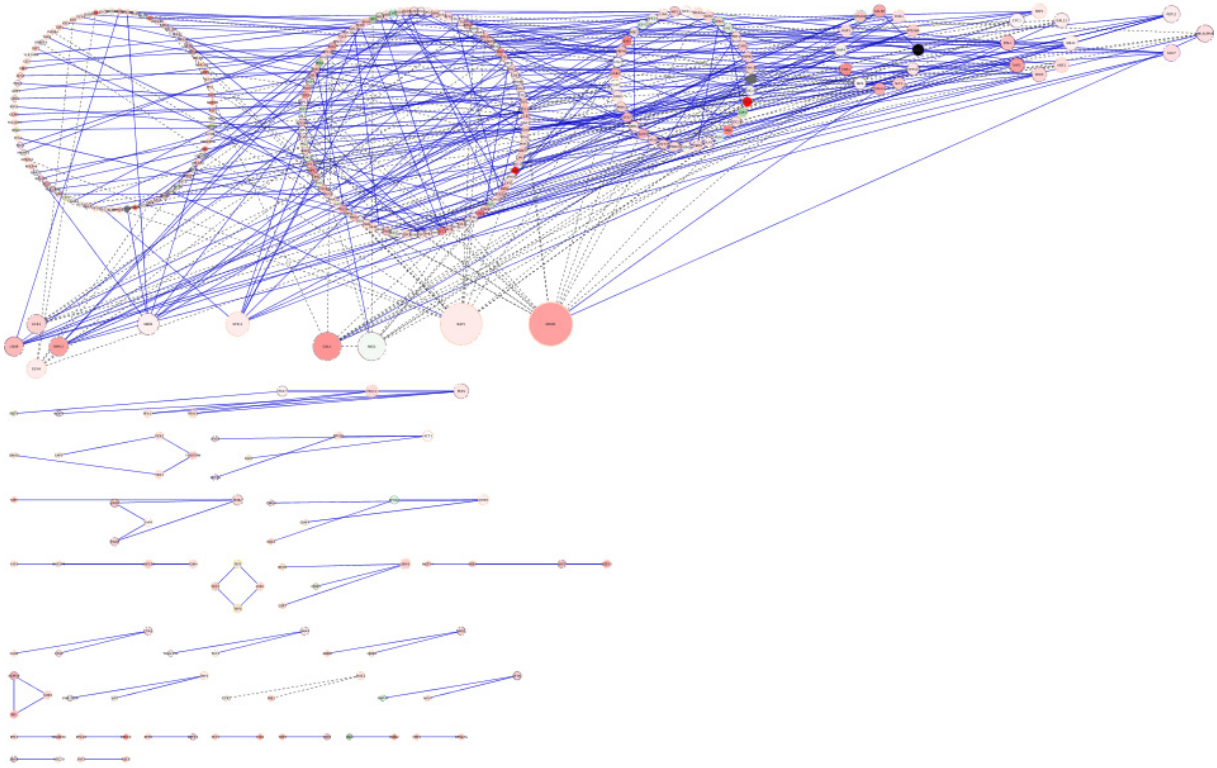


Figure 9 : Regroupement en cercles de nœuds de degré identique. Généré avec Cytoscape (3.1).

2.1.1.3.1.1 Chord diagram

Le *chord diagram* (Figure 2) est une variante spécifique des distributions circulaires convenant particulièrement aux graphes orientés. Les nœuds n'y sont pas représentés sous formes de cercles individuels comme dans la Figure 9, mais sous forme d'arcs de cercles colorés. En termes de logiciels, voir par exemple 4.2.3.2.

2.1.1.3.2 Approches hiérarchiques

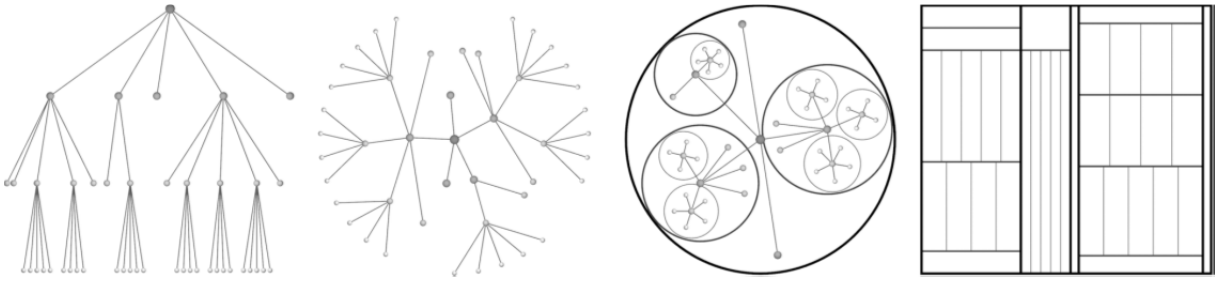


Figure 10 : Approches hiérarchiques classiques : a) arbre radial [*rooted tree*] b) arbre radial [*radial tree*], c) *balloon tree*, d) *treemap*. (Figure tirée de Holten 2006)

Pour donner un exemple d'agencement hiérarchique, il faut imaginer un réseau de mobilité présentant des hubs centraux reliés à longue distance ; à chaque hub s'articule un réseau local, constitué parfois lui aussi de hubs, donnant sur des réseaux encore plus locaux. La hiérarchie du réseau vient du fait que les réseaux locaux ne sont pas reliés entre eux, induisant des passages obligés par les hubs de niveau supérieur. Un réseau hiérarchique au sens strict possède en outre un nœud principal que la théorie des graphes désigne comme la racine (*root*) d'un arbre. Les représentations hié-

rarchiques imposent de choisir un tel nœud, ou du moins un ensemble de nœuds ; les algorithmes de visualisation de ce type construisent l'arbre à partir du nœud central.

On peut distinguer quatre visualisations principales : l'arbre à racine, l'arbre radial, l'arbre ballon, et le *treemap*.

*2.1.1.3.2.1 Arbre à racine (*rooted tree*)*

Plusieurs algorithmes permettent de représenter un réseau sous forme des arbres à racine (Figure 10a). Leur objectif est notamment de restituer clairement le niveau hiérarchique de chaque nœud et d'éviter les croisements d'arrêtes. Fréquemment utilisé, Reingold-Tilford (1981) maximise la densité et la symétrie par rapport à l'axe central – horizontal ou vertical – traversant le nœud racine. L'algorithme de Moen (1990) maximise également la densité en compactant les nœuds, tout en permettant de varier la taille et la forme des nœuds ; il ne traite pas les nœuds comme des points mais comme des polygones, et tient donc compte, par exemple, du diamètre que l'on donne au cercles qui représentent les nœuds.

2.1.1.3.2.1.1 Agencements orthogonaux

Les agencements orthogonaux ne sont pas limités à des graphes hiérarchiques mais ils leur conviennent particulièrement bien, dans la mesure où ils permettent de structurer les nœuds en couches. Un agencement orthogonal impose aux arrêtes d'un graphe de se présenter soit comme des lignes horizontales ou comme des lignes verticales. Originairement conçues pour générer le design de circuits imprimés, les méthodes permettant de les générer ont été adaptées à la représentation de graphes. Cet héritage en fait une approche qui exclut tout croisement d'arrêtes (dans un circuit imprimé, un tel croisement impliquerait un faux contact).

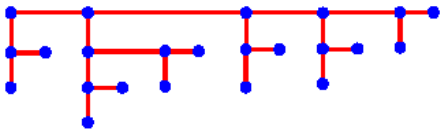


Figure 11 : Arborescence privilégiant une distribution orthogonale pour améliorer la lisibilité.

2.1.1.3.2.2 Arbre radial

L'agencement hiérarchique peut aussi être représenté sous forme d'un agencement circulaire à plusieurs niveaux (*radial tree*). L'approche permet une organisation hiérarchique à partir d'un nœud central, choisi par algorithme (en mettant par exemple les nœuds du plus haut degré ou de la plus haute centralité au premier niveau) ou par l'utilisateur. Les voisins de degré *1* au degré *n* sont ajoutés sur des cercles extérieurs. L'avantage de présenter cette hiérarchie de manière circulaire tient du fait que, notamment lorsqu'on a affaire à un grand réseau, le nombre de voisins augmente à chaque nouveau degré de voisinage. Pour cartographier un réseau de très grande taille où le nombre de nœud par degré de voisinage augmente de manière exponentielle, il est d'ailleurs fréquent d'utiliser un cercle hyperbolique[4], pratiquement consultable seulement dans des interfaces interactives.

[4] Contrairement à un cercle dans l'espace euclidien, dont la circonférence augmente de manière linéaire avec le diamètre, la circonférence d'un cercle dans l'espace hyperbolique augmente de manière exponentielle. *cf.* Lamping et al. 1994. Des logiciels interactifs pouvant être construits par exemple à l'aide de certaines bibliothèques Java sont nécessaires pour visualiser une hyper-arborescence inscrite dans un hyper-cercle.

La représentation dite de *treepie* ou *sunburst* consiste à alléger la présentation visuelle en réduisant les nœuds à des arcs de cercle (Figure 13).

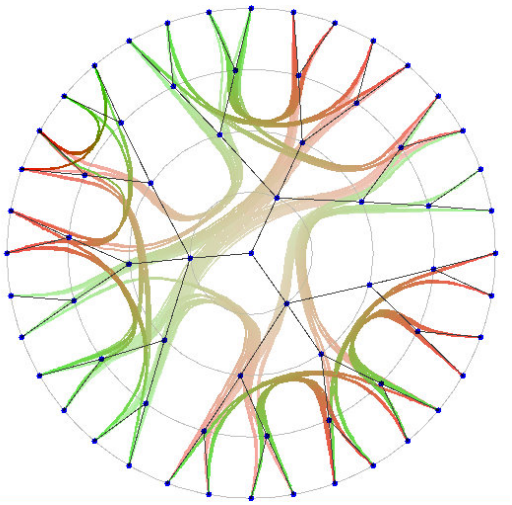


Figure 12 : Agencement circulaire (*radial tree*) à quatre niveaux de voisinage par rapport au nœud central. Produit avec Edge Bundles.

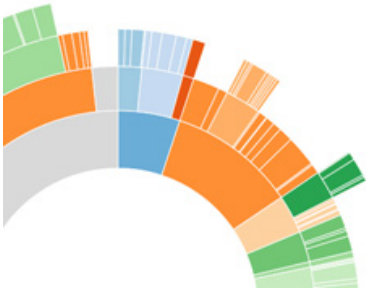


Figure 13 : Agencement circulaire à quatre niveaux de voisinage par rapport à un ensemble de nœuds centraux, sous forme graphique dite de *sunburst*.

2.1.1.3.2.3 Balloon tree

La représentation dite "*balloon tree*" ressemble fortement à l'arbre radial, mais a pour particularité de contenir les enfants d'un nœud (*i.e.* les nœuds de niveau hiérarchique immédiatement subordonné) dans des cercles de diamètre diminuant à chaque niveau, de manière à ce que les sous-réseaux ne se croisent pas. Chaque sous-réseau est à son tour présenté comme un arbre radial.

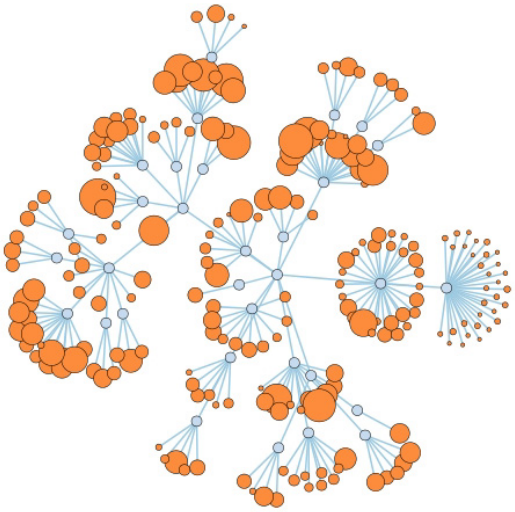


Figure 14 : Agencement en arbre radial.[5]

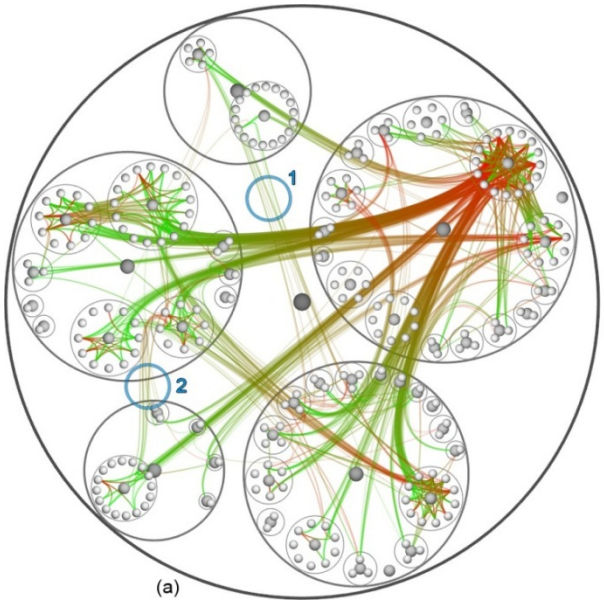


Figure 15 : Flux cartographiés sur un agencement de flux (Holten 2006).

2.1.1.3.2.4 Treemap

A priori, le graphisme d'un *treemap* (Figure 16), fréquemment utilisé pour représenter des emboîtements hiérarchiques, n'est pas prévu pour représenter des données de flux. Il permet néanmoins d'optimiser l'agencement des nœuds dans une grille rectangulaire (Figure 17). L'ajout graphique d'arrêtes à cet agencement permet dès lors de fournir une vision d'un réseau de flux de pertinence équivalente aux autres approches hiérarchiques.

[5] http://mbostock.github.com/d3/talk/20111116/force-collapsible.html

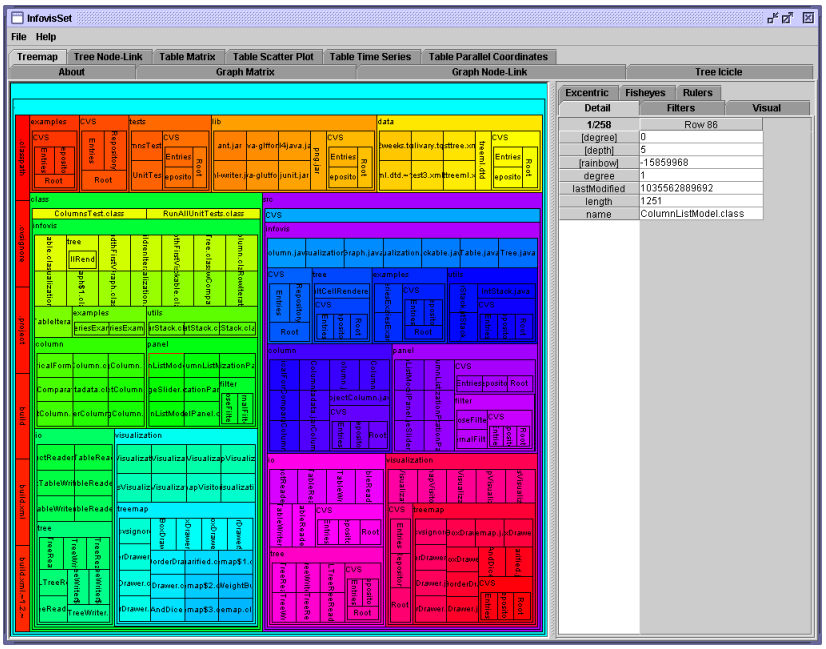


Figure 16 : *Tree-map* généré avec InfoVis.

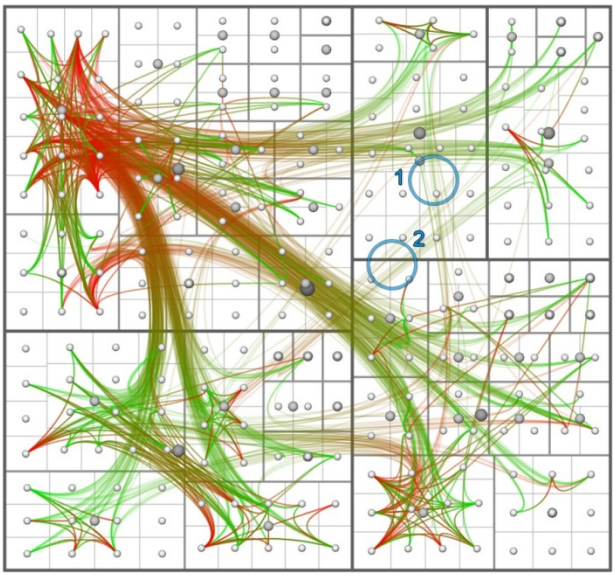


Figure 17 Flux cartographiés sur un agencement de flux (Holten 2006).

2.1.1.3.2.5 Graphe en couches successives (layered graph)

L'approche graphique de l'arbre à racine, comme celle des autres agencements hiérarchiques, convient le mieux à représenter des flux dont l'orientation serait indifférente ; on peut s'intéresser, par exemple, au nombre de personnes qui voyagent quotidiennement entre A et B en mettant entre parenthèses la direction du déplacement. Souvent, cependant, on s'y intéresse.

L'algorithme dit *Sugiyama-style* ou *layered graph drawing* (Sugiyama et al. 1981) présente dès lors une solution. La méthode consiste à ordonner d'abord les nœuds de manière à maximiser le nombre de liens procédant dans le même sens. Dans une seconde phase, les nœuds sont ré-agencés dans leurs couches de manière à limiter les croisements d'arrêtes.

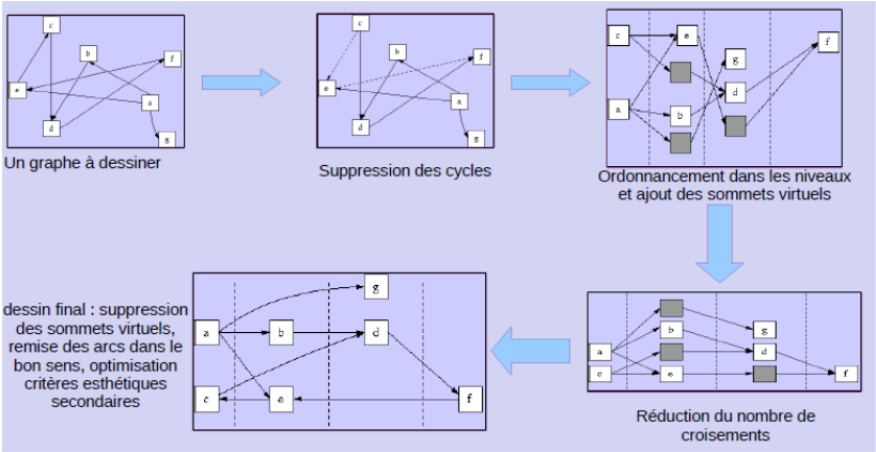


Figure 18 : Séquence d'agencement de réseau dans l'approche *layered-graph*.

Une autre forme de graphe de flux en couches successives est offerte par le diagramme de Sankey (Figure 19). La largeur des arrêtes est proportionnelle au flux représenté. La carte figurative des pertes successives en hommes de l'armée française dans la campagne de Russie 1812-1813 établie par Minard est un exemple répandu de diagramme de Sankey ; sa particularité est d'être bidirectionnelle, distinguant le déplacement à l'aller et au retour.

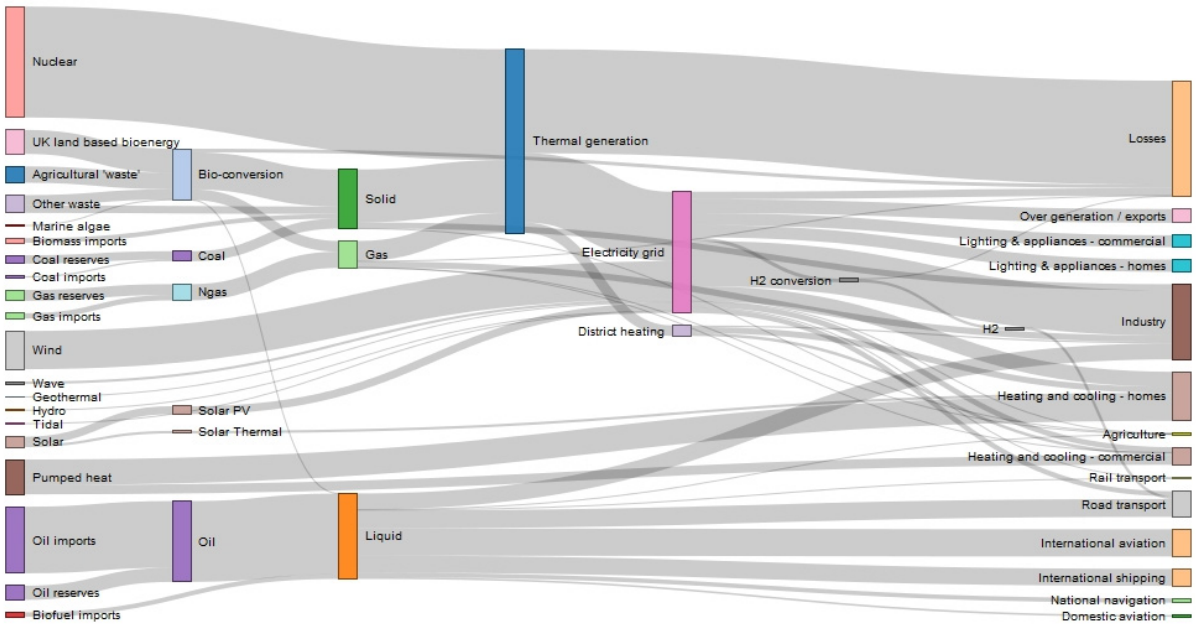


Figure 19 : Exemple de diagramme de Sankey construit dans Data Driven Documents (4.2.3.2)[6].

2.1.1.3.2.6 Hive plots

Les agencements hiérarchiques ordonnent les nœuds selon un seul critère. Les hive-plots permettent d'en prendre en compte plusieurs. Ils demandent pour cela que la population des nœuds soit divisée en plusieurs sous-populations. Dans une logique de flux notamment, on peut distinguer par exemple les nœuds majoritairement sortants (à degré-sortant fort et à degré-entrant faible), les nœuds de passage (degré-sortant et degré-entrant moyens) et les nœuds d'arrivée (à degré-sortant

[6] Source : http://bost.ocks.org/mike/sankey/

faible et à degré-entrant fort)[7]. On obtient ainsi trois groupes de nœuds que l'on répartit sur trois axes. Sur chaque axe, les nœuds sont ordonnés de manière hiérarchique, en fonction, par exemple, de leur centralité ou de la taille de la population qu'ils regroupent. Leurs liens sont finalement figurés sous forme de courbes de Bezier.

L'avantage de l'approche consiste systématiser l'espace de représentation et à rendre ainsi les réseaux aisément comparables.

Peu de logiciels - dont HiveR dans R (4.2.1.2) et HiveGraph dans le cloud - implémentent pour l'heure les représentations en hive-plots.

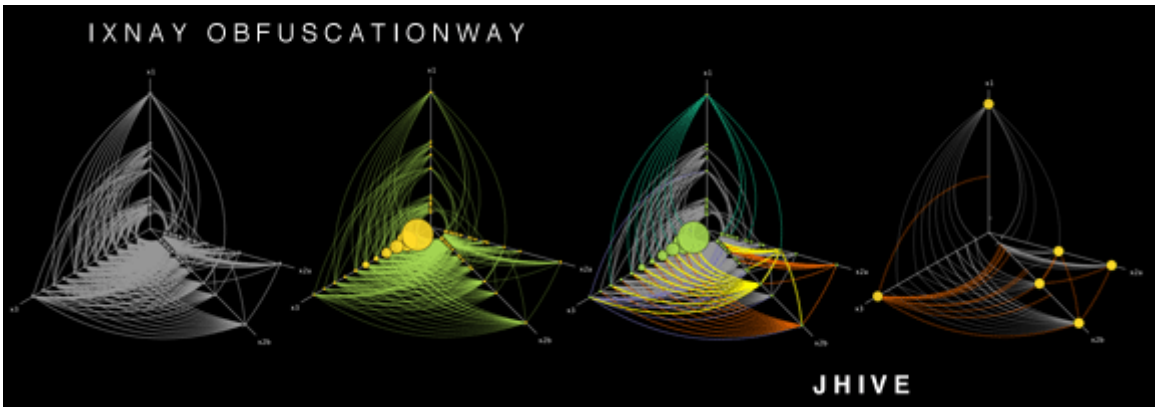


Figure 20 : Autres exemples de Hive plots.

2.1.1.4 Agencements 3D

Les approches basées force, les approches matricielles ou les agencements circulaires peuvent également être réalisés en trois dimensions. Peu utile dans la perspective d'une production de cartes imprimées, cette option permet néanmoins une visualisation interactive. Elle peut s'avérer intéressante notamment lorsqu'il s'agit de traiter des grands réseaux complexes. Des logiciels comme Pajek et Navigator permettent des visualisations en trois dimensions.

2.1.1.5 Amélioration manuelle de l'agencement

Notons enfin que, malgré les avancées du traitement automatique des agencements de graphes, un post-traitement manuel est souvent indispensable, notamment pour attirer l'attention du lecteur de la carte sur des aspects du réseau ne pouvant pas être codifiés de manière systématique dans un tableau de données. Il arrive aussi que la meilleure représentation d'un réseau ne puisse être obtenue qu'en combinant des approches pour plusieurs sous-parties – en représentant, par exemple une partie du réseau selon une logique basée force et une autre en arbre radial.

Il est utile, de ce point de vue, que les logiciels de cartographie permettent de tels traitements manuels, voir qu'ils permettent d'appliquer des algorithmes d'agencement distincts à des sous-ensembles sélectionnés de nœuds.

[7] Voir **Erreur ! Source du renvoi introuvable.** pour la notion de degré de nœud.

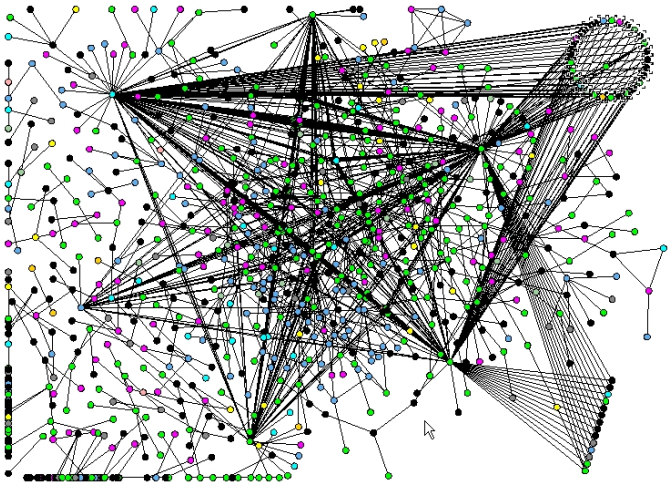


Figure 21 : Réarrangement manuel des nœuds dans le logiciel NAViGaTOR.

2.1.1.6 Diagrammes de matrices origines-destinations

En sortant tout à fait de la représentation de nœuds et d'arrêtes en termes de points et de lignes, enfin, on peut décider d'observer directement une matrice de flux, en conservant la liste des nœuds à l'horizontale et à la verticale. Les cases de la matrice peuvent être colorées pour restituer, par exemple, le poids des liens ; *heatmap* est le terme souvent dédié à ce type de représentation. Des logiciels comme NetMiner proposent de générer ce type de représentations, désignées diagrammes de matrices (Figure 22). Un logiciel comme Excel, néanmoins, suffit pour en produire, à l'aide de la mise en forme conditionnelle des cellules (Figure 38).

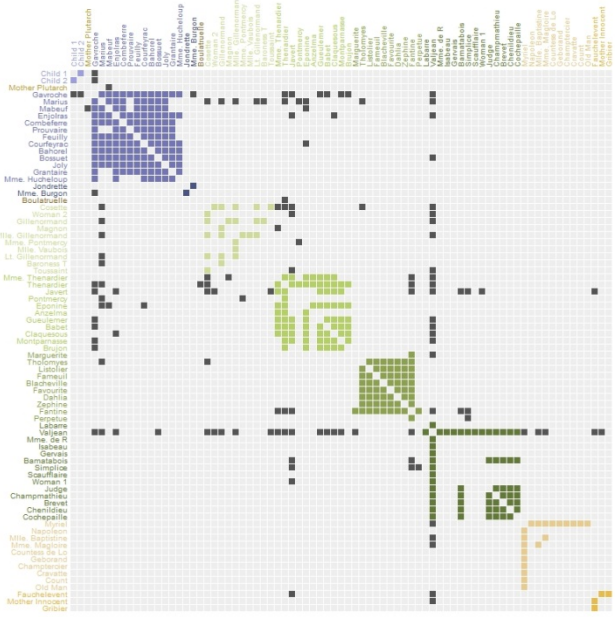


Figure 22 : Diagramme de matrice d'échange généré avec Data Driven Documents.

2.1.2 Généralisation

Les cartes de flux, comme d'autres, gagnent à être généralisées. Cette généralisation passe d'abord par des opérations de sélection, permettant de réduire le nombre d'arrêtes et de nœuds représentés. Souvent, celle-ci se fait en fonction du degré des nœuds (2.4.2) et du poids des arrêtes (2.4.1), mais d'autres aspects quantitatifs ou qualitatifs peuvent être choisis pour jouer un rôle.

Les nœuds et les arrêtes peuvent en outre être fusionnés, ou rassemblées en faisceau, ce qui limite également leur nombre et allège le visuel de la carte. Les logiciels se consacrent peu au rassemblement des nœuds – cet aspect fournirait peut-être un projet de recherche – quoique des réseaux hiérarchiques permettent, de facto, de procéder à de telles opérations. Généralement, la cartographie de flux actuels préfère conserver des nœuds séparés pour leur attribuer des couleurs distinctives suite à une analyse des communautés (2.4.4).

Le rassemblement des arrêtes, par contre, est un souci fréquemment abordé par les cartomaticiens, qui le considèrent comme le principal mode de généralisation d'une carte de flux: il limite en effet le croisement des arrêtes qui est la principale source de bruit visuel (Holten 2006). Plusieurs algorithmes existent. La plupart des logiciels les intègrent, en donnant pourtant rarement la référence de l'algorithme utilisé.

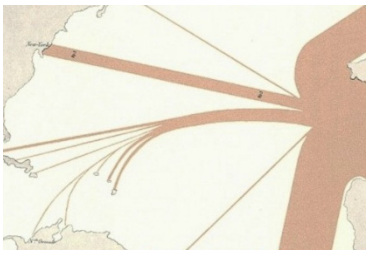


Figure 23 : Rassemblement des arrêtes (Minard).

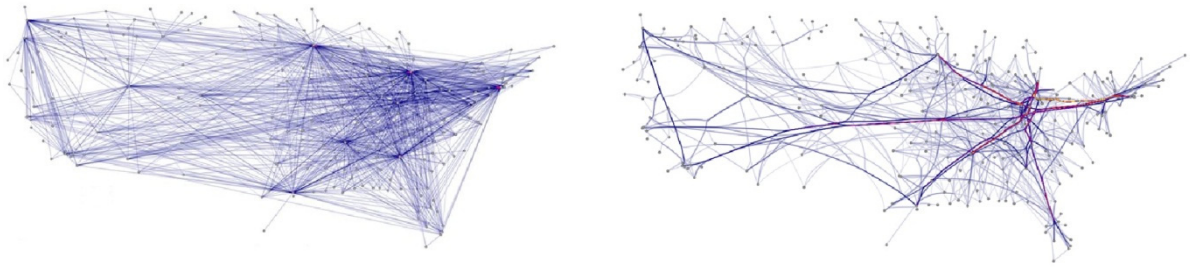


Figure 24 Rassemblement des arrêtes : réseau de vols continentaux, USA[8].

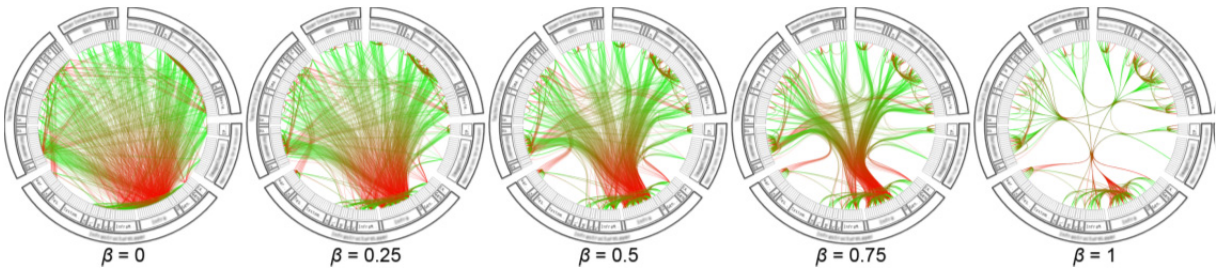


Figure 25 : Plusieurs niveaux de rassemblement d'arrêtes dans un agencement circulaire (Holten 2006). Généré à l'aide de bibliothèques Infovis (4.2.5.6).

[8] http://blog.visualmotive.com/2009/graph-visualization-edge-bundling/

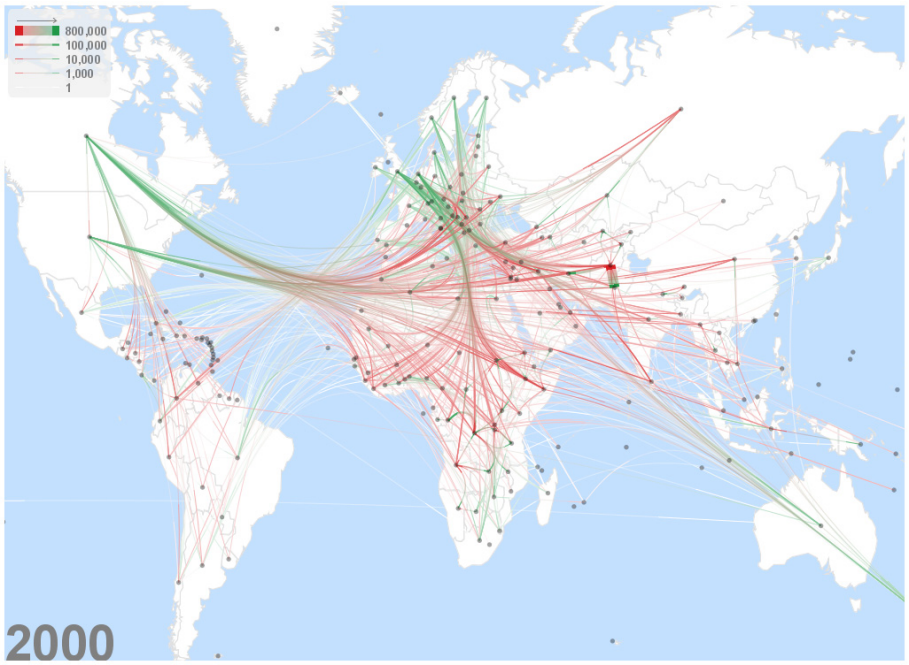


Figure 26 : *Edge bending* sur fond de carte territorial (2.2.1) dans jFlowmap (3.9).

2.1.3 Symbolisation

Pour recourir à la symbolisation cartographique, il est nécessaire que la taille, la couleur et la forme des nœuds puissent varier en fonction de leurs attributs. Pouvoir donner une forme arbitraire à chaque noud individuel facilite les opérations de symbolisation. La possibilité d'étiqueter ces éléments par leurs attributs (noms, propriétés statistiques etc.) fait également partie des opérations de symbolisation requise pour la bonne interprétation de la carte.

Figure 27 : Exemple de variation de forme de nœuds, dans Data Driven Documents. Dans ce réseau servant de pur exemple, on a donné aux nœuds le contour topographique des États-Unis[9].

Il en va de même pour les arrêtes, à qui l'on demandera de varier d'épaisseur, de couleur, de direction et de forme. Les arrêtes de graphes peuvent se présenter en lignes droites, en lignes brisées et en courbes (Figure 28). Idéalement, un logiciel cartographique permet de générer au moins la

[9] http://mbostock.github.com/d3/talk/20111018/force-states.html

première et la dernière de ces variantes. Des flèches devraient pouvoir leur être ajoutées, notamment lorsqu'il s'agit de cartographier des flux. Certains algorithmes, enfin, proposent d'adapter le contour des arrêtes en fonction d'autres formes présentes sur la carte (Figure 29).

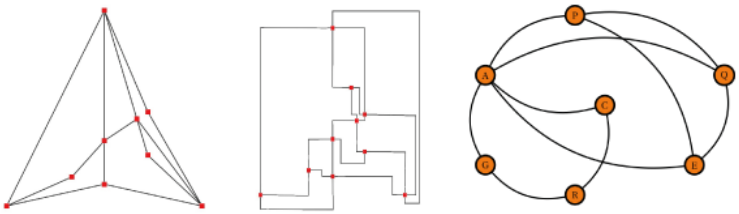


Figure 28 : Arrêtes droites, brisées et courbes.

Figure 29 : Distorsion géographique et « edge routing » dans la carte de flux de Minard.

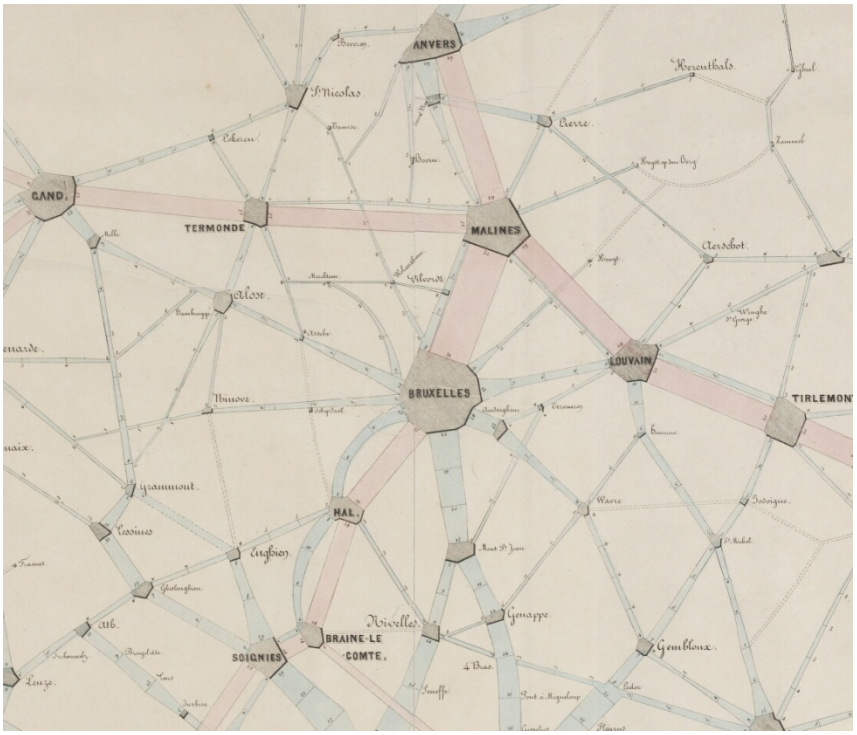


Figure 30 : Étiquetage des nœuds et des arrêtes, adaptation de la forme des nœuds à la forme topographique des villes, et des arrêtes à la fréquence des routes, dans la carte d'Alphonse Belpaire (1843)[10].

[10] Carte du mouvement des transports en Belgique, pendant l'année 1843, indiquant l'importance comparative de la circulation sur les voies de communication par terre.

2.1.4 Interactivité et post-traitement graphique

L' « interactivité » est une particularité de la cartographie informatique. Elle consiste à monter certains éléments et d'en cacher d'autres en réagissant à des évènements (choix dans des menus, *mouse-overs* etc.). Dans ce sens, l'interactivité est un procédé cartographique à mi-chemin entre la généralisation et la sélection. On peut aussi agrandir des éléments, modifier la mise en classe de couleurs etc. Les modes d'interactivité sont multiples. La plupart des logiciels de cartographie de flux ci-recensés offrent un certain degré d'interactivité, permettant de mieux étudier ou de modifier les cartes traitées. L'interaction peut également faciliter le post-traitement graphique d'une carte. Voir les figures ci-dessous ainsi que la Figure 69.

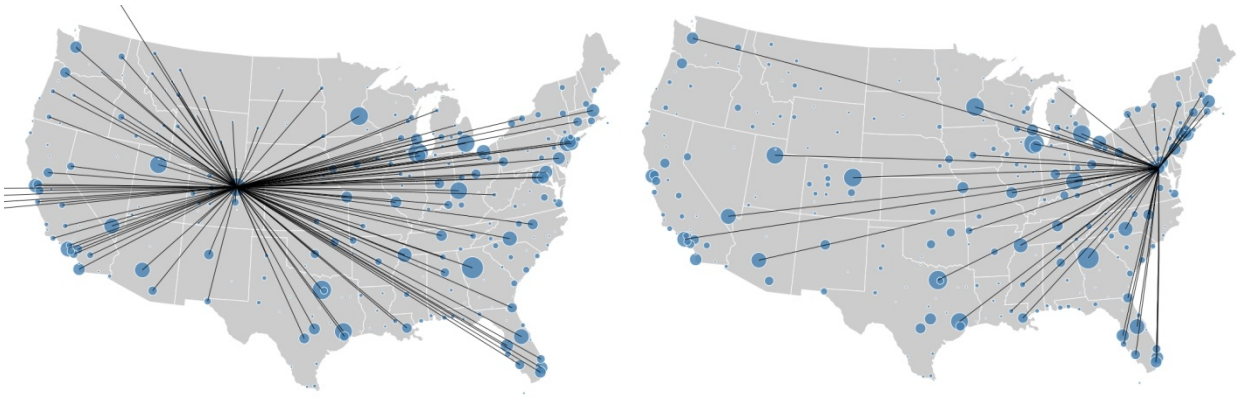


Figure 31 : Exemple d'interactivité avec la bibliothèque JavaScript « *Data Driven Douments* »[11]. *Mouseover* sur un nœud du Colorado à gauche, sur un nœud de la Côte atlantique à droite.

Figure 32 : Exemple d'interactivité avec la bibliothèque JavaScript « *Data Driven Douments* » : agrandissement fish-eye des nœuds oranges à gauche, des nœuds violets à droite.

2.2 Les fonds de carte

Quel que soit l'algorithme de positionnement et les techniques de généralisation, la cartographie de flux se confine souvent à la représentation de points et de lignes dans un plan, quel que soit

[11] http://mbostock.github.com/d3/talk/20111116/airports.html

d'ailleurs la forme que l'on donne aux points et aux lignes. La *coloration* des nœuds permet de révéler des informations structurelles sous-jacentes, tels des clusters de voisins ou des clusters de coappartenance à des unités territoriales, mais elle n'en donne pas une lisibilité optimale, surtout en présence d'une multiplicité de clusters.

L'ajout d'un fond de carte polygonal peut présenter une solution de ce point de vue.

2.2.1 Le fond de carte territorial

Le fond de carte polygonal le plus usuel est celui de la carte topographique classique, avec ses frontières administratives, fonctionnelles, ou physiques. Il permet à l'utilisateur d'identifier clairement l'appartenance territoriale des nœuds, localisés par leur géocode (latitude-longitude ou x-y projeté). Son usage présuppose néanmoins de renoncer à toute forme d'optimisation de l'agencement des nœuds (2.1.1), leur position étant d'emblée donnée et conservée.

Étant donné qu'un grand nombre de logiciels de représentation de réseaux ont été développés en dehors du champ de la géographie (en biologie moléculaire notamment), la superposition des nœuds à des cartes topographies demeure une option rare, disponible dans quelques logiciels seulement, dont GUESS ou Flowmap.

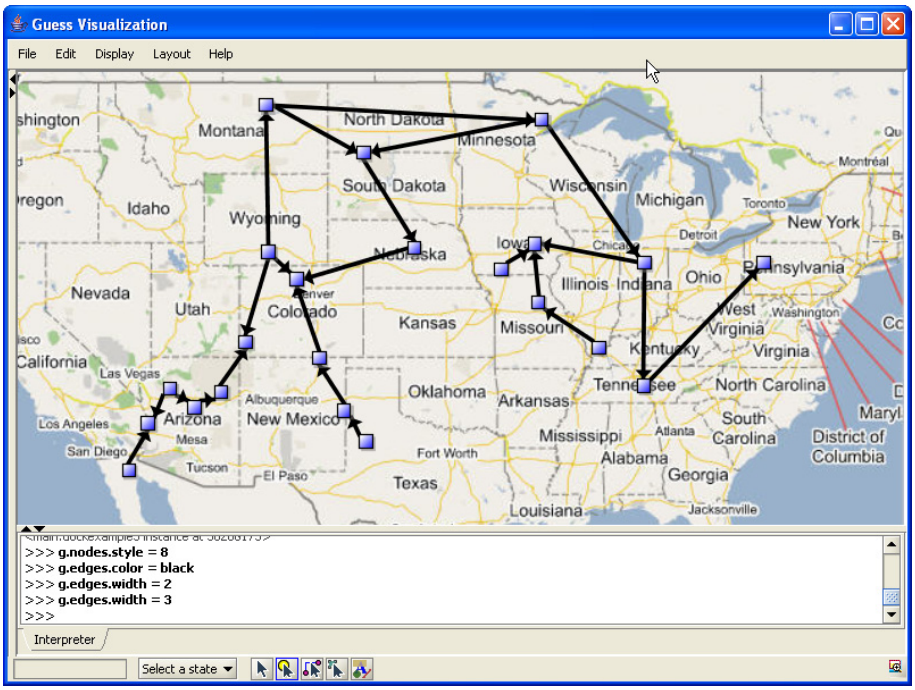


Figure 33 : Superposition d'un graphe de flux à une carte territoriale classique, dans GUESS (3.6).

2.2.2 GMap

Une seconde manière d'obtenir un fond de carte polygonal consiste à demander au logiciel de produire un tel fond de carte ad hoc. Plusieurs algorithmes ont récemment été développés à cette fin. GMap en fait partie (Gansner et al. 2009). L'algorithme prend dans un premier temps appui sur une approche d'agencement basée sur la force (2.1.1.1) ou matricielle (2.1.1.2) pour distribuer les nœuds dans le plan. Une deuxième phase identifie les clusters. Une troisième phase consiste à circonscrire les nœuds à l'aide d'une technique combinée de placement de diagrammes de Voronoi et de point aléatoires. Le résultat présente l'ensemble du réseau comme une île avec des zones clairement délimitées.

Figure 34 : Gmap du réseau de relations économiques entre pays. (Source: Gansner 2009)

2.2.3 Enveloppe convexe

Un second exemple, nettement plus simple, de fond de carte ad hoc est offert par la méthode dite d'enveloppe convexe (*convex hull*) (Schneideman 1992). Elle consiste à inscrire l'ensemble des nœuds partageant une propriété donnée – au choix du cartographe ou d'un algorithme – dans un polygone aussi petit que possible capable de les contenir tous. GUESS est capable de produire des enveloppes convexes. Le plugin Biopax[12] pour Cytoscape produit des résultats similaires, avec des enveloppes légèrement plus grandes que le minimum nécessaire. Il permet en outre de fusionner des nœuds à fins de généralisation.

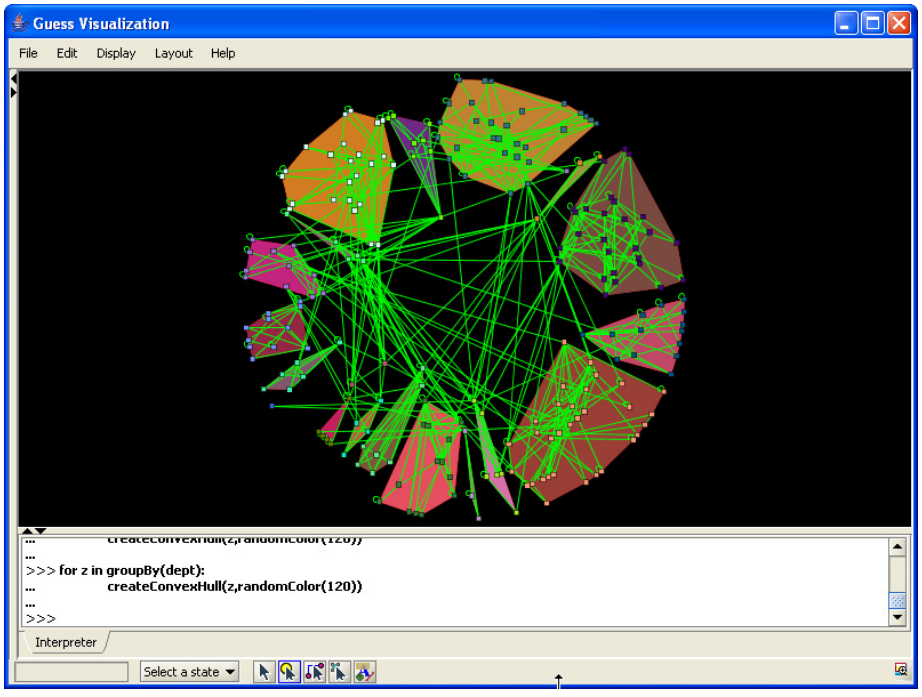


Figure 35 : Génération d'une *enveloppe convexe* dans le logiciel GUESS. Ici regroupement de nœuds appartenant au même département.

[12] http://wiki.cytoscape.org/BioPax

2.3 Les cartes de diffusion

La propriété principale - et la limite - d'un graphe tient dans le fait que ce dernier constitue un espace discret. Les flux peuvent également être conçus comme des phénomènes continus. La diffusion d'une substance chimique, la répartition d'une foule de spectateurs dans un festival, la migration d'oiseaux sont des exemples de phénomènes de flux qui gagnent à être conceptualisés de cette manière.

2.3.1 Isolignes

La représentation cartographique la plus utilisée pour ce type de phénomènes est la carte isoplèthe (*contour chart*) formée de lignes reliant des valeurs identiques. Les lignes dénotent en principe le temps qu'une particule (humaine, animale, physique) du flux observé met à se déplacer entre deux isolignes consécutives. Les espaces entre les isolignes peuvent être colorés selon un dégradé de couleurs afin d'augmenter la lisibilité du sens du processus de diffusion. Les logiciels SIG courants (ArcGIS, Mapinfo, Quantum GIS etc.) conviennent le mieux à ce type de représentation.

Notons cependant qu'un type de traitement graphique similaire peut également être employé dans une approche de visualisation réseau de type graphe discret. Les isolignes dénotent dès lors d'autres valeurs (p. ex. la centralité des nœuds) et le procédé peut être utilisé pour générer des fonds de carte ad hoc améliorant la lisibilité du graphe (*cf*. 2.2). Le logiciel NetMiner, notamment, propose ce type de traitement (3.3, Figure 48).

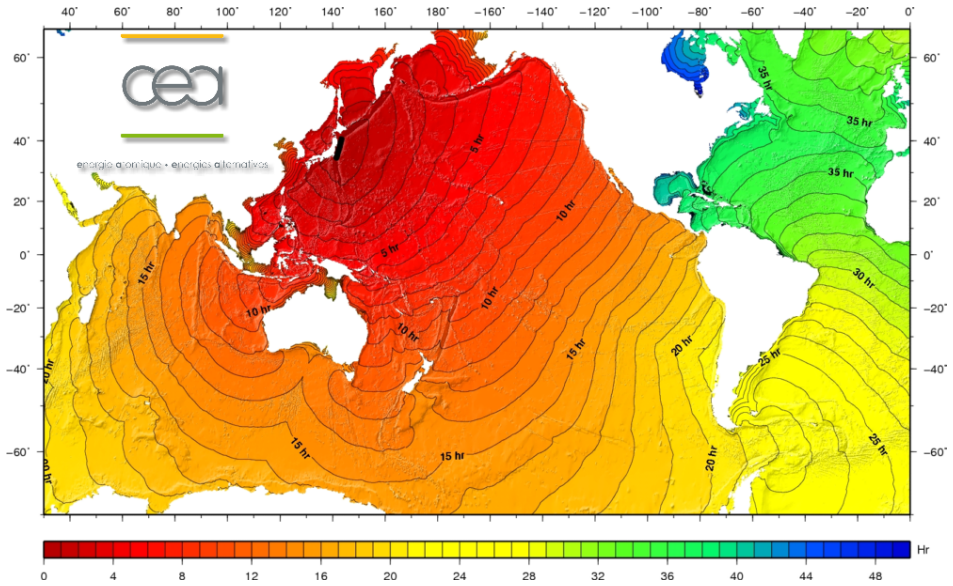


Figure 36 : Carte de diffusion basée sur les isolignes[13].

2.3.2 Cartes de vecteurs et cartes de potentiels

Deux autres manières fréquentes de représenter des flux sont les *champs de vecteurs* et les *champs de potentiel*. Les premiers consistent à monter l'orientation du flux par des flèches orientées dont la longueur peut varier en fonction de la vitesse du déplacement d'agents de flux modélisés en tant que particules (humaines, animales, physiques…). Les secondes identifient les origines et les

[13] http://www-dase.cea.fr/actu/dossiers_scientifiques/2011-03-11/images/temps_de_propagation.png

destinations finales des flux par gamme de couleurs. Les logiciels SIG courants (ArcGIS, Mapinfo, Quantum GIS etc.) conviennent le mieux à ce type de représentation.

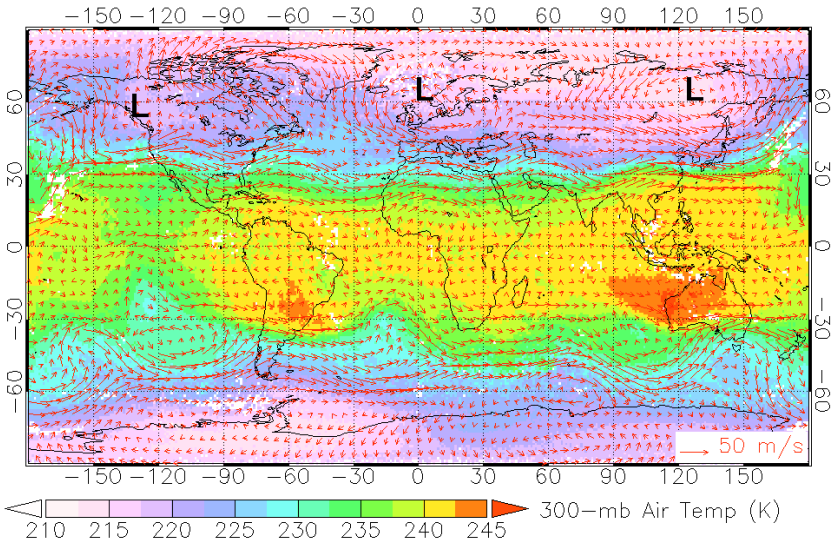


Figure 37 : Carte de flux d'air (vent) combinant un champ de vecteurs (orientation des flèches) et un champ de potentiel (couleurs).

2.4 Les métriques du réseau

La qualité des cartes de réseau produites par un logiciel est largement dépendante de sa capacité à adapter la forme, la taille et la couleur des nœuds et des arrêtes aux *métriques* du réseau représenté. Par métriques, on entend des propriétés synthétiques des nœuds et des arrêtes, résultant d'une analyse du réseau. L'usage d'un logiciel de cartographie de réseau, quel qu'il soit, présuppose une bonne connaissance de ces métriques. Mais les logiciels à privilégier sont ceux capables d'en mesurer un nombre suffisant, et permettant ainsi de tester plusieurs variantes de visualisation.

Tout logiciel de cartographie réseau devrait offrir au moins une mesure de *poids*, de *connectivité* et de *centralité*, ainsi qu'une analyse des *communautés*.

2.4.1 Les poids

La notion de poids s'applique aux arrêtes d'un réseau. Elle révèle l'intensité du lien entre les deux nœuds reliés par l'arrête. En reprenant l'exemple du Tableau 1, le lien D-A possède un poids de 12 et le lien D-B un poids de 95.

Le poids des liens est souvent pris en compte dans les algorithmes d'agencement. Dans les approches basées sur la force comme dans les approches matricielles, on s'efforce de rapprocher les nœuds liées par des arrêtes à poids fort.

Du point de vue graphique, l'épaisseur des arrêtes sera souvent proportionnelle à leur poids.

2.4.2 Mesures de connectivité

Les mesures de connectivité s'appliquent aux nœuds. Elles désignent l'intensité de connexion d'un nœud à ses voisins et à l'ensemble du réseau.

Le nombre d'arrêtes qui relient un nœud à d'autres détermine son *degré*. Dans un graphe directionnel, on peut distinguer entre *degré entrant (in-degree)*, et *degré sortant (out-degree)*. Le nœud B de l'exemple (Tableau 1) possède un degré sortant de 2 et un degré entrant de 3. Le nœud C possède

un degré entrant zéro, et un degré sortant 1. Le degré d'un nœud joue un rôle important dans les opérations de sélection cartographique. On peut par exemple éliminer les nœuds de degré inférieur à une valeur seuil pour simplifier la représentation du réseau.

La mesure du degré peut être limitée aux connexions entrantes (degré entrant), ou sortantes (degré sortant). Certaines mesures de connectivité prennent en compte non seulement le degré du nœud considéré mais aussi les degrés des nœuds voisins. L'indicateur *neighborhood connectivity*, par exemple, donne le degré moyen de l'ensemble des voisins d'un nœud

2.4.3 Mesures de centralité

Le concept de centralité renvoie à la position d'un nœud ou d'une arrête au sein d'un réseau. De nombreuses mesures de centralité invoquent la notion de cheminement (*shortest path*). On considère, pour l'appliquer, l'ensemble des chemins de longueur minimale reliant toutes les paires de nœuds. Certains nœuds individuels s'avèrent dès lors comme des lieux de passage de nombreux cheminements, alors que d'autres n'en « subissent » qu'un faible nombre. Les premiers sont considérés comme plus centraux que les seconds.

Diverses variantes de mesure de centralité existent. Le *stress* d'un nœud est, simplement, le nombre de *shortest paths* qui passent par lui. La *betwenness centrality* rapporte ce nombre au nombre de *shortest paths* total ; elle révèle notamment des nœuds connecteurs entre deux communautés de flux distinctes. La *closeness centrality* (ou *average shortest path length*) mesure la longueur moyenne des *shortest paths* d'un nœud à tous les autres nœuds du réseau. L'excentricité, faible pour les nœuds centraux et forte pour les nœuds périphériques, mesure la longueur maximale non-infinie des *shortest paths* d'un nœud à tous les autres nœuds du réseau.

Employées plus fréquemment sur les nœuds, les mesures de centralité peuvent également être appliquées aux arrêtes. En termes cartographiques, elles sont le plus fréquemment représentées en faisant varier la largeur des cercles ou l'épaisseur des arrêtes. Du point des flux, la centralité d'un nœud, calculée à partir de la structure d'un réseau, a valeur prédictive sur l'intensité empirique du passage.

2.4.4 L'analyse des communautés

L'analyse des communautés – ou, par variation terminologique, des clusters – consiste à grouper les nœuds que la connectivité réciproque (intragroupe) distingue du reste du réseau (connectivité intergroupe).

De nombreux algorithmes de détection de communautés existent : *leading eigenvector*, *walktrap*, *edge betweenness*… Certains permettent de préciser le nombre de communautés recherchées, d'autres sont capables de déterminer eux-mêmes ce nombre. Du point de vue cartographique, une couleur similaire sera souvent attribuée aux nœuds de même communauté.

Les résultats d'une analyse des communautés varient fortement en fonction de l'algorithme choisi. Il est important, à ce titre, qu'un logiciel de cartographie de flux en offre un large éventail. Ceci n'est pas le cas pour la majorité des solutions intégrées. Des bibliothèques de scripts comme iGraph pour R et Python facilitent toutefois fortement l'écriture d'algorithmes servant à l'analyse des communautés.

2.5 Données et formats d'échange

La capacité des logiciels se traduit aussi dans leur interopérabilité. Celle-ci est possible grâce aux formats d'échange, représentations informatiques de la structure d'un graphe, voire de sa mise en page cartographique. Les meilleurs formats d'échange sont ceux qui permettent de conserver toutes les avancées produites dans un logiciel, pour reprendre le traitement graphique dans un autre.

2.5.1 Tableau de flux

Une table de flux – aussi appelée matrice d'adjacence ou matrice de connexions dans la théorie des graphes –, conserve la liste des nœuds en lignes et en colonnes, ainsi que l'échange entre les nœuds de sortie et d'arrivée. L'exemple donné par le Tableau 1 est une telle matrice. Si elle est non-nulle, la diagonale représente les boucles (*loops*), c'est-à-dire le flux à l'intérieur de chaque nœud[14]. Si la matrice est symétrique des deux côtés de la diagonale, tous les allés impliquent des retours ; elle peut être traitée comme un graphe non-orienté. Si elle est asymétrique, il convient de la traiter comme un graphe orienté. Une matrice de flux peut être représentée à l'aide d'un diagramme matriciel (2.1.1.6 ; Figure 38).

La matrice de flux est un format de document dans lequel se présentent souvent les données issues d'une statistique de transport, de migrations ou de flux pendulaires. On s'attendrait à ce que les logiciels de visualisation possèdent la capacité de la transformer en graphe en vue de traitements ultérieurs, mais tel n'est souvent pas le cas. Un prétraitement à l'aide d'un logiciel mathématique, tel R ou Matlab s'avère donc nécessaire pour obtenir un graphe numérisé dans un autre format, exploitable par les autres logiciels.

Le grand désavantage d'un tableau de flux est de ne pas être en mesure de conserver des donnée autres que celles que nous venons d'évoquer : ni la disposition graphique des nœuds, ni les structures hiérarchiques, ni le type de lien qui relie deux nœuds. Des variables supplémentaires, rattachées aux nœuds et aux arrêtes, ne peuvent pas non plus être conservées. D'autres formats, plus complexes et également plus spécifiques à chaque logiciel, servent à ces fins.

[14] Une matrice de pendularité entre des communes, par exemple, peut présenter un tel cas de figure, lorsque la pendularité à l'intérieur d'une commune est considérée comme une boucle.

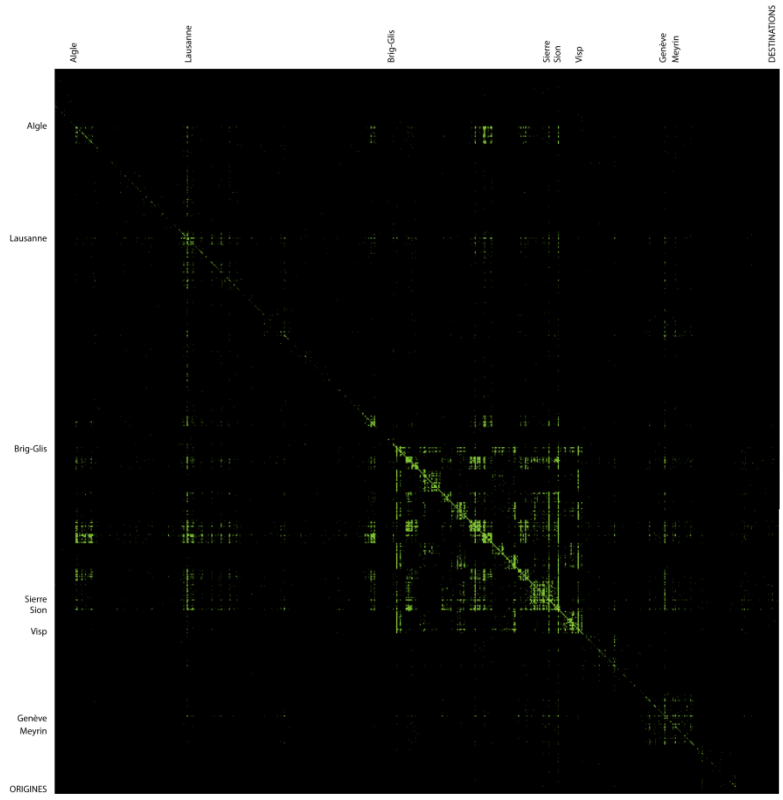


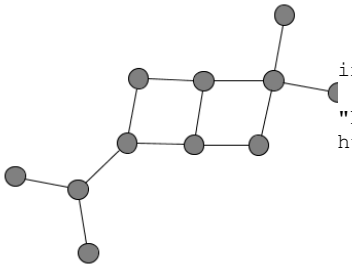
Figure 38 : Diagramme de matrice de flux pendulaire dans le canton du Valais.[15] Généré avec la mise en forme conditionnelle des cellules dans Excel.

2.5.2 De type XML

Un format type fréquemment utilisé pour stocker des données de flux sont des variantes du « langage de balisage extensible » XML. Le XML est basé sur SGML, langage de description à balise (reconnaissable par son usage des chevrons < > encadrant les balises) dont dérive également le HTML. La syntaxe du XML permet de définir différents espaces de noms, c'est-à-dire des langages avec chacun leur vocabulaire et leur grammaire. Dans le cas de cartographie de flux, nous retrouverons ainsi souvent des balises nœud `<node>` et des balises arrête `<edge>`, pouvant présenter divers degrés d'emboîtement, et divers attributs. Un nœud noté `<node id="n0"/>`, par exemple, contient l'attribut `id`, qui le dote d'un identifiant unique. Une arrête notée `<edge source="n0" target="n2"/>` identifie les nœuds reliés par les attributs `source` et `target`.

Ce type d'approche connaît de nombreux dialectes, dont les plus répandus sont GraphML, GML, GEFX (Graph Exchange XML Format), GXL (Graph eXchange Language). Les diverses variantes se distinguent par leur grammaire, par la possibilité qu'ils donnent, ou pas, de spécifier les modes de représentation, de faire des liens à des données extérieures, de spécifier la hiérarchie du réseau en arborescence ou en couches successives, de limiter l'existence des nœuds à une période de temps donnée, de stocker l'appartenance des nœuds à des clusters etc.

[15] Source : Ourednik 2012, http://www2.unil.ch/eatlasvs/wp/?page_id=238



```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns= "http://graphml.graphdrawing.org/xmlns"
    xmlns:xsi = "http://www.w3.org/2001/XMLSchema-
instance"
    xsi:schemaLocation =
"http://graphml.graphdrawing.org/xmlns
http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <graph id="G" edgedefault="undirected">
    <node id="n0"/>
    <node id="n1"/>
    <node id="n2"/>
    <node id="n3"/>
    <node id="n4"/>
    <node id="n5"/>
    <node id="n6"/>
    <node id="n7"/>
    <node id="n8"/>
    <node id="n9"/>
    <node id="n10"/>
    <edge source="n0" target="n2"/>
    <edge source="n1" target="n2"/>
    <edge source="n2" target="n3"/>
    <edge source="n3" target="n5"/>
    <edge source="n3" target="n4"/>
    <edge source="n4" target="n6"/>
    <edge source="n6" target="n5"/>
    <edge source="n5" target="n7"/>
    <edge source="n6" target="n8"/>
    <edge source="n8" target="n7"/>
    <edge source="n8" target="n9"/>
    <edge source="n8" target="n10"/>
  </graph>
</graphml>
```

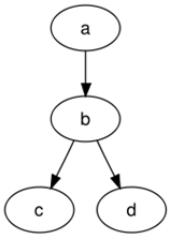
Figure 39 : Exemple de graphe enregistré sous un format d'échange XML : le GML

Malheureusement, au stade actuel du développement des logiciels, les formats xml existants sont encore trop nombreux et leurs grammaires standard ne sont pas toujours respectées. L'interopérabilité, sur l'ensemble des logiciels testés, laisse ainsi encore à désirer. Ce fait constitue un sérieux frein à l'adoption de solutions non-intégrées.

2.5.3 Autres types

Parmi les autres formats d'échange, on citera notamment les suivants, lisibles aussi bien par l'ordinateur qu'à l'œil humain :

- JSON, format générique de stockage de données structurées en flux de caractères aisément transmissibles par le Web, et particulièrement bien implémenté dans JavaScript. De par son caractère générique, JSON permet de stocker non seulement la structure du graphe mais aussi sa représentation graphique.
- NET, le format d'échange natif du logiciel Pajek (3.5), compris par de nombreux autres logiciels.
- DOT, langage de stockage de graphes natif du logiciel Graphviz (3.10).
- DIMACS, graphe stocké sous forme de lignes de texte.
- LEDA, un autre format textuel permettant de porter aussi des informations sur la disposition planaire du graphe.
- LibSea : format d'échange de l'association CAIDA, conçu pour l'exploitation dans le cadre du langage Java.




```
digraph graphname {
     a -> b -> c;
     b -> d;
 }
```

Figure 40 : Exemple de graphe décrit dans le langage DOT.

2.5.4 Formats GIS

L'utilisation de fonds de carte territoriaux (2.2.1) requiert de conserver les coordonnées de géo-localisation des nœuds traités. Le format XML le permet dans une certaine mesure. L'interopérabilité avec des solutions GIS comme ArcMap, MapInfo, ou Quantum GIS est cependant facilitée dans des logiciels permettant de traiter les formats ESRI *shape* ou Mapinfo *MIF/MID*[16]. Flowmap (3.8) fait partie de tels logiciels.

2.6 Formats de sortie graphique

Les cartes créées par les logiciels de cartographie de flux peuvent être exportées en deux catégories de formats : raster ou vectoriel. Les formats vectoriels (Illustrator, PostScript, PDF, SVG[17]) offrent la possibilité de post-traitement des cartes dans un logiciel d'édition graphique. Les logiciels permettant leur exportation devraient être privilégiés. La majorité de ces derniers conserve la direction des arrêtes lors de l'exportation (*cf*. Figure 41).

Les formats raster (png, gif, jpeg etc.) peuvent être utiles pour une exportation immédiate des cartes créées dans des sites web. Un post-traitement cartographique n'est cependant pas possible. Les logiciels qui n'exportent *que* dans ces formats sont à éviter pour un usage général.

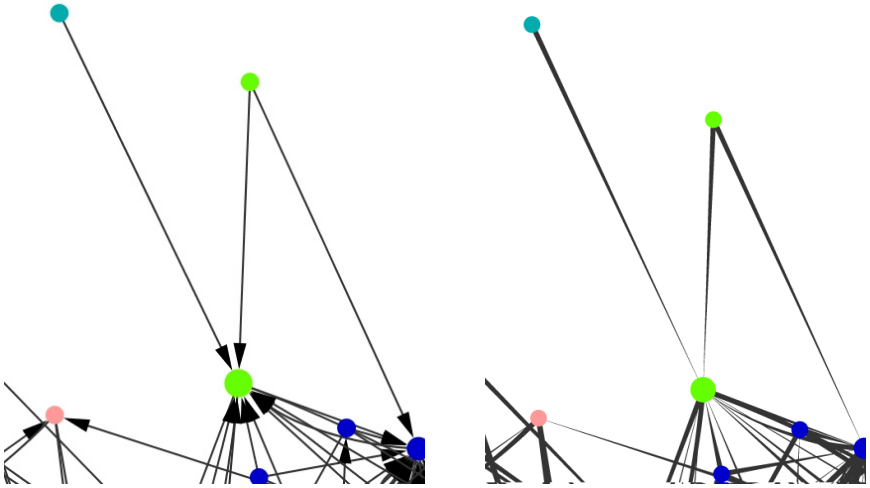


Figure 41 : Enregistrement de la *direction* des arrêtes dans le format vectoriel. Exemple avec Cytoscape (capture d'écran à gauche), export SVG, importation dans Illustrator, modification de la forme des arrêtes (droite).

16 http://reference.mapinfo.com/software/spatial_server/english/1_0/onprem/apiguide/LIM/source/UserDefinedSpatialDatabases/mapinfomifmidformat.html

17 http://www.w3.org/Graphics/SVG/

2.7 Accessibilité

Un dernier élément évalué, l'*accessibilité* des logiciels, se traduit surtout en trois paramètres : la plateforme (Windows, Mac Linux), la licence (gratuite ou payante), la compétence technique requise (programmeur ou utilisateur d'outils graphiques) et la langue (le logiciel est-il disponible en français).

3 Logiciels

Face au grand nombre de logiciels informatiques consacrés à la cartographie de réseaux, nous choisissons de présenter d'abord les plus complets et les plus génériques, avant de passer en revue quelques logiciels plus spécialisés, concentrés parfois sur un mode de représentation unique. Nous laissons de côté les logiciels de qualité seconde, dont les fonctionnalités sont redondantes avec celles d'autres logiciels déjà commentés. Nous avons cherché, en d'autres termes, à ne considérer que les solutions les plus complètes, ou alors des solutions qui proposent des traitements que d'autres ne permettent pas.

La critique de compétence des logiciels est focalisée sur les critères d'évaluation énumérés dans le chapitre 2, c'est-à-dire sur leur implémentation plus ou moins complète des transformations graphiques décrites, sur leur portabilité, sur leur capacité d'interagir avec d'autres logiciels et sur leur accessibilité. Pour les logiciels les mieux accessibles, il nous a été possible d'accomplir des tests basés sur un set de données portant sur la mobilité individuelle, et dont nous savons qu'ils possèdent une forte structure que les algorithmes mentionnés plus haut peuvent détecter. Nous nous y référerons par le terme « données test » dans les graphiques.

Comme remarque préalable, nous notons également que, quel que soit le type de logiciel, un grand nombre n'a pas été initialement conçu pour l'analyse de l'espace géographique. Un domaine de recherche particulièrement prolifique en termes de représentation visuelle de graphes est plutôt celui de la bio-informatique. Comme nous l'avons évoqué au début de cette section, néanmoins, les graphes sont des objets génériques, pouvant être soumis à des traitements similaires indépendamment des réalités qu'ils représentent. Certains des logiciels discutés ci-après ont ainsi déjà été appropriés pour la recherche géographique, et d'autres pourraient l'être.

3.1 Cytoscape

Cytoscape a été originairement conçu pour visualiser des réseaux d'interaction moléculaire dans le domaine de la biologie. Il est devenu une plateforme générique de visualisation et d'analyse de réseaux complexes. Des fonctionnalités additionnelles peuvent être implémentées par le moyen de plugins développés également par une large communauté d'utilisateurs. Cytoscape possède en outre une interface de programmation (API) Java, permettant le développement de plugins supplémentaires. La plupart des plugins existants, néanmoins, se focalisent sur les besoins de la recherche biologique.

Du point de vue graphique, le logiciel tel quel (*out of the box*) permet de varier la taille, la forme, la couleur la transparence des nœuds et des arrêtes individuellement, ou en fonction de variables choisies par l'utilisateur (Figure 43). Il est possible d'opter pour des *mapping* non-linéaires entre les tailles et les épaisseurs des éléments d'un côté, et des variables qui y sont rattachées de l'autre.

Cytoscape est capable de calculer automatiquement un large nombre de métriques réseau utiles pour la visualisation, telle la centralité des nœuds et des arrêtes ou la connectivité des nœuds.

Des filtres permettent également de sélectionner des sous-ensembles de nœuds et d'arrêtes en fonction de ces aspects. Cytoscape n'offre pas directement des algorithmes d'analyse de communautés mais des plugins (tels que « GLay ») le permettent, en faisant appel à des fonctions iGraph.

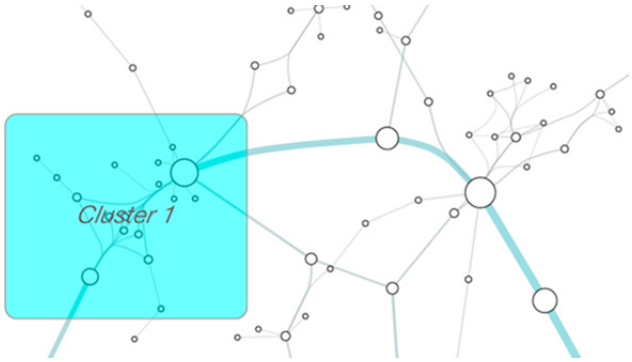


Figure 42 : Edge Bundling et annotations visuelles dans Cytoscape.

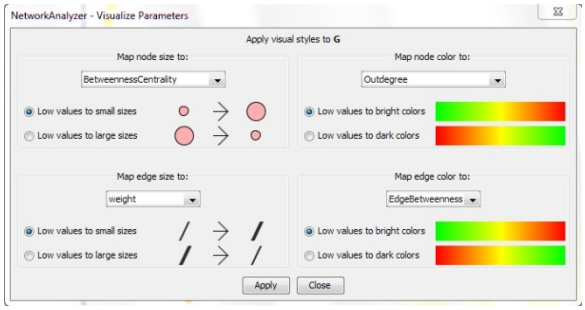


Figure 43 : Panneau de configuration de la représentation des nœuds et des arrêtes dans Cytoscape.

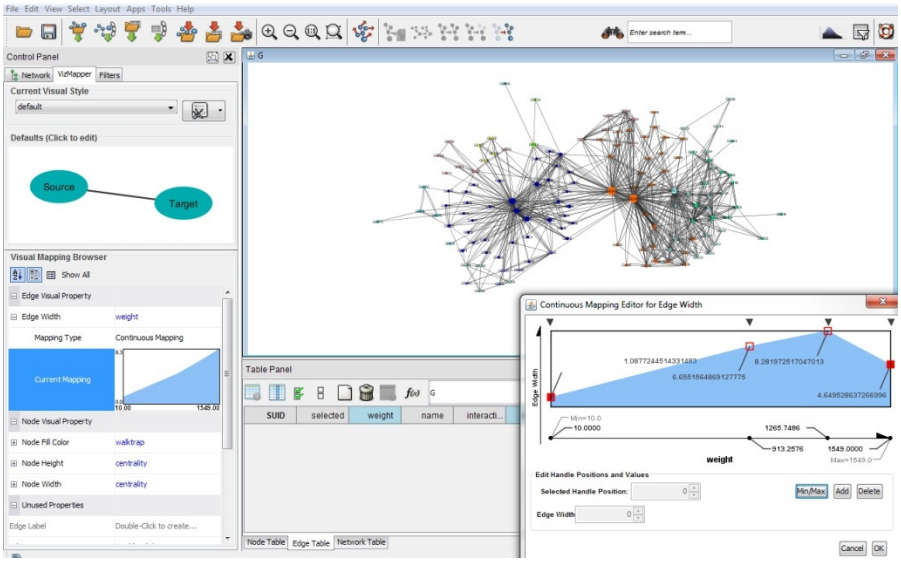


Figure 44 : Cytoscape. À noter le *mapping* non-linéaire entre épaisseur du trait des arrêtes et leurs poids (*cf.* 2.4.1).

Cytoscape propose une série de méthodes d'agencement des nœuds : circulaires, basées sur la force, cartes auto-organisées. Ces dernières peuvent être appliquées à l'ensemble du réseau ou seulement à une sélection d'éléments. Les nœuds individuels peuvent être déplacés avec la souris.

En termes de généralisation, il permet de rassembler les arrêtes (*edge bundling*), ainsi que de les plier (*edge bending*).

Il lui manque la capacité de traiter les données en termes de coordonnées géographiques ou de les superposer à un fond de carte autres (*e.g*. topographie).

Les nœuds et les arrêtes mis à part, le logiciel produit également des graphiques descriptifs de l'ensemble du réseau, tels des histogrammes des nombres de connexion par nœud (Figure 46) ou le scatterplot de la taille des clusters.

Cytoscape peut être commandé depuis le langage de programmation R à l'aide d'une extension[18]. Il est également disponible comme élément d'un assemblage CIShell (3.11).

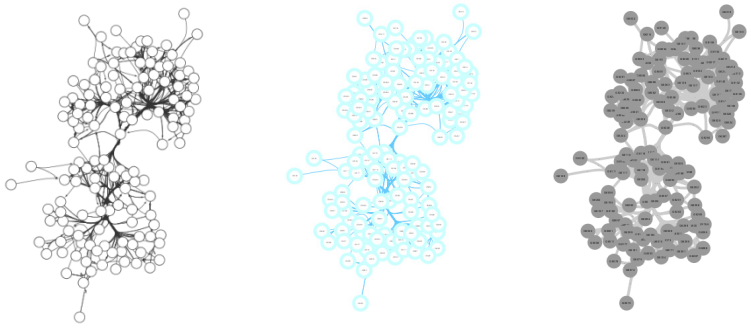


Figure 45 : Variation de modèles de mise en forme graphique dans Cytoscape.

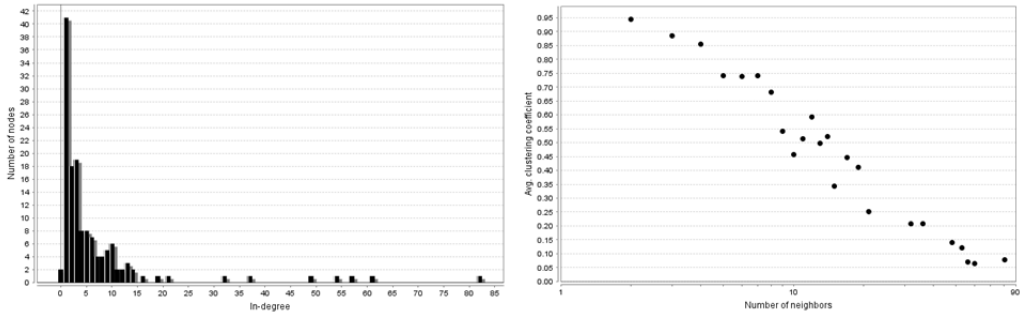


Figure 46 : a. Histogramme du nombre de connexions entrantes par nœud du réseau. b. Coefficient de clustering moyen.

Version testé : 3.0.0
Exportation graphique : vectoriel et raster
Plateforme : Windows 32 bit et 64 bit, Linux, Mac OS.
Licence: open source, GNU Lesser General Public License.
Plugins: oui, en grand nombre.
Formats d'échange : BioPAX, XGMML, GML, GraphML, Nested Network Files, OBO, PSI-MI, SBML.
Formats d'exportation graphique : vectoriels (svg) et raster, aussi bien pour les réseaux que pour les images.
Le format vectoriel *exporte la direction des arrêtes*.
Site : http://www.cytoscape.org/

3.2 Gephi

Gephi est un logiciel open-source basé Java largement utilisé aussi bien dans la recherche que dans le journalisme. Son interface graphique est rapide à prendre en main. Un grand nombre d'options de représentation est accessible directement.

[18] 0 ; *cf.* http://wiki.cytoscape.org/CytoscapeAndR

Le logiciel, même sans ajout de plugins, propose la plus vaste bibliothèque d'algorithmes d'agencement de nœuds, incluant de nombreuses approches basées sur la force (Fruchterman-Reingold, Yifuan-Hu etc.).

À l'instar de Cytoscape, Gephi calcule lui-même un nombre important de métriques réseau des nœuds et des arrêtes. Il incorpore en outre un éditeur de données permettant de modifier manuellement les variables liées à ces éléments.

Les opérations de *sélection* se font à l'aide de filtres. On peut, par exemple, sélectionner les arrêtes au poids inférieures à une valeur seuil, les nœuds de degré zéro (sans voisins), ou les nœuds représentant des villes de moins de x habitants.

Du point de vue graphique, ses possibilités sont très similaires à celles de Cytoscape.

La gestion des plugins peut être faite directement depuis l'intérieur du logiciel. De ce point de vue, relevons la disponibilité du plugin OpenOrd, implémentant l'un des rares algorithmes d'agencement basés-force capables de traiter des graphes dépassant le million de nœuds[19]. Aussi le plugin GeoLayout, permettant d'imposer aux nœuds un agencement géolocalisé, permettant de superposer les nœuds à des fonds de cartes territoriaux.

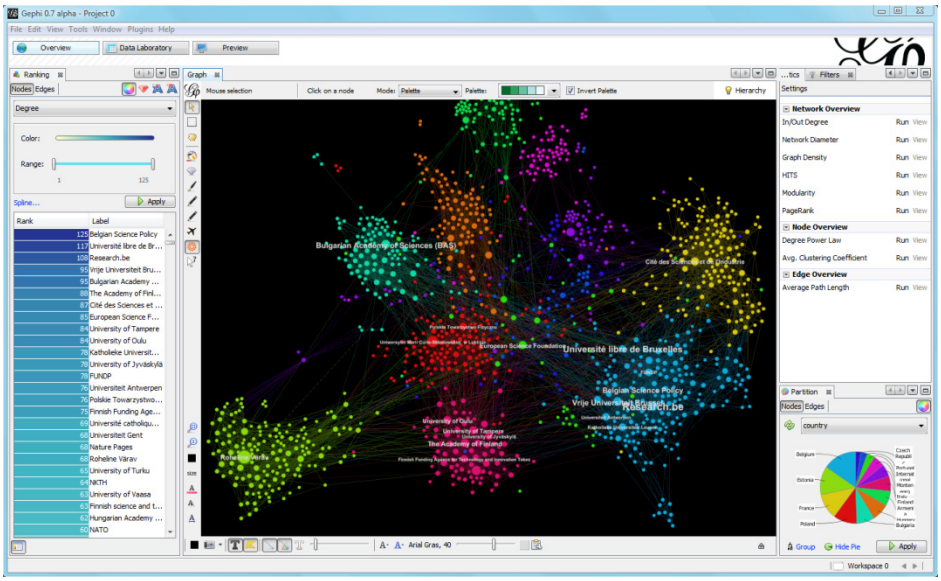


Figure 47 : Gephi.

Version testée : 0.8.2 beta
Développeur : Gephi Consortion, organisation non-profit française. Ses membres incluent SciencesPo, Linkfluence, WebAtlas et Quid.
Plateformes : Windows, Linux, Mac OS/X.
Licence : Open source
Langues : anglais, français et d'autres.
Formats d'échange : gephi, csv, edges, dl, dot, gv, gdf, gexf, gml, graphml, net, tlp, vna
Formats d'exportation graphique : vectoriel (pdf, svg) et raster. Les directions des arrêtes sont conservées.

3.3 NetMiner

NetMiner est un logiciel payant offrant un environnement intégré d'analyse et de visualisation de réseaux. Il est conçu pour représenter un large éventail de données réseau et flux modélisant des phénomènes sociaux, biologiques ou physiques.

[19] https://gephi.org/plugins/openord-layout/

Le logiciel est scriptable à l'aide de NetMiner Script, dérivé de Python, permettant d'automatiser les opérations d'analyse et de visualisation de réseau. L'option est utile dans la mesure où il s'agit d'appliquer le même traitement à une série de réseaux. Un générateur de script permet aux utilisateurs non-familiers avec le langage de conserver des procédures résultant du maniement de l'interface graphique. Quelques fonctionnalités peuvent être ajoutées à l'aide d'une bibliothèque de plugins mise à dis position par les développeurs.

Comme ses équivalents open-source, NetMiner permet de calculer une large série de métriques réseau. Les variables rattachées aux nœuds et aux arrêtes peuvent être visualisées à l'aide de variables graphiques standard (tailles et couleurs).

Il propose une série d'agencements de nœuds basés sur la force, circulaires, matriciels et hiérarchiques. Un avantage graphique du logiciel consiste en sa capacité à superposer les réseaux sur des cartes territoriales dans un environnement intégré (Figure 48a). Un élément graphique intéressant est la carte à lignes de contour, fond carte « ad hoc » basé sur les lissages du rendu visuel de variables métriques associées aux nœuds.

À part des cartes de réseau proprement dites, NetMiner génère également des tables et des diagrammes, pouvant être intégrés dans des rapports automatiques.

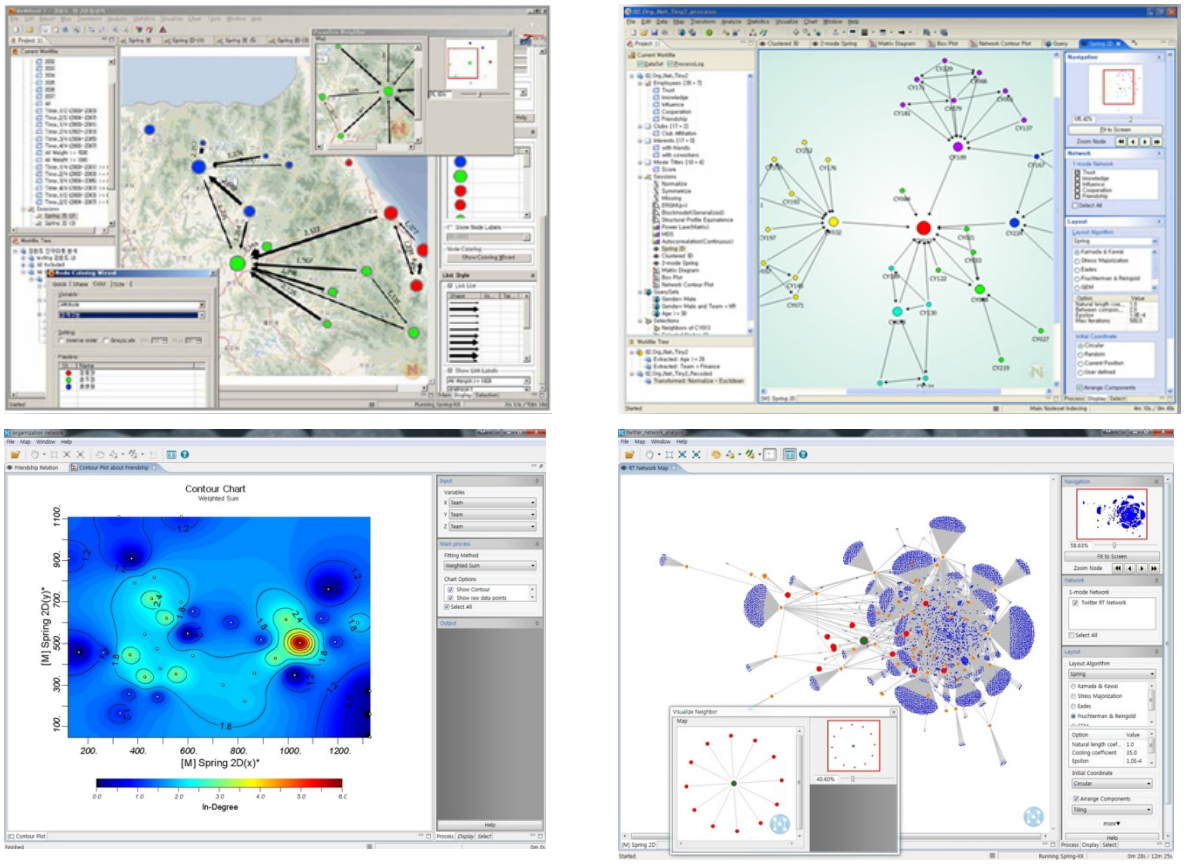


Figure 48 : Diverses visualisations dans NetMiner. a) sur Fond de carte territorial, b) réseau directionnel, c) lignes de contours, d) agencement circulaire hiérarchique.

Version commentée : Netminer 4
Développement : Cyram, Corée du Sud.
Format d'échange : NetMiner data file (.nmf). Peut importer et exporter en gml, pajek, dl et autres.
Licence : payant, prix sur demande.
Site web : http://www.netminer.com

3.4 Tulip

Tulip est, en principe, un *framework*[20] intégré de visualisation de graphes de taille pouvant aller jusqu'à un million d'éléments. Il se présente aussi comme une bibliothèque de programmation, conçue pour le design d'applications interactives.

Le logiciel offre une panoplie d'agencements – basés force, cartes auto-organisées, diagramme de matrices origine-destination,… – à laquelle s'ajoutent les visualisations 3D et des outils d'interaction. Les plugins installés d'office permettent de généraliser par rassemblement des arrêtes (*edge bundling*), et des nœuds (*clustering*).

L'application est en principe capable de calculer de nombreuses métriques réseau et d'adapter le visuel des éléments nœuds et arrêtes à ces derniers.

Tulip supporte des plugins communautaires, dont un grand nombre viennent livrés avec l'installation par défaut. D'autres peuvent être intégrés par l'intermédiaire d'une interface (déclaration des dépôts, sélection des plugins à installer).

Une API python permet d'exécuter l'application de manière programmatique.

Tulip est une application à usage relativement répandu. Les nouvelles versions testées (4.1 et 4.2), néanmoins, ne semblent pas encore documentées et présentent un problème d'installation empêchant l'exploitation du logiciel. Les problèmes rencontrés sont nettement plus importants que dans l'ensemble des autres logiciels testés. En l'état, malgré ses perspectives prometteuses et les preuves d'utilité fournies par le passé, Tulip ne peut pas être recommandé pour un usage public au stade actuel de son développement.

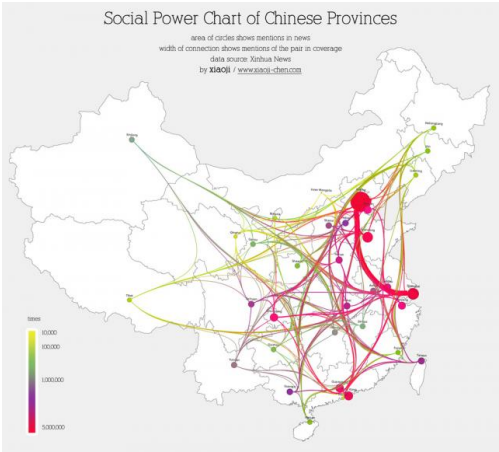


Figure 49 : Nœuds et relations dans Tulip[21].

Version testée : 4.1. et 4.2.
Développeur : Laboratoire Bordelais de Recherche en Informatique, Université de Bordeaux.
Plateforme : Windows, Linux, Mac OS.
Licence : Open source, GNU lesser general public license.
Fond de carte territorial : oui.
Langues : anglais
Formats d'échange : matrice d'adjacence, gexf, gml, net (pajek), dl (UCINET), tlp (format TULIP natif).
Formats d'exportation graphique : image et vectoriel.
Site web : http://tulip.labri.fr/

[20] En informatique, un framework est un kit de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel, en d'autres mots, son architecture.
[21] http://tulip.labri.fr/TulipDrupal/?q=node/1861

3.5 Pajek

Pajek (Nooy et al. 2005) est un logiciel à citer d'abord pour son apport historique à l'analyse et à la visualisation de larges graphes. Développé à partir de 1996, il est à l'origine du format d'échange pajek (.net), désormais standard dans de nombreux logiciels. Ses développeurs l'ont prévu comme un outil générique pour la représentation de réseaux de collaboration, d'interaction, de réseaux de molécules organiques et de récepteurs de protéines etc.

À l'instar des logiciels précédemment cités, Pajek est capable d'accomplir un nombre important d'analyses de réseau, de classifier les nœuds et les liens, de leurs associer des valeurs en termes de métriques réseau, et de visualiser ces résultats à l'aide de cartes 2D et 3D. Pajek se distingue par sa capacité d'exporter des visualisations 3D des réseaux sous divers formats. Les algorithmes d'agencement incluent des approches basées sur la force, des approches matricielles et des approches circulaires basiques.

À part les réseaux ordinaires (orientés, non-orientés, mixtes), Pajek se distingue en permettant l'analyse de réseaux *multi-relationnels*, capacité utile lorsque l'on veut analyser simultanément plusieurs types de flux reliant un ensemble de nœuds. Pajek est aussi capable d'analyser des *réseaux bimodaux*, comprenant deux ensembles disjoints de nœuds. Un exemple de réseau bi-modal pourrait être un ensemble de nœuds-stations fréquentés par un ensemble de nœuds-passagers. Il peut même analyser des graphes dynamiques, évoluant à travers le temps.

L'usage des capacités de Pajek demande une bonne connaissance des concepts de l'analyse de graphes.

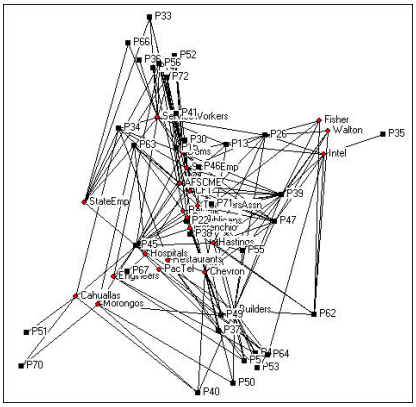


Figure 50 : Exemple de réseau bimodal reliant des donateurs et des initiatives politiques.[22]

Version testée : Pajek64 3.08.
Développeurs : Vladimir Batagelj, Andrej Mrvar, Matjaž Zaveršnik, Université de Ljubljana.
Plateforme : Windows 32 et 64 bit.
Fond de carte territorial : non.
Formats d'échange : format propriétaire Pajek seuelement.
Exportation graphique : vecteur et raster ; exportation en une grande quantité de formats, y compris des formats 3D.
Formats d'exportation graphique : vectoriel et raster.
Licence : gratuit.
Site : http://vlado.fmf.uni-lj.si/pub/networks/pajek/

[22] http://faculty.ucr.edu/~hanneman/nettext/C17_Two_mode.html

3.6 GUESS Graph Exploration System

L'intérêt spécifique de GUESS est de lier les représentations visuelles de graphes à une interface de programmation (ligne de commande), par l'intermédiaire de laquelle on intervient sur les propriétés visuelles et numériques du réseau étudié. Le langage utilisé, *Gython*, est un dérivé de Python (4.2.2). Ce contexte programmatique permet le développement de traitements spécifiques aux besoins de l'utilisateur. Des opérations algorithmiques personnalisées, par exemple, peuvent être appliquées sur toutes les propriétés des objets du graphe ; les propriétés et les méthodes des objets peuvent à leur tour être récupérées pour l'usage dans d'autres contextes. Des formats de sortie arbitraires peuvent être créés ou lus. Généralement parlant, l'approche d'emblée programmatique de GUESS offre une plus grande flexibilité. Les procédures peuvent en outre être stockées, ce qui permet d'appliquer le même traitement à plusieurs réseaux, de manière à garantir l'homogénéité de leur présentation visuelle.

Le revers de cet avantage est que de nombreuses fonctions de GUESS ne peuvent être accédées qu'à travers la ligne de commande. La ligne de commande est cependant liée à la représentation graphique de manière innovante. À titre d'exemple, passer la souris sur une liste de valeurs d'attributs surligne les nœuds correspondants dans la représentation visuelle).

Un nombre de métriques réseau peut être calculé directement dans le logiciel. Sur le plan de l'optimisation de l'agencement des nœuds, GUESS offre plusieurs algorithmes basés sur la force ou sur le positionnement multidimensionnel (MDS), auxquels on peut ajouter tout traitement spatial programmable. Il est possible d'apporter des annotations simples (texte, lignes, surfaces) à tous les éléments. Des enveloppes convexes peuvent être ajoutées de manière automatique. Le logiciel permet même la superposition de cartes réseau et de cartes topographiques.

Son défaut est une suspension du développement depuis 2008, ainsi qu'une documentation encore lacunaire.

GUESS est aussi intégré dans l'assemblage Network Workbench (3.11.1).

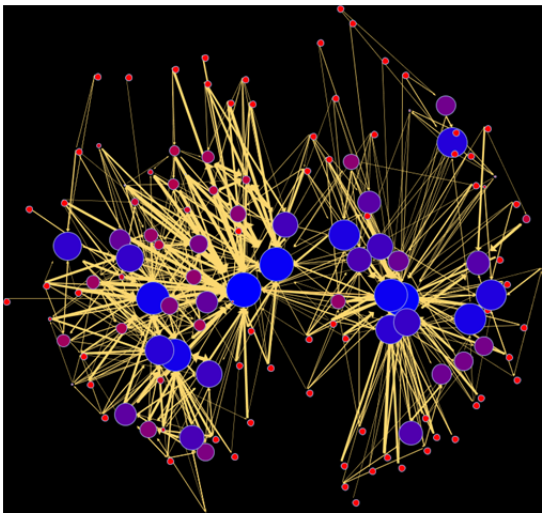


Figure 51 : Rendu des données test. Les tailles et les couleurs reprennent la *betweenness* des nœuds, l'épaisseur des traits le poids[23]. L'agencement est obtenu d'après l'algorithme basé force Kamada-Kawai.

[23] > g.colorize(Node.betweenness,red,blue) > g.resizeLinear(Node.betweenness,5,30) > resizeLinear(Edge.weight,0.1,3)

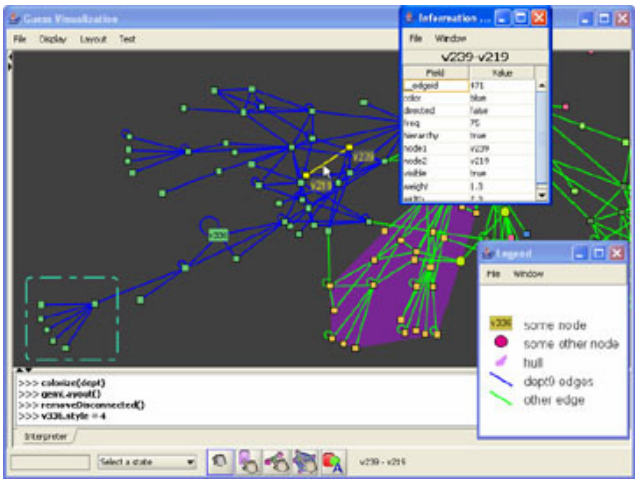


Figure 52 : Exploration et annotation manuelle des nœuds dans GUESS.

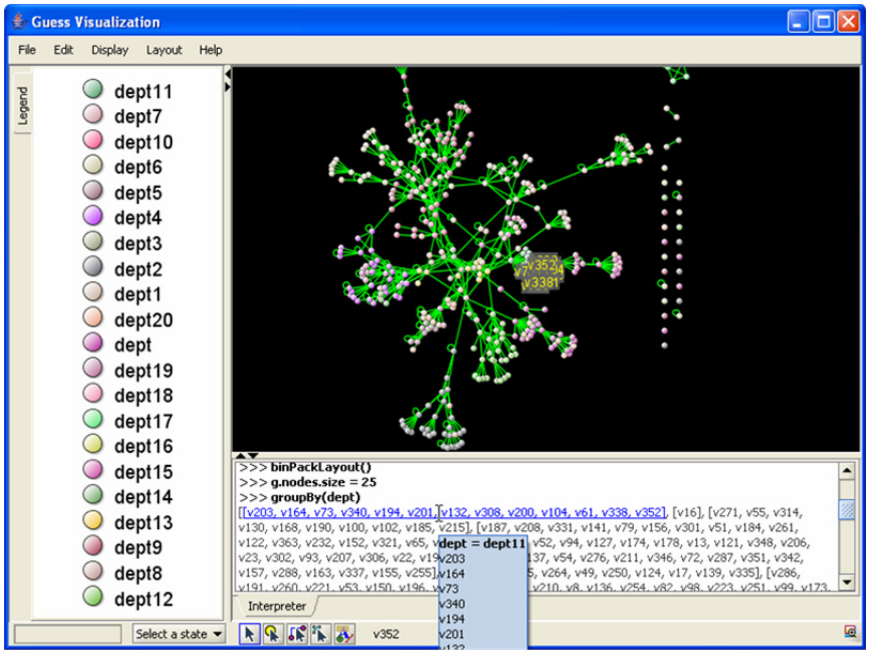


Figure 53 : Les nœuds sélectionnés dans la fenêtre graphique (à droite en haut) sont automatiquement surlignés en bleu dans la fenêtre de ligne de commande Gython (à droite en bas).

Version testée : 1.0.3 Beta
Plateforme : toutes (machine virtuelle Java).
Accessibilité : édition manuelle d'un fichier de configuration requise avant l'exécution du programme ; manipulation clairement expliquée dans le fichier readme.txt.
Exportation graphique : raster et vectoriel
Formats d'échange : pajek (.net), graphml, gdf.
Langue : anglais.
Fond de carte territorial: oui, moyennant un effort de programmation
Licence : commerciale.
Site: http://graphexploration.cond.org/

3.7 NAViGaTOR

NAViGaTOR (*Network Analysis, Visualization, & Graphing TORonto*) a été originairement conçu pour visualiser des interactions de protéines. Il permet cependant de représenter tout graphe planaire ou 3D. Son moteur graphique est bien optimisé, on peut éditer un réseau manuellement tout en laissant tourner l'algorithme itératif d'agencement, sans effets de saccade.

De fait, l'intérêt et la spécificité du logiciel pour la cartographie générique de réseaux de flux tiennent justement dans la manipulation *manuelle* des agencements des nœuds (*cf*. 2.1.4). La structure générale de l'interface utilisateur est fortement inspirée par celle d'Adobe Illustrator. De nombreux raccourcis clavier reprennent également cette logique. Un manque notable du logiciel est qu'il ne permet pas d'identifier la direction des arrêtes, à l'aide de flèches ou d'autres indices sémiologiques. Ces éléments peuvent cependant être ajoutés dans Illustrator, en revenant sur l'image vectorielle exportée.

Du point de vue de l'agencement automatique, le logiciel n'offre qu'une approche basée sur la force (*cf.* 2.1.1.1), sans précision de l'algorithme utilisé. Sa force réside dans une sélection facilitée de sous-ensembles de nœuds auxquels on peut appliquer des agencements circulaires. Sur le plan de l'analyse des métriques du réseau, NAViGaTOR offre également peu d'outils, ou alors des outils spécifiques à des réseaux d'interactions chimiques. Il dépend donc d'une analyse préalablement établie, par exemple dans R ou dans Cytoscape. Il peut dès lors exploiter ces dernières en les traduisant en couleurs, tailles ou transparences.

On peut recommander ce logiciel pour le traitement graphique manuel de réseaux dont l'utilisateur connaître préalablement la hiérarchie et la structure, et qu'il cherche à représenter.

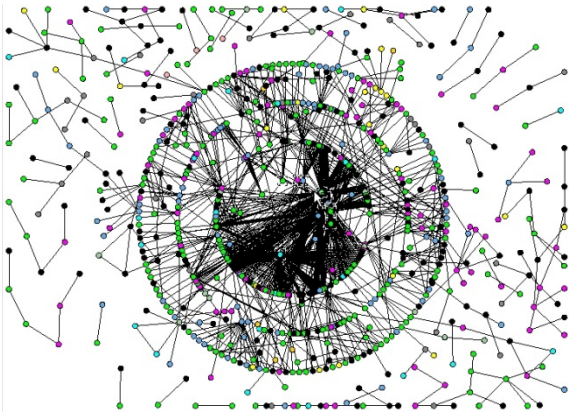


Figure 54 : Agencement circulaire à trois niveaux, à partir de sélections de nœuds à la main.

Version testée : 2.2.1
Développeur : Université de Toronto. Jurisica Lab, IBM Life Sciences Discovery Center, Ontario Cancer Institute.
Plateformes : Windows, Linux, Mac OS.
Licence : gratuit, licence non-spécifiée.
Agencements: circulaires, grid, basé-force non spécifié, surtout agencements manuels.
Langue : anglais.
Formats d'échange : gml, psi, owl, net (pajek).
Formats d'exportation graphique : vectoriel (pdf, svg) et raster. Les directions des arrêtes sont conservées.
Fond de carte territorial: non.
Licence : commerciale.
Site : http://ophid.utoronto.ca/navigator/

3.8 Flowmap

Flowmap adopte une stratégie différente des logiciels discutés jusqu'ici. Au lieu de se focaliser en premier sur le réseau, il part d'une logique GIS standard, consistant à représenter les données dans un espace territorial, puis à ajouter des éléments visuels pour montrer les liens entre des pairs de lieux distants. Il peut être conçu comme un GIS standard spécialisé dans le stockage, l'analyse et l'affichage de données de flux. Le logiciel est en outre capable de calculer des distances en termes de temps de déplacement ou de ses coûts, en se basant sur la structure du réseau de flux. Il peut enfin calculer les aires de marché ou d'influence des infrastructures (*catchment area analysis*), ainsi que les optimisations de localisation.

Bien que Flowmap fonctionne de manière autonome, il est prévu que le résultat des calculs soit exporté dans un logiciel SIG tels ArcMap pour le post-traitement en termes de représentation.

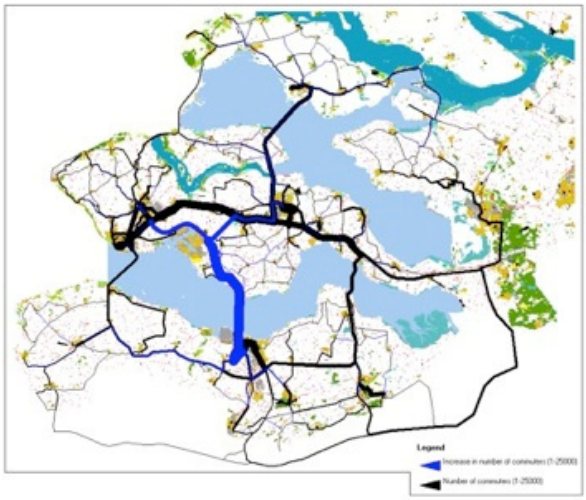


Figure 55 : Flowmapper.

Système d'exploitation : Windows, Internet Explorer 7+ est requis
Formats d'échange : MapInfo MIF, ESRI shape, Tables DBF pour l'exportation de données numériques
Exportation graphique: images BMP ou JPG. Pas d'exportation vectorielle directe, si ce n'est sous forme de format d'échange GIS MIF ou SHAPE.
Développeur : Dr. Tom de Jong, Université d'Utrecht.
Langue : anglais.
Licence : commerciale gratuite pour un usage éducatif et commerciale payante pour un usage professionnel. La version payante permet l'import/export de données à partir de/dans un fichier DBMS, l'usage de DBMS autres que dBASE III, la création de fichiers de log et opérations en mode batch, exploration MTL dans les modèles gravitaires.
Site : http://flowmap.geog.uu.nl/

3.9 JFlowMap

JFlowMap est un prototype de recherche prometteur développé par le département d'informatique de l'Université de Fribourg (Suisse). Sa capacité spécifique consiste, comme dans Flowmap, à montrer des interactions sur un fond de carte territorial. Contrairement à Flowmap, ces interactions ne suivent pas un réseau de transport mais sont montrées comme connexions géométriques abstraites entre paires de lieux.

Une autre spécificité de JFlowMap consiste cependant à prendre en compte l'évolution temporelle de l'intensité des flux. À cette fin, le logiciel propose un mode de visualisation original, baptisé

« Flowstrates »[24] et combinant une variante de diagramme de matrice (2.1.1.6) à une représentation à la fois territoriale et réseau des lieux connectés (Boyandin et al. 2011 ; Figure 56). Les origines et les destinations sont présentées sur deux cartes séparées, avec un diagramme de matrice au centre. Contrairement aux diagrammes de matrices origines (en lignes) – destinations (en colonnes), la représentation Flowstrates dédie chaque *ligne* à un seul flux origine-destination, privilégiant l'évolution temporelle présentée en colonnes (Figure 56). Le prix sémiologique de ce choix est que l'ensemble des flux ne peut être représenté sur l'ensemble des périodes temporelles. Le mode de visualisation rend possible, et nécessite, l'interactivité. Des paires de lieux particuliers, ou des collections de lieux peuvent être sélectionnées afin d'étudier l'évolution temporelle de l'ampleur des flux qui les mettent en interaction. Des animations peuvent être utilisées pour observer des données enregistrées en séquence sur plusieurs années.

Du point de vue de la généralisation, JFlowMap fait du rassemblement d'arrêtes (2.1.2). Focalisé sur la localisation territoriale des lieux, le logiciel n'offre pas d'algorithmes d'agencement, exception faite de l'agencement circulaire (Figure 57). Sur les cartes globales – sans matrice intermédiaire – les arrêtes sont colorées par segments pour distinguer les origines (côté vert) et les destinations (côté rouge).

À l'heure du présent rapport, le logiciel est encore instable et difficile à configurer. Nous le mentionnons néanmoins car il figure parmi les rares logiciels à permettre une analyse simultanée des réseaux et des localisations territoriales des nœuds, tout en tenant compte de la dimension temporelle absente des autres logiciels intégrés.

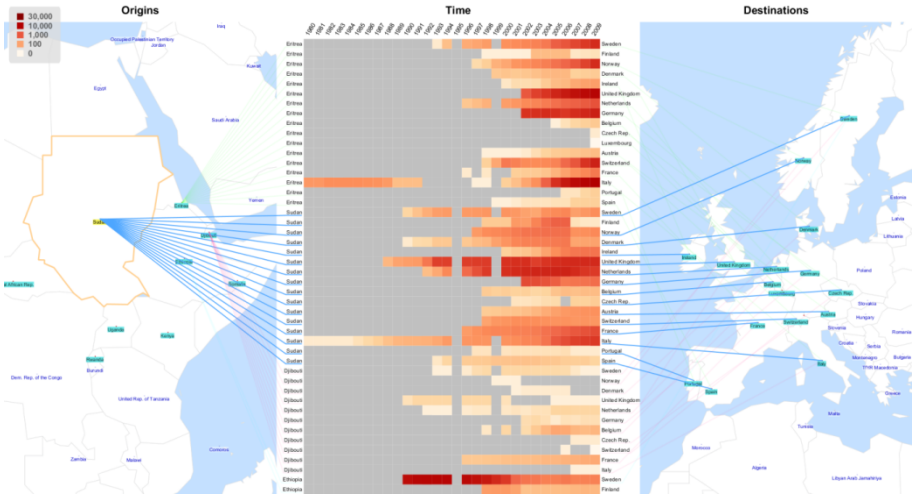


Figure 56 : Visualisation « Flowstrates » avec JFlowmap. Les origines dans les lignes de gauche, les destinations dans les lignes de droite, en colonnes, les années. Les couleurs dénotent le nombre de migrants.

[24] Pour une démonstration d'utilisation en ligne, voir http://code.google.com/p/jflowmap/wiki/Flowstrates

Figure 57 : Divers panneaux de visualisation dans JFlowMap.

Développeur : Université de Fribourg.
Version testée : 0.16.6
Plateformes: Multiplateforme (exécutable java jar).
Licence: open source.
Langues : anglais.
Accessibilité : Exécution en ligne de commande, préparation manuelle du fichier de configuration
Formats d'échange : pour importation seulement : ESRI shape pour les fonds de carte territoriaux, graphml pour les données de flux (les coordonnées xy doivent être enregistrées dans la même projection que le fond de carte), les deux comprimés avec gzip (fichiers gz) ; déclaration des sources nécessaire avec le fichier de configuration natif jfmv
Formats d'exportation graphique : vectoriel (svg) [mais arrêtes exportées comme objets polygone + crashs occasionnels lors de l'exportation]
Site : http://code.google.com/p/jflowmap/

3.10 Graphviz

Graphviz est un outil de visualisation de graphes open source en *ligne de commande*.

Du point de vue graphique, il permet, comme d'autres logiciels, de paramétrer les couleurs, les polices de caractère des labels, l'épaisseur des lignes etc. Les nœuds peuvent être dotés de formes personnalisées.

Pour la visualisation interactive et le traitement des graphes, il s'appuie sur un ensemble de bibliothèques tierces, pouvant être choisies par l'utilisateur ou appelés en fonction du graphe visualisé. Cela implique l'absence d'un environnement visuel unifié disponible dans Cytoscape ou Gephi.

Graphviz propose des agencements hiérarchiques, basés sur la force ou circulaires. Il se distingue des autres logiciels en proposant également des visualisations de type Gmap.

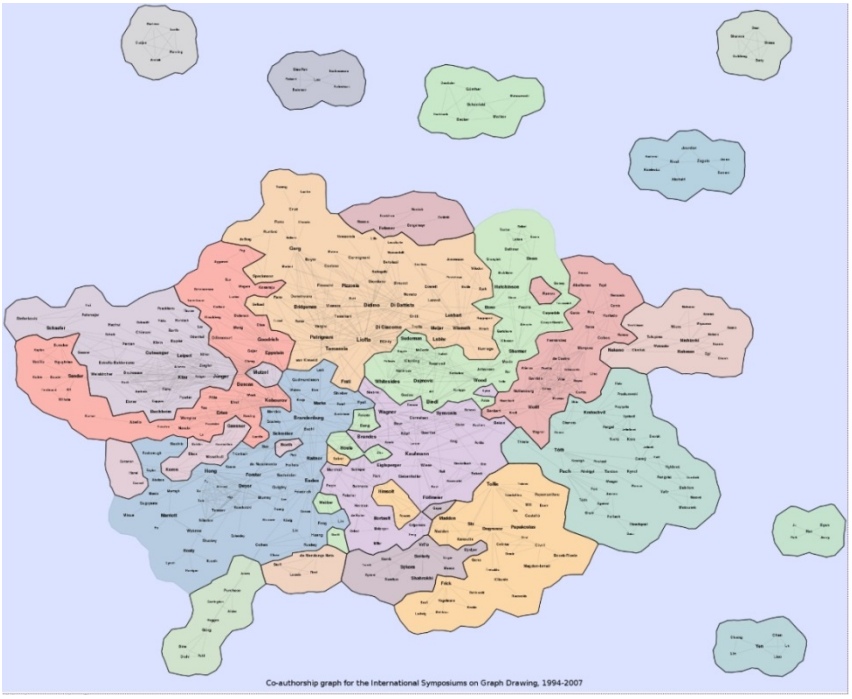


Figure 58 : Visualisation Gmap de relations d'auteurs d'articles dans GraphViz. Les auteurs sont considérés en relation s'ils cosignent une publication[25].

Environnement d'exécution : Multiplateforme Linux / Windows / Mac, exécution en ligne de commande.
Formats d'échange : format .gv propore à Graphviz, exprimé dans le langage .dot : autres formats d'échange disponibles : GXL, XML.
Formats de sortie graphique : Vectoriels (SVG, PDFk, Postscript) or display in an interactive graph browser.
Langue : anglais.
Développeurs: John Ellson, Emden Gansner, Yifan Hu, Arif Bilgin, Dwight Perry
Site : http://www.graphviz.org/

3.11 Produits CIShell

CIShell (Cyberinfrastructure Shell) est une plateforme open source, basée sur l'architecture d'Eclipse, interface majeure de programmation Java. Le projet est géré par la communauté du web et sert à l'intégration de bases de données, d'algorithmes, d'outils et de ressources computationnelles. Son avantage est d'offrir une plateforme permettant de combiner des outils de prétraitement, d'analyse et de visualisions de données sur mesure liées à une problématique spécifique à une discipline, en combinant des algorithmes existants. Plusieurs produits d'analyse réseau ont été construits sur la base de CIShell, dont *Network Workbench* et *Dynanets*.

3.11.1 Network Wokbench

Network Workbench est un projet d'outils d'analyse et de visualisation de réseaux actuellement en cours de développement. Plus qu'un outil intégré, le projet a pour objectif de constituer un assemblage d'algorithmes couvrant l'ensemble des besoins d'un cartographe et d'un analyste de graphes. Il articule d'ores et déjà plusieurs logiciels et bibliothèques présentés ici, dont JUNG (4.2.5.2), GUESS (3.6), Prefuse (4.2.5.8) etc.

[25] http://www.graphviz.org/Gallery/undirected/gd_1994_2007.html

La plateforme offre également un accès en ligne aux données réseau, aux outils d'analyse et de visualisation. Au-delà des aspects purement logiciel, l'objectif est d'offrir une plateforme de *collaboration* dans laquelle des chercheurs dans des domaines spécifiques (biologie, physique, sciences sociales), peuvent rapidement exploiter de nouveaux algorithmes d'analyse et de visualisation, ainsi que de partager leurs données.

Encore fortement orienté par la recherche sur les réseaux bibliométriques, ambitieux mais en cours de développement, de nombreuses fonctionnalités de ce projet restent vacillantes mais il est à suivre surtout en tant que solution intermédiaire entre les logiciels intégrés, complets mais peu flexibles aux besoins spécifiques, et les bibliothèques de scripts, adaptables à souhait mais demandant une prise en main difficile.

Version testée : 1.0.0
Développeurs : Network Workbench Project, Université d'Indiana.
Environnement d'exécution : Windows, Linux, Mac OS.
Formats d'échange : format natif (nwb), net (Pajek), GraphML, XGMML, PrefuseTreeML. Également formats de stockage de données bibliométriques (ISI, NSF, Scopus etc.).
Langue : anglais.
Formats de sortie graphique : diverses disponibilités en fonction de la bibliothèque de visualisation mobilisée
Site : http://nwb.slis.indiana.edu/

3.11.2 Dynanets

Similairement à Network Workbench, Dynanets est un assemblage d'algorithmes dans la logique CIShell. Cette variante a pour objectif spécifique d'analyser des réseaux de flux changeants au cours du temps. Pour la visualisation, cet assemblage CISchell incorpore notamment Cytoscape 2.6.3. En vue de conduire des simulations dynamiques, Dynanets est également articulé au programme de simulation multiagent NetLogo.

Version testée : 1.0.0
Développeurs : Université d'Amsterdam.
Environnement d'exécution : Windows, Linux, Mac OS.
Formats d'échange : format natif (nwb), net (Pajek), GraphML, XGMML, PrefuseTreeML. Également formats de stockage de données bibliométriques (ISI, NSF, Scopus etc.).
Langue : anglais.
Formats de sortie graphique : diverses disponibilités en fonction de la bibliothèque de visualisation mobilisée
Site : http://dynanets.org

4 Bibliothèques, scripts, plugins, applications en ligne

En amont, en appoint, ou parallèlement aux logiciels intégrés, la visualisation de flux se fait également par l'intermédiaire d'algorithmes indépendants, pouvant être soit programmatiquement intégrés dans d'autres logiciels au moment de leur conception, soit, comme nous l'avons déjà évoqué, rendus disponibles par le moyens de plugins. Il serait illusoire de vouloir dresser une liste d'éléments de ce type, ne serait-elle qu'approximative. Quelques solutions saillantes ou intéressantes pour le développement du domaine peuvent néanmoins être évoquées.

4.1 Plugins et éléments de logiciels

En termes de plugins, nous retenons trois exemples. Deux pour le très répandu, mais propriétaire et coûteux, *ArcGIS* d'ESRI : Geotime et FlowMapper. Une solution homyme – FlowMapper – est

également disponible pour le système open source *Quantum GIS*. Dans les trois cas, il est possible de superposer des données de flux à un fond de carte territorial (2.2.1).

4.1.1 GeoTime

Geotime est une application autonome, jouissant néanmoins d'une forte connectivité avec ESRI ArcGIS. Elle permet de cartographier les mouvements dans leur temporalité en 3D. Depuis la version 5.1, GeoTime permet de visualiser simultanément l'espace réseau et l'espace topographique, avec la possibilité d'ajouter une 3^{e} dimension aux deux pour représenter le temps.

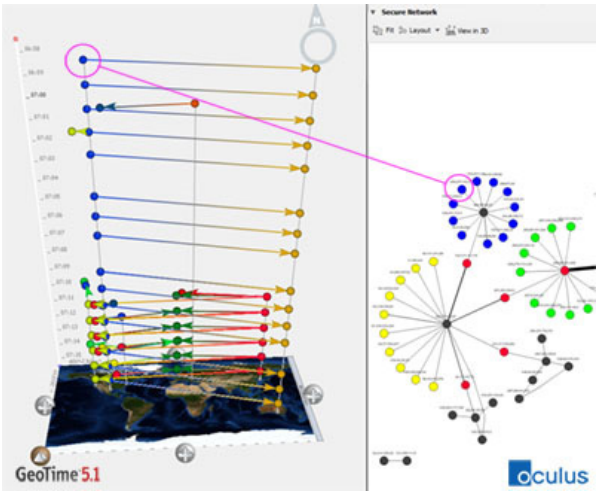


Figure 59 : Cyber-attack logs containing both link and geographical data.

Version commentée : 5.
Plateforme : Windows et plugin ArcGIS.
Licence : commerciale >3000$.
Site : http://www.geotime.com/

4.1.2 FlowMapper pour ArcView

FlowMapper est composé à partir d'une série d'algorithmes produits par Waldo Tobler (*cf.* Tobler 1981 ; Dorigo & Tobler 1983) en 1987, rassemblée dans les années 2000 par David Jones (Lemurtech Consulting, Castle Rock, Colorado). Le résultat est une application autonome, ainsi qu'un plugin ArcGIS 9.x, sous forme d'une série de macros VBA (Visual Basic), permettant de lier le système de données natif à ce SIG. Malgré son articulation à des environnements de programmation plus anciens, le logiciel demeure fonctionnel à ce jour.

Graphiquement, il accomplit des opérations de sélection basées sur l'intensité de flux. Des calculs d'autres métriques réseau ne sont pas disponibles et ne peuvent pas être reliées à la visualisation. Les traitements graphiques aujourd'hui courants, comme le *edge bending* (2.1.3), ne sont pas disponibles. Son intérêt tient dans son apport historique au domaine de la visualisation de flux, ainsi que dans sa capacité de superposer les flux à des fonds de cartes territoriaux, une capacité vraiment déployée surtout dans l'articulation du logiciel avec ArcGIS. L'application n'a plus été développée depuis de nombreuses années.

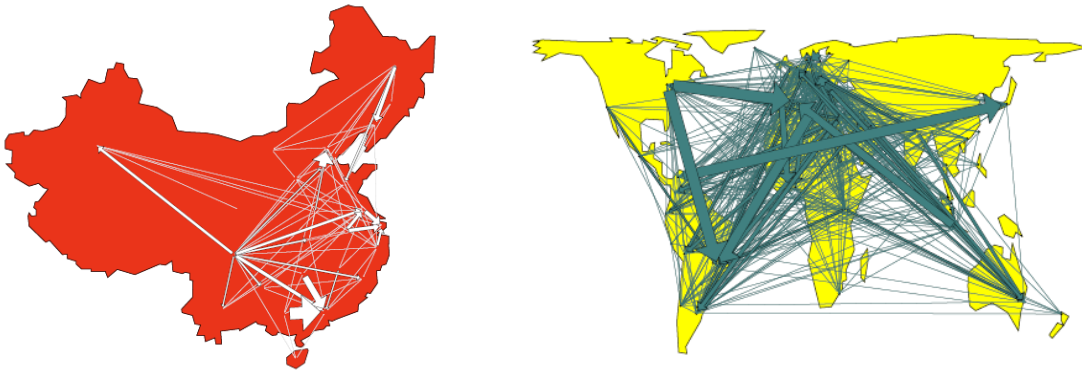


Figure 60 : Deux exemples de cartes de flux produits dans l'application autonome FlowMapper. À gauche, les migrations en Chine. À droite, échanges commerciaux mondiaux. Les fonds de cartes peuvent être améliorés en articulant l'application

Version testée : 1.1.
Plateforme : Windows et plugin ArcGIS.
Développeur : Center for Spatially Integrated Social Science, Département de Géographie de l'Université de Californie, Santa Barbara.
Formats d'échange : système de fichiers spécifique composé systématiquement d'un fichier de coordonnées projetées des lieux, d'une matrice des interactions, d'optionnels noms des lieux et de fonds de carte en format vectoriel svg. Le Flow Data Model peut néanmoins être exploité directement dans ArcGIS, par l'intermédiaire de macros VBA.
Formats de sortie graphique : raster et vectoriel (svg). Les arrêtes sont exportées en tant que polygones mal formées. Modification graphique difficile, à moins de passer par ArcGIS pour exporter à partir de là.
Site : http://csiss.ncgia.ucsb.edu/clearinghouse/FlowMapper/

4.1.3 FlowMapper pour Quantum GIS

FlowMapper pour Quantum GIS peut être considéré comme un remplacement contemporain de son homonyme pour ArcGIS. Cette version remaniée, et reprogrammée en Python, s'articule ainsi à un logiciel SIG open source largement diffusé, et cela par l'intermédiaire du gestionnaire de plugins Quantum GIS aisé à prendre en main. Elle propose essentiellement les mêmes fonctionnalités que son prédécesseur, mais les options de traitement graphique sont nettement améliorées par le déploiement du plugin dans l'environnement Quantum GIS.

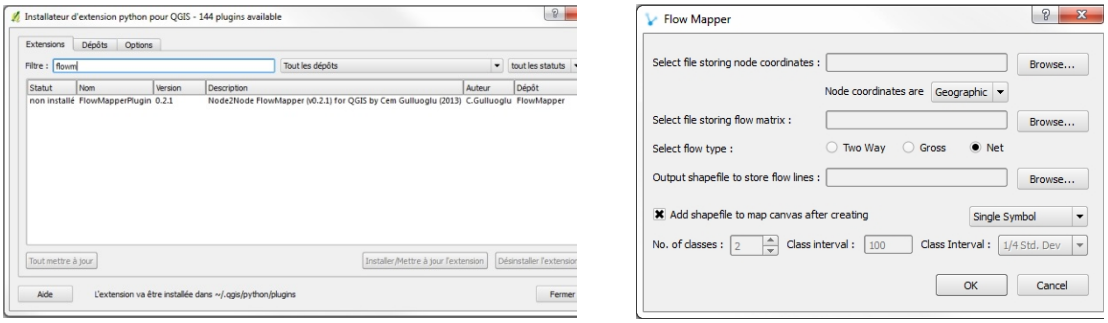


Figure 61 : Installation et usage de FlowMapper dans Quantum GIS. À gauche, l'installation du plugin à partir du dépôt (*repository*) ; à droite, l'interface de spécification du traitement.

Version testée : 0.2.1
Plateforme : Windows, Linux, Mac OS. Plugin Quantum GIS.
Formats d'échange : liste texte des coordonnées des nœuds + liste de liens. Exportation shapefile, KML (Google Maps) et TAB (MapInfo).
Formats de sortie graphique : vecteur et raster exportés par Quantum GIS. La direction des arrêtes est conservée lors de l'exportation vecteur.

Licence : open source.
Développeur : Cem Güllüoğlu, Université d'Istanbul.
Site : http://plugins.qgis.org/plugins/FlowMapper/

4.2 Langages de programmation

L'ensemble le plus vaste d'outils informatiques permettant de visualiser des flux est celui des bibliothèques de scripts. La mobilisation de ces dernières est uniquement possible à travers un effort de programmation de l'utilisateur final. Éléments fondateurs de processus de visualisation de flux, on les retrouve intégrés et articulés dans la plupart des logiciels vus jusqu'ici. C'est aussi dans ce cadre que l'on trouvera les visualisations les plus innovantes. Nous n'évoquons ici que quelques exemples, saillants par leur niveau de diffusion, par la singularité des visualisations qu'ils permettent, ou par leur rôle dans la fonctionnalité des autres logiciels. Leur présentation est organisée par langage de programmation.

4.2.1 R

Lancé en 2004 par la *R foundation*, le langage statistique R est développé par une vaste communauté de spécialistes d'une pluralité de domaines. Il est spécialement conçu pour l'exploitation de données statistiques. Trois bibliothèques de scripts R peuvent être mentionnées.

4.2.1.1 iGraph

La bibliothèque iGraph est disponible pour les langages R, Python et C. Il s'agit de la bibliothèque ouverte la plus développée et la plus diffusée parmi les spécialistes d'analyse de graphes. Toutes les métriques réseau évoquées peuvent être calculées, en grande partie par des fonctions prédéfinies. Le fait que l'on se situe dans le contexte d'un langage de programmation permet bien sûr d'implémenter toute méthode d'analyse existant par ailleurs ou créée ad hoc par l'utilisateur. Parmi les algorithmes, on citera notamment `walktrap.community`, un détecteur de communautés (2.4.4) efficace basé sur le concept de marches aléatoires (Pons & Latapy 2005), qu'iGraph est le seul environnement commenté ici à intégrer d'office.

Pour la visualisation planaire, iGraph s'appuie sur l'interface interactive Tkplot. Plusieurs types d'agencements automatiques peuvent être choisis dans ce contexte, dont l'agencement en cercle, et les agencements basés sur la force Fruchterman-Reingold, Kamada-Kawai, Spring Embedder et Reingold-Tilford. D'autres fonctionnalités peuvent bien sûr être ajoutées par voie programmatique. Des visualisations 3D sont également possibles en s'appuyant sur l'interface Rglplot.

La superposition du graphe à un fond de carte territorial n'est pas possible d'office, bien que cette fonctionnalité puisse également être ajoutée par voie programmatique.

Du point de vue de la gestion des données, iGraph permet notamment de transformer un réseau stocké sous forme de matrice d'adjacence en un objet graphe[26]. Lorsque les données sont disponibles sous cette forme, le passage par iGraph peut même s'avérer nécessaire pour les restructurer à l'usage dans d'autres logiciels. Les fonctions `read.graph` et `write.graph` permettent d'importer et d'exporter les données dans une multiplicité de formats.

Tkplot permet d'exporter les graphiques en format raster et vectoriel.

[26] Fonction `graph.adjacency`.

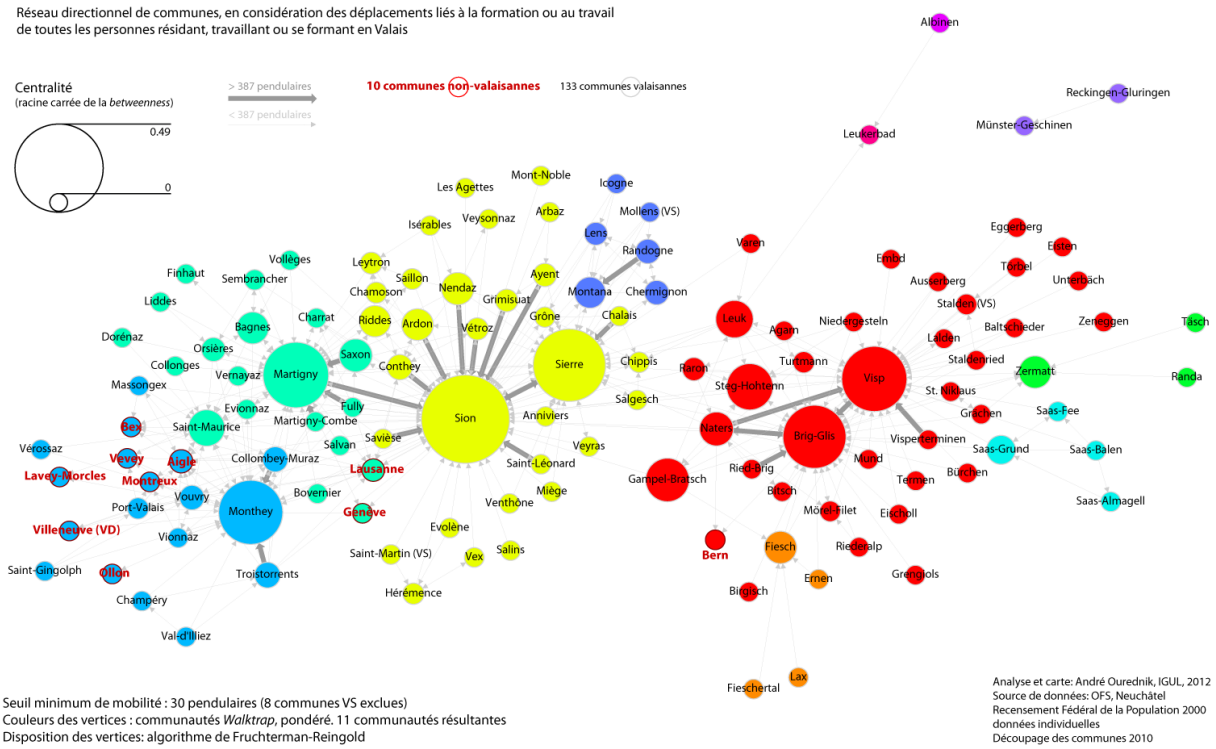


Figure 62 : réseau de mobilité du canton du Valais, avec un agencement des nœuds basée force Fruchtermann-Reingold, montrant la centralité-*betweeneness* (taille des cercles) des nœuds, les communautés *walktrap* (couleurs), ainsi que l'importance des liens en termes du nombre des pendulaires (épaisseur des traits), avec différentiation de la direction des relations. Outils : R avec la bibliothèque iGraph et Adobe Illustrator. (source : Ourednik 2012)

Développeur : The iGraph project.
Formats d'échange natifs : edgelist, pajek, ncol, lgl, graphml, dimacs, gml, dot, leda.
Site : http://igraph.sourceforge.net/

4.2.1.2 HiveR

HiveR est l'une des rares solutions actuellement disponibles permettant de générer des *hive plots* (2.1.1.3.2.6). Nativement, il importe les données au format DOT (2.5.4).

Développeur : Université DePauw.
Site : http://academic.depauw.edu/~hanson/HiveR/HiveR.html

4.2.2 Python

Python est un langage de programmation multi-paradigme (objet et séquentiel). Il est placé sous une licence libre et fonctionne sur la plupart des platesformes informatiques, des supercalculateurs aux ordinateurs personnels, de Windows, Linux, Mac OS et autres. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser. Sa syntaxe permet une prise en main aisée.

4.2.2.1 NetworkX

Hormis la bibliothèque iGraph déjà décrite, une bibliothèque de scripts open source Python à mentionner également est NetworkX (Hagberg et al. 2008). Elle permet l'importation et le stockage de graphes complexes, y compris des multigraphes (où plusieurs types d'arrêtes peuvent relier une paire de nœuds) et des multidigraphes (idem pour un graphe directionnel). Il est également possible

de générer des réseaux de flux aléatoires pour conduire des tests d'analyse et de représentation abstraits. L'implémentation dans Python offre tous les avantages de ce langage en plein essor, et utilisé toujours davantage dans les SIG comme ArcGIS ou Quantum GIS. Potentiellement, donc, NetworkX peut être programmatiquement connecté à ces derniers pour afficher des réseaux de flux sur fond de carte topographique.

Un manque sur le plan du rendu visuel est l'absence d'algorithmes d'*edge bending* (2.1.3).

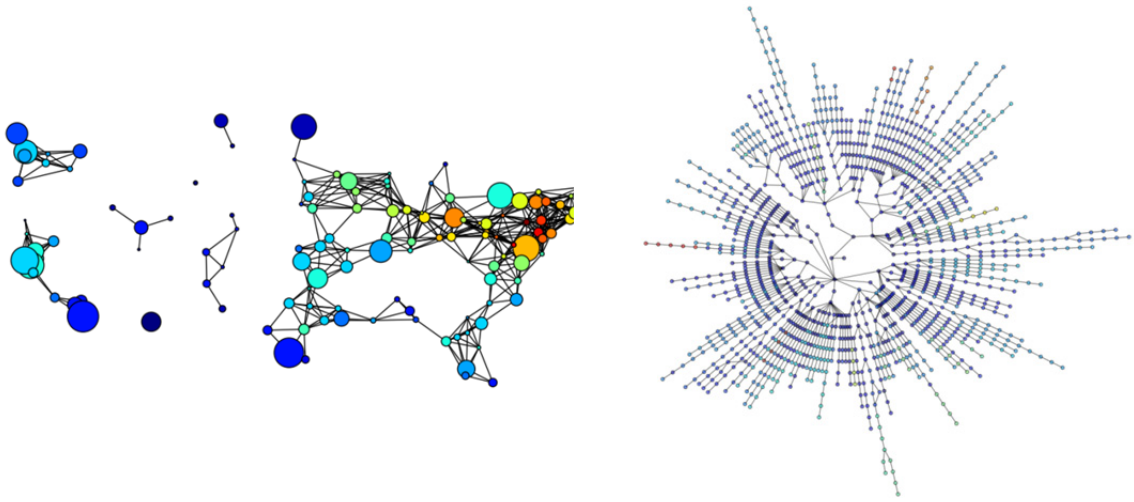


Figure 63 : Exemples de visualisation de graphes avec NetworkX. À gauche 128 villes américaines. À droite, connexions LANL de 186 sites internet.

Développeurs : Los Alamos National Laboratory ; Colgate University, Hamilton.
Site: http://networkx.github.com/

4.2.3 JavaScript

JavaScript est un langage de programmation créé en 1995 pour le compte de Netscape Communications. Il s'exécute principalement dans les navigateurs web (Firfox, Explorer, Chrome, Safari, Opera, etc.). Il est donc principalement utilisé dans les pages web interactives. Côté développement, il peut être programmé de manière séquentielle ou objet.

4.2.3.1 Sigma.js

Sigma.js permet d'afficher des graphes exportés de logiciels comme Gephi dans un environnement de navigateur web.

Format d'échange : GEXF, GML.
Format d'affichage et d'exportation graphique : affichage html5 dans les navigateurs web.
Site : http://sigmajs.org

4.2.3.2 Data-Driven Documents

Data-Driven Documents – ou D^3 – est un ensemble de bibliothèques JavaScript permettant la visualisation de données, y compris de graphes. Il se base sur des structures de données visuelles hautement standardisées du web : SVG et CSS. Les exemples de visualisation disponibles montrent l'implémentation d'agencements *circulaires*, les *treepie*, de rassemblement des nœuds, des diagrammes de Sankey, de *chord digrams*, des *hive-plots* etc.

Un grand intérêt de cette solution est de pouvoir afficher les cartes de flux dans les navigateurs web, optionnellement de manière interactive. Le coût de cet avantage est bien sûr le fait de devoir formuler les représentations souhaitées dans un langage de programmation.

Data-Driven Documents n'est pas une bibliothèque d'analyse, mais de représentations de graphes. Son articulation à des scripts d'analyse est donc nécessaire pour obtenir une solution plus complète.

Format d'échange : JSON.
Format d'affichage et d'exportation graphique : vectoriel (svg).
Site : http://d3js.org/

4.2.3.3 Processing.js

Processing.js est une adaptation JavaScript du langage de visualisation et d'animation Processing. Elle permet de visualiser les données de toute sorte, de produire des animations, voire des animations interactives. Similairement à Raphaël[27] ou Protovis[28], il s'agit d'un langage complet pour la visualisation de données dans les navigateurs Web. Comme ces derniers, il ne s'appuie pas sur une architecture plus standardisée – et plus répandue – de type HTML-SVG-CSS. Certains packages externes[29] de Processing gèrent plus directement la visualisation de graphes.

Site : http://processingjs.org/

4.2.4 Matlab

MatLab est un langage de programmation et un environnement de développement puissant, à la fois processuel, orienté objet et exécutable en ligne de commande, à la manière de Python. Il est spécifiquement conçu vers le traitement de données mathématiques, et largement répandu dans la communauté scientifique. L'interface graphique dont il est accompagné permet l'importation facilitée de données et l'affichage de résultats graphiques.

Sur le plan de l'analyse et de la visualisation de graphes, MatLab possède l'objet `biograph`, destiné à la bioinformatique, mais permettant de traiter d'autres types de réseau.

La fonction `biograph(data)` appliqué à une matrice de flux (*cf*. Tableau 1) permet de créer un objet graph. Si on exécute, par exemple, la commande `bg=biograph(data)`, on obtient un objet (nommé arbitrairement `bg`) pouvant être manipulé pour calculer des métriques réseau standard, et pour produire des représentations graphiques.

Exemple d'utilisation produisant la Figure 64, gauche :

```
    >> cm = [30 38 0 0 21 0 ; 1 4959 0 3 392 25 ; 0 28 70 0 6 0 ; 1 95 0 83 58
0 ;31488 2 3 1827 13 ; 0 4320 0 34 294]
    >> bg = biograph(cm)
    >> h = view(bg)
    >> set(h.Nodes,'Color',[.5 .7 1])
    >> set(h.Edges,'LineColor',[0 0 0])
    >> set(h.Nodes,'Shape','circle')
```

[27] http://raphaeljs.com/
[28] http://mbostock.github.com/protovis/
[29] *e.g.* giCentre Utilities : http://www.gicentre.org/utils/

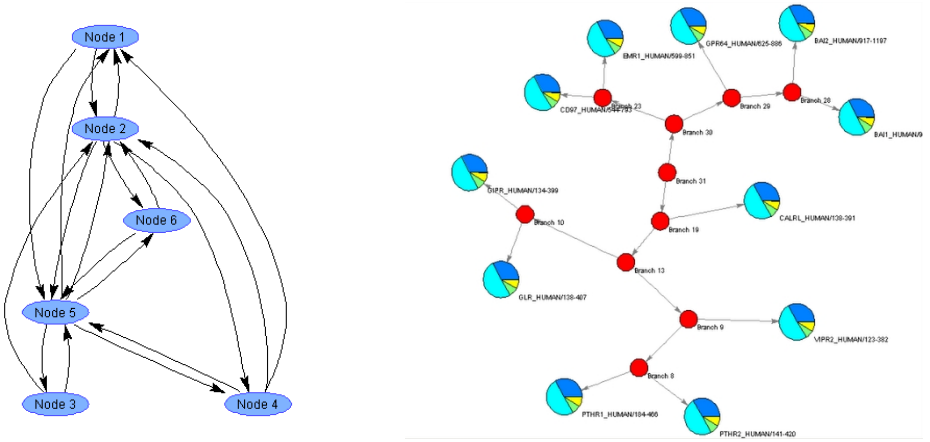


Figure 64 : Cartes de flux produites dans Matlab à l'aide de classes standard. À droite, remarquer la possibilité – souhaitable ou pas – de figurer des pie-chart sur les nœuds.

Sans doute moins approprié que d'autres solutions pour la représentation de graphes discrets, MatLab constitue un excellent outil pour la cartographie de processus de diffusion continus (2.3). Le logiciel implémente un bon moteur de lecture de données SIG, permettant de représenter ces processus sur un fond de carte territorial.

Développeur : The MathWorks.
Licence : commerciale 50 à 2000 €, en fonction de l'usage.
Site : http://www.mathworks.fr/

4.2.5 Java

Java est un langage de programmation orienté objet créé par Sun Microsystems, récemment rachetés par la société Oracle. Reprenant en grande partie la syntaxe du langage C++, sa particularité principale est que les logiciels écrits dans Java sont très facilement portables sur plusieurs systèmes d'exploitation tels que Windows, Linux ou Mac OS. Il permet en outre de développer des applications client-serveur, opérant sous le mode « applet » côté client.

Par sa large diffusion et par sa portabilité, Java est le langage de choix de nombreux développeurs, y compris dans le domaine de la visualisation de graphes.

Cytoscape, Gephi, JFlowmap, pour ne citer que quelques logiciels précédemment évoqués, ont tous été développés dans Java. Dans les exemples qui suivent, nous retrouvons donc aussi des composantes de ces logiciels. En effet, la structure orientée objet de Java permet de récupérer et d'intégrer aisément des classes (bouts de code modulaires) et de les utiliser comme éléments de construction d'une solution plus vaste.

4.2.5.1 Piccolo

Piccolo est un ensemble de bibliothèques (*toolkit*) Java et C# permettant la visualisation de graphiques 2D. Il permet des effets d'agrandissement, les animations, et les représentations multiples d'un même set de données. Il sert de moteur graphique à plusieurs applications précédemment citées ; il est également inclus dans les bibliothèques Java plus étendues, et dans des solutions intégrées comme GUESS (3.6).

Développeur : Université de Maryland.
Site : http://www.cs.umd.edu/hcil/piccolo/

4.2.5.2 JUNG

La bibliothèque JUNG (*Java Universal Network/Graph Framework*) porte une large variété de représentations de graphes directionnels, non-directionnels, multimodaux et hypergraphes. Elle permet d'annoter les graphes et leurs éléments. Elle est conçue pour la création d'outils analytiques de sets de données complexes, qui examinent en même temps les relations entre les entités (nœuds), ainsi que les métadonnées associées aux nœuds et aux arrêtes.

La distribution actuelle implémente un nombre d'algorithmes issus de la théorie des graphes et de l'analyse de réseaux sociaux, etc. Elle permet l'analyse des communautés, le calcul de nombreuses métriques réseau (centralité, PageRank, HITS, etc.).

Son framework graphique permet le développement de visualisations et la manipulation de graphes dans le cadre d'applications interactives. Des mécanismes de filtrage permettent les opérations cartographiques de sélection.

Développeurs : Google ; Microsoft Research ; RABA Technologies.
Site: http://jung.sourceforge.net/

4.2.5.3 GINY

La bibliothèque GINY sert d'interface pour des algorithmes d'analyse et de visualisation de graphes fournis par d'autres bibliothèques. Son moteur graphique est basé sur Piccolo. Le calcul des métriques réseau et les agencements s'appuient sur la bibliothèque JUNG. GINY inclut en outre ses propres algorithmes d'analyse de graphes, d'agencement des nœuds et des options de symbolisation innovantes.

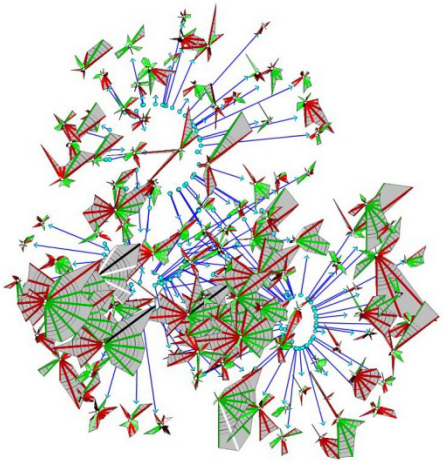


Figure 65 : « Radar nodes » - Nœuds figurés par des *diagrammes des Kiviat* (*radar chart*) (*cf*. Mayr 1887) révélant leurs propriétés statistiques, dans GINY.

Développeur : Institute for Systems Biology.
Site: http://csbi.sourceforge.net/

4.2.5.4 Hypergraph et Hypertree

Hypergraph et Hypertree sont des *frameworks* permettant la représentation interactive de graphes arborescents de cercles hyperboliques pour les graphes hiérarchiques (2.1.1.3.2).

Sites : http://hypergraph.sourceforge.net/ ; http://hypertree.sourceforge.net/

4.2.5.5 TreeMap

TreeMap permet la représentation interactive de graphes arborescents (2.1.1.3.2.4).

Développeur : MIT.
Site : http://treemap.sourceforge.net/

4.2.5.6 InfoVis

À l'instar de TreeMap, Infovis permet la représentation interactive de graphes arborescents. Les interactions comportent la navigation dans la structure des données, comme le zoom, les déformations de cartes de type *fisheye* (Figure 66, gauche), le filtrage dynamique (sélection), l'étiquetage dynamique etc. Parmi ses autres attraits, on citera également une série d'algorithmes innovants de représentations arborescentes (Figure 67).

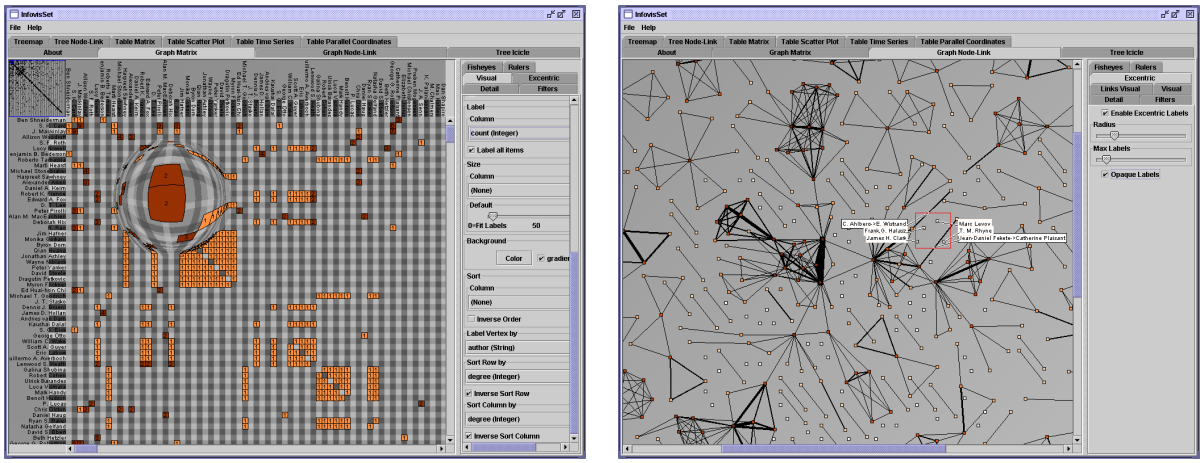


Figure 66 : Interaction dans InfoVis : *fisheye* avec un diagramme de matrice (gauche) et graphe avec les étiquettes des éléments apparaissant lors d'une sélection rectangulaire (droite).

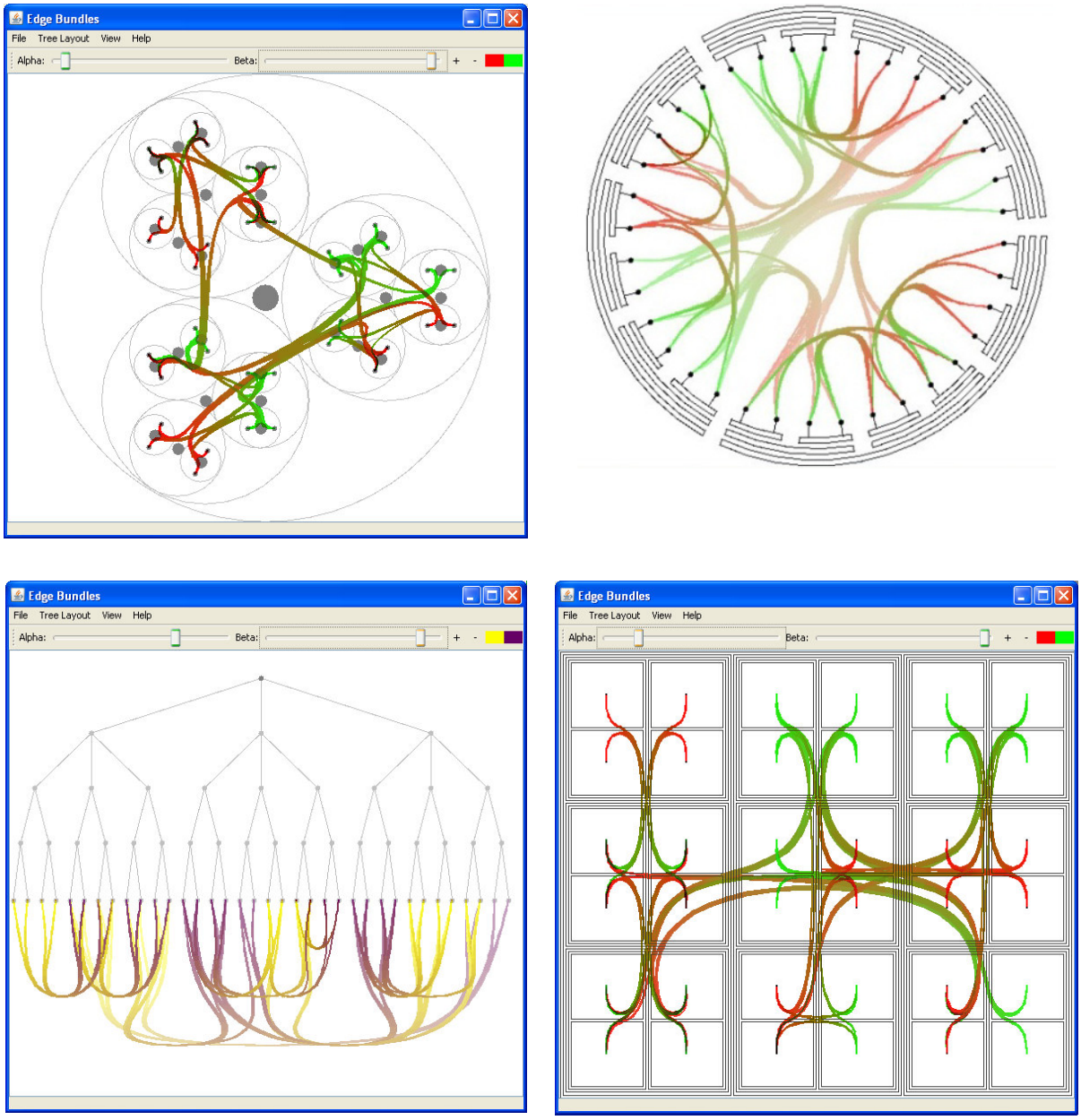


Figure 67 : D'autres représentations générées à l'aide d'Infovis.

Développeur : Inria (France).
Site : http://ivtk.sourceforge.net/

4.2.5.7 Libsea et Walrus

Libsea est un framework Java spécialisé dans le traitement de larges graphes orientés et hiérarchiques à une composante. Le programme ouvert Java Walrus permet la visualisation de tels graphs sur certaines machines, mais une mise à jour du code peut s'avérer nécessaire sur des environnements d'exploitation plus récents. Nous le mentionnons pour sa capacité de visualisation 3D.

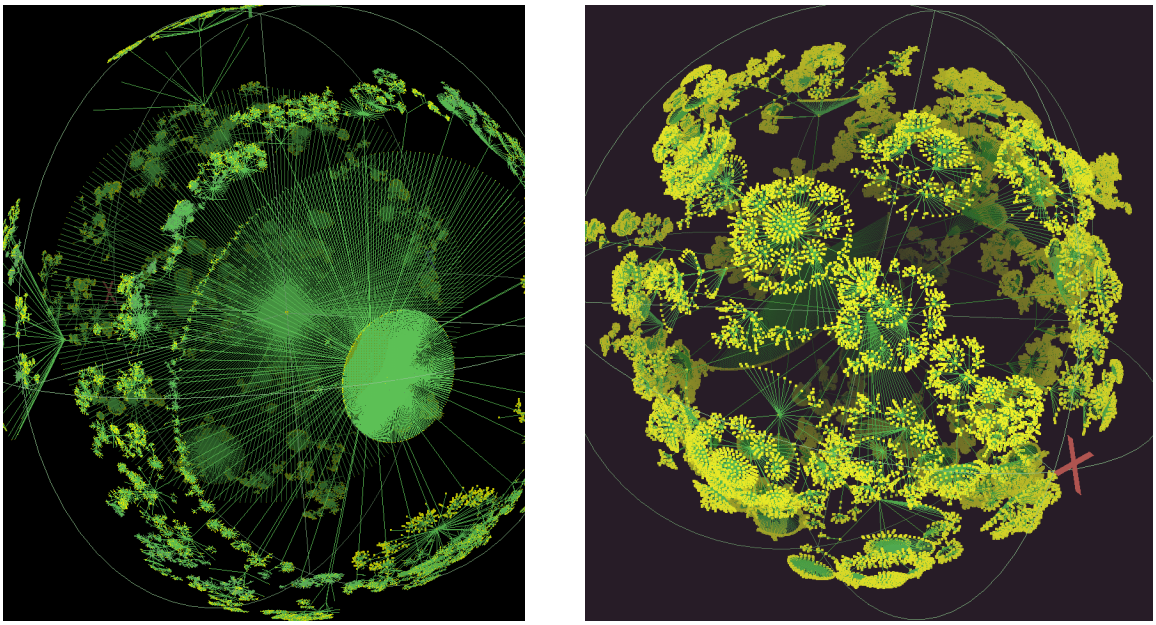


Figure 68 : Visualisations de graphes 3D dans Walrus.

Développeur : Caida, The Cooperative Association for Internet Data Analysis.
Licence : Open Source GNU GPL.
Format d'échange : LibSea
Site: http://www.caida.org/tools/visualization/walrus/

4.2.5.8 Prefuse

Prefuse est une bibliothèque Java – parallèlement aussi une bibliothèque ActionScript pour Adobe Flash Player – pour la visualisation de graphes. Elle supporte nativement de nombreux types de stockage de données de graphes (y compris les arborescences hiérarchiques), qu'elle permet de visualiser selon des agencements 2D et 3D divers. La gestion de l'interactivité est également prévue. Disponible en tant que bibliothèque seulement, elle doit être intégrée dans un logiciel pour devenir exploitable. FlowMapLayout, par exemple se base sur cette bibliothèque. Elle est également mobilisée à partir de Network Workbench.

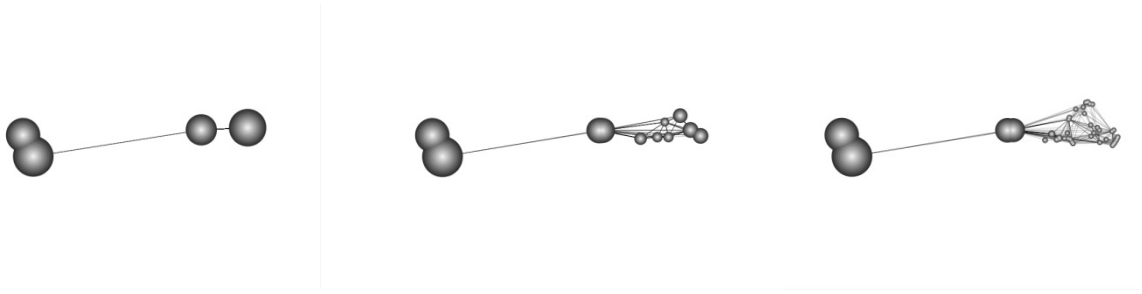


Figure 69 : Visualisations interactives dans Prefuse, exécuté à partir de Network Workbench. Les nœuds sont rassemblés en super-nœuds. Ils se décomposent en détails à l'approche du pointeur de la souris.

Développeurs : Heer, Card et Landy.
Licence : Open source (BSD).

4.2.5.9 Flow Map Layout

La bibliothèque de classes Flow Map Layout hérite en partie de celles de Prefuse. Elle se présente aussi comme un exécutable java jar à interface graphique, permettant d'importer des données de flux hiérarchiques et de les visualiser, néanmoins sans possibilité d'exportation autre que la capture d'écran.

Son apport essentiel est une série d'algorithmes permettant des rassemblements d'arrêtes (*edge merging*) de haute qualité visuelle, ainsi que des adaptations de leur parcours pour éviter des éléments du fond de carte et les croisements d'arrêtes, tout en maintenant la position territoriale des nœuds : une superposition à un fond de carte territorial en devient possible et visuellement agréable.

Version testée : 0.1 alpha.
Développeur : Université de Stanford.
Licence : Open source, BSD
Langues : anglais
Formats d'échange : format texte spécifique au logiciel
Formats d'exportation graphique : aucun – captures d'écran.
Site : http://graphics.stanford.edu/papers/flow_map_layout/

4.2.6 Autres langages : l'exemple de Circos et de HiveGraph

D'autres langages de programmation n'offrent pas la portabilité de Java mais présentent d'autres avantages : soit la capacité de gestion plus précise des ressources matérielles d'un environnement informatique spécifique (C++), soit l'exécution facilitée dans un environnement serveur-client propre au web (PHP, Perl et autres).

4.2.6.1 Circos

Le framework Circos (Krzywinski *et al*. 2009) est un ensemble de bibliothèques Perl pouvant être appelées en ligne de commande. Il est spécialisé dans la représentation des données sous forme de cercles, proposant une mise en page intéressante révélant particulièrement l'intensité de flux directionnels entre nœuds. Le pourtour du cercle représente les 100% de liens entrants et sortants. Chaque nœud se voit allouer une couleur et la part de l'arc de cercle correspondant au total de ses liens entrants et sortants.

Circos permet également des cartes circulaires en *sunburst*, ou d'ajouter d'autres formes de visualisation, comme l'ajout d'histogrammes sur le pourtour du cercle.

Comme d'autres *frameworks* de codes exécutables en ligne de commande, Circos peut être installé sur un serveur. Ses développeurs en rendent une version accessible en ligne, par l'intermédiaire d'une interface utilisateur plus conviviale. Il n'est toutefois pas recommandé de visualiser de données sensibles par cet intermédiaire.

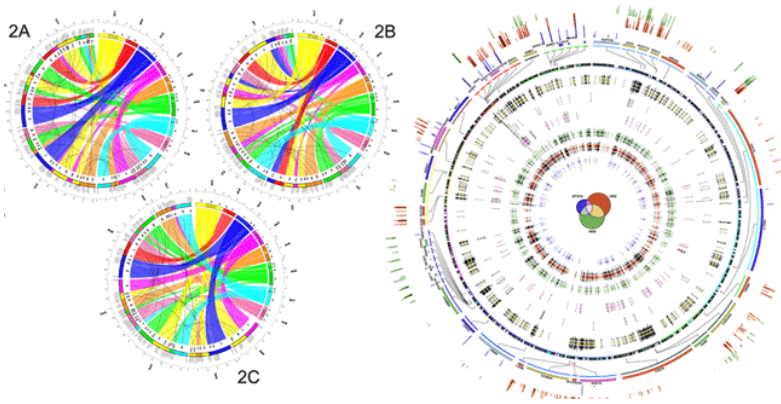


Figure 70 : Exemple d'agencement circulaire dans Circos.

Systèmes d'exploitation: Linux, peut être exécuté dans une installation Windows pour peu d'installer un environnement d'exécution Perl.
Format d'échange : matrice d'adjacence.
Exportation graphique : raster (png).
Site: http://circos.ca
Site de l'interface en ligne : http://mkweb.bcgsc.ca/tableviewer/

4.2.6.2 HiveGraph

Hive graph[30] est aux *hive plots* (2.1.1.3.2.6) ce que Circos est aux diagrammes circulaires. Cette application cloud permet d'en générer après avoir téléchargé les nœuds et les arrêtes dans l'interface en ligne. Le résultat peut être conservé sous forme de fichiers raster png.

Version testée : 1.0
Développeur : Robert Lintner, Broad Institute.
Format d'échange : duo de fichiers de description de nœuds et des arrêtes.
Exportation graphique : raster (png).
Site : http://wodaklab.org/hivegraph/

5 Services

Dans certain cas, il peut être plus judicieux pour une collectivité publique ou pour un institut de recherche d'externaliser la production de cartes de flux en faisant appel aux spécialistes du domaine. Il existe plusieurs manières de le faire, dont la plus simple est d'exploiter des solutions de cartographies de flux payantes disponibles dans le *cloud*. Parmi les applications cloud gratuites de ce type, nous avons déjà évoqué Circos 4.2.6.1) et HiveGraph (4.2.6.2). L'externalisation la plus complète consiste à charger un tiers de l'ensemble des opérations de récolte, d'analyse et de visualisation de données. Le revers de la solution est la nécessité de livrer des données potentiellement sensibles à un tiers. Nous nous limitons ici à quelques exemples.

5.1 Precog Report Grid

Le module Report Grid du lociel cloud Precog Report Grid consiste en une interface de programmation (API) à laquelle on soumet des données sous forme JASON. La visualisation est paramétrée par de brefs JavaScripts. Les résultats, cartes statiques et interactives, peuvent être affichés sur

[30] http://wodaklab.org/hivegraph/graph

un site web sous forme de HTML 5 moyennant une licence. En visualisation de réseau, Precog est surtout capable de générer des diagrammes de Sankey.

Strictement parlant, cette solution, que nous n'évoquons qu'à titre d'exemple de service *cloud*, n'offre pas d'autres avantages que Data Driven Documents (4.2.3.2).

Format d'échange : JSON
Licence : commerciale gratuite ; commerciale payante pour afficher les cartes Report Grid sur son site.
Plateforme : cloud
Site : http://www.precog.com/products/reportgrid

5.2 Quadrigram et Impure

Quadrigram est un logiciel cloud statistique au sens large, fait non seulement pour l'analyse de réseaux, mais aussi de textes. Côté serveur, il s'appuie sur R et Gephi. Il offre ainsi un riche éventail de visualisations, allant de graphiques simples jusqu'à la visualisation de données complexes : graphes abstraits ou géoréférencés, treemaps, diagrammes de flux etc. Son exploitation est basée sur un langage de programmation visuelle (VPL), permettant de construire des visualisations, des animations, voire des modules interactifs de traitement analytique et graphiques de données. À côté de solutions préconfigurées, l'utilisateur dispose ainsi d'un outil de conception de nouveaux types de visualisation.

Selon les modalités d'abonnement au service, Quadgram inclut un espace serveur partagé ou dédié, dont la taille peut varier entre 10MB et 50GB. Toutes les formes d'abonnement incluent une assistance technique personnalisée.

Génération de cartes interactives : oui.
Prix : variant entre 6€ pour l'usage académique et 249€ pour de larges groupes de travail.
Plateforme : cloud.
Format d'échange : tables Excel, CSV, XML, bases SQL etc.
Format de sortie graphique : Vectoriel (Flash).
Site : http://www.quadrigram.com/

5.3 Visumap

La compagnie Visumap Technologies fournit des services et des logiciels d'analyse de données complexes, y compris des données de réseaux de flux. Elle s'efforce de faire apparaître des relations cachées par le moyen de représentations visuelles et interactives.

Prix : les cartes existantes coûtent en moyenne 250$. Le prix des services varie entre 6000 et 25000$.
Site : http://www.telegeography.com/

5.4 TeleGeography

TeleGoeography est une compagnie d'analyse du marché et de consulting en télécommunication. Sa compétence phare est la récolte, l'analyse et la visualisation de données de flux et d'infrastructure d'information. Les résultats des recherches sont fournis en ligne sous forme de rapports et de bases de données.

Des solutions de tout type, allant des cartes aux solutions interactives sont faites sur mesure par les spécialistes de la compagnie, en fonction des besoins du client.

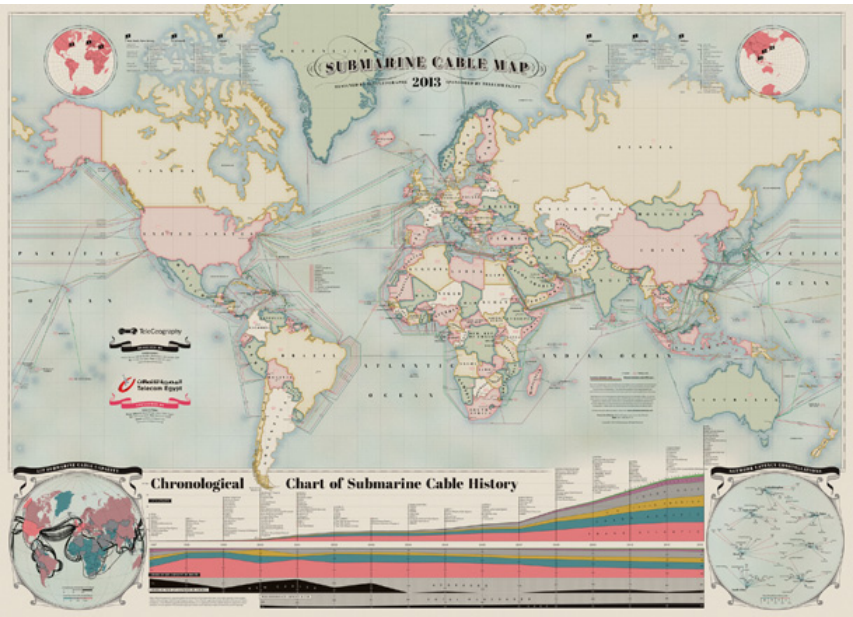


Figure 71 : Câbles.

Prix : les cartes existantes coûtent en moyenne 250$. Le prix des services varie entre 6000 et 25000$.
Site : http://www.telegeography.com/

6 Conclusion

Chaque logiciel évoqué ici permet des traitements spécifiques. Tous n'acceptent qu'un certain nombre de formats d'échange, tous ne font qu'un certain nombre de traitements, et tous ne permettent qu'un certain degré de post-traitement graphique.

Les données que le géographe souhaite analyser sont souvent des matrices de flux que de nombreux logiciels ne sont pas à même de traiter telles quelles (cf. 2.5.1). Ne serait-ce que de ce point de vue, donc, un prétraitement de premier niveau transformant la matrice d'adjacence en graphe est nécessaire.

Du point de vue graphique, à la fin de la chaîne de production, aucun logiciel évoqué ne possède les capacités d'un logiciel spécialisé comme Adobe Illustrator ou Inkscape.

Le choix de l'utilisateur se positionne donc entre deux extrêmes : (1) les logiciels intégrés ou (2) la série de logiciels hyperspécialisés, dédiés à des tâches spécifiques. La première convient à des désirs de représentation conventionnels. Si la représentation souhaitée est hautement spécifique, la seconde solution doit être privilégiée. Pour exemple, des agencements circulaires ou basés sur la force sont implémentés dans la plupart des logiciels de visualisation de graphes. Des agencements orthogonaux, les *hive-plots* ou les agencements en 3D sont plus rares, et demandent souvent un logiciel spécifique, à intégrer dans un flux de traitement.

Du même point de vue, il faut en outre privilégier soit des logiciels intégrés regroupant un grand nombre de traitements et extensibles à l'aide de plugins, soit des logiciels permettant d'*échanger* leurs résultats avec d'autres et qui permettent l'export dans des formats graphiques qui se prêtent au post-traitement. Les formats d'échange sont hélas trop nombreux et l'effort de standardisation des grammaires par la communauté des développeurs est pour l'heure insuffisant. Dans l'état actuel, cela force l'usager à opter pour des solutions intégrées ne répondant, chacune séparément, que très partiellement aux besoins. La situation devrait s'améliorer au cours des prochaines années. Pour

l'heure, notre recommandation pour une cartographie générale de graphes discrets converge vers les solutions Cytoscape ou Gephi. Une perspective prometteuse, qui permettra peut-être d'offrir une solution à la fois intégrée et modulable, sont les assemblages de logiciels individuels de type CIShell (3.11).

Quelque soient les besoins, enfin, les logiciels multiplateforme sont à privilégier, dans la mesure où de larges réseaux de chercheurs sont mobilisées pour conduire une analyse. Dans la mesure où des équivalents open-source existent pour la plupart des logiciels payants, ils peuvent être privilégiés pour une cartographie générale de graphes discrets.

7 Bibliographie

Biggs, N., Lloyd, E. and Wilson, R. (1986), *Graph Theory, 1736-1936*, Oxford University Press.

Boyandin, I., Bertini, E., Bak, P., & Lalanne, D. (2011). Flowstrates: An Approach for Visual Exploration of Temporal Origin-Destination Data. *Eurographics / IEEE Symposium on Visualization*, *30*(3), 971–980.

Chartrand, G., Lesniak, L., & Zhang, P. (2010). *Graphs and digraphs* (5th ed., p. 586). CRC Press.

De Nooy, W., Mrvar, A., & Batagelj, V. (2005). *Exploratory Social Network Analysis with Pajek*. Cambridge University Press.

Doantam Phan (2005). Flow Map Layout Stanford University InfoVis 2005

Dorigo, G., & Tobler, W. (1983). Push-Pull Migration Laws. *Annals of the Association of American Geographers*, *73*(1), 1–17.

Eades, P. (1984). A heuristic for graph drawing. *Congressus Numerantium*, (42), 149–60.

Even, S. (2011). *Graph algorithms*. Cambridge University Press.

Fruchterman, T., & Reingold, E. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, *21*(11), 1129–1164.

Gansner, E. R., Hu, Y., & Kobourov, S. G. (2009). GMap: Drawing Graphs as Maps. *CoRR*, *abs/0907.2585*.

Gansner, E. R., Hu, Y., Kobourov, S. G., & Volinsky, C. (2009). Putting Recommendations on the Map -- Visualizing Clusters and Relations. *CoRR*, *abs/0906.5*.

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. *Proceedings of the 7th Python in Science Conference (SciPy2008)* (pp. 11–15). Pasadena, CA USA.

Harris, R. L. (1999). *Information Graphics: A Comprehensive Illustrated Reference. Visual Tools for Analyzing, Managing, and Communicating*. Oxford: Oxford University Press.

Holten, D. (2006). Hierarchical Edge Bundles, Visualization of Adjacency Relations in Hierarchical Data. *IEEE Trans. on Visualization and Computer Graphics*, *12*(5).

Hu, Y. (2005). Efficient and high quality force-directed graph drawing. *The Mathematica Journal*, *10*, 37–71.

Kamada, T., & Kawai, S. (1988). An Algorithm for Drawing General Undirected Graphs. *Information Processing Letters*, *31*, 7–15.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*(1), 59–69.

Koren, Y. (2005). Drawing graphs by eigenvectors: theory and practice. *Computers & Mathematics with Applications*, *49*(11–12), 1867–1888.

Krzywinski, M. I., Schein, J. E., Birol, I., Connors, J., Gascoyne, R., Horsman, D., Jones, S. J., et al. (2009). Circos: An information aesthetic for comparative genomics. *Genome Research*.

Krzywinski, M., Birol, I., Jones, S., & Marra, M. (2011). Hive Plots — Rational Approach to Visualizing Networks. *Briefings in Bioinformatics*.

Lamping, J., Rao, R., & Pirolli, P. (1995). A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. *Proc. ACM Conf. Human Factors in Computing Systems, CHI. ACM.* (pp. 401–408).

Leetaru, K. H. (2011). Culturomics 2.0: Forecasting Large-scale human behavior using global news media tone in time and space. *First Monday*, *16*(9). Retrieved from http://www.uic.edu/htbin/cgiwrap/bin/ojs/index.php/fm/rt/printerFriendly/3663/3040

Mayr, G. von. (1877). *Die Gesetzmäßigkeit im Gesellschaftsleben* (p. 78). München: Oldenbourg.

Moen, S. (1990). Drawing Dynamic Trees. *IEEE Software*, *7*(4), 21–28.

Ourednik, A. (2012). The Valaisan mobility network. *Maps and Spaces*. Retrieved from http://ourednik.info/maps/2012/04/05/the-valaisan-mobility-network/

Phan, D., Xiao, L., Yeh, R., Hanrahan, P., & Winograd, T. (2005). Flow Map Layout. *IEEE Information Visualization (InfoVis)*, 219–224.

Pons, P., & Latapy, M. (2005). Computing communities in large networks using random walks (long version). *ArXiv Physics e-prints*.

Purchase, H. C. (2000). Effective information visualisation: a study of graph drawing aesthetics and algorithms. *Interacting with Computers*, *13*(2), 147–162.

Reingold, E. M., & Tilford, J. S. (1981). Tidier Drawings of Trees. *IEEE Transactions on software engineering*, 223–228.

Shneiderman, B. (1992). Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, *11*(1), 92–99.

Sugiyama, K., Tagawa, S., & Toda, M. (1981). Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC-11*(2), 109–125.

Tobler, W. R. (1981). A Model of Geographical Movement. *Geographical Analysis*, *13*(1), 1–20.