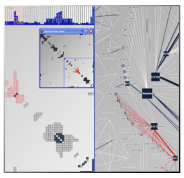
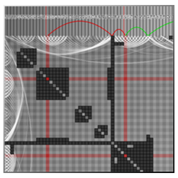
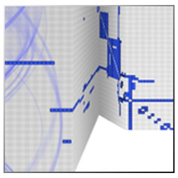
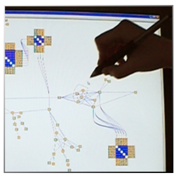
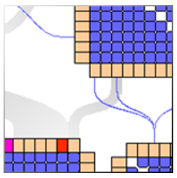
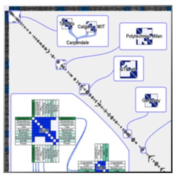
Exploring Social Networks with Matrix-based Representations

—

Explorer les réseaux sociaux avec des représentations matricielles de graphe

Nathalie Henry

Ph. D. in cotutelle
defended july 3, 2008
submitted for the degree of

Thèse en cotutelle N° 9076
présentée le 3 juillet 2008
pour l'obtention du grade de

DOCTEUR EN SCIENCES DE L'UNIVERSITÉ PARIS SUD, FRANCE
DOCTOR OF PHILOSOPHY OF THE UNIVERSITY OF SYDNEY, AUSTRALIA

Ph.D. supervisors — directeurs de thèse :

Pr. Jean-Daniel FEKETE	Directeur de recherche, INRIA Saclay, France
Pr. Peter EADES	Professor, University of Sydney, Australia

Thesis reviewers and doctoral committee members — rapporteurs:

Pr. Guy Melançon	Professor, University of Bordeaux & INRIA Bordeaux, France
Pr. Ulrik Brandes	Professor, University of Konstanz, Germany
Pr. Jarke van Wijk	Professor, Eindhoven University of Technology, Nederlands

Doctoral committee members — examinateurs:

Pr. Françoise Détienne	Directrice de recherche, INRIA Rocquencourt & ENST, France
Dr. Eric Lecolinet	Maître de conférence, ENST, France

UNIVERSITÉ PARIS-SUD 11 INRIA The University of Sydney

Le mot ordinateur était apparu en 1955 dans la langue française. À cette époque un constructeur de matériel américain, IBM, avait demandé à un professionnel de lettres de traduire l'expression "Electronic Data Processing Machine" qui veut dire "Machine Électronique de Traitement de Données". Le traducteur avait alors retenu le mot ORDINATEUR *parce qu'au Moyen Âge Dieu était le grand Ordinateur, "celui qui mettait de l'ordre dans le Monde". Il remettait en usage un terme inusité depuis six siècles. On peut s'interroger sur le bien-fondé d'une telle traduction.*

RAVALEC — *L'effacement progressif des consignes de sécurité*

Acknowledgments

My first thank goes to my supervisor and mentor Pr. Jean-Daniel Fekete. I deeply thank him for his guidance during these four years. Thanks to his support and patience, I learnt to gain confidence in my work and produce a high quality research. I greatly appreciated our long animated discussions, especially in airports boarding rooms, that boosted my creativity, sharpenned my critical thinking and trained my argumentative nature. I would like also to thank my co-supervisor Pr. Peter Eades and Dr. Masahiro Takatsuka, from University of Sydney, who made me discover the marvelous world of research and information visualization.

Today, I would not be a researcher without you three.

I also would like to thank the reviewers of this dissertation Pr. Guy Melançon, Pr. Ulrik Brandes and Pr. Jarke van Wijk who did the job twice (!) for university of Paris-Sud and university of Sydney. Many thanks to my examinators Dr. Eric Lecolinet and Pr. Françoise Détienne for their questions and comments. I would like to thank the whole jury and audience present at my defense for their kindness and smiles to my jokes attempts.

A very special thank you to all of my collaborators and co-authors, thanks to them, the work presented in this Ph.D. has a much higher coolness factor. I would like to thank especially Michael for his fantastic creativity leading to very sexy animations, and an enigmatic Yann, official movie director of all my research advertisements. I thank them both for listening to my conference talks dozens of times (always including the same jokes). I would like to thank my friend, Anastasia, for her kindness and her great skills at designing experiments! We had and are still having a lot of fun collaborating. I dearly value the high quality of her reasearch and her terrific personnality. My next big thank goes to my viking colleague Niklas, his dear wife Helen, my french bordelaise colleague Fanny and my always-zen colleague Tomer. I am thankfull for all of our discussions, crazy working evenings and week-ends and our successfull past, current and future collaborations.

I had amazing fun working with you guys.

I especially thank my office-sharer colleague Pierre, who patiently supported my constant oral rambling and private joking. I am thanking also all members of AVIZ and InSitu teams: Caroline for her good vibes, Emmanuel for our fruitful morning chats, Jean-Baptiste, Olivier C., Olivier B., Nicolas R., Nicolas M., Aurélien, Guillaume, Mathieu, Rémi, Emmanuel N., Pascal, Renaud, Stéphane, Fanis, Jérôme, Howard, Nghi and all the ones I forgot. A big thank to Neo, Gonzo all my friends from Lyon and Orsay and a special thank to Julien, brilliant cellist, who gave me a breath of fresh air between two sessions of hard-working research.

For the aussie side, I would like to thank Tim and Amelia for their warm welcome when I first discovered Australia, Damian and Kathrin, Ahmed and his family, Thomas, Joachim, Barbara, Lanbo, Kelvin, David, all the people I forgot and my dear housemates Linda M. and Linda B. I had a wonderful time in Oz thanks to all of you.

During my various conference experiences, I had the pleasure to meet talented and friendly researchers. Thank you Catherine for the fun we had sharing rooms. Thank you Sheelagh for our discussions and your guidance. Thank you Bongshin for you happy mood and friendship. Thank you Petra, Chris and all students from Calgary for the time we spent both discussing research and having fun. Thank you Ben, George and Frank for being infovis stars with a friendly aura.

My most special thanks goes to my family and my husband's family, who supported us during the past four years. Thank you Aline for your smile, good mood, fabulous creativity and constant chocolate supply. Thank you Jeannine and Daniel for beeing close while several thousands of kilometers away. My parents Andrée and Christian were the most precious support, helping me live through several flat moves, many conferences, many paper deadlines, a dissertation, a defense and more importantly a wedding.

During this Ph.D., I had many ups and many downs. I adopted a cat, started cello and discovered what I wanted to do with my life.

During this Ph.D., I met a lot of people, made new friends and brought the dear ones closer. Thank you Bénédicte and Sébastien for welcoming us into your family, thank you Jacques for the crazy horse riding and non-stop talking. Thank you all three for being who you are.

During this Ph.D., I met my soul mate, Yann. We lived through two Ph.Ds, both in cotutelle between France and Australia, we were separated (by 16 000 km) and reunited (24h a day, 7 days a week), we laughed and cried, got engaged in Brazil and got married, both as Ph.Ds. Thank you for being here. I love you.

Remerciements

Je tiens tout d'abord à remercier mon directeur de recherche et mentor Jean-Daniel Fekete. Je le remercie pour ces quatre années de soutien et de patience pendant lesquelles il a guidé mes premiers pas de chercheuse. Je retiendrai ses précieux conseils qui m'ont permis de prendre confiance en mon travail et de produire une recherche de qualité. Je me souviendrai particuliérement de nos discussions animées, surtout dans les salles d'attente des aéroports, qui m'ont permis de nourrir ma créativité, d'affiner mon sens critique et surtout d'affirmer mon esprit de contradiction. Je souhaite aussi remercier mon co-directeur de thèse Peter Eades ainsi que Masahiro Takatsuka, chercheur à l'université de Sydney, pour m'avoir fait découvrir le merveilleux monde de la recherche et de la visualisation d'information.

Sans vous trois, je ne serai pas aujourd'hui une chercheuse.

Je voudrais également remercier les rapporteurs de ce manuscrit de thèse Guy Melançon, Ulrik Brandes et Jarke van Wijk, qui ont fait ce travail deux fois (!) pour l'université Paris-Sud et l'université de Sydney. Merci également à mes examinateurs Eric Lecolinet et Françoise Détienne pour leurs questions et commentaires constructifs. Je tiens à remercier mon jury de thèse ainsi que toutes les personnes venues assiter à ma soutenance pour leur présence et leurs chaleureux sourires lors de mes escapades humoristiques.

Un merci très spécial à tous mes collaborateurs et co-auteurs. Grâce à eux, les travaux présentés dans cette thèse sont bien plus fashion. Je remercie particulièrement Michael, qui a su concevoir un certain nombre d'animations très sexy et un certain Yann, réalisateur officiel de tous les spots publicitaires de mes travaux de recherche. Je voudrais aussi vous remercier tout deux, d'avoir eu la patience d'écouter mes présentation orales plusieurs dizaines de fois (et toujours avec les mêmes blagues). J'aimerai aussi remercier ma collaboratrice et amie, Anastasia, pour sa générosité et surtout son don exceptionnel pour concevoir et analyser des expériementations contrôlées dans la joie et la bonne humeur! La série de remerciements qui suit est destinée à mon collègue viking Niklas, sa chère femme Helene, ma petite collègue bordelaise Fanny ainsi que mon collègue super-zen Tomer. Je vous remercie pour nos discussions fructueuses, nos folles soirées et week-ends de boulot ainsi que nos brillantes collaborations passées, présentes et à venir.

Travailler avec vous tous fut un véritable bonheur, c'était trop la fête!

J'aimerai addresser un merci tout spécial à mon collègue Pierre, qui a patiemment supporté mes constantes divagations, cohabitant dans le même bureau que moi. Je remercie aussi bien sûr tous les membres passés et présents des équipes AVIZ et InSitu: Caroline pour ses bonnes vibrations, Emmanuel pour nos conversations fertiles du levée du jour, Jean-Baptiste, Olivier C., Olivier B., Nicolas R., Nicolas M., Aurélien, Guillaume, Mathieu, Rémi, Emmanuel N., Pascal, Renaud, Stéphane, Fanis, Jérôme, Howard, Nghi et tous ceux que j'ai oublié. Merci aussi á Neo, Gonzo ainsi que tous mes amis lyonnais et orsayiens et merci à Julien, violoncelliste de talent, pour m'avoir permis de prendre quelques bouffées d'air entre deux séances de recherche à plein temps.

Du côté australien, je voudrais remercier Tim et Amélia, qui m'ont chaleureusement accueilli lors de mon premier séjour en Australie, Damian et Kathryn, Ahmed et sa famille, Thomas, Joachim, Barbara, Lanbo, Kelvin, David, tous les gens que j'oublie et mes très chères co-locatrices Linda M. et Linda B. J'ai passé des moments fantastiques en Australie grâce à vous tous.

Mes expéditions à de multiples conférences m'ont permis de rencontrer de talentueux et sympathiques chercheurs. Merci Catherine pour les bons moments passées dans les hotels***. Merci Sheelagh pour toutes nos discussions et tes précieux conseils. Merci Bongshin pour ta bonne humeur et ton amitié. Merci à Petra, Chris et tous les étudiants de Calgary pour le temps agréable passé à discuter et rire. Enfin, merci Ben, George et Frank pour être des stars de l'infovis avec un grand cœur!

Mes remerciements les plus chaleureux vont à ma famille et à celle de mon mari, qui nous ont soutenu pendant ces quatre longues années. Merci Aline pour ton sourire, ta bonne humeur créative et ton approvisionnement constant en chocolat. Merci Jeannine et Daniel pour avoir été présent même à des milliers de kilomètres de nous. Mes parents Andrée et Christian m'ont apporté un soutien des plus précieux, m'aidant à survivre à plusieurs déménagements, beaucoup de conférences et séjours à l'étranger, une bonne tonne d'articles de recherche, la rédaction de ma thèse, ma soutenance et le plus important mon mariage.

Pendant cette thèse, j'ai eu beaucoup de hauts et beaucoup de bas. J´ai adopté un chat, commencé le violoncelle et découvert ce que je voulais faire de ma vie.

Pendant cette thèse, j'ai rencontré un sacré paquet de gens. J'ai eu la chance de découvrir de nouveaux amis et de me rapprocher de ceux qui m'étaient chers. Merci Bénédicte et Sébastien pour nous avoir accueilli dans votre famille. Merci Jacques pour nos folles expéditions équestres et nos multiples fous rires. Merci à vous trois d'être qui vous êtes.

Pendant cette thèse, j'ai rencontré mon âme sœur, Yann. Nous avons survécu à deux thèses, toutes deux en cotutelle entre la France et l'Australie, nous avons été séparé (par 16 000km) et réunis (24h par jour, 7 jours par semaine), nous avons ri et pleuré, nous nous sommes fiançés au Brésil et nous sommes mariés, tous deux en tant que DR. Merci d'être à mes côtés. Je t'aime.

Un bon croquis vaut mieux qu'un long discours
— Napoléon Bonaparte

Préambule: des représentations visuelles alternatives pour les réseaux sociaux

Grâce à la démocratisation des technologies Internet, il devient plus facile de collecter de grandes quantités de données et d'extraire de grands réseaux sociaux afin de les étudier. Par exemple, au travers de sites tels que Facebook ou Friendster, les réseaux sociaux disponibles sont très vastes (des millions d'utilisateurs), riches (contenant de multiples relations ou informations sur les acteurs du réseau) et évoluent dans le temps.

Si la collecte des données est grandement simplifiée, le problème de l'analyse se pose de manière aigue. Plusieurs méthodes et outils existent pour analyser les réseaux sociaux. La majorité de ces systèmes sont basés sur des méthodes statistiques et fournissent un grand nombre de fonctionnalités d'analyse et de modélisation. UCINet [1] est un bon exemple de ce type de logiciel, permettant de calculer un grand nombre de mesures, comme les différents types de centralité, ou bien de vérifier si le jeu de données correspond à un modèle statistique donné. De nombreux autres outils existent et sont présentés plus en détail dans l'article de Huisman et Duijn [2].

Dès les années 1930 sont apparues des représentations visuelles de réseaux sociaux. Par exemple, Jacob Moreno a été un des pionniers à utiliser la représentation nœud-lien pour communiquer sur ses travaux (Figure 1). Dans cette représentation, un nœud représente un acteur et un lien représente une relation du réseau. Si les visualisations ont très souvent été utilisées dans ce cadre, i.e., communiquer des résultats ou illustrer une théorie, elles peuvent aussi servir à analyser les données. Le statisticien Tukey, en introduisant l'Analyse Exploratoire des Données [3], démontre que la représentation graphique peut être un outil puissant d'analyse. Son intérêt réside dans l'observation de multiples représentations de données brutes, qui permet de soulever des questions ou établir des hypothèses qui n'auraient pas été imaginées a priori. Cette utilisation de la représentation visuelle pour l'exploration a donné lieu à un domaine de recherche à part entière: la visualisation d'information.

[1]UCINet. voir `http://www.analytictech.com/`

[2]Huisman, M. et M. Duijn (2005). Software for social network analysis (chapter 13). In V. de Nooy, A. Mrvar (Ed.), *Exploratory Social Network Analysis with Pajek*.

[3]Tukey, J. (1977). *Exploratory Data Analysis*. Addison-Wesley.

La représentation la plus courante des graphes ou des réseaux (dans la suite de cette préface nous utiliserons ces deux mots indifféremment) est la représentation nœud-lien. Par exemple, en parcourant le répertoire de l'International Network for Social Network Analysis (`http://www.insna.org`), il est possible de compter plus de cinquante systèmes basés sur les diagrammes nœud-lien. Si cette représentation a donc l'avantage d'être familière à la majorité des chercheurs, elle souffre de problèmes de lisibilité lorsque les réseaux représentés sont soit grands (beaucoup de nœuds), soit denses (beaucoup de liens). Ces problèmes de passage à l'échelle sont d'autant plus importants que les données à analyser sont de plus en plus nombreuses ; ils sont devenus rédhibitoires avec l'apparition des réseaux sociaux en ligne qui sont à la fois grands et denses.

Dans les cinq dernières années, le domaine de la visualisation d'information a connu plusieurs avancées et a permis de trouver des représentations alternatives (et complémentaires) aux représentations noeud-lien. Dans cette préface, nous faisons un état de l'art de ces nouvelles représentations visuelles des réseaux, nous focalisant sur les représentations basées sur les matrices d'adjacence de graphe présentées en détail dans cette thèse.

Représentations Noeud-Lien

L'analyse des réseaux sociaux a débuté il y a plus de 70 ans, avec les travaux empiriques de Jacob Moreno [4]. Wasserman et Faust [5] présentent les diverses catégories de méthodes : analyses statistiques, structurelles et exploratoires. Freeman [6] retrace l'historique des visualisations de réseaux sociaux et montre que les représentations visuelles peuvent être un outil efficace pour illustrer des concepts tels que les acteurs centraux ou les groupes sociaux. Les Figure 1 et Figure 2 présentent deux exemples utilisant les diagrammes nœud-lien.

Le très grand avantage des diagrammes nœud-lien est leur intuitivité : la grande majorité des lecteurs peut les comprendre. En revanche, qu'ils soient dessinés manuellement (Figure 1) ou générés automatiquement (Figure 2), leur lisibilité dépend totalement du placement des nœuds dans le plan. Ce problème épineux a d'ailleurs donné naissance à un domaine de recherche à part entière nommé le dessin de graphes (graph drawing).

Dessin de graphes et visualisation d'information

L'ouvrage de di Battista et al. [7] constitue une bonne introduction au domaine du dessin de graphes, présentant plus de trois cents algorithmes. Un état de l'art de la visualisation de graphes [8], plus récent, recense également un grand nombre de techniques, parfois interactives, pour représenter les graphes et les réseaux. En général, ces techniques tentent d'optimiser un certain nombre de critères esthétiques, comme, par exemple, limiter le nombre de croisements des liens ou bien placer les nœuds pour avoir une longueur uniforme de tous les liens. Plusieurs études

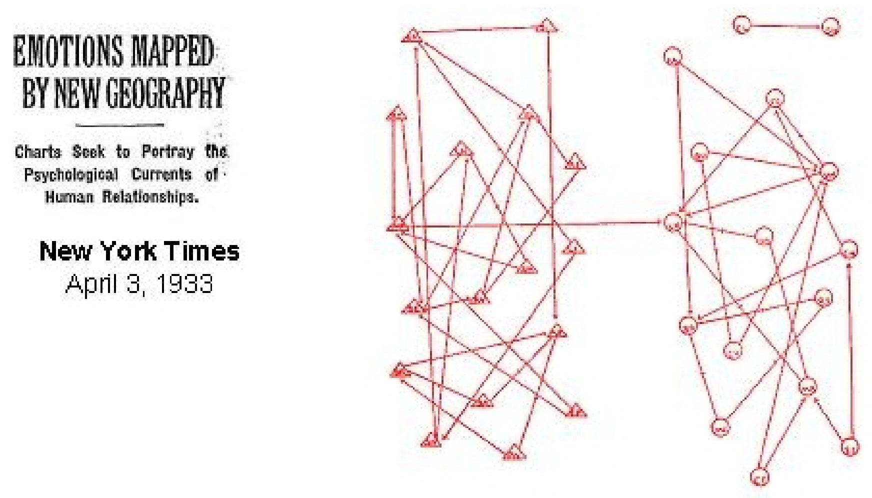
[4]MORENO, J. (1934). *Who shall survive ?*. Nervous and Mental Disease Publishing Company

[5]WASSERMAN, S. ET K. FAUST (1994). *Social Network Analysis*. Cambridge Univ. Press

[6]FREEMAN, L. (2000). Visualizing social networks. In *Journal of social structure*, 1(1)

[7]DI BATTISTA, G., P. EADES, R. TAMASSIA, ET I. G. TOLLIS (1998). *Graph Drawing : Algorithms for the Visualization of Graphs*. Prentice Hall.

[8]HERMAN, I., G. MELANÇON, ET S. MARSHALL (2000). Graph visualization and navigation in information visualization : A survey. In *IEEE TVCG*, 6(1)



EMOTIONS MAPPED BY NEW GEOGRAPHY

Charts Seek to Portray the Psychological Currents of Human Relationships.

New York Times
April 3, 1933

Figure 1: Réseau d'amitiés entre garçons (triangles) et filles (cercles) par J. Moreno. Un acteur central lie les deux groupes (triangle du milieu gauche).

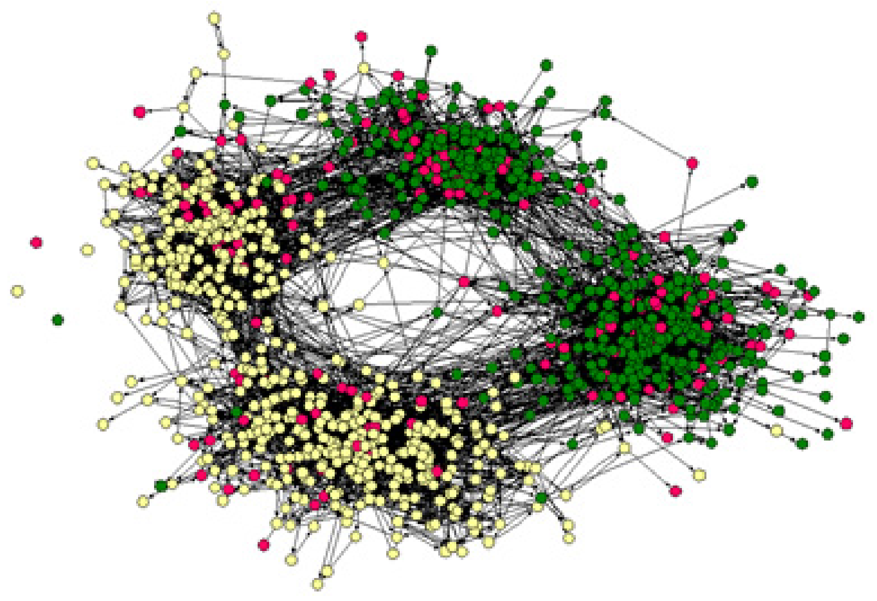


Figure 2: Réseau d'amitiés entre lycéens par J. Moody, la couleur marque l'origine ethnique. Quatre groupes sociaux émergent après application d'un algorithme de clustering (clusters par origine ethnique et âge).

ont été réalisées pour déterminer quels critères sont les plus importants et ainsi tenter de découvrir le meilleur algorithme de dessin de graphe. Cependant, il existe un tel nombre de critères qu'il est difficile de former des conclusions générales.

Le domaine de la visualisation d'information a une perspective différente sur le sujet. Ce domaine de recherche se focalise sur l'exploration visuelle et la découverte ou la communication d'informations. Ainsi, les réseaux présentés en Figure 1 et Figure 2 n'optimisent certainement pas des critères esthétiques, comme la minimisation du nombre de croisements de liens, mais ils permettent de mettre en lumière certaines informations importantes sur la structure du réseau. Dans cet article, nous présentons une série de représentations alternatives aux diagrammes nœud-lien, non pas pour les remplacer mais plutôt pour les compléter. Dans le contexte de l'analyse exploratoire, nous pensons qu'il est nécessaire d'avoir plusieurs perspectives sur les données pour en découvrir toutes les arcanes.

Exploration de réseau et passage à l'échelle

Un problème récurrent se pose quand le réseau à représenter contient beaucoup de nœuds ou de liens : le diagramme nœud-lien se transforme en un amas de traits et de points difficile voire impossible à transformer en représentation lisible, ni manuellement ni automatiquement. Dans la littérature, il existe trois techniques distinctes pour tenter de résoudre ce problème :

1. *Réduire la quantité d'information représentée par filtrage ou agrégation.* Il existe plusieurs méthodes pour échantillonner les réseaux [9] ou bien calculer des clusters [10] (groupe d'éléments similaires), qui peuvent être ensuite agrégés en un élément représentatif du groupe.

2. *Proposer une représentation interactive pour explorer la totalité du réseau.* Plusieurs outils de visualisation ne représentent qu'une partie du réseau et proposent de naviguer interactivement pour explorer le reste du réseau. Les stratégies varient : certaines se concentrent sur un nœud particulier et permettent la navigation en suivant les connexions [11] [12], d'autres sont basées principalement sur du filtrage interactif [13] [14] ou bien permettent la navigation dans un réseau agrégé [15].

3. *Utiliser des représentations alternatives aux diagrammes nœud-lien.* Cette stratégie consiste à utiliser différentes métaphores visuelles pour la représentation de réseaux. L'idée est de découvrir des représentations permettant d' "augmenter l'espace visuel", permettant d'afficher un plus grand nombre d'information de façon plus lisible.

[9]FRANK, O. (1978). Sampling and estimation in large social networks. In *social networks*, 1

[10]JAIN, A., M. MURTY, ET P. FLYNN (1999). Data clustering : a review. In *ACM Computing Surveys*, 31(3)

[11]HEER, J. ET D. BOYD (2005). Vizster : Visualizing Online Social Networks. In Proceedings of IEEE Infovis'05

[12]LEE, B., C. PARR, C. PLAISANT, B. BEDERSON, V. VEKSLER, W. GRAY, ET C. KOTFILA (2006). Treeplus : Interactive exploration of networks with enhanced tree layouts. In *IEEE TVCG (Infovis'06 proceedings)* 12(6)

[13]PERER, A. ET B. SHNEIDERMAN (2006). Balancing Systematic and Flexible Exploration of Social Networks. In *IEEE TVCG (Infovis'06 proceedings)*, 12(5)

[14]SHNEIDERMAN, B. ET A. ARIS (2006). Network visualization by semantic substrates. In *IEEE TVCG (Infovis'06 proceedings)*, 12(5)

[15]ABELLO, J., F. CAN HAM ET N. KRISHNAN (2006). Ask-graphview : A large scale graph visualization system. In *IEEE TVCG (Infovis'06 proceedings)*, 12(5)

La psalmodie de la visualisation d'information [16] est : "vue d'ensemble d'abord, zoomer et filtrer, puis détails à la demande" (overview first, zoom and filter, then details on demand). Cette phrase résume le processus d'exploration visuel. Ainsi, il est primordial de présenter une vue d'ensemble du réseau afin d'initier le processus d'exploration. Les deux premières stratégies, visant à supprimer l'information ou à n'en présenter qu'une partie, ne sont donc pas satisfaisantes. Nous nous sommes intéressés à la troisième solution : utiliser des représentations alternatives.

Représentations alternatives

3D et espaces géométriques non-Euclidiens

Afin d'augmenter l'espace visuel, un certain nombre de chercheurs ont tenté d'utiliser la 3ème dimension. L'idée est qu'une troisième dimension permet d'avoir plus d'espace pour dessiner de plus grands réseaux mais aussi permet d'optimiser certains critères esthétiques tels que la minimisation du nombre de croisements. Plusieurs programmes utilisent la 3D, la Figure 4 (gauche) montre un exemple créé avec la boîte à outils de dessin de graphes Tulip [17].

Le principal problème lié à l'utilisation de la 3D est l'occlusion : les utilisateurs ont l'impression que les représentations sont " encombrées " et trouvent difficile d'avoir une représentation mentale de la totalité du graphe [18]. Pour palier à ces problèmes, plusieurs stratégies ont été adoptées [19], comme l'utilisation de multiples vues ou des techniques de navigation pour voir la représentation sous de multiples angles. Malheureusement, dans la majorité des cas, ces techniques désorientent l'utilisateur et rendent les visualisations peu efficaces [20]. Plusieurs études ont montré que, si les visualisations 3D sont attractives, elles n'améliorent pas les performances et les détériorent même pour certaines tâches [21] .

Une autre façon d'augmenter l'espace visuel est l'utilisation de la géométrie hyperbolique au lieu de la géométrie Euclidienne [22]. Dans l'espace hyperbolique, le postulat du parallélisme est rejeté : deux lignes parallèles dans l'espace euclidien divergent l'une de l'autre. Donc, si l'on considère un disque dans l'espace hyperbolique, l'espace augmente exponentiellement lorsque l'on s'éloigne de son centre. Ainsi, un réseau dessiné sur ce disque dispose d'un espace infini sur ses bords, ce qui permet d'afficher plus d'information, comme le montre Figure 4 (droite). La métaphore s'applique aussi à la 3D [23]. Malheureusement, similairement à la 3D, naviguer dans un espace géométrique non euclidien désoriente l'utilisateur. Il est de plus difficile de se constituer une carte mentale du réseau sans recours à la géométrie 2D euclidienne [24].

[16]SHNEIDERMAN, B. (1996). The eyes have it: a task by data taxonomy for information visualization. In *Visual Languages*

[17]AUBER, D. (2003). Tulip : A huge graph visualisation framework. In *Graph Drawing Software*. Springer-Verlag

[18]COCKBURN, A. ET B. MCKENZIE (2002). Evaluating the effectiveness of spatial memory in 2D and 3D physical and virtual environments

[19]ELMQVIST, N. ET P. TSIGAS (2007). Taxonomy of 3D occlusion management techniques. *Proceedings of VR'07*

[20]SUTCLIFFE, A. ET U. PATER (1996). 3D or not 3D : is it nobler to the mind ? In *British HCI Conference*

[21]COCKBURN, A. ET B. MCKENZIE (2000). An evaluation of cone trees. In *People and Computers XV*

[22]LAMPING, J. ET R. RAO (1996). The Hyperbolic Browser : A focus + context technique for visualizing large hierarchies. In *Journal of Visual Languages and Computing*, 7 (1)

[23]MUNZNER, T. (1997). H3 : Laying out large directed graphs in 3d hyperbolic space. In *Proceedings of Infovis'97*

[24]RISDEN, K., M. CZERWINSKI, T. MUNZNER, ET D. COOK (2000). An initial examination of ease of use for 2D and 3D information visualizations of web content. In *IJHCS*, 53(5)

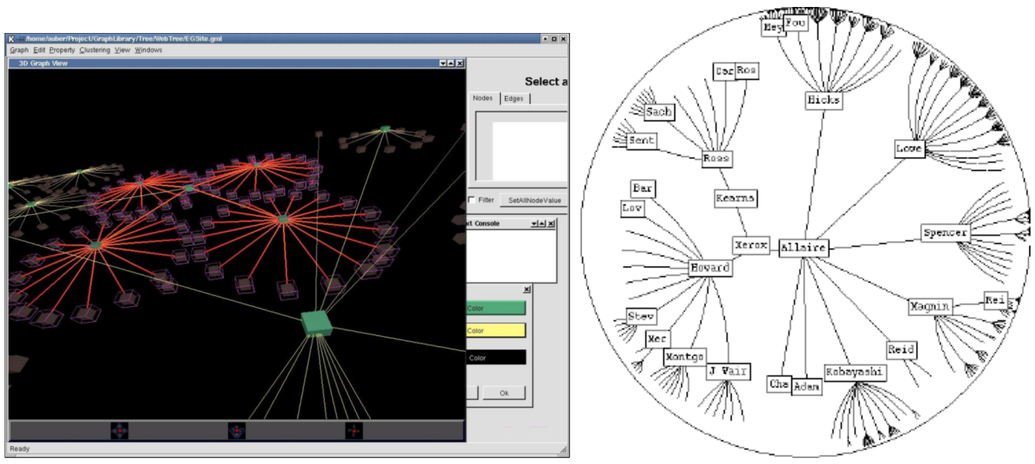


Figure 3: Représentation d'un réseau en 3D (à gauche) et dans l'espace hyperbolique (à droite).

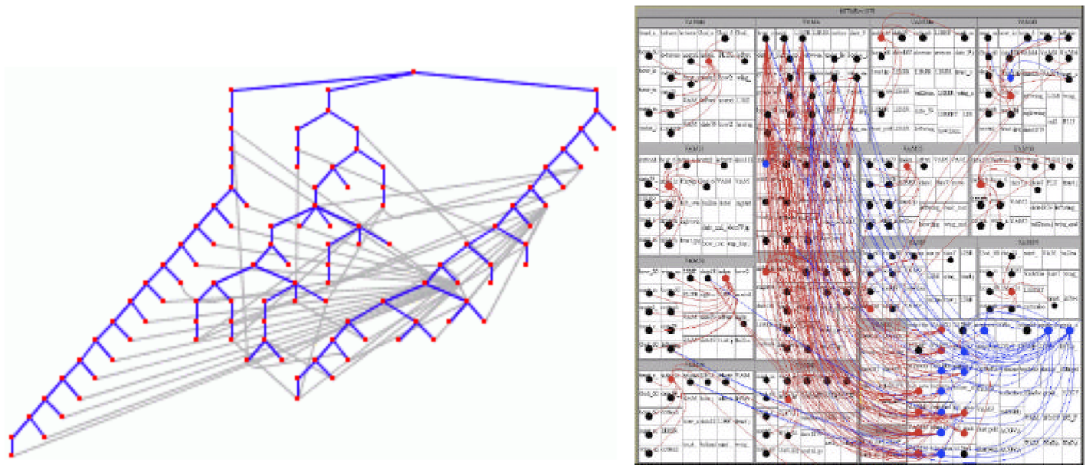


Figure 4: Représentation d'un réseau sous forme d'arbre avec des liens supplémentaires (à gauche) et sous forme de Treemap avec des liens supplémentaires (à droite).

Représenter un graphe par un arbre

Une autre stratégie pour représenter des réseaux peu denses et augmenter leur lisibilité est de les représenter sous forme d'arbres et de le "réparer" avec des connexions supplémentaires. Les algorithmes de dessin d'arbres permettent effectivement d'ôter tous croisements voire d'exploiter au maximum l'espace utilisé si l'arbre est dessiné en utilisant des inclusions de formes.

Pour dessiner un réseau sous forme d'arbre, il est nécessaire de calculer un arbre recouvrant (généralement on prend le plus grand), de dessiner cet arbre puis d'ajouter les liens supplémentaires [25]. Un exemple est présenté dans la Figure 3 (gauche). Cette technique fonctionne bien pour des réseaux ayant une structure différant peu de celle d'un arbre tels que les arbres généalogiques par exemple, mais la représentation devient illisible pour des réseaux plus denses.

Une alternative est la représentation sous forme de Treemap [26]. Le principe est identique au précédent, mais la représentation utilisée pour l'arbre recouvrant diffère : il s'agit de découper l'espace successivement pour représenter la hiérarchie (imbrications de boîtes). Par exemple, un arbre très simple composé d'une racine, de deux nœuds et de quatre feuilles sur chacun d'eux, est alors représenté par un rectangle occupant tout l'espace visuel, contenant deux rectangles de taille identique (représentant les nœuds), eux-mêmes sub-divisés en quatre rectangles de taille identique. Les liens supplémentaires du réseau sont alors dessinés par dessus le Treemap [27] comme montré sur la Figure 3 (droite). Cette représentation devient également difficile à lire si le nombre de liens supplémentaires est important.

L'ajout d'interaction est une piste intéressante pour améliorer la lisibilité de ces représentations. Elle est particulièrement bien exploitée dans TreePlus [28], un système visualisant une portion du réseau sous forme d'arbre et permettant de naviguer interactivement dans le reste. Afin de résoudre le problème des cycles, dans cette représentation sous forme d'arbre, des nœuds sont dupliqués. Si ce système permet de mieux réaliser un certain nombre de tâches, il rend difficile la constitution d'une carte mentale de la totalité du réseau.

Représentation matricielle

Les graphes ont deux représentations canoniques : les diagrammes nœud-lien et les matrices d'adjacence. Une matrice d'adjacence représente chaque sommet d'un réseau à la fois comme une ligne et comme une colonne. Si deux sommets sont connectés, la case correspondant à l'intersection de la ligne et de la colonne est marquée. Traditionnellement, on utilise une valeur numérique (0 marquant l'absence de connexion, 1 marquant la présence), soit par une marque graphique comme dans la Figure 5.

[25] HERMAN, I., G. MELANÇON, M. DE RUITER, ET M. DELEST (1999). Latour — A tree visualisation system. In *Proceedings of Graph Drawing 1999*, LNCS 1731

[26] SHNEIDERMAN, B. (1992). Tree visualization with tree-maps : A 2-D space-filling approach. In *ACM Transactions on Graphics*, 11 (1)

[27] FEKETE, J.-D., D. WANG, N. DANG, ET C. PLAISANT (2003, October). Overlaying graph links on treemaps. In *Proceedings Compendium of IEEE Infovis'03*

[28] LEE, B., C. PARR, C. PLAISANT, B. BEDERSON, V. VEKSLER, W. GRAY, ET C. KOTFILA (2006). Treeplus : Interactive exploration of networks with enhanced tree layouts. In *IEEE TVCG (Infovis'06 proceedings)* 12 (6)

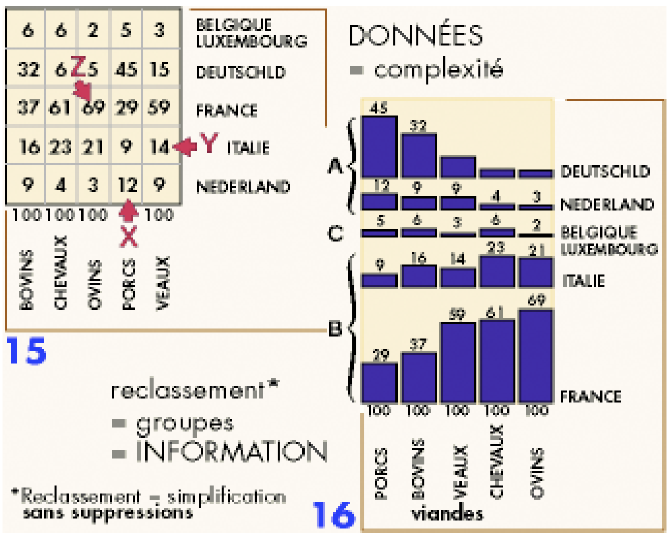


Figure 5: Matrice réordonnable de J. Bertin, extraite de la Sémiologie graphique

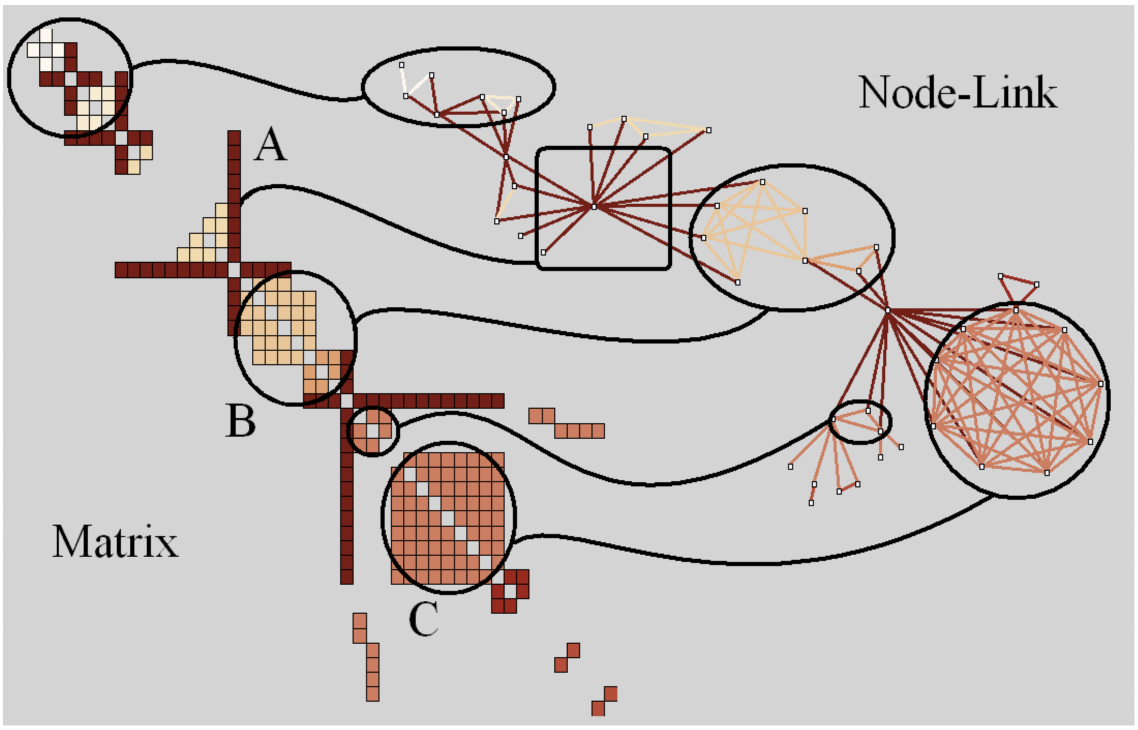


Figure 6: Correspondance entre les motifs visuels des matrices et ceux des diagrammes nœud-lien : A montre un acteur connectant plusieurs communautés, B deux communautés et C une clique

Nous listons ci-dessous les principaux avantages et inconvénients des représentations matricielles par rapport aux diagrammes nœud-lien.

+ Absence de superpositions des nœuds, ce qui permet de pouvoir toujours lire les étiquettes portées par les nœuds;
+ Absence de croisements des liens, ce qui permet de toujours identifier la source et destination des connexions;
+ Facilité avec laquelle il est possible d'identifier les absences de connexions;

- Taille de l'espace visuel requis à un niveau de détail équivalent plus important que le diagramme nœud-lien;
- Difficulté à effectuer des tâches de suivi de chemin (suivre les liens de Pierre à Paul passant par Marie);
- Manque de familiarité, les matrices paraissent moins intuitives que les diagrammes nœud-lien.

Plus de détails sur les performances des deux représentations de réseaux sont présentés dans les travaux de Ghoniem et al [29]. Il s'agit d'une comparaison plus quantitative de la lisibilité des représentations matricielles et des représentations nœud-lien pour des graphes de différentes tailles et densités. Dans cette étude, une liste de tâches de base a été utilisée pour évaluer la lisibilité des représentations : estimer le nombre de nœuds, estimer le nombre de connexions, trouver le nœud le plus connecté, trouver le nœud portant une étiquette donnée, trouver la connexion entre deux nœuds donnés, trouver un nœud commun à deux nœuds donnés et trouver un chemin entre deux nœuds donnés.

De manière générale, les résultats de leur expérimentation montrent que les nœud-lien sont plus efficaces pour les graphes de petite taille et peu denses alors que les matrices ont de meilleures performances dès que les graphes sont plus grands (>20 nœuds !) et plus denses pour toutes les tâches excepté le suivi de chemin. Ces résultats ainsi que les pros et cons listés précédemment révèlent le vaste potentiel de le représentation matricielle pour explorer les réseaux sociaux.

Utiliser la représentation matricielle

Deux facteurs principaux entrent en jeu lors qu'il s'agit d'utiliser les représentations matricielles pour explorer de grands réseaux : être capable de *réordonner leurs lignes et colonnes* et pouvoir *naviguer dans des matrices de grande taille.*

Réordonner les matrices

Dans son livre faisant référence en visualisation d'information [30], Bertin présente la matrice réordonnable. Au travers de plusieurs exemples, il démontre que réordonner les lignes et colonnes d'une matrice permet d'améliorer significativement sa compréhension. Prenons pour exemple la Figure 5 extraite de ce livre; la matrice présentée contient seulement cinq lignes et cinq colonnes représentant la production de cinq types de viande dans cinq pays.

[29] GHONIEM, M., J.-D. FEKETE, ET P. CASTAGLIOLA (2005). On the readability of graphs using node-link and matrix-based representations : a controlled experiment and statistical analysis. In *Information Visualization*, 4(2)

[30] BERTIN, J. (1967). *Sémiologie graphique : Les diagrammes - Les réseaux - Les cartes* (Les réimpressions ed.). Paris, France : Editions de l'Ecole des Hautes Etudes en Sciences

En transformant la matrice originale, contenant des valeurs numériques, en matrice visuelle, codant les valeurs par des formes géométriques de différentes tailles et en réordonnant “ intelligemment ” ses lignes et ses colonnes, il devient beaucoup plus simple d’interpréter ce jeu de données. En effet, il est pratiquement immédiat d’observer que la France est le pays ayant la plus grande production, tout type de viande confondu ; que la Belgique a la plus petite et qu’il existe trois différents profils de production (marqué par A, B et C dans la figure).

Pour aller une étape plus loin dans l’interprétation de ce simple exemple, imaginons qu’une loi doive être votée pour limiter la production d’un type de viande produit en grande quantité par les pays. Disposant de ces données et de la représentation visuelle associée, un analyste pourra rapidement observer que la Belgique est le pays a convaincre de ne pas accepter cette loi, car son profil de production est plutôt neutre alors que les autres pays voterons dans leur propre intérêt, contraire à celui de l’autre groupe.

Cet exemple démontre l’importance du réordonnancement des matrices: si les représentations matricielles dont les lignes et les colonnes sont ordonnées aléatoirement sont lisibles, une fois réordonnées elles deviennent réellement exploitables.

Algorithmes de réordonnancement

Effectuer un état de l’art des techniques de réordonnancement automatique est un véritable défi car un grand nombre de problèmes sont liés à la permutation des lignes et colonnes. Par exemple, un algorithme de linéarisation de graphe (alignement de tous les sommets d’un graphe de façon à minimiser un critère esthétique tel que le nombre de croisements) ou bien un algorithme d’optimisation de tables de calculs peut être utilisé pour réordonner une matrice d’adjacence. Les algorithmes sont donc disséminés dans divers domaines tels que la biologie ou le dessin de graphe; et leurs critères de qualité varient en fonction du contexte d’utilisation.

Plusieurs outils tels que PermutMatrix [31] ou VisuLab [32] offrent une représentation visuelle des matrices (ou de tables) et permettent d’expérimenter avec diverses techniques de réordonnancement et leurs paramètres. Nous ne détaillerons pas les diverses catégories de méthodes dans ce résumé, mais il est important de comprendre que la méthode de réordonnancement sélectionnée a un fort impact sur l’interprétation de la représentation, similairement aux méthodes de placement de nœuds des diagrammes nœud-lien. La Figure 6 montre un exemple de motifs visuels émergeant lorsque l’ordonnancement de matrice et le placement du nœud-lien sont optimisés.

Naviguer dans de grandes matrices

à niveau de détail équivalent, la représentation matricielle utilise plus d’espace que les diagrammes nœud-lien. Par exemple, sur un écran de 17 pouces, il est possible d’afficher seulement une centaine de nœuds si l’on souhaite pouvoir lire confortablement leurs noms. Ainsi, afin de tirer parti du potentiel des matrices, il est nécessaire d’avoir des outils efficaces de navigation.

[31]CARAUX, G. ET S. PINLOCHE (2005). Permutmatrix : A graphical environment to arrange gene expression profiles in optimal linear order. In *Bioinformatics*, 21

[32]VisuLab. voir `http://www.inf.ethz.ch/personal/hinterbe/Visulab`

De nombreuses techniques existent dans le domaine de l'Interaction Homme-Machine pour permettre de se déplacer dans de grands espaces visuels et/ou à divers niveaux de détails. Par exemple, les techniques de Focus+Context [33] telles que la vue à vol d'oiseau (bird's eye view) permettent d'avoir une vue d'ensemble miniature et une vue zoomée d'une partie de la représentation. Ces techniques ne sont pas directement utilisables dans notre contexte, elles requièrent quelques aménagements. Il est notamment indispensable de toujours voir les étiquettes des lignes et colonnes des matrices et de faciliter les transitions entre les divers niveaux de détails pour éviter à l'analyste de se perdre lors de l'exploration. Plusieurs systèmes intègrent ces outils de navigation tels que MatrixZoom [34] et la matrice d'appels multi-niveaux présentée par van Ham [35].

Combiner matrices et noeud-lien

Les représentations matricielle et nœud-lien ont chacune des avantages et inconvénients. Dans cette section, nous présentons notre système MatrixExplorer [36] [37] qui utilise une représentation duale des réseaux et tente de combiner le meilleur de deux mondes.

Initier l'exploration

L'avantage majeur des matrices par rapport aux diagrammes nœud-lien est de permettre d'avoir une vue d'ensemble du réseau toujours exploitable et ce, même pour de très grands graphes, alors que les diagrammes nœud-lien ne montrent souvent qu'un amas de liens et de nœuds superposés.

Prenons l'exemple d'un réseau social contenant les données d'échanges de courrier électronique entre plus de 450 personnes pendant un an (les sommets sont les personnes et les liens sont les courriers). La Figure 7 montre deux représentations de ce réseau. Alors que la représentation nœud-lien (utilisant un algorithme de placement traditionnel par forces) permet éventuellement de former l'hypothèse que le réseau est très dense; la représentation matricielle, elle, donne beaucoup plus d'information. Pour le démontrer, analysons brièvement la matrice présentée à la Figure 7.

Dans cette représentation, chacun des points noirs représente une connexion entre une ligne et une colonne (un lien dans le réseau), le gris marquant donc l'absence de connexion. Tout d'abord, on peut clairement rejeter l'hypothèse précédente : le réseau social n'est pas très dense car on peut observer une majorité de gris dans cette matrice. Deuxièmement, si l'on regarde de plus près la représentation, on peut voir plusieurs blocs de points noirs. Il s'agit de clusters : des groupes de recherche dont les membres échangent des courriers électroniques les uns avec les autres. Une autre observation permet de voir une sorte de croix : une ligne constituée de points noirs verticale et une horizontale (à peu près de la longueur d'une moitié de la matrice). Il s'agit d'un membre du service des missions de cet institut, celui-ci échange effectivement des courriers électroniques avec beaucoup des membres d'équipes, il a donc un grand nombre de connexions.

[33] CARPENDALE, M. S. T. (1999). *Framework for Elastic Presentation Space*. Ph. D. thesis, Simon Fraser Univ.

[34] ABELLO, J. ET F. VAN HAM (2004). Matrix zoom : A visual interface to semi-external graphs. In *Proceedings of IEEE Infovis'04*

[35] VAN HAM, F. (2003). Using multilevel call matrices in large software projects. In *Proceedings of IEEE Infovis'03*

[36] HENRY, N. ET J.-D. FEKETE (2006). MatrixExplorer : a Dual-Representation System to Explore Social Networks. In *IEEE TVCG (Infovis'06 proceedings)*, 12(5)

[37] HENRY, N. ET J.-D. FEKETE (2006). Matrixexplorer : Un système pour l'analyse exploratoire de réseaux sociaux. In *Proceedings of IHM2006*

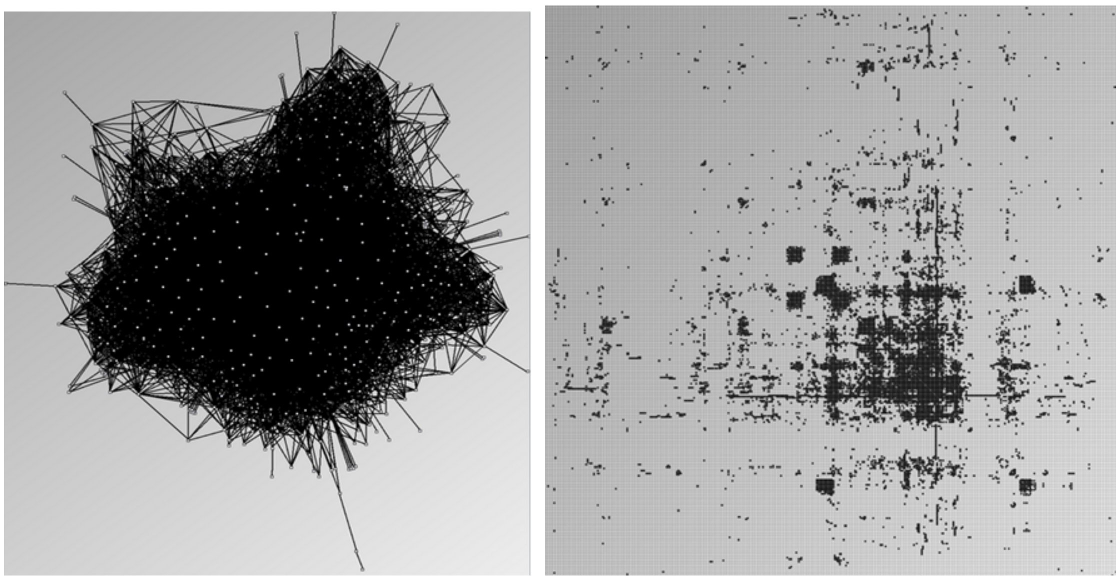


Figure 7: Réseau d'échange de courriers électroniques entre plus de 450 chercheurs, pendant un an

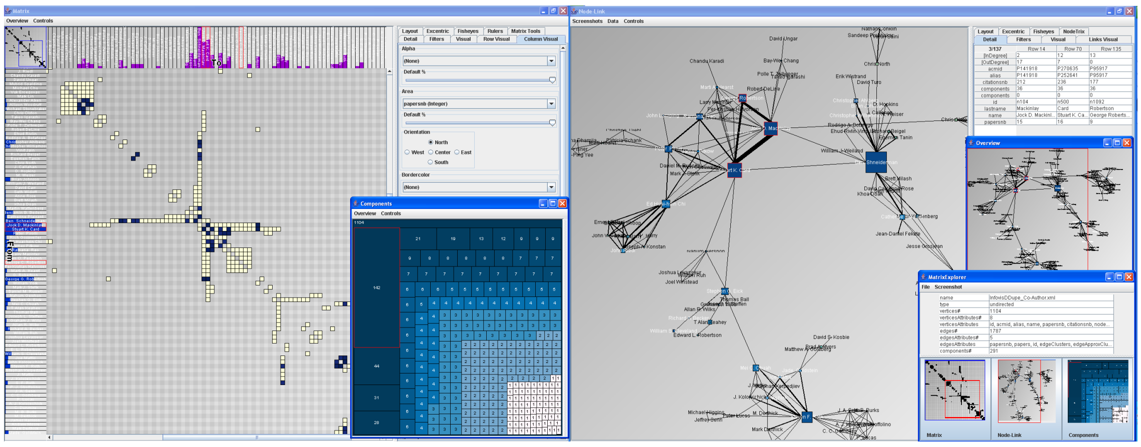


Figure 8: Interface de MatrixExplorer, combinant matrices (grande fenêtre de gauche) et nœud lien (grande fenêtre de droite). Une représentation Treemap (petite fenêtre de gauche) montre la macrostructure du réseau, i.e. ses composantes connexes. Des vues miniatures des représentations permettent la navigation (petites fenêtres de droite)

Ce simple exemple démontre que les matrices procurent une vue d'ensemble permettant d'initier l'exploration d'un jeu de données en mettant en évidence plusieurs éléments saillants. Le point important est d'apprendre à les décoder, car ces représentations sont beaucoup moins familières que les diagrammes nœud-lien. La Figure 6 donne quelques exemples de correspondance entre les motifs visuels courant dans les réseaux sociaux à la fois pour les représentations nœud-lien et matricielles.

Explorer interactivement

à l'issue d'entretiens avec des analystes et de séances d'observation, nous nous sommes rendu compte que le processus d'exploration visuel est constitué d'opérations successives pour configurer les représentations (ajuster la position des éléments graphiques et les attributs visuels), filtrer certains éléments, les grouper et éventuellement les agréger. Il est important de créer de vues multiples d'un jeu de données, l'analysant sous divers aspects et plusieurs niveaux de détail afin d'identifier sa structure. Par exemple, une représentation matricielle peut permettre d'identifier plusieurs communautés alors qu'un diagramme nœud-lien égocentré peut permettre l'analyse détaillée des connexions d'un acteur important. Dans MatrixExplorer, nous avons donc décidé de proposer plusieurs vues sur les données (Figure 8) et d'offrir un grand nombre d'outils pour manipuler directement [38] [39] les représentations.

Tout d'abord, nous avons synchronisé les deux visualisations (matricielle et nœud-lien) afin de combiner leurs avantages et de faciliter l'identification des motifs visuels. Il est ainsi possible d'utiliser les matrices pour certaines tâches et les diagrammes nœud-lien pour d'autres. Si un analyste souhaite avoir une vue d'ensemble sur un réseau et, par exemple, identifier ses principales communautés, il est préférable qu'il utilise une représentation matricielle. Par contre, lors d'une analyse plus détaillée, s'il est nécessaire d'identifier un plus court chemin entre deux acteurs du réseau, le diagramme nœud-lien constitue une meilleure alternative. De plus, synchroniser les deux représentations par la sélection permet de comparer la représentation d'un motif visuel dans l'une et l'autre représentation, facilitant ainsi leur compréhension (et l'apprentissage des matrices).

Pour manipuler interactivement les deux représentations nous avons mis en place un certain nombre de fonctionnalités :

1. Spécification interactive des attributs visuels comme la couleur des nœuds, des en-têtes des lignes et de colonnes dans les matrices, ou bien encore l'épaisseur des liens ou la couleur des cellules de la matrice.
2. Méthodes automatiques et interactives pour le placement des nœuds des diagrammes nœud-lien et le réordonnancement des lignes et colonnes des matrices. En particulier, nous avons développé plusieurs outils d'aide au réordonnancement en proposant d'appliquer des algorithmes automatiques sur des portions de matrices ou bien de bloquer l'ordre de plusieurs lignes et colonnes lorsque l'analyste souhaite les conserver.

[38] SHNEIDERMAN, B. (1983). Direct manipulation : A step beyond programming languages. In *IEEE Computer*, 16(8)

[39] AHLBERG, C., C. WILLIAMSON, ET B. SHNEIDERMAN (1992). Dynamic queries for information exploration : An implementation and evaluation. In *Proceedings of ACM SIGCHI'92*

3. Filtrage interactif de nœuds ou de connexions en fonction d'une sélection ou d'un attribut particulier présent dans les données (l'âge ou le sexe des personnes contenues dans le réseau social par exemple).
4. Clustering interactif permettant d'identifier les différents groupes ou communautés détectées pour les marquer par un attribut visuel tel que la couleur.
5. Finalement, plusieurs vues d'ensemble sont proposées : miniatures des visualisations dans lesquelles l'analyste peut se déplacer ou représentations sous forme de Treemap de la macro-structure du réseau (voir Figure 9).

En combinant les deux représentations et permettant de les manipuler interactivement, les utilisateurs de MatrixExplorer peuvent créer de multiples vues sur leurs données et les explorer à divers niveaux de détail.

Trouver un consensus

Comme chaque visualisation présente une perspective différente, nous avons tenté de favoriser l'établissement de consensus à l'aide d'interactions simples. La Figure 9 donne un exemple.

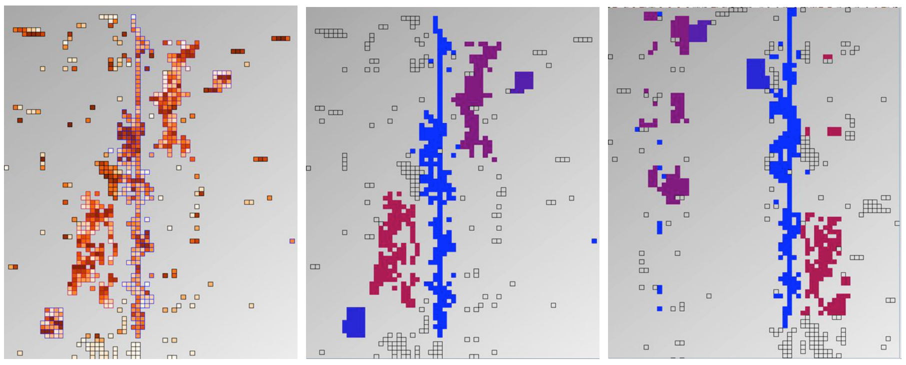


Figure 9: L'image du milieu montre l'identification de plusieurs clusters de la matrice initiale (à gauche). Après réordonnancement, nous constatons que l'un des clusters les plus foncés est conservé alors que l'autre se retrouve divisé en trois morceaux. Il y donc seulement consensus sur une partie des clusters identifiés (celui de couleur claire et l'un des foncé), il faut continuer à explorer pour en identifier éventuellement d'autres.

Pour faciliter cette recherche de consensus, nous avons également introduit la sélection approximative : possibilité de marquer le degré d'appartenance d'un élément à un cluster (représenté par une couleur plus claire par exemple) ; et le clustering multiple : possibilité de marquer l'appartenance d'un élément à plusieurs clusters.

Présenter l'information

Alors que la matrice permet principalement d'explorer les données et d'en manipuler une grande quantité, le diagramme nœud-lien est indispensable pour présenter des résultats et communiquer des découvertes. Ainsi, une fois le réseau filtré et éventuellement agrégé, MatrixExplorer permet d'extraire directement des images nœud-lien du résultat.

Representations hybrides

Combiner deux représentations présente beaucoup d'avantages, mais aussi plusieurs inconvénients. Tout d'abord, la taille de l'espace visuel : une utilisation confortable nécessite au moins deux écrans (un par représentation), un troisième offrant un confort supplémentaire en permettant de consulter les données détaillées éventuellement textuelles ainsi que plusieurs vues d'ensemble. De plus, nous avons observé que le passage d'une représentation à l'autre entraînait un coût cognitif non négligeable : un nœud unique se transforme en deux éléments dans la matrice (ligne et colonne) et un lien (visuellement représenté par un trait) devient une cellule (un rectangle) dans la matrice. Pour limiter ce coût cognitif, nous avons donc tenté d'améliorer les représentations matricielles pour limiter le recours au diagramme nœud-lien lors de l'exploration.

Augmenter les matrices

Comme l'ont montré les travaux sur la lisibilité de la matrice , la tâche de base pour laquelle cette représentation est moins performante que les diagrammes nœud-lien est le suivi de chemin. Pour remédier à ce problème, nous avons créé MatLink [40], une représentation matricielle augmentée (Figure 10).

Le principe de MatLink est de dessiner des liens sur les bords de la matrice afin de mieux représenter la connectivité et faciliter le suivi de chemin. Deux catégories de liens sont ajoutées sur la matrice : des liens statiques (en blanc sur l'image) et des liens interactifs (en plus foncé sur l'image) qui apparaissent uniquement lors d'un survol ou d'un clic de la souris sur les en-têtes des lignes et colonnes. Lorsque l'utilisateur sélectionne deux éléments, les liens apparaissent montrant un plus court chemin.

Nous avons réalisé une expérimentation contrôlée similaire à celle de Ghoniem et al. afin d'étudier si cette représentation améliorait significativement la représentation matricielle standard. Nous avons testé un lot de tâche de lisibilité de base ainsi que plusieurs tâches spécifiques à l'analyse de réseaux sociaux : trouver un point d'articulation, trouver une clique et identifier plusieurs communautés (groupe fortement connecté).

D'une manière générale, nous avons démontré que MatLink améliore significativement la représentation matricielle standard, en particulier pour la tâche de suivi de chemin, auparavant très laborieuse à réaliser. Pour ce cas précis, MatLink montre même des performances supérieures au diagramme nœud-lien pour les graphes denses (et ce, même augmenté de surlignage interactif) à cause du grand nombre de superpositions et de croisements. La seule tâche pour laquelle les diagrammes nœud-lien restent plus performants est l'identification de points d'articulation. Cette tâche requiert assez grande familiarité avec la représentation matricielle, ce qui n'était pas le cas de nos participants (qui, en moyenne, disposaient de 30 minutes d'entraînement avec les matrices).

[40]HENRY, N. ET J.-D. FEKETE (2007). Matlink : Enhanced matrix visualization for analyzing social networks. In *Proceedings of Interact'07*, LNCS 4663

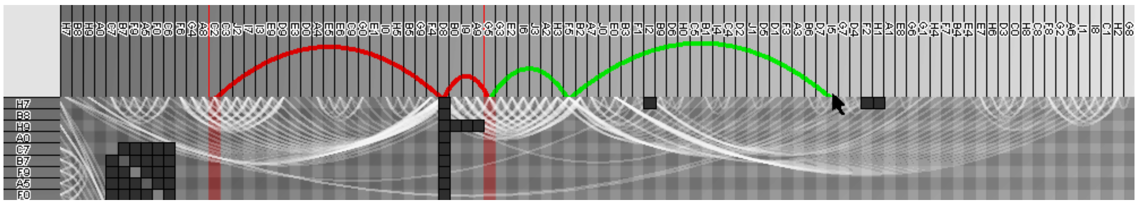


Figure 10: MatLink augmente les matrices en ajoutant des liens statiques (en blanc) et des liens interactifs (en plus foncés sur les en-têtes)

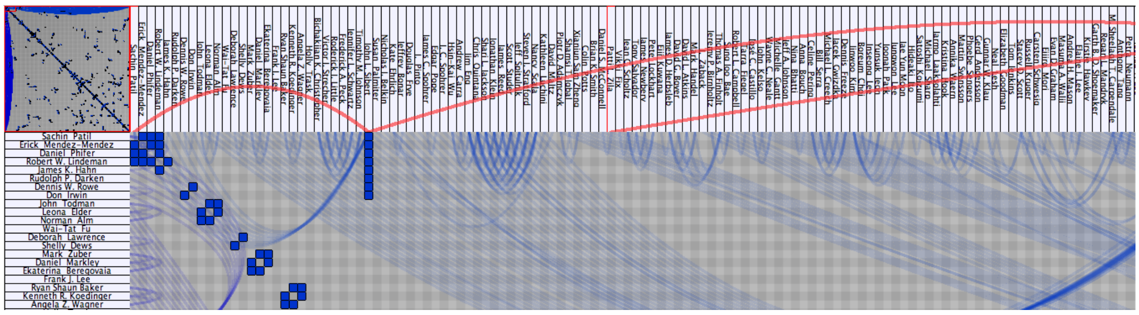


Figure 11: Les liens permettent à l'analyste d'être conscient que des éléments connectés aux nœuds qu'il étudie existent en dehors de la vue courante.

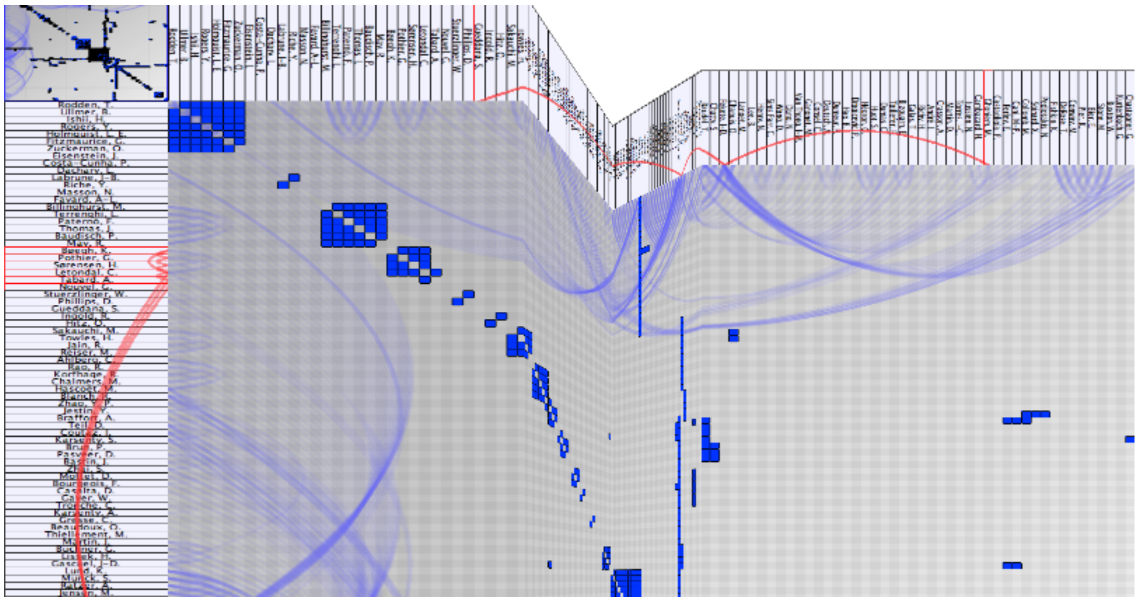


Figure 12: Mélange permet de plier l'espace pour permettre de voir deux régions éloignées de la matrice.

Naviguer dans de grandes matrices

Le second avantage de MatLink est l'amélioration que la technique offre pour la navigation dans de grandes matrices. Un des inconvénients des matrices est la grande taille requise pour les représenter. Ainsi, il peut (souvent) arriver qu'un acteur soit connecté à plusieurs autres acteurs placés en dehors de la vue. Il est nécessaire de naviguer sur toute la largeur (ou hauteur) de la matrice pour identifier tous les voisins d'un acteur particulier, ce qui est laborieux. Grâce aux liens affichés dans MatLink, et par un simple clic sur l'acteur étudié, l'analyste a un retour visuel immédiat sur tous les éléments connectés à cet acteur et possède une indication pour ceux placés hors de l'écran. La Figure 11 présente un exemple.

En outre, afin de faciliter l'exploration de ces grandes matrices (et le déplacement jusqu'aux éléments placés hors de la vue), nous avons mis au point une technique de navigation " pliant l'espace " : Mélange [41]. Cette technique permet de visualiser deux parties de la matrice dans une même vue, en offrant un aperçu de l'espace intermédiaire (Figure 12). Mélange offre même la possibilité de zoomer sur l'une des régions d'intérêt indépendamment de l'autre.

Fusionner Matrices et Nœud-lien

La structure des réseaux sociaux varie beaucoup : certains exhibent une structure d'arbre (ou presque) très peu dense tels les arbres généalogiques, d'autres ont des structures quasi complètes (réseau bipartite très dense) tels que les réseaux d'import/export entre pays du monde et enfin, certains ont une structure intermédiaire, localement dense mais globalement creuse, il s'agit des réseaux petit-monde [42] tels que les réseaux d'amitiés ou encore de communication.

Les représentations visuelles que nous avons présentées précédemment sont plus ou moins performantes selon le type de réseau à visualiser. Par exemple, les diagrammes nœud-lien ainsi que les représentations basées sur des arbres avec liens supplémentaires (Treemap+liens par exemple) sont particulièrement efficaces pour les réseaux peu denses présentant une structure d'arbres. Au contraire, si le réseau à représenter est plutôt dense ou bipartite et présentant une structure quasi complète, la représentation matricielle est alors tout indiquée. Par contre, le choix n'est pas si simple pour les réseaux petit-monde.

Le problème que posent les réseaux petit-monde réside dans la variation de la densité locale. Si l'on choisit un diagramme nœud-lien, alors les parties denses (communautés) ne sont pas lisibles. Inversement, si l'on choisit une matrice, alors la majorité de l'espace est vide et l'exploration requiert beaucoup de navigation. Une possibilité est alors d'utiliser MatrixExplorer, combinant les deux représentations, mais le coût cognitif est plus élevé.

Pour ne plus avoir à choisir, nous avons mis au point une seconde représentation hybride, fusionnant matrice et diagramme nœud-lien : NodeTrix [43].

[41]ELMQVIST, N., N. HENRY, Y. RICHE, ET J.-D. FEKETE (2008). Mélange : space-folding for multi-focus interaction. In *Proceedings of ACM SIGCHI'08*

[42]WATTS, D. ET S. STROGATZ (1998). Collective dynamics of Òsmall-worldÓ networks. In *Nature*, 393

[43]HENRY, N., J.-D. FEKETE, ET M. MCGUFFIN (2007). Nodetrix : Hybrid representation for analyzing social networks. In *IEEE TVCG (Infovis'07 proceedings)*, 13(6)

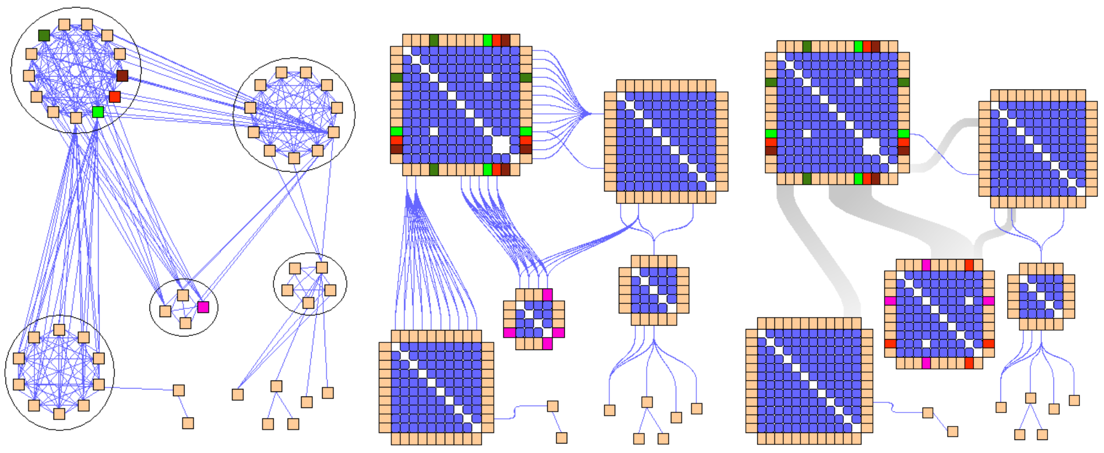


Figure 13: Représentation nœud-lien (à gauche) dans laquelle il est difficile d'identifier que les nœuds foncés ne sont pas connectés. La représentation NodeTrix (au milieu) visualise les sous-parties denses en matrices, il devient alors possible de voir les absences de connexions (matrice du haut gauche avec des cellules manquantes). Enfin, dupliquer des nœuds (à droite) permet d'améliorer la lisibilité en supprimant des croisements de liens inter-communautés.

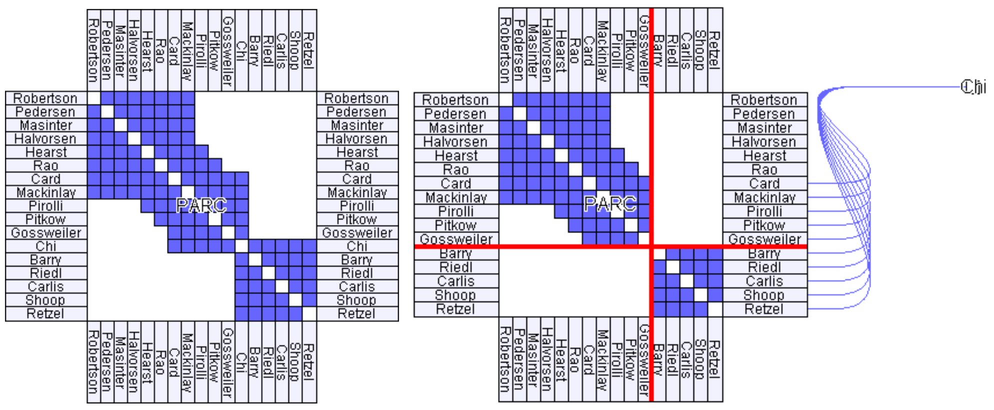


Figure 14: Interagir avec la représentation matricielle pour mieux comprendre la structure de la communauté. Ici, en ôtant une personne de la communauté, il est possible de voir qu'il s'agit d'un point d'articulation.

Le principe de NodeTrix est de représenter le réseau global comme un nœud-lien mais d'utiliser une représentation matricielle pour ses sous parties les plus denses. Deux exemples sont présentés Figure 13 et Figure 15.

Explorer

Afin de faciliter la création, l'exploration et l'édition des matrices représentant les parties denses du réseau, nous avons développé un ensemble d'outils interactifs basés sur la technique du glisser-déposer. Il suffit de sélectionner au lasso un ensemble de nœuds fortement connecté pour transformer cette partie du réseau en matrice. Des informations sur la connectivité intra-communauté apparaissent immédiatement comme, par exemple, lÔabsence de connexion entre deux membres de la communauté, auparavant difficile à percevoir à cause de la multitude de croisements de liens à cet endroit.

En outre, la représentation matricielle offre l'avantage de placer linéairement en ligne et colonnes les membres de la communauté ; il devient plus aisé d'identifier les membres connecté à des acteurs extérieurs. Pour permettre l'exploration de cette représentation, il est possible d'ajouter ou d'ôter interactivement un acteur à la communauté. L'interaction est simple : il suffit de cliquer sur l'élément (nœud ou bien en-tête d'une ligne ou colonne dans la matrice) et ensuite le glisser-déposer à l'intérieur ou à l'extérieur de la matrice (Figure 14). D'autres interactions permettent de fusionner deux matrices ou bien de les casser pour revenir à une représentation normale. Enfin, pour aider l'utilisateur à comprendre le changement de représentation, une courte animation permet de suivre la transformation (vidéo disponible sur `http://www.aviz.fr/~nhenry`).

Les matrices des parties denses peuvent être créées manuellement ou automatiquement en utilisant un algorithme de clustering. NodeTrix étant intégré à MatrixExplorer, il est également possible de glisser-déposer des parties d'une matrice dans une fenêtre NodeTrix. Cela permet de mieux comprendre la connectivité inter-communauté, ce qui est surtout important si les deux sous-matrices ne sont pas côte à côte dans la représentation matricielle standard.

Un inconvénient de cette représentation est la représentation concrète des communautés. Si un élément est membre de deux communautés, il faut faire un choix et le placer soit dans l'une des matrices, soit le laisser entre les deux (c'est un problème classique posé par l'agrégation hiérarchique de réseaux). Placer un acteur dans une des communautés peut biaiser l'interprétation de la représentation. Par contre, le laisser entre les deux communautés dégrade la lisibilité en introduisant beaucoup plus de liens et de croisements. Pour résoudre ce problème, nous avons proposé de dupliquer les acteurs pour les placer dans deux ou plusieurs communautés à la fois !

Notre étude utilisateur [44] a montré que l'impact potentiellement négatif des duplications sur l'interprétation du réseau peut être minimisé en représentant explicitement le lien de duplication. Après avoir exploré l'espace de conception possible, nous avons décidé de représenter ce lien par une bande (un lien plus épais) de couleur claire permettant de minimiser les interférences avec la représentation des connexions “ réelles ” du réseau. Un exemple est présenté dans la Figure 13

[44]HENRY, N., A. BEZERIANOS, ET J.-D. FEKETE (2008). Improving the readability of clustered social networks by node duplication. In *IEEE TVCG (Infovis'08 proceedings)*, 14

(droite). Notre évaluation portait sur six tâches de lisibilité du réseau (dont certaines spécifiques à l'analyse de réseaux sociaux) pour deux réseaux de densités différentes. Les résultats ont démontré que les duplications permettaient de significativement améliorer la lisibilité de représentation et d'offrir une vue non biaisée de chaque communauté.

Présenter l'information

Le grand avantage de NodeTrix est sa versatilité : il est possible de l'utiliser pour explorer un réseau et découvrir sa structure mais aussi pour en faire une représentation très compacte et l'utiliser pour présenter des résultats. Un exemple est présenté Figure 15. Cette souplesse d'utilisation est liée à l'intégration des représentations matricielles et à leur capacité à produire à la fois une vue d'ensemble et une vue de détail. En mode présentation des résultats, il est alors possible de réduire au maximum la taille des communautés représentée par des matrices, faisant disparaître les étiquettes individuelles de chacun des membres (mais en conservant une étiquette pour la communauté par exemple) mais permettant de conserver un aperçu de la connectivité à l'intérieur de la communauté.

Conclusion

Les représentations visuelles sont importantes à la fois pour l'analyse des réseaux sociaux et la communication des résultats d'analyse sur ces réseaux. Actuellement, la très grande majorité des représentations de réseaux sont des diagrammes nœud-lien. Si ces représentations sont familières et paraissent très intuitives, elles souffrent d'importants problèmes de lisibilité lorsque le réseau comporte un grand nombre de nœuds ou de liens. Dans cette préface et la suite de la thése, nous avons présenté plusieurs représentations alternatives permettant de résoudre ces problèmes de lisibilités.

En particulier, nous nous sommes concentrés sur les représentations basées sur la matrice d'adjacence de graphe. Cette représentation présente plusieurs avantages par rapport aux nœud-lien, le plus important étant de supprimer toute superposition de nœuds et croisements de liens ce qui est en particulier utile pour les réseaux de forte densité. Ces représentations ont aussi plusieurs inconvénients comparées aux diagrammes nœud-lien : il est notamment très laborieux d'utiliser les matrices pour identifier des chemins entre plusieurs nœuds.

Combinant les avantages des deux représentations, nous avons présenté MatrixExplorer, un système interactif permettant d'explorer les réseaux sociaux en manipulant matrices et diagrammes nœud-lien pour avoir de multiples perspectives sur les données. Nous avons également présenté deux représentations hybrides, fusionnant les deux représentations : MatLink, permettant d'améliorer significativement les tâches de suivi de chemin dans les matrices; et NodeTrix, particulièrement efficace pour visualiser les réseaux petit-monde.

Ces représentations offrent de nouvelles perspectives sur la représentation visuelle des réseaux sociaux, permettant à la fois de les explorer et constituant également une solution pour communiquer de l'information et présenter des résultats d'analyse. Ils constituent des solutions à l'analyse visuelle des nouveaux réseaux sociaux en ligne.

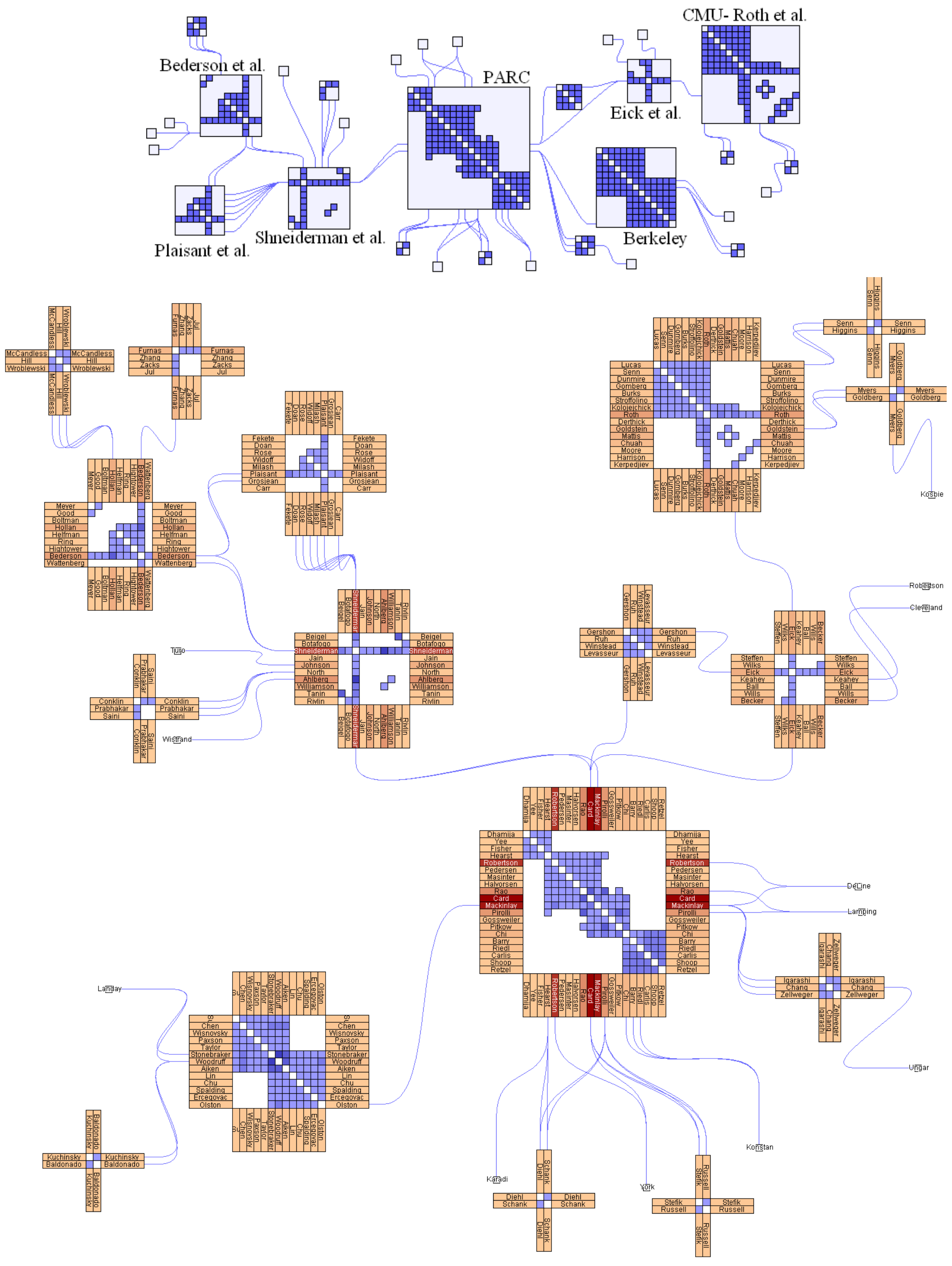


Figure 15: Réseau de co-publication d'infovis contenant plus d'une centaine d'acteurs. Représentation NodeTrix compacte pour la communication (en haut) et détaillée pour l'exploration (en bas).

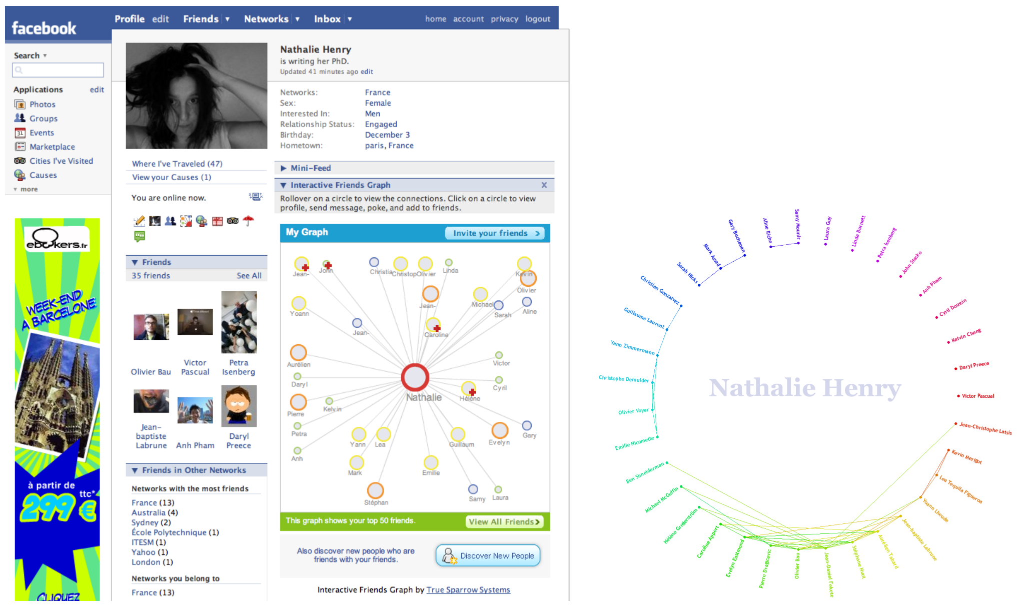
A picture is worth a thousand words...
if it is the right *picture*

Table of Contents

List of Figures

Introduction

CHAPTER 1



Nathalie Henry
is writing her PhD.
Updated 41 minutes ago edit

Networks:	France
Sex:	Female
Interested In:	Men
Relationship Status:	Engaged
Birthday:	December 3
Hometown:	paris, France

Rollover on a circle to view the connections. Click on a circle to view profile, send message, poke, and add to friends.

Networks with the most friends
France (13)
Australia (4)
Sydney (2)
École Polytechnique (1)
ITESM (1)
Yahoo (1)
London (1)

Networks you belong to
France (13)

Also discover new people who are friends with your friends.

Interactive Friends Graph by True Sparrow Systems

Figure 1.1: Create and Visualize your social network on FaceBook.com

1.1 Motivations

In the last decade, the popularity of social networking applications has dramatically increased. Any collection of persons or organizations connected by relations is a social network. For example, a genealogical tree is a social network composed of persons linked by their family ties. Today's web surfers are often part of many online social networks: they communicate in groups or forums on topics of interests, exchange emails with their friends and colleagues, express their ideas on public blogs, share videos on YouTube, exchange and comment photos on Flickr, participate to the edition of the online encyclopedia Wikipedia or contribute to daily news by collaborating to Wikinews or Agoravox.

Recent online networking systems with a racing popularity such as Friendster, LinkedIn or FaceBook are even exclusively dedicated to manage and extend one's own social network. Registered users voluntarily enter their contacts (family, friends or colleagues) and the nature or their relationships. Contacts not already registered on the website are personally invited to join the community. Therefore, these online communities grow rapidly and without effort thanks to a snowball effect. Large social networks existed before such as telephone networks listings, postal communication or bank transactions. However, these recent online systems tremendously simplify the collection of data. Compared to data collected through polls and interviews, collected networks are far larger and contain much richer information. This avalanche of vast new datasets raises new challenges: tools need to support the analysis of a very large amount of data often evolving through time.

Analyzing how people communicate, collaborate, what information they exchange, what role they play in the social group is becoming a point of interest of a large variety of organizations, out passing the personal use. The stakes of social networks analysis are becoming very high: since September 11, there is an added demand from intelligence agencies to monitor terrorists networks, attempting to discover when they will act; epidemiologists require effective analysis tools to study transmission networks seeking to detect and contain outbreak of diseases such as SARS or the bird flu; company managers and research institutes aim at studying the flow of communication between their employees or the strength of the collaboration between researchers to evaluate them and improve their productivity. We have no reliable models of such networks; therefore there is a need for tools supporting the exploratory analysis of the structure of real social networks.

In this thesis, we attempt to address the following research question:

➤How can we help social scientists visually explore large social networks?

1.2 Approach

Several approaches are possible to tackle the problem of analyzing a large amount of data. The fields of data mining and knowledge discovery in databases provide automatic or assisted technologies to "extract useful information from large data sets or databases" [HMS01]. In this perspective, closely related to artificial intelligence, the computer is performing partially or completely the data analysis for the user.

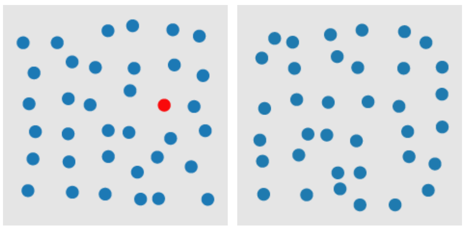
Our approach is different, we aim at helping the user *perceive* the information, *making sense* of the data and discover *insights*. This thesis draws ideas from the field of information visualization and visual analytics: the computer provides one or several visual representations of the data; the analyst performs the data analysis by interacting with the representations.

1.2.1 Information visualization

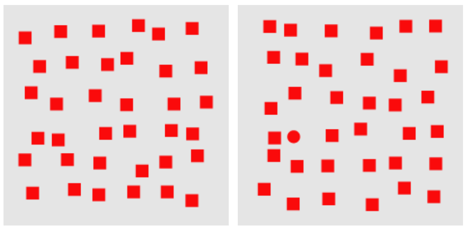
The research field of computer-based visualization relies on the fact that the human brain is particularly effective at processing visual information. For example, providing a map to teach the geography of the main cities of France is much more effective than giving a list of longitudes and latitudes.

Card et al. [CMS99] define information visualization as "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition". A good visual representation can amplify the cognition by providing *more* information, *faster*, with *less cognitive effort*. Communicating visual information can be more effective than textual information because several visual features called preattentive [Tre85] are easier to process by the human brain than symbolic information. The perceptual system can recognize preattentive features such as curvature and color instantly (Figure 1.2). Carefully designed, a visual representation can convey a large amount of data very rapidly.

Another valuable advantage of exploring data using information visualization is unexpected discoveries. Unlike classical or Bayesian statistical analysis, which evaluate *a priori* questions according to a model, visualization offers the potential to start the analysis without assumptions



(a) Color

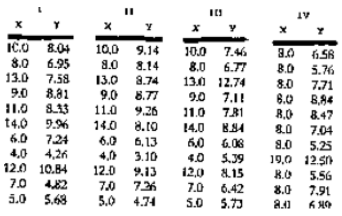


(b) Curvature

Figure 1.2: Did you notice a round red target? In both cases, it is very fast as curvature and color are preattentive features.

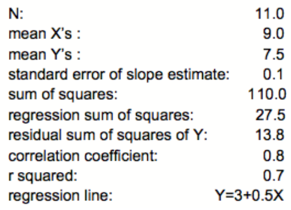
and can open new perspectives on a previously analyzed dataset. This approach called Exploratory Data Analysis (EDA [Tuk77]) has been introduced by the statistician John Tukey in 1977. Twenty years later, Ben Shneiderman presented the visual information seeking mantra [Shn96] to design information visualization systems: “overview first, zoom and filter, then details-on-demand”. This mantra suggests initiating the analyzing process by providing general information on the dataset: showing the whole datasets, its main characteristics and/or representative information. The goal is to first have a perspective of the entire data to analyze, eventually understand its macro-structure and finally identify interesting areas to explore. Understanding the macro-structure is a combination of zooming and filtering interaction to isolate the selected areas and analyze them more closely. Finally, the details on the data should only be shown on demand to avoid submerging the user into a large quantity of information, potentially degrading the representations readability. Our research work attempts to follow these lines, offering social scientists interactive visualizations to explore their data.

A popular example of the potential of information visualization to perform EDA is Anscombe’s number (Figure 1.3). This example demonstrates that computing a set of general statistics (*a priori* describing the data) on four different tables of numbers might not show any difference whereas creating simple graphics (such as scatterplots) to directly visualize the raw data can reveal in a glimpse that they are very different in distribution.



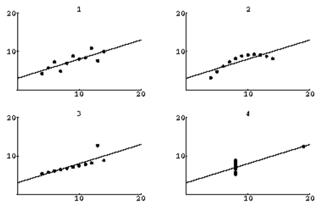
I		II		III		IV	
X	Y	X	Y	X	Y	X	Y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

(a) Four different datasets



N:	11.0
mean X's :	9.0
mean Y's :	7.5
standard error of slope estimate:	0.1
sum of squares:	110.0
regression sum of squares:	27.5
residual sum of squares of Y:	13.8
correlation coefficient:	0.8
r squared:	0.7
regression line:	Y=3+0.5X

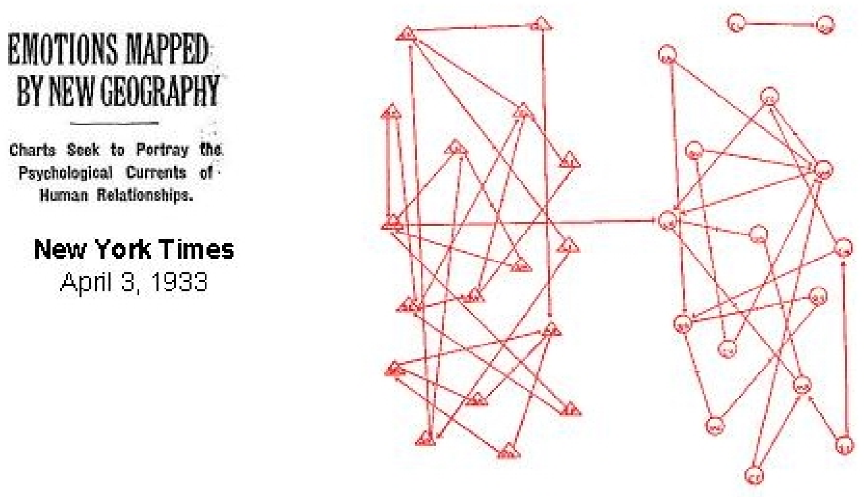
(b) Same statistics



(c) Different visualizations

Figure 1.3: These four datasets have been generated by Anscombe in the early 1970’s.

Visualization has been applied to social network analysis since the 30's with the work of Jacob Moreno [Mor34]. He drew social networks as node-link diagrams, the traditional representation of graphs. An example of this representation is presented in Figure 1.4. Nodes represent fourth grade students (triangles represent girls, circles represent boys) and the links between them represent friendship. Representing the network visually shows at a glance the general trend as well as unexpected discoveries. The general trend is that friendship occurs between children of the same gender. The unexpected discoveries are the two boys apart from the rest of the boys as well as the unique friendship relation between a girl and a boy.


EMOTIONS MAPPED BY NEW GEOGRAPHY

Charts Seek to Portray the Psychological Currents of Human Relationships.

New York Times
April 3, 1933

Figure 1.4: Moreno's social network. With this node-link diagram published in the New York Times in 1933, Jacob Moreno shows relationships between fourth graders.

1.2.2 Matrix-based representations

With the increasing popularity of social networking and the progress of internet technologies, many systems emerged to visualize and analyze social networks. Last years conferences in information visualization saw around 10 new systems, the INSNA [1] repository lists more than 60. The very large majority of these systems still use node-link diagrams, as did Moreno in the 30's. This representation is extremely popular and has its own dedicated research field called graph drawing. It mainly deals with the layout of node-link diagrams: how to place the nodes in space to produce a drawing with special properties such as having the minimum number of links crossing each other. A very large number of algorithms exist to draw graphs [DETT98]. However, the key issue with these representations is *scalability*. When the network grows larger than several hundred nodes or denser with many links between them, node-link diagrams become hardly readable as nodes overlap and links cross each other.

[1] `http://www.insna.org`

Usual strategies to overcome this readability problem are *filtering* or *aggregation*. Filtering is the principle of removing nodes or links from the network. It reduces its size and density, potentially improving the node-link diagram readability. Filtering can be done randomly (sampling), by discarding some data values or based on a computed criterion such as node or link importance. Aggregation is the principle of grouping a set of nodes into a super-node. The group of nodes and their connection become an abstraction and, if links are also aggregated, the network size and density also decreases. In both cases, information is removed from the representation, making it impossible to get an overview of the whole data. This breaks the "overview first, zoom and filter, then details-on-demand" mantra, in which it is crucial to visualize the whole dataset in order to initiate a detailed analysis. Therefore, we have sought alternatives to node-link representations, able to handle larger datasets.

From a study realized by Ghoniem et al. in 2004 [GFC05], comparing the readability of node-link diagrams and adjacency matrix representations for several low-level tasks of exploration (such as finding if two nodes are connected), we discovered that matrix-based representations had a vast potential to represent social networks. Authors showed that matrices outperformed node-link diagrams for large and dense graphs, but that node-link diagrams were still more effective for sparse or small graphs for most of the low-level tasks. Social networks vary from very sparse (genealogy trees) to locally dense (small world networks) to very dense (tables of goods exchange). Our approach is to take advantage of both matrix and node-link representations, improving them, combining and merging them to handle many different cases.

1.2.3 Research approach and itinerary

We used user-centered and participatory design methods to gather requirements for a visual exploratory system. The philosophy of these methods is to imply users of a future system into its design as early as possible. We initiate this research work by meeting a group of social scientists to better understand their needs and wishes for a visual analysis system. Chapter 3 describes in details our approach as well as the initial results of a first participatory design session. Our research itinerary is composed of 6 phases, each of them informed by sessions with users and informal feedback.

❶ **Make matrices usable.** Contrary to node-link diagrams, a matrix always allows performing basic tasks such as finding if two persons are connected, even for very large or very dense graphs. However, Bertin showed that reordering rows and columns of a matrix can not only ease these tasks but also make insights or trends in the data emerge. During this first phase, we worked on the reordering problem, presenting a review of existing methods and attempting to evaluate their quality.

❷ **Combine node-link and matrices.** In this second phase, we provided our users with a system combining both node-link diagrams and matrices called *MatrixExplorer*. As social networks vary from very sparse to very dense, we proposed to combine node-link, effective for sparse graphs, and matrices, effective for dense graphs. Our goal was to support the exploration process from the initial step of loading data to the final step of creating pictures to illustrate the analyst's findings. We provided a set of interactive tools to reorder, layout, filter, and cluster both representations. To validate this system and get feedback on its use, we performed a case study on a dataset of 20 years of publication data in Human Computer Interface.

❸ **Augment matrices.** In this third phase, informal user feedback reported that combining both representations required two screens and that switching back and forth increased their cognitive load. We decided to augment matrices to overcome their main weakness: the task of following a path between two nodes in the network. We designed an interactive system called MatLink, overlaying static and interactive links on the matrix borders, and ran a controlled experiment proving that it outperformed standard matrices.

❹ **Support the navigation in large matrices.** Inspired from this previous design and by the need for navigation in large matrices that we discovered during the case study, we created a navigation technique (Mélange) to navigate in a large space while keeping two points of focus.

❺ **Merge node-link and matrices.** In this fifth phase, we tackled the problem of representing small-world networks. These networks are composed of dense communities connected by few links. We noticed that node-link diagrams could reveal how communities are connected but not their intra-connectivity. On the contrary, the intra-connectivity of the communities is readable with matrices but the inter-connectivity is hard to perceive, as matrices are large and almost empty. To solve this problem, we created a hybrid representation called *NodeTrix*.

❻ **Help identifying communities.** In this last phase, we studied with social scientists the problem of overlapping communities; we proposed as a solution the use of duplicated nodes and ran a study to understand their effects on important analysis tasks.

$$\begin{tikzcd}
 & & ❸ \arrow[r] & ❹ \\
❶ \arrow[r] & ❷ \arrow[ur] \arrow[dr] & & \\
 & & ❺ \arrow[r] & ❻
\end{tikzcd}$$

The result of these 6 iterative phases is a visual and interactive system prototype including all features for exploring social networks. We attempted to evaluate our work at all stages, based not only on informal user feedback, but also controlled experiments and a case study.

1.3 Contributions

The contributions of this thesis are:

- a study with social scientists to collect requirements for visual social networks analysis systems;
- an annotated bibliography of reordering methods for tables and matrices;
- a controlled experiment attempting to assess how reordering affects the understanding of visual tables;

- the design and implementation of MatrixExplorer: an interactive system combining and linking the node-link diagram and matrix representations;
- the design and implementation of MatLink, a hybrid matrix representation overlaying links on the matrix borders, and its controlled experiment;
- a novel interaction technique called Mélange, supporting the navigation in large matrices while keeping two focus points, and its controlled experiment;
- the design and implementation of NodeTrix, a representation merging node-link diagrams with matrices to represent the dense parts of a network;
- the design and implementation of node duplications within NodeTrix and its controlled experiment;
- a case study on publication data to validate our prototypes.

1.4 Thesis organization

Chapter 2 discusses the related work. We present the state-of-the-art systems in the field of social network analysis and describe the existing techniques to visualize graphs in information visualization. As interacting with a representation is crucial when the dataset is large, we dedicated a section for the work relative to navigation and interaction techniques. Finally, we present an overview of evaluation methods to validate information visualization systems.

In Chapter 3, we describe our design process and present the outcomes of our interaction with social scientists. We explain our methodology and formalize a list of requirements for social network analysis. This chapter introduces the central concepts of our design: exploration using multiple interactive visualizations to gather insights and find consensus.

The five main contributions of this PhD are described from Chapter 4 to Chapter 8.

Chapter 4 tackles the problem of matrix reordering. We present an annotated bibliography of reordering techniques and discuss how to assess the quality of an ordering. We then report our attempt at collecting empirical evidence on how ordering can affect human understanding of matrix representations.

Chapter 5 describes MatrixExplorer, a visual interactive system combining node-link diagrams and matrices. We explain how this system is designed for exploration and present its innovative features for interactive layout, ordering, filtering and clustering. The design process and initial study from Chapter 3, as well as several ordering algorithms from Chapter 4, are included in MatrixExplorer.

Chapter 6 exposes matrices weaknesses: the difficulty to perform path-related tasks such as finding how many hops are needed to connect one person to another. We present how we designed and implemented MatLink, a hybrid matrix representation overlaying links on the matrix borders and providing interactive feedback when performing path-related tasks. We conclude the first part of this Chapter with a controlled experiment comparing MatLink to standard node-link and matrix representations. The second part of this chapter present Mélange, a navigation technique inspired by MatLink. Mélange supports the navigation of large matrices while keeping two points of focus, which is often important when performing path-related tasks or comparison tasks. We present the design of the technique and the controlled experiment we performed to compare it to standard pan and zoom and split-screen navigation techniques.

Chapter 7 deals with the problem of representing small-world networks. These networks are globally sparse but locally dense. Since node-link diagrams are more effective at representing sparse graphs, and matrices at dense ones, we decided to merge both representations. We present NodeTrix, a hybrid representation that visualizes the network as a node-link diagram with dense part as matrices. We describe a set of interactive techniques to manipulate it. In the second part of this chapter, we introduce the concept of duplicated elements. We present our design and user study within NodeTrix.

In Chapter 8, we discuss how we validated our work. We first discuss the informal feedback of a second participatory design session. Then, we explain our choices in term of selecting tasks and datasets for controlled experiments aiming at evaluating representations readability. The rest of this chapter presents a case study of 20 years of publication data in the field of Human Computer Interface. We describe our process and results using the prototypes we designed (MatrixExplorer, MatLink and NodeTrix).

We conclude this thesis with Chapter 9 presenting a summary of our contributions and drawing perspectives on future work.

1.5 Publications

The majority of the contributions presented in this dissertation have been previously published in international conferences and journals.

Refereed journal articles

. *Improving the Readability of Clustered Social Networks by Node Duplication* . (Chapter 7)
N. Henry, A. Bezerianos and J-D. Fekete.
In IEEE Transactions on Visualization and Computer Graphics (Proceeding of the IEEE Visualization / Information Visualization Conference 2008.), to appear (8 pages).

. *20 Years of 4 HCI Conferences: a Visual Exploration.* (Chapter 8)
N. Henry, H. Goodell, N. Elmqvist and J-D. Fekete.
In International Journal of Human Computer Interaction, Reflections on Human-Computer Interaction, special issue in honor of Ben Shneiderman's 60th birthday, December 2007. Vol. 23, No 3, pp. 239-285 (46 pages).

. *NodeTrix: A Hybrid Visualization of Social Networks.* (Chapter 7)
N. Henry, J-D. Fekete and M. J. McGuffin.
In IEEE Transactions on Visualization and Computer Graphics (Proceeding of the IEEE Visualization / Information Visualization Conference 2007.) Vol. 13, No. 6, November/December 2007, pp. 1302-1309 (8 pages).

. *MatrixExplorer: a Dual-Representation System to Explore Social Networks.* (Chapter 5)
N. Henry and J-D. Fekete.

In IEEE Transactions on Visualization and Computer Graphics (Proceeding of the IEEE Visualization / Information Visualization Conference 2006.) Vol. 12, No. 5, September/October 2006, pp. 677-684 (8 pages).

Refereed full-length international conference articles

. *Mélange: Space Folding for Multi-Focus Interaction.* (Chapter 6)
N. Elmqvist, N. Henry, Y. Riche and J-D. Fekete.
In Proceedings of the ACM SIG CHI conference 2008 (10 pages).

. *ZAME: Interactive Large-Scale Graph Visualization.* (Mentioned in Chapter 5)
N. Elmqvist, T-N. Do, H. Goodell, N. Henry and J-D. Fekete.
In Proceedings of the IEEE Pacific Visualization Symposium 2008 (8 pages).

. *MatLink: Enhanced Matrix Visualization for Analyzing Social Networks.* (Chapter 6)
N. Henry and J-D. Fekete.
In Proceeding of the eleventh IFIP TC13 International Conference on Human-Computer Interaction (Interact 2007), September 2007. Springer Verlag. pp. 288-302 (14 pages). Received the **Brian Shackel award** (best paper award).

. *MatrixExplorer: Un Système pour l'Analyse Exploratoire de Reseaux Sociaux.* (Chapter 5)
N. Henry and J-D. Fekete.
In Proceedings of IHM2006, International Conference Proceedings Series, September 2006, Montréal, Canada. pp. 67-74 (8 pages). ACM Press.

Refereed workshops and doctoral consortium

. *A Methodological Note on Setting up Logging and Replay Mechanisms in InfoVis Systems.*
N. Henry, N. Elmqvist and J-D. Fekete.
Position paper for the 2008 SIG CHI workshop on BEyond time and error: novel evaLuation methods for Information Visualization (BELIV'08).

. *Visually Exploring Social Networks.*
N. Henry.
Doctoral Consortium. In Extended abstract of the international conference Interact 2007.

. *Evaluating Visual Table Data Understanding.* (Chapter 8)
N. Henry and J-D. Fekete.
In Proceedings of the 2006 AVI workshop on BEyond time and error: novel evaLuation methods for Information Visualization (BELIV'06). ACM Press.

. *Task Taxonomy for Graph Visualization.* (Mentioned in Chapter 8)
C. Plaisant, B. Lee, C. Sims Parr, J-D. Fekete and N. Henry.
In Proceedings of the 2006 AVI workshop on BEyond time and error: novel evaLuation methods for Information Visualization (BELIV'06). ACM Press.

Related work

CHAPTER 2

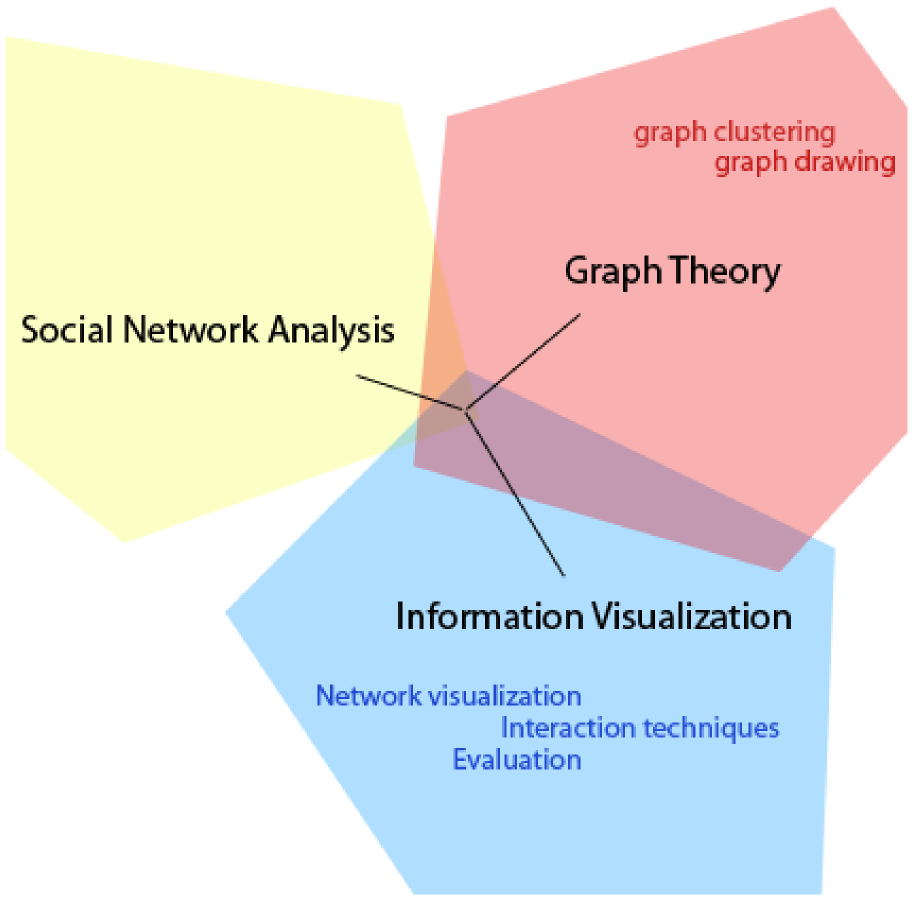


Figure 2.1: Three main threads of related work.

In this chapter, we discuss three main threads of related work: social networks analysis, graph theory and information visualization. Before starting the literature review, we introduce the terminology of each field in section 2.1. This section can be dismissed for readers familiar with these three research fields.

We initiate the literature review with section 2.2, presenting examples of social networks as well as introducing what social network analysis is. This section presents an overview of the techniques used to analyze social networks and, therefore, introduces our application domain.

Section 2.3 introduces the field of graph drawing. This section is dedicated to node-link diagrams, the most popular representation of graphs. We present an overview of the large range of techniques to layout node-link diagrams and explain the problem of scalability of these representations. To answer this scalability problem, we present techniques of data reduction in section 2.4. We describe two approaches: graph filtering (as well as sampling) and graph clustering. We explain the strengths and weaknesses of each approach before introducing solutions from the field of information visualization.

In section 2.5, we describe alternative graph representations as well as interaction techniques used to explore large visual representations or dynamically reduce visual complexity while maintaining some context. We briefly present the challenge of evaluating information visualization systems. Our last section deals with a sub-area of information visualization: network visualization (section 2.6). We present information visualization systems to explore networks in general and social networks in particular. We show the gap between network visualization and social network analysis and situate the context of this dissertation. Finally, we conclude on novel use of information visualization systems by non-expert users and show that social networks benefit of an increasing interest.

2.1 Terminology

This dissertation is cross-disciplinary and deals with social networks analysis, graph theory and information visualization. All three research fields use different terms for similar concepts. It is important to clarify the definitions and formalisms used in each of these disciplines. We attempt to introduce the basic concepts of all three fields as well as the formalism we use throughout this dissertation. Note that in each section of the related work, we will use the terms of the corresponding field.

2.1.1 Graph theory

A *graph* $G < V, E >$ consists of a finite set of vertices V and a finite set of edges E. If a graph contains two different types of vertices, it is called a *bipartite* graph. Each edge $e \in E$ connects a pair of vertices $(u, v) \in V$ and is noted e_{uv}. An *undirected* graph is composed of undirected edges, *i.e.,* the pair of connected vertices (u, v) is not ordered. In this case, e_{uv} is equivalent to e_{vu}, u and v are adjacent, we generally say that u and v are neighbours. A *directed* graph or digraph is composed of directed edges or arcs, *i.e.*, the pair of connected vertices (u, v) is ordered. In this case, e_{uv} is not equivalent to e_{vu}, the first vertex is called source and the second is called target. Considering e_{uv}, v is accessible from u but u is not accessible from v. Edges can be assigned weights, a value representing the strength of the connection, the graph is then called a *weighted* graph.

The *adjacency matrix* of a graph is a $V \times V$ matrix depicting how all the vertices of a graph are connected to each other. The adjacency matrix A of a graph is defined as:

$$A_{i,j}; i, j \in V = \begin{cases} 1 & \text{if } e_{ij} \in E \text{ ;} \\ 0 & \text{otherwise.} \end{cases}$$

Adjacency matrices of undirected graphs are symmetric as e_{ij} is equivalent e_{ji}. For weighted graphs, the weight of the edge e_{ij} is placed in the matrix instead of 1. By convention, we read adjacency matrices of directed graphs from row to column: rows are sources and columns targets of the arcs.

The *degree* of a vertex v, noted $deg(v)$, is the total number of incident edges to this single vertex. When dealing with directed graphs, the degree of a vertex v is the sum of the *in-degree* (total number of arcs having v as a target) and the *out-degree* (total number of arcs having v as a source).

The *Laplacian matrix* of a graph is a $V \times V$ matrix used in spectral graph theory [Chu97]. As we explain later (cf section 2.3), computing the smallest non-null eigenvectors of this matrix can be used to layout a node-link diagram [KCH03].

$$L_{i,j}; i,j \in V = \begin{cases} \text{deg}(v_i) & \text{if } i = j; \\ -1 & \text{if } e_{ij} \text{ exists,} i \neq j; \\ 0 & \text{otherwise.} \end{cases}$$

A *walk* is a sequence of vertices and edges. If all vertices are distincts, this sequence is called a *path*. Note that sometimes a path is defined as a collection of vertices only or edges only. The length of a walk or path is the number of edges of the sequence. The *shortest path* between two vertices u and v is a path with the minimum length possible. A *closed walk* is a walk beginning and ending at the same vertex. If a closed walk contains at least three vertices with all edges and vertices distinct (except the start and end), it is called a *cycle*. Two vertices are said to be *reachable* when there is at least one path connecting them. The *topological* or *geodesic distance* between u and v, noted $dist(u, v)$ is the number of edges of the shortest path between u and v. When the graph is weighted, the distance is computed by addition of the weights of the edge sequence (in this case, the minimum number of edges might not be the shortest path). If no path connects u to v, the distance is ∞. The *eccentricity* of a vertex v, noted $\epsilon(v)$, is the maximum distance between v and any other vertex.

A *sub-graph* is composed of vertices and edges that are a subset of a graph G. A *connected component* is a maximal sub-graph of G in which all vertices are reachable from one another. A *cut point* is a vertex that disconnects the graph if removed. A *minimum spanning tree* of a graph is a subgraph without cycles (a tree) which connects all vertices of the graph with a minimum number of edges. A *group* or *cluster* is a subset of the vertices of G. Members of a group are close together according to a given *metric*. A common metric is the length of the shortest path between two vertices. A *partition* is a set of clusters of G with no overlapping vertices. Dividing all the vertices of G in two partitions is called a *cut*. The *size of the cut* is the number of edges connecting any vertex of one partition to the second one. Finding the minimal cut (with the minimum size) is a common problem called *MinCut*.

The *diameter* of a graph is the maximum eccentricity of any vertex of the graph, *i.e.*, the maximum distance between any two vertices of the graph. A vertex that achieves the maximum distance to any other vertex is called a *peripheral* vertex. The *density* of a graph is the ratio of the number of edges noted m compared to the number of vertices noted n. Two definitions exist, a discussion of their use can be found in [Mel06]:

· Ratio depending on n:

$$d_G1 = m/n, d_G\epsilon[0, \frac{n-1}{2}] \quad \text{[MH00]}$$

· Ratio varying in a fixed interval:

$$d_G 2 = \sqrt{m/n^2}, d_G \epsilon [0, \frac{1}{\sqrt{2}}] \quad \text{[GFC05]}$$

According to [WS98], a graph is called *small-world* if it has:

1. a "small" mean shortest path;
2. a "high" mean clustering coefficient;
3. a power-law degree distribution.

The *clustering coefficient* of a vertex is the proportion of existing edges connecting its neighbours divided by the number of possible edges connecting them. Formally, if we consider an undirected graph and define the neighbours of v_i as $N_i = v_j$ where $e_{ij} \in E$, and if we consider v_i having k neighbours and therefore $k(k-1)/2$ possible edges between them. The clustering coefficient of v_i is expressed as:

$$CC(v_i) = \frac{2|e_{jk}|}{k(k-1)}, v_j, v_k \epsilon N_i \quad \text{[WS98]}$$

The clustering coefficient for the whole graph is defined as:

$$CC_G = \frac{1}{n} \sum_{i=1}^{n} CC(v_i) \quad \text{[WS98]}$$

The *degree distribution* of a graph is a function noted $P(k)$ describing the number of vertices per degree. This distribution is called power-law if $P(k) \sim k^{-\lambda}$. Figure 2.2 illustrates the difference between a normal degree distribution and a power-law degree distribution.

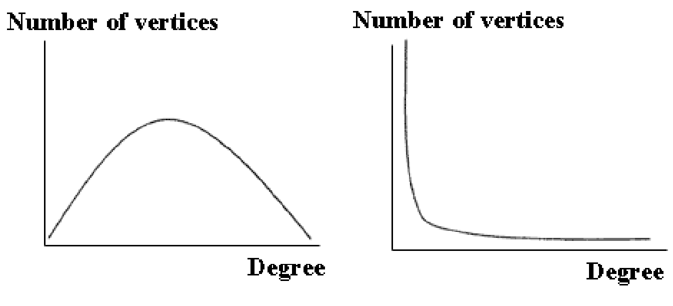


Figure 2.2: The difference between normal (a) and power-law (b) distribution. While in a normal distributions, the mean degree is representative, it is not the case in power-law distribution in which very few vertices have a very high degree and many vertices have a rather low degree.

2.1.2 Social network analysis (SNA)

A *social network* consists of a set of *actors* connected by *relationships* or *relation ties*. The term network is often used for graph. However, a social network links both structure (graph) and data (attributes). Actors correspond to vertices associated to a set of properties or attributes such as the name and age of a person; relationships correspond to edges connecting actors, also associated with a set of attributes such as the label of the tie and its strength for example. The SNA counterpart of a bipartite graph is a 2-mode network.

All measures defined in graph theory are used in SNA: degree, distance, eccentricity, clustering coefficient. Some measures, however, are specific to SNA. For example, the *centrality* of an actor is a concept attempting to reflect the structural location of an actor in the network. Several formal definitions of the centrality exist; we explain them in details in section 2.2. The concept of *social groups* is similar to what groups and clusters are in graph theory, *i.e.*, a set of actors. Groups of two actors are called *dyads* and groups of three actors are called *triads*.

The notion of equivalent classes of actors or *positions* is specific to SNA. To explain this concept, we need to define the relation *equivalence*. It consists in comparing the patterns of connection of two actors: *structural equivalence* exists when two actors have similar patterns of connection to the same actors; *regular equivalence* exists when two actors have similar patterns of connection to (possibly) different actors [Fau88]. Actors structurally equivalent are placed in a class of equivalence called position. While positions are collections of actors similarly embedded in the network, *roles* are collections of relations and their associations. As finding central actors, social groups, positions and roles are fundamental in SNA, these concepts are further defined in section 2.2.

2.1.3 Visualization

The most common visual representation of a graph is called a *node-link diagram*. The visual counterparts of vertices are *nodes* and the visual counterparts of edges are *links*. However, very often, the terminology from graph theory and visualization is not distinguished and the term "node" is often used for "vertex". The visual representation of an adjacency matrix is called a *visual adjacency matrix* or simply a *matrix representation*. Sometimes, it can be referred to as a *visual similarity matrix*. The *layout* of a node-link diagram is the position of its nodes in the space. The corresponding notion for matrices is the *ordering* of their rows and columns. Layouts are computed to optimize a set of *aesthetic rules*, improving the visualization readability. For example, one aesthetic rule is to minimize edge crossings. Some graphs, called *planar* graphs, can be drawn in 2D without any edge crossing.

2.2 Social networks analysis

2.2.1 What are social networks?

A social network is composed of actors connected by relationships or relation ties. Actors can be people or groups of people (such as organizations, cities or countries). Relationships range from depicting kinship, friendship or communications such as email exchange or phone calls for people, to transit maps or transactions such as products import and export between countries.

A large variety of social networks exist, with different structural properties, including:

· **Genealogy trees.** These networks may have cycles but are generally close to a tree structure with a low density, their relations are directed and they possess several types of links (parent, husband or wife);

· **Coauthorship networks.** These networks of researchers are small-world, they have undirected relations but generally weighted by the number of publications coauthored;

· **Communication or Trade network.** These networks describing product exchange between countries for example are almost complete, with directed, weighted relations representing the amount of products exchanged;

· **Affiliation network.** These networks are bipartite, including two types of actors and relations depicting how they relate to each other, such as which persons subscribed to which associations for example.

2.2.2 Social networks analysis

Social network analysis (SNA) began more than 70 years ago, with the empirical work of Jacob Moreno. Wasserman and Faust [WF94] provide a good introduction to the SNA methods. We can classify SNA methods in three categories: statistical, structural and exploratory analysis. These three methods are complementary and allow analyzing the network from different perspectives. They are generally performed at different levels of analysis (vertex, dyad, triad, group or whole network). We briefly outline their philosophy before presenting a comprehensive overview of the methods used in each type of analysis.

Statistical modeling constitutes a large part of social network analysis. This analysis process is based on hypotheses testing or model fitting following a set of assumptions about the data. This approach can be schematized as:

$$\text{Data collection} \longrightarrow \text{Hypotheses} \longrightarrow \text{Model} \longrightarrow \text{Analysis} \longrightarrow \text{Conclusions}$$

The second category aims at describing the structural properties of the network, without any assumptions about the data. This approach can be schematized as:

$$\text{Data collection} \longrightarrow \text{Hypotheses} \longrightarrow \text{Analysis} \longrightarrow \text{Conclusions}$$

An alternative perspective for analyzing social networks is exploratory data analysis (EDA), introduced by John Tukey [Tuk77] in 1977. EDA is qualified as *exploratory* in opposition to *confirmatory*. The EDA process differs from conventional approaches in that it proposes to directly analyze the data, without prior assumptions or modeling, in order to form hypotheses or discover insights. Techniques of EDA are based on the idea that one has to "see" the raw data to understand it. As most of the EDA techniques are graphical, we named this category of methods *visual analysis*. This approach can be schematized as:

$$\text{Data collection} \longrightarrow \text{Analysis} \longrightarrow \text{Hypotheses} \longrightarrow \text{Validation} \longrightarrow \text{Conclusions}$$

2.2.3 Statistical analysis

The statistical approach to social network analysis [Lyn07] has been used since the beginning of social network analysis in the 1930's. We can divide these methods in two categories: descriptive and inferential statistics. The goal of each category is very different.

Descriptive statistics Descriptive statistics provide a quantitative overview of a network. General descriptive measures exist for each type of data object. For example, statistics on the whole network, on connected components and on groups of actors include the number of actors, of relationships, of connected components, the density, the diameter, the average distance, the average degree or the minimum or maximum degree. Statistics on vertices and edges include the degree or the eccentricity. Statistics are also computed on the vertex and edge attributes, indicating the mean value for numerical ones or their distribution. Descriptive statistics constitute an initial analysis and can help discover general features such as the small-world property but are rarely sufficient to answer a specific question or study a particular behavior.

Inferential statistics The role of inferential statistics is completely different; it consists in measuring how plausible a model is. The main idea is to verify if a pattern found in the data is typical of the studied population or if it is a random result. To test their hypothesis and validate their results, a large number of social scientists use either standard statistical models or specialized ones created for social network analysis purposes.

Standard techniques Standard methods include hypothesis testing: answering yes/no to particular questions. *Student's t-test* and *ANalysis Of VAriance* are common techniques. Correlation and regression procedures are also used to discover if two factors affect each other (correlation) and identify potential trends (regression).

Specific techniques Statistical models specifically designed for social network analysis are presented in [WF94]. We give examples of three common techniques: dyadic and triadic techniques as well as statistical modeling of the full network.

A number of methods are based on the *statistical analysis of dyads and triads* within the whole network. Statistical models on dyads answer a number of questions such as:

· *Reciprocity.* "Do actors tend to reciprocate relationships?"
i.e., if $r_{u,v}$ exists, is the probability that $r_{v,u}$ exists higher than in a random case?

· *Multiplexity.* "Do actors tend to multiply several types of relationships?"
i.e., if $r1_{u,v}$ exists, is the probability that $r2_{u,v}$ exists higher than in a random case?

· *Exchange.* “Do actors tend to exchange different types of relationships?”
i.e., if $r1_{u,v}$ exists, is the probability that $r2_{v,u}$ exists higher than in a random case?

Statistical models on triads answer a number of questions such as:

· *(In)Transitivity.* “Does the friend of my friend tend to be my friend?”
i.e., if $r_{u,v}$ and $r_{v,w}$ exist, is the probability that $r_{u,w}$ exists higher than in a random case?

· *Closure/openness.* “Do my friends tend to be friends?”
i.e., if $r_{u,v}$ and $r_{u,w}$ exist, is the probability that $r_{v,w}$ exists higher than in a random case?

· *(Dis)Similarity.* “Do actors having the same friend tend to be friends?”
i.e., if $r_{u,w}$ and $r_{v,w}$ exist, is the probability that $r_{u,v}$ exists higher than in a random case?

· . . . and so on with the other configurations.

The principle of statistical modeling of whole social networks is to generate vertices and edges according to various probabilities. Many types of models exist, from random generation of graphs to models with complex parameters. We only give here a brief overview of the models used in social network analysis.

The simplest model is the *random generation* of graphs [ER59], which randomly generates edges given a set of vertices with a uniform degree distribution. Watts and Strogatz [WS98] show that this model fails to capture the properties of most real networks. Instead, they proposed a *small-world model* attempting to reproduce the small-world effect observed in real networks. The principle of the basic model is that the probability of adding an edge between two vertices depends on how their neighbours are already connected. The more their neighbours are connected; the more likely they will be connected. Other models aim at generating networks with a power-law degree distribution also called scale-free networks (only one characteristic of small-world networks). These *scale-free generators* [BA99] are based on the concept of preferential attachment: the more a vertex is connected the most likely it will receive a new connection. Finally, the *p-model* family [HL81, vDSZ04, RPKL07] is a category of model based on the probability of generation of dyads. The principle of p1 is to create dyads depending on the actors attribute. Evolutions of this model add a dependence between the probabilities of each actor and introduce social phenomena such as reciprocity.

A model is effective if the resulting synthetic network exhibits the same properties as real networks. However, these models face one main challenge: it is hard to capture the properties of real networks. Even when a set of properties is captured (such as the three properties of small-world networks), it is difficult to match all of them. We show in Chapter 8 how visualization can help identifying the weaknesses of these generators.

In the next section, we present the structural analysis of social networks, which do not rely on statistical assumptions. Directly analyzing the raw data, social scientists aim at discovering structural and locational properties of social networks, attempting to answer several high-level questions such as who are the “important” actors? How are they connected to others? Statistical and structural approaches are complementary; social scientists almost always perform both of them.

2.2.4 Structural analysis

Structural analysis of social networks is constituted of three main high-level tasks [WF94]: identify ”important” actors, identify “social groups” and identify “roles” and “positions”. These three tasks rely on a number of intuitive concepts such as the “importance” of an actor or the notion of “group”. The main difficulty when performing this analysis is that there is no clear unique formal measure for each theoretical concept. Instead, analysts have a panel of computational measures providing them with different perspectives on the concept. We attempt to introduce the concepts and present the common measures for each of them.

1. Identify the “most important” actors. The concept of importance of an actor is related to the actor's location in the network. Most important actors are involved in many relationships with other actors and placed at strategic locations in the network. The notion of *centrality* [Fre70] emerged in the late 1940's and represents how an actor is involved with others in the network. Several formal measures can be computed to match different facets of centrality:

. *degree* centrality is simply the number of connections of an actor. The actor connected to the highest number of others has the highest degree centrality;

$$C_D(v) = \text{deg}(v) \quad \text{[Fre70]}$$

. *betweenness* centrality is a measure of how often an actor is on a shortest path between two other actors of the network. For example, if a network is divided in two parts only connected by one actor, this actor will have a very high betweenness centrality because he will be on every path from one part to the other;

$$C_B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}, \quad \text{[Fre77, Bra01]}$$

with $\sigma_{st}(v)$, the number of shortest paths from s to t passing through v

. *closeness* centrality is a measure of how close an actor is to all others; it is based on the shortest path from an actor to all others in the network. For example, if an actor is directly connected to all others, it will have the highest closeness centrality;

$$C_C(v) = \frac{1}{\sum_{w \in V} d_G(v,w)} \quad \text{[Sab66]}$$

. *eigenvector* centrality assigns relative scores to all actors in the network based on the principle that connections to actors having a high score contribute more to the score of the actor in question.

$$C_E(x_i) = \frac{1}{\lambda} \sum_{j=1}^{N} A_{i,j} x_j \quad \text{[Bon72]}$$

with λ, being a constant and A, the adjacency matrix of the graph

Many other measures of centrality have been proposed: based on eccentricity [HH95], radiality [VF98], authority [Kle99] or pagerank [BP98] for hyperlink networks; the four measures presented here are the most widely used. Because of the difficulty of mapping an abstract concept to a single formal measure, social scientists compute a set of these measures, then compare the various results to match their interpretations.

For directed graphs, the notion of *prestige* refines the notion of centrality. An actor is prestigious if he *receives* many connections (see [WF94] for more details).

2. Identify "social groups" of actors. Social groups or cohesive subgroups are subsets of actors within the network that are strongly connected with each other. The concepts of social cohesion and of subgroup have many possible definitions and several formal measures. In the social network analysis literature, four properties have influenced the definition of cohesive subgroups: mutuality of relations, closeness or reachability of subgroup members, frequency of relations among members and the relative frequency of relations of members compared to non-members.

· *cliques* are based on a complete mutuality of relations: every member is linked to all other members;

· n*-cliques* are based on the notion of reachability: the largest distance between every member of the clique is n (refined notion of n-clans and n-clubs also exist, see [WF94] for more details);

· k*-plexes* and k*-cores* are based on the degree: every member has a degree at least equal to k in k-cores and may be lacking relations no more than k in the k-plexes;

· λ*-sets* are based on edge-connectivity within the subgroup: removing relations should not disconnect the subgroup.

In addition to these formal definitions, a large number of methods exist [JMF99] to find cohesive subgroups in a social network and more generally in a graph. This research area is named *graph clustering* and includes for example classes of methods such as hierarchical clustering, multidimensional scaling or graph partitioning. The general principle is to define a measure of similarity or dissimilarity (distance) between actors in order to extract clusters (cohesive subgroups). We give an overview of graph clustering methods in section 2.4.2.

3. Identify "roles" and "positions". The dual notion of social roles and social positions relates to the structural similarities and patterns of relations within a social network. Positions are equivalent classes of actors having similar patterns of connection. Roles are collections of relations between actors or positions. For example, when analyzing the social network of a research laboratory, two social positions may emerge: "supervisor", connected with many other actors, and "student", connected to a single actor. An analyst may observe a role such as "advise", always occurring from "supervisor" to "student" and grouping a set of relations such as "have meeting with", "exchange email with" and "co-author article with". Roles can also be based on the combination of the relations, for example in a kinship network, the role "aunt of" could emerge as a combination of "mother of" and "sister of".

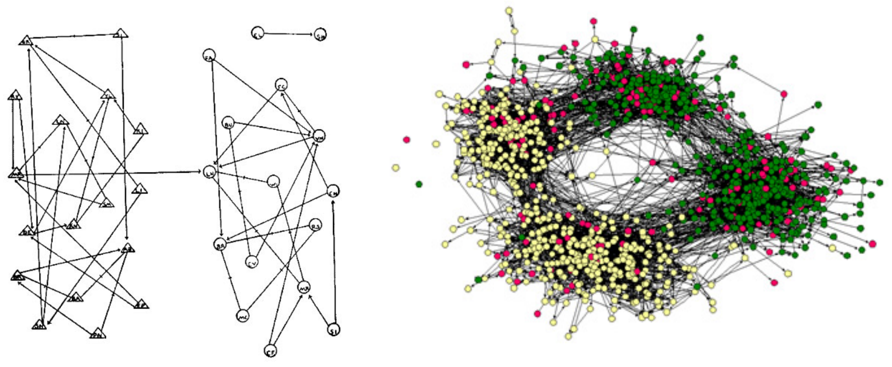
A widely used technique in social network analysis to perform positional analysis is *blockmodeling* [DBF05a]. The principle is to extract positions and analyze how they relate to each other. The blockmodels, first introduced in [WBB76] refer to models of each position; these can be predefined by the analyst or estimated statistically (stochastic blockmodeling [HLL83, WA87]).

Methods to analyze role structure have been introduced in [BW76] and referred to as *global role analyses*. A role structure is a general description of how relations are associated in the network. The main goal is to describe similar features of the structures and measure the degree of similarity between the structures. These methods are based on relational algebras: notations describing combinations of relations.

From the real world to the formal world, and back. To a question or concept such as important actors corresponds a set of formal measures to be computed and interpreted before getting any clue about the answer. Visual exploration faces a similar challenge. However, it aims at reducing the gap between reality and analysis by looking directly at the data.

2.2.5 Towards visual analysis

From the early work of Jacob Moreno, visually representing social networks is common when performing social networks analysis [Fre00]. Visual representations are a powerful way to convey information and help illustrate high-level concepts such as central actors or groups. As a picture is worth a thousand words [CMS99], Figure 2.3 illustrates these concepts using node-link diagrams, the most common representation of social networks.



(a) Central actors (b) Social groups

Figure 2.3: Intuitive notions of central actor (a) and social groups (b) in a visual representation. (a) friendship among boys and girls (triangles and circles) of a fourth grade school by J. Moreno. The central actors emerge as the persons linking the boys group and girls group. (b) friendship amongst high school students according to ethnic origin from the work of James Moody, created by M. Newman. Four social groups clearly emerge in this picture.

Until recently, visual representations were not used as a proper exploration technique but rather to illustrate prior findings for confirmation and communication purposes. It is important to distinguish the visualization for communication purposes (visual communication) and the visualization for exploration purposes (information visualization or data visualization) as their goal and means differ.

Visualization for communication often leads to the creation of a static representation, illustration of a result or a theory. It aims at communicating a message and constitutes a perspective on the data. In visual analytics, the term story telling is often used: the analyst has a story, he tells it visually. Carefully designing such representation is important to avoid communicating lies as shown by Tufte in [Tuf83].

Visualization for exploration has a different purpose. The goal is to explore data, *i.e.*, represent it from different perspectives to get insights (patterns, outliers or correlations for example). Therefore, visual exploration requires multiple and/or interactive views as well as various representations to crosscheck potential findings (avoiding potential visual artifacts). Visualization for exploration has lead to its own research field: information visualization. Related work is presented in section 2.5.

The need for visualization of graphs and networks has been constantly growing to become a distinct research area of graph theory: graph drawing [DETT98]; and of information visualization: network visualization [CMS99]. In the next section, we present an overview of the graph drawing field.

2.3 Graph drawing

The whole research area of *graph drawing* [DETT98] is dedicated to designing algorithms for laying out node-link diagrams.

2.3.1 Graph layout aesthetics

The objective of layout algorithms is to find 2D or 3D coordinates for placing the nodes of a graph in the space. There are also several ways of drawing the links but we will not detail this work. One major issue of graph drawing is to optimize the readability of node-link diagrams. The assumption is that drawing a graph “nicely” helps to understand it better and possibly faster. Defining the important aesthetic criteria and how they affect the understanding of a graph has been the topic of several works [PCJ95, Pur97, WPCM02]. Some specifically studied the effect of these aesthetic criteria in interpreting social networks [MBK97, BMK95]. Common aesthetics criteria include:

. minimization of *node overlapping* so labels are readable;

. minimization of edge lengths and of the maximum length of an edge, more generally minimize the *edge length variance* as it may mislead users that tend to interprete node linked by a longer edge as further apart than they actually are [MBK97];

. minimization of the number of *edge crossings* as it is more difficult and may be ambiguous to follow a link with many crossings;

. minimization of the number of *edge bends* as it makes connections more difficult to track;

. minimization of the number of *small crossing angles* and more generally maximization of the number of orthogonal edges making it easier to track edge crossed by others;

- maximization of the *path continuity* as it helps tracking shortest paths for example [WPCM02];
- maximization of the *symmetries* of the graph, which is known to facilitate the creation of a mental map;
- minimization of the *total drawing area* to optimize the visual scan and fit smaller displays;

Additional criteria include the *computation time and complexity*, the *aspect ratio* which is important when fitting a standard computer screen and the *stability* and *predictability* of the layout to help users build a mental map [PHG06], preserving it when adding or removing a node.

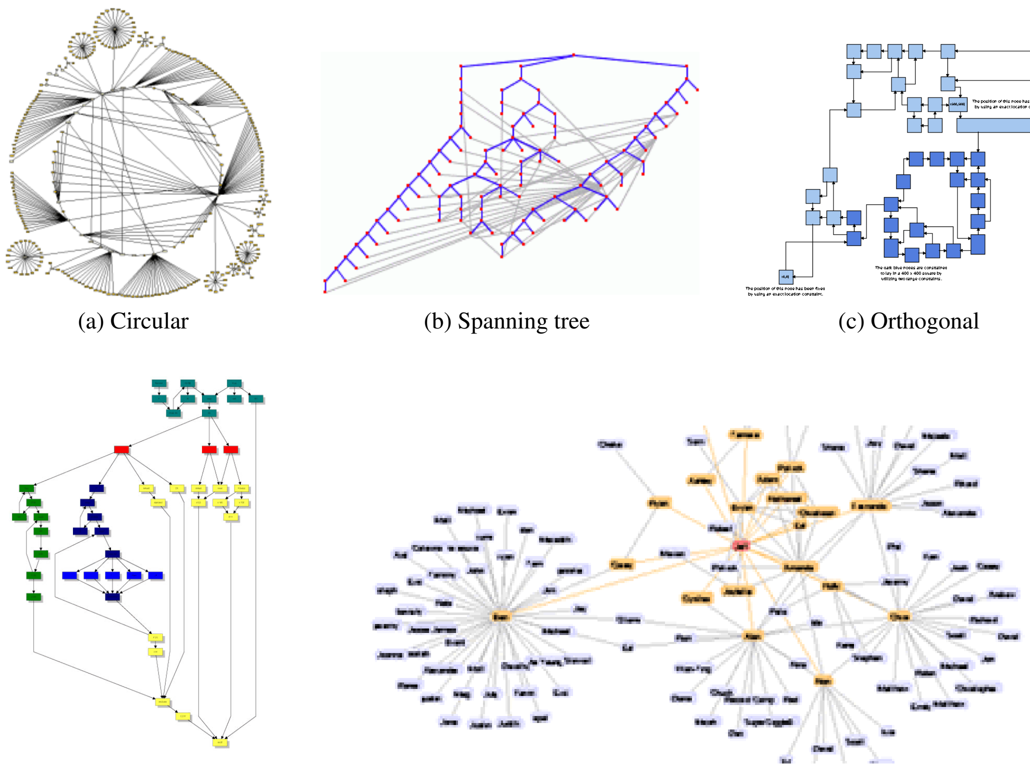
Producing a drawing that satisfies all these criteria is impossible as they interfere with each other. For example, maximizing the symmetries of a graph may increase the number of edge crossings. Several experiments [Pur97, WPCM02] showed that the most important criterion was edge crossing minimization when performing tasks such as finding the shortest path between two nodes. However, aesthetics criteria depend on the context of use and the tasks performed. For example, when analyzing a social network, it is important to reduce node overlapping so that all labels are readable. Grouping nodes in clusters is preferable but potentially increases the number of edge crossings. In the next section, we present common layout algorithms and discuss how they improve different sets of aesthetics criteria.

2.3.2 Graph layout algorithms

A number of surveys and books exist reviewing the various layout algorithms; [DETT98] being a good introduction. This topic is an entire field of research and has an annual international conference called Graph Drawing dedicated to it. Progress in this domain is important and we only report here an overview of the different methods.

Circular layout. The circular layout [DMM97] (Figure 2.4(a)) is a common layout in social network analysis. The principle is to place all nodes around a circle. This method suppresses node overlapping and has the advantage of being very fast to compute. However, links connecting nodes all around the circle cross each other, their cross-angle can be small and the length variance is not minimized. Methods exist to minimize edge crossing but these factors may dramatically decrease the readability of the diagram for relatively dense graphs.

Radial and Tree layouts. A large number of algorithms have been designed for rooted trees, which are objects easier to lay out as they have a starting point (the root) and no cycles (which allows the use of recursive algorithms). The most popular algorithm to draw trees is certainly the one presented by Reingold and Tilford in 1981 [RT81]. This algorithm reflects the hierarchy of the data by drawing parents in the tree before the children (top-down or left-to-right for example). Another common layout is the radial layout [Ead92]. The root is placed in the center and nodes are positioned in successive concentric circles according to their depth. Some of these algorithms have been extended to graphs. The majority of techniques visualizing graphs as trees first compute a spanning tree (a spanning tree contains all the nodes of the graph and the maximum number of edges are filtered to obtain a tree), lay it out using a tree layout algorithm, and finally draw missing links. Examples are provided in Herman et al. [HMdRD99] (Figure 2.4(b)).



(a) Circular

(b) Spanning tree

(c) Orthogonal

(d) Hierarchical

(e) Force-directed

Figure 2.4: Five layouts for node-link diagrams. (a) traditional circular layout where all node labels are readable by yFiles. (b) layout by Latour, a minimum spanning tree is computed and additional links added. (c) Orthogonal layout by Tom Sawyer software, particularly used to layout software diagrams. (d) Hierarchical (k-layered) layout by the GoVisual library. (e) force-directed layout by Prefuse, the most popular layout based on a physical model metaphor.

Hierarchical layout. K-layered layout or hierarchical layout first appeared in the seminal work of Sugiyama [KST89] (Figure 2.4(d)). This class of algorithms computes a layout for directed acyclic graphs (DAGs) in three steps: (1) layering, (2) crossing minimization, and (3) final coordinate assignment. The layering consists in assigning each node to a layer, which often preserves the path continuity. This layer can be computed from topological properties (such as the distance from a given source if the graph is directed) or from a given ordered node attribute such as time for example. The crossing minimization step aims at reducing edge crossings while the third step assigns final coordinates (layer and position on this layer) to the nodes aiming at suppressing label overlap. Usually, these layouts also preserve the number of bends in the links. To layout any directed graph (with cycles), an additional step is required, consisting in reversing a set of arcs in order to suppress cycles. Unfortunately, both optimal crossing minimization and cycle reduction are NP-complete problems.

Orthogonal layout. Orthogonal layouts, issued from the seminal work of Tamassia [Tam87], replace links by a set of vertical and horizontal lines, trying to optimize the number of bends and crossings (Figure 2.4(c)). This representation is generally preferred for schematic diagrams such as UML class diagrams. A set of methods called Topology-Shape-Metrics [EKS03, Eig03] compute layout for UML diagrams in three steps: (1) planarization, (2) orthogonalization, and (3) compaction. Planarization consists in adding artificial nodes to a non-planar graph to transform it in a planar one. Orthogonalization assigns final coordinates to nodes using metrics such as minimizing the total edge length and compaction removes artificial ones. The most complex step is the planarization, which is often very difficult to achieve with real-world graph of a large size.

Topological layout. An interesting idea consists in taking advantage of the topological properties of a graph to compute its layout. The principle of TopoLayout [AMA07] is to use different algorithms to layout parts of a graph having different topological features. This technique implies preprocessing the graph to identify a set of topological features. For example, if a graph is composed of two bicomponents, one very dense and one exhibiting a tree structure; it is interesting to layout the dense one with a circular layout, so the labels are readable while keeping the visual impression of high connectivity, and to use a traditional tree layout for the tree-like structure.

Force-directed layout. Force-directed layouts algorithms [Ead84] are probably the most popular graph layout algorithms (Figure 2.4(e)). The strength of these algorithms is that they do not rely on any specific graph properties (contrary to the previous approaches), making them directly usable for any graph to analyze. Force-directed layouts or spring layouts are based on the analogy with a physical model consisting of repulsive masses (the nodes) linked by springs (the links). An interesting property is the motion of nodes slowly going into place, as algorithms are iterative. Their complexity is usually n^2. Thus, this becomes very time consuming for large graphs. A large number of variants exists with different properties: most popular include Fruchterman and Reingold [FR91] introducing a friction factor on the nodes to reduce the number of iterations of the algorithm, Kamada-Kawai [KK89] with edge variance minimized or LinLog [Noa05] with repulsive edges instead of nodes. This class of algorithms is particularly interesting for social network analysis as visual groups of nodes emerge naturally. However, these algorithms do not provide deterministic models: a different configuration is computed at each invocation of the program.

The key issue of layout algorithms is *scalability*. Computing layouts for large graphs implies high computational time and high memory space (due to the quadratic nature of the algorithm) added to another major problem: the size of the display, which is often limited to a standard computer screen.

2.3.3 Handling large graphs

Large graphs induce readability problems because of the high *number of nodes* (size) to display and because of the *clutter* (density) induced by having many connected nodes in a limited space. Very few libraries provide optimized implementations of the layouts presented in the previous section able to handle large graphs in acceptable time. NicheWorks is an example, providing faster force-directed algorithms by smartly initializing the layout, reducing the number of iterations and only rendering the visible part of the graph on the screen. Tulip [Aub03], a C++ library issued from the PhD work of David Auber [Aub02], is a second example, handling graphs with more than a million nodes using mechanisms to reduce the time and memory consumption. However, if these optimized algorithms are usable, they still suffer readability problems due to edge clutter.

A recent experimental study [HJ06] investigates the performance of different categories of algorithms (coming from multi-dimensional scaling or spectral graph theory) to draw large graphs. The authors compared a traditional force-directed algorithm GRIP [GK01] to

· *multi-scale* or *multi-level* algorithms (FMS [HK00a] and FM^3 [HJ04]): computing the layout of a recursively coarsened graph (shrinking nodes close together to a single one);

· *spectral* algorithms (ACE [KCH03]): computing the eigenvectors of the Laplacian matrix associated to the graph [Chu97], the second smallest and third smallest can be assigned to x and y coordinates shown to be best solution for a two-dimensional embedding [Hal70];

· and *high-dimensional embedding* algorithm (HDE [HK00b]): first computing a limited number of pivot-nodes associated with vectors representing the distances of nodes of the graph to pivots, then projecting this collection of n-dimensional vectors to two dimensions using the principal component analysis.

The experiment showed that these three classes of algorithms are very fast compared to traditional methods and produce good results with artificial graphs (with regular features such as meshes for example). Real-world graphs drawings seem to be of lesser quality but these algorithms can still be tried for little cost due to their speed.

Several other strategies have been investigated to overcome the scalability problems of node-link diagrams:

1. Reduce the complexity (size and density) of the data to represent by using filtering or clustering (section 2.4);

2. Increase the amount of data (size) that can be visualized in a given display by using more than 2D or investigate alternative graph representations (section 2.5.1);

3. Provide interaction techniques to dynamically reduce the complexity of the data and navigate through large visual space (section 2.5.2).

2.4 Data reduction

2.4.1 Graph filtering

An obvious technique to reduce the size of a graph is to remove some of its vertices. Similarly, to reduce its density, edges can be filtered.

Filtering. Filtering allows analysts to select what perspective of the graph to explore. Filtering is done according to a given *metric*. In the graph visualization survey [HMM00], Herman et al. refer to the concept of node metric: “a measure that is associated with a node in the graph”; however we simply use the term metric, as measures can also be associated with edges of a graph. A metric can be computed according to *structural properties* of the graph: for example, one could filter by connected components. Another example of a structural metric is the degree of a vertex or the betweenness of an edge. Filtering the vertices or the edges can also be done according to metrics based on *data properties* of the network: for example, when representing a network of coauthorship including data for several years, the data can be filtered by year. If the edges are weighted, one can choose to display only edges above a certain threshold in order to reduce the graph density for example. If vertices or edges are categorized, displaying each category separately may dramatically reduce the size and density of the network displayed. Finally, filtering can be done according to a combination of structural and data metrics on both vertices and edges.

Sampling. The notion of sampling is a bit different as it is based on the idea that a subset of the data can be representative of the entire dataset. Therefore, the user does not choose what to filter as previously. Instead, an automatic technique is used to select a representative graph. Moreover, theoretically, a single drawing is enough as it is supposed to exhibit the same characteristics of the whole graph. A common technique is *random sampling*, a random selection of a subset of vertices. Other methods include *stratified sampling*, in which the elements are separated in categories (say, according to a given attribute) to ensure all categories are represented and *cluster sampling*, in which the elements are selected according to specific areas (either in space or as a combination of attributes or structural properties). Frank studied a large panel of sampling techniques for social networks analysis [Fra77, Fra78, Fra80, Fra88]. The main issue with sampling is that it often does not work for graphs (especially small-world networks) as structural properties can dramatically change. In the case of small-world networks the distribution of the degree of vertices follows a power-law, *i.e*, most nodes have a low degree and only a few have a large degree. Therefore, random sampling is very likely to filter nodes with a high degree, producing a very disconnected graph. Another problem for social network analysis is that communities tend to disappear.

Drawing filtered graphs

While generally the filtered elements of the graph are removed and the visual representation redrawn without them, other techniques prefer to keep track of the context by either: *ghosting*, de-emphasizing elements by drawing them in a light color or smaller for example; or *hiding* the elements but keeping the original configuration of the representation. In all cases, only a partial view of the graph is given and several drawings or an interactive visualization (section 2.5) are required to explore the entire graph.

2.4.2 Graph clustering

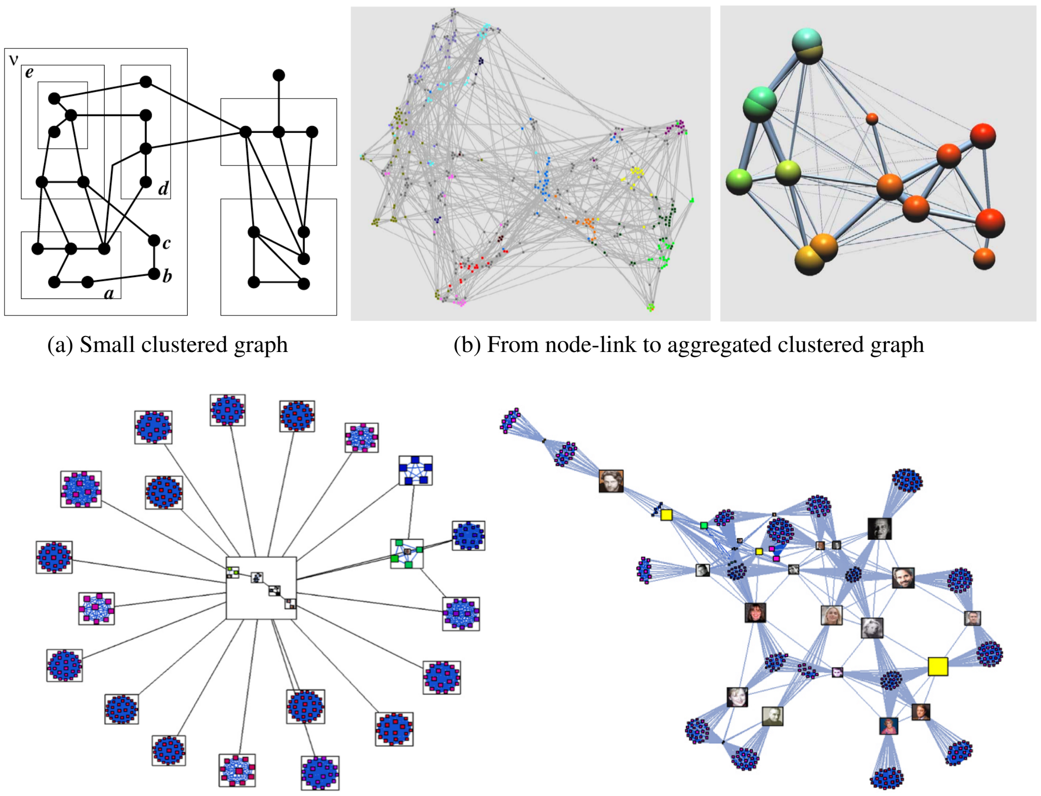
Another approach to handle large graphs is to reduce their size by grouping individual nodes into super-nodes or clusters. Similarly to filtering, clustering is done according to given metrics on vertices and edges. As we mentioned before, graph clustering (and data clustering in general) can also be used for analysis purposes and has a variety of applications such as finding similar genes in biology or identifying groups of customers potentially interested in a new product in business intelligence. There is a substantial amount of work in data clustering, in this section we only present the different classes of methods, see extensive surveys [JMF99, Ber02] for more details. [1]

Hierarchical clustering. The principle of hierarchical clustering is to recursively *agglomerate* individual elements into clusters (bottom-up) or to recursively *divide* an initial cluster regrouping all the elements (top-down) into smaller ones. This clustering is qualified as hierarchical because its output is a tree of clusters. Cutting this tree at a certain level gives a finite number of clusters as shown in Figure 2.5(a). Both agglomerative and divisive hierarchical clusterings rely on *metrics* to create clusters. A metric allows comparing individual elements; common metrics compute a similarity or a distance such as the Euclidian or Manhattan distance. Many variants of hierarchical algorithms exist, comparing clusters in different ways. For example the complete, single and average linkage algorithms [Ols95] respectively use the maximum, minimum and mean distance between the elements of two clusters. Other algorithms use more sophisticated functions such as Ward's algorithm [War63] merging the two clusters producing the minimal increase of variance.

Graph partitioning. Graph partitioning [Els97, Fja98] consists in dividing graphs into partitions (non overlapping sets of elements). Checking all possibilities to assign elements to clusters is not possible; this is why a category of algorithms performs *iterative optimization*. These greedy algorithms, such as the first heuristic presented by Kernighan and Lin [KL70], are based on the relocation of the elements to optimize an *objective function* starting from a random initial partition. The most popular class of algorithms performing graph partitioning are the k-means algorithms [Har75]. Several variants exist [KMN$^+$02, DH04] but they are generally composed of two steps performed iteratively until convergence: first split the graph in *k* partitions (*k* chosen by the user) and compute a mean point or centroid for each according to a given metric or an objective function, then reassign the elements to their closest centroid and iterate until the clusters are stable. In practice, k-means converges very fast, which explains why it is widely used.

Another class of methods is based on *graph-theoretic concepts* or structural properties to compute partitions. A traditional algorithm [Zah71] is based on the construction of a minimal spanning tree followed by the deletion of a number of edges to create clusters. More recently, an efficient algorithm built upon the social network analysis measure of edge betweeness (how often an edge is part of a shortest path between two actors of the network) (see section 2.2) has been presented by Girvan and Newman [GN02]. The principle is to iteratively compute a betweenness score for each edge of the graph and to remove the edge with the highest betweenness to disconnect the graph until obtaining *k* connected components or clusters.

[1]These techniques address the general problem of data clustering, for graph clustering simply replace the term element by vertex or node.



(a) Small clustered graph (b) From node-link to aggregated clustered graph

(c) Inter-cluster connectivity loss (d) Clustered graph with details

Figure 2.5: Drawing clustered graphs. (a) Feng and Eades draw hierarchical clusters, all details are preserved but the scalability problem remains. (b) Van Ham et al. draw aggregated clustered graphs; inter-cluster and intra-cluster details are lost. (c) Auber et al. draw clustered graphs, inter-cluster connectivity details are lost. (d) Auber et al. propose a compromise where impressions of connectivity inter and intra-cluster are perceivable (but their details not readable).

Probabilistic clustering. A class of clustering methods is based on the idea that elements are following a *mixture model*, *i.e.*, a combination of several probability distributions. These different distributions are considered as “natural” clusters. The goal is to identify each distribution as well as its parameters (mean, variance, etc). The Expectation Maximization algorithm [DLR77] can be used to estimate these parameters.

Another class of probabilistic methods called Markov clustering [Don00] is based on the idea of performing *random walks* in a graph. The intuition is that a random walk performed in a cluster is likely to visit many of its elements, being trapped because of the density of the connections. Several optimized variants exist [HK01, PL06].

Spectral clustering. This class of methods is based on the computations of the eigenvectors of the adjacency matrix or the Laplacian matrix of a graph. The k smallest eigenvectors of the Laplacian matrix can be mapped to k clusters minimizing the total edge length [NJW01], similarly to spectral graph layout algorithms (section 2.3.2). For more information on spectral graph theory see [Chu97].

Computational intelligence methods. Finally, other clustering methods use techniques from artificial intelligence or more precisely computational intelligence [Eng02]. These methods include artificial neural networks, self-organizing maps or Kohonen maps and evolutionary approaches with the use of genetic algorithms. More details on the use of these methods to compute data clustering are given in [JMF99].

Drawing clustered graphs

Ideally, the output of graph clustering techniques is a set of clusters regrouping vertices having a similar trend. To gain space and solve the scalability problem of node-link diagrams, vertices appearing in the same cluster can be aggregated in a single representative super-node (Figure 2.5(b)). However, if this can partially solve the initial problem, several new problems arise due to the loss of detailed information:

- How to represent the connectivity intra-cluster?
- How to represent the connectivity inter-cluster? For example, how to differentiate two clusters with all nodes connected to each other from two clusters connected by a single link? Similarly, how to represent one node of a cluster connected to all the others in the second cluster?
- If the nodes have attributes, as it often happens in social networks, how to compute the aggregated attributes of the super-node? For example, numerical attributes such as the age of a person can be averaged but what about nominal or categorical data such as the name or sex?

In addition, interesting approaches such as fuzzy clustering are based on the notion of membership to a cluster. In this case, a node can be in several super-nodes or overlapping clusters, which raises a new problem.

Several approaches have been investigated to draw clustered graphs. Figure 2.5 describes four techniques.

2.5 Information visualization

In this section, we present techniques drawn from the field of information visualization to handle large datasets and support their exploration. Information visualization consists in visual representation associated with interaction techniques. In a first part, we present techniques to increase the display space as well as alternative representations to handle large graphs. In the second part, we present interaction techniques supporting both the exploration process and the visualization of a large quantity of data. Finally, we present a comprehensive review of network visualization, an emerging area of information visualization.

2.5.1 Visual representation

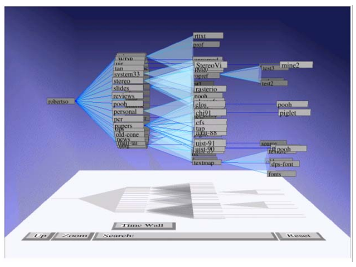
To solve the problem of scalability of node-link diagrams, researchers first focused on the visual representation, attempting first to increase the visual space, then to find alternative representations.

More than 2D

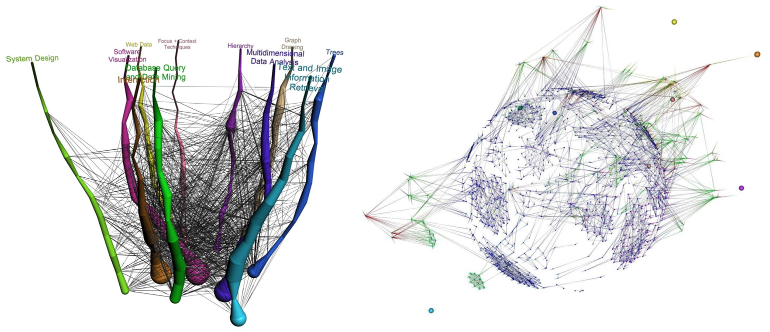
3D space. In order to "increase" the display space, a number of researchers chose to visualize graphs in 3D instead of 2D. The idea is that the third dimension would provide additional space to draw larger graphs. The best-known example in information visualization is the ConeTree [RMC91] (Figure 2.6(a)). The principle is to draw hierarchical trees as a hierarchy of cones; the shadow of the 3D model projected in 2D provides the user with an overview. Using ConeTree to visualize large trees uses less space (as the nodes are arranged in cones) but requires interaction to navigate through all of them. Almost all algorithms to layout node-link diagrams are extendable to 3D. For example, Tulip, SemNet3D and WilmaScope generate aesthetic 3D graph representations as shown in Figure 2.6(b,c,d).

The main issue when dealing with 3 dimensions is the occlusion problem: users feel that representations with three dimensions are "cluttered" and less efficient [CM02]. To overcome this problem, occlusion management techniques exist [ET07] such as x-ray vision, helping users see through objects placed in the foreground or providing multiple viewports to see the representation from different angles. The use of navigation techniques is also a support for disambiguating 3D graph representations. Unfortunately, navigating in 3D representation is disorienting [SP96, CM01] especially without stereoscopic or motion cues [WF96] and using a 3D input device to control 2D screen space is difficult to carry out. User studies comparing 3D to 2D interfaces for information visualization show that if users are more attracted by 3D applications, their performance is not improved and can even be deteriorated for some tasks [War04, SP04]. For example, Cockburn and McKenzie [CM00] showed that users were slower with ConeTree than with a traditional 2D tree explorer and that their performance decreased rapidly as the tree became more complex (more branches).

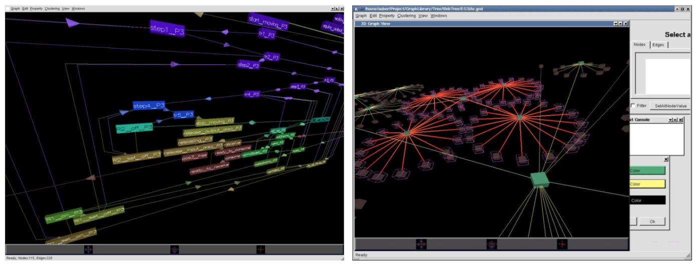
Hyperbolic space. Another way to increase display space is to use hyperbolic geometry instead of Euclidian geometry. In hyperbolic space, the parallel postulate is rejected, meaning than two parallel lines diverge away from each other. Considering a graph drawn on a hyperbolic disk, the space grows exponentially with the distance to the center, giving more space to draw large



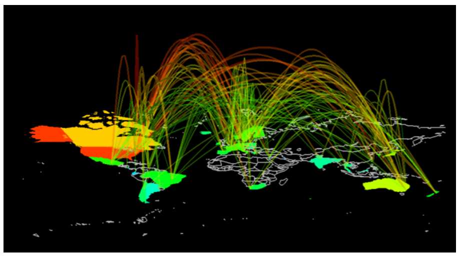
(a) ConeTrees



(b) WilmaScope

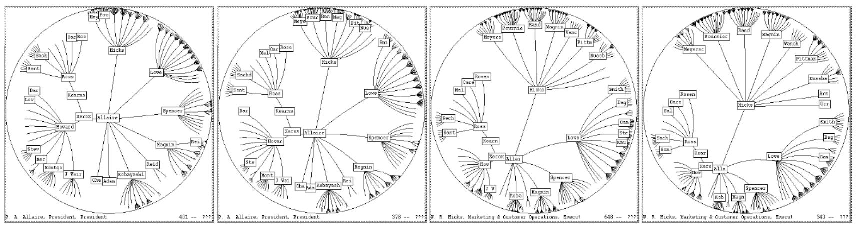


(c) Tulip

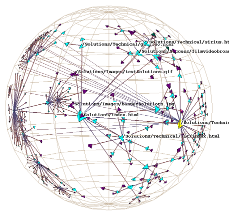


(d) Web Traffic

Figure 2.6: Drawing graphs in 3D. (a) Cone trees by Robertson. (b) 3D representations of a tree and a hierarchical layout created by Tulip, toolkit of David Auber. (c) 3D representations of an evolution graph and a citation graph with Wilmascope. (d) 3D visualization of the web traffic by Cox et Eick.



(a) Hyperbolic layout in 2D



(b) H3

Figure 2.7: Drawing graphs in Hyperbolic space. (a) Lamping and Rao draw graphs in hyperbolic layout. Their application is interactive: users drag the node of interest towards the center to better see its neighborhood. (b) Tamara Munzner generalized the hyperbolic layout to 3D.

graphs as shown in Figure 2.7. Lamping and Rao first presented the hyperbolic layout [LR96]. Tamara Munzner later presented an extension in 3D: H3 [Mun97, Mun98a]. Similarly to 3D representations, navigation techniques are necessary to explore the graph, complicating user actions but providing better results when coupled with 2D interfaces [RCMC00].

Alternative graph representations

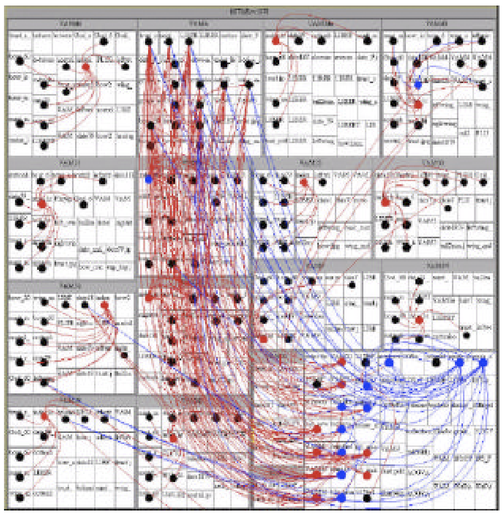
Due to the scalability problems of node-link diagrams and the complexity of controlling diagrams represented in other geometries, researchers have investigated alternative representations.

Visualizing graphs as treemaps. An alternative representation designed to fill the maximum space available when representing a tree is the Treemap [Shn92]. The principle is to represent trees by nested boxes: the space is recursively divided into boxes representing the different level of hierarchy. For example, a simple tree composed of a root having two children, each of them having four children is represented by a box occupying the total space available (root), divided into two boxes (children level1), each of these two boxes divided into four boxes (children level2).

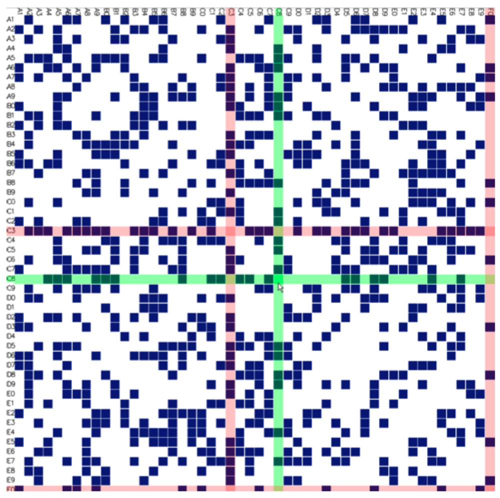
To generate this approach to graphs, the method is to compute a minimal spanning tree, to represent it as a treemap and to draw missing links over the representation [FWDP03]. Figure 2.8(a) shows an example of this representation, which is effective when the graph is not too dense (limiting the number of links drawn on top of the treemap). Zhao et al. investigated a combination of node-link diagrams and treemaps: elastic hierarchies [ZMC05] (Figure 2.8(c)). Users benefit from compact representations (treemaps) providing overviews of large sub-trees and detailed representations (node-link diagrams) to explore their point of interest. Further research is required to estimate the cognitive cost of switching representations and to extend this technique to graphs.

Visualizing graphs as matrices. Another alternative representation to node-link diagrams are visual adjacency matrices. An adjacency matrix is a table containing the graph vertices both in rows and columns. When two vertices are connected, the corresponding cell is marked. Early use of matrices to represent social networks has been done by Forsyth and Katz [FK46] in 1946. Matrices have also been used for representing various types of graphs: Becker et al. [BEW95] in 1995 visualize phone calls made between states and later Abello and Korn [AK02] in 2002, while MatrixBrowser uses visual matrices to explore file systems [ZKB02]. Van Ham [vH03] uses visual matrices to navigate in very large graphs (Java program with more than 25 000 Java classes) and explains that matrices were particularly interesting when user interest was on the connections more than on the nodes. Figure 2.8(b) shows a matrix representation.

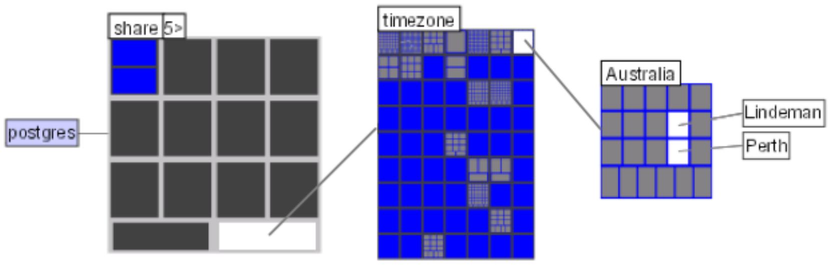
Experimental evidence [GFC05, KEC06] showed that matrices perform better than node-link diagrams for several low-level tasks such as finding if two nodes are connected or find the most connected node. They are a particularly good alternative when graphs become larger than a few hundred nodes and/or denser (than a tree) as they completely suppress node overlapping (nodes are linearly ordered in rows and in columns) and edge crossing (links are replaced by cells that do not overlap). However, a main issue is that they require to be ordered (row and column permuted) to be readable as shown early by Jacques Bertin [Ber83]. We detail this problem in Chapter 4. Another problem is that they use more space than node-link diagrams. Finally, performing the task of following paths in a matrix is tedious [GFC05].



(a) Treemap with links



(b) Matrix representation



(c) Elastic Hierarchy

Figure 2.8: Alternative representations of trees and graphs: (a) treemap with overlaying links, performing well for graphs with a tree structure and a few additional links; (b) matrix representation, performing well for dense graphs; and (c) elastic hierarchies for large trees relying on node-link diagrams and treemaps representations.

Another important trend of research focuses on *interaction techniques* used to navigate in large visualizations (solving the size problem) or helping to reduce the clutter occurring in areas of focus (solving the density problem).

2.5.2 Interaction techniques

In the field of information visualization, interaction techniques are tightly coupled to visual representations in order to interactively explore the data visualized. Interaction techniques can help organize and query a visual representation, adjust the amount of information displayed, select an adequate level of detail, and navigate through a large visual space. Interaction techniques are essential to apply Shneiderman's visual information-seeking mantra "Overview first, zoom and filter, details on demand" [Shn96].

Direct manipulation

In 1983, Shneiderman [Shn83] introduced a concept now central in information visualization and graphical interfaces design: direct manipulation. The principle is to directly manipulate the object of interest represented in the display. The idea behind direct manipulation is to provide users with rapid, incremental actions producing an immediate visual feedback, somewhat similar to what one would expect in the physical world, in order to offer a more intuitive interface to users.

Common direct manipulation interactions on a node-link diagram are: clicking on a node to display more detailed information, dragging a node to change its position, or dragging its corner to resize it. Direct manipulation can help disambiguating cluttered regions of a network by changing manually the positions of the nodes and therefore reduce edge crossing and node overlapping. While its overall efficiency is limited when exploring large cluttered graphs, direct manipulation is a step towards user-friendliness: avoiding the frustration of not being able to easily modify a visual representation.

Dynamic queries and incremental exploration

When dealing with large graphs, two techniques can help reduce the amount of information displayed: dynamic queries allowing users to control the amount of data displayed interactively; and incremental exploration supporting the display of portions of data currently analyzed, the remaining data being displayed as needed.

Dynamic queries. Dynamic queries [AWS92] rely on the idea that users should have a direct feedback while doing an action to query a dataset so the query can be adjusted dynamically by the user while doing the action. Therefore, dynamic queries can be used to filter a large graph in order to fit it in the display space. Consider a traditional slider controlling the number of nodes displayed on the screen. It is difficult to estimate in advance the right number of nodes to filter so that the node-link diagram would fit the screen. Using dynamic queries, the feedback is directly displayed on the screen while manipulating the slider, making it possible for the user to actually "see" when the node-link diagram fits the screen and stop his action on the slider or possibly reverse the interaction to get the optimal filtering value.

Incremental exploration. Incremental exploration becomes mandatory when dealing with graphs or networks so huge that a system cannot handle the full graph at any time. Similarly to Google maps [2] or Microsoft live search maps [3], two online systems displaying the maps of the entire planet at several level of details, in some cases the data is so large that it cannot be placed in memory all at once. To fix this problem, incremental exploration displays a small portion of the data and other parts when required (when the user explores a different region). The principle of incremental exploration is to place a window on the visual representation of the graph and to recompute its content upon user request. It can also be seen as a patchwork of frames over the graph to be displayed when analyzed by the user. A strategy to improve the time performance of loading data and displaying its content is to anticipate the users' actions by pre-loading all windows adjacent to the current one.

Most of the work on incremental exploration for graphs has been applied to node-link diagrams, improving layout computation time significantly (because reduced to the computation of a very small portion of the graph), and therefore providing interactive manipulation. Two solutions have been used to compute the layout of each portion of the graph: the simplest one is to draw each graph portion using the same layout algorithm [HEW98]; other strategies propose different layouts for each portion [Nor95, BW97], providing the user with dynamic controls of the parameters. Recently, incremental exploration has been used for matrix representations by James Abello and Frank Van Ham [AvH04].

Navigation techniques

When the visualization is larger than the display size, users need to navigate to explore the whole representation. Traditional techniques include *scrolling*, *panning* and *zooming*.

Scroll and pan The real world metaphor behind scroll and pan is that one looks at the visual space through a window or viewports. Scrolling is widely used in everyday application. By moving scrollbars on the side of the window, the visual representation moves horizontally or vertically (one could also imagine that the window is moving over the visual space). Directly dragging the visual space inside the window is called panning, similarly to the hand tool in Acrobat Reader. Both scroll and pan have the same effect: navigating through different portions of a larger space.

Zoom Another traditional technique to adjust the level of detail displayed is zooming. Two types of zoom exist: *geometric zooming* and *semantic zooming*. *Geometric zooming* simply scales the graphical elements of the representation. Zooming in while displaying a node-link diagram displays larger nodes and links, which takes more space, and therefore allows displaying a smaller amount of information but provides more graphical detail. Zooming out has the opposite effect: graphical elements become smaller and allow to fit larger node-link diagrams in the display. *Semantic zooming* takes into account the level of detail of a representation. For example, when zooming in a node-link diagram, nodes may not only become larger but also display additional information such as their labels. Similarly, when zooming out, one could imagine a group of nodes being merged in a super node (cluster) providing a higher-level representation of a graph.

[2] `http://maps.google.com`
[3] `http://maps.live.com`

Pan and zoom. A common navigation technique is the combination of pan and zoom [vWN03]. Presented in the Pad and Pad++ zoomable interface [PF93, BH94], this interaction technique allows to efficiently navigate though large spaces by reducing the level of zoom before panning for example, which is more effective as one can pan on much more space with the same gesture. Several pan and zoom techniques have been tuned, for example in 1D with OrthoZoom [AF06].

Alternative techniques for graph exploration. Many alternative techniques exist for navigating through large spaces. When exploring large graphs, connectivity tasks often require to use a navigation technique: for example, following an edge from its source to its target, navigate to all neighbours of a given vertex, or compare two connected vertices are common exploration tasks that can be greatly eased by appropriate navigation techniques. Relevant techniques include hopping [IGY06], in which the user is aware of off-screen elements and can quickly navigate to them, and split-scrolling [SG07], in which the user navigates while keeping the initial object in focus.

Focus+Context and space distortion techniques

While navigation techniques support the exploration of large visual spaces, they do not offer both context and detail. In this section, we present a range of Focus+Context and distortion techniques that center on a focus point while providing a view of the whole representation. For more details on distortion techniques and lenses, consult Carpendale's thesis [Car99] on elastic presentation space.

Disjoint context. A general solution to provide both details and context is the use of an overview, displayed in a separate window.

Distortion techniques The principle of distortion techniques is to deform a portion of the visual space in order to improve the readability of a focus point and/or compact the rest of the information to provide context. Compared to bifocal displays, the context is gradually degraded, providing a smooth transition between high-level and low-level of detail. Early examples are the perspective wall [MRC91a] and the document lens [RM93]. In these visualizations, the data analyzed is placed in focus at the center of the screen. The rest of the data is placed on either side, in 3D perspective in order to display more information in lesser details. A similar approach is RubberSheet [NBM$^+$06], in which two or more points of focus are displayed in detail, with intermediate portions of the data shrunk, *i.e.*, geometrically compacted in order to show the context. Other distortion techniques include edge bundle [Hol06], grouping links into bundles in order to better visualize high level connections in hierarchical graphs. Graph folding has also been presented by Carpendale in [CCF95], using the metaphor of a node-link diagram drawn on a piece of paper, with context areas placed inside 3D folds.

Lenses. The principle of lenses is to provide a small window over a portion of a visual representation to locally change its properties. Many types of lenses exist to perform locally and interactively geometric zooming, semantic zooming, distortion or other types of actions such as filtering. The most common lenses are fisheye techniques [CM01, Fur86, Fur06], which display interactively a geometric zoom of a portion of a visualization. Compared to disjoint context techniques, using separate windows; the strength of fisheye techniques is to gradually transform one level of details to another one. Blending two levels of detail contributes to maintain the constancy of the data representation. Similarly to semantic zooming, lenses can be used to provide detail on given portions of the representation, such as the labels of nodes with eccentric labels [FP99]. Distortion techniques can also be applied as lenses. For example, EdgeLens [WCG03] helps the user to disambiguate portions of the graph presenting a high number of edge crossings. Finally, other types of lenses include magic lenses [BSP$^+$93] that can apply several types of modification such as filtering or color change to a portion of the representation, and sigma lenses [Emm08], which define new transitions between focus and context regions.

The strength and weakness of Focus+Context and space distortion techniques is the deformation of the visual space, both detail and context are visible but it might mislead users because, in the representations exhibited, several levels of detail are blended into a single representation.

2.5.3 The evaluation challenge

Evaluating information visualization (InfoVis) systems is a challenge [Pla04]. The main reason is that InfoVis systems support exploratory data analysis: they aim at collecting insights (without a priori questions or models) on real datasets to solve real world problems. Plaisant [Pla04] explains that *"discovery happens over a long period of time, looking at the data from different perspectives, often answering questions you did not know you had"*. Moreover, what should be evaluated is the InfoVis systems´*"potential to increase the chances of discovery"*. Therefore they should ideally be evaluated *in situ*, which is a challenge when performing research, as *in situ* evaluation requires a robust system with a minimum of usability problems and the involvement of real users over a long period of time.

To refine the problem of InfoVis systems evaluation, Laskowski and Plaisant [LP05] define three levels of evaluation: the component level, the system level and the work environment level.

The component level is the evaluation of a technique or a specific visualization. It can be done through metrics or objective measures and controlled experiments taking place in a laboratory with a group of recruited users. One of the challenges of performing controlled experiments is the selection of low-level tasks reflecting real high-level ones (operationalization) and the selection of representative datasets. Another problem is that tasks in InfoVis often require an intellectual effort to understand the data and collect insights. This requires the participation of motivated users. This type of evaluation is currently the most common in the field of network visualization.

An example of metric evaluation is the comparison of algorithms to draw large graphs in term of computation time [HJ06]. These types of evaluations are often accompanied with qualitative comments from the users. Examples of controlled experiments include the comparison of two visual representations in terms of time performance and/or error rates such as matrix representations and node-link diagrams [GFC05, KEC06] or the comparison of two or more interaction techniques such as 1D scrolling [AF06].

The system level is the evaluation of the general system including multiple components and their interface. It is often compared to systems currently used amongst users. The most common approach is to use short scenarios or a set of tasks to perform the evaluation. A good example is the evaluation of different tools to analyze trees and graphs [GPB02, LPP+06]. The modern approach is to promote insight-based evaluation [PFG08, Nor06].

Task-based. The traditional approach is to define low-level tasks and use them in controlled experiments in a laboratory. Amar et al. [AES05] describe tasks for the analytic activity in information visualization:

“

Retrieve Value: Given a set of cases, find attributes of those cases.
· *Filter:* Given some conditions on attributes values, find data cases satisfying those conditions.
· *Compute Derived Value:* Given a set of data cases, compute an aggregate numeric representation of those data cases (e.g. average, median, and count).
· *Find Extremum:* Find data cases possessing an extreme value of an attribute over its range within the data set.
· *Sort:* Given a set of data cases, rank them according to some ordinal metric.
· *Determine Range:* Given a set of data cases and an attribute of interest, find the span of values within the set.
· *Characterize Distribution:* Given a set of data cases and a quantitative attribute of interest, characterize the distribution of that attributeś values over the set.
· *Find Anomalies:* Identify any anomalies within a given set of data cases with respect to a given relationship or expectation, e.g. statistical outliers.
· *Cluster:* Given a set of data cases, find clusters of similar attribute values.
· *Correlate:* Given a set of data cases and two attributes, determine useful relationships between the values of those attributes.
· *Scan:* Quickly review a set of items.
· *Set Operation:* Given multiple sets of items, perform set operations on them. For example, find the intersection of the set of nodes.

”

Given these tasks, InfoVis systems should be comparable in terms of error rate and time performance. However, evaluating high-level tasks such as “understanding a social network” is hard to decompose in several lower-level tasks. It is not clear if a system supporting well at all previous tasks performs also well for higher-level tasks.

Insight-based. A recent approach is to promote insight-based evaluation. Plaisant et al. [PFG08] advocate the role of benchmarks and contests to be able to compare InfoVis systems together. Each contest, held during InfoVis conferences, offers datasets with a set of associated tasks. InfoVis systems description and collected insights are reported by contestants and placed in a repository which aims at becoming a source of evaluation material for future designers wishing to evaluate their systems on similar problems.

In a more radical approach, North [Nor06] advocates the evaluation of InfoVis systems solely based on insights collection on benchmarks. North argues that tasks cannot match the complex nature of insights as they are:

“

. *Complex.* Insight is complex, involving all or large amounts of the given data in a synergistic way, not simply individual data values.

- *Deep.* Insight builds up over time, accumulating and building on itself to create depth. Insight often generates further questions and, hence, further insight.
- *Qualitative.* Insight is not exact, can be uncertain and subjective, and can have multiple levels of resolution.
- *Unexpected.* Insight is often unpredictable, serendipitous, and creative.
- *Relevant.* Insight is deeply embedded in the data domain, connecting the data to existing domain knowledge and giving it relevant meaning. It goes beyond dry data analysis, to relevant domain impact.

”

Therefore, North recommends an insight-based evaluation involving an open-ended protocol, a qualitative insight analysis, and an emphasis on domain relevance. This method requires running studies out of the laboratory and for long period of time, bridging the system level to the work environment level.

The work environment level deals with the adoption of the system in a real setting over a long period of use. This level of evaluation requires more qualitative and ethnographic methods such as Multidimensional In-depth Long-term Case Studies (MILCS) [SP06] proposed by Shneiderman and Plaisant.

Evaluation is a challenge for information visualization systems. A recent workshop, Beyond time and errors: novel EvaLuation techniques for Information Visualization (BELIV) [BPS07], is held biannually and presents recent work related to evaluation, such as new experimental protocols, approaches, metrics or task taxonomies.

In this section, we have presented different visual representations of graphs, a comprehensive overview of interaction techniques used in information visualization and given an overview of the challenge of evaluating information visualization systems. In the next section, we present related work in network visualization, which is becoming an important research area [CMS99]. We highlight the difference between network visualization and graph visualization or graph drawing: while graph drawing mainly deals with node-link diagram layout, network visualization primarily focuses on the data. Network visualization uses information visualization techniques to analyze both the structural and data properties of a network.

2.6 Network visualization

Network visualization is the topic of a very large number of publications in information visualization. In recent years, the information visualization conference saw 10% of its accepted papers dealing with this topic. As we cannot describe all existing systems, we only present an overview of the research area and the different philosophies of network visualization systems.

2.6.1 Interactive visualizations of large networks

Interactive visualization of networks is the topic of at least three PhD dissertations in information visualization: Tamara Munzner's in 2000 [Mun00], Frank van Ham's in 2005 [Ham05] and Bongshin Lee's in 2006 [Lee06].

Munzner's dissertation presents three systems aiming at *improving node-link diagrams visualization*. H3 provides a 3D hyperbolic layout able to handle larger node-link diagrams with fluid navigation. The two other systems, Planet Multicast, a geographical 3D network visualization, and Constellation, a semantic network visualization, also deal with layout of node-link diagrams.

Van Ham's dissertation deals with very large graphs (so huge in fact that it may be impossible to load the full graph in the computer memory). The main contribution to the field is the use of *alternative representations and interactive navigation through different levels of detail*. This work demonstrates the strength of matrix representations [vH03, AvH04] and shows how interactive navigation through different levels of detail can improve the scalability of node-link diagrams [**?**, vHvW04].

Lee's dissertation emphasizes the role of *simple therefore readable visualizations and their associated interaction techniques*. This work shows that analysis can be effectively done through simple data representations such as histograms or tables connected with each other and highly interactive. Winner of the InfoVis contest 2004 on coauthorship data, PaperLens [LCRB04] is a direct application of this philosophy. TaxonTree and TreePlus [LPP$^+$06] are two other contributions relying exclusively on navigation to explore large networks. The principle is to represent a very small portion of the network as a tree, (being a readable representation) and providing navigation to explore the full network.

These works are representative of the general direction of network visualization: from the improvement of node-link diagrams to the use of alternative representations and the increasing importance of interaction techniques for exploration. We now review a panel of existing systems along this line, focusing on how to apply them to social networks.

Node-link based systems. Only a few systems based on node-link diagrams can handle large networks while providing reasonable interactive manipulation. Most of these systems however focus on the structure of the network (the graph), and give less importance to data attributes. Tulip [Aub03] and JUNG [FOS$^+$05], respectively a C++ and a Java library offer a large range of layouts for node-link diagrams. They require programming skills but can handle large networks. More recently, Guess [Ada06], built upon JUNG, proposed an interactive system using both interaction and a simplified scripting language. A number of systems provide a graphical interface, more suitable for computer science novices, as well as a limited set of interactions (such as navigation and selection). H3 [Mun97, Mun98a] draws graphs in 3D hyperbolic space [Mun98b]. While it can display larger graphs than with a 2D hyperbolic layout, it requires training to master the technique of navigation and avoid getting lost in the data. NicheWorks [Wil98, Wil99] can handle very large graphs and provides a range of layouts for node-link diagrams. It is now integrated to Visual Insights, a commercial product.

These systems can handle large datasets and provide the representation of full networks. They are limited by node-link diagram weaknesses: node overlapping and edge crossing but provide interactions to adjust node positions and disambiguate edge crossing.

Navigation based systems A number of systems give up representing the whole network as a node-link diagram because of the challenge in term of readability, layout computation time and memory management. These systems represent a small portion of the network and provide interactive navigation to explore the full graph. TreePlus [LPP$^+$06] uses this strategy and favors readability of a small portion rather than an unreadable overview of the network. Another example is Vizster [HB05], providing an ego-centered visualization of a social network (Friendster's contact networks), *i.e.*, a network centered on an actor. This is particularly effective as it is designed for friendster's users, each interested in the small portion of network including them, not the full network.

Filter based systems A different strategy to explore a large network while avoiding node-link diagram readability issues is to provide filters as dynamic queries. SocialAction [PS06] is a social network visualization tool using this technique. Its strength is to provide a set of structural and statistical measures of social network analysis, using them interactively to filter the network and end up with a readable node-link diagram.

Aggregation based systems Another technique to reduce the data visualized is the aggregation. This aggregation can be done according to an existing hierarchy of the networks vertices such as MatrixZoom [AvH04, vH03] or MatrixBrowser [ZKB02]. A clustering algorithm can also be computed to aggregate vertices according to a given metric [**?**, ACJM03]. An approach focusing on the data rather than on the graph structure is PivotGraph [Wat06], aggregating the network according to its attributes and showing aggregated relations between the categories. For instance, given a friendship network, a starting point of the exploration could be only two super-nodes (male and female) and four weighted super-links (average number of friendship relations male-male, male-female, female-male, female-female). PivotGraph's approach introduces the next category of systems focusing on understanding the data rather than displaying the graph structure.

Data focused systems A number of recent systems focus on understanding the data carried by the graph structure. This approach is not new: Bertin, in his book "The semiology of graphics" [Ber83], proposes to organize node-link layout algorithms according to the data properties (see Figure 2.9) to represent rather than the traditional classification by algorithm type [DETT98, HMM00].

Semantic substrates [SA06] is a recent system illustrating this concept. The principle is to provide the node-link diagram layout according to the data of the network. For instance, if the network is composed of two categories of nodes, each having a temporal dimension, the nodes are layed out in two boxes representing the two categories, and ordered in each of the box according to the time dimension. Links are then drawn between connected nodes. Authors added dynamic queries to filter the relations and limit edge crossing.

A second type of system favoring data over structural properties is PaperLens [LCRB04] and its evolution NetLens [KPLB06]. The idea here is to completely give up the node-link representation and use instead simple data representations such as histograms. Using dynamic queries and linking visualization to each other (filtering one histogram is propagated to the list view for example) helps analysts understand their data. A video available at `http://www.cs.umd.edu/hcil/netlens/` demonstrates the effectiveness of NetLens.

Multiple representations and collaborative systems On the same philosophy, several systems such as ManyEyes [VWvH$^+$07] favor a combination of multiple simple and interactive representations to analyze a dataset. The current trend is also to increase the collaboration between analysts to analyze large datasets collaboratively [HVW07].

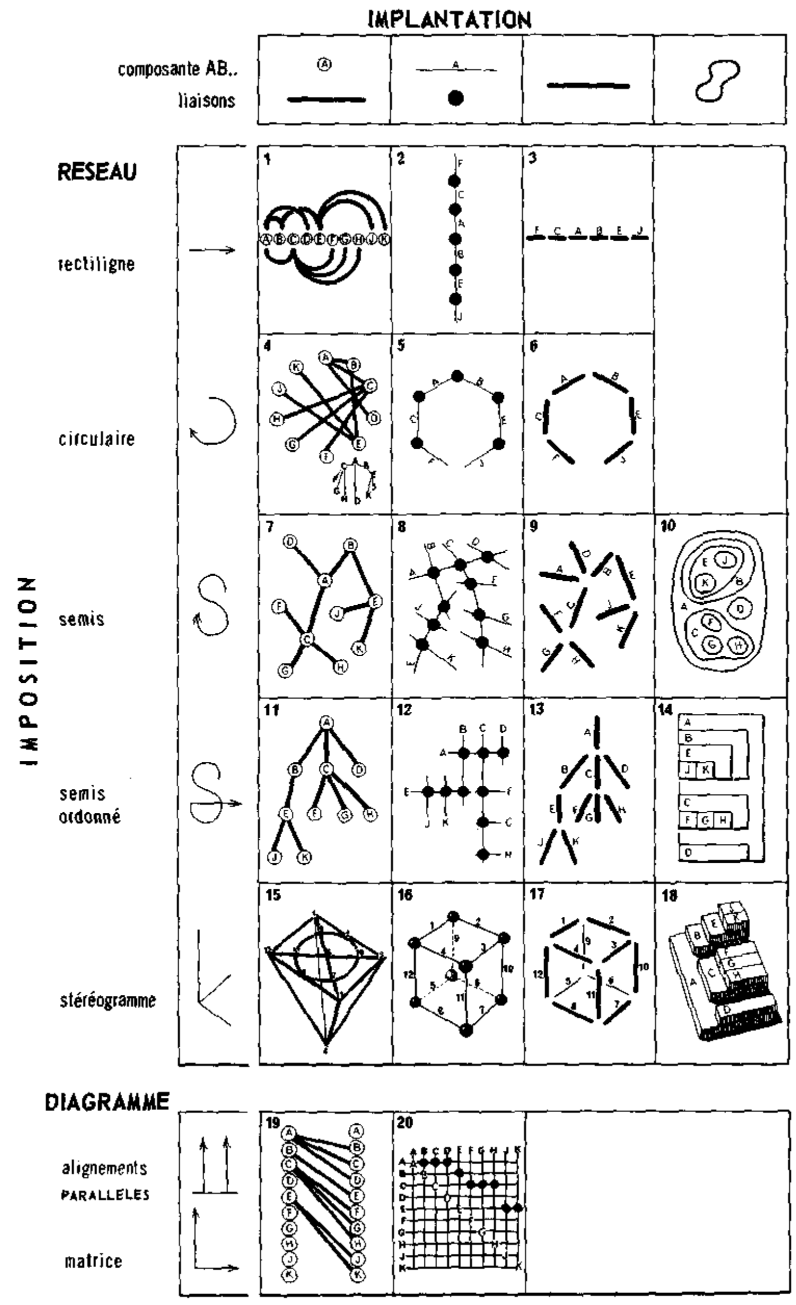


Figure 2.9: Classification of graph drawing algorithms according to the data properties, extracted from [Ber83].

2.6.2 Bridging the gap between SNA and network visualization

We have described most recent network visualization systems; we now review the most popular social network analysis systems among social scientists. Most of them provide both statistical and structural analyses, however very few support visual exploration. For a complete list, see the International Network of Social Network Analysis website [INS] and [HD05].

Statistical analysis. General tools to perform descriptive statistical analysis include R [R D06, Alb07] and spreadsheet calculators such as Microsoft Office Excel [Exc]. Excel is broadly used among social scientists to enter their data and perform general statistics including the creation of simple statistical graphics. Packages designed for social network analysis exist both for R: SNA-R [SNA] and for Excel: MatMan [Mat]. A system called StOCNET [HvD03] performs advanced statistical analysis of social networks. Through menus and numerical fields to enter various parameters, analysts specify the data, then the statistical model before performing the analysis. StOCNET is principally based on the work of Tom Snijders and offers five different models.

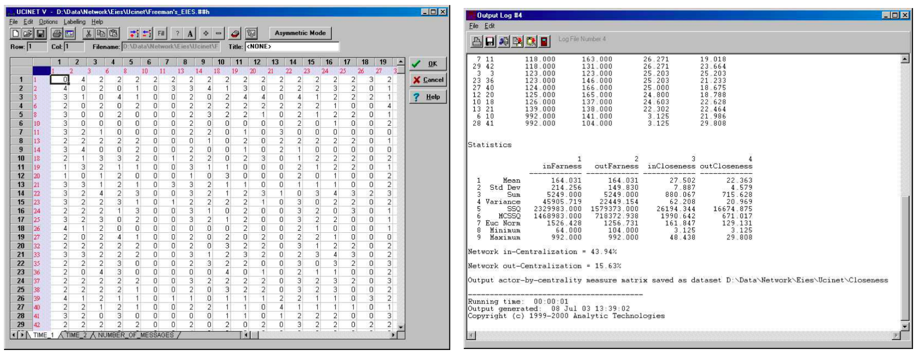
Combining statistical analysis, structural analysis, and visual representations. Most popular SNA systems combines statistical and structural analysis, and provide visual representations of the analyzed networks. These systems do not support visual exploration but rather offer visual representations for confirmation and communication purposes.

UCInet [BEF99] provides descriptive statistics, from general inferential statistic models (ANOVA and regression procedures) to more sophisticated ones (p models), as well as a large range of structural procedures to compute centrality, extract clusters and perform role and positional analysis. Analyses are performed through complex menus and sub-menus; results are logged in textual files. The strength of UCInet is to provide a spreadsheet editor as well as a number of basic data visualizations such as scatterplots. In addition, the NetDraw module is released with UCINet to visually represent networks. NetDraw essentially contains tools to perform layout and control the visual variables such as shape, color or size according to actorsáttributes or pre-computed groups.

MultiNet [Sea05b], product of Seary's PhD work [Sea05a] offers basic structural measures and a variety of statistical models. The originality of this software is that it is designed for contextual analysis, *i.e.* the analysis takes into account the actors and relationships attributes. Although not as complete as UCInet, the strength of MultiNet is to provide both textual and graphical outputs. Various representations are provided to represent statistical results. Node-link diagrams as well as visual adjacency matrices are available to represent networks. MultiNet provides a limited interaction to inspect details of these representations such as attribute values of actors or relationships.

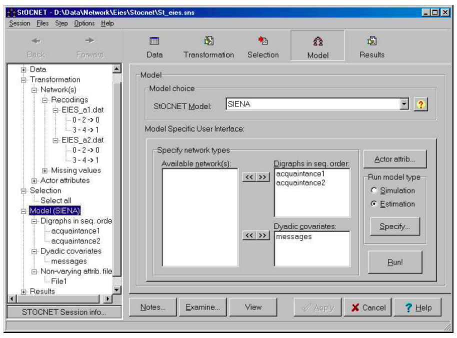
Pajek [dNMB05] is designed to handle large social networks. Pajek provides a very large number of menus to perform structural analysis and blockmodeling. Pajek provides a set of menus to create visual representations of social networks. However, the user is required to be an expert in social network analysis, graph theory and the Pajek system itself to navigate through multiple menus, finding the right combination of features in order to produce a readable visual representation. Node-link diagrams and visual adjacency matrices are provided. Pajek provides a limited interaction on node-link diagrams allowing nodes to be moved by drag and drop and displaying textual information on the node connectivity (its neighbours).

We described JUNG in the network visualization section, but this Java library also provides a range of descriptive statistics and structural measures. Its strength is to handle very large networks.

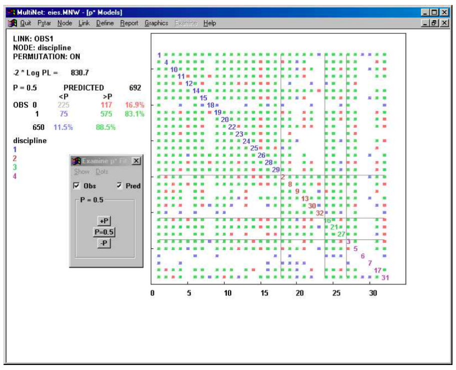


```
  7 11       118.000      163.000       26.271      19.018
 29 42       118.000      131.000       26.271      23.664
  3  3       123.000      123.000       25.203      25.203
 23 36       123.000      146.000       25.203      21.233
 27 40       124.000      166.000       25.000      18.675
 12 20       125.000      165.000       24.800      18.788
 10 18       126.000      137.000       24.603      22.628
 13 21       139.000      138.000       22.302      22.464
  6 10       992.000      141.000        3.125      21.986
 28 41       992.000      104.000        3.125      29.808


Statistics

                         1            2            3            4
                 inFarness   outFarness  inCloseness outCloseness
               ------------ ------------ ------------ ------------
  1      Mean      164.031      164.031       27.502       22.363
  2   Std Dev      214.256      149.830        7.887        4.579
  3       Sum     5249.000     5249.000      880.067      715.628
  4  Variance    45905.719    22449.154       62.208       20.969
  5       SSQ  2329983.000  1579373.000    26194.344    16674.875
  6     MCSSQ  1468983.000   718372.938     1990.642      671.017
  7  Euc Norm     1526.428     1256.731      161.847      129.131
  8   Minimum       64.000      104.000        3.125        3.125
  9   Maximum      992.000      992.000       48.438       29.808

Network in-Centralization = 43.94%

Network out-Centralization = 15.63%

Output actor-by-centrality measure matrix saved as dataset D:\Data\Network\Eies\Ucinet\Closeness

----------------------------------------
Running time:  00:00:01
Output generated:  08 Jul 03 13:39:02
Copyright (c) 1999-2000 Analytic Technologies
```

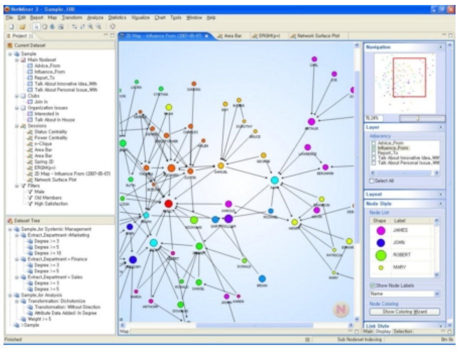
(a) UCINet



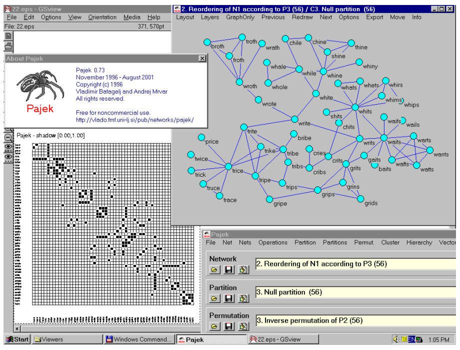
(b) Stocnet



(c) MultiNet



(d) NetMiner



(e) Pajek

Figure 2.10: Five interface of five SNA software. (a) UCINet 6.0 spreadsheet editor and example of statistical report. (b) Stocnet platform providing five statistical procedures. (c) MultiNet report for a p* model. (d) NetMiner 3 combining statistical, structural analysis and visualization. (e) Pajek interface with number of menus for complex procedure, graph and matrix representations.

However, JUNG is designed for analysts with programming skills.

These SNA systems provide a lot of measures and metrics through menus or programming languages. However, they require from their users a high level of expertise and do not support *exploratory data analysis* as users either run all implemented procedures or select one model given initial hypotheses. A number of systems attempt to provide a more exploratory process, analyzing visually the data (without model specification) and providing interaction to interpret analytical results.

Towards visual analysis A number of social network analysis systems aim at supporting visual analysis. The commercial system NetMiner combines statistical, structural and visual features. Similarly to UCInet and Pajek, this software is menu-driven: users are required to have a high-level of expertise to enter various parameters and combine the procedures. The strength of NetMiner is to present the results visually and to enable users to control the visualization parameters. However, the process to explore a visualization is tedious and requires users to memorize a number of actions previously done before seeing their actual results.

A second system is visone [BW04], a research software, still in development. It currently provides structural measures on actors (centrality) and groups of actors (cohesiveness), and visualizes networks as node-link diagrams. visone differs from the previous systems as it aims at "complementing each analytical procedure with tailored means of graphical interactions" [BW04], *i.e.*, support the visual exploration of networks. This approach is carried out by integrating graph layout algorithms tailored for each measure. However, in its current state, visone still requires the user to be familiar with graph drawing algorithms in order to select the most appropriate one and enter the right parameters.

The commercial tool InFlow and the simple module NetVis favor a limited number of interactive visualizations instead of providing a large panel of measures and algorithms. Similarly to VisOne, only structural measures are provided, however, both InFlow and NetVis include visual controls to manipulate algorithm parameters, producing a direct feedback on the visualization as well as direct interaction on the visual elements. This approach favors *exploratory data analysis* as users can interact with the visualization to discover insights.

Finally SocialAction [PS06], a recent system from the university of Maryland, explicitly aims at supporting exploratory data analysis. Based on node-link diagrams, SocialAction provides a set of structural measures that can be displayed on the diagram and act as filters. SocialAction suffers from node-link diagram weaknesses and cannot provide a readable overview of a large network. The strength of SocialAction is to provide interactive manipulation and support the exploration process by offering history and annotation features to the analysts.

An overlap is starting to appear between social network analysis systems and network visualization systems. However, in practice, very few social scientists use visual exploration tools. We believe that combining statistical, structural and visual analyses can dramatically improve the understanding of complex social networks. We also believe that information visualization can bridge the gap between researchers building complex models and social scientists analyzing the data: first by providing tools to compare models and real networks; then by supporting the exploration of large networks while limiting the knowledge needed to apply and parameterize the high number of algorithms and models.

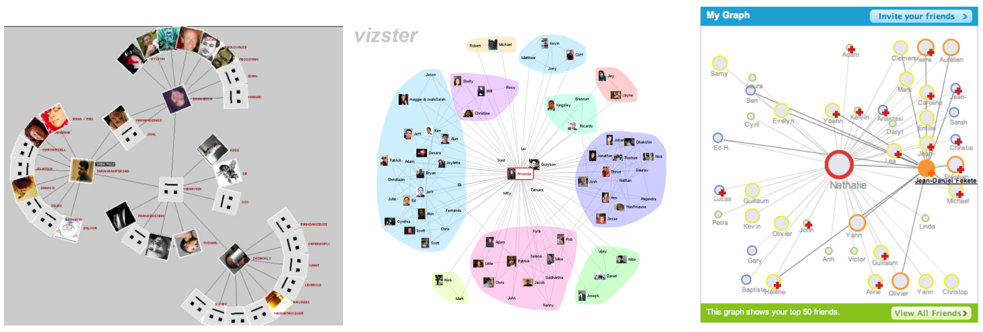
2.6.3 Towards non-expert users

An increasing number of systems are designed for people to *visualize their personal social networks*. These applications do not aim at task effectiveness or efficiency but rather focus on user interest and entertainment. They attract a wide audience by representing social data, raising interest among online communities (with questions such as "Who is friend with my friends?") and offering aesthetic or artistic visualizations, often interactive and fun-to-use. These applications are situated at the boundary of ambient information visualization [PS06], social information visualization and artistic information visualization. The term casual information visualization (casual InfoVis) [PSM07] recently emerged to qualify this class of applications.

In recent years, the popularity of internet technologies made social networks data more accessible. For example, social networks can be extracted from email conversations, forum discussions or chat logs and is even directly available through social networking websites such as FaceBook. Therefore, a sheer number of casual InfoVis applications have been created to directly show this data to its owner. VisualComplexity [VC] references more than 60 of them, while information aesthetics [Aes] exhibits the very recent ones. This section concludes our literature review and presents examples of existing applications for non-experts in order to demonstrate the increasing interest for social network visualization.

Researchers communities Social networks raise a general interest in the research community. As publication data are available on digital libraries and online databases, it is possible to extract social networks of scientists depicting collaborations (Who co-signed an article with who?) and scientific influences (Who has cited who?). These examples differ from non-experts applications, as their goal is not only to entertain the user but also to answer some particular questions about the whole network: what are the successful patterns of collaboration? Who are the known researchers and what is their strategy? ReferralWeb [KSS97] is an early example of visualizing collaboration among researchers. Extracting scientific networks from the web, ReferralWeb provides "chains" of researchers on a main topic. The user selects topics or researchers and sees how they are connected. Other examples provide pictures of an entire community [Ex] and eventually their evolution over time [EHK$^+$03]. Researchers such as MEJ Newman study social networks of entire fields of research, attempting to identify their structure and to compare them to each other [Newar, New04a]. In this perspective, scientists become analysts performing scientometrics (the science of analyzing science). More than observing their social networks, they become experts and aim at *analyzing* them [Kre94, MP96].

Friends communities Most of the applications developed for online communities members are ego-centered visualizations (Figure 2.11). For example, Vizster [HB05] provides an interactive visualization to explore one's Friendster contacts. The user starts to visualize his contacts and then interact to find groups of contacts connected to each other or to explore his friends' contacts. Another example is FlickrGraph [FG], showing contacts of the Flickr community, aimed at exchanging and commenting photos. The principle is the same for both applications, a *fun-to-watch interactive visualization* with contacts represented by their name and possibly pictures connected by links. Each link acts as a spring. When one contact is moved, the rest of the visualization accompanies it with a smooth motion. The interaction is intuitive as users directly click and drag the visual elements representing their contacts. Several other applications appeared to represent online communities centered on a single user. For example, CommentFlow [CF] is built on a similar idea for the users of MySpace. Finally, the popular FaceBook, reaching 46 million members in summer 2007, inspired several visualization tools such as FaceBook TouchGraph [TG], Facebook friends wheel [Fle] and the interactive friends graph [TSS].



(a) FlickrGraph (b) Vizster (c) Interactive Friends Graph

Figure 2.11: Online communities visualized for non-experts.

Digital identity and group activity Many other casual information visualization systems attract non-expert users: email visualization, or group and forum discussions. For example, Social Network Fragments [BP03, VBN^{+}04] attempts to show one's digital identity through the history of email conversations. It aims at showing people how they segment their email communication in several facets, for example a facet for communicating with friends, one with family and one with colleagues. This visualization is dynamic and shows a picture of the email conversations per slice of time. For each slice, the name of the email contacts are colored and positioned according to the facet they are a part of. The idea is to help understand the evolution of the communication. Another example is the identification of electronic communities and group discussion through associations of keywords. The Visual Who [Don95] project represents the evolution of community activities using colored keywords in motion forming *visual patterns*.

Participatory design

CHAPTER 3

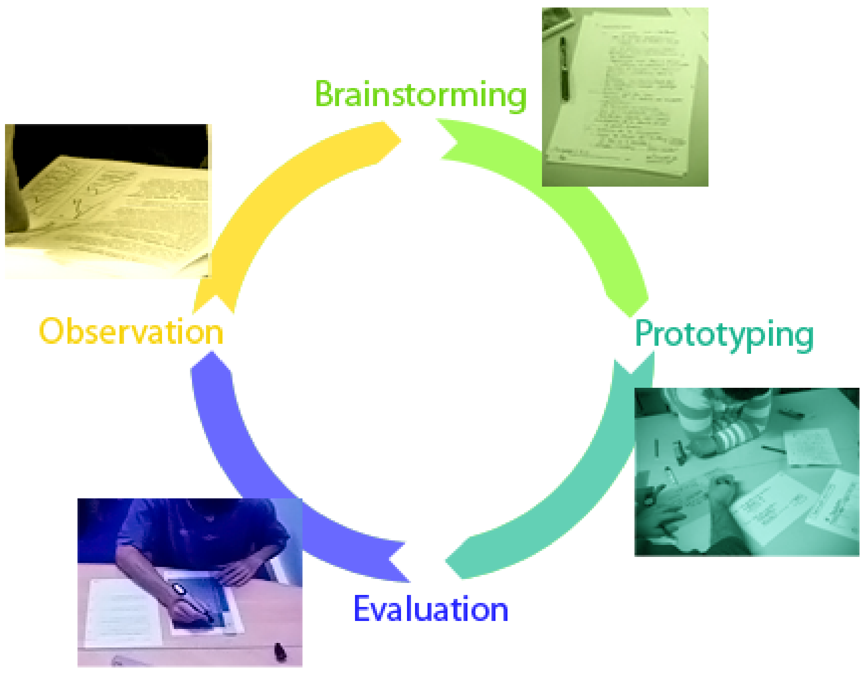


Figure 3.1: Participatory design process.

This chapter is a preamble to our four major contributions and presents our methodology as well as the design directions we extracted from our work with users. Chapter 8 complements this chapter as it presents user ecological validations of the techniques we designed and implemented.

3.1 Approach

In this thesis, we adopted user-centered and participatory design methodologies [MK93]. The goal of these methodologies is to focus on users instead of tasks. We chose a user-centered approach for two main reasons: first we wished to identify real issues faced by analysts so as to provide them with solutions they will adopt; second, we needed the expertise of social scientists to understand and decompose their complex analysis processes into manageable tasks and requirements on which to base the design of our system.

There is no consensus on the exact boundary where a user-centered approach ends and a participatory design approach begins. Both methodologies follow the same philosophy: involving users into the design process (Figure 3.1) as early as possible, which has been shown to speed up the convergence of the cycle. The difference lies in the amount of involvement of users and their role in the design: in participatory design, users are active members of the design team whereas in user-centered design users are less involved and act more as consultants. Metaphorically, in participatory design, users take a step into the designersẃorld whereas in user-centered design, designers take a step into the usersẃorld.

We consider our approach as an intermediate practice. Because we could not provide a full working system, we were careful on the amount of investment (time) we asked from our users. Social scientists actively participated to the creative process and at various stages of the design, giving informal feedback and performing evaluations. However, during our sessions with users, we gave priority to novel approaches for visualizing and interacting with social networks while avoiding discussion on ergonomic and minor usability issues.

3.2 Design process

Figure 3.1 depicts our design process. We went through several iteration of this process with ten social scientists: five sociologists from the INRIA (French National Institute of Research in Computer Science and Control), France Telecom and EHESS (School of High Studies in Social Sciences); four historians from CNRS (National Center of Scientific Research), BnF (French National Library) and the French Archives as well as one senior analyst, director of a decision support firm. All ten were not involved at the same level concurrently during the design, but they all contributed to this project. We now describe each stage of this process, attempting to give an overview of our methodology.

Observation The goal of this stage is to observe social scientists *in situ*, with their own data and tools. This observation leads to the identification of issues faced during the exploration and help understand what is the activity of our users. During this stage, we mainly used *informal interviews*.

To initiate the collaboration, we participated to a *workshop* [1] and a *seminar* [2], in which several social scientists presented and discussed their work, findings and issues. At the end of these two events, we had the opportunity to interview several researchers. Interviews were oriented on five points but they were informal and encouraged interviewees to explain, elaborate or even show tools and results to the person conducting the interview.

1. *Datasets.* What are your datasets? How did you collect it?
2. *Tasks.* What are your questions? What steps do you perform when analyzing a network?
3. *Findings.* What do you consider findings? Can you show an example of findings or results of a previous analysis?
4. *Tools.* What tools do you use? How do you proceed with a new dataset? Can you perform a demonstration?
5. *Needs and wishes.* Which features do you use/not use/like/dislike/would like?

Brainstorming *This stage aimed at identifying real issues and needs as well as suggesting solutions and desired features. The goal of the brainstorming was to generate novel ideas in small groups, discussing ideas without coming to their feasibility.*

We organized two participatory design workshops [3]. Divided in groups of 3 or 4, our users first identified very concrete problems they were facing and suggested solutions. We asked them to think out of the box, to suggest ambitious solutions to their most difficult problems. In order

[1]Workshop held during Graph Drawing, 2005 september 12, Limerick, Ireland
[2]Seminar INED-EHESS, 2005 december 8, INED, Paris
[3]Two sessions ran the 6th of january 2006 and 12th of june 2007, at INRIA, Orsay

to help them “thinking out of the box”, we preceded the brainstorming with a presentation of various rather “radical” visualizations and interaction techniques, from graph representation to input device, including a panel of software user interfaces.

Prototyping *At this stage, users and designers selected a subset of ideas from the brainstorming, discussed and video prototyped them. Users created initial prototypes with pen, paper and video to illustrate the data representation (or interface) and the matching interaction before its actual implementation.*

These video prototypes were created during both participatory design workshops. The actual implementation was done after a careful analysis of both observations and video prototypes; with appropriate design decisions made through the implementation process.

Evaluation *This final stage is the collection of user comments and feedback, generally leading to the observation of new issues and problems to solve.*

In our case, evaluation methods varied from informal feedback to controlled experiments evaluating user performance of a particular task and included a case study on a higher-level use of the system.

. . . and back *The process we described is a cycle: from observations performed during the evaluation came new issues and new solutions to prototype and evaluate.*

We repeated this process 6 times leading successively to novel matrix reordering techniques and how to assess their quality (Chapter 4), MatrixExplorer (Chapter 5), MatLink (Chapter 6), NodeTrix (Chapter 7), Mélange (Chapter 6) and node duplications (Chapter 7). For each of these prototypes, presented in the next four chapters of this dissertation, we detail our observations and the problem to be solved followed by our solution and its evaluation.

3.3 Participatory design outcomes

Most of our interviews were informal and very specific to particular tasks and datasets of our users. These interviews provided means to initiate the dialog with our users rather than acting as a mechanism for collecting design requirements. The main reason was that finding common observations and extracting design goals among all interviews was a challenge. Therefore we decided to organize a participatory design workshop with several of our users on a full day. In this section, we first give a glampse of our participant background and present the participatory design workshop and its outcomes. Linking these outcomes to previous observations made during the interviews, we ended up with a set of design goals for visual exploration systems.

3.3.1 Participant background and data

The background of our participants was very diverse. While being all social sciences experts, they all differed in their research questions or approaches of data analysis. They also had very heterogeneous types of data.

For example, one historian dealt with a large network constisting on co-occurrence of persons in documents. While, she had a very deep-knowledge of her data, her approach was very exploratory. She was interested in having an overview of her data and answering a large range of questions such as is this person appearing in other communities? who has influences over who? are not these two entities a same person? etc. On the contrary, the professional analyst dealt with data on irrigation in regions of India and polical influences in theses regions and had a very specific question : how can we improve agriculture in India? Finally, some researchers had a blend of very specfic questions, attempting to support a given theory and a more exploratory approach. For example, scientists working with the french national archives of the administration aimed at identifying the organization of the french administration before and after the revolution, attempting to understand its evolution.

3.3.2 Workshop organization

The workshop was organized in four stages:

1. A presentation by our team of novel visual systems and interaction techniques applicable to graphs and social networks. The aim was to give social scientists a broad view of cutting edge research, without leading them towards a specific technique. Thus they gain enough knowledge on information visualization and human computer interaction to boost their creativity. Presentation available at `http://www.aviz.fr/~nhenry/SocNetWS`.
2. A brainstorming where social scientists and designers generated as many ideas as possible. We focused the brainstorming on 2 questions: how would you like to explore your network? [visualize/explore] how would you like to create and edit your network? [create/edit]
3. A video prototyping stage where a selection of ideas were recorded. Divided in small groups, people discussed and created their prototypes using pen and paper; they recorded their ideas, showing interactions, interfaces and adding oral comments to describe the future system in action.
4. A final stage in which video prototypes were shown to all groups for comments and feedback before closing the workshop.

3.3.3 Workshop outcomes

In this section, we first present an example of ideas generated during the brainstorming sessions (step 2). Then, we detailed each selected idea prototyped by our users using pen, paper and video camera (step 3).

Brainstorming outcomes A large number of points were raised during the brainstorming, here is a small example of the ideas generated for one of the groups on the question "how would you like to explore your network?":

On representations:
- present different representations of networks;
- present different layouts for node-link representations;
- present different orders for matrix representations;
- for multivariate networks: one visualization per type of links;
- for evolving networks: showing an animation of the evolution (stoppable and replayable);

On interaction techniques:
- provide keyboard search to find a given actor;
- provide a full list of actors and provide various reordering to find a given actor;
- offer comparison tool (and stats) between two matrices;
- superpose two representations (same network, different relations or same network, different years) to observe common points and differences;
- show an indication of important relations (tag feature and computational measures);
- show an indication of the information density;
- provide a zoom (lens);
- provide a zoom in different places to compare them (multiple foci points);

On exploration history:
- show an history with all actions;
- allow the annotation of the history (with the option to plan ahead);
- choose the steps of the history (taking snapshots) and navigate to previous states by clicking on thumbnails;
- allow recording of a movie of the exploration (to communicate findings and the context of their discovery).

This example shows a sample of the variety of the ideas proposed by users. It demonstrates that our users integrated a number of information visualization techniques presented in the morning (such as zoom and lenses) and were able to generate many novel ideas to solve their problems. Each idea was briefly explained within the group. During this stage, we collected a large number of insights we were able to link afterwards to previous interviews. The main outcome of this phase was the emergence of issues shared between social scientists, which were difficult to identify during the observation phase.

Video prototypes After the idea generation phase, all groups voted to select the most important ones and prototyped them using pen, paper and video camera. We present here the results of the video prototyping. Videos (commented in French) are available at `http://www.aviz.fr/~nhenry/SocNetWS`.

Creating a network and/or its visualization [create/edit] Three ideas emerged to create a network from loading data and editing it interactively. The first one concerned the introduction of different types of links in the data through interactions on the visualization. The idea is to load the unconnected vertices as a node(-link) diagram, using a circular layout, then create the different categories of relations (appearing as checkboxes in the visualization) and finally create the actual relations in the network by selecting a source node and dragging the selected link towards a target one. The second one concerned the creation of a node-link diagram from an incidence table (containing edge information starting with source and target vertices, then edges attributes). The idea is that only selected elements in the table (a set of edges and a subset of their attributes) are visualized. A variation of the latter is the third idea: creating a set of nodes by dragging selected values from a table into a visualization panel, the links being displayed by dragging the attributes or types of the links from the table into the visualization panel.

Attributing visual variables [visualize/edit] The first idea that emerged and thrilled the participants was the potential of visualizing information and being able to assign dynamically visual variables. For example, displaying data attributes as well as computed analysis measures through shape or color of the visualization. They particularly liked the idea of direct manipulation, interacting on elements directly to change their visual properties and position. The need to edit the network while visualizing it was also raised, for example being able to edit the data while assigning visual variables (say, by using the right click directly on elements).

Clustering and aggregating portions of a network [visualize/edit] Three ideas are related to the topic of clustering and aggregation of portions of a network. The first one deals with identifying and tagging subgraphs interactively, adding this information to the network data. The idea is to identify clusters by selecting a group of elements with a lasso selection for example (drawing a visual feedback around the set of vertices and edges) and then entering a representative name. The second idea shows the importance of identifying vertices connecting two or more clusters and being able to extract them from the clusters easily, by simple drag and drop for example. Finally, the third idea is the aggregation of the clusters into a super-node, showing super-edges between them, which could help in the exploration of large networks. To mark the number of connection between two super-nodes, the width or shape of the link is used. Figure 3.2 shows pictures extracted from the video brainstorming illustrating the third idea.

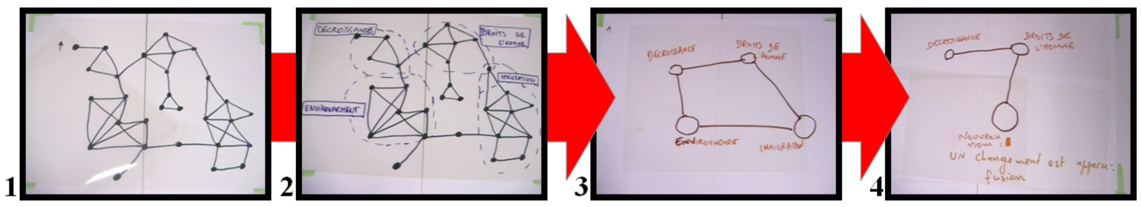


Figure 3.2: Video prototype illustrating the aggregation of portions of a network.
1. Network displayed as a node-link diagram. **2.** Identification of communities (groups of actors and relations). A dotted line represent the community, a label is entered by the analyst. Adding or removing actors from the community should be possible by simple interaction. **3.** Aggregation of each community into a super-node. Relations between communities are also aggregated into super-links. **4.** Super-nodes can also be merged and labeled with a new name.

Exploring a multivariate network [explore] One of the groups identified matrices as good representations to explore multivariate networks. The idea is to create several matrices, each displaying each a different type of link and possibly different types of vertices in rows and columns. For example, from a friendship network, one could compare three matrices representing friendship occurring within females, males or within different genders (by simply filtering rows and columns according to given attributes). Similarly comparing two matrices displaying two types of links could help understand common points and differences between friendship and dislike relations for example.

Supporting the analysis process [explore] To support the analysis process itself, social scientists imagined a customized history that would support the analysis over long periods of time, serving both as memory of what has been done and as a planner for what needs to be done. The principle is to allow analysis to save states of the system at key points of their exploration. These "snapshots" would result in time stamped thumbnails of the visualizations so analysts could go back to previous steps or simply have an overview of the analysis. All were convinced that automatic saving of the system would lead to too numerous states, difficult to understand later. All were thrilled by an annotation feature that could help them plan ahead analysis to be done or visualizations to be tried. Annotation would also help them comment on their various findings or hypotheses. Several persons suggested this history could serve as reference for other analyses (applying a previous successful scheme of exploration) and also could help plan how much time is needed to perform a given analysis. During the final discussion, several persons suggested this history could be used to communicate findings, by navigating to previous states and explaining the context of discovery. Figure 3.3 shows images extracted from the video prototype to illustrate this history-planner feature.

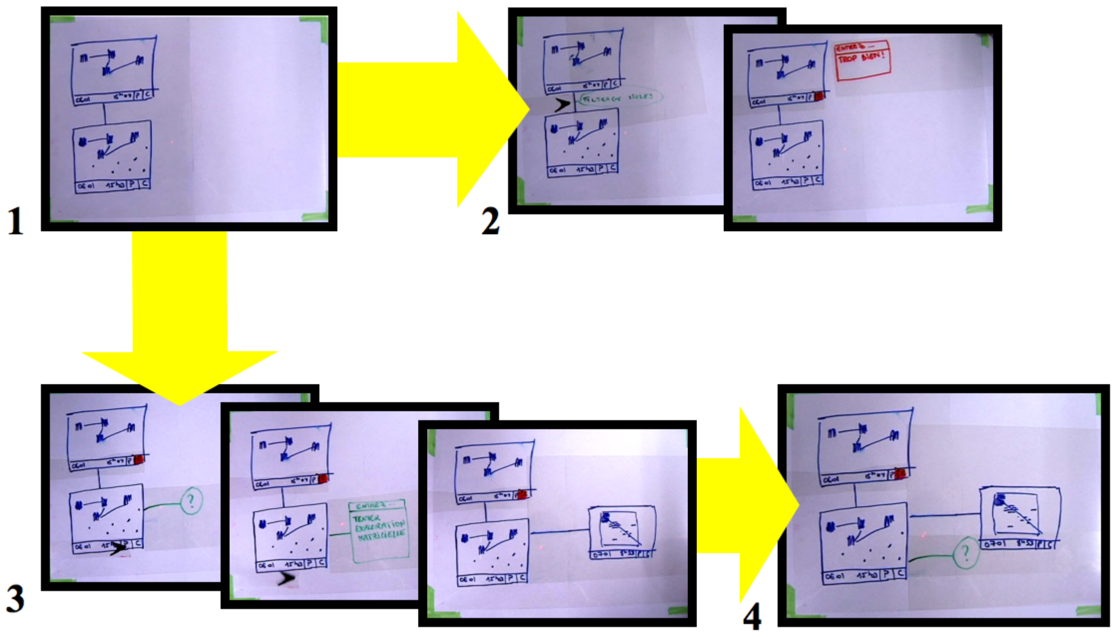


Figure 3.3: Video prototype illustrating how to annotate and navigate through history in order to plan the analysis ahead or replay past analyses.
1. Time stamped snapshots of the system are taken by the analyst. A click on any thumbnail restores the system to this state. **2.** Comments can be added both to the thumbnails and their transitions. **3.** The analyst can plan ahead if (s)he needs to stop the current session or as a reminder if there are two paths of exploration to perform. Once the exploration is done, this planning step can be replaced by the corresponding snapshot of the system. **4.** After an interruption in the analysis, both history and planning features reduce the effort required to go on with the analysis.

Comparing two or more networks [explore] An idea emerged to compare two networks using matrix representations. The focus was on networks very likely to have many actors in common, such as the comparison of two types of relations with the same set of actors (multivariate network) or a network at different points of time. The idea is to overlay one matrix representation on another one, coloring common elements in one specific color (green for example) while keeping different elements in the color of each matrix (black and red for example). Social scientists also point out the importance of adding basic statistics such as the percentage of common elements. Figure 3.4 illustrated this idea by showing extracted pictures from the video. The comparison of completely different network raises other challenges, as it is a matter of comparing the structure (connections patterns) rather than the actual elements. While no video was recorded for this particular problem, it was suggested this could be done by visualizing two matrices side by side (therefore comparing the connections patterns inside the matrices).

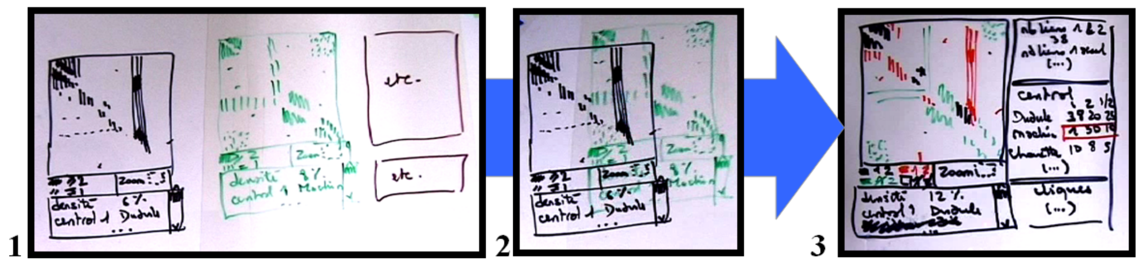


Figure 3.4: Video prototype illustrating the comparison of two or more networks.
1. Two networks represented by two matrix representations. In this case, each matrix has a different color shade. **2.** Comparing two or more matrices is simply done by drag and dropping one matrix over a second one. **3.** The common elements are displayed in a combination of colors (here it is red), while the differences are colored the same color of the corresponding original matrix (here one is black and the other green). Statistics sum up the percentage of difference.

3.3.4 Design goals derived from the outcomes

The participatory design workshop really helped us initiating a dialog with our users on concrete cases but they were far less specific than during interviews. Grouping several social scientists together during the workshop helped us understand common issues and provided us with clues on their exploration process and needs. Coupled with the data gathered during the informal interviews, we extracted a number of common goals for both visualization and interaction. We classify them intro three classes: visualization, interaction and advanced features.

Visualization

G1 - Multiple representations Participants used both node-link diagrams and matrix-based representations. Although node-link diagrams are familiar and effective to communicate (for relatively small or filtered-clustered graphs), they acknowledged that matrix-based representations were fast to display and easier to manipulate for large or dense graphs.

G2 - Connected components Real graphs contain several connected components. Handling several connected components and being able to navigate within each of them, or compare them, is necessary for a system dealing with real datasets.

G3 - Overview Overview is a challenge for large graphs. However, overviews are crucial for the exploratory process. They are used both as starting points for the exploration and as stable maps during the navigation. Overviews help users to build a mental map of their network. Participants asked for an overview of each visualization at all stages of the exploration.

G4 - Dataset general information A set of measures that are always useful for an analysis was identified: type of graph, number of vertices, edges and their respective labels, types and number of attributes. All these measures should be displayed initially and be easy to access at any moment.

G5 - Data attributes Taking attributes into account makes the difference between graph drawing and information visualization. Participants were not interested in displaying a unique graph; they wanted to build several representations according to the different attributes of the edges and vertices. The structure of the graph may be different depending on the chosen attributes. Comparing these structures, understanding why they are similar or how they are different was their main concern. They need an information visualization system that helps them to choose visual variables for each attribute and create multiple views of their dataset. Consulting details for each vertex or edge was also a primary interest and therefore, details should always be visible or quickly accessible.

G6 - Analytical information Visualizing and interacting with the data does not exclude statistical and analytical features. Participants wanted to get at least the basic network analysis data, such as number of actors, relations, density, diameter, five most connected actors, degree distribution. They also asked for computed attributes on demand, such as centrality measures.

Interaction

G7 - Interaction vs. parameter tuning Most of the participants were familiar with graph drawing and clustering algorithms. However, their understanding was limited and they were unable to fine tune the parameters for these algorithms. Thus, they asked for more interaction with the graph and less, or predefined, parameters. They also noticed that manipulating and reorganizing the network interactively facilitated understanding and memorization.

G8 - Layout Computing the layout of a graph is necessary to find insights. It means computing coordinates for vertices in node-link diagrams or computing a permutation of rows and columns for matrices. In both cases, participants acknowledged that several layouts were required to understand a graph. Participants asked for both automatic and manual (interactive) solutions. Automatic algorithms rarely provide satisfactory results but save users time and effort. Interaction let the user improve the resulting layout. Moreover, participants quoted that not being able to interact to drag a node or move a row or columns of a matrix was frustrating.

G9 - Filtering For large networks, filtering is a requirement that allows fine analysis of the network and its sub-parts. However, filtering data may confuse users and lead them astray, especially if it alters the visible data structure. Therefore, participants asked that the system remind them that filtered data still exists (through statistics or specific representation of filtered data for example).

G10 - Clusters In social networks analysis, cluster detection or community detection is very important and required for exploration. A large panel of automatic methods exists to cluster networks. Users may also detect them manually. As for the graph layout, participants wanted to combine interaction with automatic algorithms. They expressed the need to handle several clusterings for a network and to annotate their clusters (giving each a name and a description).

G11 - Outliers Social researchers are interested in outliers. For example, they try to understand why an actor has a different connection pattern or why two actors do not communicate within the same cluster. A system should not only filter outliers as dataset noise but also support their discovery.

G12 - Consensus Participants deal most of the time with multivariate data, i.e., several kinds of relationships and attributes for actors. Thus they compute several clusterings depending on the attributes chosen and the visual representations. Participants asked for tools to identify a consensus among the clusters; or at least showing the main differences between the results.

G13 - Aggregation Participants agreed that aggregating networks based on the clusters or communities was a useful feature to reduce the network size and help present results. However, they were concerned by the loss of information when dealing with aggregated networks and insisted in being able to get back to the full data when exploring the aggregated networks.

A step further

G14 - Multiple datasets Participants expressed the wish for a system handling multiple datasets, supporting both their comparison and the application of the recorded schemes of analyses.

G15 - Time-varying data Analyzing the evolution of a network or its communities is the next step towards the analysis of social networks. This is made possible by the novel ways of collecting data and is becoming necessary for analysis.

G16 - Supporting the exploration process Participants all agreed that history acting as a navigation tool through the previous steps of the analysis, as well as a notebook for annotating findings and planning the analysis ahead would be precious features to support the exploration process.

3.3.5 Design decisions

During our literature review, we identified that matrix representations had a vast potential to explore large networks. The participatory design session confirmed that matrices were used among social scientists, even if they were most of the time handwritten and only used to represent very small portions of the network. Even though node-link diagrams have a number of weaknesses for large and dense networks, we decided to use them in conjunction with matrices, as they remain widely used and the main representation for communicating findings or illustrating theories.

Concerning the appropriate interaction and interface to explore social networks, we decided to have an information visualization approach, favoring graphical interfaces, direct manipulation of objects and dynamic queries over menu-driven or programming interfaces. As matrices can handle large and dense networks, we formed the hypothesis that they would be used for exploration, while node-link diagrams would be effective as communication means for filtered or aggregated networks. Therefore, we decided to primarily work on interaction issues with matrix representations.

Finally, social scientists expressed their needs for several perspectives on a same dataset in order to find a consensus about the data. As a first step, we focused on the matrices and node-link diagrams, attempting to provide our users with several ways of ordering rows and columns (matrices) and laying out nodes (node-link diagrams).

In the following chapters, we present MatrixExplorer, a system that is a direct outcome of our participatory design observations and decisions. The intermediate chapter 4 is an annotated bibliography of matrix reordering techniques and an attempt to understand how they affect matrix readability.

Matrix Reordering: making matrix representations usable

CHAPTER 4

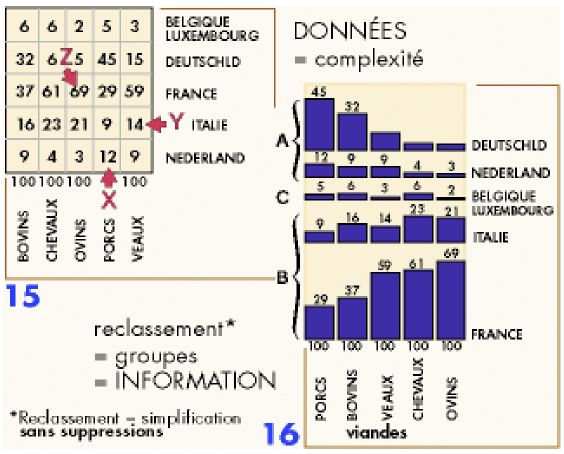


Figure 4.1: Meat production in 5 countries.

This chapter presents an annotated bibliography of matrix reordering techniques as well as our attempts to assess their quality.

4.1 Research problem

Bertin [Ber83] showed that permuting rows and columns of a matrix can dramatically improve its understanding: randomly ordered matrices are readable but reordered, they become usable. Figure 4.1 demonstrates this idea on a very small matrix containing only five rows and five columns. By transforming the original table of numbers by visual shapes (varying in size according to their values), and by reordering rows and columns "nicely", trends in the data emerge. An analyst easily sees that France has the largest production overall, Belgium the smallest and that there are three different profiles of meat production (A, B and C on the picture). To go a step further, Belgium has a rather neutral profile of production, which means that if a law is to be voted to limit the production of a type of meat highly produced by either group A or C, Belgium is likely to be the key country to convince, the other being driven by their own interests.

This simple example shows that rows and column reordering are essential to a better understanding of matrices. However, we face two main research problems:

- What are the existing reordering techniques and their different properties?
- How can we assess their quality?

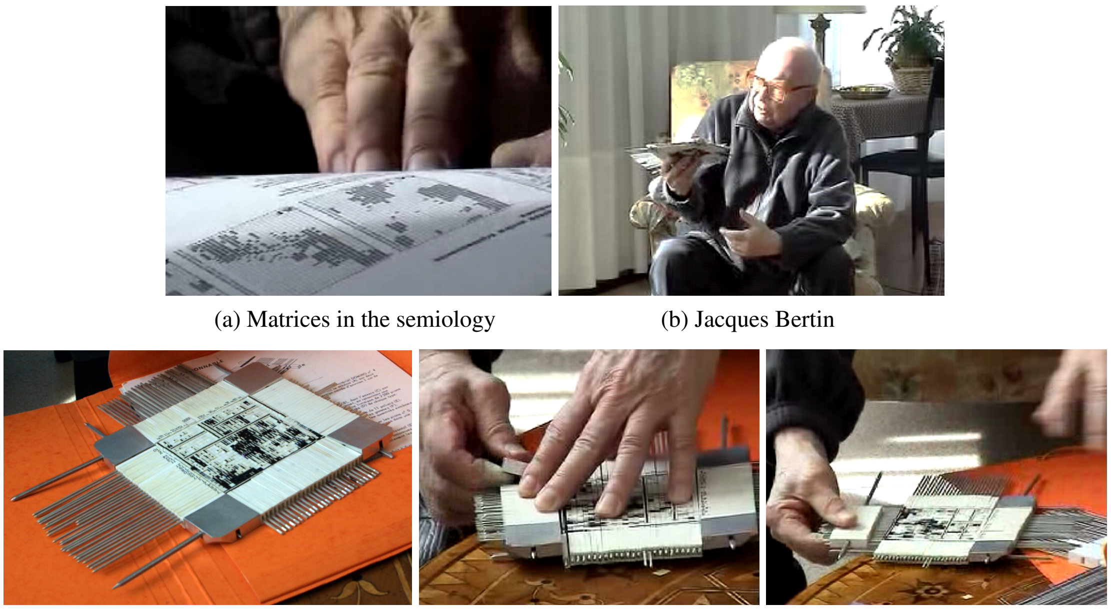
Performing a survey on reordering techniques is a real challenge as many problems are related to the permutation of rows and columns, having applications in many different domains for many different purposes. In the next section we will attempt to give a comprehensive overview of the categories of methods. Assessing ordering quality is similar to assessing graph layout quality: no clear critera have been defined and, to our knowledge, studying how an order improves the readability of a matrix has never been done.

4.2 Reordering techniques

Initial surveys have been presented on graph layout algorithms [DPS02] and matrix reordering [CHHGWJ^{+}04, MML07]. However, they are incomplete because reordering algorithms are disseminated through several research fields such as information visualization, biology, social networks, economy, linear algebra or even construction engineering (reordering of Design Structure Matrices); and therefore designated by many different names: *matrix reordering, table reordering, vertex linearization, permutation, seriation, ordination and also optimal linear ordering or minimum linear arrangement*. In this section, we focus on the different categories of methods from information visualization techniques to clustering algorithms of microarray data including graph linearization. For each category, we only give a sample of important articles. For example, there is subsequent amount of litterature dealing with the reordering of matrix representations of social networks not detailed here (see Wasserman and Faust [WF94], section on sociomatrices). A complete survey remains to be done.

4.2.1 Interactive techniques

The first tangible device to reorder matrices was created by Jacques Bertin. Figure 4.2 shows the physical reorderable matrix he called Domino. These photographs were taken during the interview we had the opportunity to perform in March 2005.


(a) Matrices in the semiology (b) Jacques Bertin

(c) Tangible matrix (d) Unlocking rows and columns (e) Unlocking the rows

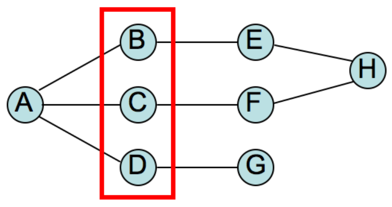
Figure 4.2: Tangible matrix. From our interview of Jacques Bertin (15th of march 2005).

Other interactive techniques exist in the information visualization field. With TableLens [RC94] and InfoZoom [SBB96], users can reorder a table by interactively moving rows and columns or sorting according to a given column. An early example can be found in [BK77]. Other systems provide these basic features including the reorderable matrix [SM05] and our system MatrixExplorer [HF06a].

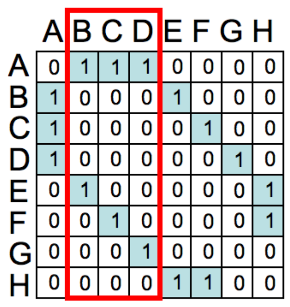
4.2.2 Algorithm for automatic reordering

In the next sections, we present methods to reorder tables and methods to linearize graphs. Graphs benefit of a dual representation: node-link diagrams and adjacency matrices. Thus, methods to linearize node-link diagrams can be applied to reorder corresponding adjacency matrices. Moreover, methods used to reorder tables can be applied to reorder adjacency matrices, which are in fact, potentially sparse tables. While applying node-link diagram linearization algorithms to matrix reordering is immediate, applying table reordering methods to adjacency matrices requires an additional operation.

Applying table-based methods to adjacency matrices A large number of methods to reorder tables are based on metrics comparing rows or columns. As adjacency matrices are much sparser than microarray tables, the differences between a set of columns may be harder to identify. For example, Figure 4.3 shows a simple graph (a) and its corresponding adjacency matrix (b). Computing a simple distance (Manhattan) between B, C and D is not enough to decide of a particular order. To remedy this situation and introduce more information for the reordering algorithm, we decided to apply the reordering algorithm to the matrix of shortest paths instead of directly using the adjacency matrix. Figure 4.3(c) shows that if, instead of considering only the direct neighborhood of vertices, the whole pattern of connection is taken into account then it is possible to identify that B and C are more similar and should be placed beside each other.

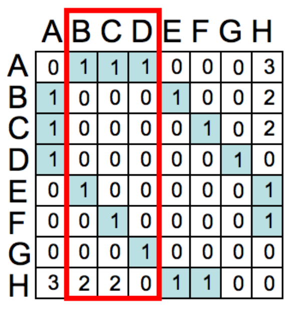


(a) Node-link diagram



	A	B	C	D	E	F	G	H
A	0	1	1	1	0	0	0	0
B	1	0	0	0	1	0	0	0
C	1	0	0	0	0	1	0	0
D	1	0	0	0	0	0	1	0
E	0	1	0	0	0	0	0	1
F	0	0	1	0	0	0	0	1
G	0	0	0	1	0	0	0	0
H	0	0	0	0	1	1	0	0

(b) Adjacency matrix



	A	B	C	D	E	F	G	H
A	0	1	1	1	0	0	0	3
B	1	0	0	0	1	0	0	2
C	1	0	0	0	0	1	0	2
D	1	0	0	0	0	0	1	0
E	0	1	0	0	0	0	0	1
F	0	0	1	0	0	0	0	1
G	0	0	0	1	0	0	0	0
H	3	2	2	0	1	1	0	0

(c) Shortest path matrix

Figure 4.3: Consider a simple graph and its dual representation: node-link diagram(a) and adjacency matrix(b). Deciding how to reorder columns B, C and D using a simple Manhattan distance between the elements of the adjacency matrix is not conclusive ($d_{B,C} = 2, d_{B,D} = 2, d_{C,D} = 2$). However, using the same metric on the matrix of the shortest path provides additional information: B and C have more similar connection patterns ($d_{B,C} = 2, d_{B,D} = 4, d_{C,D} = 4$).

Thus, to apply table-reordering algorithms to adjacency matrices of graphs, we perform the following steps (for each connected component):

1. compute the shortest paths (SP) matrix
2. compute the distance matrix between rows of the SP matrix
3. compute the distance matrix between columns if the graph is directed
4. run the algorithm to find a linear order of the distance matrix (a TSP heuristic for example)
5. reorder the rows and columns of the adjacency matrix using this linear order.

The intermediary step of computing the SP matrix reduces the impact of noise (as the whole connection pattern is taken into account) and gives more information for low degree vertices (for which the rows and columns are very sparse). However, computing the SP matrix is quadratic, as is the computation of the distance matrix for rows and the distance matrix for the columns.

4.2.3 Table-based reordering techniques

In this section, we present methods applied to tables (complete numerical tables), essentially coming from the biological field, where researchers need to reorder microarray data.

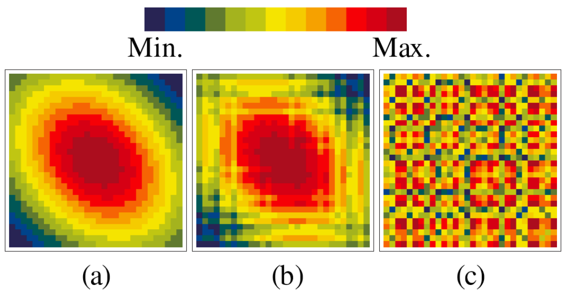


Figure 4.4: Robinson (a) and pre-Robinson matrices (b) and (c). Figure extracted from [CHHGWJ$^+$04]

Robinson matrix techniques These techniques are based on ordering optimally a table or matrix given a line or column similarity/proximity metric. The principle is to reorder the similarity matrix so it becomes a Robinson matrix, which has been shown to compute the minimum path length linking all the elements [Rob51]. A matrix is called Robinson if the elements in its rows and columns do not increase when moving away horizontally or vertically from its main diagonal. Figure 4.4 shows a Robinson matrix as well as two examples of pre-Robinson (a permuted Robison matrix is a pre-Robinson). The principle of these methods is to transform a given matrix optimally into a Robison matrix or into a near-Robinson matrix. Chen et al. present a survey on these techniques in [CHHGWJ$^+$04]. We present the three most common categories of methods:

· *Hierarchical clustering and seriation* These techniques use a hierarchical clustering algorithm followed by a seriation. The clustering algorithm provides a cluster tree according to a given similarity matrix. However, the leaves of the tree can still be permuted to obtain an optimal order. Several work have been done, especially in biology, to compute the optimal leaf ordering [ESBB98, BJDG$^+$02, Bra07].

· *Traveling salesman problem (TSP)* To compute an optimal order of the element of a matrix, these techniques aim at solving the traveling salesman problem. The principle is to compute a distance table between all elements and compute the optimal tour. The traveling salesman is a NP-complete problem widely studied and many heuristics exist to compute this optimal tour. The early idea has been presented in [Len74], while the first implementation to biological microarray table has been showed in [CZ04] and we realized the implementation on adjacency matrix [HF06a].

· *Spectral methods* Atkins et al. [ABH99] demonstrated that a matrix could be transformed into a Robinson one or a good approximation using spectral algorithms. The principle is to compute the eigenvectors of the Laplacian matrix of the initial matrix, the second smallest eigenvector also called Fiedler vector is used to reorder the matrix. The major strength of these methods is their fast computation time.

Data dimension reduction This category contains statistical methods, used to reduce multidimensional datasets to lower dimensions. Principal Component Analysis (PCA) is the most common technique. PCA consists in computing the eigenvector of the data covariance matrix of the table. The principle is to retain those characteristics of the data set that contribute most to its variance, called the first principal component. Reordering a table is done by using this first principal component. Examples of the application of this method can be found in [TY85, CR95]. For very large datasets, Harel and Koren propose a multi-scale method called HDE [HK00b].

Heuristics A number of heuristics exist to reorder matrices, mostly based on Sugiyama algorithm to reduce the edge crossings in bipartite graphs [KST89]. Examples can be found in [Sii99, SM05, MS05].

4.2.4 Graph-based reordering techniques

In this section, we present methods to linearize a graph. The linear order of the graph vertices is used to reorder the corresponding adjacency matrix.

Objective Functions A large number of methods are based on the optimization of an objective function as shown in [GJ79](pp199-201). They are multiple ways of defining this objective function (for example 6 measures are presented in [DPS02]) and no clear quality assessment is known.

A common example is to compute the Minimum Linear Arrangement (MinLA). The objective function is to find the linear ordering the vertices of a graph such as the sum of edge lengths is minimized. Koren and Harel review several of these techniques included a multi-scale method for large datasets [KH02]. Several experiments are presented in [Pet01]. Building on this work, Bar-Yehuda et al. [BYEFN01] present a heuristic algorithm in polynomial time for the MinLA problem and minimization of the maximum cut.

Spectral Method These methods are based on the spectral graph theory, computing eigenvector of related graph matrices. We already presented the principle of these techniques in the previous section as well as for drawing graphs and computing clusters. More details can be found in [Kor05]. For very large datasets, Koren et al. presented a multiscale method [KCH03].

Blockmodeling The last category of method is widely used in social analysis and is called blockmodeling [DBF05a] (see sections 2.1 and 2.2). Blockmodeling is based on the notion of structural and regular equivalence [Fau88]. Two vertices are structurally equivalent if they have similar patterns of connection to the same neighbors. This notion is generalized to regular equivalence if they have the same or similar patterns of connection to (possibly) different ones. The goal of blockmodeling is to find sets of vertices (blocks) that are equivalent (structural or regular equivalence). The matrix is then reordered from this decomposition in blocks.

4.2.5 Towards assisted reordering

During participatory design and various interactions with matrices and software for manipulating them, we observed that no automatic ordering seems to fulfill the user needs. PermutMatrix [CP05] is probably a first step toward the interactive use of automatic algorithms to reorder matrices. In this system, Caraux provides the user with a large number of algorithms and allow to fix their parameters. However, this system requires a very thorough knowledge of the various techniques. After the participatory design sessions, we formed the hypothesis that one needs to manipulate the data, reordering it with different algorithms (possibly without advanced expertise in all of them), arranging it manually to better understand the data and its trends. Therefore, we designed a number of interaction techniques to perform assisted reordering. The principle is to let the user guide automatic algorithms by selecting a given part of the matrix to reorder. Chapter 5 presents these techniques to manipulate, compare and find a consensus among different orderings.

4.3 Evaluation

Assessing the quality of a matrix ordering is a challenge similar to define the aesthetics of graph layouts. While several objective measures exist, we attempted to get an empirical evaluation of reordering algorithms and their quality in matter of supporting the finding of groups and outliers in a dataset, maximizing both the readability and the interpretation potential.

4.3.1 Objective measures

There are a number of objective measures used in graph theory that can be also used as objective measures to evaluate matrix-reordering algorithms. A number of these measures are presented in [DPS02] such as *Bandwidth*, *Minimum Linear Arrangement*(MinLA) and *Minimum Cut Width*(MinCut). These measures describe the properties of a given order and help to compare several of them. For example, Bandwidth provides an idea of how diagonal the matrix is while MinLA ensure that the vertices placed beside each other minimize the sum of the total edge length.

However, these measures fail to capture a number of visual features we can intuitively identify. For example, the number of blocks and their size is important when looking for communities in a social network. Several measures are presented in [FB04] but are only computable on reordering methods based on an initial clustering process. To better understand what are the visual features affecting readability and understanding of matrices, we performed a user experiment.

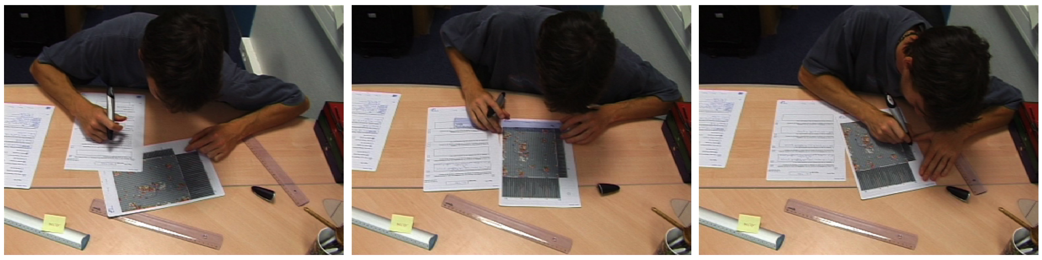


Figure 4.5: Experimental set up

4.3.2 Controlled experiment

Our goal with this experiment was to measure if visual table data layout affects the user understanding and how it affects it (Figure 4.5). We primarily focused on static visualizations and avoided all interaction issues. We designed this experiment as a 3 ordering $\times$ 2 datasets within-subject. We preferred a within-subject design to limit the inter-subject variability (the difference between the exploration processes of two different participants).

Reordering techniques To conduct this experiment, we used three table-ordering algorithms commonly used to reorder rows and columns of table data. The first order is alphabetic (A) and considered as a control order. The two others (C) and (T) are issued from the field of bioinformatics. (C) is a hierarchical clustering followed by a linearization and is presented in [ESBB98]. (T) is based on the traveling salesman problem and described in [CZ04]. They are used to reorder DNA microarrays (visual table data presenting gene expressions). We chose them based on their robustness and scalability: they can handle real datasets with noise up to ten thousands of elements in a few seconds.

Datasets We chose two real datasets in order to conduct this experiment in a realistic context. Our goal was to find interesting datasets of reasonable size to keep our participants motivated. Motivation is a crucial factor as we intend to evaluate how a user explores and understands a dataset. This experiment had to keep participants interested in the data and willing to explore its structure to get realistic results. As our participants were mainly computer science students and researchers, we chose the university masterÕs grades dataset (Figure 4.6) in which they appeared as students or teachers. We picked also a more general table data issued from the CIA World Factbook (Figure 4.7). This dataset gives statistics (productivity, export, population) for most of the countries of the world.

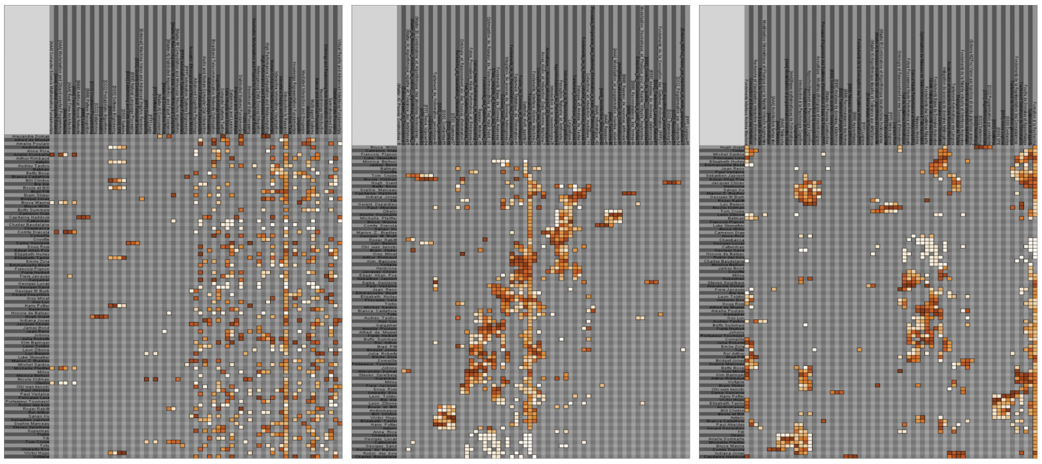
Tasks We proposed to express the exploration process in terms of tasks, defining what lies under Òunderstanding table dataÓ. After several interviews with novice users and visualization experts, we ended up with a set of tasks and we organized them hierarchically. We organized this set in three complexity levels and also distinguished readability from interpretation (see chapter 8, section 8.4.1 for more details). Readability tasks deal with the visual representation and are independent of the dataset. In other words, participants can perform readability tasks without interpreting the representation. Understanding a visual table is a combination of readability and interpretation so we attempted to capture both.

Data collection Participants had to answer a 4-page questionnaire using an Anoto digital pen on Anoto paper. With this technology, users can read and draw on almost regular paper with an almost regular pen [1]. The strokes drawn on paper are captured by the pen and can be sent to a computer. In addition to the stroke trajectory, Anoto pen also identify pages and provide the precise time of each stroke. We chose Anoto technology to simplify the tasks and perform the evaluation without using a computer, in a comfortable setting for reading matrices.

When we designed the questionnaire, we kept in mind two goals: do not orient the exploration and provide tasks from basic to complex. The questionnaire contained 32 questions and a visual

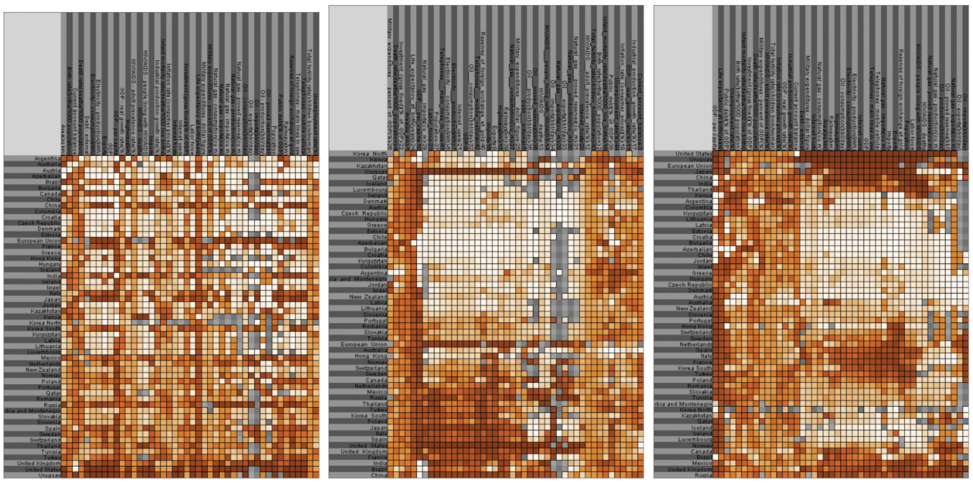
[1]Anoto paper has special patterns of small dots recognized by the digital pen. `http://www.anoto.com/`

table representation. Half of the questions were multiple-choice questions, the other half were open questions. Participants had to tick boxes for multiple choice questions, write comments and explanations for open questions and draw circles or annotations directly on the visual table printed on a separated page. The digital pen recorded the time of each stroke and the stroke location on the digital paper. Thus, we collected quantitative and qualitative data.



(a) Alphabetic (b) C (c) T

Figure 4.6: Visual table data of the university master's grades: 88 students(rows)*65 courses(columns). The color intensity represents values (low values are white, high values are red, grey represents no value). Alphabetic order (a), automatic orders C (b) and T (c).



(a) Alphabetic (b) C (c) T

Figure 4.7: Visual Table Data for World Factbook 2005 (filtered): 55 countries(rows)*44 criteria(columns). The color intensity represents values (low values are white, high values are red, grey represents no value. Alphabetic order (a), C order (b) T order (c).

Results We analyzed the time of answer using a classic analysis of variance and found no significant results (the variability between participants being too large). The analysis of the multiple-choice questions and comments did not lead to major results either.

Consensus on groups The most interesting results came from the labels and drawings directly written on the visual representation. The related tasks were: circle groups, give them a meaningful label, and circle outliers. To analyze this data, we used a visualization superposing all the groups for all the subjects for each ordering algorithm as shown in Figure 4.8.

In this figure, we can clearly see a consensus among all participants on the groups circled. This is especially true for the C and T layouts. In addition to this visual data, we analyzed labels and classified them into categories in order to determine first if groups found by most of the participants have the same connotation, then to compare groups from one layout to another. Although some of the results are not analyzable, our analysis points out notable facts.

For example, most of the participants identified a group of students attending cognitive science courses. This group is circled in Figure 4.8. They labeled it Òcognitive sciencesÓ, ÒcognitionÓ, Òcognitive psycho.Ó, ÒpsychologyÓ or even Òsocial sciencesÓ. It represents a real trend in the data as this master course is the result of a fusion of three previous masters, one being the cognitive sciences master. They were able to identify this group only in C and T ordering, which shows that reordering the table lead to additional insights on the data.

Identifying a group as several visual clusters Another interesting observation is that most of the participants identified this group in both C and T layouts even though it was split into three visual clusters in the T layout. This result means that the concept of group is not only a visual pattern but also a matter of interpretation. This explains the importance of distinguish readability from interpretation while studying the understanding process and might also suggest that the most readable representation might not be the optimal one (if it interferes with the interpretation).

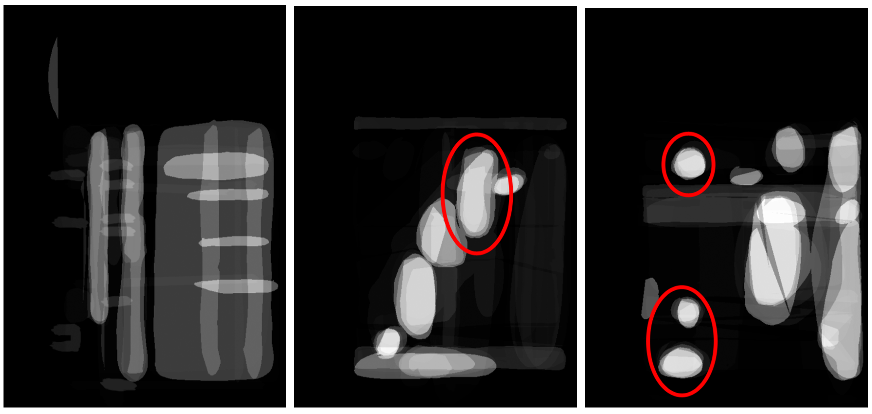

(a) Alphabetic (b) C (c) T

Figure 4.8: Superposition of groups for all participants. Master's dataset. Alphabetic order (a), C order (b) and T order (c). A group of cognitive science students (identified by our participants) is circled in red.

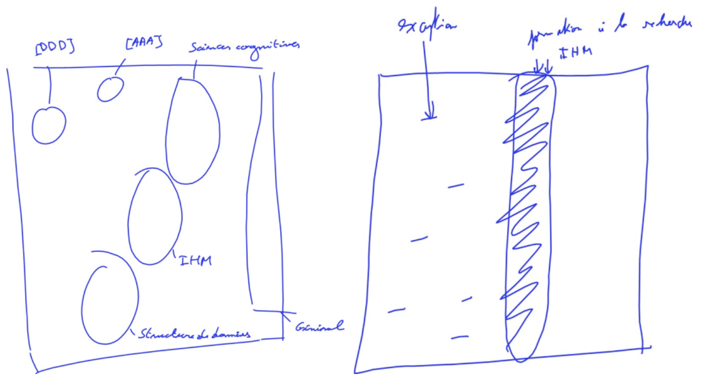


Figure 4.9: Memorization sketches for masterÕs grades. The left drawing is a combination of C and T orderings: the participant identified several groups of C including cognitive science students while the larger group on the right side is the general master courses appearing on the right of T. The right drawing is a combination of A and C: the participant recalled the courses selected by only few students (left side of A) and the large group of general master courses appearing in the center of C.

Merging the ordering landmarks During the experiment debriefing, we asked our participants to sketch what they recalled of the visual tables (Figure 4.9). We formed the hypothesis that participants would find more insights in the data with a good quality ordering and therefore better remember the representation. In the data we collected, we indeed observed significant landmarks of each of the ordering. However, participants often merged the visual features of the three orderings and no particular algorithm could be qualified of producing a better recall. This result is also important as it might suggest that each ordering provides valuable insights on the data, even the most basic one (alphabetic in our case).

Conclusion We presented an experiment attempting to evaluate how visual table layout affects user understanding. We only could witness consensus among the participants especially on groups. Assessing the quality of an ordering is a difficult problem. We believe more experimentation with expert users is required to identify the visual features impacting human understanding and interpretation and formalizing up measures to capture them.

4.4 Conclusion

A very large number of techniques exist to reorder matrices and tables. However, understanding how different orderings affect the readability of a representation and its interpretation remains a key question. While our experiment showed that different matrix orderings lead to different insights on the data, we did not collect evidence that one ordering outperformed another for specific tasks (for identifying groups or outliers for example). After this user study, we believe that each ordering gives a different view on the data helping the user understand it. We also believe that manipulating the matrix itself helps to understand the data and find insights. This is why we chose to provide our users with a set of automatic algorithms as well as interaction techniques to manipulate matrices.

MatrixExplorer: combining matrix and node-link representations

CHAPTER 5

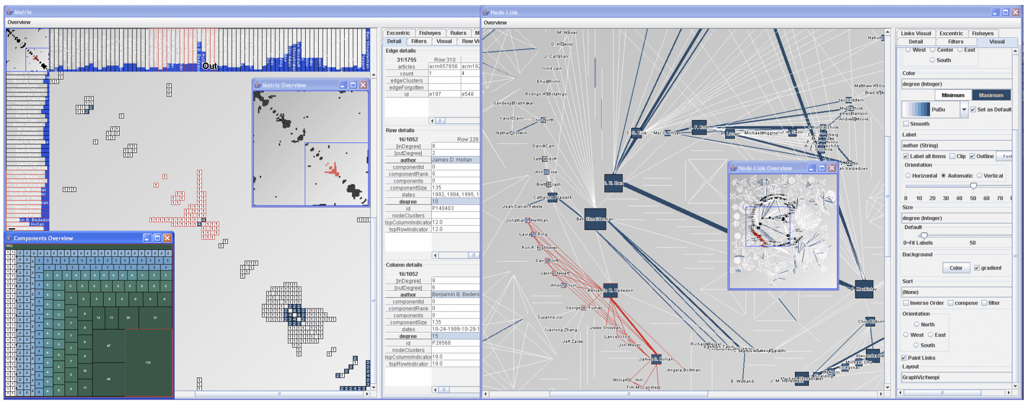


Figure 5.1: MatrixExplorer

This chapter presents MatrixExplorer [HF06b, HF06a] (Figure 5.1), our system to visually explore large social networks. We first describe our prototype and design choices, then its evaluation done through a case study on scientific publication data. We conclude on the lessons learnt and the novel research problems raised during the conception and evaluation of this first tool.

5.1 Research problem

Current systems for analyzing social networks are mainly based on statistical and analytical analyses. Our goal is to bridge the gap between research in information visualization and social network analysis, designing interactive techniques to support the visual exploration of social networks. The research problem we address in this chapter is the focus of the whole dissertation:

➤How to support the interactive exploration of social networks?

We presented elements of answers in Chapter 3 through a set of design goals. In this chapter, we describe our initial prototype. From the information we collected and our observations, we divided the complex process of exploration in three main steps, using these steps as keys to design our system.

1. Initiate the exploration. This step consists in launching the exploration process, it should help analysts to identify an entry point to the analysis of a new dataset but also support the re-analysis of a previous one. Therefore, we face two main challenges: presenting a readable overview of the whole network and helping researchers to find landmarks on a network they already know.
2. Explore and collect insights. This step is the core of the exploration, the challenge is to offer new perspectives on the data and support the discovery of a maximum of insights with a minimum of effort. Our objective is to provide efficient visualization techniques as well as intuitive interactions to ease the exploration process.
3. Find a consensus. This final step aims at finding real trends in the data. The problem here is to minimize bias and misleading visualizations or representation artifacts.

5.2 Design

MatrixExplorer is a first attempt to fulfill the design goals we collected with social scientists and support the three main steps we defined earlier. In this section, we describe our initial prototype and design choices.

5.2.1 Coupling node-link diagrams and matrices

MatrixExplorer is based on two representations: matrix-based and node-link diagrams (Figure 5.2) meeting the first requirement of our users (G1-Multiple representations). Node-link and matrix visualizations are synchronized in order to let the user work with both representations, our goal is to allow them to switch smoothly from one to the other.

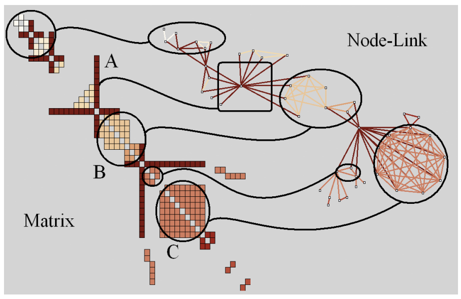


Figure 5.2: Visual patterns in Matrix and Node-link representations of social networks. A represents an actor connecting several communities, B a community and C a clique (complete subgraph).

Multiple visualizations are synchronized by selection and filtering. Basically, if a user selects a set of actors in the matrix, this same set will be selected in all other visualizations (selection) and data filtered in one visualization will disappear from all others (filtering). Selection improves the transition from one representation to the other and constitutes the core of the coupling. Filtering preserves the coherence of the visualizations by presenting the same data, even if the attributes visualized are different. In addition, visualizations can be synchronized by any visual attribute, simply by interactively setting the same attribute for the same visual variable. Thus, the user still has the possibility to not synchronize the visualizations in order to compare two attributes.

Our objective is to support network exploration using both representations, allowing users to accomplishing tasks with one representation or the other (according to what representation is more effective or familiar) and visualizing the effect of a selection, or filtering, on all visualizations and their overviews. Figure 5.2 shows a dual-representation of a co-authoring network and the correspondence of visual patterns in matrix and node-link representations. As an example of how users can couple both representation, we describe our process to obtain Figure 5.2. We first automatically ordered the matrix, identified clusters (communities) by looking for blocks and attributed colors to identify them directly on the matrix. Then, we switched to a node-link diagram, displaying the community colors and laying the network out manually in order to better visualize how communities are linked and organized. Finally, moving back and forth between both representations, and adjusting the communities found by changing node-link layouts, we identified the global structure of the network.

While it takes time and practice to quickly switch from one representation to the other (especially to read quickly patterns exhibited by a matrix representation as node-link diagrams are more familiar), there is a clear correspondence between the two representations and they complement each other. For example, node-link diagrams may show groups densely connected fairly quickly while matrices can particularly highlight the details of the connectivity, supporting the discovery of missing links within a group for example.

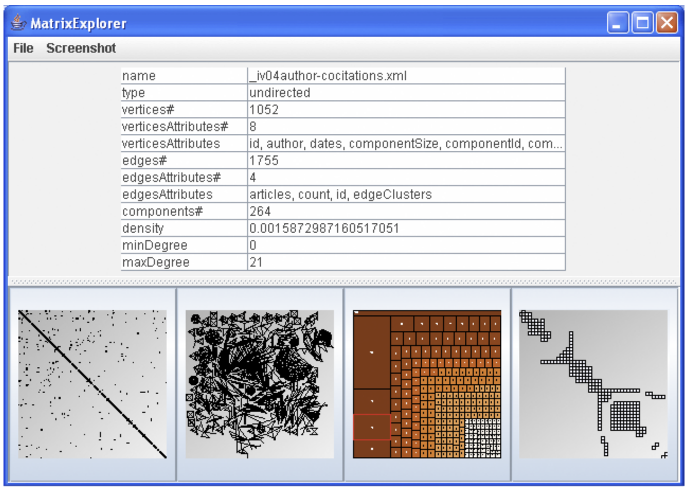
5.2.2 Initiating the exploration

To ease the launching of a new dataset exploration and support the re-analysis of a previous one, we combined overviews of the data, of the visualizations and of the workspace.

Workspace overview MatrixExplorer proposes a quick overview of the user workspace. This overview includes for each dataset: general information of the network and a visual overview of the related visualizations created by the user (Figure 5.3). This workspace overview covers design goals G4 (dataset general information) and G6 (analytical information) as well as G3 (Overview). General statistics on the dataset as well as thumbnails of the different visualizations related are shown in a single window providing an overview of the exploration state. As the exploration generates many windows, we offered the possibility to show/hide visualizations in one click on the corresponding thumbnails.

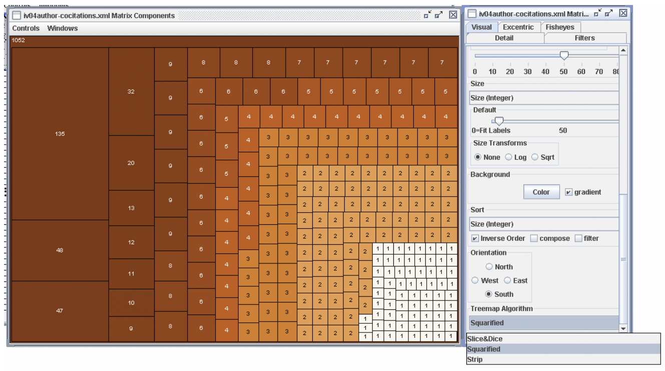
Data overview: statistics Information appearing for each network is: name, directed or not, number of actors, number of relations, number of connected components, number of actor attributes and their labels, the number of relation attributes and their labels, global density of the network, minimum and maximum degree, and in/out degree if the relations are directed. We defined this list with social-science researchers, who specifically pointed out the lack of information about the attributes of actors and relations (G4 and G6). Our latest addition to this information is examples of attributes values as well as an indication of their distribution, helping finding landmarks for previously analyzed networks.

Data overview: connected components The connected component visualization plays a special role in network exploration. First, it is always a readable overview of the network, quickly showing its macro-structure. Secondly, it is a starting point of the exploration as it filters matrices and node-link visualizations according to the current selected connected component. Social sciences researchers expressed this need in almost all interviews (G2). We choose to visually organize connected components as a compact rectangle in order to build a mental map of the macro-structure of the network (Figure 5.4). This visualization is very useful as it can give a glimpse of the small-world structure of a network for example. In Figure 5.4, the visual variables (rectangle size and color) are mapped to the number of actors of the components (their size) and sorted by decreasing size from top left to bottom right. One click on a rectangle representing a connected component filters the synchronized visualizations. The user may map the visual variables to other attributes (using the control panel shown on the right); change the layout, or even the desired representation.



name	_iv04author-cocitations.xml
type	undirected
vertices#	1052
verticesAttributes#	8
verticesAttributes	id, author, dates, componentSize, componentId, com...
edges#	1755
edgesAttributes#	4
edgesAttributes	articles, count, id, edgeClusters
components#	264
density	0.0015872987160517051
minDegree	0
maxDegree	21

Figure 5.3: Workspace Overview. In this session, the user created 4 distinct visualizations (from left to right): matrix-based representation, node-link diagram of the full network, connected component visualization, and finally, matrix-based representation of one selected connected component.



Size: Size (Integer)
Default: 0=Fit Labels 50
Size Transforms: [✓ None] Log Sqrt
Background: [✓ gradient]
Sort: Size (Integer)
[✓ Inverse Order] compose filter
Orientation: North West East [✓ South]
Treemap Algorithm: Squarified
Slice&Dice
[✓ Squarified]
Strip

Figure 5.4: Connected components visualization.

Visualization overviews In MatrixExplorer, an overview is provided for each visualization, meeting the design goal G3. The primary goal of an overview is to provide an overall picture of the visualization and thus helps users identify the structure of the network and build a mental map. In addition, we observed that users also use overviews as context reminders. While working on a matrix-based representation, they keep an eye on the node-link overview, to verify which part of the full network they are working on. Moreover, they can directly observe the impact of their manipulations on the node-link diagram. This is especially useful when few screen space is available to show all visualizations of interest at the same time. Finally, overviews are also navigation tools. A rectangle represents the current visualizationÕs view. The user can grab it and move it to display a different part of the network.

5.2.3 Exploring and collecting insights

This step, the core of the exploration process, is a cycle composed of successive interactions on the visualizations to configure them (visual variables and layout), filter them, cluster them or compare them, attempting to capture a maximum of insights by creating multiple views of the data, eventually drilling to details of a portion of the dataset or reasoning on its overview. To support this phase, we chose to maximize direct interaction on the representation and favor the use of dynamic queries [AWS92].

Visual variables Assigning visual variables for each network attribute (G5) is a key to create effective visualizations. The InfoVis toolkit [Fek04] provides the framework to interactively map attributes to visual variables for both node-link and matrix-based representations. Users are able to control the visualizations of actors choosing shape, size, color, texture and label and the relations (links) by choosing shape, length, color, thickness and label. Useful interactions are provided to favor direct manipulation and improve the readability of the representations: a control panel lets the user assign each attribute to one or more visual variables, and dynamic filtering and sorting let the user choose what actors are shown and in what order (G7). Moreover, labels are often a main concern of social sciences researchers (G5). To show both actors and relations labels legibly on any visualization, The InfoVis Toolkit provides Eccentric labeling [FP99] and fisheye views [Fur86].

Node-link layout MatrixExplorer provides a number of graph drawing algorithms for laying out node-link diagrams. They are mostly based on the JUNG and GraphViz packages. Essential interaction is also implemented such as moving nodes of the graph by clicking and dragging. In this section, we focus only on innovative features to manipulate matrix-based representations. We detail our interactive tools to reorganize their layouts (G8). We favored interaction and direct manipulation instead of iteratively adjusting a set of parameters (G7).

Interactive matrix ordering MatrixExplorer provides a set of basic interaction tools essential, but not sufficient, for ordering large matrices. These tools include moving one or more rows or columns using drag and drop. They also include a feature (inspired by spreadsheet calculators) that allows users to sort rows and columns according to one attribute; for example, sorting rows according to the vertex names and columns according to the vertex degrees. Thus, rows can be used to find a specific vertex, and columns to find most/least connected vertices. Compared with InfoZoom [SBB96] or TableLens [RC94], the two dimensions can be ordered, and then be used to show the impact of one attribute on another one. MatrixExplorer also provides a tool to permute rows and columns circularly, similar to a ÒpanÓ tool in a paint program with the cells leaving on one side and entering on the opposite side.

Automatic matrix ordering As described in chapter 4, ordering algorithms focus either on ordering visual tables or on finding an optimal linear order for all the vertices of the graph. In MatrixExplorer, we mix the two perspectives and provide both table-based and graph-based techniques. We achieve this by applying methods to reorder tables not directly on the adjacency matrix but on the matrix of shortest paths (SP matrix). This way, we take into account a larger neighbourhood (distance>1) and vertices with similar connection patterns are placed next to each other.

Table-based techniques We selected two fast ordering algorithms from the bioinformatics field to reorder the SP matrix. The first one is based on a hierarchical clustering, followed by a seriation (HCS) and is described in [BJDG^{+}02, Bra07]; the second one is based on the traveling salesman problem (TSP) as presented in [CZ04]. To solve TSP, we use a fast heuristic described in [Hel00]. Both algorithms are based on a metric — similarity for HCS, distance for TSP — between the rows (respectively columns) of the SP matrix. Usually, this metric is either a Manhattan or Euclidian distance, or the Pearson correlation coefficient. By default, we choose a Manhattan distance. Matrices up to 1000 rows* 1000 columns can be ordered in seconds. Ordering larger matrices introduces a noticeable delay. Figure 5.5 presents a matrix reordered using TSP. The resulting matrix exhibits clearer blocks (diagonal with dense blocks); users can identify more clusters (G10) and articulation vertices between these clusters as dark color crosses here. A well-ordered matrix also helps identify outliers (G11) such as isolated relations, missing relation in a community, or actor with special connection patterns.

Graph-based techniques We added a set of other much faster techniques to reorder matrices. The fastest example is a reimplementation of HDE [HK00b] performed by our collaborator Thanh-Nghi Do. This method based on the Principal Component Analysis is extremely fast and can deal with very large matrices in interactive time (see chapter 4). However, according to qualitative comments of our users and our own empirical experience, the resulting matrix is more difficult to interprete and less blocks are identifiable compared to the one ordered by TSP.

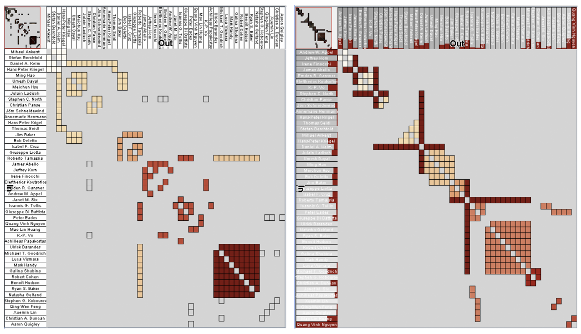
Towards assisted ordering In the second edition of ÒSemiology of graphicsÓ [5], Bertin presents the results of three automatic ordering algorithms: automatic classification, factorial analysis and hierarchical analysis. He argued that none of these algorithms found a satisfactory matrix layout and performed some manual permutations to perfect them. We observed that automatic algorithms rarely provide a satisfying order for a given matrix and a userÕs taste. However, they save a substantial amount of time and effort and offer an initial layout, which is better than a random one or a simple sort. MatrixExplorer´s goal is to propose a good initial matrix layout and to provide interactive tools to improve it, if needed. A ÒgoodÓ order, according to our participants, is one that reveals dense blocks and conversely avoids sparse isolated values. Sometimes, the initial layout — reflecting the data construction or collection method — is already good, as can be seen in Figure 5.6. We took advantage of this potentially good initial order and tuned TSP to improve it iteratively. Then, once the whole matrix is reordered, we propose that users interactively reorder sub-matrices they wish to explore.

Ordering sub-matrices There are two options to adapt our automatic ordering algorithm to sub-matrices: 1) extract the sub-matrix from the initial SP matrix, or 2) compute the sub-graph corresponding to the sub-matrix selected by the user and then compute a new SP matrix. The obvious drawback of computing a new sub-graph is the additional computations required: extraction of the sub-graph, computation of connected components and computation of SP matrix for each component. However, we observed that the number of vertices selected by the user is usually low, and thus the computation time is insignificant. The second drawback has more impact on the user´s understanding. The SP matrix computed for a given sub-graph contains notably less information than the initial SP matrix. Moreover, it may be misleading as the influence of all the unselected vertices is not taken into account. We implemented both solutions. Figure 5.6 shows an example of the two sub-matrix ordering methods. We observed that results obtained with the first solution were more interesting as they let more blocks emerge as expected (SP matrix containing more information). This led us to favor the first version over the second but we tend to provide both, as more experimentation is needed to better understand pros and cons of both methods.

Locks MatrixExplorer allows locking a set of rows and columns together before reordering a matrix. This functionality was not explicitly requested by our users but was detected during the use of our prototype. This feature is useful when, for example, a user identifies a community (set of actors) and wants to find out which external actors communicate with it. It is a constraint taken into account during the order computation: a single element is computed, representing the full set. The order computed takes into account this single element instead of the whole set. Our algorithm only keeps the first and last elements of the sequence and fills the distance between them in the SP matrix with a value of zero, to ÒglueÓ them together. The order is then optimized, as shown earlier, to integrate the sequence of elements, but these two are kept together. At the end, we insert the set back between the two elements.

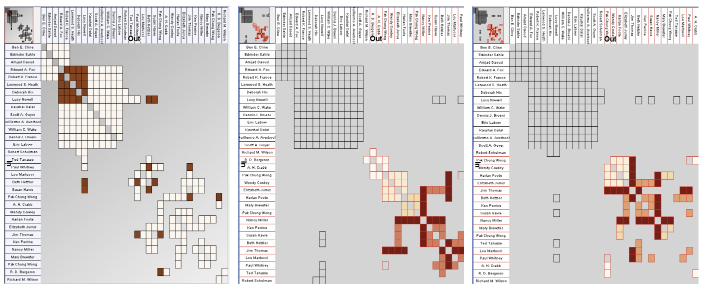
Filtering or forgetting We have implemented filtering in MatrixExplorer to fulfill the design goal G9 to reduce the size and complexity of a network and finely analyze its sub-parts. Users can filter either actors or relations, according to one or a combination of all existing attributes (numerical, categorical or computed). In addition, to visualize the impact of an actor, or a set of relations on the network, MatrixExplorer provides a feature that ÒforgetsÓ actors or relations and visualizes the resulting structure.

This tool is slightly different from filtering: first, the element is still visible although it is made translucent (ghosting); second, the changes in the new structure are highlighted to let the user rapidly identify the impact. In Figure 5.7, the user identifies an actor collaborating with a community as well as a few external actors. He asks MatrixExplorer to forget all collaborations of this actor with the community by selecting it and visualizes the result. After the operation, the user can clearly see that the bottom right of the matrix is affected. The matrix is split up into two independent blocks: two sub-graphs. Thus, the forgotten actor identified is an articulation vertex (or a cut-point) between these two sub-graphs. Moreover, as several relations have been ÒforgottenÓ, *i.e.* not taken into account during the reordering, the lower sub-matrix has been reordered to let the finer structure emerge. Therefore, new blocks (communities colored in different nuance of brown) can be identified, exhibiting the sub-graph structure.



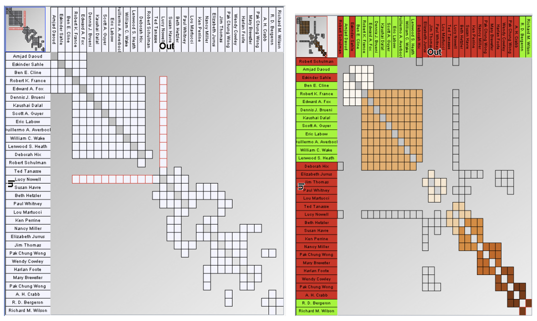
(a) Initial order (b) TSP order

Figure 5.5: Initial order (left) and TSP order (right). Colors represent clusters found by the user. Clusters are different in the two representations. Users found more clusters with TSP order. Headers red indicators (right) represent the distance between adjacent rows/columns.



(a) Initial order (b) TSP order full (c) TSP order new graph

Figure 5.6: In red sub-matrix to reorder. TSP order using the SP sub-matrix (middle), a new graph SP matrix (right)



(a) Initial order (b) TSP order full

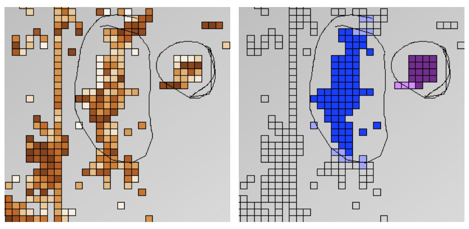
Figure 5.7: Forgetting a number of collaboration for a key actor. Red headers indicate rows/columns with different neighbors, green indicates same neighbors. Other colors indicate communities identified by the user.

Clustering The community structure of the network changes, depending on the actor attributes or the relation taken into account. For example, two clusters of actors may be identified while visualizing a kinship relation, but three different clusters may be identified while visualizing their phone calls. Therefore, MatrixExplorer supports multiple clusterings (expressed as G10).

Automatic clustering Many automatic clustering algorithms exist. In MatrixExplorer, we included only the most common in social network analysis: the community detection algorithm based on edge-betweenness [GN02] implemented in JUNG [FOS$^+$05]. This algorithm gives very good results with networks having a small-world structure. We also integrated automatic clustering according to a given attribute.

Interactive and fuzzy clustering For interactive clustering, MatrixExplorer proposes two selection modes: click and drag or lasso. While click and drag exactly transform the elements selected into a community, the lasso is used for a more "relaxed" clustering. We implemented the fuzzy selection in response to usersóbservations. As matrices are very similar to tables or spreadsheets, users tend to adopt an ÒexactÓ or ÒpreciseÓ behavior when selecting groups. They spend considerable time determining whether a particular edge is included or not in a cluster, while in practice, community identification is often approximate. The idea of fuzzy clustering is to make elements at the border of the lasso more translucent to denote their weakest belonging to the community (Figure 5.8). This simple feature really helps making the exploration more fluid as users spend less time tuning their selection, feeling the cost of making an error (including a wrong actor into the community) is somewhat reduced.

Clusters visualization We observed that users also created clusters based on edge attributes — such as isocontours — and not only on blocks. We provided a tool to quickly switch from the standard visual mode showing colors based on an edge attribute, to colors based on cluster indices. These cluster indices are displayed in a transient mode, similar to Vizster´s X-ray mode [HB05]. Users switch to this cluster visualization mode by pressing a key or a mouse button and switch back to the normal visualization mode by releasing the key or button. Users may also choose to display one clustering with a particular visual variable such as color or shape, since each clustering is implemented by a categorical attribute added to the edges. This also allows propagating a given clustering to other representations.



(a) Lasso selection (b) Fuzzy clustering

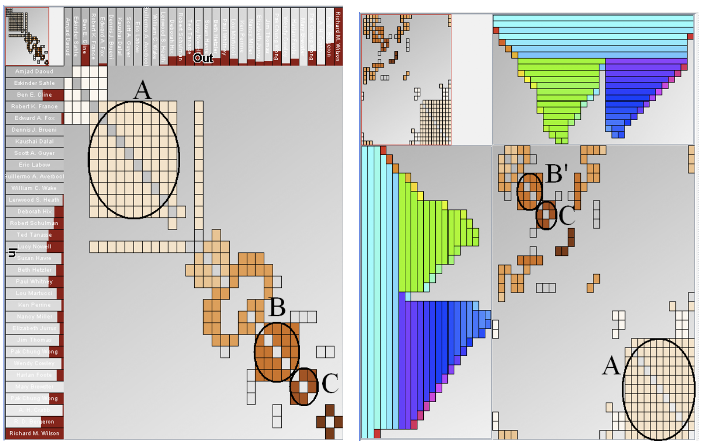
Figure 5.8: Lasso selection on values visualization mode and resulting cluster visualization.

Articulation point A main drawback of matrix-based representations is the 1D order of all vertices, which makes it difficult to represent articulation vertices between several clusters. However, well-ordered matrices let the user quickly identify communities and articulation vertices with a little training. Once communities are identified, the node-link diagram may be reorganized and clearly present the results. This is a major advantage of our dual-representation system: explore and discover with matrices, and present with node-link diagrams.

5.2.4 Finding a consensus

Consensus among layouts Different layouts often imply different clusterings. It is important to be able to identify common clusters among layouts: i.e. to find a consensus when it exists (G12). MatrixExplorer offers this possibility. The procedure simply consists in identifying clusters as described in the previous section and ordering the matrix according to another layout. Clusters either explode in several parts or are conserved (Figure 5.9). This helps to identify real groups in the data to ordering artifacts and, in the case of fuzzy clustering, can help refining community composition by eliminating or including border side elements (translucent ones). The same method can be used to compare clusters of actors for different kind of relations. In Figure 5.9, additional information on the clustering is displayed in the row and column headers. Depending on the algorithm used to reorder the matrix, this additional information is either a red histogram showing the distance between adjacent rows (columns) computed by TSP or the hierarchical clustering tree resulting of HCS presented as an icicle tree colored according to the similarity of its elements (blue and green in the Figure).

Consensus among clusterings To find the differences between two clusterings, users may switch from one clustering visualization mode to the other (X-ray) or simply choose a visual variable for one clustering and another for the second; for example, shape and color. A consensus is identified when both visual variables are in adequacy.



(a) TSP order (b) HCS order

Figure 5.9: Consensus between TSP and HCS orders. We observed that a consensus exists for A, B and C. However, B is slightly different and lost some of its elements with HCS.

5.3 Evaluation

Because the succession of tasks involved in the exploration of social networks are hard to identify and because of the lack of existing tools supporting visual exploration of social networks, we opted for a case study rather than a controlled experiment comparing our system to another one.

5.3.1 Case study

This case study deals with the analysis of 20 years of 4 Human Computer Interface (HCI) conferences publications. We selected this dataset because it is of interest to the whole HCI community: from young researchers wishing to get an overview of the field, to senior researchers positioning themselves within the community. Thus, our findings are valuable and evaluable by any HCI researchers. We also selected publication data, as we are both designers and users in this perspective. Our results in terms of gathered insights and research methodology (to collect and analyze data) is presented in Chapter 8. We only report in this section the usability problems or missing features and conclude with higher-level problems that emerged during the study.

The general objective of this case study is to explore a social network containing **26,942 actors** and **118,865 relations**. It is composed of three types of actors: conference, researcher and article; as well as six types of relations: articles accepted to conferences, articles written by authors, articles referencing articles, and additional computed relations: authors collaborating with authors, authors citing authors and conferences citing conferences. We ran the case study over **four months** and observed a number of problems related to the interface usability, the physical devices settings missing information about the exploration process and the data state, the history and saving mechanisms and missing features to support story-telling and the creation of pictures from the data. We now detail the most important points.

1-Physical settings. The exploration of a large network requires many windows: workspace overview, data information and statistics as well as several visualizations of the whole dataset and of some sub-parts. Most of the visualizations require additional windows when manipulated such as an overview for navigation or a control panel for filtering. An additional constraint exists for synchronized visualizations, which often requires to be placed side by side to observe how modifying one visualization affects the others. Handling this large number of windows (around 8 or 10) is very tedious when having a standard single screen (19”) and observing two visualizations side by side is almost impossible when the network contains more than a hundred nodes. We realized that a comfortable setting includes at least 3 screens (of 19”), larger screens being more comfortable for large matrices. This observation is important in light of our group of users, who rarely possess more than one screen. In addition, we observed that MatrixExplorer requires at least 2GB of RAM and the latest generation of processors (Pentium IV) to provide a comfortable use.

2-Switching representations. Our main design choice with MatrixExplorer was to couple node-link diagram and matrices, attempting to use the best of both representations. We expected that according to the current task to perform, the user would select the most efficient representations. First, we noticed that switching from one representation to the other was cognitively demanding as graphic elements in both representations are very different (nodes become row and column header while links become a square in the matrix cell). Therefore changing often of representations required a solid practice. We observed that we mainly used matrices, even for tasks *a priori* more efficient with node-link diagrams. In many cases, switching representations required more time and effort than performing the actual task with the current representation. In general, we used node-link diagrams for communication purposes (explaining our findings to visitors) or for illustration purposes.

3-Synchronizing representations. As we previously explained, MatrixExplorer allows synchronizing several representations. For example, the connected component view can be linked to matrix and node-link representations, allowing filtering these representations by simply clicking on a given connected component. A main usability problem we faced was to let users define which visualizations must be synchronized (and how) and which should not. This problem has been solved soon after we discovered it by researchers from UFRGS, in Brazil [PF06].

4-Searching. MatrixExplorer missed the keyboard search; therefore we developed alternative strategies such as filtering all elements except the point of interest or sorting rows alphabetically to find the right element.

5-Predicting actions. We observed two frustrating aspects in using MatrixExplorer: not being informed of the duration or the current state of a computation such as a layout or ordering computation. This was particularly frustrating when dealing with a large amount of data as precise ordering algorithm could require several hours. The other source of frustration is the indeterminist aspect of some algorithms, which never produce the same result. Some graph layouts for instance produce different results (often simple rotations are needed, but it becomes tedious to find previous points of interest) for a same graph. In addition, many algorithms are not stable, and a minor change in the data can dramatically affect the results.

6-Summarizing the current exploration state. We noticed that there was missing information to help summarizing what was the current state of exploration. We identified two kinds of problems: reminding users what attributes are currently affected to what visual variables and what are the current statistics of the network visualized. The first problem is inherent to the organization of the large number of controls, categorized in several tabs, which makes tedious their browsing. The second problem is a matter of displaying and differencing the information on the network data and on its visualizations. As a visualization can be filtered or aggregated, providing information on the number of actors and relations currently visualized would be very useful.

7-Undo and history. MatrixExplorer missed the undo feature, which is frustrating when manipulating a large amount of data. For example, running a second time an ordering algorithm on a same matrix only to get back to a previous state is very upsetting. Coupled with the lack of history mechanism, MatrixExplorer required us to save files regularly, sometimes even writing down the succession of actions performed.

8-Data editing. We realized that the data we collected was suffering of noise, in particular researchers name duplications. Being able to edit the data and merge two actors together would have been very useful. Moreover, changing the data file also invalidate the previous representations: for example merging two actors alter the ordering of the matrix rows, the result is simple to propagate when the element indexes do not change (it consists in simply removing the actor from the ordering), however, the global ordering might change, which requires a new computation.

9-Saving. During the exploration, we implemented a number of saving features. From modified data file to png pictures including ordering or clustering configurations.

10-Multiscale aspect. We noticed that MatrixExplorer was missing a multiscale aspect. While it integrates fisheye views for local magnification and provides overview of the whole representation, MatrixExplorer is missing more general zooming features allowing the manipulation of the visualization at several levels of details. Supporting aggregation of groups of actors or relations would also be useful.

11-Temporal aspect. We noticed that MatrixExplorer was missing a temporal aspect as we were dealing with time-varying data. While it is possible to filter the data by year and successively visualize it, more advanced feature such as computing an ordering or a node-link layout to show the network evolution would be useful.

12-Creating figures. The main problem we faced was the integration of our representations into a standard A4 format. We had to manually add a legend as well as potential useful annotations or comments. We realized that having these simple features available in MatrixExplorer would save a lot of time. To communicate on our findings, we also used very simple representations in addition of node-link diagrams and matrices: histograms. We created these histograms by displaying attributes on matrix headers as visual variables (color and area), sorting rows and generating a global picture. Then, we manually extracted the portion of interest from the whole picture. Providing a feature to select portions of a representation to create a picture would have saved us some time too.

5.4 Conclusion

This case study raised a number of new research problems. While we brought elements of answers for several of them, many remain to be studied in the future.

Aggregation and multi-scale navigation A recurrent need while analyzing large data are the aggregation and zooming features. In collaboration with members of AVIZ, we designed and implemented **ZAME** [FED$^+$08], a multi-scale matrix with geometric zoom and automatic aggregation to explore huge graphs at several levels of abstraction.

Exploration of the matrix When exploring a large matrix, we observed two major issues: following paths between two given actors and navigating to connected actors located far away of a given actor. We present solutions to these problems in Chapter 6 with **MatLink** and **Mélange**.

Controlling visualizations Figure 5.10 shows a subset of the control panels needed to control a visualization. There are at least five categories of panels needed when performing an analysis: *data details* possibly editable; *visual variables* for each types of graphic elements (nodes and links in node-link diagrams; rows, columns and edges in matrices); *dynamic queries* such as filtering; *navigation techniques* to parameter such as fisheyes or eccentric labels; *various algorithms* to apply to the data or visualization such as layout and clustering algorithms. In MatrixExplorer, we opted for the organization provided by the InfoVis Toolkit: a tab by control type. However, navigating through the tabs is tedious especially for novice users, and some controls should be always visible as they alter the visualization (such as filtering for example). Reorganizing the controls and provide a visualization of their states is a new challenge. One way of suppressing controls on visual variables would be to integrate them into an interactive legend, included in the visualization.

Managing the visualizations Another problem arising from the use of MatrixExplorer is the management of the various visualizations. MatrixExplorer is missing a way of controlling which visualizations are synchronized, providing a feedback to users. One idea would be to provide a node-link representation of the different visualizations and how they relate to the dataset. This problem also leads to the history mechanisms required to perform undo functions and understand the combination of actions applied to the visualizations.

Presentation of the results and storytelling After this case study, we realized that presenting results of our exploration is an important problem. Storytelling and fitting results either in an article, a poster or a talk raise a number of challenges. We believe this perspective should be handled by information visualization system and therefore, we reformulate our initial research problem to add a fourth stage:

1. Initiate the exploration;
2. Explore and collect insights;
3. Find a consensus;
4. **Present the results.**

During this case study, we mostly used matrix representations, visualizing small portions of the network as node-link diagrams occasionally. We originally thought that filtered node-link diagrams would be enough to present our results. However, it is not at all the case. We required other representations and we faced the problem of fitting a large matrix into an A4 paper sheet. We solved this problem by annotating an overview of the matrix *a posteriori* with drawing software, which required an additional effort for an unsatisfying result.

Figure 5.11 shows an example in which we aimed at representing groups and their connectivity within a global context (annotated matrix on the left) as well as details of one of them (submatrix on the right). From this experience, we learnt that *annotation* was a key feature within the exploration process and that providing support for *storytelling* would really benefits analysts. This experience also leads us towards a novel visualization technique to represent both details and context: NodeTrix. We present **NodeTrix** in Chapter 7.

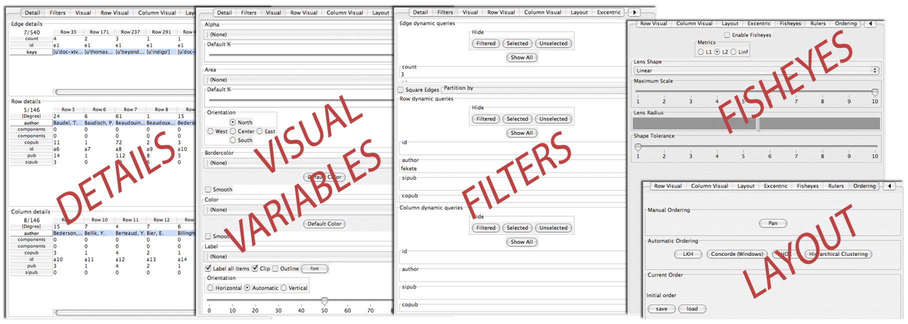


Figure 5.10: An example of MatrixExplorer control panels.

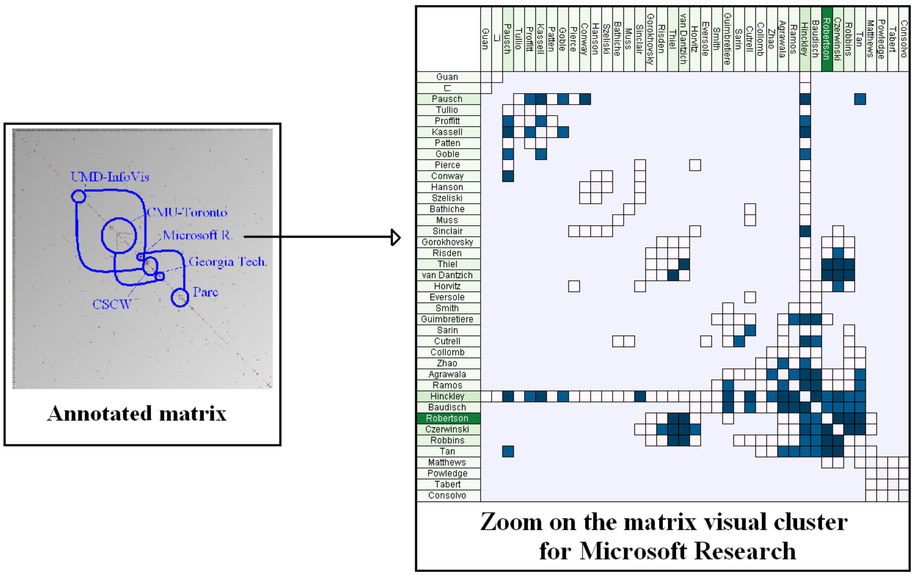


Figure 5.11: Manual *a posteriori* annotation of a matrix representation of the largest component of the coauthorship network of four conferences of HCI. The matrix on the left counts more than 2500 rows and columns.

MatLink and Mélange: augmenting matrix representations

CHAPTER 6

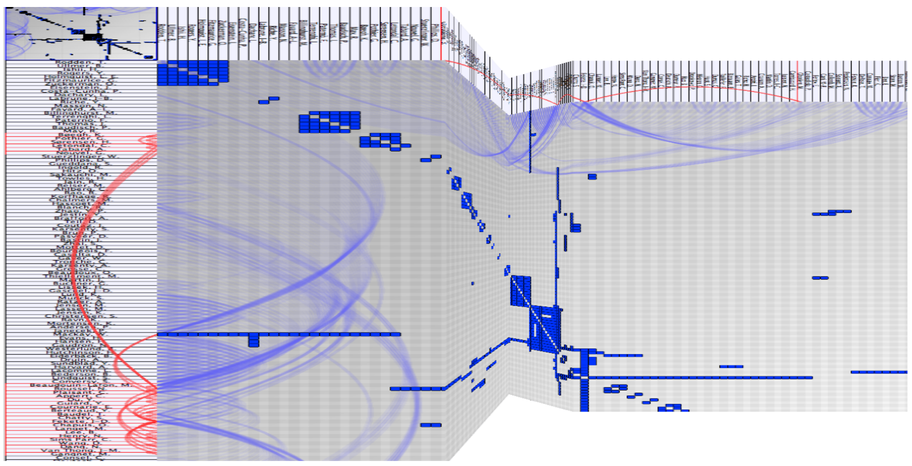


Figure 6.1: MatLink and Mélange

In the previous chapter we observed that combining matrices and node-link diagrams requires multiple screens for a comfortable use and that switching between the two representations could be cognitively demanding. Thus, we decided to select a single representation: matrices, and attempt to fix a maximum of its weaknesses. This chapter presents MatLink [HF07], an augmented matrix representation, and Mélange [EHRF08], an interaction technique to navigate through large matrices (Figure 6.1).

6.1 Research problem

Node-Link diagrams are familiar representations and effective with sparse or small size networks [GFC05]. However, they suffer from node overlapping and edge crossing when the density or the size of the network increases. Matrices overcome these major readability problems but they are difficult to manipulate for path-related tasks (such as how many persons connect A to B?). While these tasks are always possible to perform, they are tedious and require multiple readings from rows and columns. The second problem inherent to matrix representation is the large space required to draw them. In this case, performing path-related task is even more tedious as users need to scan a potentially very large amount of space. In this chapter, we attempt to answer these two problems:

➤How can we solve the weakness of matrices for performing path-related tasks?
➤How can we effectively navigate through large matrices (potentially performing path-related tasks)?

We first present MatLink, a visualization technique enhancing matrix representations to help performing path-related tasks. We describe a controlled experiment proving its performance over standard matrix and node-link representations. Then, we describe Mélange, a general purpose interaction technique to navigate through large visual spaces. We present a controlled experiment showing that Mélange better supports path-related tasks in very large matrices compared to standard techniques.

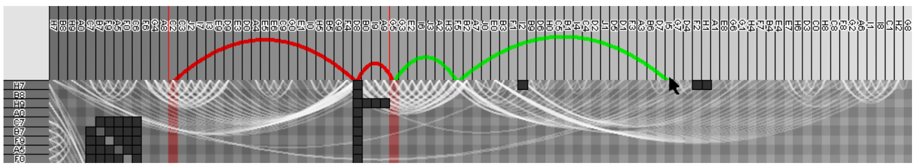


Figure 6.2: Performing path-related tasks with MatLink

6.2 MatLink

To address the weakness of matrix-based graph representations for path-related tasks, we designed MatLink(Figure 6.2) , a matrix representation with links overlaid on its borders and interactive drawing of additional links and highlighting of the cells included in a path from the cell under the mouse pointer. Recently, similar work has been done by Shen and Ma [SM07].

6.2.1 Design

Static links MatLink displays the full graph using a linearized node-link representation we call the full linear graph (Figure 6.3). Its links are curved lines drawn interior to the vertex displays at the top and left edges of the matrix similarly to arc diagrams [Wat02]. Links are drawn over the matrix cells, using transparency to avoid hiding them. Longer links are drawn above shorter ones. The linear graph conveys detailed and long-range structure together without hiding any detail of the matrix: a feeling for link densities and sub-graphs, but also paths and cut points.

Interactive links When the user has selected a vertex in the rows or columns, it is highlighted in red, and the shortest path between this vertex and the one currently under the mouse pointer is drawn in green on the vertex area, mirror-imaged to the links drawn in the matrix border[1]. This dynamic visualization of the shortest path is designed to make paths preattentively visible on the matrix. Early versions of MatLink drew these dynamic paths over the full linear graph, but users complained about visual complexity and difficulty seeing cells under the path links. This was not the case with paths drawn in the vertex area.

When several vertices are selected, their related rows or columns are highlighted in red, and the shortest path is visualized using red curved links in the vertex area. When the user moves the mouse pointer, the shortest path between the last selection and the vertex under the pointer is drawn in green (Figure 6.3).

Shortest path We display only one shortest path even if several equivalent ones exist (displaying multiple paths dramatically reduce the readability, making it very hard to identify the length of the path for example). To avoid confusion, we ensure that the same path is always displayed between two vertices. If the path from A to E contains C, then the same links will be displayed when only the path between A and C is visible. We considered also showing a shortest cycle within a subset of vertices but decided it would be confusing, because adding one vertex to the selection could completely change the links drawn. Moreover, when the selection grows, computing a shortest cycle requires noticeable time. Other analytical attributes could be displayed on the linear graph, statically or dynamically. In this article, we focus on the shortest path.

[1]A video is available at `http://insitu.lri.fr/~nhenry/matlink/matlink.mov`

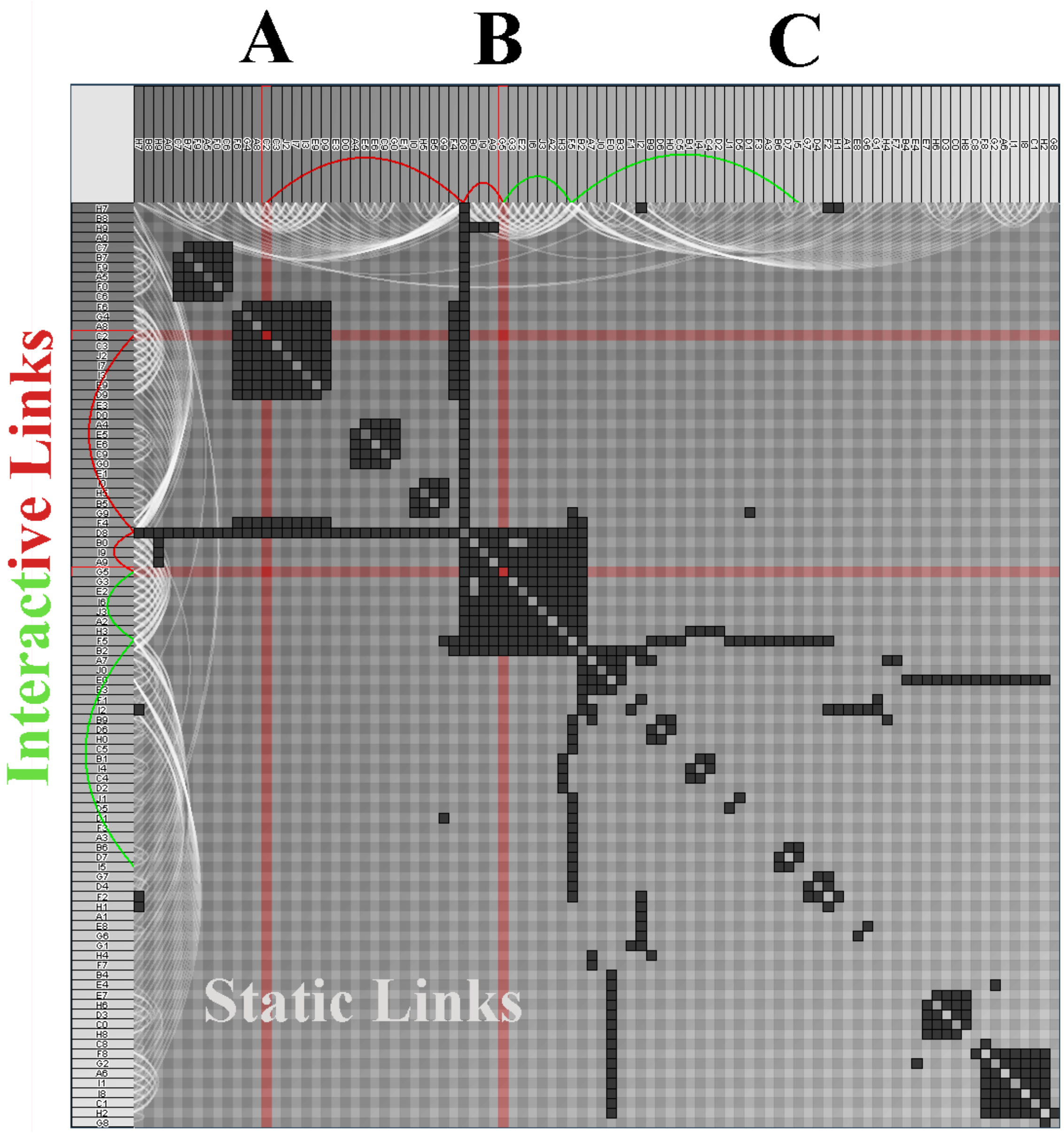


Figure 6.3: Illustration of a selection of two vertices and mouse over a third one. The shortest path between selected elements A and B is drawn in red, interactive shortest path appears in green on mouse over C.

Matrix ordering Displaying the linear graph of a randomly permuted matrix is useless and even confusing because they present many crossings. However, after the matrix is reordered using one of the reordering algorithm presented in chapter 4, such as a table-based ordering placing vertices with connection patterns beside each other or a graph-based seriation minimizing the total number of edge crossings of the linear graphs, the linear graph appears well organized. Short links connect nearby vertices, forming a lattice for dense clusters and cliques, and some long edges connect these clusters.

Displaying the linear graph is particularly important when visualizing a large network (with many vertices not visible on the screen). In this case, the linear graph reveals that some vertices, primarily belonging to one cluster have also links leaving the screen. These links represent the connection to vertices in other clusters. Finally, the linear graph seems to facilitate the understanding of the matrix representation when users are familiar with node-link diagram.

Information visualization MatLink is implemented with the InfoVis Toolkit [Fek04], so all the attributes of the network can be assigned as visual attributes such as color, or label for the vertices and the edges. An example of the high number of visual variables available is presented in Figure 6.4. (We do not recommend using them all but show the potential of MatLink).

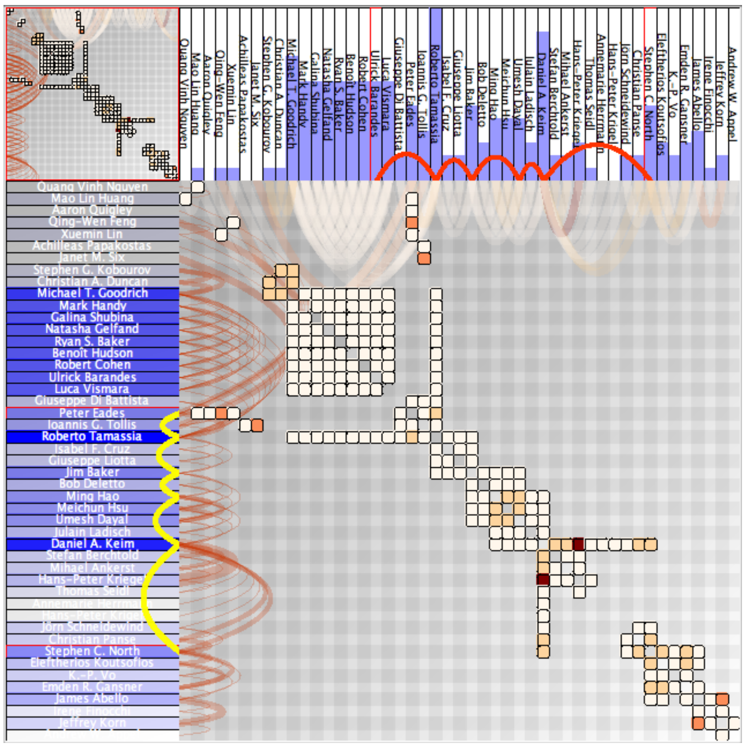


Figure 6.4: Many visual variables can be assigned to MatLink. In this example, we used the translucence in rows, areas in columns to show the degree of each actor. Colors in the cells of the matrix, width of the rows static links and color of the column static links show edge weight. Two different shortest paths are shown in red (columns) and yellow (rows).

6.2.2 Evaluation

We performed a controlled experiment to compare MatLink with traditional matrices (MAT) and node-link diagrams (NL). Our experiment was a within-subject design consisting of:

3 visualizations x **6** datasets x **5** tasks x **36** participants = **3240** trials

Datasets were divided in 2 densities and 2 sizes. Our independent variables were: *Vis*(MatLink, MAT, NL), *Density*(Sparse, Dense), *Size*(Small, Large) and *Task*(commonNeighbor, shortest-Path, mostConnected, articulationPoint, largestCommunity).

Tasks This experiment evaluated primarily mid-level readability tasks [HF06]. Low-level tasks have been evaluated by Ghoniem et al. [GFC05] for MAT and NL (whose paradigms MatLink combines) while performance on high-level interpretation tasks depends more on the domain and the subject's background, requiring a subjective and time-consuming evaluation.

The three most important high-level tasks in social network analysis [WF94]: evaluating connectivity, finding central actors and identifying communities. For each, a couple of formal measures exist, such as degree centrality for finding central actors. We evaluated performance on one or two of these measures, selecting the ones we could easily explain to novice users, and for which answers could be objectively validated.

Connectivity
T1. *commonNeighbor*: given two actors, find an actor directly linked to both;
T2. *shortestPath*: given two actors, find a shortest path linking them;
Central Actors
T3. *mostConnected*: find the actors with the highest number of relations;
T4. *articulationPoint*: find a cut point, i.e. an actor linking two sub-graphs;
Community
T5. *largestClique*: find the largest set of actors who are all linked to each other (a set with a couple of missing links was considered a clique).

Datasets Because existing social network generators did not provide realistic data (for details, see Chapter 8), we collected examples of real social networks from Pajek [dNMB05], UCINET [BEF99] as well as the Graph Drawing and the InfoVis contests. To select a representative subset of networks, we drew a portrait of each dataset's characteristics: vertex number (*Size*), edge number (*Edges*) and density (*Density*). Moreover, we used the clustering coefficient (*Clustering Coeff.*), to detect how clustered the dataset was. Average path length (*Aver. Path*) and degree distribution are two measures we considered to detect further small-world properties [WS98]. We present measures of *Aver. Path*, however, our networks were too small to clearly exhibit power-law degree distribution. Selected datasets are presented in Table 6.2.2.

We attempted to counterbalance *Size*, *Clustering Coeff.*, and *Density* (*Aver. Path* being too difficult to control). We excluded three kinds of graphs for which we considered the results obvious or already proven in Ghoniem et al. experiment [GFC05]:

· very small graphs containing less than two dozen vertices;
· small and very sparse graphs for which NL performs very well;
· very dense graphs for which MAT performs very well.

Datasets	*Size*	*Edges*	*Density*	*Clustering Coeff.*	*Aver. Path*
Infovis	47	114	0.23	0.83	3.84
Dolphins	47	202	0.30	0.42	3.49
Fraternity	47	294	0.36	0.72	2.36
Genealogy	94	192	0.15	0.59	6.00
Collaboration	94	313	0.19	0.90	2.80
USairports	94	990	0.33	0.84	2.69

Table 6.1 : : Datasets used for our experiment. We balanced Size (*Size*), Density (*Density*) and Clustering Coefficient (*Clustering Coeff.*).

To avoid scrolling and navigation issues, we chose to limit dataset sizes to what all three representations could display on one screen, slightly less than a hundred nodes. To match the selected graphs in size, we produced a set of randomly filtered graphs (composed of a single connected component) for each instance and selected the filtered graphs with *Density* and *Clustering Coeff.* properties similar to the originals. We relaxed the constraint on *Density* when shrinking large graphs, as it is difficult to keep a low *Density* with a high *Clustering Coeff.* when filtering it.

Experimental setup Adding interactive highlighting can affect the performance of visualizations. For example, finding the shortest paths between two actors can become preattentive with highlighting. For these reasons, we decided to augment both MAT and NL with interactive highlights to run a fair comparison with MatLink.

Subjects were limited to three interactions: mousing over an element, clicking on an element, and dragging on several elements using the left mouse button. Clicking on a row or column header selected one element, indicated by displaying it in red. Similarly, dragging the mouse over a group of elements selected them all. Only vertices were clickable, not edges. In addition, the space bar was used to start and terminate each trial.

We drew NL diagrams using the LinLog algorithm [Noa05], a recent spring layout algorithm showing good results. To reorder matrices, we chose the augmented TSP algorithm described in [HF06a]. We were careful to maintain readability of vertex labels in all conditions. To minimize extraneous interpretation issues, we anonymized all actors. To reduce memorization effects, we assigned different labels when presenting the same graph twice. We used the same node sizes for small and large graphs. Since NL visualizations are usually more compact than MAT, and thus use less space, we considered it fair to use the space by enlarging their nodes, potentially increasing their readability.

Subjects and Apparatus We used a total of thirty-six subjects, divided into two distinct groups of eighteen. The first group was composed of students and researchers from the university of Sydney, mostly from a graph drawing research group. The second group consisted of students and researchers from the university of Paris-Sud, primarily specialists in HCI. Only one subject in each group was familiar with matrix-based representations. Both groups used a 3GHz Pentium IV computer with 1GB of RAM and one 19" screen oriented vertically. The screen was divided in two parts: the upper part displayed the question and the lower the visualization.

Procedure Before starting the experiment, subjects were briefly interviewed to gather information about their previous experience with graphs and visual representations. A tutorial sheet introduced the visualizations, and an experimenter demonstrated the experimental environment, how each representation worked and how to complete the tasks. Subjects could then practice with the program for a few minutes. The training ended when the subject answered all questions correctly for all representations. Subjects spent an average of forty minutes on the training, usually longer than they did on the experiment itself.

The rule for the experiment was to answer correctly as rapidly as possible. If the subjects felt unable to answer a question, they were allowed to skip it. Each subject had a total of 90 questions to answer: 5 tasks performed on 3 visualizations of 6 graphs. To limit the experiment duration, the system limited the completion time (*Time*) to 60 seconds. To limit the subject fatigue, we split the experiment in two sessions with a ten-minute break in between. Moreover, subjects could rest after any question by delaying the start of the next trial.

Sessions were not balanced to limit the learning effort: the first session presented small datasets, while the second presented large datasets. Sessions were split into three blocks, one for each representation. The order of the representations was complete and counterbalanced across subjects. Each representation block was split into three blocks of three datasets (small for the first session, large for the second) counterbalanced across subjects using a Latin square. We alternated the order of representations to reduce memorization effects: subjects remembering the answer from the previous representation and dataset. However, we kept the order of datasets constant for each session and counterbalanced across subjects.

At the end of the experiment, subjects had to answer a questionnaire about their use of each representation, rating them by task and preference. Finally, a debriefing was conducted to answer any remaining questions and collect their final comments.

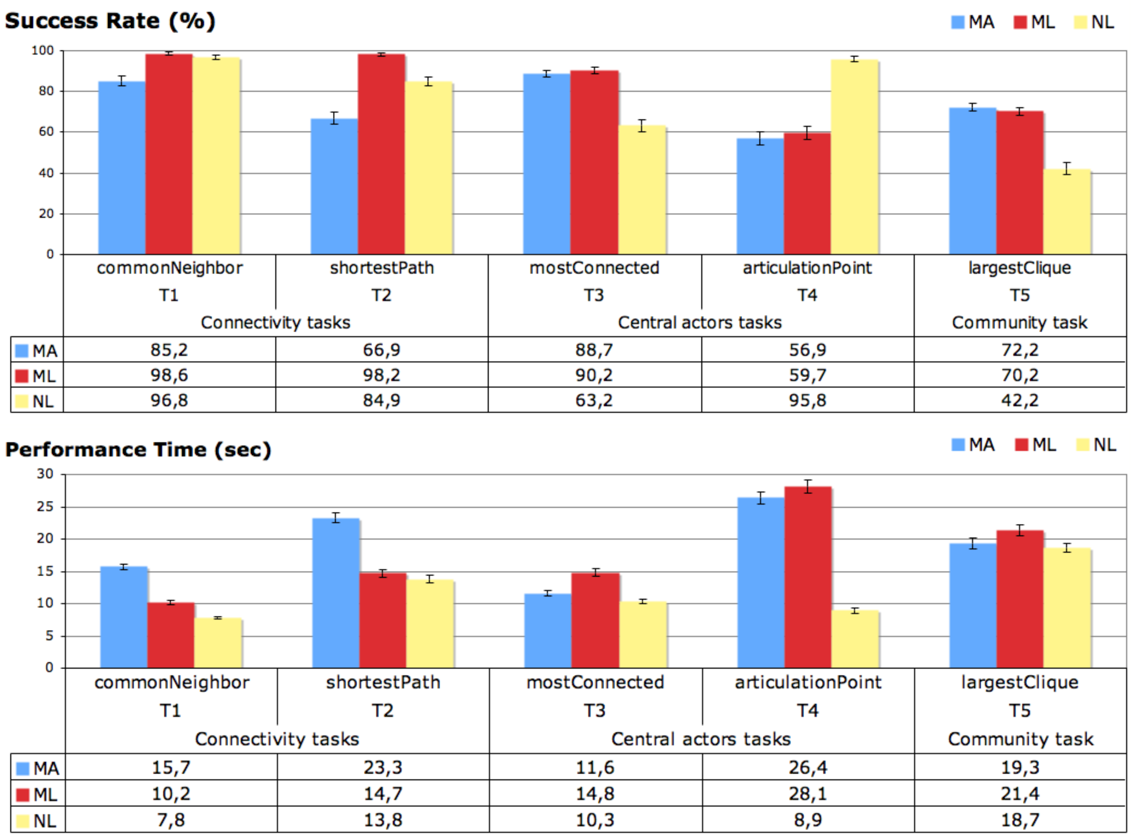
Data collected Our primary measure was answer correctness. We scored answer correctness as a percentage, with 0 meaning error and 100 the best answer. For example, analysts exploring a large network are interested in finding primary actors, but while finding the most important rated a 100, other main actors got partial credit. We consider *Time* as a secondary measure: a fast wrong answer is not useful for data analysis. Therefore, the analysis of *Time* is performed with correct answers only, wrong answers being replaced by the mean time of correct ones.

Results

We performed the Friedman's non-parametric test and Wilcoxon signed-rank test on score data. We used an analysis of variance (ANOVA) for analyzing the performance time followed by Tukey's HSD Post Hoc test(TT) for paired comparison. Parametric tests such as ANOVA assume the distribution of the data is normal, which is generally the case for the completion time. However, for ranks or scores, this hypothesis may not apply. In this case, ANOVA might present erroneous p values. Thus, it is safer to apply non-parametric tests, as these tests require a stronger effect to discover it but are more robust.[2]

Results for score and performance time per visualization for the 5 tasks are illustrated in Figure 6.5. Results of the ANOVA are presented in Table 6.2.

[2]In our MatLink paper [HF07], we presented ANOVA results for both *Score* and *Time*, as we were not aware of the subtility at this point. In this chapter, we chose to present the non-parametric tests results to ensure of the reliability of our findings. Effects remain significant.



Success Rate (%)

	commonNeighbor T1	shortestPath T2	mostConnected T3	articulationPoint T4	largestClique T5
	Connectivity tasks		Central actors tasks		Community task
🟦 MA	85,2	66,9	88,7	56,9	72,2
🟥 ML	98,6	98,2	90,2	59,7	70,2
🟨 NL	96,8	84,9	63,2	95,8	42,2

Performance Time (sec)

	commonNeighbor T1	shortestPath T2	mostConnected T3	articulationPoint T4	largestClique T5
	Connectivity tasks		Central actors tasks		Community task
🟦 MA	15,7	23,3	11,6	26,4	19,3
🟥 ML	10,2	14,7	14,8	28,1	21,4
🟨 NL	7,8	13,8	10,3	8,9	18,7

Figure 6.5: Score and Performance time per visualization for the 5 tasks.

		Task 1		Task 2		Task 3		Task 4		Task5	
Performance ***Time***											
Vis	F(2,70)	83.4673	***	56.4941	***	17.0372	***	206.8297	***	5.9018	**
Size	F(1,35)	11.2244	***	218.5899	***	14.8107	***	14.7163	***	30.9148	***
Density	F(1,35)	31.0351	***	93.5538	***	36.2844	***	248.3838	***	152.5933	***
Vis**Size*	F(2,70)	1.0601		8.0762	***	0.8878		4.4137	*	3.3366	*
Vis**Density*	F(2,70)	0.207		0.963		5.2016	**	49.5183	***	0.3993	

$***p < .0001$ $**p < .01$ $*p < .05$

Table 6.2 : : ANOVA for each Task for *Vis*, *Size*(Size), *Density*(Density) and their interactions.

Task 1: Connectivity (commonNeighbours)

H: We predicted that for both connectivity tasks (commonNeighbor and shortestPath) MatLink Score would be better than MAT, and that it would not be affected by Size and Density. We also expected that MatLink would be faster than NL.

***Score*:** The Friedman's chi square test revealed a significant effect on *Vis*$(p < .001)$. Pairwise comparison using the Wilcoxon's test showed that MatLink was significantly better than MAT and NL. NL provided also significantly better scores than MAT. *Density* and *Size* had no significant impact on the visualizations. MAT and MatLink tended to have better scores when the *Density* increased, while NL was negatively affected.

***Time*:** ANOVA revealed a significant effect on *Vis*, *Size* and *Density*. As expected, TT showed that MatLink was significantly faster than MAT but not from NL. All visualizations were affected by *Size* and *Density*, MAT being particularly affected by *Size*.

Preference: Most of the users preferred MatLink.

Task 2: Connectivity (shortestPath)

H: We predicted that for both connectivity tasks (commonNeighbor and shortestPath) MatLink Score would be better than MAT, and that it would not be affected by Size and Density. We also expected that MatLink would be faster than NL.

***Score*:** The Friedman's chi square test revealed a significant difference between *Vis*$(p < 0.001)$. As expected, the Wilcoxon's test showed that MatLink outperformed both MAT and NL for this task. NL still provided better scores than MAT. MAT was particularly affected by *Size* and *Density*. Surprisingly, NL provided higher scores for higher *Density*.

***Time*:** ANOVA revealed a significant effect on *Vis*, *Size* and *Density*. As expected, TT showed that MatLink was significantly faster than MAT but not from NL. *Time* increased for higher *Size* and *Density* for all three visualizations.

Preference: Most of the users preferred MatLink.

Task 3: Central Actors (mostConnected)

H: We expected MatLink to perform as MAT both in term of Score and Time. We expected MAT to be faster than NL and not affected by Size or Density.

***Score*:** The Friedman's chi square test revealed a significant effect of *Vis*$(p < 0.001)$. MatLink and MAT produced significantly better scores than NL. *Density* had a negative impact on scores for this task, NL being the most affected (from mean scores of 80% for spare graphs to less than 50% for dense ones).

***Time*:** ANOVA revealed significant effects for *Vis* and *Density*. TT showed that MatLink was slower than both MAT and NL. There was no significant difference between MAT and NL. *Time* increased with *Density* and MatLink was particularly affected by it (from a mean of 12s for sparse graphs to 17s for dense ones). *Size* had no significant effect on the visualizations.

Preference: Most of the users preferred MAT. Several subjects report that MatLink lacked a highlight or mouse-over effect on the core of the matrix (it only provided feedback on the interactive links) to ease the comparison of actors, especially to help them counting the number of edges in the matrix row/column.

Task 4: Central Actors (articulationPoint)

H: We predicted that NL would outperform MatLink and MAT both in term of Score and Time. We expected MatLink to show higher Scores than MAT especially for sparse graphs.

***Score*:** The Friedman's test revealed a significant effect of *Vis*($p < 0.001$). As predicted the Wilcoxon's test showed that NL outperformed both MAT and MatLink. However, it did not show a significant difference between MAT and MatLink. Moreover, MAT and MatLink were especially affected by *Density* (from a mean success rate around 80% for sparse graph to less than 40% for dense ones).

***Time*:** ANOVA revealed that *Vis*, *Size* and *Density* had a significant impact on the *Time*. TT confirmed that NL significantly outperformed MAT and MatLink for this task.

Preference: All subjects except two preferred NL for this task. Their level of confidence was very high and their performance good with this representation. Almost all subjects report that this task was the most difficult one when dealing with MAT and dense graphs. Several subjects commented that MatLink was slightly better for sparse graphs.

Task 5: Communities (largestClique)

H: We expected MatLink to perform slightly better than MAT in terms of Score. We also predicted NL would present low scores for dense graphs. We did not expect any significant difference in Time.

***Score*:** The Friedman's test revealed a significant effect of *Vis*. The Wilcoxon's test showed that MAT outperforms both NL and MatLink, and MatLink outperformed NL. NL was dramatically affected by *Density* (from a mean success rate around 65% for sparse graphs to 20% for dense ones).

***Time*:** ANOVA revealed a significant difference by *Vis* and *Density*. TT showed that MatLink was significantly slower than NL and that all three visualizations were negatively affected by a high *Density*. The interaction *Vis*Size* was significant: MatLink and MAT are slower when *Size* increases.

Preference: Almost all subjects preferred MAT or MatLink for this task. They comment that they had a much higher level of confidence identifying if a visual cluster was a clique or not. However, several added that finding the largest clique was difficult without reordering the MAT to place sub-parts next to each other. Most reported that links were not useful, but several argued that links helped them find a member of the clique distant from the others.

General preference Overall, for both group and all tasks, 18/36 users preferred MatLink, 13/36 preferred NL and 4/36 could not decide between MatLink and NL. All users spontaneously reported that they liked MatLink at first sight. After the experiment, they commented that they preferred NL for sparse networks and MatLink for dense networks. Most commented that MatLink was a good compromise for all tasks and graphs except for the articulationPoint (Task 4). We noticed a difference between groups. Subjects with a graph drawing background showed equal interest in NL and MatLink (8/18 MatLink, 8/18 NL) whereas those from the HCI field preferred MatLink (11/18 MatLink, 5/18 NL).

Discussion

Connectivity For the social networks we selected, users were significantly more accurate with MatLink than with MAT for connectivity tasks. Although NL is usually considered more accurate than MAT for these path-related tasks, MatLink proved to be competitive with NL, performing as well as NL for identifying a common neighbor and outperforming it for finding the shortest path, even when the path was highlighted in NL. A surprising detail is that users had fewer errors with NL on dense graphs than sparse ones. This appears to be an artifact of the highlighting: users ignored the highlighting information in NL when the network was sparse and thus missed the shortest path, whereas they systematically used highlighting on dense graphs.

Central Actors Our results confirm the results of the Ghoniem et al. experiment, showing that MAT outperforms NL for finding the most connected actor. As predicted, users were also more accurate using MatLink than using NL. However, surprisingly, the completion time was longer for MatLink than MAT. Our interpretation is that the added overhead of drawing the auxiliary visualizations created a small time lag that impacted this task. Moreover, compared to MAT, MatLink did not provide feedback on the cells of the matrix (only on its headers) on mouse over or selection. Several users complained about the lack of this feature when they clicked near but not on the most-connected vertex. The articulation point was the only task for which NL outperformed both MAT and MatLink. Our explanation is that finding such an actor was easier in 2D as it appears on the periphery of the node-link diagram, only connected to the rest of the graph by a single link. In the matrix-based representations, articulation points were placed in the middle of rows and columns, and thus more difficult to find.

Community MAT and MatLink produce better scores than NL for finding the largest clique. We used the best layout for each representation, exhibiting blocks in matrices and groups densely connected in NL. In matrix-based representations users found communities quickly and commented that it was easier to estimate their size and find out if they were clique or not. MatLink produce slower results as users attempted to identify communities using the static links first.

Conclusion

From our results, we can conclude that MatLink solves the weakness of traditionnal matrices for connectivity tasks such as finding the shortest path. While we limited the size of the datasets to fit the screen (for experiment purposes), we also used MatLink on datasets far larger and discovered an additional benefit of this techniques.

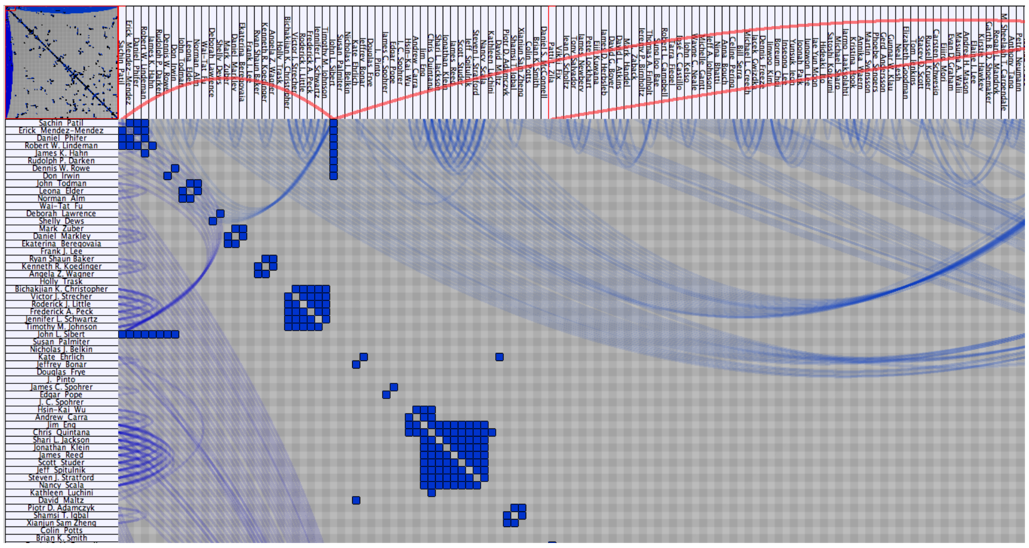


Figure 6.6: Coauthorship of 20 years of CHI (1400+ authors). This view of the matrix shows only a very small portion of the entire matrix (less than 10%). The whole overview is on the top left corner. On the columns headers, the user selected two actors, the shortest path highlighted in red is composed of an actor out of the screen.

Exploring large matrices involves navigating through a large visual space showing a small portion of the matrix at a time. We noticed that MatLink was particularly effective to convey information about the connectivity of visible actors to actors out of view (Figure 6.6). Observing links leaving the viewport gives an indication on potential important actors outside the view (estimating how far they are using the amplitude of the links). This information is impossible to perceive on a standard matrix.

In SNA, understanding why two connected actors are placed far away in the matrix is important. It can reveal how two different communities are linked or simply show an artefact of the matrix ordering. In both cases, it is required to go back and forth to compare the neighborhood of the distant actors. The strength of MatLink is to support the discovery of these cases. However, it remains tedious to navigate from neighbor to neighbor. To solve this problem, we had the idea to use the links to travel from one place to the other, folding the space between them to be able to see both departure and arrival point at the same time.

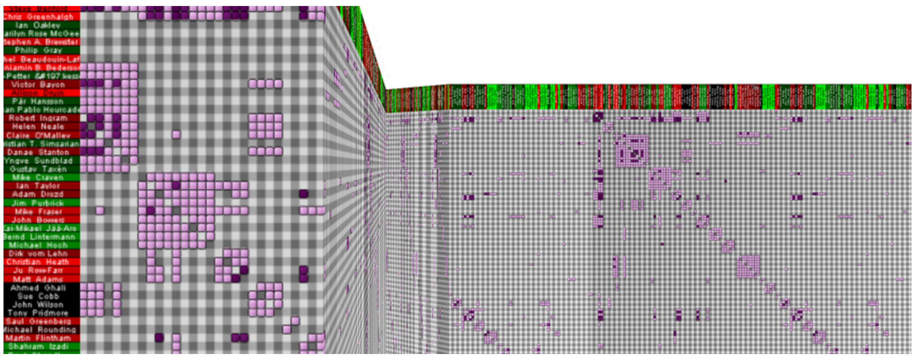


Figure 6.7: Mélange

6.3 Mélange

In this section, we present Mélange, an interaction technique we designed with members of the AVIZ and InSitu teams, to navigate through large visual spaces. Mélange is a space deformation technique that folds 2D space into 3D in order to bring several regions of interest into focus while preserving the awareness of the intermediate context (Figure 6.7). We describe a controlled experiment comparing our techniques to standard split-screen and pan and zoom techniques for performing path-related tasks in large matrices.

6.3.1 Design

When exploring a large matrix, it is often important to see several parts of it at the same time. As we explained in the previous chapter, using MatLink, it is important to identify how communities are connected and why two neighbors may be placed far away from each other. This task requires several concurrently visible focus points and, depending on the size of the communities, analysts may want to adapt the magnification of each focus region independently. Moreover, awareness of the intermediate context (content and quantity of space) can be precious information to understand the network structure. Based on this example, we formulated a number of design goals:

G1. *guaranteed focus visibility*: multiple foci at the desired zoom level should be visible simultaneously, regardless of their location on the space;
G2. *surrounding context visibility*: as much as possible of the area surrounding each focus region should be visible;
G3. *intervening context awareness*: the space between focus regions should be shown to give a frame of reference; and
G4. *distance awareness*: some notion of the distance between the focus regions should be available.

To fulfill these general goals, we designed Mélange, a technique that automatically folds intervening space between focus regions to guarantee their visibility. This work is issued from a collaboration with Niklas Elmqvist and Yann Riche. I contributed to the design of the technique and performed part of the experiment (design and analysis of the results).

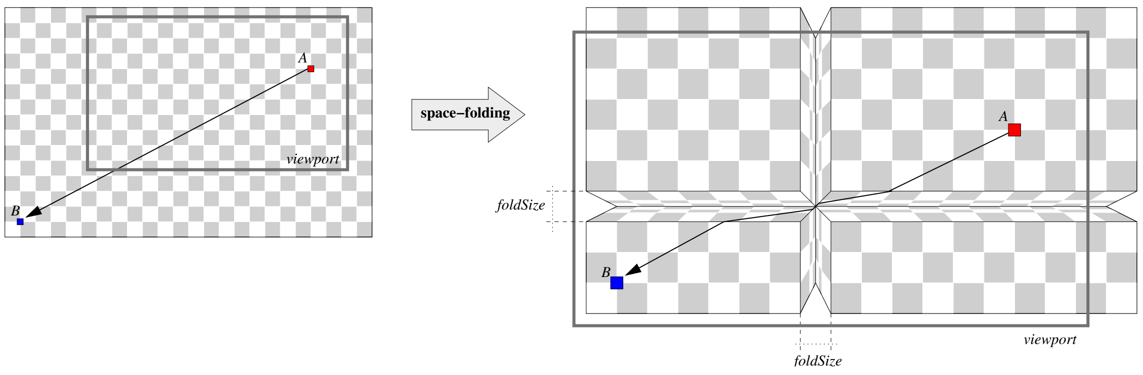


Figure 6.8: Folding a 2D space with two focus points A (main) and B. The space is folded to make best use of the available area in the viewport. Focus points can be independently zoomed by changing their 3D depths.

Multiple Foci: Guaranteed Focus and Context Visibility Given a set of focus points and the location and extents of the current viewport on the canvas, the objective of the Mélange technique is to combine different parts of the visual space so that the focus points and as much as possible of their surrounding context are visible on the user's screen. Thus Mélange provides *guaranteed focus visibility* (G1) and *surrounding context visibility* (G2). Focus points are specified as 2D positions on the visual space, and also have an associated depth parameter that allows each point to be zoomed independently of the others. This supports interactions where different parts of the visual space must be viewed at different scales, such as a social scientist studying a particular actor in relation to a larger clique of actors on a matrix representation of a social network.

Folding Space: Intervening Context Awareness A split-screen approach to multiple foci would remove space outside of the focus regions and show each region as small subwindows in the main viewport. Mélange instead *folds* the space into the negative depth dimension (i.e. into the screen, see Figure 6.7). If there is no extraneous space to fold away, the space is instead stretched, similar to the rubber sheet [SSTR93] but with support for independent depths for each focus point. The folds themselves are shown in 3D perspective as they stretch away into the depths of screen, and they also indicate the relative positioning of the focus points. Thus, Mélange provides *intervening context awareness* (G3). Furthermore, the mechanism gives a tangible and compelling metaphor for the user that is close to how real paper or fabric is folded. We believe that this metaphor is easier to understand than merely compressing the space, as in rubber sheet-inspired models.

Figure 6.8 shows a schematic overview of the folding process. The user's viewport (denoted by the smaller rectangle in the left part of the figure) is centered on the focus point A—the main focus—but the user has also designated a second focus point, B. Given the available space in the viewport, the Mélange technique folds away some of the intervening space below and to the left of A to also bring B onto the screen. All folds are rectilinear to simplify understanding of the deformed space. A certain amount of screen real estate (*foldSize*) is used to show the contents of the folded space in 3D perspective as it stretches away into the depths of the screen. These regions serve as context between the focus regions. The above method generalizes to any number of additional focus points. One of the foci is always designated as the main one and is used as a baseline for computing the size allocations for the others.

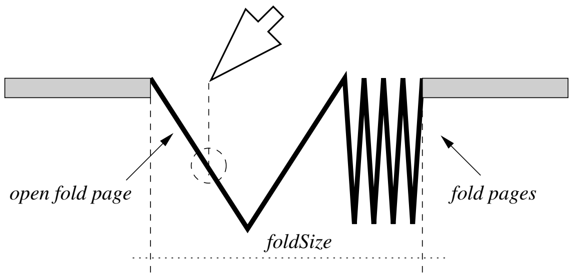


Figure 6.9: Fold pages for conveying a sense of distance between focus regions.

Interacting with Folds: Context and Distance Awareness Deforming the space to bring several foci onto the screen may give the user an incorrect idea of the size of the visual space. For example, folding a world map to bring London and New York into focus at high detail level will certainly convey a false sense of the distances between the two cities. Mélange supports better *distance awareness* (G4) than compression-based techniques like the rubber sheet method [SSTR93] since the 3D perspective of the folds gives an indication of the distance between the regions.

To further improve distance awareness, we introduce fold pages and interaction techniques for flipping between them. The folded space is split by a suitable and tangible unit, such as the size of the screen. Only one such unit is shown at full detail, and the rest are shown as thin fold pages (Figure 6.9). Each fold page represents one screen of compressed space. Mélange provides distance awareness (G4) by allowing the user to quickly estimate the number of fold pages to find the distance between the focus points (like estimating a book's length from its thickness). Another benefit is that context awareness is improved by allocating more screen estate to each individual fold page (although some overview is lost). Pages could potentially also show condensed context information on its one-pixel representation, akin to the compact contextual views of the City Lights [ZMG$^+$03a] technique. Hovering with the mouse over the pages flips through them like leafing through a book. Furthermore, clicking on a fold adds a focus point on the designated location, and double-clicking removes all of the other focus points and creates a new primary focus point at the position. The effect is that the user stops folding space and travels to the new location.

Fold geometry The Mélange space-folding mechanism is different to most Focus+Context techniques in that it compresses uninteresting space as opposed to expanding the focused space. To fully support the metaphor of folding paper or fabric, the space should probably be folded in a smooth curve. However, this would cause most screen estate to be afforded to the middle region of the compressed space. Most often, the space closer to a focus region is more important than the space halfway between regions. Therefore, in our realization, the folds are sharp and angular (more like paper origami than fabric folding), similar to a multi-focus Perspective Wall [MRC91b]. 3D perspective foreshortening gives a form of fisheye effect on the contents of the folds.

Perspective correction When rendering the visual canvas and the folds in 3D, we must correct for perspective to get a correct visual appearance for the folds. Otherwise, the perspective projection of the 2D space deformed into 3D causes uneven distribution of screen space. Carpendale [CM01] calls this *folding* a region over other regions, unrelated to our use of the term. We solve this by performing all layout in the 2D screen space, and then unprojecting to 3D space.

6.3.2 Evaluation

We performed a controlled experiment to evaluate whether the Mélange technique assists users in exploring large visual spaces by comparing it to single and split-screen viewports. We designed the experiment to test our design goals in the context of a matrix visualization of a large graph with MatLink [HF07] arcs connecting relevant nodes in the graph.

Participants We recruited 12 unpaid subjects (1 female, 11 male) for our study. The participants were from 20 to 35 years of age, had normal or corrected-to-normal vision, and were screened to not be color-blind. No specific skills were required other than basic computer experience.

Apparatus The experimental apparatus consisted of an Apple iMac Core 2 Duo 2.33 GHz workstation with 2 GBs of memory and equipped with a standard two-button mouse (with wheel) and keyboard. The 21-inch display was fixed at 1680×1050 resolution and powered by an ATI Radeon X1600 with 256 MB of video memory.

Tasks A common task in social network analysis is to compare the local neighborhood of two actors to find similar patterns of connectivity. Therefore, participants were given a source node and its neighborhood on an adjacency matrix representation of a social network, and were then asked to perform three tasks in sequence:

T1. *Guarranted focus visibility (G1) and surrounding context visibility (G2)*
Find one destination node connected to the source node with the same neighborhood.

T2. *Distance awareness (G4)*
Estimate the distance between the source and destination nodes (in 1:1 screen units).

T3. *Intervening context awareness (G3)*
Estimate the number of contextual targets between the source and destination nodes.

Targets Targets in our study were blue squares measuring 20 pixels (at 1:1 zoom level), surrounded by a neighborhood of four half-size (10 pixel) squares of different colors (Figure 6.10). We chose five colors for these neighborhood squares: white, magenta, orange, green, and blue (a selection that is preattentively perceptible [Hea96]). Neighborhood nodes were placed in a 5×5 grid around the blue rectangle, and whole targets were placed in one line on the visual space, like columns in a matrix visualization. Targets were identical if both the position and color of their neighborhood nodes were identical. Only one other target neighborhood matched the source target, others were distractors. Connections between the source node and the potential targets were visualized using MatLink arcs. Not all nodes on the visual space had a MatLink arc from the source node; those without were background nodes that also served as distractors, and participants were instructed to disregard them when looking for the destination target.

Contextual targets Contextual targets (T3) were red squares six times the size of primary targets (i.e. 120 pixels) and below the line of primary targets. The motivation for this was that being aware of intervening context is only important for large-scale features such as mountain ranges or large bodies of water on a map, or communities of actors in a social network.

Visual space All targets on the visual space—i.e. target nodes, neighborhood nodes, and contextual targets—were guaranteed to be rendered with at least a single pixel, forcing them to be visible even if the view was zoomed out or distorted. The visual space itself was represented by a checkered gray rectangle that was 30 screens wide and one screen high. Each scenario had randomly-generated distractors. The source node was always located on the left edge of the rectangle, so that participants would always have to pan right to find the target. The view was initialized to center on the source node at 1:1 zoom level for every new scenario (started by T1), and was then left in its previous position for each consecutive task (T2 and T3). Finally, to give users a frame of reference for distance, screen units were indicated on the visual space by black lines drawn on the checkered gray rectangle. Figure 6.10 shows a screenshot of our experiment application.

Experimental Design We performed a within-subject design experiment consisting of:

> **3** techniques x **3** distances x **2** distractor densities x **2** contextual target densities
> x **2** trials x **12** participants = **864** trials.

Presentation Technique The primary objective of our experiment was to study the performance of different presentations of the visual space for supporting our design goals. In addition to the Mélange technique, we included single and split-screen viewport conditions for comparison. While none of these two fulfill our design goals, they are commonly used in practice, suggesting that they are suitable competitors. We considered comparing our technique against Accordion Drawing [MGT+03]. However, AD does not seem to support independently zoomed foci. Furthermore, Nekrasovski et al. [NBM+06] have shown that pan and zoom for a large hierarchical dataset is more efficient than navigation in AD spaces, hence our choice of competing techniques.

· *Single viewport (SV).*
  The standard baseline consisting of a single window showing a view of the visual space. SV has no direct support for any of our stated design goals. Participants must use a lot of interaction.

· *Split-screen viewport (SSV).*
  The main viewport is split vertically into two equal-sized subwindows, each showing a different view of the visual space. In our setup, the left subwindow was fixed to always show the source node at 1:1 zoom, while the user could interact with the right one.

· *Mélange (M).*
  Our space-folding technique with the primary focus point on the source node and the secondary point controlled by the user. Moving the focus point (in the horizontal and depth dimensions) thus caused the visual space to be folded to accommodate both focus points in the viewport. Fold pages were disabled to not unfairly give a direct distance measure to the participants (i.e. only the 3D perspective foreshortening of the folds indicated distance).

Interaction All three techniques were controlled using standard zoom and pan operations. Dragging the mouse while clicking the left mouse button caused horizontal movement of the focus point (the camera for single viewport, the right subwindow for split-screen, and the folding focus point for Mélange). The focus could be zoomed in and out by dragging with the right mouse button, or by spinning the mouse wheel.

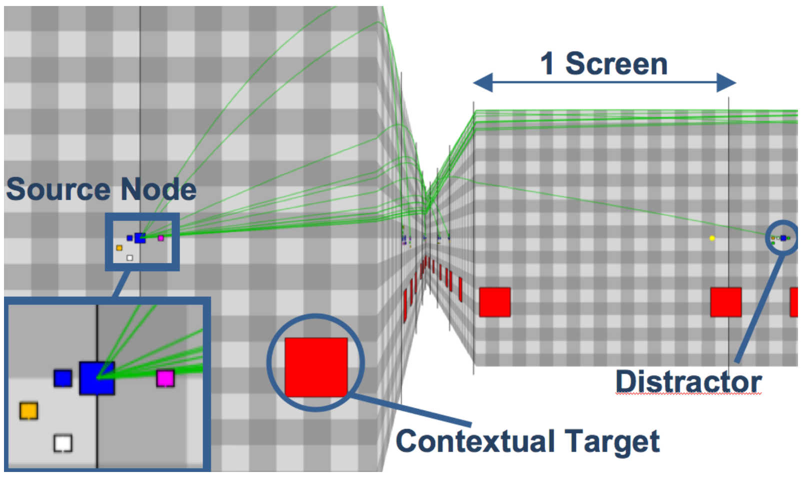


Figure 6.10: Screenshot from the user study application.

Off-Screen Distance We wanted to see whether performance varied with the distance to traverse on the visual space, so we tested three different distances: 4, 8, and 16 screen widths of distance (in our experimental setup, the screen width was 1680 pixels). In a matrix representation, this corresponds approximatively to networks containing 400, 800, and 1600 actors.

Distractor Density The number of false targets (i.e. distractors) between the source and destination nodes will clearly affect the time spent finding the destination node (T1). Thus, we included two different densities: *low* and *high*. This corresponded to one or two potential targets per screen (half of them background nodes with no MatLink arcs to them).

Contextual Target Density We studied two levels of density for the contextual targets between the source and destination nodes: *few* (less than or equal to five) or *many* (more than five).

Procedure Participants were introduced to the study and randomly assigned to one of the six order groups for the presentation technique. They then performed three blocks of trials, one per technique, in succession. Before each block, the test administrator explained how to use the technique and then let the participant practice on six training trials. Participants were not allowed to proceed past each training trial without answering correctly to all three tasks.

The order of the techniques was counterbalanced. Participants were asked to complete 3 blocks—one per technique—of 24 trials in randomized order. Each trial consisted of performing the three tasks T1 to T3 in sequence. A screen with instructions was given prior to each task, and the participant proceeded to the task by clicking a button or pressing the space bar. Task T1 ended when the participant clicked the right target (which then turned from blue to yellow); for the other tasks, the participant pressed the space bar to end the task. After task T2 and T3, participants were presented with a multiple-choice question asking about their answer to the task.

Participants were instructed to work as quickly as possible. For every trial, the software silently collected the time and correctness measures for the three tasks (only time for T1). Participants were instructed to pause between each block to avoid fatigue. At the end of the test, they were given a post-session questionnaire asking to rank the techniques.

Predictions We made three predictions:

· *P1: Mélange is as fast as single or split-screen viewport*
We believe that the space-folding technique will not introduce significantly slower completion times for standard visual search (task T1). In other words, we think that the added visual complexity and space allocations of the fold region and the additional focus point will not cause slow-downs for a user trying to locate a specific target on the canvas.

· *P2: Mélange provides more efficient context awareness*
None of the two techniques we compare Mélange to support contextual views explicitly, but participants are nonetheless exposed to this context when navigating over the visual space. We submit that the intervening context shown in the fold regions of the technique will cause significantly lower completion times for tasks T2 and T3.

· *P3: Mélange provides more accurate context awareness*
Analogously to P2, we also believe that participants will be more accurate when answering contextual tasks (T2 and T3) with Mélange than the other two presentation techniques. Mélange provides an integrated overview of the context, whereas the other two require the user to manually pan and zoom around in the space to discover this information.

Results

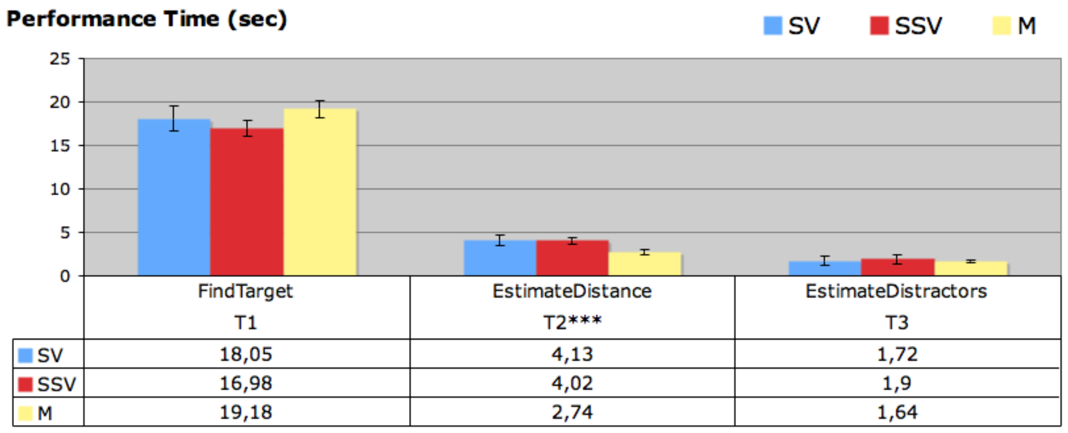
Completion Time Table 6.3 summarizes the main effects for time. Figure 6.11 shows mean time to completion for all three tasks.

T1. The average completion time was 18.05 (s.d. 1.42) seconds for SV, 16.98 (s.d. 0.85) seconds for SSV, and 19.18 (s.d. 0.99) seconds for M (SSV $<$ M $<$ SV). A repeated-measures analysis of variance (ANOVA) showed no significant main effect of Presentation technique.
T2. The average time was 4.13 (s.d. 0.64) seconds for SV, 4.02 (s.d. 0.43) seconds for SSV, and 2.74 (s.d. 0.35) seconds for M (**M** $<$ **SSV** $<$ **SV**). ANOVA yielded a significant main effect for Presentation technique ($F_{2,22} = 9.203, p = .001$).
T3. The average time was 1.72 (s.d. 0.57) seconds for SV, 1.90 (s.d. 0.50) seconds for SSV, and 1.64 (s.d. 0.19) seconds for M (SV $<$ M $<$ SSV). ANOVA yielded no significant main effect for Presentation technique.

Correctness For task T2, the average correctness was 98.6 % (s.d. 0.7) for SV, 94.8 % (s.d. 1.3) for SSV, and 98.3 % (s.d. 0.8) for M (SV $>$ M $>$ SSV). This is a significant difference (Friedman test, $p = .008$). A Wilcoxon test for paired comparison shows that M and SV have higher correctness than SSV (M vs SSV: $p < .025$, SV vs SSV: $p < .012$). Figure 6.12 shows the mean correctness for T2.

For task T3, the average correctness was 98.3 % (s.d. 0.8) for single viewport, 96.5 % (s.d. 1.1) for split-screen, and 98.3 % (s.d. 0.8) for Mélange. This is a non-significant difference (Friedman test, $p = .189$).

Subjective Preference When asked about their preference on the presentation technique, 5 out of 12 participants ranked Mélange first (5 for split-screen and 2 for single viewport). Comments from the participants were favorable for our new technique, particularly for contextual tasks.



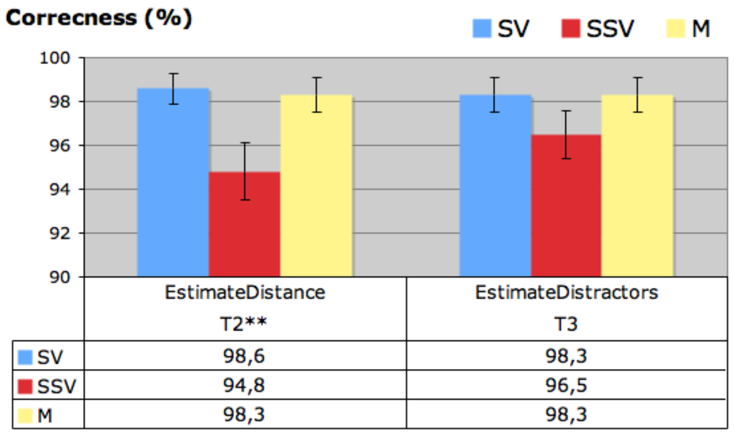
	FindTarget T1	EstimateDistance T2***	EstimateDistractors T3
🟦 SV	18,05	4,13	1,72
🟥 SSV	16,98	4,02	1,9
🟨 M	19,18	2,74	1,64

Figure 6.11: Average completion times for presentation technique across T1, T2, and T3. ANOVA reveals a significant difference for T2 ($p \leq 0.001$).

Task	Factors	F	p
T1	Distance	38.740	***
	Distractors	55.155	***
T2	Technique	8.695	**
	Distance	6.560	**
	Technique*Distance	6.658	***
	Distance*Distractors*Context	4.216	*
T3	Distance*Context	5.335	*
	Technique*Distance*Context	2.660	*

* $p \leq 0.05$, ** $p \leq 0.01$, *** $p \leq 0.001$.

Table 6.3 : : Significant effects of completion time on the factors.



	EstimateDistance T2**	EstimateDistractors T3
🟦 SV	98,6	98,3
🟥 SSV	94,8	96,5
🟨 M	98,3	98,3

Figure 6.12: Correctness for presentation technique for T2 and T3. Friedman's test reveals a significant difference for T2 ($p \leq 0.01$).

Discussion

Summarizing the previous section, our user study yields the following results:

· Our experiment showed no significant differences between the three techniques for visual search (T1) so we cannot conclude about our prediction P1. With 12 participants, the techniques seemed comparable in performance.
· Mélange was significantly faster for the contextual task T2 than both single and split-screen viewport, confirming prediction P2. The difference was almost one-third of the completion time for the competing techniques.
· Mélange promoted significantly better correctness than split-screen viewport. This partially confirmed prediction P3. There was no difference for Mélange in comparison to single viewport, but this may be due to single viewport simply not supporting quick contextual assessment.

Benefits of Mélange These results confirm that the Mélange space-folding technique provides extra benefit beyond the standard split-screen method. More specifically, the results show that providing an awareness of intervening context and distance between focus points helps with contextual tasks, while clearly not consuming too much screen space or cognitive effort to cause poorer performance than split-screen viewports.

Analyzing results for T1 Looking at the completion times for task T1, we note that there is no large difference between single-focus (single viewport) and the two double-focus (split-screen and Mélange) presentation techniques. The reason for this is that T1 is a relatively simple visual search task where the target appearance can be memorized, so two foci are not strictly necessary. We designed the study this way to avoid very long completion times—instead, the objective of task T1 (rather than strictly confirming G1 and G2) is to show that space-folding does not introduce slow-downs in navigation compared to single or split-screen viewports (prediction P1).

Combining T2 and T3 We found no significant difference in completion time for the T3 task, so our prediction P2 only holds for contextual task T2. However, we observed that participants in the user study tended to solve both T2 and T3 simultaneously during the T2 time. This was possible because distance indicators and contextual targets were visible for both tasks. If we combine the completion times for both tasks, the average time was 5.74 seconds for SV, 5.79 seconds for SSV, and 4.17 seconds for M. Removing outliers, this is a significant difference ($F_{2,22} = 4.289, p = .027$): M < SV < SSV.

Outperforming split-screen While Mélange was significantly more correct than split-screen, there was no difference in comparison to single viewport. We believe this is due to single viewport simply not supporting quick assessment of context. Our hypothesis is that, with Mélange, users can easily retrieve the contextual information, whereas split-screen and single viewport require users to invest considerable time to reach the same accuracy.

Generalizing the Results Our results show that the Mélange technique fulfills most of our predictions for the chosen scenario and tasks. The question is naturally whether these results generalize to the whole class of large visual spaces discussed in the introduction.

Application to other domains The answer to this question is two-fold: we believe that the tasks and the scenario used in the study are realistic enough to be ecologically valid, yet general enough to allow us to extend the results to other domains. For the first point, the tasks selected are based on typical user tasks for network analysis [PLP+06]. For the second, the study scenario is sufficiently abstract so that there is nothing in the tasks or the scenario that limits the results. Figure 6.13 shows a large world map being folded using Mélange to bring both northern Italy and eastern Canada into view at high magnification, In this example, contextual tasks may become even easier due to the inherent multi-scale properties of a map (i.e. large-scale features like ocean, land, and mountains are visible even from long distances and under great distortion).

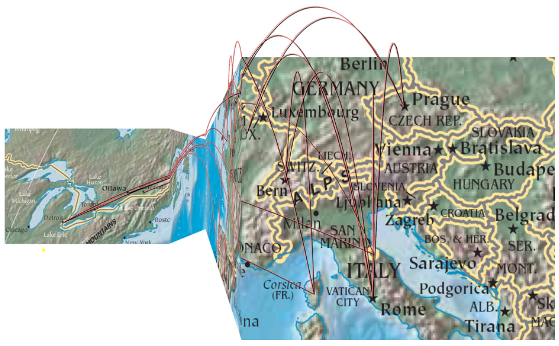


Figure 6.13: Folding a map using Mélange.

2D navigation One specific threat to generalizing the results is that we only tested one-dimensional navigation (horizontal) in one direction (left to right). Two-dimensional tasks may exhibit differences depending on the relative positions of the foci.

Larger distances For larger distances (more than the 16 screens tested in our study), the performance may degrade since the folds become very small and dense. This would happen when navigating a DNA sequence, for example. Supporting this situation is left for future work.

Multi-Focus Interaction in Practice

One important issue with all multiple-foci techniques, including split-screen and space-folding as well as overview windows, is that they divide the user's attention between several different viewports and consume valuable screen estate. Even for a Focus+Context technique like Mélange, there is a non-trivial cognitive effort associated with comparing the different focus regions. As for screen space, users typically interact with only one area of the visual space at a time, so multiple-foci techniques reduce the amount of screen space available for this interaction. Mélange is slightly worse than split-screen due to the fold regions also consuming screen space. Having just a single viewport sidesteps both of these concerns. However, this loss of screen space is balanced by improved context awareness.

As has been shown in this paper, split-screen is perhaps the primary competitor to space-folding. One of its major advantages is its simplicity, both for interaction and implementation. Mélange is unquestionably more complex in both aspects, but we believe that its advantages outweigh this fact. Not only does space-folding better show contextual information, as has been proven in this paper, but it also integrates several foci into the same continuous view, and directly gives the relative positioning of the foci. By the same token, split-screen viewports are fully independent of each other, so they give no intrinsic indication of what part of the space they are showing in relation to the others. In fact, both subviewports may be showing the same target, causing the user to mistake the source node for the destination node, as happened to one of our study participants.

We can anticipate many additional applications for Mélange beyond those discussed in this paper. Figure 6.14 shows an example of a video editing timeline—essentially a 1D visual structure—being folded using our technique. This may be useful for an editor who is synchronizing shots in different parts of a video, or looking to perform color correction between different clips on the timeline. Other potential applications could include finding sections in a large text document using a word search and matching words in the surrounding paragraphs, looking for patterns and trends of geospatial data overlaid on 2D maps that occur in several locations, and even deforming user interface components in applications containing complex menus and toolbars.

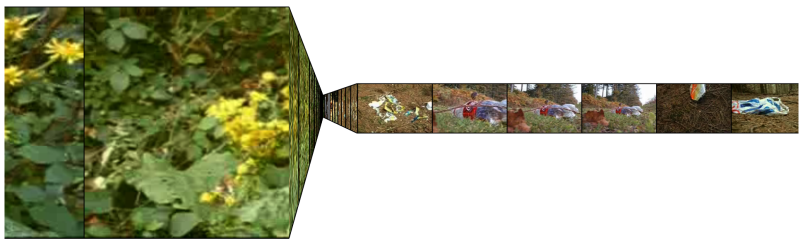


Figure 6.14: Editing a video using Mélange.

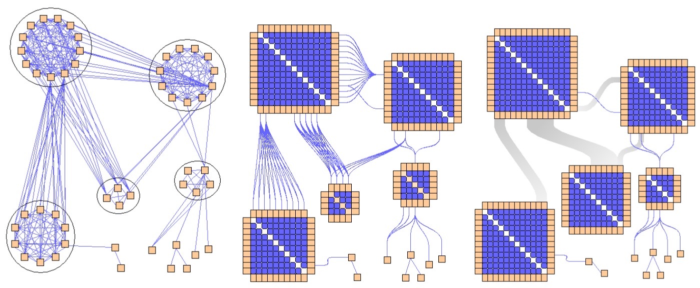
6.4 Conclusion

In this chapter, we focused on two weaknesses of matrix-based representations: the difficulty to perform path-related tasks and to navigate through large matrices. We first presented MatLink, an interactive visualization to help perform path-related tasks. We proved that MatLink did fix this weakness and that for this type of task, MatLink was competitive with node-link diagrams.

In a second section, we presented Mélange, an interaction technique we designed in collaboration with members of AVIZ and InSitu, aiming at following MatLink links in larger matrices. We proved that Mélange allows keeping two or more connected actors in focus while preserving the awareness of their intermediate context. This technique is precious as it supports the understanding of the matrix organization, which leads to the identification of the network structure. Moreover, Mélange is a general interaction technique and can be applied to many other domains.

NodeTrix: merging matrix and node-link representations

CHAPTER 7



(a) Clustered node-link (b) NodeTrix (c) Node duplications

Figure 7.1: NodeTrix and duplications

7.1 Research problem

As previously explained in section 2.2, social networks can vary a lot in structure: from sparse graphs exhibiting a tree structure to very dense ones presenting a table-like structure. Selecting the most suited representation is strongly correlated to the network density. For example, node-link diagrams are particularly effective for very sparse networks while matrix representation clearly outperform them for very dense networks [GFC05]. The difficulty is to identify the density threshold beyond which matrices are more suited than node-link diagrams. This choice is especially ambiguous for small-world networks, a very common category of social networks. The particularity of small-world networks is their global sparse structure with dense local parts. The major difficulty faced when representing these network is to show first how members of communities are connected (intra-community connectivity), then how communities are connected (inter-community connectivity) and finally who the central actors are. Thus, in this chapter, we attempt to solve the following question:

➤How can we design a representation for small-world networks? *i.e.,* improving intra-community and inter-community connectivity readability as well as highlighting central actors?

The second part of this chapter is dedicated to the problem of ambiguous clustering. When an actor is connected to two or more communities, there are three solutions: placing the actor in one or the other, extracting it and placing it between them or generating overlapping communities. While extracting the actor and producing overlapping communities dramatically decrease the representation readability, choosing one or the other community to place the central actor also raises problems as it changes the visual representation, which is potentially misleading. Thus our research question is:

➤How can we solve the problem of ambiguous clustering without degrading the representation readability or misleading the user?

To solve these problems, we present the NodeTrix [HFM07] representation (Figure 7.1b), merging node-link diagrams and matrices to visualize social networks as well as the technique of node duplication [HBF08] (Figure 7.1c) to solve the ambiguous clustering problem.

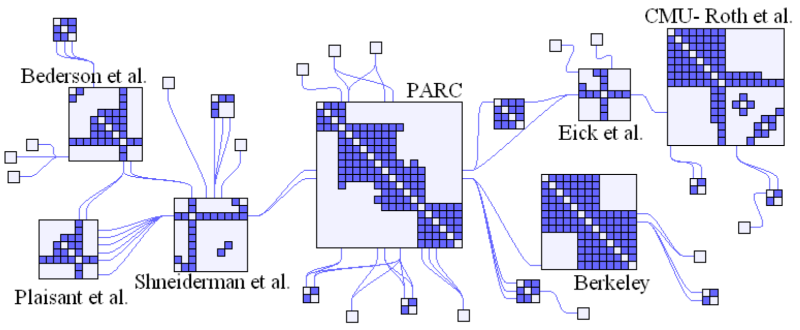


Figure 7.2: NodeTrix

7.2 NodeTrix

In this section, we present our solution for representing small-world networks: NodeTrix. NodeTrix is a node-link diagram where communities (often dense) can be represented as matrices. We present the design of the representation, the design of the interaction and finally demonstrate its strengths through a case study on publication data.

7.2.1 Design

Data Structure Two networks are involved in a NodeTrix representation: the raw *underlying* network (composed of underlying nodes and links) that serves as initial input, and an *aggregated* network (composed of aggregated nodes and links) that is derived from the underlying network. Each aggregate node may correspond to either a unique underlying node or to a group of underlying nodes that typically form a community. Underlying nodes are never shared by aggregate nodes, *i.e.,* there is a many-to-one mapping from underlying nodes to aggregate nodes (and also from underlying links to aggregate links.)

Because our goal with NodeTrix is to provide a readable representation for dense subgraphs, only a single level of aggregation is used: dense subgraphs are simply aggregated and displayed as matrices. Some aggregated nodes may correspond to only one underlying node rather than a group of underlying nodes and these are displayed as a simple node rather than a matrix. However, operations are designed to be uniform over all aggregated nodes. In particular, the user can add a single node to an aggregated node or merge aggregated nodes, whether each node involved corresponds to just one or many underlying nodes. The interaction is described later.

Aggregating attributes Attributes of the underlying nodes and underlying links are combined and propagated up to the aggregated elements. For nominal attributes, values are combined through simple concatenation. Numerical attributes are aggregated either using the average, the min or the max values. An interesting benefit of using matrices in NodeTrix is that they can display the attributes of both underlying elements and aggregated elements, for both links and nodes. Furthermore, because users can dynamically switch between the two representations, more visual variables are available to show attributes. For example, the background color of a matrix can correspond to an aggregated node attribute, while attributes of each underlying node can be shown along the *axes* (the sides) of the matrix. Similarly, the axes can be used to display labels of individual underlying nodes, while a global aggregated node label is also shown.

NodeTrix visualization

To render the NodeTrix representation, a standard node-link layout is used for the aggregated graph, and in addition aggregated nodes containing more than a single underlying node are overlaid with a matrix representation.

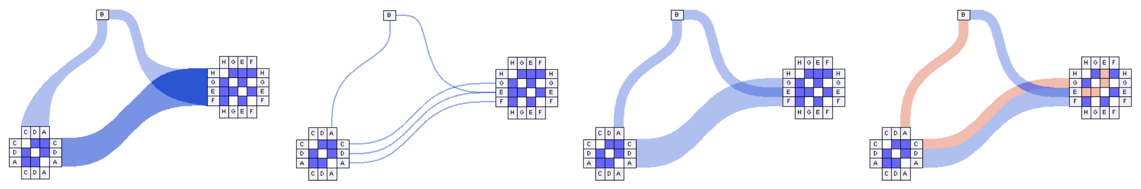
Drawing matrices NodeTrix is built using the InfoVis Toolkit [Fek04] and uses its rendering mechanism to create the visualization. The rendering mechanism involves a pipeline of renderers, which makes it simple to draw a matrix over a standard node. For example, a simple rendering pipeline for a node-link diagram would be: compute_position, compute_size, set_color, fill_shape, draw_border, draw_label. To overlay matrices on standard nodes, we introduced a matrix renderer between the fill_shape and the draw_border renderers. This renderer displays the matrix after having rendered the background node (with a given position, size and color) and before drawing the label and border used for selection.

Matrices have two advantages which make them more readable than node-link diagrams to represent an aggregated node: first, as nodes are placed linearly, links from the rest of the network to the underlying nodes are readable and only suffer from a limited number of crossings; secondly, as nodes are represented both in rows and columns, links can be drawn from any of the four sides of the matrix, which also reduces crossings and overlapping problems. Finally, rows and columns of matrices can be reordered (manually or automatically) to improve readability and further reduce the number of edge crossings (see chapter 4).

Rendering matrices To save memory and allow the user to control all the matrices' properties with a single general control panel, the matrix renderer uses a single matrix visualization object, applying a different permutation and filtering for each aggregated node. Therefore, changing the color attribute for the matrix axes will affect all displayed matrices. We considered creating a separate matrix object for each aggregated node instead, allowing the user to display different attributes on different matrices. Indeed, it would have been very confusing for the user to manage all the controls in a single huge panel (one set of controls for each matrix). A second solution would be to provide the user with a selection mechanism to control sets of matrices. However, this would have added additional complexity for both implementation and user interaction (as drag and drop are used for editing the representation). We decided that sharing the visual attributes for all the matrices was the best option.

Drawing Links To display links in NodeTrix, we considered three options: displaying only aggregated links, displaying only the underlying links, or displaying both.

Aggregated links Displaying aggregated links (Figure 7.3a) provides simple visual feedback on how communities interact. Moreover, an aggregated attribute can be mapped to a visual variable (*e.g.* color, thickness, opacity) of this link. However, the details of which actors of the two communities are interacting are not visible. On the other hand, displaying each underlying link (7.3b) provides connectivity details and enables visualization of the attributes of each link



(a) Aggregated links (b) Underlying links (c) Underlying links with full size (d) Underlying links with attributes

Figure 7.3: Drawing links

independently, but at the cost of many more links and potential crossings. Because small-world networks are globally sparse, they have few inter-community relationships. However, displaying both aggregated and underlying links at the same time could be confusing, due to the possible interaction between visual variables and link crossings or overlap.

Underlying links We chose to visualize underlying links, but with the added flexibility of allowing the user to control the thickness of the links through a slider. Increasing the thickness of the underlying links eventually causes them to merge, and the resulting visual feedback (7.3c) is similar to visualizing aggregated links (7.3a) with increased precision. Moreover, when an underlying link attribute has a color, the thickness of the blended bands of color represents the number of underlying edges (7.3d). The slider that interactively controls the thickness updates the visualization with smooth, immediate feedback. Manipulating this slider allows the user to quickly switch from one kind of overview mode — How are communities linked? What kinds of links? — to a detailed mode — Who link the communities together?

Node-link diagram layout Because the aggregated network in NodeTrix is laid out as a traditional node-link diagram, any existing graph layout could be used. However, because NodeTrix is intended to be used as an interactive exploration tool and we do not want to confuse the user with large, sudden changes to the layout, it seems appropriate to support incremental, interactively-driven changes to the layout, such as aggregating or splitting nodes. The initial layout computed for the graph is Noack's [Noa05] LinLog layout, chosen to give prominence to clusters so they can be quickly identified. After this initial layout step, the user may make local changes by dragging nodes to change their positions, grouping a set of nodes, or removing a node from a group.

Matrix ordering To (re)order the nodes within an adjacency matrix, many different algorithms can be used. As these matrices are typically small, the running time is not an issue. Nevertheless, we chose not to reorder the matrices automatically as they are usually very dense and do not need any particular optimization. Instead, we preferred to allow the user to interactively move rows and columns by drag and drop.

Visual variables and control panel NodeTrix relies on the InfoVis Toolkit to generate controls to filter and affect visual variables. The user controls two sets of visual variables: one for the node-link diagram, and one for the matrices displayed in the aggregated nodes. Both sets of variables consist of the following, for nodes and links: color, transparency, shape size, filled area of the shape, border color, width, and labels. The user filters and associates visual variables to aggregated and underlying network attributes using simple controls such as combo boxes or sliders. The visualization is continually updated, following the principles of direct manipulation [Shn83].

Interaction

We designed a set of interaction techniques to create, edit and manipulate NodeTrix in a very simple and powerful way because we believe that manipulation is key to understanding a network and its potential multiple interpretations. [1]

NodeTrix Editing NodeTrix can be created starting with a pure, traditional node-link diagram. We propose a set of interactions based on dragging and dropping nodes, matrix axis items, and matrix core elements (cells). We feel these interactions are easy to understand as the user simply grabs one of these elements and drops it to another location (possibly over existing elements) to perform an action. When dragging an element, the user has immediate visual feedback and is able to read the element's label.

- *Moving a node or a matrix* to adjust its position and improve the readability of the representation can be done by grabbing the element, dragging it and releasing it at a new position. As the element is dragged, its connecting links are updated.

- To *aggregate a group of nodes* into a matrix, the user may lasso-select the desired nodes, which are then immediately converted into a matrix. To make the transition to a matrix smooth, the transformation from node-link diagram to matrix is animated. The animation speed is adjustable to suit both novice users (who may benefit from seeing a slow animation, to better understand how nodes and edges become organized into a matrix) and advanced users (who would presumably prefer a brief animation).

- *Splitting a matrix* back into a group of nodes is done by right-clicking on it, in which case nodes are positioned with a circular layout around the center of the previous matrix.

- To complete these basic aggregation features, we provide additional interactions for finer-grain editing of the aggregated elements. If users missed an element with the lasso selection or simply wants to *add an additional node* to a matrix, they can drag-and-drop a single node into the matrix. The node will integrate with the matrix, appearing in the matrix axis items (in both rows and columns). Its connections with the matrix elements will be displayed in the matrix core, whereas its connections with the external elements will be displayed as links starting from the matrix axis items and ending at the external elements. If a single node is dragged onto another single node, then the two will be aggregated into a 2×2 matrix.

- On the other hand, if users wish to *extract a node* from a matrix, they can grab the corresponding matrix axis item (either on the row or column axis) and drop it outside the matrix. The dropped item is then displayed as a standard node with appropriate links between itself and the matrix, and the corresponding row and column in the matrix is removed.

[1]A video of NodeTrix is available at `http://www.aviz.fr/~nhenry`

- To increase readability or visualize different combinations, users may want to *move an item* from one matrix to another. This can be done by grabbing a matrix axis item and dropping it on the other matrix. During the transfer, the user is able to read the node label and may cancel the interaction by dropping the element back into the original matrix. This may result in a change to the ordering of nodes in the matrix. The *order of items* in the matrix normally corresponds to the item addition order, with the last item added in the last position. However, when two matrices are merged, the item ordering follows the indices of nodes in the underlying network. The ordering of nodes can be changed by grabbing nodes and dropping them back into the matrix, one at a time, in the desired order.
- Finally, users can *merge matrices* together by dragging-and-dropping a matrix over another.

Matrix axes and zooming feature An aggregated matrix may occupy more space than the original group of nodes in node-link representation. This is partly due to the labels displayed on each side of the resulting matrix. However, while reading labels on each side of the matrix is required to perform community analysis and local editing operations, the axis labels are not required on all matrices at all times, and the size of the matrix core can be reduced to fit the minimum level of readability (Figure 7.4). Moreover, as each matrix possesses a label (reflecting its composition), axis labels for individual underlying nodes may not be necessary at all in a final layout. Figure 7.11a and Figure 7.11b shows respectively the compact and detailed version of the same dataset.

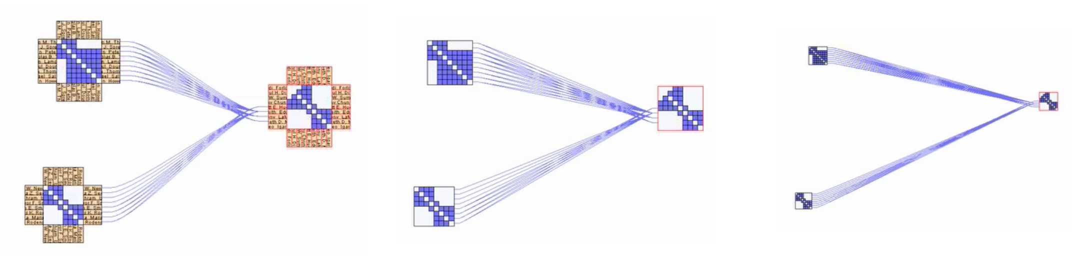

(a) Reducing axes size (b) No more axes (c) Reducing matrix size

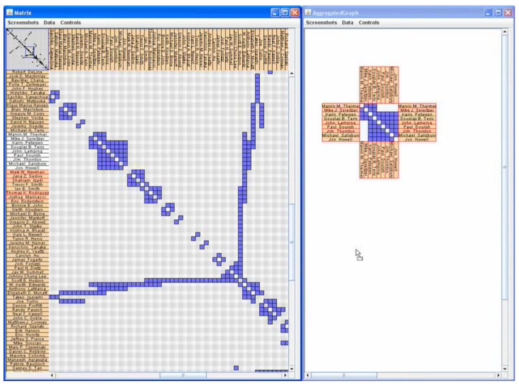
Figure 7.4: From details to overview. The strength of this representation is to provide readable intra-community community even for very small size matrices.

We tried displaying the axis labels on demand following the eccentric label principles [FP99]. For example, if the mouse pointer hovered over a matrix, its axis labels as well as its neighbors' axis labels would be displayed. In this case, axis labels needed to remain visible after the mouse pointer moved (to avoid frustrating the user by losing a landmark when pointing at another item). However, during a case study, we observed that it was more comfortable to be able to read all axis labels when editing, and to remove all axis labels at once and reduce the size of the matrices to get an overview of a final layout. For these reasons, we added two sliders in the control panel to control the size of the matrices and the axis labels.

Supporting the exploration of matrices One weakness of the matrix representation, when exploring a network, is the tedious work required to perform path-related tasks. For example, finding how two communities are connected is tedious as it requires going back and forth alternately reading rows and columns. Moreover, if communities are far apart in the matrix, this task requires a scan of the full length of matrix rows or columns, and connections in a large matrix may lie outside the viewport. Obviously, the task is worse when dealing with three matrices as the user needs to check for intersections of rows and columns in each of the three communities.

We noticed in a participatory-design session reported in [HF06a] that social network analysts also use the matrix representation for some of their analyses. To help perform community analysis and provide support for path-related tasks in general, we provide users with a couple of interaction techniques that work across separate matrix-NodeTrix windows that might be arranged in a dual-viewport or split-screen fashion. These techniques are still based on drag-and-drop, however this time, the user drags a group of elements from one window to another one.

From Matrix to NodeTrix The interaction is made of two steps: first, the user selects a group of nodes in the window of the pure matrix visualization and then drags this group to the NodeTrix window (Figure 7.5). To select the group of nodes, we provide lasso selection directly on the pure matrix representation. Alternatively, the selection can be done on an axis (rows and columns). When a group of cells is selected, the corresponding set of vertices transferred is the union of the edges' source vertices and sink vertices. Dropping the selected group inside the NodeTrix window performs the addition of an aggregated node to the NodeTrix visualization. The group is then displayed as a matrix. Selecting and dropping a second group allows the user to see how these groups are connected to each other visualizing the result with links. The process can continue to visualize connections between several communities.


(a) From matrix to NodeTrix

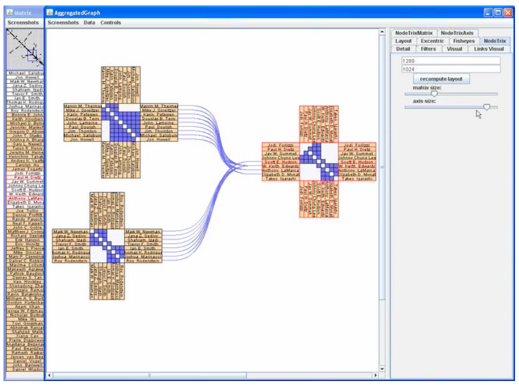

(b) Inter-community connectivity appears

Figure 7.5: Dragging communities from standard matrix to NodeTrix helps analyzing how they are connected. (a) A community has already been dragged into NodeTrix (rows are colored in white in the matrix to show they are already copied). User is transferring another community (rows selected in red in the matrix), the cursor shows that he can drop the selection into NodeTrix. (b) Three communities have been dragged into NodeTrix, inter-community relations can be studied.

Animation

Proper use of animation has much potential to increase the effectiveness of user interfaces and visualizations [Woo84a, BS90a, Bar97a]. To help users maintain their mental model of the network across interactions, we considered how to continuously animate the aggregation of nodes into an adjacency matrix. Typically, animating over transitions involves some kind of interpolation of graphical elements from one state to another. In the case of transitioning from a node-link diagram to a matrix, however, the visual design of the animation is non-trivial, because node-link diagrams and adjacency matrices are composed of very different graphical elements. There is a sort of duality between the two forms: nodes correspond to *points* in node-link diagrams, but to *line segments* (rows and columns) in matrices, and, conversely, edges correspond to *line segments* in node-link diagrams, but to *points* (intersections of rows and columns) in matrices. The key problem is to find an intermediate graph. This work has been mostly conducted by Michael McGuffin. He designed and implemented the animation mechanism, while I integrated it into the NodeTrix prototype.

Prototyping To find solutions, we conducted sessions of sketching, brainstorming, and analysis of how networks can be depicted with node-link diagrams and matrices. We noticed that, although each node corresponds strictly to an entire row and column within a matrix, the node can also be identified with special points in the matrix, that occur where the diagonal and the axes (or sides) of the matrix intersect the node's row and/or column. Furthermore, it is possible to draw a node-link diagram overlaid on a matrix grid, in such a way that the nodes fall on some of these special points, and such that the edges (drawn as poly-lines or curves) pass through their own corresponding locations in the matrix. Figure 7.6, sub-figures 3–7, show some possibilities.

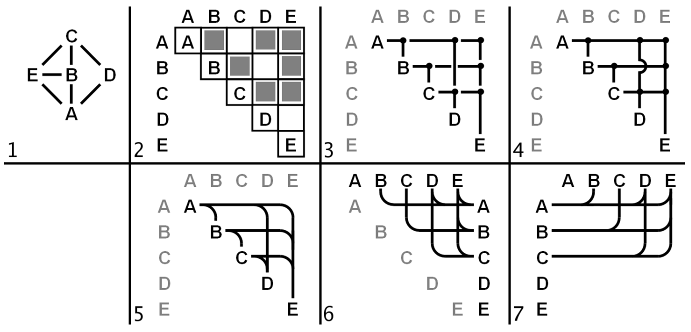


Figure 7.6: 1: A node-link diagram of a network. 2: The corresponding adjacency matrix. For simplicity, only the upper half is shown, since the matrix is symmetric. 3 through 5: different ways of depicting the edges in a node-link diagram laid out over the matrix, using poly-lines or curves. The "corners" of the edges coincide with the filled-in cells of the matrix in 2. 3 and 4: inspired by circuit wiring diagrams. 5 through 7: different choices for the locations of nodes in the node-link diagram laid out over the matrix. 6 and 7: each node is duplicated and has two locations in the node-link diagram.

As can be seen, there are several possibilities for the intermediate state that an animation might interpolate through. We identify a few different design dimensions. First, the edges in the intermediate state might be depicted using poly-lines or curves (Figure 7.6, sub-figures 3–5). Second, the location of nodes might be along the diagonal or along the sides of the matrix (sub-figures 5–7); in the latter case, each node must be duplicated at some point during the animation. (We also note that a simple calculation shows that the average length of links in sub-figures 5, 6, and 7, for large matrices, is 1/3, 1/3, and 1/2 of the side of the matrix, respectively; so 5 and 6 minimize average link length.) Third, the intermediate state might show only the upper half of the matrix (after which the animation might fade in or unfold the other half of the matrix as a mirror image), or the intermediate state might show the whole matrix (before which the animation would have to duplicate the edges somehow, since they occur in each half of the matrix).

Implementation We made a first set of choices along each of these design dimensions and implemented an animated transition from node-link diagrams to adjacency matrices, both in the NodeTrix software and in an additional piece of software. Figure 7.7 shows the latter implementation, where the network has colored nodes and edges. As can be seen, the intermediate state (sub-figure 3) shows both halves of the matrix, hence the animation begins by duplicating edges (sub-figure 2). The positions of the nodes, and of the control points for the edge curves, are gradually interpolated to reach their final locations (sub-figure 3). Then, the edge curves are faded out as the normal depiction of the matrix is faded in (sub-figure 4). Notice that the "corners" of the edge curves coincide with the appropriate cells of the matrix (sub-figure 4), and the opacity of the curves is varied such that these corners are the last part of the curve to fade away, to reinforce their visual correspondence to the matrix cells that fade in.

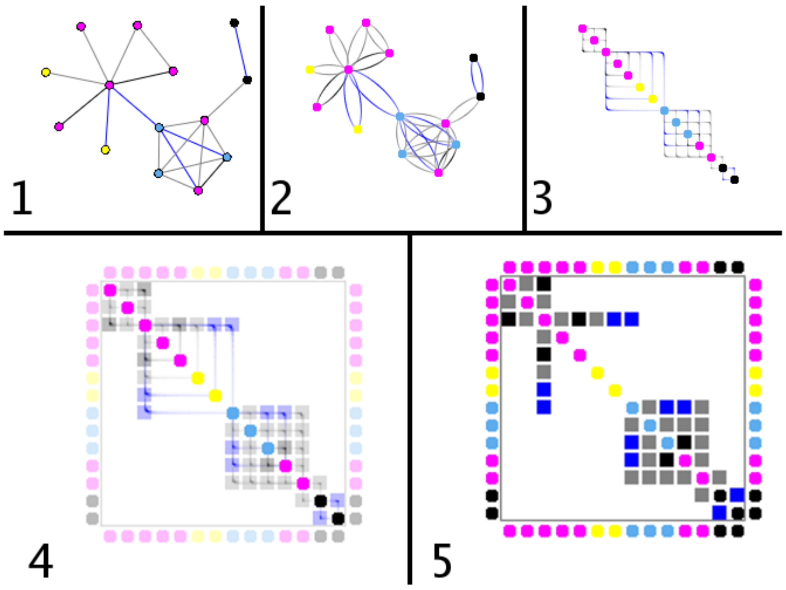


Figure 7.7: The stages of an animation from a node-link diagram (1) to an adjacency matrix (5). Figure 7.6, sub-figure 5 was chosen as the intermediate form through which we interpolate.

Compared with other animated transitions in visualization systems, this animation may seem rather complicated, and in practice an expert user may prefer that the animation be brief (e.g. lasting 0.5 seconds). However, novice users may appreciate having these animations last longer, at least initially. We expect that, in addition to helping the user maintain a mental model of the visualization across transitions, these animations may also have an educational benefit, to help users learn how adjacency matrices are constructed and how to interpret them.

7.2.2 Evaluation

We designed and implemented NodeTrix to fulfill needs we discovered during the case study we performed with MatrixExplorer (chapter 5) on publications data. We observed that neither standard matrix representation nor traditional node-link diagrams could represent small-world networks while preserving intra-community and inter-community connectivity readability. We also realized that presenting the results in a standard article was a real challenge (due to the large quantity of data). In this section, we show how NodeTrix can tackle both of these problems using a subset of our publication data. The use of NodeTrix on the whole dataset is presented in chapter 8.

Tasks In this case study, we focus on the three most important tasks for social network analysis (see section 2.2): identify communities (T1), identify central actors (T2), and analyze roles and positions (T3).

Dataset We selected the data from the InfoVis 2004 contest [IEE04]. It provided us with a clean dataset from which we extracted the co-authorship network of the Information Visualization field. This network is disconnected into 291 components and contains 1104 vertices (researchers) and 1787 edges (co-authorship). It has a low density and a high clustering coefficient, making it a small-world network. We only present here the analysis of the largest connected component, containing 122 vertices and 311 edges. This network could be considered small, but it already presents challenges for exploration and presentation using traditional matrix and node-link diagrams. Detailed communities are not readable in node-link diagrams, while finding connections between communities is tedious in matrices.

Apparatus To manipulate NodeTrix, we used an interactive pen display. Pen-based interactions on NodeTrix are intuitive and comfortable using this input device. The user can simply grab elements by pressing the pen over them, drag them moving the pen on the screen and finally release them by raising the pen. Lasso selection provides also a very intuitive feedback similar to the use of a real pen.

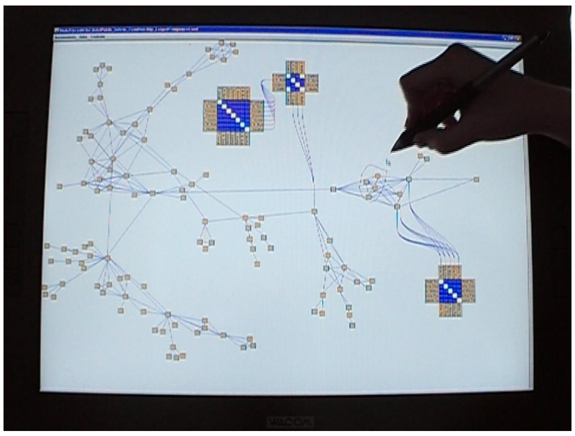
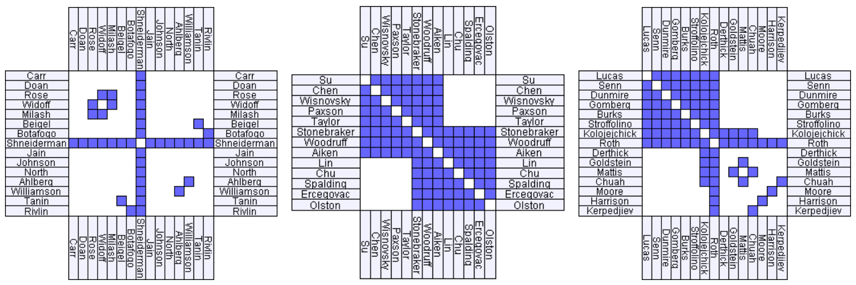


Figure 7.8: Using NodeTrix on an interactive pen display.

Results In this section, we present the results of our case study with NodeTrix. We show how NodeTrix helps analyzing networks through interaction and its versatility to present findings and communicate results at several levels of details.

Finding collaboration patterns The main result of our case study is the identification of different collaboration patterns: cross patterns and block patterns (T1, T3). Using matrices helps quickly identifying these patterns even at the overview level (Figure 7.2). The other benefit of using matrices is the quick identification of missing connections, especially in very dense communities almost clique.

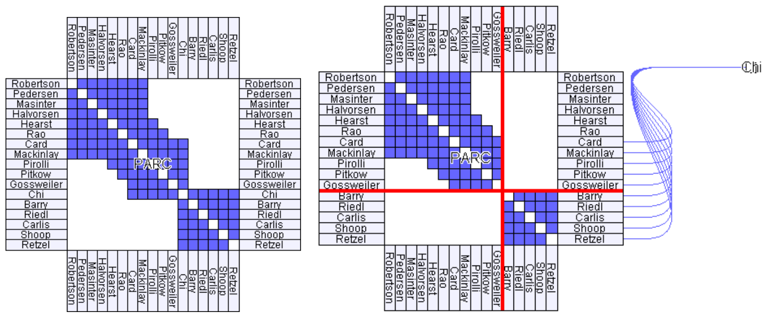
- Figure 7.9a reveals the collaboration pattern of Ben Shneiderman, main actor of the InfoVis field. This aggregated matrix is very sparse and shows only a complete row and column. We named this pattern a cross pattern because if Shneiderman is placed almost anywhere in the matrix (except on the first and last rows-columns), the visible pattern is a large cross. This pattern reveals that Shneiderman collaborates with all researchers in this matrix. However, the low density shows that Shneiderman's collaborators generally do not work together: they are perhaps students he has supervised. Figure 7.11a reveals several matrices with this pattern of collaboration: Plaisant et al., Bederson et al., and Eick et al.
- Figure 7.9b reveals the collaboration pattern of researchers from Berkeley. The aggregated matrix is almost a clique, it is a very dense community. Contrary to the previous pattern, this one reveals researchers strongly collaborating with each other rather than only a single one. Figure 7.11a shows that PARC has the same collaboration pattern.
- The community formed by Stephen Roth is in an intermediate category (Figure 7.9c). Roth is central in this community, but a large block is also visible, meaning that some researchers also collaborate with each other.



(a) Cross pattern (b) Block pattern (c) Intermediate pattern

Figure 7.9: Three collaboration patterns: (a) Shneiderman and his collaborators, (b) Researchers at Berkeley, (c) Roth and his collaborators at CMU.

Supporting the understanding through interaction While exploring the network, the interactions provided with NodeTrix ease the analysis. For example, moving an actor in and out of a community (matrix) helps clarify his influence on this community (T2, T3). Figure 7.10a illustrates this operation, showing that if Ed Chi is extracted from the PARC community, then the community is disconnected. This operation also helps clarify the matrix representation to novice users, as they can drag each actor out of the matrix, one at a time, comparing the visualization of his relationships within and outside the matrix representation.



(a) PARC Community (b) Ed Chi's influence

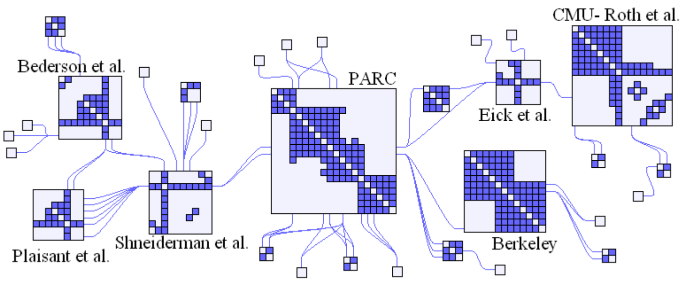
Figure 7.10: Moving a node in and out of a matrix. In the second case, red lines indicate that the matrix is disconnected in two groups (upper left and lower right). Ed Chi is the bridge between these two groups.

Presentation and level of details Presenting results on paper generally requires some filtering when using either pure matrices or pure node-link diagrams. The matrix representation requires space that grows quadratically with the number of nodes: it cannot fit in a printed article with readable labels for networks of more than about a hundred nodes. On the other hand, edge-crossings and node-overlap are issues with node-link diagrams. Thus, with these traditional non-hybrid representations, filtering is required to reduce the size and density of the network to make it more readable.

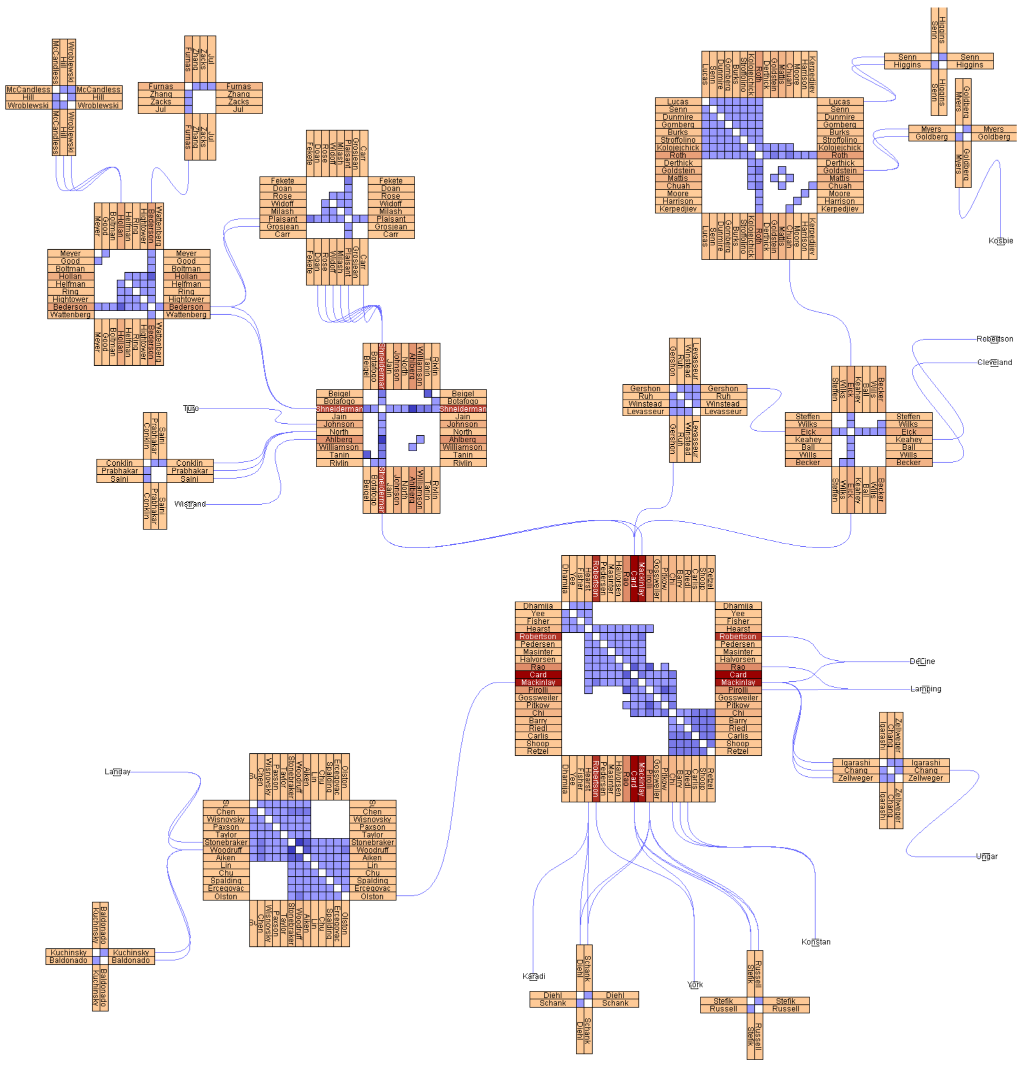
NodeTrix solves these presentation problems as it is a flexible representation for which the level of aggregation as well as the level of details is controllable. For example, Figure 7.11a and Figure 7.11b show the same dataset: the largest component of the InfoVis co-authorship network. In the compact representation, the goal was to provide a brief overview of main communities in the field (T1), whereas in the second representation, the goal was to be able to identify all nodes of the network including actors bridging communities together (T2). In both representations, patterns of connections inside matrices and between them are readable (T3).

Conclusion

In this section, we presented NodeTrix, a hybrid representation merging standard node-link diagrams and matrix-based representations. NodeTrix is particularly suited to represent small-world networks as they are constituted of dense communities connected by few links. Thus, NodeTrix improves intra-community connectivity readability by using matrices and preserves the familiarity of node-link diagrams for representing the global structure.



(a) Compact NodeTrix



(b) Detailed NodeTrix

Figure 7.11: Two NodeTrix representations of the information visualization field. The top one presents a compact version, which aims at presenting communities and their connectivity patterns (intra-community and inter-community). The second representation shows all details of the exact same dataset, which can be used for exploration. Colors on axes of matrices represent the number of citations of each author. Color intensity within the matrices represents the strength of each collaboration.

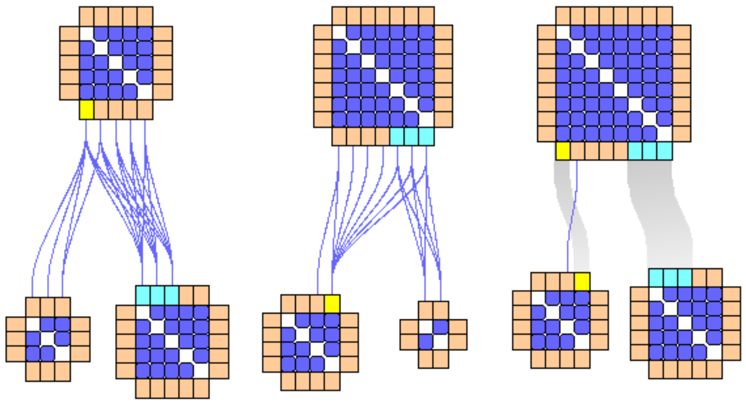


Figure 7.12: The problem of ambiguous clustering. Actors shared among communities are ambiguously placed in only one community (left, center), or using node duplication in all their communities (right).

7.3 Node duplication

As we previously explained, an important task when analyzing social networks is examining communities and their pattern of connections (intra-community and inter-community connectivity). When such communities are identified (for example by clustering algorithms), they can be grouped visually and structurally in a *clustered graph*. This clustered graph is generally represented by a node-link diagram with visible clusters of nodes (Figure 7.2). In the previous section, we presented NodeTrix, a first improvement for the readability of such representations. However, two problems remain:

- *Clustering ambiguity:* When an actor is connected to two or more communities: 1) most clustering algorithms place the actor in one of the communities (Figure 7.12 left,center); 2) others place shared actors between their communities, solving the unique assignment problem but increasing link crossings when several nodes belong to several communities; and 3) in few clustering algorithms, communities that share actors are visualized as overlapping clusters, increasing the visual complexity of the graph by introducing node overlap and link crossings due to the tight space packing. Visualizing the overlapping nodes is difficult or even impossible when the number of intersections increases.
- *Readability:* When two communities share many connections, their links intersect and cross several nodes, hindering the identification of the particular actors connected. Compare to clustered node-link diagrams, NodeTrix improves the intra-community connectivity readability by aligning nodes within communities. However, we still face layouts with a high number of edge crossings.

To improve readability and cluster ambiguity, we propose the *duplication of actors* [2]. To understand how node duplication affects human understanding, we designed and performed a controlled experiment.

[2]In the literature the concept of "duplication" is not clearly defined. Several terms exists reffering to duplicated elements as clones, mirrors or aliases. In graph drawing, duplication is also named vertex splitting.

Node duplication in the literature

The presence of non identified duplicates in a dataset is generally considered as noise. It might skew statistics and provide misleading information. However, when appropriately introduced into a network representation, duplications can improve its readability or ease its exploration. For example, Eades and Mendoca [EdM95] show that duplicating nodes can reduce edge-crossing and reveal symmetries, two important aesthetic criteria in graph drawing.

In information visualization, examples are scattered among the literature where duplication appears among features of a system with few details given, if any. We particularly noticed duplication use in tools visualizing graphs as trees such as OntoRama [ERG02] or TreePlus [LPP$^+$06]. Thanks to duplications, graphs can be presented as trees by suppressing cycles. In these examples, duplicated elements are displayed using a specific color and, in TreePlus the user can click on an element to see its aliases highlighted. Other examples include genealogical tree visualization systems such as GenoPro [Gen]. Here, duplications are used as a presentation or exploration tool, for example by duplicating married couples, members of each side of the family can analyze their own family sub-tree. In GenoPro, a dash line links the original node to its alias. While duplications are particularly used in systems visualizing trees or graphs represented as trees, to our knowledge, they have never been used in clustered graph representations and no work has been published on their effect on human understanding.

7.3.1 Design

As social networks evolve over time, they become denser and thus suffer more from readability and clustering issues. In the 25 years of CHI coauthorship network for example (studied in chapter 8), we observed two common behaviors that particularly challenge the network clustered representation: researchers moving across labs, and researchers co-supervising students. As we will discuss, duplications can provide a clearer picture of such relationships, but we must carefully consider subtle duplication variations and different visual designs.

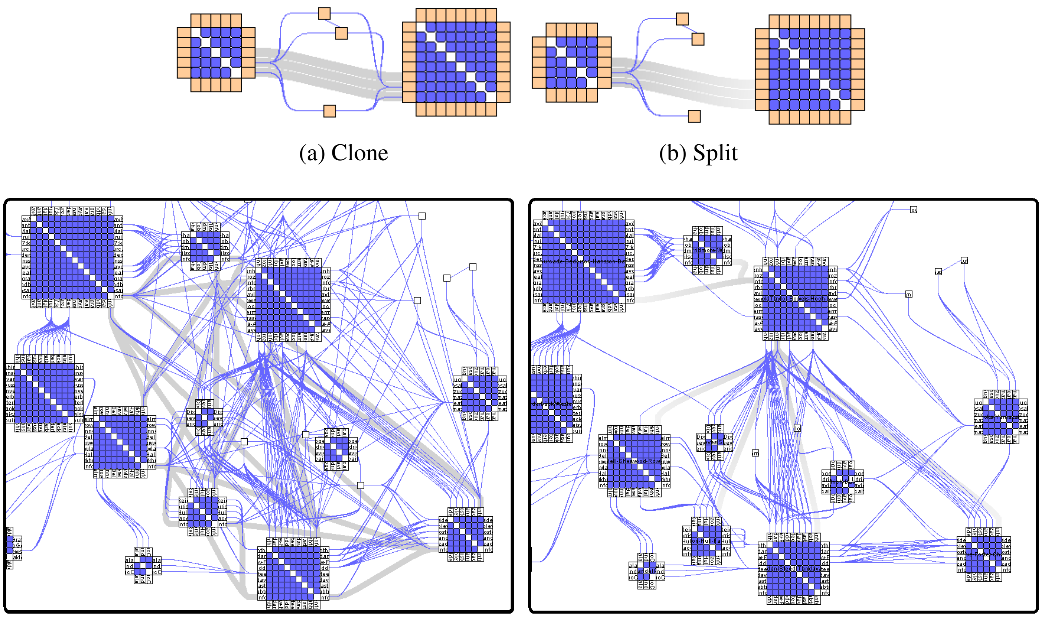
When to duplicate? As researchers move to new research labs they change coauthorship communities and become bridges between them. For example, George Robertson worked at Xerox Parc before moving to Microsoft Research. Should he be placed in Parc, MSR, or outside both of them? In 25 years, many researchers moved or collaborated with different labs (often more than 3), resulting in cluttered network representations, which may be also suffering from arbitrary clustering. By duplicating these central actors, we can provide accurate views of the communities themselves (their size for example), while reducing the number of links displayed.

Another common problem is the representation of two senior researchers (from different research groups) supervising a group of common students. Should the students be placed in one group or should the two groups be merged? Here again we can use duplication in two ways: either by duplicating both senior researchers (to place each one in the community of the other), or by duplicating the group of students. In the second case, more actors are duplicated but it might be more relevant to show all students of each researcher for example.

Although these examples discuss the CHI coauthorship network, duplication can be used in any type of social network where community assignment and clustering is ambiguous.

How to duplicate? We investigated two different types of duplication for graph representations (Figure 7.13 illustrates the differences between both and their visual impact in a whole network):

- *Clone:* An exact copy of the node and all its connections is created, increasing the number of links in the graph and potentially causing clutter.
- *Split:* The node is copied, but its connections are split between the original and the duplicate node. A visual connection (link or label) between the original node and its duplicate is provided, but not between duplicates.


(a) Clone (b) Split

(c) Central actor cloned (d) Central actor split

Figure 7.13: Clone and Split duplications examples. (7.13c) and (7.13d) represent the same subset of CHI coauthorship, a single central actor (Steve Benford) being duplicated in both cases.

How to visualize duplications? Understanding how central actors connect communities is important in social network analysis [WF94], thus the visual connection between a node and its duplicates should be clear and easily accessible.

Existing approaches Existing approaches color-code all duplicated nodes and use their labels as disambiguation mechanisms [ERG02], or interactively highlight the duplicates of a node when selected [LPP+06]. Other approaches link duplicates [Gen] to provide more immediate visual connections. Our hypothesis being that visual links are helpful, our goal was to create links that would be easily distinguishable from regular graph links, while minimizing interference.

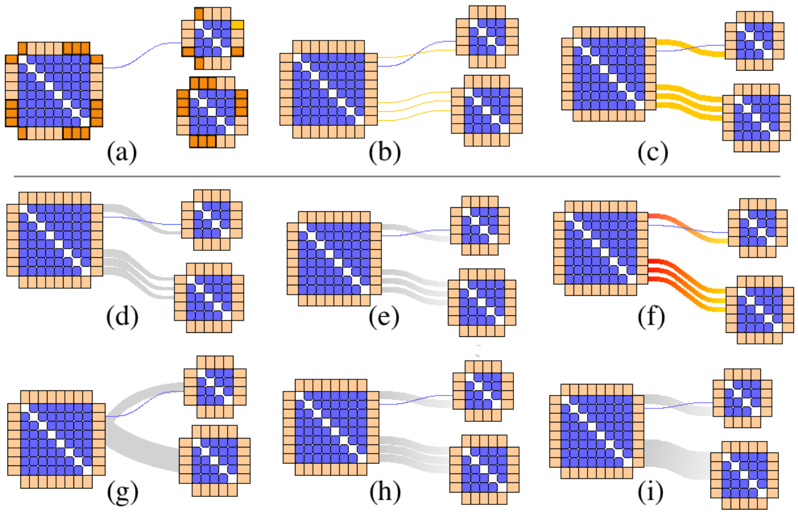


Figure 7.14: Visual design alternatives for representing duplications. Top row: clone duplication using (a) node coloring, (b) link coloring and (c) link coloring and large width. Second and third row: split duplication. As the direction of the duplication is important, we present it using variations in (d) thickness, (d) saturation and (e) color. Duplications towards the same community can be grouped or "bundled" using (f) angle differences, and (g,e) width.

Duplication links Different preattentive features [Tre85] were considered as parameters in the design of duplication links. To minimize interference and create a subtle effect, we rejected early on attention grabbing visualizations such as animated links [BW02, BWC03], or curly and zigzag type links such as the genogram of GenoPro [Gen]. Social networks contain a fair amount of regular links that often cross, thus dotted or dashed lines, as well as using angular changes or curvature in duplication links as differentiation mechanisms would not be very effective.

We thus decided on using combinations of color hue, saturation and width, key visual variables to help differentiate duplication links at a glance [Ber83, CM97a], while creating a subtle effect that can be ignored when users are not interested in the duplication (Figure 7.14). Two designs were considered: representing duplication links as thick de-saturated lines (*linkWidth*) and regular width links of different color (*linkColor*) than regular graph links. For the *linkWidth* lines we chose gray color for a subtle effect, but gray was hard to see in the thinner *linkColor* and was replaced with light orange (same as actor nodes).

Split duplication links In the case of clone duplication all duplicates of a node are exactly the same, but in split the original node and its duplicates differ, an aspect we highlight in our links. Duplication links in split nodes thus fade-out from the original towards the copy. To minimize confusion, we feel that for split nodes all duplicates should be derived from a single original node (1 level duplication) and not from other duplicates, easing the identification of the original node. To maintain the coherence of the duplication amongst the representations, we chose to always place the original node in the most central community — the community having the largest number of connections to other communities.

7.3.2 Evaluation

In collaboration with Anastasia Bezerianos, we designed an experiment to determine the effect of the different duplication types and designs described above. A repeated measures within-participant full factorial design was used, consisting of:

4 visualizations x **2** difficulties x **6** tasks x **2** repetitions x **12** participants = **1152** trials

Pilot study We decided to run an initial pilot study to limit our number of independent variables (duplication types and link designs). After an initial selection, we compared:

· **2 types of duplication**: complete *Clone* duplication (actor and all its connections duplicated) and *Split* duplication (only actor node duplicated).

· with **2 design variations** for expressing the duplication connection (link between duplicates and original actor): links of the same width as other graph links, but of different *Color* (orange); and links of larger *Width* and faint grey color.

· against **2 base case visualizations**: first, clone duplication with color coding (orange) of duplicated nodes [ERG02, LPP$^+$06], indicating their special nature *cloneNode*. No link between original actor and duplicates was present. And second, against *noDuplication* to explore potential issues in using any type of duplication.

Thus the examined visualizations were: *cloneLinkColor*, *cloneLinkWidth*, *splitLinkColor*, *splitLinkWidth*, *cloneNode* and *noDuplication*.

This pilot study revealed that the representation of duplication links using color (*cloneLinkColor* and *splitLinkColor*) performed somewhat worse than the visualization that represents duplication links as faint gray thick links (*cloneLinkWidth* and *splitLinkWidth*). This was especially true in tasks where duplication may hinder performance. As one participant noted "it is easier to perform tasks using the gray thick lines, because I can ignore them when I want to", indicating that gray links of larger width are easier to distinguish from other graph links and can be ignored more effectively. Therefore, for our experiment we decided to use the gray thick links to visualize connections between duplicates, to limit interference with the remaining links of the graph.

Visualizations Thus participants performed tasks using four visualizations:

· *cloneLink*: complete clone duplication with links represented as gray lines of large width as shown in Figure 7.13a.

· *splitLink*: split duplication with links represented as faint gray lines of large width as shown in Figure 7.13b.

· *cloneNode*: complete clone duplication with duplicates colored in a darker orange but not connected by links as shown in Figure 7.14a.

· *noDuplication*: without any duplication.

In all duplicated conditions (including *cloneNode*) participants could interactively select a duplicated node/actor to highlight in red all its duplicates in the graph [LPP$^+$06].

Tasks We decided to first evaluate duplications against two commonly used *graph readability* tasks [PLP+06, GFC05, WPCM02]. We selected an overview task (RO) and a low-level detailed task (RD).

T1 (RO): *actorEstimation*
Participants are asked to compare the size (number of actors) of two sub-networks. To encourage estimation over accuracy and examine how visually similar the two graphs are, participants select one of three answers: larger, equivalent or smaller.

T2 (RD): *actorConnectivity*
Participants are asked to enter the (topological) shortest distance between two actors as a numerical value (number of edges of the shortest path).

In *social networks analysis*, it is important to identify community structures and their connections (C), as well as central actors and their influence on different communities (CA). To that end we selected another 4 commonly performed tasks [WF94]:

T3 (C): *communityCentrality*
Participants are asked to identify and select the most central community, the community that shares central actors with the largest number of other communities.

T4 (C): *communityCloseness*
Participants are asked to identify and select the two communities that share the larger number of central actors (have the strongest cohesion).

T5 (CA): *mostConnected*
Participants are asked to identify and select the most connected actor, the actor with most connections to other actors (equivalently the node with the largest degree).

T6 (CA): *articulationPoint*
Participants identify an actor that lies between a community and the rest of the network. To help participant understand this notion, actors are described as actors that, when removed, disconnect one or more communities of from the network (graph cut-point).

These six tasks are representative of general purpose graph readability and social network practices. Moreover, they were selected so as to cover analysis tasks ranging in granularity from overview understanding (*actorEstimation*), intermediate community structure (*communityCloseness*, *communityCentrality*) and central actors (*actorCentrality*, *actorConnectivity*), to detailed tasks focusing on specific nodes or links (*actorBetweeness*). Since duplication affects the general layout of the network, the selected tasks are relevant to the topology of the network, and not specific attributes of individual actors or links (another set of common social network analysis tasks [PLP+06]).

Dataset Generating synthetic representative graphs is still a challenge. As explained in [HF07], current small-world generators do not provide realistic models. Therefore, to conduct the experiment using realistic graphs that follow properties of small-world networks, we used subsets of actual datasets. Specifically coauthorship data for 25 years of CHI conferences and 20 years of UIST conferences. The labels of actors in the dataset were replaced by codes to avoid interpretation issues. Subsets of the graphs were carefully chosen (but not altered) to provide a balanced design: we balanced the link density between graphs, the number of communities, the number of duplicated actors and ensured that communities of actors are always cliques (to make their identification in the *noDuplication* visualization easier).

Clustering In social networks, it is important to see both communities and central actors but existing automatic clusterings only give communities. For the experiment purposes, we ensure communities were all cliques and assumed that actors falling into two or more cliques (connected to all actors or the communities) were central actors. We used the edge-betweeness clustering algorithm [GN02] implemented in the JUNG [FOS$^+$05] package, and edited a few of the communities a posteriori to ensure they were all cliques. All central actors were duplicated. In the splitLink, original actors are placed in the largest community. In the noDuplication condition, central actors belonging to more than one community are placed in the largest community (clique) they belong to. We used NodeTrix [HFM07] to represent the topological structure of the social network. The graph layout was performed using a manual layout minimizing edge-crossing, tuned from an initial LinLog [Noa05] algorithm layout.

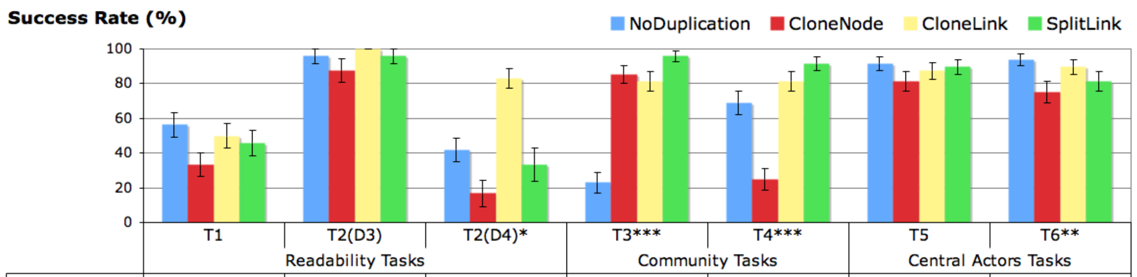
Difficulty In order to better understand the effect of duplication, participants were asked to perform the mentioned tasks in graphs of two types of difficulty (*Diff:easy* and *Diff:hard*). Graph difficulty was determined based on the graph density, as this attribute that has been proven to affect task performance in graph understanding [GFC05]. The density (ratio of existing number of links over all possible links in a graph, a ratio that is fairly small in small world networks [WS98]) in the easy condition was 0.14, as opposed to 0.19 for the hard one. As both networks had small-world properties, they had a high clustering coefficient (0.86 in both cases).

Participants and apparatus Twelve participants (1 female) took part in the study. Aged from 23 to 40, they were all researchers or students of a graph drawing research group, to ensure familiarity with computers and graph representations. We used a 3GHz Pentium IV computer with 1GB of RAM and one 19" screen during the controlled experiment. Participants entered their answers using mouse or keyboard.

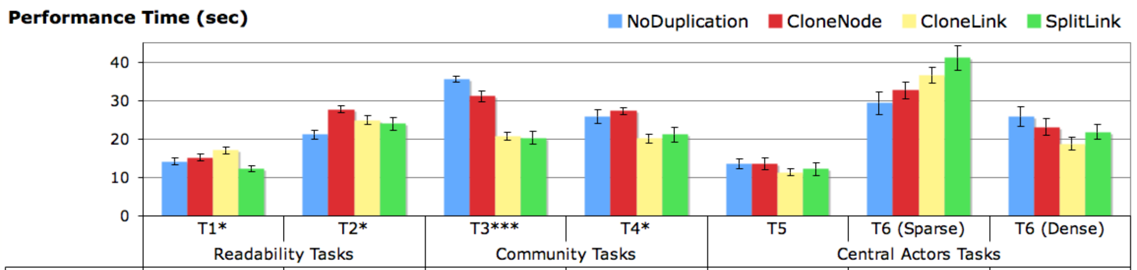
Procedure Participants were randomly assigned to 4 groups of 3. In each group participants used all 4 visualizations, in an ordering balanced using a Latin square. For each visualization participants completed a single block, containing 2 trial repetitions for all combinations of task (6) and difficulty (2). To reduce memorization of graph layouts between trials, graph labels were randomly generated and the entire graph layout was rotated randomly.

Before the experiment, participants were interviewed to gather information about their previous experience with graphs and visual representations. A tutorial sheet introduced the NodeTrix representation, the duplication designs and each of the six tasks to complete. An experimenter was present to answer all questions. Participants could then practice with the experimental system for random trials on a training dataset. Training was also given at the beginning of each visualization block. The training sessions lasted 10 min on average and ended when the experimenter ensured all tasks and visualizations were understood. At the end of the experiment, participants completed a questionnaire eliciting their visualization preference per task and overall, and their free comments about each visualization.

Participants were asked to perform the task correctly as fast as possible. To prevent random answers, if participants felt unable to answer a question, they were allowed to skip it. To limit the experiment duration, task completion time was limited to 60 seconds. Neither of these events occurred.



	T1	T2(D3)	T2(D4)*	T3***	T4***	T5	T6**
	Readability Tasks			Community Tasks		Central Actors Tasks	
🟦 NoDuplication	56,3	95,8	41,7	22,9	68,7	91,7	93,7
🟥 CloneNode	33,3	87,5	16,7	85,4	25	81,2	75
🟨 CloneLink	50	100	83	81,3	81,3	87,5	89,6
🟩 SplitLink	45,8	95,8	33,3	95,8	91,7	89,6	81,2



	T1*	T2*	T3***	T4*	T5	T6 (Sparse)	T6 (Dense)
	Readability Tasks		Community Tasks		Central Actors Tasks		
🟦 NoDuplication	14,19	21,14	35,59	25,84	13,59	29,32	25,79
🟥 CloneNode	15,16	27,67	31,07	27,26	13,53	32,62	23,13
🟨 CloneLink	17,01	24,9	20,72	20,1	11,33	36,55	18,69
🟩 SplitLink	12,18	23,97	20,28	21,14	12,19	41,07	21,86

Figure 7.15: Success Rate and Performance time per visualization for the 6 tasks.

Hypotheses and Results

In this experiment, the independent variables were *Task* (actorEstimation, actorConnectivity, communityConnectivity, largestCommunity, articulationPoint, mostConnected), visualization *Vis* (noDuplication, cloneNode, cloneLink, splitLink), and difficulty *Diff* (easy, hard). The performance measures for each visualization were: *Success Rate*, performance *Time* and overall preference for each of the tasks. ANalysis Of Variance (ANOVA) on performance *Time* and non-parametric tests of Friedman and Wilcoxon on *Success Rate* were performed independently on each of the tasks, as they differ greatly in granularity and scope. Figure 7.15 illustrates our results.

Task 1: actorEstimation (RO)

H: The use of duplication will degrade the accuracy and performance time for comparing the size of two graphs, as duplicated actors result in larger number of nodes in a graph.

Success Rate: Surprisingly, the Friedman's chi square test showed no significant effect of *Vis*. This task was very error prone for all visualizations: cloneNode (33% success rate), splitLink (45%), cloneLink (50%) and noDuplication (56%).

Time: A significant effect of *Vis* on time was present ($F_{3,33} = 7.43$, $p < .05$), as well as a significant *Vis* x *Diff* interaction ($F_{3,33} = 9.94$, $p < .0001$). Mean times were: splitLink (12.17sec), no duplication (14.18sec), cloneNode (15.16sec) and cloneLink (17.01sec). Post-hoc pairwise mean comparisons showed cloneLink to be significantly slower than splitLink but all other pairs were not statistically significant (all adjustments Bonferonni). The interaction *Vis* x *Diff* showed that cloneNode performed better for *Diff*:Hard graphs while the other were negatively affected. Contrary to our prediction, duplications did not degrade performance time and accuracy for this task.

Preference: Not surprisingly 9 of 12 participants preferred noDuplication for this task, explaining that "the networks to compare feel more different when there are duplications". The remaining preferred either cloneLink or splitLink. However several reported that they "usually overcompensate the number of duplications" and had low confidence in their answers. 7 out of 12 thus ranked this task as the most difficult.

Task 2: actorConnectivity (RD)

H: The introduction of duplication will negatively affect the performance time and accuracy of counting the distance between two actors, as extra duplication links are introduced between actors.

Success Rate: The Friedman's test showed a significant effect of *Vis* ($p < .05$) on Success Rate. Pairwise comparison using the Wilcoxon's test showed that noDuplication (69% success rate) was more accurate than cloneNode (53%) and cloneLink (55%), but not splitLink (65%). For this unique task, we avoid memorization effect by using one trial to count a distance of 3 (D3), the second for a distance of 4 (D4). Thus we varied the tasks difficulty and added this independant variable to our analysis. The Wilcoxon's test revealed a significant difference between the task difficulty: D4 (25% success rate) is far more error prone than D3(95%). If we split the results by task difficulty, the Friedman's test reveal a significant difference for the most difficult task D4 ($p < .05$). The Wilcoxon's test shows that noDuplication and splitLink perform both better than cloneLink (17% success rate). No significant difference is revealed between noDuplication (42%) and splitLink (33%).

Time: A significant effect of *Vis* on Time was present ($F_{3,33} = 5.99$, $p < .05$), as well as a significant *Vis* x *Diff* interaction ($F_{3,33}$ = 10.31, $p < .0001$). Mean times were: noDuplication (21.13sec), splitLink (23.96sec), cloneLink (24.90sec) and cloneNode (27.66sec). Post-hoc pairwise mean comparisons showed cloneNode to be significantly slower than both cloneLink and noDuplication. The interaction *Vis* x *Diff* showed stronger differences for *Diff*:hard graphs. Analysis split by task difficulty showed that there was no difference in *Vis* in the *Diff*:easy graphs. In the *Diff*:hard graphs cloneNode was significantly slower than all other visualizations, whereas noDuplication was faster than cloneNode and cloneLink, but not from splitLink (all adjustments Bonferonni). Although we expected duplications to degrade time performance, this was only true in the clone duplication cases (cloneNode and cloneLink). SplitLink did not significantly degrade the performance time or success rate compared to noDuplication.

Preference: Surprisingly only 4 out of 12 participants preferred the noDuplication condition for counting the distance between two actors, saying it was easier "as you did not need to go through an extra link of cost 0". From the duplication conditions, splitLink was generally preferred (4/12 ranked it first, 5/12 second). Several participants reported that it could be "tricky to see where the link goes" in noDuplication whereas "duplications cleans the graph" and "makes it easier to see the shortest path".

Task 3: communityConnectivity (C)

H: The introduction of duplications (especially using links) will help identify faster and more accurately the two communities that share the larger number of actors.

Sucess Rate: The Friedman's test showed a significant effect of *Vis* ($p < .0001$). As expected, the Wilcoxon's test showed that all 3 duplication visualizations performed better than noDuplication (only 33% sucess rate). Identifying actors (nodes) that are clustered in one community but belong to others is a harder task than counting duplicated actors. SplitLink was the most accurate visualization (96% success rate), followed by cloneNode (86%) and cloneLink (82%). splitLink was also significantly more accurate than cloneLink ($p < .05$). The better accuracy of splitLink is due to the "cleaner" resulting graphs, where links of the duplicated actors are not duplicated themselves.

Time: There was a significant main effect of *Vis* on *Time* ($F_{3,33}$ = 22.63, $p < .0001$) and a significant *Vis* x *Diff* interaction effect ($F_{3,33} = 6.5$, $p < .0001$). Mean times were: splitLink (20.28sec), cloneLink (20.72sec), cloneNode (31.07sec) and noDuplication (35.58sec). Post-hoc mean comparisons showed splitLink and cloneLink to be significantly faster than both cloneNode and noDuplication. The interaction *Vis* x *Diff* showed stronger differences for *Diff*:hard graphs. In the *Diff*:hard graphs noDuplication was significantly slower than all the duplication techniques. Indeed times in the case of duplication using links (splitLink and cloneLink) were generally faster, as it is easier to notice the width of the duplication link bundle going to different communities to identify the two communities that share the most actors. Nevertheless, in complicated graphs even simple cloneNode outperformed noDuplication, as looking for duplicated actors is easier than counting links from actors to other communities.

Preference: 11 of 12 participants preferred duplication techniques for this task, with splitLink and cloneLink being the most preferred (5/12 each). Participants reported that "grey lines tell the story", that it was easy to estimate the width of the grey lines to see community-Connectivity. They described splitLink as "cleaner", but as grey lines between multiple duplicates of the same actor were missing, they their confidence was lower. No one preferred noDuplication.

Task 4: communityCentrality (C)

H: Duplications (especially using links) will help identify faster and more accurately the most central community, that shares actors with the most other communities.

Sucess Rate: Friedman's test showed signicant effect of *Vis* ($p < .0001$). Wilcoxon's test showed cloneNode, (25% sucess rate) as more error-prone than the remaining techniques. Also splitLink (92%) performed significantly better ($p < .05$) than noDuplication (69%), with no difference between cloneLink (82%) and noDuplication (69%).

Time: There was a significant main effect of *Vis* on *Time* ($F_{3,33}$ = 3.07, $p < .05$) and a significant *Vis* x *Diff* interaction effect ($F_{3,33} = 6.78$, $p < .05$). Mean times were: cloneLink (20.01sec), splitLink (21.13sec), noDuplication (25.83sec) and cloneNode (27.26sec). Post-hoc pairwise mean comparisons showed cloneLink to be significantly faster than cloneNode in all graph difficulties. The interaction *Vis* x *Diff* showed stronger differences between *Vis* for *Diff*:hard graphs. cloneNode and cloneLink are positively affected by *Diff*:hard while the other techniques are negatively affected. All other pairs were not significantly different. Contrary to our expectations, the duplication conditions using links did not yield significantly faster times (although their mean times were faster overall), indicating that the normal links between shared actors in the noDuplication condition are enough to identify central communities. Nevertheless, duplications using links make this identification more accurate.

· **Preference:** Not surprisingly 11 of 12 participants preferred duplication techniques for this task, with cloneLink and splitLink being the most preferred (often ranked at the same position)(6/12 for each). A participant commented the grey links helped a lot as "you can stop paying attention on the blue (regular) lines and concentrate on the grey only". Several participants reported that splitLink could be misleading as they thought the choice of the original node was decisive. Almost all of them reported the cloneNode "required exploration", "many clicks and memorization".

Task 5: articulationPoint (CA)

H: Duplications using links will help in the identification of articulation points (actors bridging two communities), as fewer links are present between communities.

Sucess Rate: Friedman's test showed no significant effect of *Vis* on Success Rate: noDuplication(92% success rate), splitLink (90%), cloneLink (88%) and cloneNode (82%).

Time: There was no significant effect of *Vis* on time, with mean times for duplication techniques being slightly faster: cloneLink (11.32sec), splitLink (12.18sec), cloneNode (13.53sec) and noDuplication (13.85sec).

Preference: Almost all participants considered the representations equivalent for this task. 8 of 12 participants reported that this task was the easiest. When asked about their strategy, almost all replied that they "look at the network periphery to find a community linked by a few or single connection". Several reported that splitLink was confusing as they "wondered about missing links".

Task 6: mostConnected (CA)

H: The introduction of duplications will make the identification of the most connected actor (larger number of connections) harder, as the actors are now in multiple communities.

Success Rate: Contrary to previous studies [GFC05], where datasets were artificially generated and the most connected actor degree increased of 5% between trials, we did not modify our datasets. In the easy graph, two actors were candidates with only a small degree difference between them, therefore they were both considered as a right answer. Friedman's test showed a significant effect of *Vis* ($p < .01$). The Wilcoxon's test revealed a significant difference between noDuplication (94% success rate) and cloneNode (75%) but no difference between noDuplication and cloneLink (90%) or splitLink (81%). CloneLink performed significantly better than cloneNode and splitLink($p < .05$).

Time: There was a significant main effect of *Vis* on *Time* ($F_{3,33}$ = 17.47, $p < .0001$) and a significant *Vis* x *Diff* interaction effect ($F_{3,33} = 7.39$, $p < .05$). Mean times were: cloneNode (24.93sec), noDuplication (28.76sec), cloneLink (28.84sec) and splitLink (39.28sec). Post-hoc pairwise mean comparisons showed splitLink to be significantly slower than all other visualizations. This effect is present in the *Diff:*easy graphs but not in the *Diff:*hard ones (Figure 7.15 (T6)). Moreover in *Diff:*easy graphs, cloneLine was also slower than cloneNode. SplitLink performed worse than all the techniques.

Preference: 8 of 12 participants preferred duplication techniques for this task. They explained that they roughly estimated the most connected actor as one that is part of many communities. They also reported that splitLink looks cleaner but that the missing links greatly lower their confidence. Half of the participants reported this task as most difficult.

Overall user preferences and comments Almost all participants reported that cloneNode, used often in practice, would be useless without node highlighting. Even with it, they describe their strategy as “a trial and error process”, tedious and cognitively demanding. Surprisingly, 6 out of 12 participants counted splitLink as their preferred visualization overall (2 for noDuplication, 1 for cloneNode and 3 for cloneLink). When asked why (as splitLink was not ranked first in most tasks), roughly all participants reported that “it looks cleaner” but they feel less confident, as links between multiple duplicates are not present (only between the original and duplicates). However, they all commented that this feeling would probably disappear with more practice.

Discussion

Our experiment showed that the most common technique for duplications (cloneNode) performed very poorly, even when coupled with interactive feedback, and that visual duplication links were more effective. Our design choices attempted to create two types of links easily distinguishable by using a subtle combination of color, intensity and width. We hoped that users could pre-attentively identify and ignore the duplication links when needed, thus not affecting the performance of readability tasks (RD and RO) and similarly ignore the regular links while performing community tasks(C). The good performance of duplication across tasks and qualitative results of our experiment indicate that this was achieved. Indeed most of our participants clearly stated that they could ignore one or the other type of links, which was helpful when performing the tasks.

Readability tasks (RO, RD) We expected duplications to degrade the success rate and performance time of overview and detailed readability tasks, but results from our experiment proved that *split duplication was as effective as noDuplication for both tasks*.

As our overview task (RO) was particularly error prone, results showing no significant impact of duplications on the performance time and success rate must be interpreted carefully. Previous studies [GFC05] also showed that counting nodes or links is a particularly error prone task. Some participants reported that the visualization was not familiar and thus estimating the number of nodes was hard. During the pilot, one participant noted that “the matrices increased the visual effect of the number of nodes (because you compare areas larger than the actual number of actors they contain)”. This could be considered an artifact of the NodeTrix representation, but it should not have affected our results for duplication as both noDuplication and duplication cases are affected.

Interestingly, duplications do not strongly affect the detailed task (RD). Participants even reported the task was easier to perform with splitLink as it suppressed a number of link crossing, making it easier to follow the connections. For clone duplications, the detailed task seems more affected in denser networks, where clone duplication results in multiple copies of regular connections for all duplicates.

Community tasks (C) Our experiment showed that *duplications help identifying the connections between communities*. The visual design showing links has been proven more efficient and almost all participants reported the grey lines were really helpful. Qualitative results from the pilot showed that using a grey thick line instead of a thin line of a different color provides two levels of readability. All participants of the pilot reported that it was easier to concentrate on grey lines and ignore other types of links (and vice versa).

The introduction of *duplications also helps to find more accurately a central community*, especially using splitLink. Here it is easier to follow the direction of the different duplication links (leaving the most central community), than in cloneLink or noDuplication which suffer from regular link clutter. CloneNode performs really poorly, as the only way to identify central communities is to select all the duplicated actors of a community and identify (through highlighting) the other communities it is connected to.

As duplications solve the problem of ambiguous clustering, they *display the true size of communities*, providing more accurate comparisons between communities. This task was not part of our experiment, but part of our motivation.

Central actors tasks (CA) The concept of central actors in a social network is intuitive but hard to perform in a controlled experiment. In practice, it requires several measures (such as computing several centrality metrics) and the interpretation of the actors' attributes. For our experiment, we selected two common tasks that are simple to explain and validate: finding an articulation point and identifying the most connected actor. We expected that these tasks would be positively and negatively affected respectively by duplications. Our predictions were wrong, as *no difference was shown for the articulation point* while *duplications were preferred for the most connected actor*.

While using duplications, we expected that summing up all duplicates and their links to identify the most connected actor, would degrade the overall performance, but it only did for the ambiguous case (Figure 7.15, T6 (sparse)). Most participants reported that they considered the most connected actor as one part of many communities. Duplications were thus preferred, as participants counted the outgoing grey lines or the highlighted duplicates of an actor. This is the only task where cloneNode performed well, as participants directly clicked only duplicated nodes (that are of a discrete color) to find one disseminated across many communities, instead of trying to identify the directions of the grey links.

Our experiment showed that the most common technique for duplications (cloneNode) performed very poorly, even when coupled with interactive feedback, and that visual duplication links were more effective. Our design choices attempted to create two types of links easily distinguishable by using a subtle combination of color, intensity and width. We hoped that users could pre-attentively identify and ignore the duplication links when needed, thus not affecting the performance of readability tasks (RD and RO) and similarly ignore the regular links while performing community tasks(C). The good performance of duplication across tasks and qualitative results of our experiment indicate that this was achieved. Indeed most of our participants clearly stated that they could ignore one or the other type of links, which was helpful when performing the tasks.

Duplication guidelines

Our results apply to NodeTrix representations and more generally to clustered node-link diagrams. The difference between these representations lies in how communities are visualized but both suffer from the same readability and ambiguous clustering problems. While NodeTrix improves intra-community readability by removing link crossings, both representations could benefit from duplications to improve inter-community readability (by reducing the number of links) and to provide more accurate views of the communities, showing their actual size and highlighting shared actors. Based on our experimentation, we propose the following general guidelines for node duplications in clustered graph representations:

➤**When to duplicate?**
(1) To reduce visual complexity in graphs that have many actors shared among communities.
(2) To emphasize central actors that connect multiple communities. To highlight their importance, such actors may also be extracted from their communities when duplicated.
(3) To provide accurate community-centered views, an important aspect of many social network analysis tasks.

➤**How to duplicate?**
(4) Using either split or clone, but not a combination of the two, as they are both complex representations.
(5) Clone can be used as base case, as it requires less practice (at the expense of cluttering the network).
(6) Split reduces visual complexity, but interactive highlighting of the duplication links may be required for novice users.

➤**How to visualize duplication?**
(7) Simple colored nodes are not enough for representing duplications. Links between duplicates are more effective.
(8) To increase readability, visual links that connect duplicates should be easily distinguishable from other graph links.
(9) Interactive highlighting of duplicated nodes and links is desirable.

Potential interaction for duplication

Our participants suggested a couple of interesting directions for supporting the creation, editing and visualization of node duplications. Providing interactive feedback on links, by mousing over them and highlighting all links of the duplicated actor could help identifying duplicates much faster. It could be particularly useful to find the original node in the case of split duplication. A second interaction could provide several levels of details: a default state could show duplications links grouped in a bundle between communities (to provide a rough estimation); then it could become independant (countable) on mouse over. Finally a smooth animation merging back all duplicates of a node into a single one, could improve the understanding of duplications effects and show the impacted network areas.

7.4 Conclusion

In this chapter, we presented NodeTrix, a hybrid visualization designed to improve the readability of social networks presenting a small-world structure. These networks, very common, are globally sparse but locally dense. Therefore, there is no clear advantage of choosing either matrix representations or node-link diagrams to represent them. NodeTrix merges both representations, using a node-link diagram to visualize the global sparse structure and matrices for the dense sub-parts. Thus, compared to pure node-link diagrams, NodeTrix improves intra-community and inter-community connectivity readability.

In the second part of this chapter, we studied the problem of ambiguous clustering (choosing in which community to place a central actor). We proposed to solve it by using node duplications and performed an experiment to understand how duplications can affect human understanding. We concluded on guidelines on how to use duplication in general clustered graph representations.

Ecological validation

CHAPTER 8

Figure 8.1: Evaluating with users

In this chapter, we present how we evaluated our prototypes to explore social networks with users (Figure 8.1). We explain the challenges we faced when selecting tasks and datasets for evaluating the readability of representations using controlled experiments. The largest part of this chapter is dedicated to the results of the case study we realized using MatrixExplorer and NodeTrix on 20 years of publication data for 4 Human-Computer Interface (HCI) conferences [HGEF07].

8.1 Research problem

The main questions we address when evaluating an information visualization system are:
➤How do users perceive and interact with the visualizations we designed for them to better understand their data?
➤More generally, what is the benefit of using a given information visualization system to analyze a given type of data?

As we explained in section 2.5.3 of the related work, evaluation is a challenge in information visualization, especially when supporting exploratory data analysis. Standard controlled experiments used in HCI, that collect performance time and error rate for a simple task (such as pointing to an object to validate Fitt's law for example) are difficult to adapt to information visualization techniques. First, operationalizing (decomposing a complex task in several simple ones) data exploration is very hard and can be done in several different ways. Secondly, these lab studies only offer a very focused evaluation in a very controlled environment far from a realistic use and therefore potentially not capturing the real benefit of using the system.

This research problem is a topic of interest to the whole information visualization community. A series of workshops initiated a reflection on novel evaluation methodologies in 2006 [BPS07]. Recent methods include insight-based evaluation [PFG08] or longitudinal field studies [SP06]. These methods emphasize the close collaboration with real users of the system and the findings collected with it. Inspired by this philosophy, we explain our approach, constraints, and how we attempted to evaluate our various prototypes.

8.2 Approach

The goal of this Ph.D. work is to provide social science researchers with an interactive visualization for exploring large social networks. We used three methodologies to validate our work and present them in this chapter:

- *A second participatory design workshop:* after the initial prototypes and informal feedback collected during the implementation process, we conducted a second participatory design workshop. We collected feedback from our real users and gathered insights for future work.
- *Controlled experiments:* we evaluated specific representations and interaction techniques through 4 controlled experiments. We attempted to measure quantatively their readability and performance on a set of tasks and datasets. In this chapter, we present our choices for selecting appropriate tasks and explain our strategy to create representative datasets.
- *Case study:* to show the benefit of using our system as a whole, we decided to perform a case study. We selected a dataset of interest to the whole HCI and information visualization field and present the results in the next sections.

We aimed at performing an additional longitudinal study but faced two challenges: our users limited time and availability and the tradeoff of providing them with a system finalized enough to be usable while not requiring an extensive amount of implementation.

8.3 Second participatory design workshop

The second participatory workshop we ran with our users addressed two questions: “How would you like to explore your data?” and “How would you like to present your findings?”. Almost all of the participants selected the first question on exploration.

We initiated the session by presenting a panel of recent existing systems for analyzing (social) networks [1] including AskGraph-View [AvHK06], SocialAction [PS06], NetLens [KPLB06], SemanticSubstrates [SA06], TreePlus [LPP$^+$06], TimeTree [CSP$^+$06] as well as our prototypes: MatrixExplorer, MatLink and NodeTrix. While several of the participants were already familiar with our research, we presented the whole panel of tools to show them the larger range of possibilities. Similarly to the previous workshop, the presentation was followed by a brainstorming in small groups and the video prototyping of a subset of ideas. We present here the major trends of the results.

Using our prototypes We had very positive feedback on MatrixExplorer and NodeTrix, with participants asking when they would get both pieces of software. In the case of NodeTrix, a small group of participants envision how they would like to use it and proposed an extension. Figure 8.2 presents the video prototype illustrating their idea.

A step further The other prototypes went a step further in the analysis of social network data: the analysis of time-related data and of heterogeneous sources of data.

[1]Workshop presentation and prototypes available at `http://insitu.lri.fr/~nhenry/SocNetWS2/index.html`

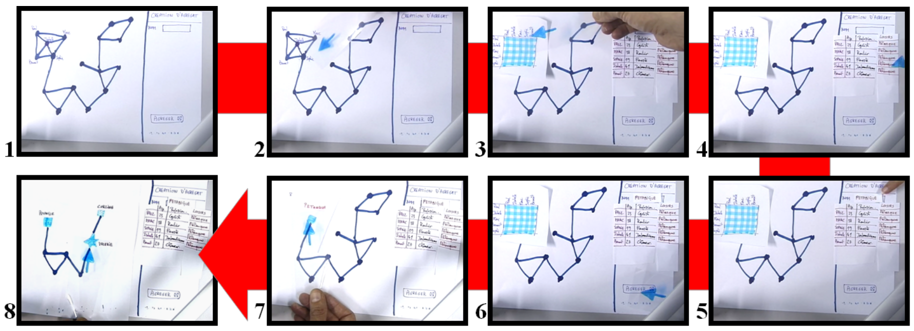


Figure 8.2: Video prototype illustrating the aggregation of portions of a network using NodeTrix. **1-6.** The network is first displayed as a node-link diagram. A subset of nodes is identified and aggregated into a matrix. **7.** Once the community identified and labeled, it can be reduced to a minimal size. **8.** An extension could be to let user create representative icons, such as the star for a star pattern community or a rectangle for a clique.

Exploring time-related data Several of our participants deal with documents evolving through time. For example one of our participants works for the French archives and studies how the government organization changed before and after the revolution. Several other participants are historians and deal with many administrative documents depicting wedding, divorce or birth certificates. In both cases, the data evolve with time and contain multiple types of relations. Thus, we collected several video prototypes depicting how users would like to explore data through time using several coordinated views to compare two networks at different points of time and/or with different types of relations. These prototypes were strongly inspired from SemanticSubstrates [SA06] and NetLens [KPLB06]. Our participants aimed at using simple representations and query them interactively.

Towards visual analytics Several of our participants explained the problem they faced when analyzing several heterogeneous sources of data such as documents, social networks extracted from these documents and pictures or schemas. They showed several examples on how they would like to retrieve documents and analyze them while navigating the social network of the extracted entities. These video prototypes are leading towards a more visual analytic process and towards systems such as Jigsaw [SGLS07], handling several views (social networks, timeline, document views) to analyze a whole set of heterogeneous sources of data.

This second participatory design workshop showed how our users envision using our prototypes to visualize social networks and embed them in larger scale systems able to handle time-related data and heterogeneous sources of data. The workshop outcomes constitute an informal validation of our prototypes as well as a base for future collaboration.

8.4 Controlled Experiments

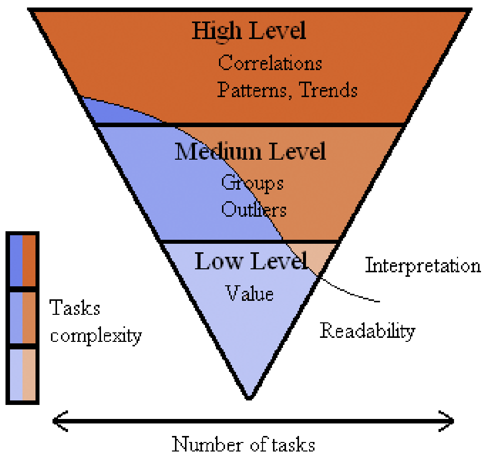
In this Ph.D., we performed four controlled experiments to evaluate the visualization and interaction techniques we designed to explore social networks:

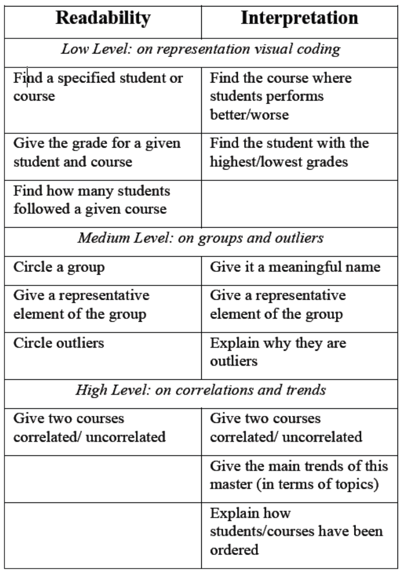
1. Evaluating two matrix reordering algorithms and their effect on human understanding (chapter 4);
2. Evaluating MatLink against node-link and matrix-based representations (chapter 6);
3. Evaluating Mélange against single screen and split-screen for navigating through large visual spaces (chapter 6);
4. Evaluating the effect of node duplications on human understanding (chapter 7).

For each of these experiments, we faced the challenge of selecting representative tasks and datasets. In the following sections, we explain our choices.

8.4.1 Selecting tasks

Our first experiment (matrix reordering) consisted in evaluating how matrix reordering affects the understanding of visual data table. After several interviews with novice users and visualization experts, we ended up with a set of tasks and we organized them hierarchically. We decomposed the tasks in three levels (high, medium, low) further distinguishing readability from interpretation. An initial schema of our task hierarchy is presented in Figure 8.3. We selected tasks from each level evaluating both readability and interpretation. We ended up with more than thirty questions (tasks), half of which were multiple choices, half open for comments or explanations. Three additional tasks consisted of directly annotating the visualization.





Readability	Interpretation
Low Level: on representation visual coding	
Find a specified student or course	Find the course where students performs better/worse
Give the grade for a given student and course	Find the student with the highest/lowest grades
Find how many students followed a given course	
Medium Level: on groups and outliers	
Circle a group	Give it a meaningful name
Give a representative element of the group	Give a representative element of the group
Circle outliers	Explain why they are outliers
High Level: on correlations and trends	
Give two courses correlated/ uncorrelated	Give two courses correlated/ uncorrelated
	Give the main trends of this master (in terms of topics)
	Explain how students/courses have been ordered

Figure 8.3: Task hierarchy and tasks for the experiment.

Results are reported in chapter 4. The main problem we faced was the cognitive effort required from our participants over a rather long period of time (two sessions of one hour and a half in average). We attempted to motivate our users by selecting data in which they could be potentially interested and provide them with a familiar technology to use (digital pen and paper). The participants we recruited did invest some time and effort in analyzing the visualizations. However, their motivation significantly decreased during the experiment and the results show a lot of disparities. From this experiment, we learn two lessons:

· Complex tasks that are subject to interpretation should be avoided due to the variability between participants in term of knowledge, motivation and tiredness. These tasks are more suited to qualitative study performed with domain experts (case studies or a longitudinal studies).
· The number of tasks should be kept to a minimum and the results should be easily controllable and explainable to avoid causing the participants frustration.

Using generic tasks to be able to compare our results to other systems is also important. Therefore, we collaborated with other researchers on a common task taxonomy for graph analysis [PLP^{+}06]. We ended up with several categories of tasks: on the graph topology, on the nodes and links attributes, on navigation and browsing, on overview and higher-level analysis (such as finding trends in the data or comparing two graphs).

From these four categories, we selected only topology-based tasks. The results of these tasks are easily controllable, their level of complexity is reasonable and we remove any interpretation issues. We selected a set of generic tasks (performed on any graph) and a set of tasks dependant on the social network analysis field, but associated with formal graph theory measures (thus easily controllable). Thus the tasks we used in the following experiments are:

· *Connectivity tasks* (generic): find the common neighbours of two nodes, find the shortest path between two nodes.
· *Communities´ tasks* (social network analysis): find a clique, find the largest clique, find the two most connected cliques.
· *Central actors tasks* (social network analysis): find the most connected node, find an articulation point.

8.4.2 Selecting datasets

Performing controlled experiments raises the problem of finding representative datasets. In our case, as we selected topology-based tasks, our problem was to find social networks with representative structure (attributes of actors or relationships being irrelevant). Possible solutions are:

1. Selecting one or two real datasets or benchmarks, hoping they are representative.
2. Selecting a good number of datasets, attempting to match various categories.
3. Generating synthetic datasets with well-known characteristics shared by social networks.

While the generalization of results obtained on one or two datasets (1) is questionable, attempting to select more datasets (2) makes the experiment tedious for both experimenters and subjects (as it requires either a higher number of subjects or a longer experiment duration). With the third method, one should generate datasets with a controlled set of properties and evaluate the systems knowing the properties in advance. (3) should then eliminate the biases that are present in existing datasets and eases the replication of experiments. Therefore, we first attempted to generate synthetic random datasets.

Reviewing the literature in social network analysis, we classified the networks in three categories: tree-like structure, table-like structure (complete graph) and small-world structure (globally sparse and locally dense). While generating tree-like and table-like graphs is relatively straightforward, generating graphs with a small-world structure is still a research topic for computer scientists and physicists.

In Figure 8.4, we show our attempts [2] at using several popular generators publicly available in Pajek [dNMB05] and JUNG [FOS$^+$05]. We visualize several of these synthetic networks using node-link diagrams and matrices and show how their artificial nature is easily distinguishable (in light of the structure of real social networks presented in Figure 8.5). Thus, we present our method to create datasets suitable for evaluations from real-world ones.

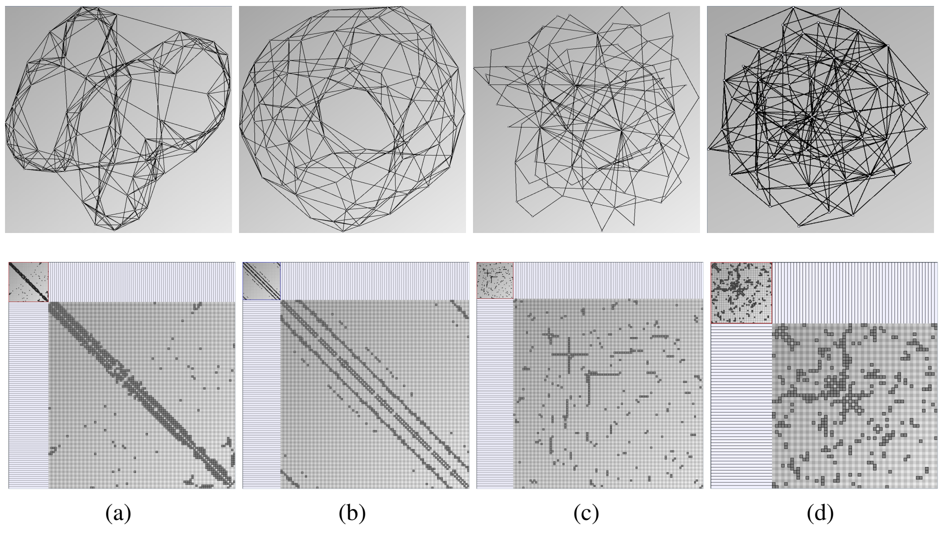


Figure 8.4: Examples of synthetic networks generated with (a) Watts Beta Small World Generator, (b) Kleinberg Small World Generator and (c,d) Barabasi-Albert Generator.

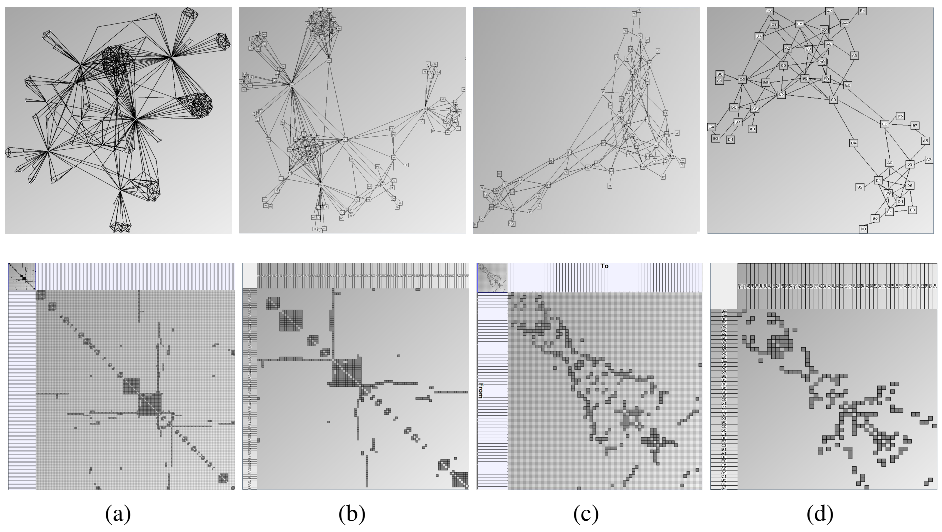


Figure 8.5: Examples of real and filtered networks represented both as node-link diagrams and matrices. (a) Coauthorship network, (b) Filtered coauthorship network, (c) Companionship network (d) Filtered companionship network.

[2]In this dissertation, we only show a subset of the dataset generated, the complete set is available at `http://www.infovis-wiki.net/index.php/Social_Network_Generation`

From real networks to experimental datasets The problem with using real datasets is the difficulty to have them all match a given property. For example, in the MatLink experiment (chapter 6), we needed datasets that could be represented entirely on the screen, thus constituted of a number of vertices slightly over a hundred. To fix this problem, we then decided to filter existing datasets while preserving their characteristics:

1. we compute a set of properties on the original graph: density, clustering coefficient, average path length and degree distribution (section 2.1.1);
2. we generate several versions of the original graph filtered to reach the given size;
3. we compute the same set of properties on the filtered graphs;
4. we select either the filtered graph with the closest property values, or the one that reaches a certain level of precision defined in advance.

When filtering a high number of vertices, it is not always possible to preserve some of the properties (such as a high clustering coefficient with low density or a power-law degree distribution). However, the results are still satisfying compare to the automatic generator we tried. Two examples of filtered networks are presented in Figure 8.5 along with the original ones.

8.4.3 Towards an ecological validation

In this work, we used controlled experiments to validate the readability of our representation. We first selected a set of topology-based tasks, keeping them low-level and avoiding interpretation issues. We then carefully selected real-world datasets of various structures. We altered them to suit our experimental conditions but ensured they kept their major structural properties (such as density or clustering coefficient). Using the above conducted experiments, we collected quantitative evidence of our representations readability compare to standard techniques. To complete the validation of our entire interactive visualization system and attempt to understand its benefit when analyzing real networks, we decided to perform a study in more realistic settings.

We initially opted for running a longitudinal study (MILCS style [SP06]) with three or four of our users. However, these methods require a large investment from both designers and users. On one hand, we would need to provide them with a system usable enough, that includes most used features of their familiar analysis tool, and the technology to collect the information during the study. On the other hand, few of our experts could invest a large quantity of their (expensive) time without insurance of results. Thus, we chose to perform a case study. We selected a dataset for which we could be expert analysts: analysis of 20 years of publication data in our field.

8.5 Case study: 20 years of publication data in HCI

The two primary components of this work were data collection, cleaning and processing followed by visual exploration of the resulting datasets. In fact, these occurred in numerous stages and cycles. Often it was the visual exploration that revealed faults with the data cleaning or suggested new data to collect or combinations and calculations that would be useful to explore.

8.5.1 Data Collection and Processing

We restricted our analysis to the four conferences CHI, UIST, AVI and InfoVis for a variety of practical reasons. First, the Metadata of the first three is managed by ACM, is publicly available in a usable format and is relatively complete and accurate compared with that from other sources.

In contrast, the IEEE Digital Library metadata does not contain reference and citation information. Since this information was added manually up to the year 2003 by the IEEE InfoVis 2004 Contest organizers, we have been able to use it. In contrast, the HCI Bibliography (`hcibib.org`) does not provide references and citations so we have not used it.

Another consideration was limiting the dataset size, which is already near the limit of what many current visualization tools can analyze. We also considered the selected conferences as a good overview of the HCI field. In particular, while data from the ACM Computer-Supported Cooperative Work (CSCW) conference would have been interesting to include, we opted not to because two analyses of this community have been published, one in 2004 and another in 2006 [HFB$^+$04, JSGF$^+$06]. Finally, we restricted our dataset to conference data because they are considered as the most important form of publications by many HCI practitioners. Furthermore, journal articles and books are sufficiently different in their time scale and impact on the community that we felt comparisons between the two would be difficult.

While it may be argued that the AVI conference is less significant in comparison to the other conferences selected for this analysis, we picked it due to precisely this reason: it is a young and upcoming conference which exhibits many of the typical patterns of newcomers. The analysis shows signs of a still-immature conference, such as unstable co-authorship network and unformed communities.

Data Collection

We began with the InfoVis 2004 Contest dataset, which covers the InfoVis conferences from 1995 to 2002. The data originally provided by the IEEE Digital Library (DL) had been extensively cleaned and corrected by the contest organizers. We used a version with additional curation provided by the Indiana as part of their contest submission. The datasets for the other 3 conferences were provided by the ACM Digital Library: the CHI conferences from 1983 to 2006, the UIST conferences from 1988 to 2005, and the AVI conferences from 1994 to 2006 (AVI is held every 2 years). The ACM DL provided an XML file for each conference with the title, authors, and other information about each article, including the unambiguous ACM identifiers of the articles it references wherever the curators were able to resolve them (see Figure 8.6).

Figure 8.7 shows an overview of the timeline of the four conferences as well as the coverage of the publication data used in this case study. Note that data is missing for AVI 2002 and that the coverage of InfoVis ends in 2002.

We only collected information for full-length papers, excluding short articles, poster and demo submissions, contest entries, keynotes, panels, and so forth. For each conference, we collected the

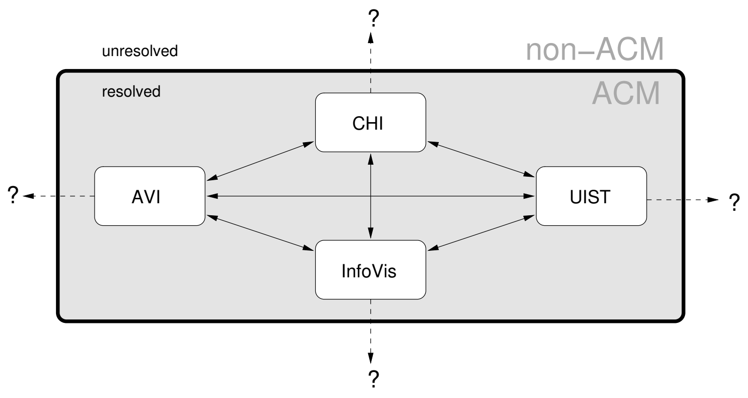


Figure 8.6: Resolved and unresolved references. References between the four conferences are resolved completely. Other references contained in the ACM DL are resolved with a unique identifier but no other information. References outside the ACM DL are not resolved.

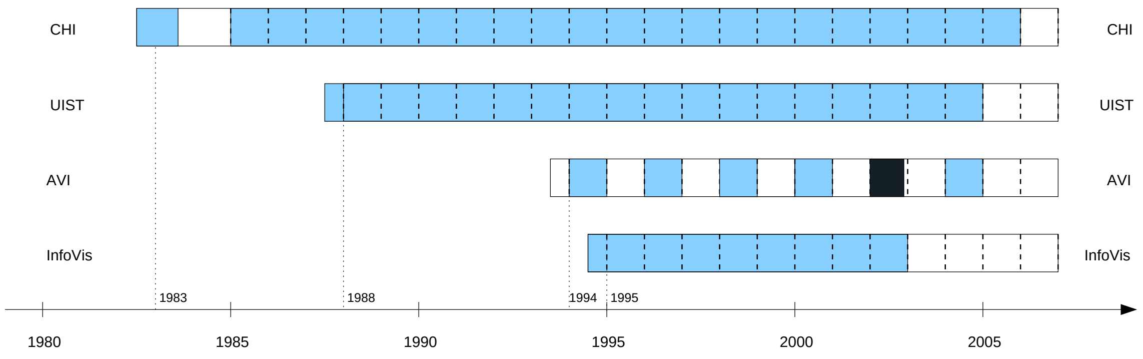


Figure 8.7: Timeline of the CHI, UIST, AVI and InfoVis conferences. The solid bars indicate the coverage of our publication data; AVI 2002 is missing.

following information: proceedings ACM identifier, conference ACM identifier and its acronym, proceedings title, proceedings description and copyright year. For each article, we collected the following information: article ACM identifier, title, subtitle, list of keywords attributed by the authors, abstract, page numbers in the proceedings, a list of citations to the article with the citing paper's ACM identifiers where identified, a list of authors, and their authoring sequence number. Self-citations were not removed from the dataset. Finally, for each author we collected their ACM identifier, first, middle and last names.

Data Processing

It is important to note that our dataset is incomplete. First, the ACM metadata is incomplete, especially for early conferences. While it does contain basic information such as title, authors, and dates for each conference article, not all references are present, and not all references that are present have been unambiguously resolved. Secondly, because we only processed files from the four conference series, even identified articles from other conferences have missing detailed information, such as authors. Because such missing data could easily have misled our analysis, considerable caution is advised in interpreting both the visualizations and the statistics.

In addition to missing information, the datasets contain duplicated author identifiers, a common problem when dealing with publication data. Author names may be misspelled or use initials instead of full names, or authors may change their names or use different combinations of formal and informal names and initials on different papers, producing multiple identifiers we call *aliases* for a single person. Our efforts were aided by the recently-developed *D-Dupe* program from the University of Maryland [BLGS06]. D-Dupe uses both name and co-authorship similarity in an interactive process to resolve aliases. We divided our de-duplication process into four stages, from the easiest to the more complex cases.

· We merged authors according to an alias attribute previously computed for the InfoVis 2004 Contest. Katy Börner and her students had cleaned this dataset manually. For each of the 109 authors with aliases, they added an attribute to the original identifier in their database.

· We merged authors with exact similarity of last, middle and first names. Authors who used only a last name and a first name were merged according to 2 criteria: if they had at least one co-author in common, and if their name subjectively and/or objectively did not seem to be common. (For example, two "Pedro Szekely"s would have been merged, but not two "J. Smith"s.) To define if a name was common or not, we used our own knowledge in addition to the search feature of D-Dupe. In the above example, for instance, a D-Dupe search on "Szekely" returns only 4 results, against 39 for "Smith".

· We merged authors with similar last name and more than one co-author in common. In that case we also used our knowledge of the field to avoid merging, for example, husband and wife Gary M. Olson and Judith S. Olson who have 7 co-authors in common. Still, we merged the 7 identifiers of William Buxton (as W. Buxton, William Buxton twice, William A. S. Buxton, Bill Buxton twice and B. Buxton).

· Finally, we had to deal with more complex cases: two persons with similar last names (relatively common) without any co-authors in common. To solve that case, we searched for information on the Web, looking for home pages and list of publications. Interestingly, in these cases the results were almost equally divided: half turned out to be the same individual collaborating with different teams, and half were different persons. This result implies that such cases will be difficult to resolve automatically.

The process took almost a day. We stopped when name similarity was less than 80%, being aware that duplicated authors still remained. We found a total of 516 aliases over the 6143 authors (8.3%). The maximum number of aliases was 7 apiece for Ben Shneiderman and William Buxton.

8.5.2 Visual Exploration Method

The collected results from the above data collection and processing produced a graph with 26,942 vertices and 118,865 relations. This graph contains three types of vertices: 332 conferences, 5,109 authors and 21,501 articles. Of the articles, 18,573 are missing some information, and 4,797 do not even have an ACM identifier. The network has three types of relations: 3,254 edges linking articles to the conference they appeared in, 9,030 edges linking articles to their authors, and 85,319 edges between articles (i.e. references). From these three, we computed additional relations: author-author for both co-authorship (10,631 relations) and citation, and conference impact (citations aggregated at the conference-conference level).

As stated in the introduction, we used an exploratory process to analyze the cleaned HCI publication data. This process does not require a priori hypothesis or questions to evaluate, but seeks to generate and evaluate hypotheses — about global and local trends and outliers — interactively during the exploration.

Visualizing and interacting with this data requires a system able to handle large graphs. Our analysis primarily used MatrixExplorer [HF06a] and NodeTrix [HFM07] (both built upon the InfoVis Toolkit [Fek04]), GUESS [Ada06] (based on JUNG[3]), and the R statistical package [R D06].

We used GUESS and its powerful scripting language to query graphs and manipulate their attributes. However, handling these large node-link diagrams induced some delay. Getting a readable overview of the full graph was also a challenge. For this reason, unlike most other studies, we choose to use an adjacency matrix representation of the graphs to explore the data in ways that would have been difficult otherwise.

We used the MatrixExplorer and NodeTrix tools to provide us with both matrix and node-link representations of the graphs. These systems offer interactive tools to manipulate matrices (filtering, ordering and visual variable affectations) and allow for synchronized node-link diagrams. They also suffer some delay handling the full graph (especially to compute reordering), but the readability of the final representations was far better than with a node-link diagram.

We used matrix representations to explore the graph, following an iterative exploration process that we will attempt to describe. We loaded our full dataset and filtered it by types of vertices, group of conferences and/or type of relations. For example, we extracted the co-authorship networks for InfoVis conferences, the citations network across conferences, or the citations network of CHI authors. For each of the filtered graphs, we then visualized its macro-structure: the connected components size and number followed by the analysis of each component independently. For each component, we interactively applied reordering, filtering, and visual variable affectations. We ended up with a set of insights such as communities or patterns for each filtered networks. At this stage, we created node-link visualizations of filtered graphs for each insight we found interesting. We fine-tuned the node-link visualizations in turn to get readable representations illustrating our findings.

At each stage, our analysis raised many additional questions. Organizing the exploration process to avoid diverging in several directions was difficult; since we were tempted to follow each insight independently. We recorded all the interesting questions but attempted to explore in a breadth-first manner instead of analyzing every individual question in depth, which often would have required time-consuming investigation on the Web or interviewing experts.

Although adjacency matrices were effective for exploration, presenting them on a static page with limited space is a challenge. Therefore, we present both zoomed views of our large matrices and node-link diagrams of filtered networks to illustrate our analyses.

[3]`http://jung.sourceforge.net`

8.5.3 Results

This section describes the results of our visual exploration process. It primarily documents many observations, tentative explanations and questions for further analysis.

Case study contents

Overview

The first few subsections that follow present fundamental components of the HCI field and our datasets: its highly-cited authors and articles, the general characteristics of the four major conferences (CHI, UIST, AVI and InfoVis), and also an analysis of the evolution of their topics over the years.

Our relatively simple data analysis of this data, using primarily simple statistics, histograms and plots, explained many general characteristics of the data, but it also raised many additional interesting questions. We present a subset of these additional results we actually explored, and also try to give a feeling for a variety of additional queries that can be performed by filtering, combining, and correlating the data.

The last two subsections are a more in-depth analysis of two networks derived from the original data: citation networks for conferences, articles and authors, and co-authorship networks between researchers. Together, they provide a wealth of data about the structure of the HCI community: the influence of different researchers, institutions and conferences; the groups of researchers who collaborate strongly and the wider-ranging collaborations between them.

Authors

We used three measures to identify important researchers of the field (Figure 8.9). We collected the total number of articles accepted to define the *most prolific authors*. We computed the number of citations to researchers' articles to define the *most cited researchers*. Finally, we computed the social network analysis measure of *betweenness centrality* for each researcher in the largest connected component of the co-authorship networks for each conference and for all the conferences together. This measure is an attempt to determine how central an actor is by counting the number of shortest paths between other authors that go via this researcher.

The common social-network concept of "betweenness-centrality" in this context must be interpreted carefully: it may not necessarily indicate success. For example, researchers who move from one institution to another or students who graduate and take a job elsewhere become more central not because of their work *per se*, but because of geographic (topographic) factors. Nevertheless, very central actors do link communities and are therefore perceived as central.

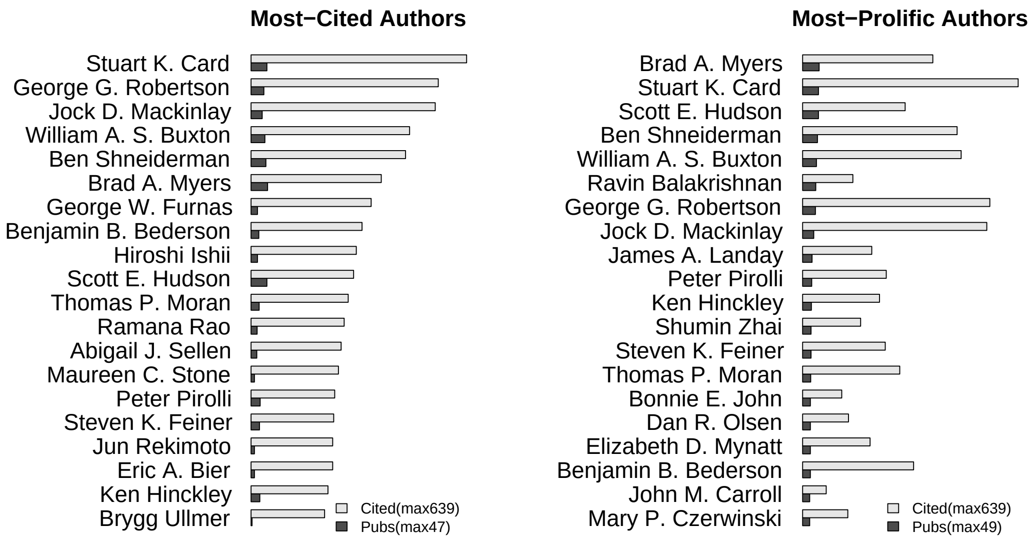


Figure 8.8: Statistics for authors

Citations and Number of Articles When examining Figure 8.9 and the general statistics on authors, we observe a correlation between the number of citations and the number of articles. In general, the most cited researchers are also the most prolific, implying that they are actively contributing to the field in terms of quality and quantity. The five most-cited include the trio of Stuart Card, Jock Mackinlay and George Robertson (abbreviated as Card-Mackinlay-Robertson), followed by William Buxton and Ben Shneiderman.

We notice two exceptions to this trend: Edward Tufte and Ravin Balakrishnan. Edward Tufte has only two referenced works (both books), but he is cited almost forty times. This is easily explained: Tufte has few publications in this field because he is not an HCI researcher, but these books are seminal works for information visualization that are frequently cited by articles in the field. Ravin Balakrishnan is exceptional in the opposite direction: the sixth most prolific author with almost forty published articles, he is nevertheless cited approximately 50% less than similarly-prolific authors such as William Buxton or George Robertson. One interpretation might be that much of his work relies on specialized technologies unavailable to the majority of HCI researchers, which limits the number of citations until and if they become more generally accessible. Another is that despite his high number of publications, he is much younger than the other most-cited researchers, and his articles did not have as much time to be cited.

Centrality Each conference has a different set of most-central researchers. For the CHI community, they are William Buxton, Thomas Landauer and Thomas Moran. For the UIST community, Scott Hudson is the most central researcher, while Takeo Igarashi, Ken Hinckley and Brad Myers have a similar betweenness-centrality. For InfoVis, Ben Shneiderman and Stuart Card are almost equal as the most-central figures. AVI has a very disconnected network with many small connected components, the largest of which contains only about twenty researchers. Therefore, we cannot rely on centrality measures to identify a particular researcher. Our conclusion is that AVI does not yet have a stable set of communities.

Considering the centrality of the aggregated conferences, notice that all the central authors of CHI, UIST and InfoVis are in the top twenty except Takeo Igarashi. This would imply that he does not collaborate much with the other central figures of HCI, and in fact he is more active in the interactive 3D community than in HCI.

Collaboration and influences Figure 8.10 shows the collaboration between the most referenced researchers in our dataset. Figure 8.11 shows the influences of these researchers in matter of citations (who is citing who).

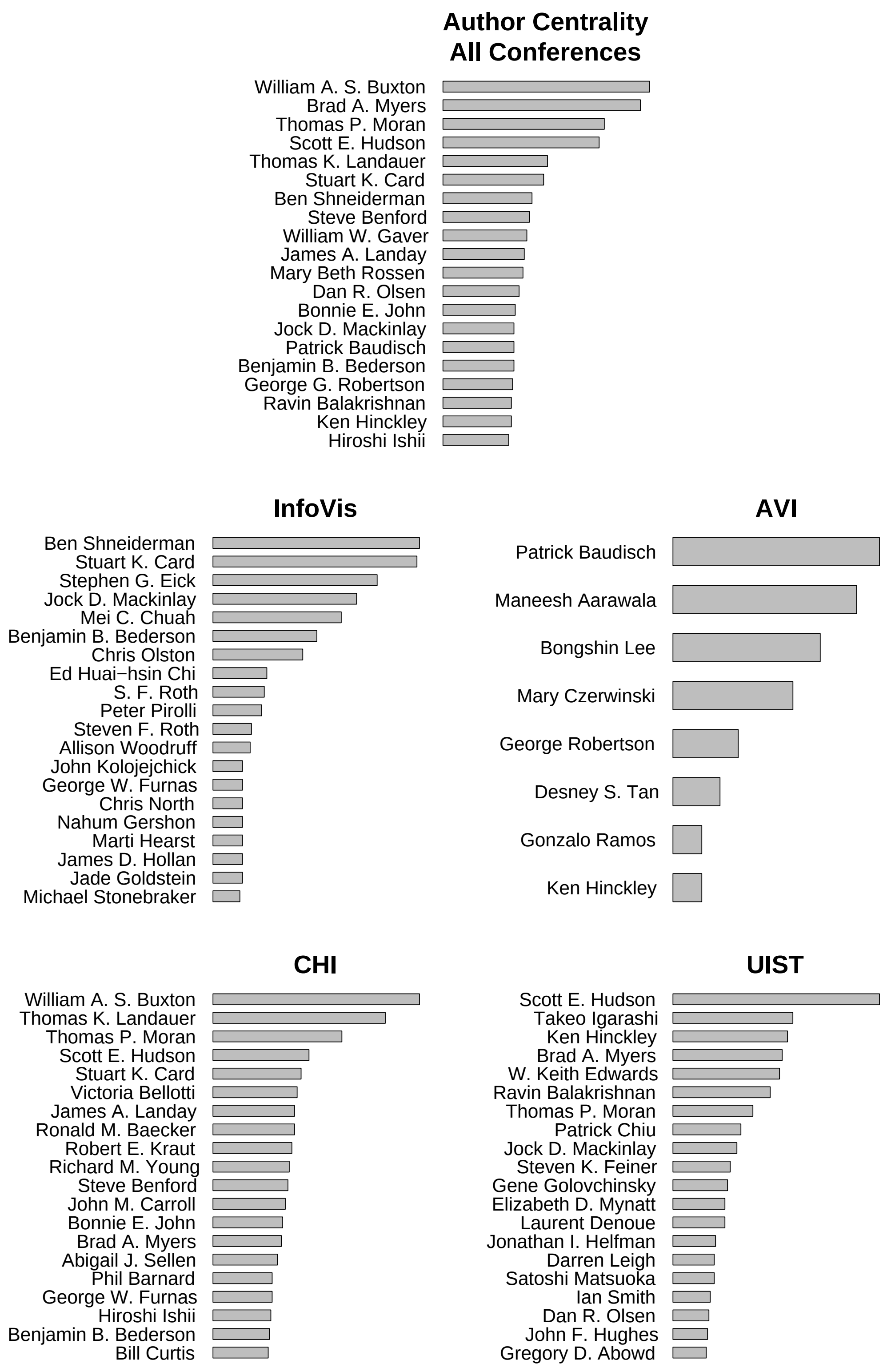


Figure 8.9: Most central authors

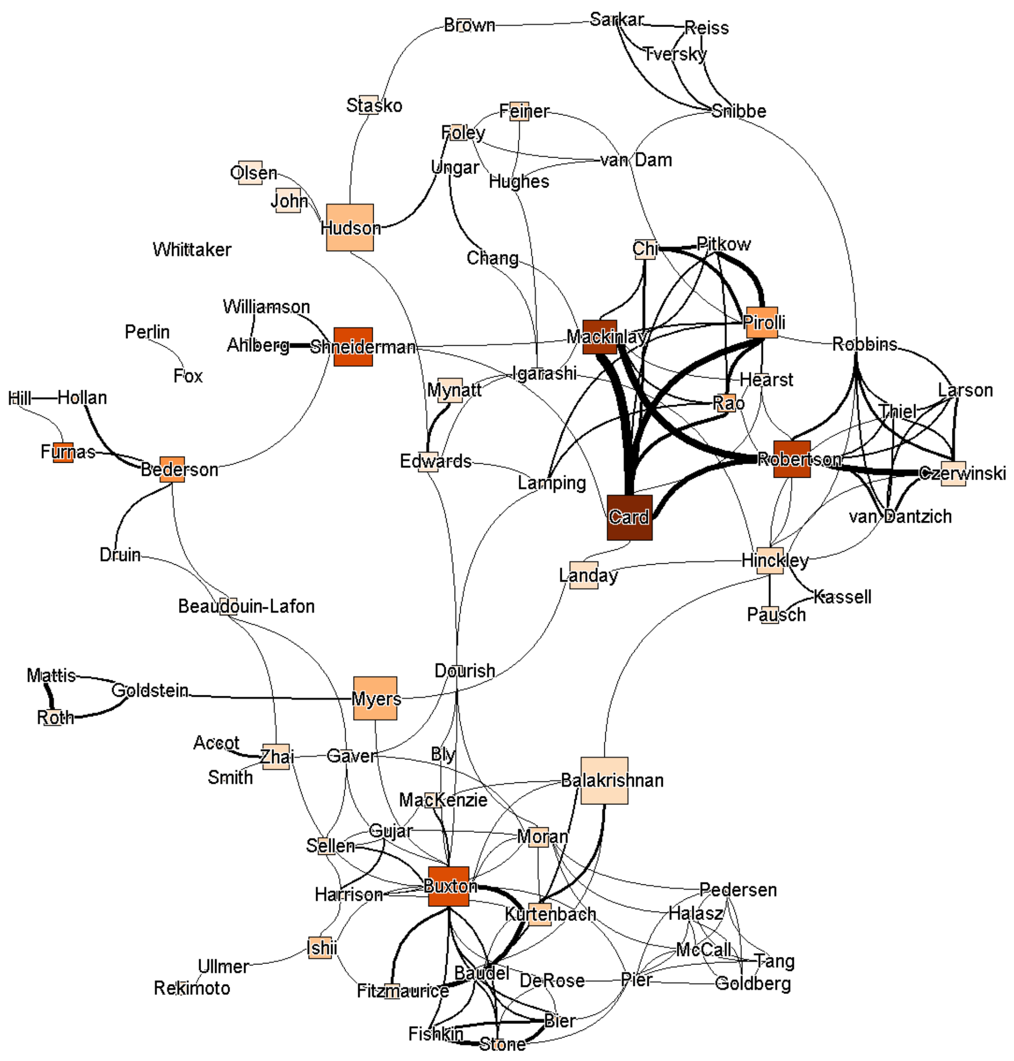


Figure 8.10: Overviews of the HCI field in terms of collaboration (co-authorship). Nodes represent researchers; size indicates their number of articles published and darkness shows number of citations. Links represent co-authorship; link width is proportional to the number of co-authored papers.

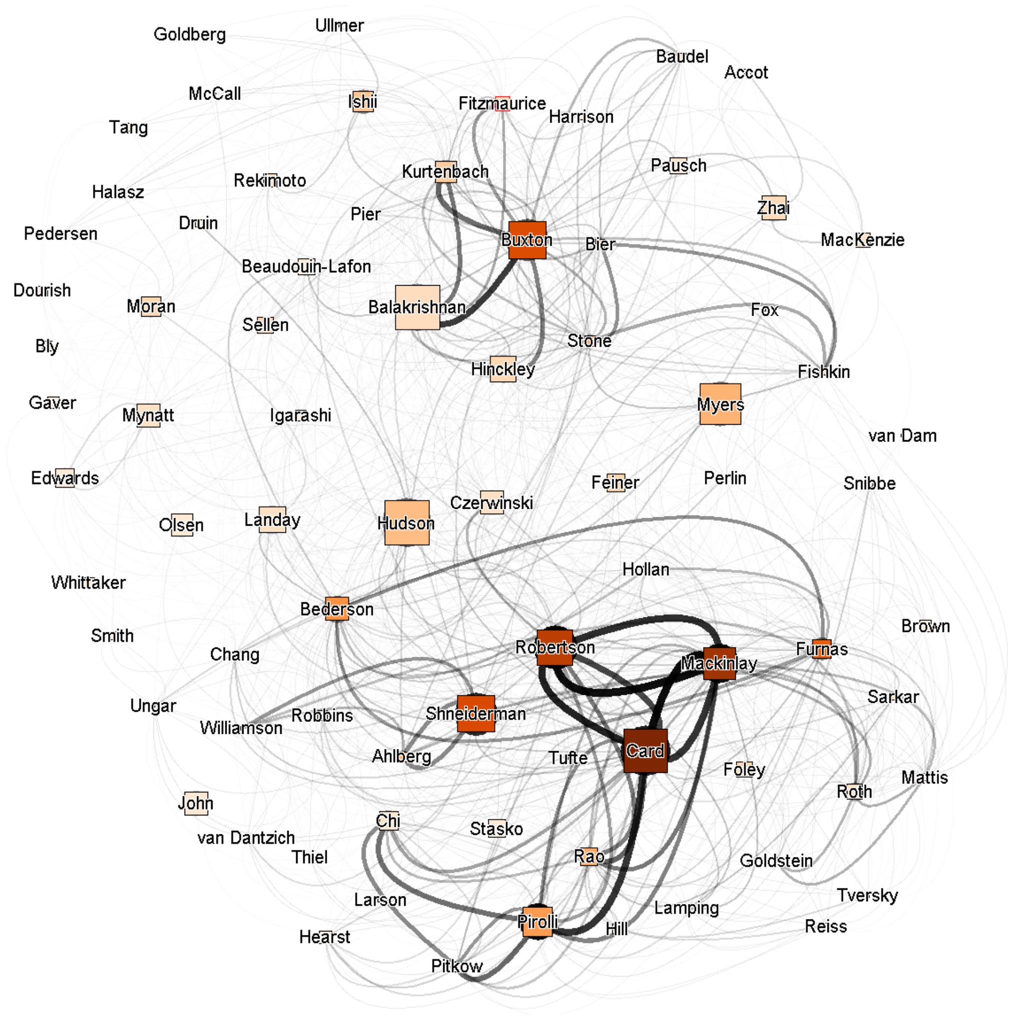


Figure 8.11: Overviews of the HCI field in terms of influence (citations). Each node represents a researcher with its size showing the number of articles published and its darkness represents the number of citations. Links represent citations. Their width is the strength of these relations.

Articles

The two most cited articles across CHI, UIST, AVI and InfoVis are "Cone Trees: Animated 3D Visualizations of Hierarchical Information" [RMC91], published at CHI in 1991 and cited 70 times, and "Generalized Fisheye Views" [Fur86], published at CHI in 1986 and cited 66 times (Figure 8.12).

Sources of Key Articles Articles from the CHI conference are the most heavily cited, representing six of the top ten and seven of the top twenty. Interestingly, browsing the keywords of these articles reveals that the majority deal with information visualization. Moreover, Edward Tufte's book, "The Visual Display of Quantitative Information" [Tuf83], one of the seminal works of information visualization, is the third most cited research work.

While this shows that information visualization is an active topic in HCI, the result should be interpreted carefully; since visualization is the major focus of both the InfoVis and AVI conferences. Interestingly, articles from the InfoVis conference itself appear unexpectedly low in this ranking. The first, "Visualizing the Non-Visual: Spatial Analysis and Interaction with Information from Text Documents" [WTP$^+$95], appears at the 20th position. These low impact numbers are probably partly due to the fact that information visualization as a specialized sub-field is more likely to cite general HCI papers than the reverse.

The ages of the conferences are another key. Not only are authors likely to submit their best work to established conferences, but influential papers often amass citations for many years. Similarly, the first-ranked article of the AVI conference (held every other year since 1992 in Italy, but becoming much more prominent around 2000) appears only at the 43rd position: "Fishnet: a fisheye web browser with search term popouts" [BLH04a]. By contrast, four articles from the also-small UIST conference appear in the top twenty, including one in the top ten: "SATIN: A Toolkit for Informal Ink-Based Applications" [HL00]. Besides its longer history (at 18 years it is the second-oldest), this may also reflect UIST's more general HCI focus.

Another interesting insight is that two articles of SIGGRAPH 1993 are much-cited in HCI (in the 14th and 24th position): "Pad: an alternative approach to the computer interface" [PF93]. and "Toolglass and magic lenses: the see-through interface" [BSP$^+$93]. This could suggest that SIGGRAPH has at least as much impact on the community as internal conferences.

Authors of Key Articles Figure 8.11 shows references among authors of key articles. Some key articles have a single author: George Furnas, Edward Tufte and Jock Mackinlay each individually authored one of the field's ten most-cited articles. However, collaboration seems to be a more reliable route to success. Not only did the trio of Card-Mackinlay-Robertson co-author three articles in the top ten, but Jock Mackinlay holds the record of six articles in the top twenty, and Stuart Card is the single most-cited researcher in the field.

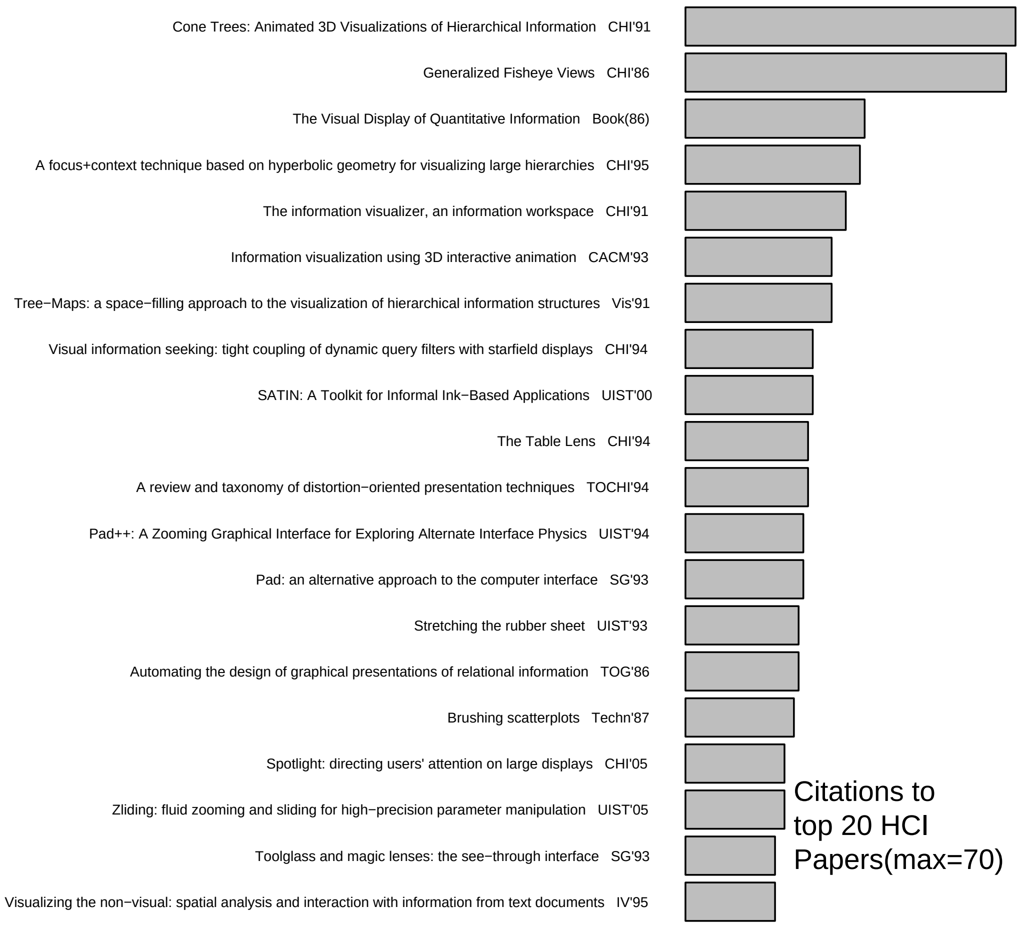


Figure 8.12: Top 20 most referenced articles.

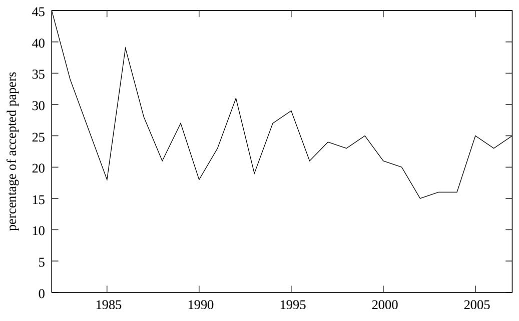


Figure 8.13: Acceptance rate for CHI.

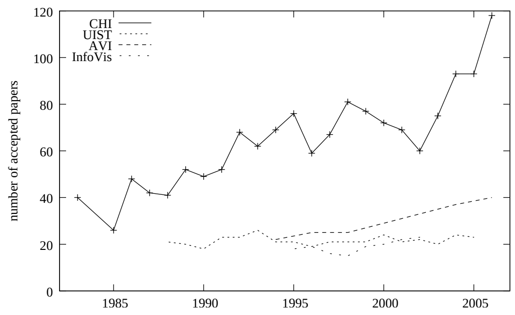
Conferences

For each paper, we extracted its number of *references* to other articles, and the number of *citations* from other articles to it. Then, for each conference we computed the number of articles accepted and the total numbers of references and citations for all its papers (Figure 8.14.) Conferences are grouped by category and ordered chronologically from the oldest to the most recent.

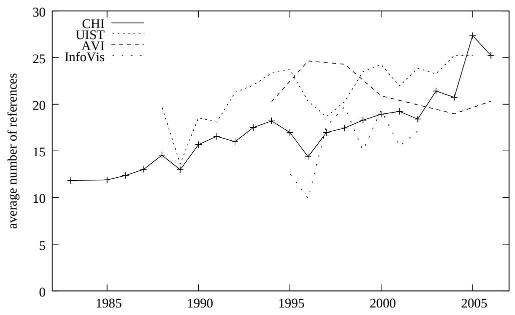
Accepted Articles A global trend for all four conferences is that the number of accepted articles has increased over the years. CHI accepted 60 articles for its first conference in 1983, rising to 151 long articles in 2006, a 2.5-fold increase over 23 years. AVI and InfoVis also slowly increased their number of accepted articles. UIST's pattern was more variable. On the average, it accepts about 30 articles. However, it started with 22 articles at its first conference, doubled the number of accepted articles in 1994; then remained almost stable with an average of 30 articles accepted each year. The only other exception was 2003, its 20th anniversary and the largest UIST conference, which accepted 50 articles. We observed that CHI 91, 92 and 93 accepted more articles than the following conferences: all three accepted over a hundred articles, around 30 articles more than in 1990 and 1994. One could ask if a particular event happened during these three following years (for example, 1993 was the decennial of CHI, and was also a joint conference with the Interact conference), if the submitted articles were of better quality or simply if the program committee decided to increase the number of accepted articles.

Number of References As the number of accepted articles increased, obviously so did the total number of references. However, the average number of references per article also increased. It was stable from 1983 to 1993 with 10 references per article (although the earlier conferences seem to have a high rate of missing references in the ACM Metadata) but increased to 15 references in 1994; then remained stable for 5 years before finally increasing in 1999 to 20 references and remaining stable through 2006. UIST 92 is the only exception with an average of 21 references per article. An interesting observation is that the average number of references evolved similarly for all conferences. Further investigation would be required to define if the number of pages of submitted articles increased or if another factor explains this increase.

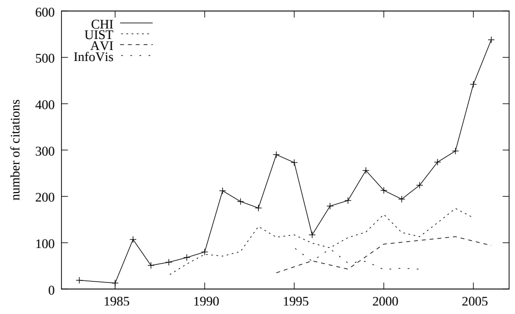
Acceptance Rate and Most Cited Articles The CHI conference published its most-cited articles in 1986 (#1 most-cited), 1991 (#2, 4 and 5), 1997 (#8) and 1994 (#9). However, Figure 8.13 shows that the conference's acceptance rates in those years were relatively high: 39% in 1986 (the highest ever), 23% in 1991, 24% in 1997 and 27% in 1994 — versus its historic average, the lowest being a 15% acceptance in 2002. Typically, a low acceptance rate is an indicator of quality: only strong work should be published if so many papers are rejected. However these results do not concur. Does a low acceptance rate imply a more conservative article selection process that deters or filters out unconventional, ground-breaking articles?



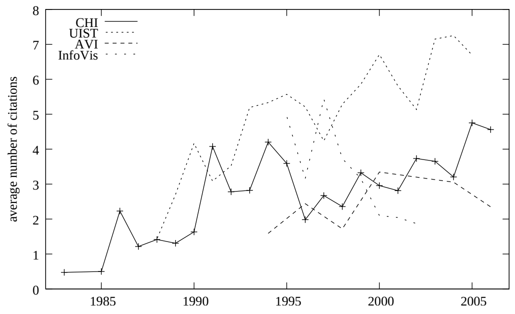
(a) Number of accepted articles



(b) Average number of references per article



(c) Number of citations per article



(d) Average number of citations per article

Figure 8.14: Statistics per conference.

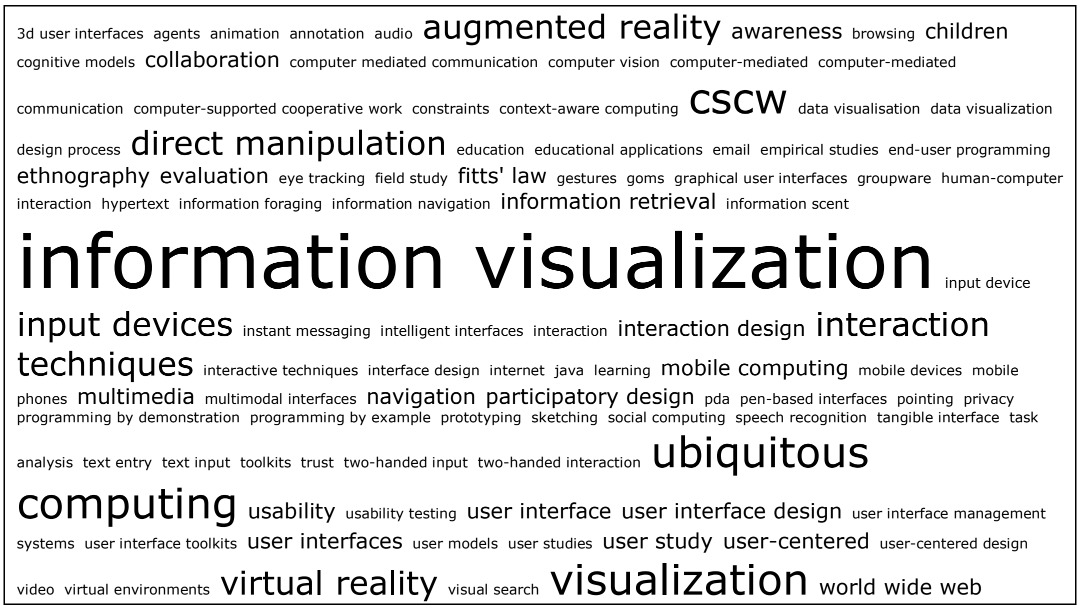


Figure 8.15: Keyword frequency cloud for all four conferences (100 terms).

Keywords

Our data contains information about the additional keywords authors have added to their articles (i.e. beyond the standardized ACM Computing Classification System[4] keywords required for some conferences). These keywords are interesting because they serve as indicators to the ideas and concepts that were current in the scientific communities at different points in time.

Figure 8.15 shows a frequency visualization of the 100 most common terms in the combined keyword corpus for all conferences in the dataset (4,843 unique keywords in total). Here, keywords are scaled in size according to their relative frequency of appearance in the dataset. Looking at this figure, it is clear that "information visualization" (95 counts) is a key concept in the community of those 4 conferences, but that terms like "CSCW" (62 counts), "ubiquitous computing" (57 counts), and "visualization" (52 counts) are important as well.

In Figure 8.16, we see similar frequency visualizations for the 50 most common terms of the individual conferences. We notice that the CHI conference (3,321 terms) has a much wider variety of terms than any of the other three conferences, and it is clear that CHI has a broader scope than the others. Also, the emphasis on information visualization is less pronounced for the CHI dataset, and the most common term here is actually "CSCW" (46 terms as opposed to 38 for "information visualization"). Both AVI (494 terms) and InfoVis (474 terms) are much more focused on visualization. Looking more closely at the individual keywords it seems that AVI has a wider array of general HCI subjects, whereas InfoVis — not surprisingly — focuses on visual representations of different kinds of data. Finally, the UIST (1,206 terms) conference shows a mix of the other three, yet has also a strong emphasis on user interfaces, toolkits, and programming.

Finally, we are also interested in studying the use of these keywords and concepts over time to get an idea of how ideas and trends rise and fall in the history of the four conferences. Figure 8.17 presents a timeline from 1983 to 2006 of the 59 most common keywords for all conferences. Darkness indicates high counts, so we can immediately notice the high emphasis on information visualization and interaction techniques in 2000. Other insights include the introduction of the term information visualization in 1991 (corresponding to the publishing of the three highly-cited papers by PARC at CHI that year [CRM91, MRC91a, RMC91]), the large number of popular concepts that were introduced in 1992, and the late shift to trends such as privacy, ethnography, and, particularly, ubiquitous computing in the 90s.

Of equal interest are keywords that no longer are in use, or which have exhibited periods of revival. For the former category, "user interface management systems" is a good example, appearing only in articles published in 1987 and then never again. The term "constraints", similarly, appeared in 1992 and then immediately went out of fashion. For the latter category, the term "usability" is perhaps the best example. It appeared in the very first CHI conference in 1983; then disappeared; made a strong comeback in 1992, remained prominent for a long time, but has not been seen since 2004.

[4]`http://www.acm.org/class/1998/`

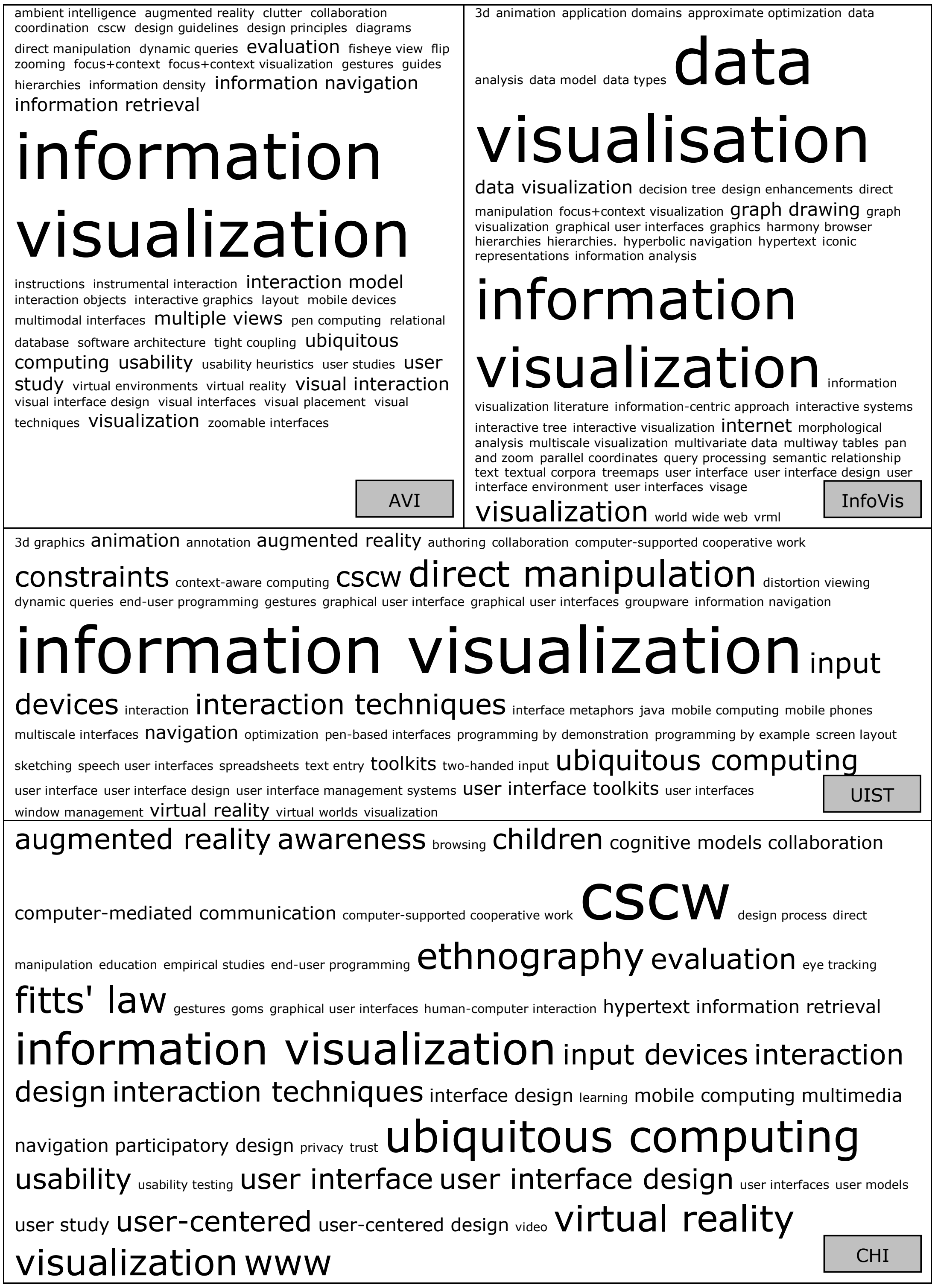


Figure 8.16: Keyword frequency cloud for AVI, InfoVis, UIST and CHI (50 terms each).

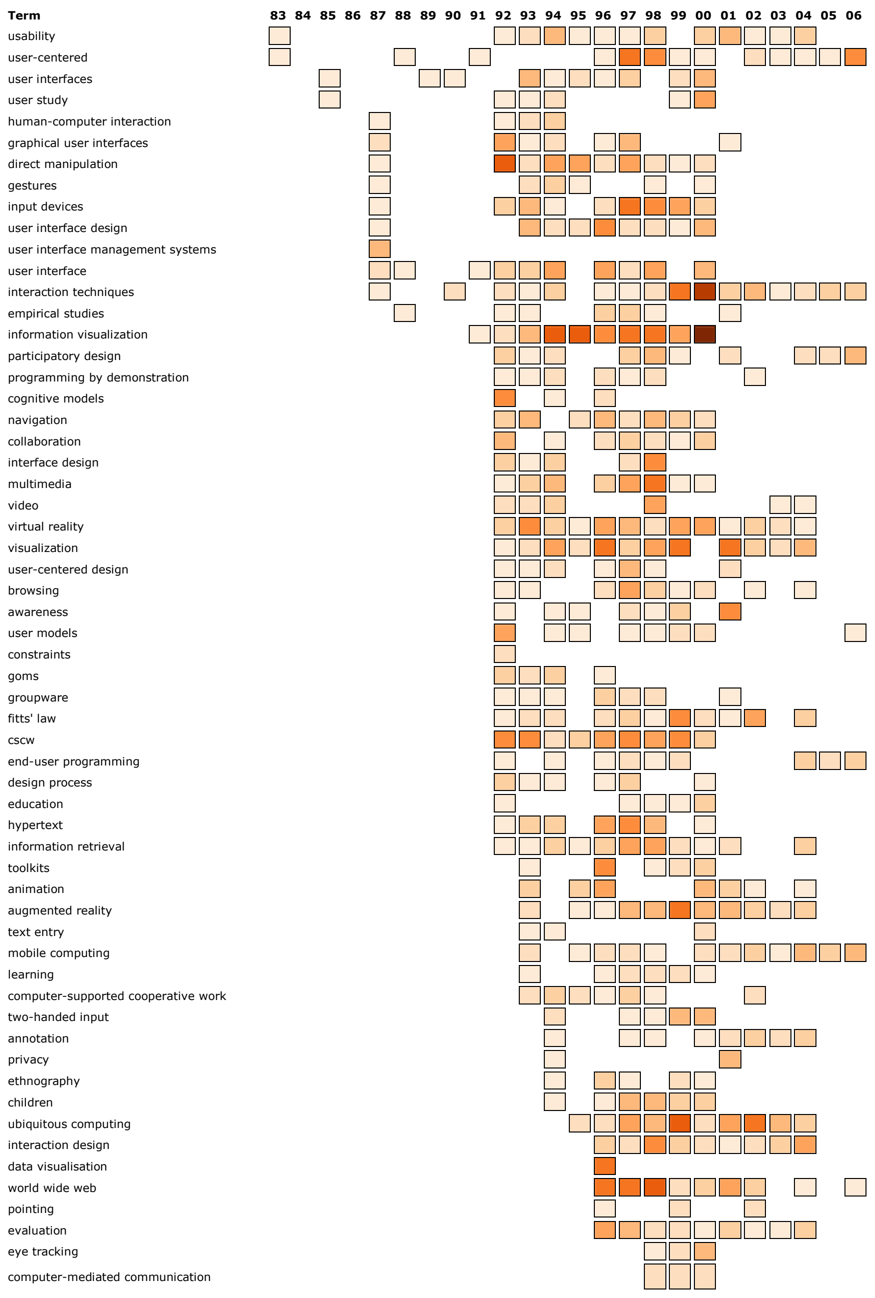


Figure 8.17: Keyword timeline for all four conferences from 1983 to 2006. Terms are listed in chronological order of appearance. Darkness indicates high density.

Citation Networks

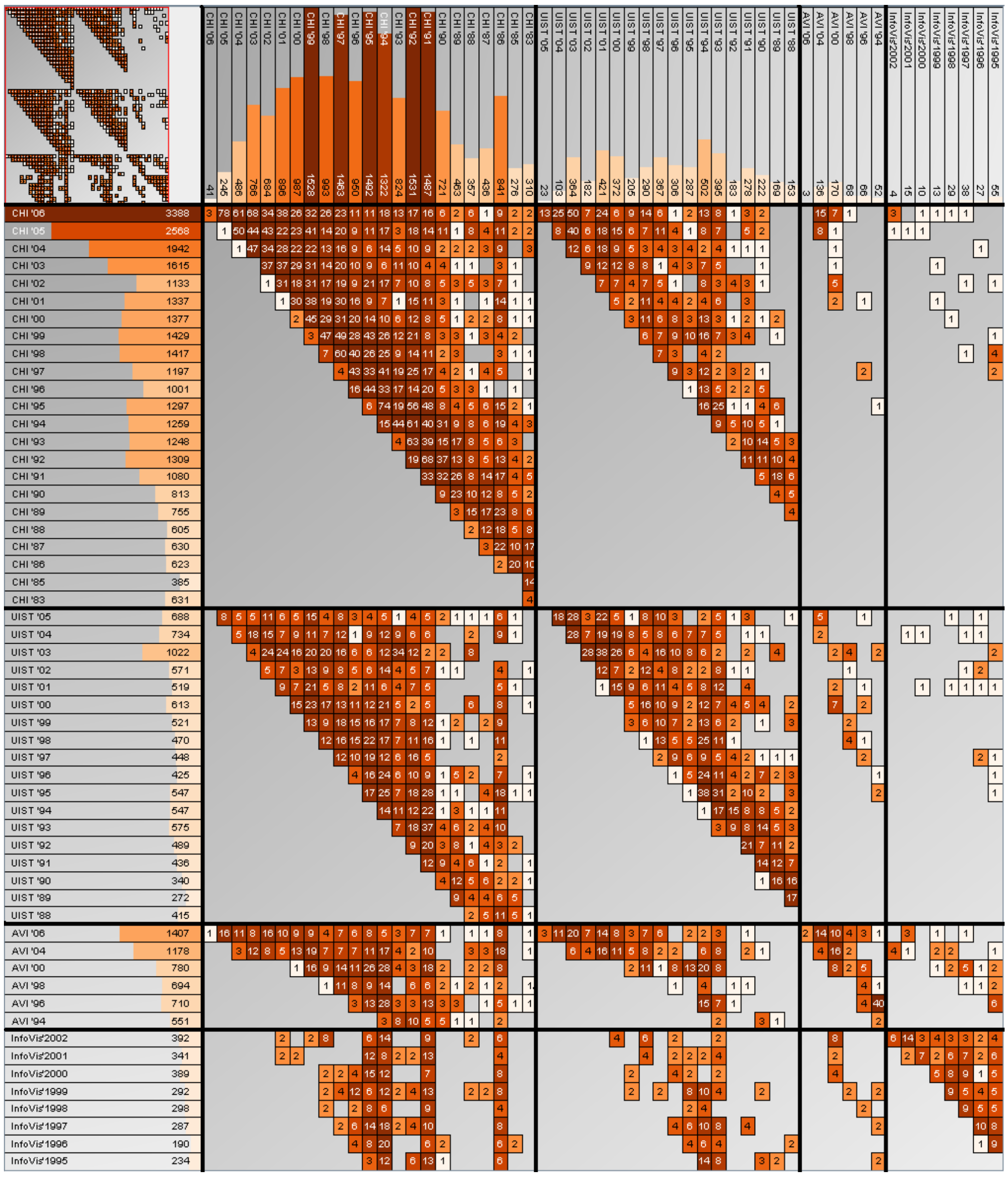
This section analyzes three citation networks: citations between conferences, between articles and between authors. Conference citations show the impact of each conference on the others; article citations highlight key articles and their relationships. The author citation network has the most interesting patterns, because how authors cite each other reveals patterns in the community. Citation patterns reveal many influences, and demonstrate research trends over time.

Citations Between Conferences Figure 8.18a is a matrix visualization of the inter-conference citation network, showing how the conferences reference each other. The four conferences, CHI, UIST, AVI and InfoVis, are arranged on the rows and columns, grouped by conference and then ordered by year, most-recent first. The darkness and numeric value in each matrix cell show the number of citations *from* the conference printed on the row *to* articles of the conference printed on the column. Elements on the diagonal are articles referencing another article in the same year, which are most interesting when they refer to articles submitted to the same conference.

Conference Impact In informal interviews, researchers in the field frequently described the CHI conferences as having the most impact and prestige, pointing to its high number of articles published despite a low acceptance rate and large number of attendees as indicators that articles published at CHI have the most impact in the field. If we define the *impact* of a conference as its number of articles cited by other conferences over the years, we can observe that CHI conferences have indeed had a strong impact on the field. Figures 8.14c and 8.18a show that CHI conferences have a strong impact on the other three. Articles from CHI 99, CHI 97, CHI 95, CHI 92 and CHI 91 represent the majority of references, while CHI 86 has the unique distinction of having been referenced by *every* subsequent conference and year except UIST'03 and CHI'96. In terms of evolution across time, Figure 8.18a shows that a typical CHI conference has a high impact for the six or seven following years, whereas the impact of UIST or InfoVis is only high for three or four years.

Analyzing the impact of CHI conferences on AVI and InfoVis, we were interested to notice that only CHI 86, CHI 91, CHI 94 and CHI 95 have had a strong impact. To analyze this further, we visualized the impact of the CHI articles independently, filtering to keep only the most-cited ones, resulting in Figure 8.18b. Comparing the totals for articles with those for the whole conference brought an even more interesting observation: for at least two of the four high-impact years, virtually all the references from all the InfoVis conferences to a particular CHI conference year were to a single article. Fully 100% (42/42) of the InfoVis references to CHI 86 are for "Generalized Fisheye Views" [Fur86], and 85% (68/80) of the references to CHI 91 are for "Cone Trees" [RMC91]. It is surely significant that so much of the impact of the CHI conference on the InfoVis conference depends on these two early articles.

Average Number of Citations Given that the impact (total citations) of a conference hinges significantly on a few very highly-cited papers, it is interesting to look at the average number of citations per paper in a conference as well. Interestingly enough, as Figure 8.14d shows, according to this metric it is UIST and not CHI papers that clearly have a higher average number of citations than the other conferences. At the other end, the smaller AVI conference, which usually has higher impact than the larger InfoVis, beats it even more dramatically in citations per paper.


(a) Conference citations

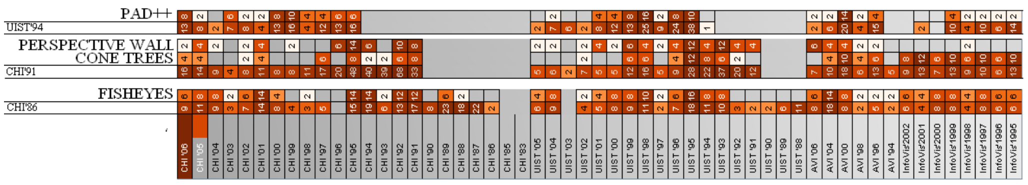

(b) Conference impact

Figure 8.18: Matrix of inter- and intra-conferences citation networks. Conferences are grouped by category and ordered by year. Number of references in rows, number of citations in columns

UIST's higher average citation count comes at a price. Its number of accepted papers is one clue: UIST has accepted only 20-30 papers since the beginning of the conference, against nearly 120 for CHI 2006. This is possible because UIST has maintained a focus on core HCI topics, whereas CHI caters to a much wider range of interests and accepts papers on a broader range of topics. Like for InfoVis and AVI's focus on visualization (see below), these specialized topics may have a narrow audience and thus lower UIST's average impact. Clearly, UIST is more selective, but this may mean that its impact suffers.

It would be interesting to differentiate impact figures by sub-area, for instance by keyword. However, CHI's broader focus is also probably a reason for its larger total audience and impact.

Citation Patterns Figure 8.18a also implies a correlation between the core topics of CHI and UIST. Although UIST is much smaller, almost every CHI conference has referenced at least one UIST article and vice versa, suggesting that the basic interests of their communities are strongly connected. Similarly, the two visualization-oriented conferences InfoVis and AVI cite one another. Interestingly, both conferences cite CHI and UIST articles far more than the reverse. Presumably, this is a case of a specialized field needing to cite basic principles of the parent field (however note the above results about much of the impact depending on a few articles). It is also possible that CHI and UIST are less open to external articles.

Finally, an unexpected finding is an unusually high number of intra-citations (citations between articles within the same annual conference) for UIST conferences. The CHI 91 conference also shows a high number of intra-citations (33 articles referencing articles of the same conference year). Because intra-citations require authors to know of other submissions in advance, they indicate an intertwined community with many co-authorship relationships between groups, and/or prolific research groups that have multiple papers accepted in a year. By contrast, intra-citations are rare in InfoVis, which suggests that research groups there are less intertwined or individually less prolific than for CHI or UIST conferences. Alternate explanations might include reviewing styles and prejudices: for instance blind reviewing such as CHI uses would make it more difficult to "ration" multiple acceptances to the same research group.

Article Citation Network In an article citation network, articles are the vertices and references between articles are (directed) edges. We do not present any visualizations of article-citation structure as they are very large (up to 23000 nodes). Even if heavily filtered, they would be useless without readable node labels, which is difficult because article titles are typically longer than names. Therefore, the next few sections of this section present the results of interactive exploration, illustrated by selected highlights.

Structure An overview of the article citation network is useful to identify how articles in a conference reference each other, as well as articles outside. Unfortunately, it is impacted by missing data, in particular for article references outside our core datasets that are much less effectively resolved.

A first observation is that for AVI and especially InfoVis, the graph of citations within the conference articles is much sparser than for CHI or UIST. CHI and UIST have a longer history, so one interpretation could simply be that articles in these conferences have had more time to impact the field than articles at InfoVis and AVI. Another reason could be that CHI has far more articles in total (UIST does not, however), or that UIST and CHI generate more key articles.

Interesting observations concerning the citation matrix presented in Figure 8.18a is that CHI and UIST cite each other, AVI cites articles from all three conferences, and InfoVis is more isolated, primarily citing articles in its own conference. Of the few links that point outside the InfoVis area (towards the top of the diagrams) in the UIST (right side) or CHI area (left middle and bottom part), most are to a very limited subset of articles, as previously discussed. This observation confirmed that a conference impact may rely on a small set of articles (Figure 8.18b).

Citation Patterns The general observation is that most-cited articles reference each other. Within those, "Generalized Fisheye Views" [Fur86] is the only article cited by others without referencing any of the most cited — trivially explainable as it was written before them. This article is seminal in the history of both HCI and InfoVis, as its citations reveal. Studying the top twenty key articles, only two articles cite others without being cited by them: "The Table Lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information" [RC94] and "Pad++: A Zoomable Graphical Interface System" [BH94]. The explanation is also chronology: published in early 90's, they are the most recent of our most-cited article set.

Finally, we noticed that two of these articles cite one another: the "The Information Visualizer: an Information Workspace" [CRM91] and "The Perspective Wall" [MRC91a]. Again, the explanation is trivial: both were written by the same authors, the trio of Card-Mackinlay-Robertson all then of PARC, and published at the same conference, CHI '91.

Author Citation Network In the author citation network, the authors are the vertices and their references to other authors are the edges. This network is derived from the article citation network by aggregating articles that connect citing to referenced authors. This network shows how the important contributors in the field influence each other.

Figure 8.19 presents heavily-filtered node-link diagrams of the author citation networks for CHI, UIST, InfoVis and AVI. Filtering all but the most-cited authors allowed us to see how they cite one another. Node size and darkness redundantly encode each researcher's total number of citations, while the width and darkness of the links do the same for the number of citations from one researcher to another.

Citation Patterns A first observation is that the trio of Card-Mackinlay-Robertson appear prominently in both the CHI and InfoVis networks, referencing one another heavily in both article sets. An obvious interpretation was that they were referencing the breakthrough articles they co-authored in both HCI and information visualization.

In the CHI author citation network, we saw that CHI's single most-cited author, William Buxton, is heavily cited by six of the other leading researchers. All cite him much more than the reverse, with the striking exception of Abigail Sellen, whom he cites far more. He also cites Hiroshi Ishii and Scott Mackenzie relatively frequently.

Examining the InfoVis author citation network, we observed that Ben Shneiderman has a pattern similar to William Buxton. Curved links underlined the mutual citation of Ben Shneiderman and Christopher Ahlberg. These two collaborated (with Christopher Williamson) on "Dynamic Queries for Information Visualization" [AWS92], one of Ben Shneiderman's most-referenced articles.

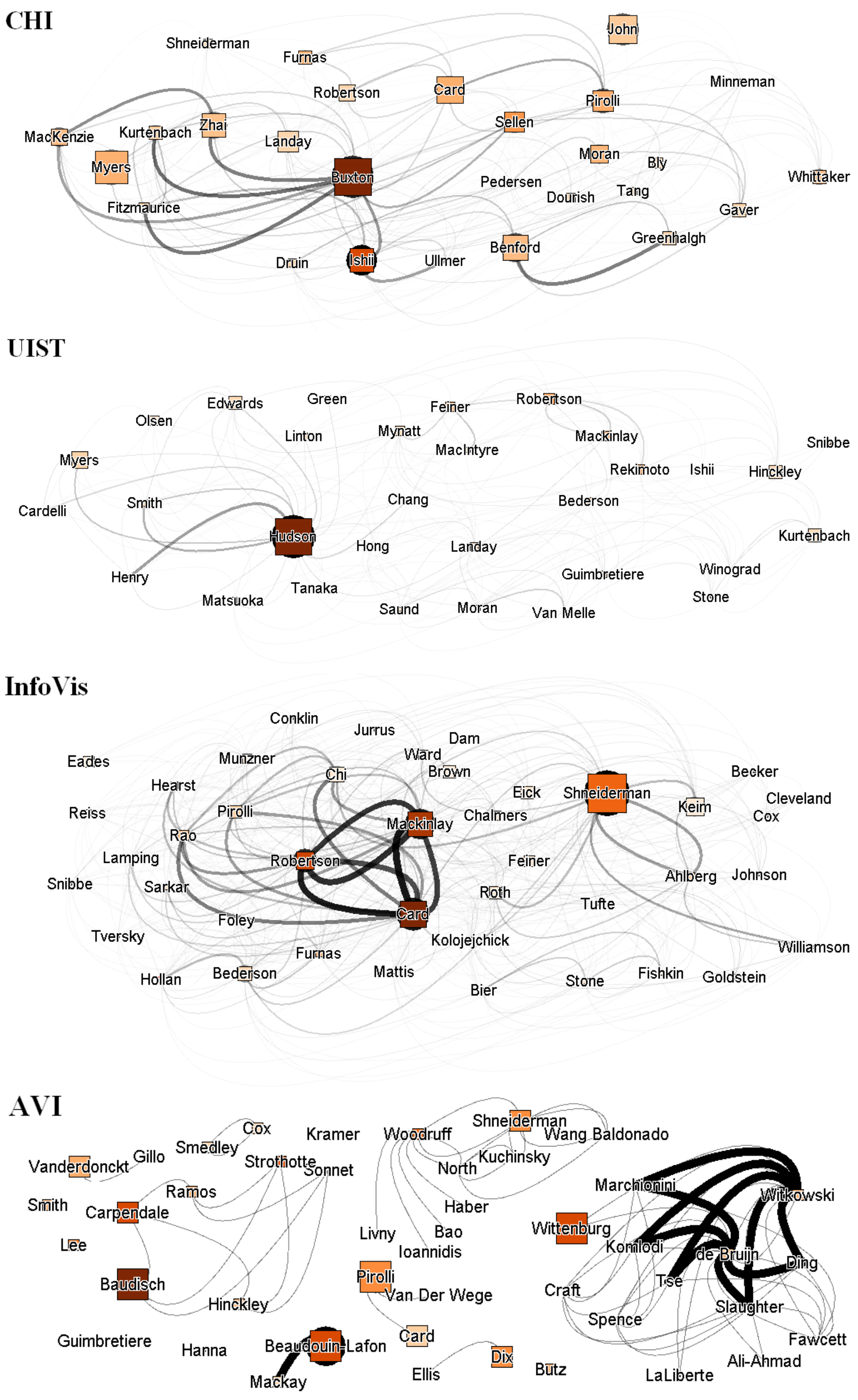
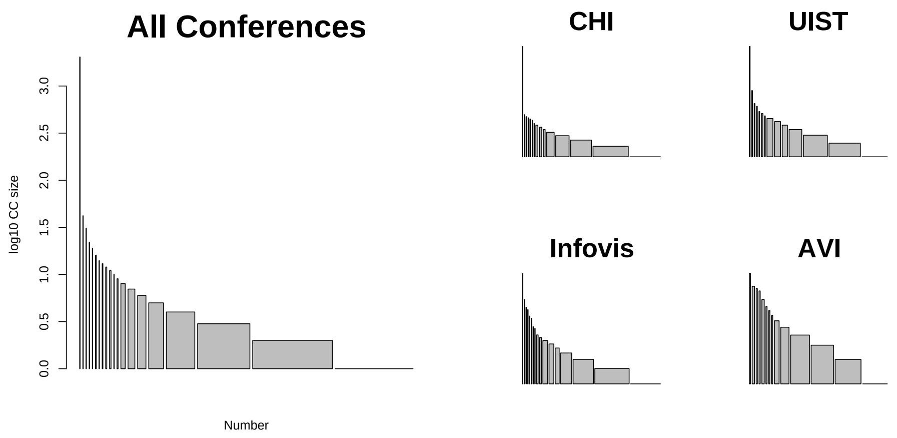


Figure 8.19: Author citation networks for CHI, UIST, InfoVis and AVI. Networks are filtered by number of citations, showing only how most-cited researchers cite one other. Size and colors indicate the number of citations. Nodes are filtered by number of citations.

Finally, the much smaller author citation networks of UIST and AVI did not show strong patterns of citations. For UIST, we could only observe that Scott Hudson is referenced most often by the most-cited authors.

Considering self-citation, we observed a global pattern that the most-cited researchers heavily reference their own work. This is not true for AVI, perhaps because many participants only began contributing after 2000; so the pattern has not had time to emerge (especially on a biennial schedule). The self citation trend is particularly strong for the Card-Mackinlay-Robertson trio at CHI and InfoVis, for Hiroshi Ishii and William Buxton at CHI, as well as for Ben Shneiderman at InfoVis and Scott Hudson at UIST. Our interpretation is that these authors of multiple breakthrough articles in the field naturally cite them.



(a) Co-authorship connected components: size(log10) vs. number

	All 4	CHI	UIST	InfoVis	AVI
Number of authors	5 109	3 422	956	325	375
Number of articles	3 209	1 943	542	152	159
Articles per author	1.8	1.6	1.6	1.5	1.2
Authors per article	2.8	2.8	2.8	2.7	2.8
Average number of collaborators	4	4	3.8	3.2	2.9
Giant component	49%	50%	49%	13%	9%
Number of components	929	627	169	291	99

(b) Connected component count and size per conference

Measure	Biomed	HEP	CS	HCI
Number of authors	152 0251	56 627	11 994	23 624
Number of articles	216 3923	66 652	13 169	22 887
Articles per author	6.4	11.6	2.6	2.2
Authors per article	3.8	9.0	2.2	2.3
Average number of collaborators	18.1	173	3.6	3.7
Giant component	92.6%	88.7%	57.2%	51.3%
Mean distance	4.6	4.0	9.7	6.8
Largest distance	24	19	31	27

(c) Statistics for other fields

Figure 8.20: Macro structure of co-authorship networks.

Co-Authorship Networks

We analyzed co-authorship data in two stages. First, we surveyed the macro-structure of each conference community, describing its connected-components structure and global statistics (with some comparison to other fields.) In the second stage, we performed a detailed analysis of communities we had identified within this data, first for the whole HCI community (aggregating the data of all four conferences), and then for each conference community independently.

Macro Structure A connected component is a maximal connected sub-graph: a vertex in one connected component has no path to any vertex from another connected component. In this context, this information told us whether the research field is primarily composed of distinct communities that do not publish together or a single one connected by various degrees of co-authorship. Figure 8.20a is a bar chart of these connected components. Each bar represents *all* the components of a given size. Its height is the log of the component size, and the width represents the number of components of that size. Note that even at a log scale, CHI and UIST as well as the aggregated data of all the conferences show a single "giant component", a very tall and thin (because it has only one element) bar representing a component containing approximately half the authors, all of whom interact. This is shown more precisely in Table 8.20b. By contrast, the largest component in the InfoVis and AVI graphs is far smaller, representing only 13% and 9%, respectively, of their authors. The most likely explanation seemed to be that the citation patterns of these newer conferences had not developed as fully (as well as having time for students to graduate and researchers to move between institutions); so the joint publications that would link different community components have not had time to appear. Alternate explanations included commercial constraints in the visualization field (such as some research being done with very expensive hardware or proprietary software) that restrained collaboration between communities.

By way of comparison, Table 8.20c presents data on several fields extracted from [New01b] (Medicine, biology and computer science) and [HFB$^+$04] (the HCI field). The HCI data in this table comes from a different source, HCIbib.org, which does not contain any information on article references. We computed similar measures for our own data, as (Table 8.20b) shows, to provide some comparison with other fields. However, these comparisons should be made with caution, for two reasons:

1. The percentage of incompleteness and errors in these datasets is unknown; and
2. Because the measures are computed on variables which often follow power-law distributions, averages might not be a good comparison.

Communities of HCI Our first analysis was performed on a network composed of the data of all four conferences. Here, the largest component is a subgraph containing 2,522 authors. Standard node-link diagrams of such a large graph would be unreadable without heavy filtering. Instead, we used the adjacency matrix representation provided by our tool MatrixExplorer [HF06a]. The analog of graph layout for this representation is *matrix reordering*: finding a 1-D ordering of the nodes that groups closely-related ones; so the patterns become visible. Traveling Salesman Problem (TSP) approximation algorithms give good results for reordering many kinds of data. By placing authors with similar co-authorship patterns nearby, ordering reveals community structures effectively (even preattentively) as blocks of adjacent edges.

Unfortunately, large matrix visualizations are even harder to fit on printed pages than node-link ones. Therefore, we present several *NodeTrix* [HFM07] visualizations of selected details of these graphs. This representation represents the large-scale network structure with a standard node-link diagram but converts dense regions that would be unreadable in node-link as multiple small matrix representations. It includes flexible tools for dragging and dropping groups of nodes from one to the other. The NodeTrix visualization is particularly effective for small-world networks. For co-authorship networks, strongly-connected communities appear as preattentively-visible block patterns on the matrix display. We created NodeTrix representations by interactively dragging visual clusters appearing in a matrix representation into a NodeTrix visualization window. Very large clusters were edited into separate communities to show their detailed structure. This visualization allowed us to represent the main communities together with the details of their connections. However, because of the interactive editing and labeling, the results are subject to interpretation.

Figure 8.21 presents the visualization created during our analysis process. Reordering the matrix of the largest component of the co-authorship network reveals several visual clusters that we have outlined in the upper right corner. A *visual cluster* in the matrix is a sub-matrix denser than the others. It means that the researchers of this sub-matrix collaborate with each other, i.e. form a *community*. By zooming in to examine these clusters closely and applying our own knowledge of the domain, we discovered that these clusters group researchers primarily by institution or by research topic.

Dragging these visual clusters into a NodeTrix window and dividing them into smaller communities centered on a main researcher resulted in the visualization at the top of the Figure 8.21. A zoomed-in view in the lower left corner shows one of these communities in detail.

In the data combining all four conferences, we located four main communities:

- CMU-Toronto: a community centered on William Buxton that is composed primarily of researchers from Carnegie Mellon University and the University of Toronto;
- CSCW-UMD: a community of CSCW researchers that includes a large group of researchers from Nottingham University: Steve Benford and Chris Greenhalgh, and also researchers from other institutions such as Ben Bederson from the University or Maryland and Michel Beaudouin-Lafon from the University of Paris-Sud;
- PARC: a community centered on Stuart Card and Jock Mackinlay, containing Ben Shneiderman from University of Maryland as well as Elizabeth Mynatt from Georgia Tech;
- Microsoft Research: a community mainly centered on George Robertson, Ken Hinckley and Patrick Baudisch.

We broke these four large communities in smaller ones and present the NodeTrix visualization in Figure 8.21. Each small matrix is a community centered around a researcher and/or an institution. Two distinct patterns recur in these small matrices: crosses and blocks. Dark crosses indicate a single researcher who collaborates with many others, while dark blocks indicate groups of researchers collaborating with each other (a perfectly-collaborative block, meaning that each member interacts with every other member, is called a *clique*, which appears as a fully filled-in dark block, since there is an edge in each position between them). For example, the detailed matrix view in the lower right corner shows Ken Hinckley is linked to many other researchers with a cross-pattern, while also being part of a smaller clique of Agrawala - Ramos - Hinckley -

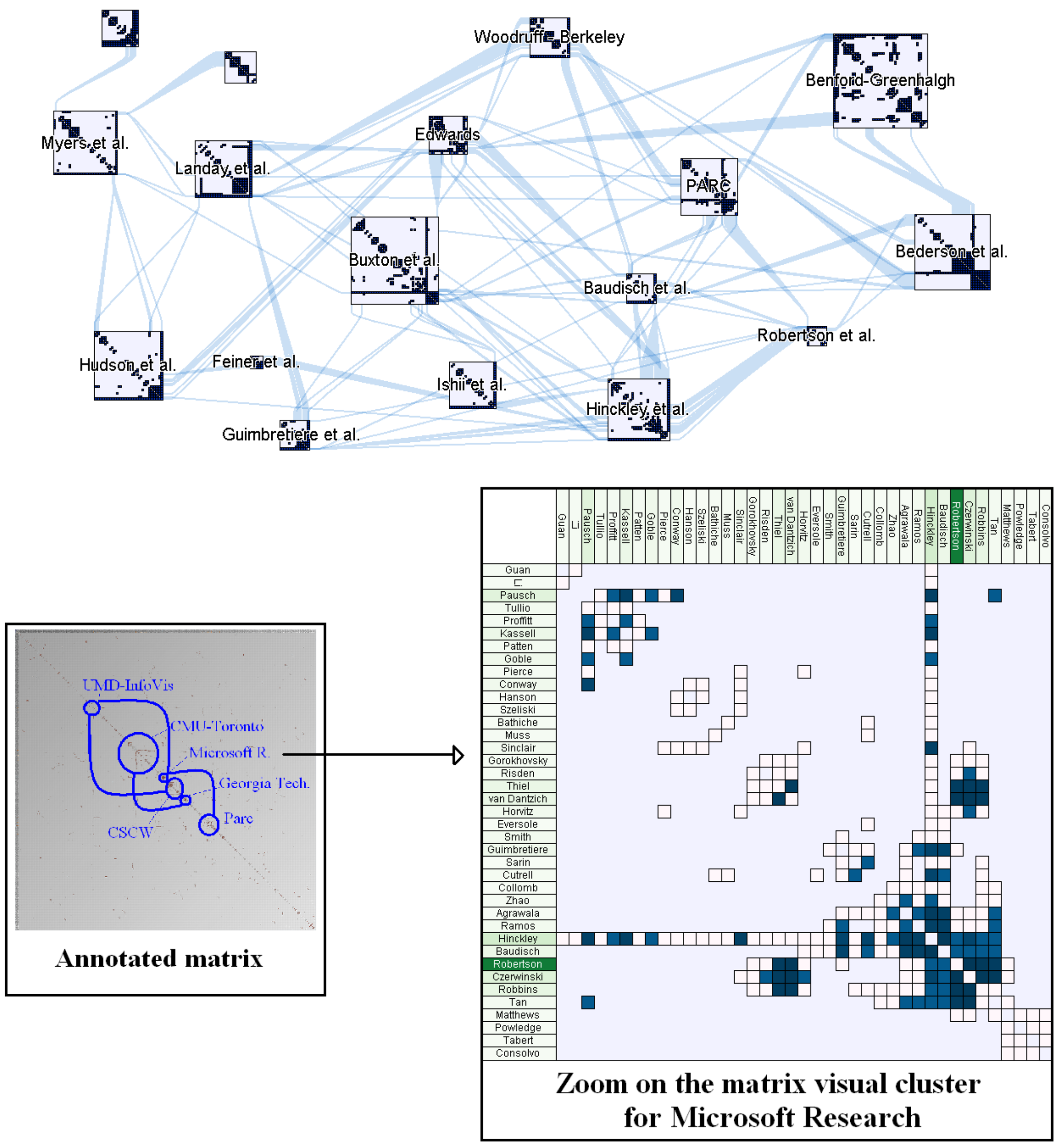


Figure 8.21: Largest component of the co-authorship for all conferences. We annotated the whole matrix with the different communities' labels (lower left corner), a zoom of the Microsoft Research cluster is provided on the lower right corner. Shades in the headers row and column indicate the number of citations. We dragged the visual clusters into a NodeTrix visualization, edit them and present the visualization in the upper part of the figure.

Baudisch - Robertson - Czerwinsky - Robbins - Tan. In NodeTrix, the links between the matrices show how communities are linked at a high level. The width of the link lines shows the number of researchers involved in the collaboration: for example, George Robertson collaborated with a third of the researchers in the PARC community and around half of the researchers in the Hinckley et al. community.

Interacting with the visualization revealed that Ben Shneiderman bridges the PARC and CSCW-UMD communities. He effectively collaborated with Stuart Card of PARC and also with researchers from his home institution, the University of Maryland, such as Ben Bederson and Catherine Plaisant. George Robertson is a bridge between Microsoft Research (his new institution) and PARC (his former one). The co-authorship collaboration patterns of other central researchers such as William Buxton have a more prominent cross pattern, showing that they are the center of collaborations with a large number of researchers. In the node-link regions between matrices, a cross pattern becomes a dense web of links converging on the central researcher.

The following sections describe these different communities in more detail. We present four zoomed-in visualizations of the largest component of the matrix. These show the clusters CMU-Toronto in Figure 8.22, CSCW-UMD in Figure 8.23 PARC in Figure 8.24 and a portion of the Microsoft Research community in Figure 8.21.

CMU-Toronto: The central researchers of this cluster are William Buxton, Thomas Moran, Brad Myers and Iroshi Ishii. Figure 8.22 is a matrix visualization showing the major part of this community centered on William Buxton. Shades inside the matrix mark the strength of the collaborations. Shades in rows and columns indicate the number of citations of these researchers. It is clear that William Buxton has had many collaborations with the most-cited researchers. These researchers have collaborated with each other in small groups (noticeable as blocks in the matrix). For example, William Buxton, Ravin Balakrishnan, Tovi Grossman, Thomas Baudel, George Fitzmaurice and Gordon Kurtenbach form a near-perfect clique. Thomas Moran and Brad Myers appear here as collaborators of William Buxton, but the remainder of the communities formed around these two individuals are located off-axis, in another part of the matrix that is not shown. Finally, the community centered on Iroshi Ishii is visible at the upper left corner of the matrix. His pattern is similar to William Buxton, a large “cross” of coauthors who did not collaborate strongly with one another.

CSCW and UMD Figure 8.23 shows two large cliques connected through Ben Bederson as well as a large community centered on Chris Greenhalgh and Steven Benford (sparse block occupying the main part of the matrix). The community at the upper left mainly contains researchers from the University of Maryland linked to Steven Benford. The second large block connects members of the European Union-sponsored InterLiving project. It is interesting to note that the strongest collaboration of this community is Benford-Greenhalgh (11 co-authored articles) and that they both have very similar connection patterns, i.e. they have collaborated with the same researchers. The community centered on them can be further broken down into several smaller groups (blocks) of researchers who collaborating actively with each other.

Microsoft Research An enlarged NodeTrix view of this community appears in the lower left corner of Figure 8.21. The NodeTrix view of its detailed structure includes three main subcommunities labeled Baudisch et al., Robertson et al. and Hinckley et al.). A general observation

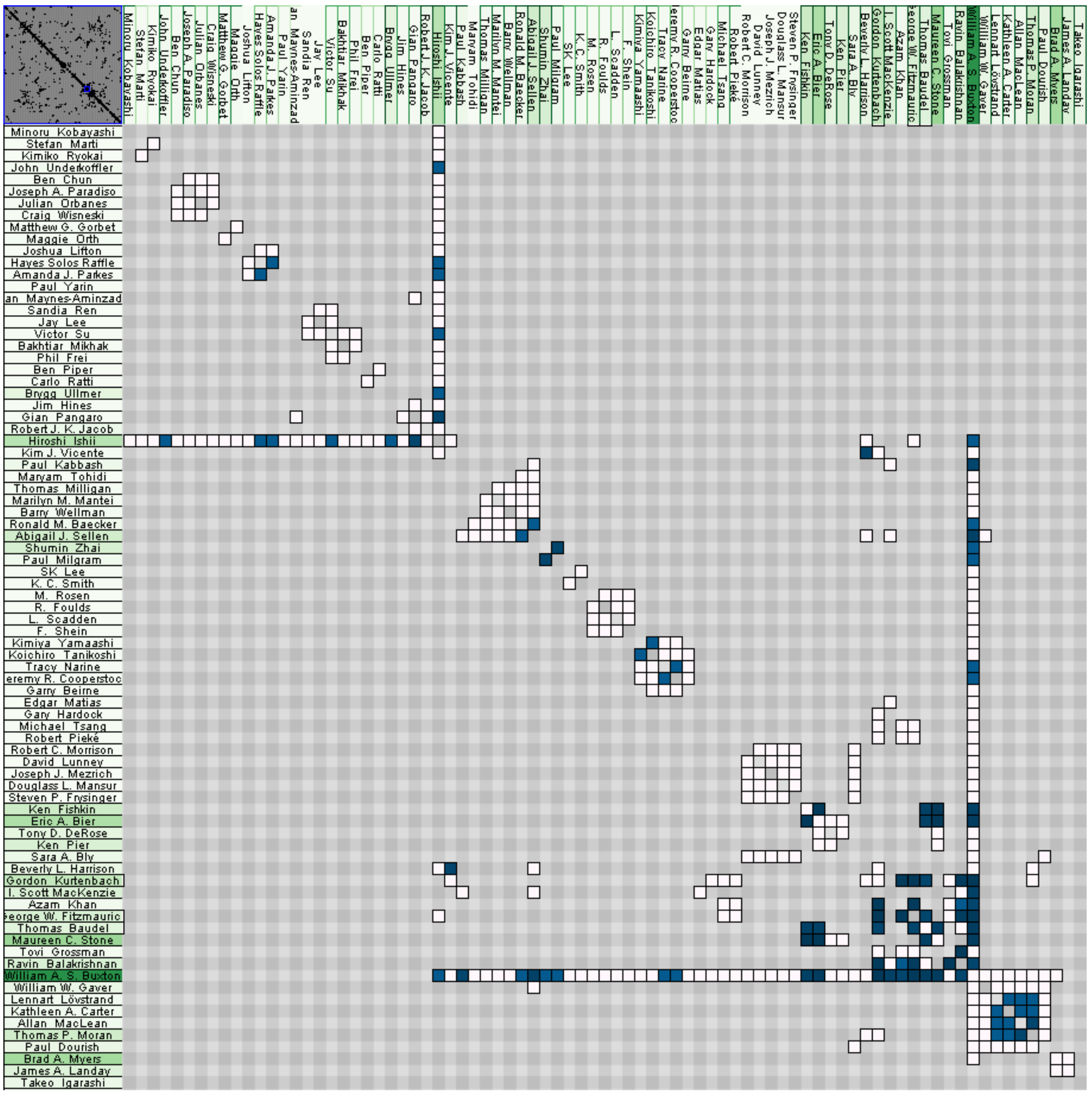


Figure 8.22: Zoom on the main cluster: CMU-Toronto based upon the matrix of co-authorship for all conferences. In rows, areas are the number of articles a researcher published, in column the number of citations. Values in the matrix indicate number of articles published together.

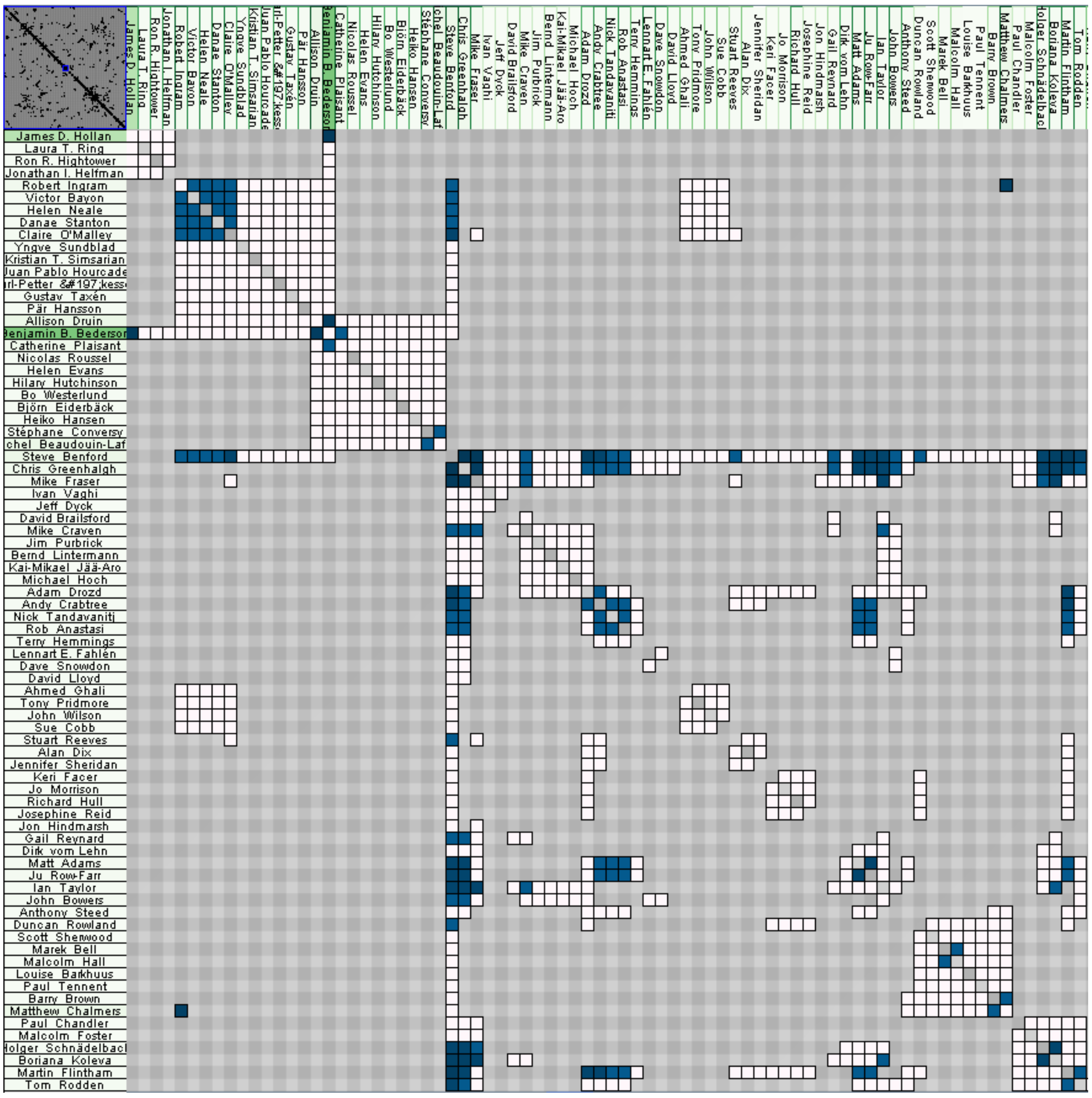


Figure 8.23: Zoom on a community CSCW - UMD based upon the matrix of co-authorship for all conferences. In rows, areas are the number of articles a researcher published, in column the number of citations. Values in the matrix indicate number of articles published together.

for this cluster is the strong collaborations within Microsoft Research, especially between George Robertson and Mary Czerwinski who co-authored 16 articles. This strength is visible in the matrix representation as gray-scale indicates the strength of the collaboration.

PARC: The NodeTrix representation of this community has wide links going to George Robertson, and also to the Berkeley community, Alison Woodruff in particular. Figure 8.24 is a zoomed-in view of the matrix showing the Alison Woodruff and Keith Edwards community. It shows small sub-communities, such as the one centered on Peter Pirolli connected to Stuart Card and Jock Mackinlay, the one centered on Alexander Aiken connected to Alison Woodruff and the one centered on Elizabeth Mynatt, connected to Keith Edwards. Ben Shneiderman also appears in this community, primarily because of a single reference, the much-cited handbook "Readings in Information Visualization" he coauthored with Stuart Card and Jock Mackinlay.

UMD-InfoVis: We did not break out this community as a separate chart, but we annotated it off-axis in the original matrix. Several well-known InfoVis researchers appear in this community: Tamara Munzner(British Columbia), Martin Wattenberg(IBM) and Ben Shneiderman's collaborators Christopher Ahlberg and Christopher Williamson. This is easily explainable as an artifact of our reordering algorithm, which places the largest groups in the center of the matrix as it computes a 1D ordering. Because of Ben Shneiderman's surprising appearance in the PARC cluster in the primary ordering, the remainder of this community of which he is the center was pushed to the side of the matrix, still intersecting with him but off-axis. Note that Ben's cross pattern therefore appears as separate vertical and horizontal pieces in the symmetrical upper and lower matrices.

Communities of Each Conference This section presents NodeTrix visualizations for the CHI, UIST, InfoVis and AVI conferences separately, attempting to show both communities and important actors.

As we zoom into the NodeTrix visualization, the rows and columns of each matrix become readable, and thick consolidated links resolve into specific links between individual researchers. The figures do not provide detailed view of the whole networks here because of the lack of space, but they show a few selective enlarged portions. However, it must be kept in mind that we performed editing, analysis and labeling using interactions on the representation (drag and dropping elements to and from matrices) and zooming to produce these representations.

CHI: The organization of the co-authorship network containing only CHI data is shown as a NodeTrix in Figure 8.25a. The matrix visualization of the whole largest component revealed a main visual cluster centered around William Buxton and Thomas Moran. We present a zoomed-in view of the matrix visualization showing this cluster in Figure 8.25b.

By interactively filtering and ordering the matrix visualization of the largest component, we were able to distinguish five different communities (Figure 8.25b):

1. The largest community centered on William Buxton and Thomas Moran, including Abigail Sellen, William Gaver, Paul Dourish and Shumin Zhai. We also notice that a smaller community formed around Hiroshi Ishii;
2. The Brad Myers and Stuart Card community;
3. The community centered on Steve Benford and Chris Greenhalgh

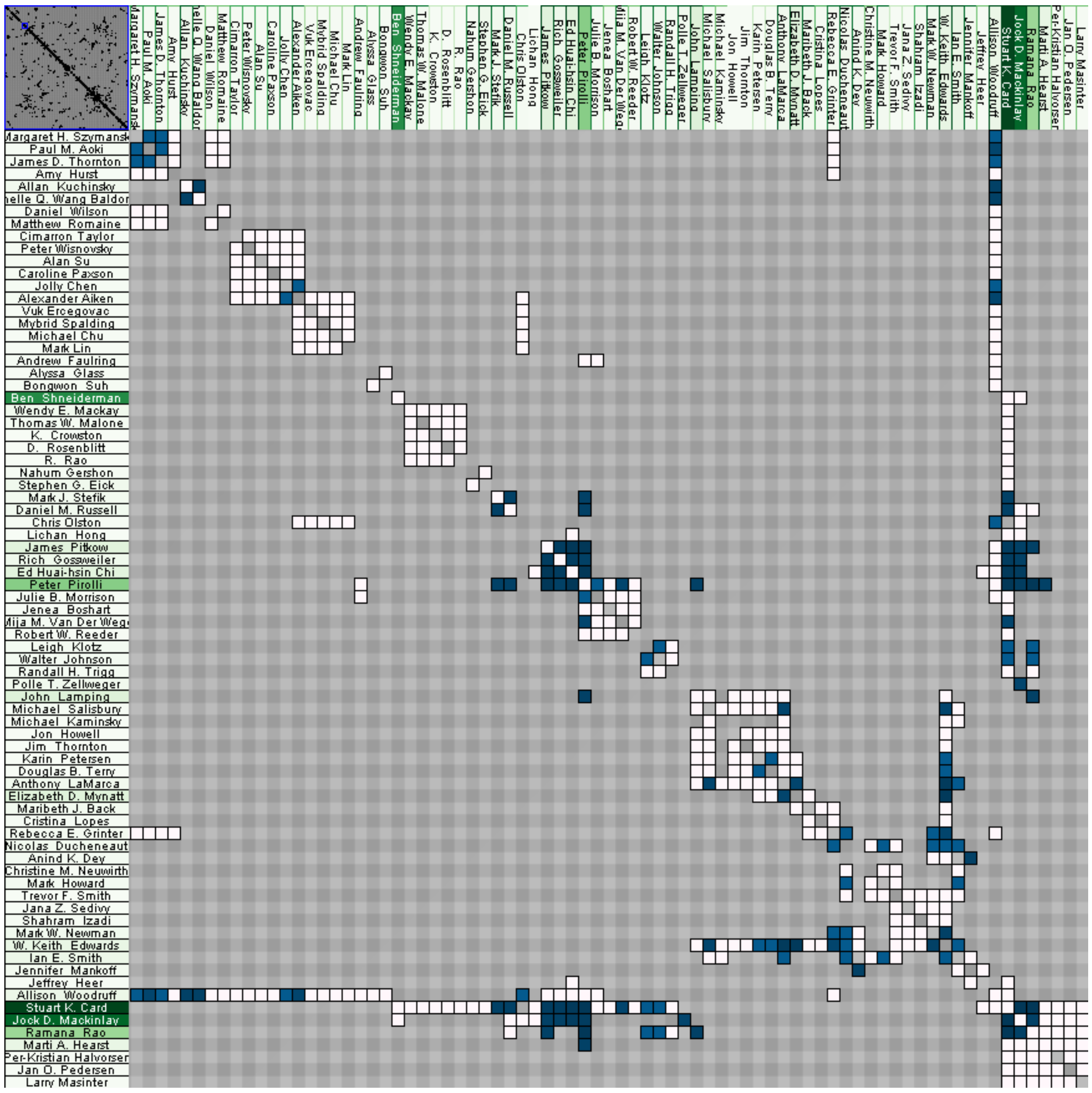


Figure 8.24: Zoom on a PARC community based upon the matrix of co-authorship for all conferences. In rows, areas are the number of articles a researcher published, in column the number of citations. Values in the matrix indicate number of articles published together.

4. The community centered on Ravin Balakrishnan and Ken Hinckley; and
5. The CMU community centered on Scott Hudson, Sara Kiesler and Robert Kraut.

Other zoomed views in the co-author matrix show interesting communities such as a clique (fully connected community) formed by researchers of UMD and the French INRIA research institute, or the Microsoft Research community where collaboration between researchers is strong (9 articles co-authored by Mary Czwerwinski and George Robertson).

It is interesting to note that the largest community in the NodeTrix visualization above appears to be the one centered on Steven Benford and Chris Greenhalgh, but this is only because we split up William Buxton's community into several smaller ones. This breakdown was natural, because Buxton's matrix has many links to other matrices. This indicates that William Buxton's many collaborators are actually active in many small communities, but all these communities are pulled into Buxton's community by their central members who collaborate with him, just as Ben Shneiderman's UMD community was dragged beside PARC. These strong effects of a few individuals on the ordering may not be optimal for showing each group's individual structure, but they do outline the largest communities clearly. This is evident in the zoomed-in matrix view in Figure 8.25b, which shows almost all the collaborators of William Buxton in a single clearly-delineated view.

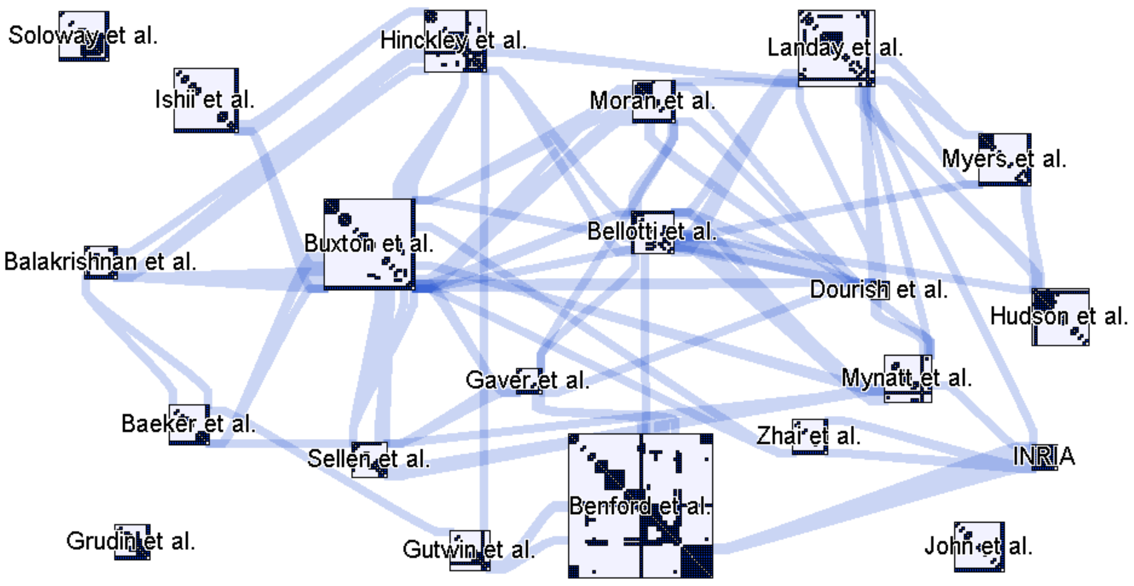
UIST: Figure 8.26 shows the largest component of the co-authorship network of UIST as a NodeTrix visualization. Two sections have been enlarged to show several communities in details.

First, central actors are identifiable because their large number of connections and often make them bridges between communities. We can identify Ken Hinckley, Ravin Balakrishnan, Elizabeth Mynatt, Scott Hudson and Keith Edwards as central actors in UIST. It is interesting to notice that Elizabeth Mynatt is a bridge between the community centered on Blair MacIntyre and the rest of the network. Similarly, Igarashi acts as a bridge between researchers from University of Tokyo and the community centered on Jun Rekimoto.

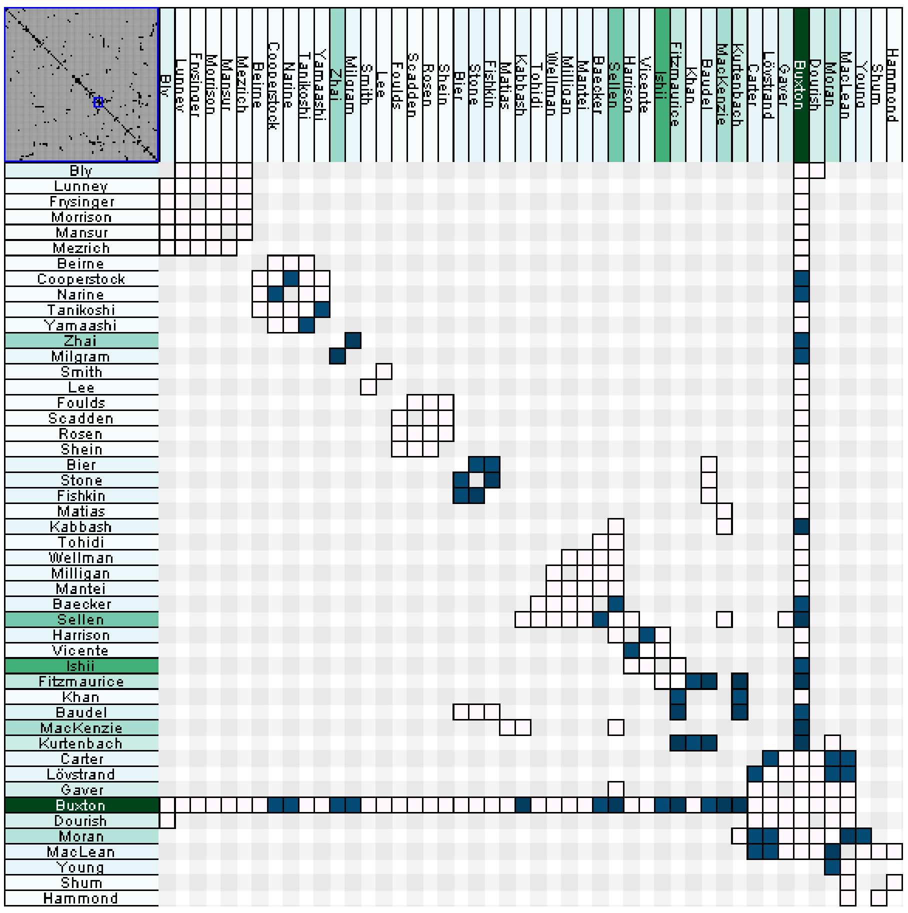
As before, the cross and block patterns indicate the extremes of collaboration via a single individual and widespread collaboration between many members. In a node-link diagram, the cross becomes a star pattern: the others collaborate often with the center actor but rarely with one other. Usually, this can be interpreted as a senior researcher advising junior ones. In Figure 8.26, we can identify these types of communities centered on Ravin Balakrishnan, Gordon Kurtenbach, Scott Hudson, and Keith Edwards and Jun Rekimoto.

The zoomed-in matrix in the lower left corner of this figure shows the largest community centered on Scott Hudson and Keith Edwards. In this community, we can notice that collaborators of Keith Edwards tend to collaborate with each other, as shown by the three blocks in the upper left corner of the matrix. Other examples of this pattern can be found in two matrices labeled PARC as well as in the community centered on Ken Hinckley: Microsoft Research, and the community labeled Berkeley. We characterize this as a mixed pattern, with a dark cross centered on one researcher, but included in a fairly dense block of mutual collaboration. As we previously saw for Ken Hinckley, the block refers to the strong connections within Microsoft Research: the cross is composed of researchers who only collaborate with Hinckley.

The zoom on the lower right corner clearly shows the two patterns. Ravin Balakrishnan has a high number of collaborators who did not collaborate with each other, whereas Forlines in the upper matrix is a bridge between two cliques of researchers who collaborate extensively with each other.



(a) Overview of the CHI co-authorship network



(b) The largest CHI community centered on William Buxton and Thomas Moran

Figure 8.25: CHI co-authorship network. Values in the matrix indicate number of articles published together.

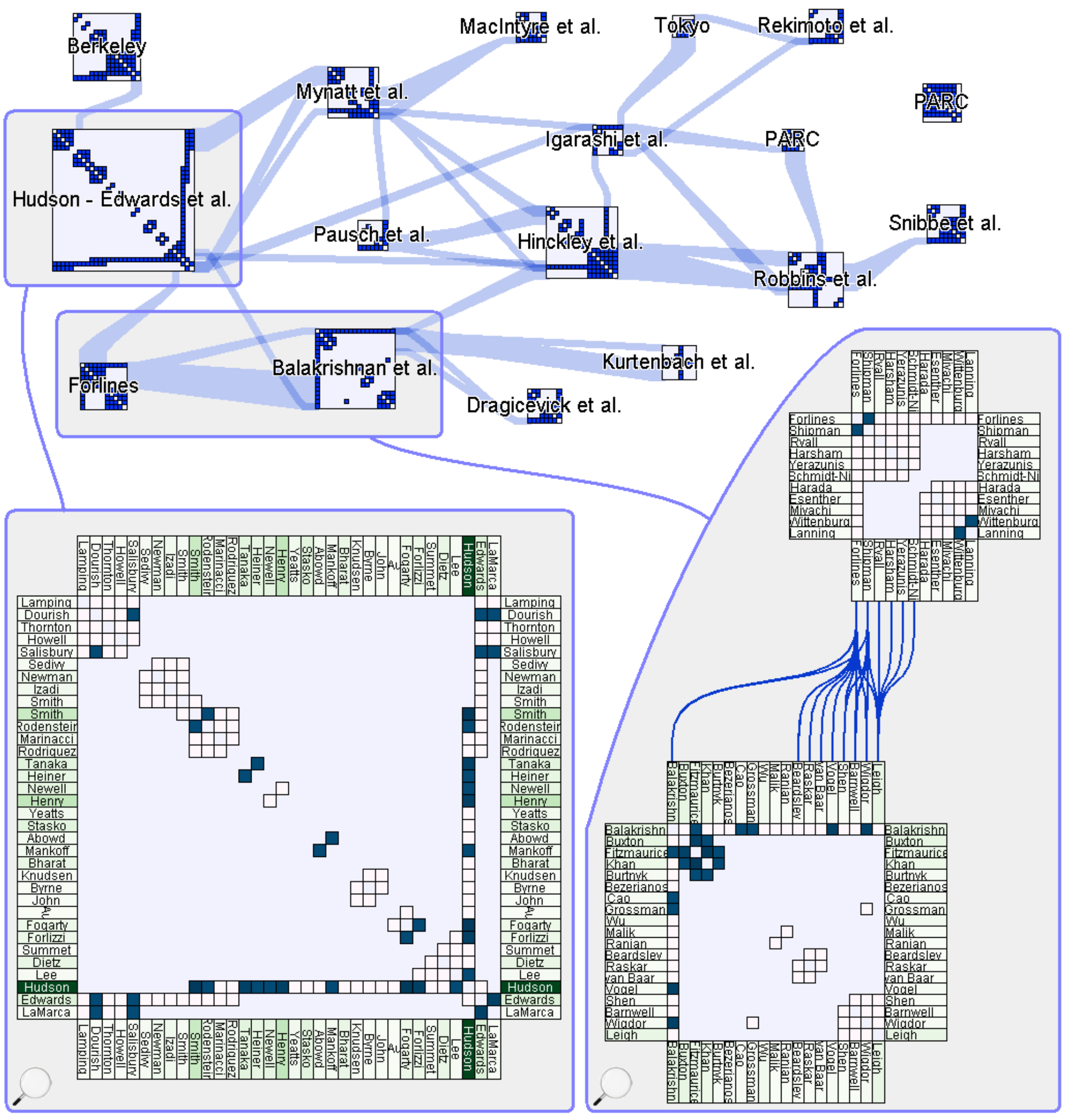


Figure 8.26: UIST co-authorship network.

InfoVis: Figure 8.27 shows the largest component of the co-authorship network of the InfoVis conference. The lower right corner shows the overview of whole InfoVis matrix, labeling the main actors of this network: PARC and Ben Shneiderman. The largest cross identifiable is Ben, the most central actor in the InfoVis community.

The NodeTrix representation in the lower left corner shows how Ben Shneiderman acts as a bridge to the other UMD researchers grouped in a community centered on Ben Bederson.

Finally, the upper part of the figure is a zoomed-in NodeTrix view showing how the PARC community collaborates with other communities. It is interesting to note that Berkeley and Microsoft Research strongly collaborate with each other. Similarly Stuart Card, Jock Mackinlay and Ed Chi collaborators are strongly connected.

AVI: Because the co-authorship network of AVI is quite small, we were able to fit the full matrix representation in Figure 8.28. This matrix is composed of many connected component, identifiable as disconnected blocks placed on the matrix diagonal. We present the details of several of these blocks as NodeTrix visualizations above and below the diagonal. The NodeTrix view of the largest component displayed in the bottom left of the picture shows that Patrick Baudisch from Microsoft Research is the central researcher of this component. The zoomed-in view on the upper right side of the matrix shows the connected component containing the most-cited researcher within AVI: Michel Beaudouin-Lafon from the University of Paris-Sud.

The collaboration within AVI must be interpreted with caution, because the conference has only become prominent since 2000 and is held only biannually (and also because the 2002 data is missing). However, these features make this conference data an interesting contrast to the others: a co-authorship network at a very different state of maturity. Relative to CHI or UIST, its network is very disconnected and with very low collaboration strength; since most research groups have only submitted a limited number of articles here. It is interesting to note that this network still presents a small-world effect, however.

Author-Author Collaboration Finally, in Figure 8.29, we present node-link diagrams of the co-authorship networks filtered by number of citations. The node darkness represents the researchers' number of citations, and the node size their total number of articles published. The darkness and width of the links redundantly encode the strength of the collaboration, i.e. the number of co-authored articles.

These four node-link diagrams reveal how most cited authors collaborate with each other. They highlight once again the three researchers Card-Mackinlay-Robertson who collaborate in both the CHI and InfoVis communities.

The global trend is that the most cited-researchers are both the most prolific and also have the largest number of collaborators. For all the conferences, most co-authors collaborate with each other. Within CHI and UIST, we observe that these collaborations are strong and shaped as a star pattern centered on the most cited authors: William Buxton and Scott Hudson, who have a large number of co-authors, but these co-authors do not collaborate strongly together.

Within InfoVis and AVI, the most-cited authors also have a high number of collaborators. The pattern of collaboration of InfoVis is different from a single star shape: the collaboration seems more distributed, which makes sense given the relatively fragmented connected-component structure seen in Figure 8.20a.

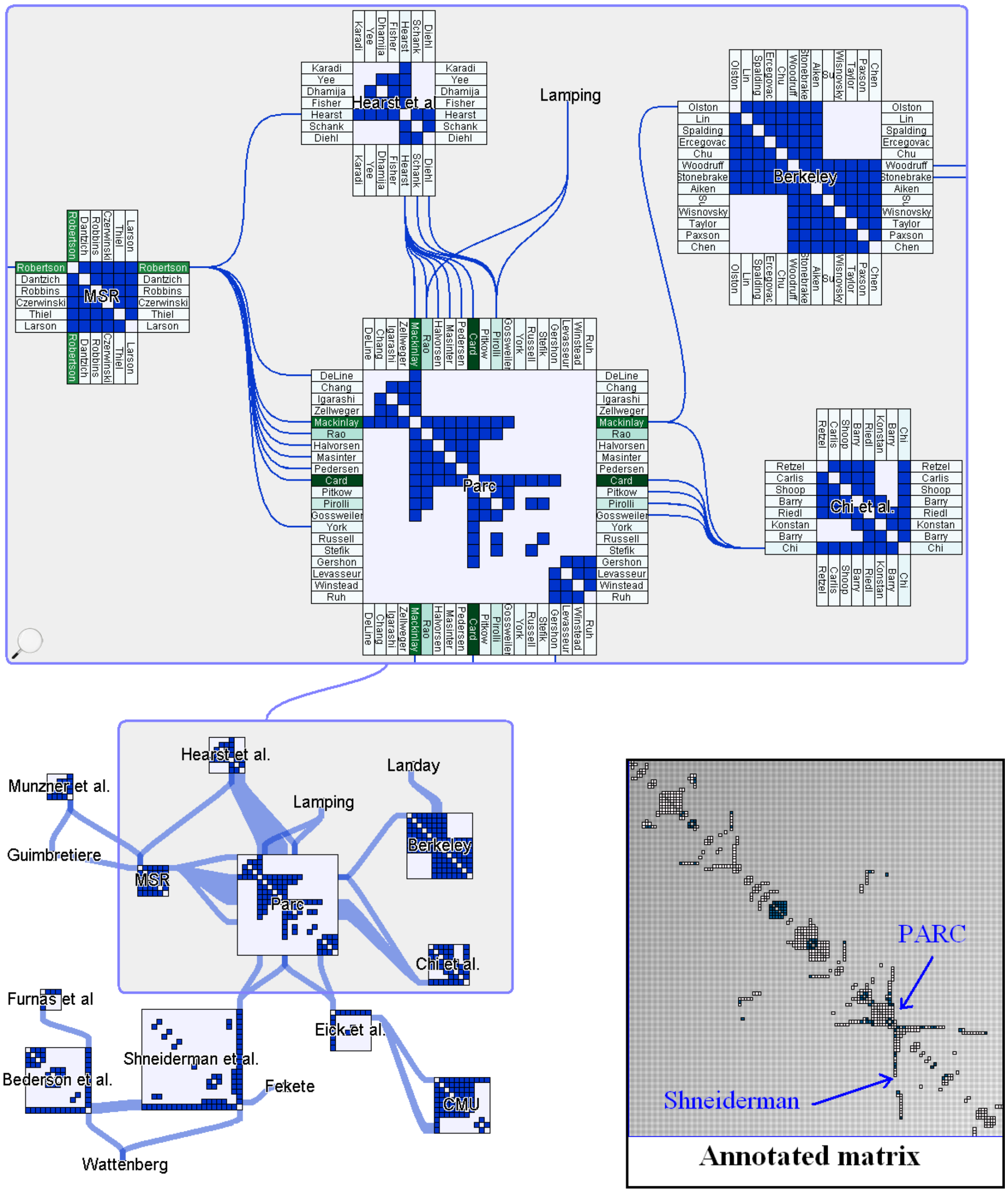


Figure 8.27: The largest component of the co-authorship network of InfoVis. Communities are displayed as matrices.

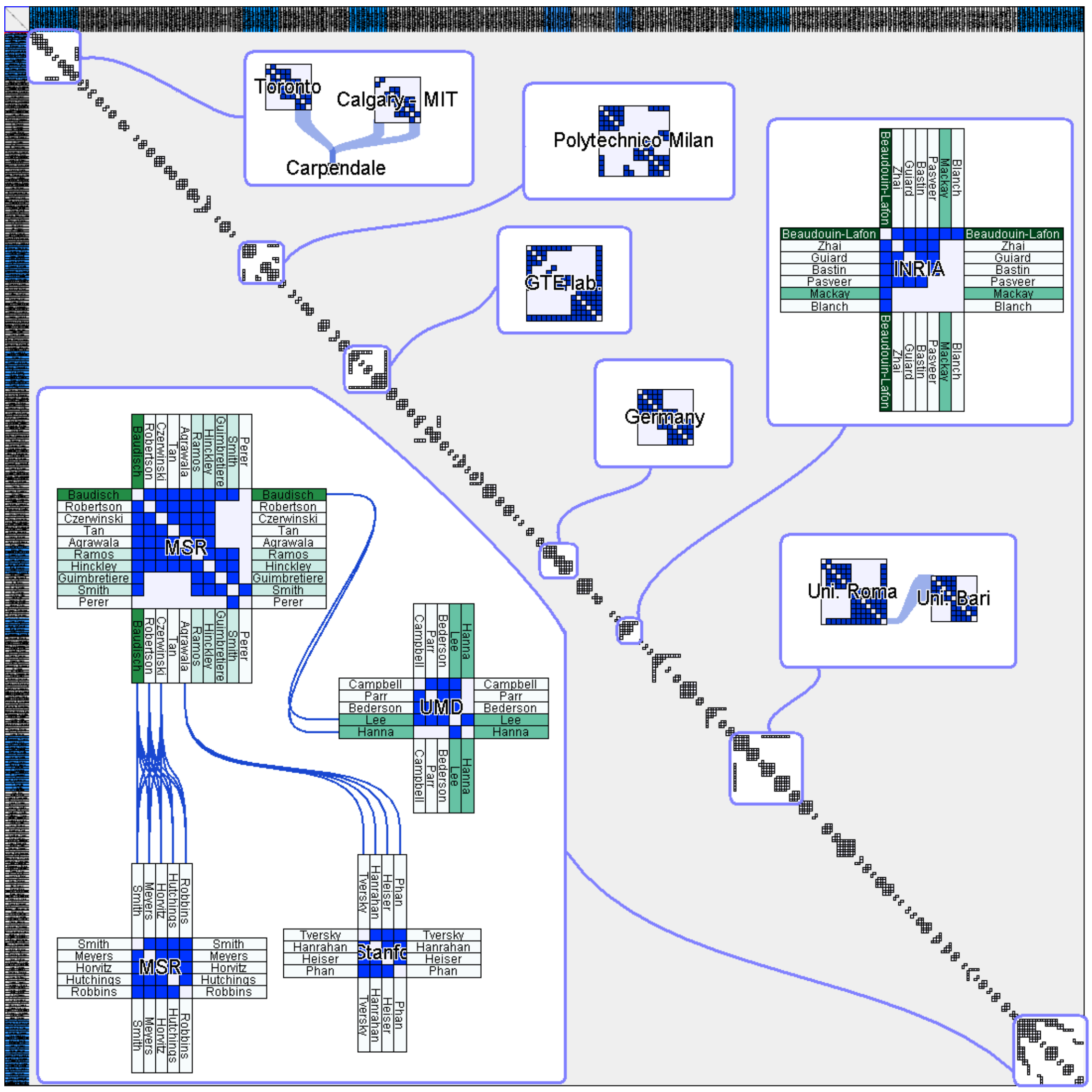


Figure 8.28: AVI co-authorship network is composed of many separate connected components. This figure shows the matrix of the complete network. Distinct connected components are visible in the matrix as non-connected blocks on the diagonal. Details of several of these components are shown in more details as NodeTrix representations with labels we consider representative. On the upper right of the matrix is the detailed component containing the most cited researcher in AVI. On the lower left of the matrix is the largest connected component.

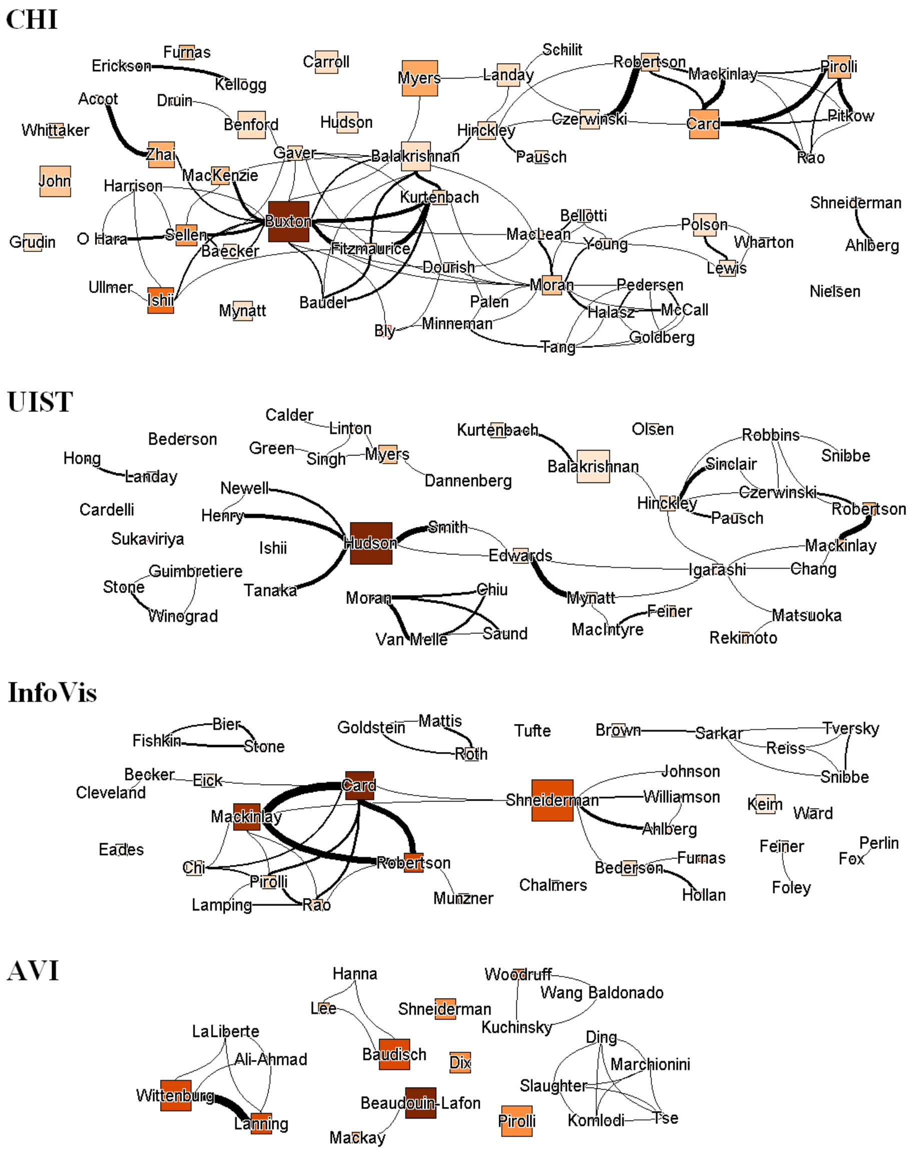


Figure 8.29: Co-authorship networks filtered by number of citations within the community. Nodes represent researchers: size shows the number of articles published to the conference, darkness shows the number of citations by articles of this conference. Links represent co-authorship, their width is the number of articles co-authored. These node-link diagrams use the LinLog layout with some manual modification to avoid label superposition.

8.5.4 Insights and Interpretation

In this section we try to interpret and summarize the results we collected during the analysis process.

Strategies to Produce Key Articles

In light of our data exploration, we identified several different "strategies" that the most-cited researchers (authors of key articles) could be said to follow.

Have the Right Idea at the Right Time Write a book or an article in an emerging field. For example, Edward Tufte's *The Visual Display of Quantitative Information* [Tuf83] presented key aspects of information visualizations just as personal computers and spreadsheets were giving a much larger group of people the ability to create them. A second example is George Furnas, who wrote his article on generalized fisheye views [Fur86] in the early years of the CHI conference.

Collaborate with Other Senior Researchers By working with other senior and respected members of a field, you can achieve much more than you can on your own. This strategy is clearly visible in Figure 8.10 where the collaboration Card-Mackinlay-Robertson emerges.

Supervise a Good Number of (Good) Students Work with your students to publish in few targeted conferences. This strategy is visible in the collaboration patterns of the key InfoVis researcher Ben Shneiderman (Figure 8.27) and the CHI key researcher — William Buxton (Figure 8.25a). The matrices in these Figures reveal large "crosses" for both of them, meaning that these authors have a high number of co-authors (students) who may not frequently collaborate with each other. As a bonus, if you chose and taught them well, and they become successful and prolific themselves, they may lift your numbers and connectivity even higher by collaborating with you. For example, the InfoVis section of Figure 8.19 shows the collaboration between Christopher Ahlberg and Ben Shneiderman.

Publish in the Right Conferences Select the venue for your papers wisely. The four conferences chosen for analysis in this paper are all well-regarded in the field; yet, there is a clear difference between their impact and average number of citations. The CHI conference remains the most prestigious of these, with the highest number of citations. However, UIST has a higher average number of citations per article, so it would appear that UIST holds a higher overall quality than all of the other conferences.

Collaboration Strategies

Whereas the previous publication strategies are based primarily on the researcher's own abilities, two more rely on collaboration. We identified two that depend strongly on the research environment. Co-authorship in non-academic research institutions such as PARC or Microsoft Research has a very different pattern from that in academia such as University of Toronto or the University of Maryland. Researchers in the non-academic institutions collaborate with one another more freely; so they appear in matrix representations such as (Figure 8.23) as blocks, showing that most of the researchers have co-authored several articles together. The appearance of academic research group collaborations has a completely different pattern: each professor and senior researcher has a cross pattern showing their co-authorship with a large number of students they advise. The students rarely publish with one another or with outside researchers without including their professor. For example, Figure 8.22 shows William Buxton's collaborators. These different patterns suggest that senior researchers within academic research group work on different topics or are in competition with each other, i.e. they relatively rarely collaborate directly with each other.

Our interpretation is that each of the above strategies is well-adapted for its institutional environment. In non-academic institutions, researchers are judged by the number of citations and their quality so they collaborate to produce the best possible articles. In contrast, universities insist on clear delineation of each researcher's contribution for tenure, promotion and other rewards; the more individualistic strategy adopted by most professors is rational: the merit of each non-student author is clear even if the overall impact is less.

Invisible Researchers

The visualizations and statistics only show one part of the picture. Non-American research centers are almost invisible. Why are so few authors from European, Asian and South American research centers listed among the top researchers? This question requires investigations deeper than the scope of this case study allows, but it should raise questions both for the selection process of the conferences and for the selection process of non-American research centers. Are conferences outside North-America being evaluated fairly? Is the review process of the CHI-UIST-InfoVis conferences strongly biased against non-native-English speaking researchers?

8.6 Conclusion

In this chapter, we presented our strategy to validate our work. We first reported the results of a second participatory design workshop, showing that our users were envisioning using and extending our prototypes (NodeTrix especially). This session also showed that they needed to go a step further, towards a system handling time-related data and heterogeneous sources of data. In a second part, we explained our method and choices for performing controlled experiments, thus collecting more quantitative data on the readability of the representations we designed. Finally, we demonstrated the benefit of using our system by performing a case study on 20 years of publication data in HCI. We selected this dataset because the whole HCI community benefits from its analysis. This case study, running over 4 months, raised a number of research problems but also numerous usability issues. It helped us to validate our work as well as prepare our system for a future longitudinal study.

We attempted to validate this Ph.D. work and demonstrate the benefit of its use for exploring large social networks: first, by informing the design and validating informally the system through participatory design sessions with social scientists, second by collecting quantitative data on the readability of our representations and finally by presenting the insights gained from using our tool to analyze 20 years of publication data in HCI.

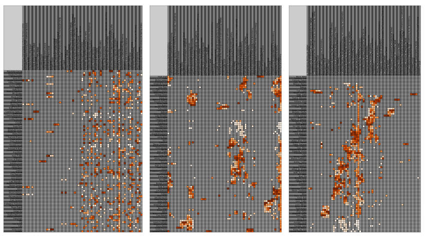
Conclusion

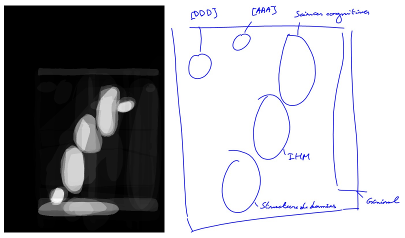
CHAPTER 9

In this thesis, we addressed the general problem of providing an interactive visualization to explore large social networks. This work bridges three main fields: social network analysis (SNA), graph drawing and information visualization. Our objective was to fill the gap between SNA and the visualization domain, *i.e.,* provide social scientists with a tool to visually explore their data.

Our approach was a mix of user-centered and participatory design: we collaborated with a small group of social scientists. Based on our hypothesis and the outcomes of our collaboration, we decided to investigate how matrix-based representations could help exploring social networks. In this chapter, we present a summary of our five contributions before concluding on future research directions.

9.1 Contributions


(a) Matrix reordering



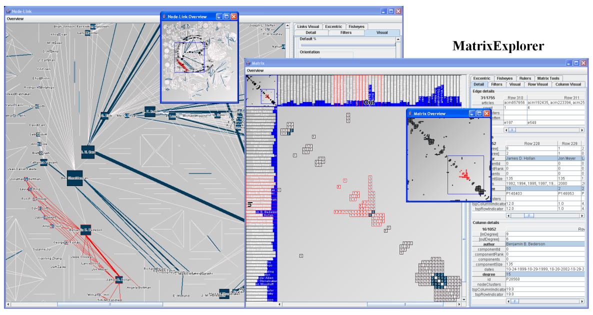
(b) Matrix readability

Figure 9.1: Matrix reordering techniques (a) and experiment on how they affect understanding (b)

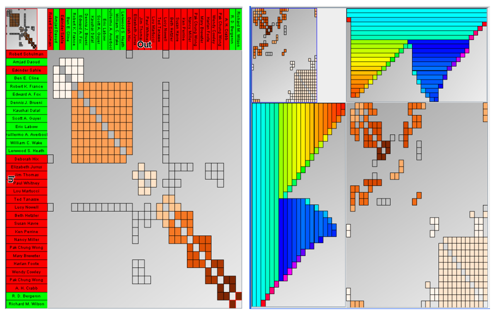
9.1.1 Matrix reordering

Research problem Matrix-based representations are readable for large and dense graphs whereas node-link diagrams suffer from node-overlapping and edge-crossing. However, Bertin [Ber83] showed that permuting rows and columns of a matrix could significantly improve the understanding of the represented data. Thus, the main research questions are to identify the different (re)ordering techniques, their characteristics and appropriate ways to assess their quality, (*i.e.*, how do they affect the understanding of the data).

Solution and evaluation There are numerous techniques to reorder tables, matrices or graphs. This problem has been studied across many fields and has many applications. While several surveys exist, they are often limited to the approach of one particular field. Our first contribution is to present a comprehensive overview across the different fields of the different categories of methods. We also present our attempt at characterizing how different orderings affect the understanding of visual tables. The user experiment we conducted proved that several orderings give different perspectives on the data, providing users with more insights and helping them to find a consensus among perspectives. We also observed that manipulating the data helps users understand it. These results are presented in chapter 4.



(a) MatrixExplorer



(b) Interactive clustering

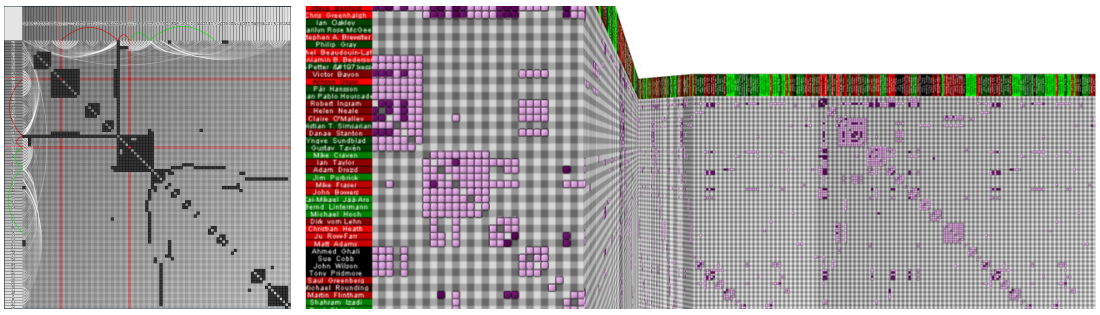
Figure 9.2: MatrixExplorer: an exploratory visualization system combining node-link diagrams and matrices (a) including interactive ordering and clustering (b).

9.1.2 MatrixExplorer

Research problem Our objective is to bridge the gap between information visualization and social network analysis. Existing works mostly rely on node-link diagrams and provide very limited interaction with the data. The challenge here is to provide social scientists with representations capable of handling larger datasets, while simultaneously allowing rich interactions to perform exploratory analysis. A secondary problem is to facilitate the use of the system, providing familiar representations and analysis measures.

Solution We propose to combine the strength of matrices and node-link diagrams. Matrices are able to handle large and dense networks whereas node-link diagrams are familiar representations very common to illustrate findings. We designed and implemented MatrixExplorer, a system coordinating these two representations. We form the hypothesis that matrices would be particularly well suited for exploration (as they can handle large datasets) and node-link diagrams for presentation purposes (as they are familiar to a wide audience). Thus, we focused on designing interaction techniques to manipulate matrices. We provided tools to perform interactive reordering (mostly designed from the empirical results we collected in the matrix reordering quality experiment), interactive clustering and favoring the discovery of consensus in the data. These results are detailed in chapter 5 and presented in two articles [HF06a, HF06b].

Evaluation MatrixExplorer is directly issued from the requirements we collected during a participatory design workshop with social scientists. We also attempted to validate it *a posteriori* by running a case study. This case study was performed on twenty years of publication data for four major conferences in HCI. Its outcomes raised several interesting research problems (section 5.3) and led us to our two other contributions MatLink and NodeTrix. Results and visualizations are reported in chapter 8.



(a) MatLink (b) Mélange

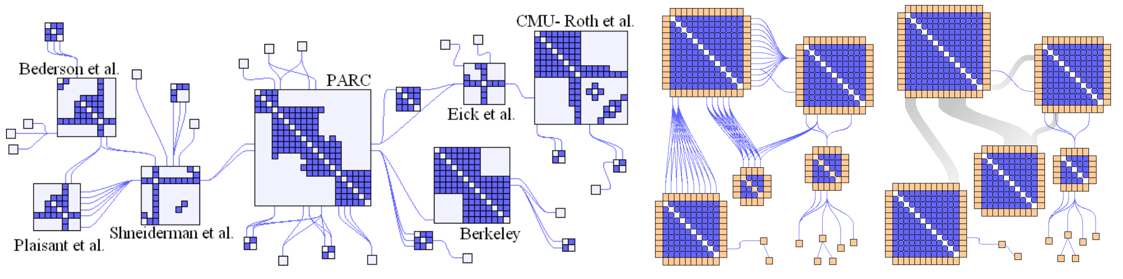
Figure 9.3: MatLink: an augmented matrix representation (a) and Mélange: a general purpose interaction technique to navigate through large visual spaces (b).

9.1.3 MatLink and Mélange

Research problems When analyzing social networks, a very common task is to follow paths in the network. For example, finding how many persons connect Mary and George or how many friends Jean-Daniel and Wendy have in common are key questions to understand the network structure. Performing these tasks on matrices is always possible (contrary to node-link diagrams that are not readable for dense networks). However, it is a very tedious process, requiring multiple readings of rows and columns. This task is particularly challenging in large matrices as it requires navigating back and forth through a large amount of space.

Solutions We proposed two solutions to this problem. First, we designed and implemented MatLink, an interactive visualization augmenting standard matrix representations. The principle is to overlay links on the sides of a standard matrix. MatLink provides both static links of the whole graph and interactive links showing a shortest path between selected actors of the network. Details are presented in section 6.2 and in [HF07]. Secondly, in collaboration with Niklas Elmqvist, member of the AVIZ team, we designed and implemented Mélange, an interaction technique to perform path-following tasks on very large matrices. This technique provides multiple foci points while preserving an awareness of the intermediate context. The principle is to fold the space in the intermediate region. Mélange is a general purpose interaction technique and can be applied to many other domains. Details are presented in section 6.3 and in [EHRF08].

Evaluations To evaluate this work, we performed two controlled experiments. The first one compared the readability of MatLink vs standard matrix and node-link representations. We used two types of tasks: general connectivity tasks for graphs and tasks dedicated to social network analysis (community and central actors identification). This experiment was performed by thirty-six subjects on six representative datasets. Our results showed that MatLink outperformed matrices and was even competitive with node-link diagrams. We performed a second controlled experiment to compare Mélange against standard single viewport and split-screen navigation techniques. We used a path-following task through three sizes of matrices with different levels of complexity. Our experiment showed that Mélange provides a better context awareness than split-screen and is faster than a single viewport. Details are provided in chapter 6 and the corresponding articles [HF07, EHRF08].



(a) NodeTrix (b) Node duplication

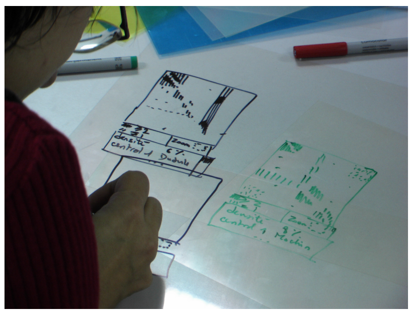
Figure 9.4: NodeTrix: a hybrid representation merging node-link and matrix representations (a) and a study on node duplication and their impact on readability (b).

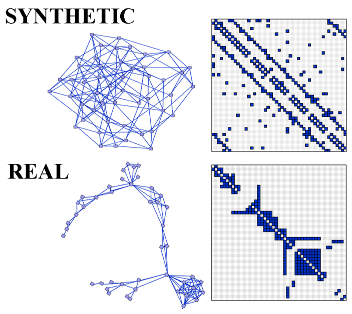
9.1.4 NodeTrix and node duplication

Research problems Small-world networks are very common amongst social networks. The characteristic of these networks is to present a globally sparse structure with locally dense communities. Therefore, the choice between node-link diagrams (effective for sparse graphs) or matrices (effective for dense graphs) remains unclear. Considering the two most important tasks of SNA — identifying communities and central actors — does not solve the problem. In fact, some matrices, when adequately reordered, allow the identification of communities as they appear in blocks. However, node-link diagrams better support the identification of central actors. A possible solution is to switch back and forth between the representations, however we observed that this process is cognitively demanding and requires multiple screens. A second problem is inherent to the identification of communities: where to place actors that are part of two or more communities? While it is possible to place them between the communities in node-link diagrams (potentially decreasing their readability), it is much more difficult in matrices where actors are placed along a linear order.

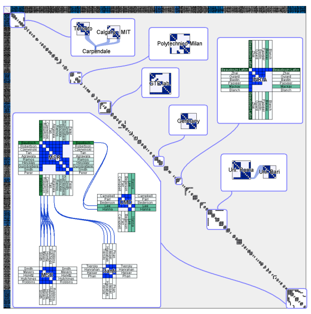
Solutions and evaluations To solve the initial problem of representing small-world networks, we designed and implemented NodeTrix, a representation merging node-link diagrams and matrices. This work is issued from a collaboration with Michael McGuffin, from the university of Montreal. NodeTrix represents the general structure of a social network as a node-link diagram and the dense communities as matrices. Therefore, it benefits of the readability of node-link diagram for sparse graphs and the readability of matrices for dense ones. To validate this work, we complemented the case study initiated with MatrixExplorer on publication data in HCI. We provided representations that are particularly effective to communicate our findings on communities and central actors in the HCI community. Detailed results can be found in chapter 7.2 and in [HFM07].

To solve the clustering ambiguity problem, in collaboration with Anastasia Bezerianos, member of National ICT Australia research institute, we proposed to duplicate an actor in all the communities he is part of. We proposed several designs and performed a study to understand how node duplication would impact the representation readability. Our results showed that node duplication (with explicit representation of the link between duplicated actors) does not decrease the readability of simple connectivity tasks and improves the correctness of community related tasks. Details of this study are presented in section 7.3.


(a) Participatory design



(b) Preparing data for evaluation


(c) Case study

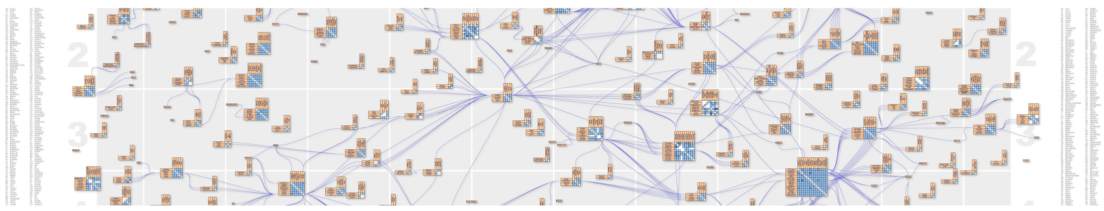

(d) Representing 20 years of collaboration at UIST (poster extract)

Figure 9.5: Outcomes of our collaboration with social scientists (a) and collection of requirements for a visual exploratory system, tasks and data selection for controlled experiment (b) and results of the analysis of 20 years of publication data in HCI (c-d).

9.1.5 Evaluation

Research problem Evaluating information visualization systems supporting exploratory data analysis is a challenge [Pla04] as the process is complex, long and very difficult to emulate in laboratory conditions. However, designing novel visualizations is of limited interest if we have no methodologies to evaluate them. To perform these evaluations, the InfoVis community needs to better understand how people perceive the visualizations, how they manipulate them and what factors lead to an effective exploration.

Elements of solution The problem of evaluation of InfoVis systems is a complex one, topic of a recent workshop series [BPS07]. In this dissertation, we report our approach and the results of our collaboration with social scientists. We present a set of general requirements for visually exploratory networks. We believe these results will guide future designs.

We also bring some elements of methodology for evaluating the readability of visual representations. We detail the problem we faced when selecting tasks and datasets for our controlled experiments. We briefly present the tasks taxonomy we established with researchers of the university of Maryland and detail our technique to generate experimental datasets from real ones.

To better understand the benefits and insights collected using our prototypes, we reported the results of our case study on publication data in HCI. These results not only constitute a validation of our work but we believe they are of general interest to researchers in HCI. Indeed, the analysis we present help us to better understand the field and its evolution. Details are presented in chapter 8.

9.2 Research directions

In this dissertation, we presented a panel of techniques to reorder matrices and initiated a reflection on how to assess the readability of their results (chapter 4). We combined matrices and node-link diagrams to explore large social networks (chapter 5). We augmented matrices for path-related tasks and improved the navigation in large ones (chapter 4). Then we merged node-link and matrices to explore small-world networks and helped solving clustering ambiguity by duplicating actors. Finally we presented our methodology to design and evaluate our work (chapter 3) and reported the complete analysis of our case study on HCI publication data (chapter 8).

There are many different ways to extend this research. Each of this chapter could be extended and a lot of work remains to be done when it comes to interaction and exploration of very large quantity of data (initiated with ZAME [FED$^+$08] in collaboration with the AVIZ team). Several concepts from this thesis could be generalized and studied: for instance the general idea of duplicating data and how it affect user understanding. In the following, we present the short-term and long-term directions we consider most promising.

Short-term projects We are currently working on two short-term projects on the problem of reordering matrices and assessing their quality as well as on the methodology for setting up a longitudinal study of our system.

Matrix reordering Performing a second empirical analysis on how people perceive matrix representations and the differences between the ordering. The main objective is to perform a complete survey of the techniques and attempt to identify a set of quality measures based on empirical evidence.

Longitudinal study Setting up a longitudinal study of our system. We already presented our strategy to log information at the recent BELIV workshop [1]. We plan on starting the study with a simplified version of NodeTrix available for a wide audience including expert and non-expert analysts.

Long-term vision Our more general research directions and our longer-term vision following this work deals with three main topics: evaluation, collaboration and story-telling.

Evaluation and logging The general theme of evaluation is an important problem in the InfoVis community. To improve the design of interaction and visualization techniques, we need to better understand how interactive system are used and what features effectively support the exploration of data. Along this line, several research problems remain to be solved. First, identifying what data should be logged and establishing an effective technology for logging is essential. Second, identifying appropriate methods for analyzing the collected data, potentially huge, complex and varying along time. Several work have been done to analyze time-varying data, however, it remains a rich direction to investigate.

[1]position paper available at `http://www.dis.uniroma1.it/~beliv08/`

Collaborative exploration One of the current trends of the InfoVis field is oriented towards collaborative visualization and analysis such as the ManyEyes project initiated by IBM Research [VWvH$^+$07] allowing distributed collaboration. Furthermore, research in HCI on large displays has known major advances. We would like to investigate how collaborative analysis can be performed on very large datasets visualized on large displays, that allow colocated collaboration. We believe several issues are interesting. First, how can we handle conflicts when several user interact on the same representation? Secondly, visual systems are cluttered by a very large number of controls, usually available through menus or a control panel located at a specific location. Finding a way to organize these controls to be easily available for several users and potentially integrating them to the visualizations is an interesting problem.

Story-telling and communication In recent years, visual analytics [TC05], a new thread of research closely linked to information visualization, has emerged. Work in this area focuses on integrating visualization and other techniques (data mining, linguistic analysis or document retrieval for example) in analysis tool to explore heterogeneous sources of data that usually are large and complex. We believe that one of the next important topics in visual analytics is the concept of story-telling: how to visually present findings and tell the story of the exploration. Because the objective and format of visual communication is quite different from exploratory visualization, research remains to be done to provide adequate tools for capturing data, organizing it and presenting it to the world.

Bibliography

General information visualization references

[Aes] Information aesthetics. http://infosthetics.com/. Accessed November 2007.

[AES05] Robert Amar, James Eagan, and John Stasko. Low-level components of analytic activity in information visualization. In *Proceedings of the 2005 IEEE Symposium on Information Visualization*, pages 111– 117. IEEE Computer Society, 2005.

[Ber83] Jacques Bertin. *Semiology of graphics*. University of Wisconsin Press, 1983.

[BPS07] Enrico Bertini, Catherine Plaisant, and Giuseppe Santucci. Beliv'06: beyond time and errors; novel evaluation methods for information visualization. *interactions*, 14(3):59–60, 2007.

[CM97a] Stuart K. Card and Jock Mackinlay. The structure of the information visualization design space. In *INFOVIS '97: Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, page 92, Washington, DC, USA, 1997. IEEE Computer Society.

[CM01] Andy Cockburn and Bruce J. McKenzie. 3d or not 3d? evaluating the effect of the third dimension in a document management system. In *Proceedings of SIG CHI 2001*, pages 434–441, 2001.

[CM02] Andy Cockburn and Bruce McKenzie. Evaluating the effectiveness of spatial memory in 2d and 3d physical and virtual environments, 2002.

[CMS99] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman. *Readings in information visualization: Using vision to think*. Morgan Kaufmann Publishers, San Francisco, 1999.

[ET07] Niklas Elmqvist and Philippas Tsigas. A taxonomy of 3d occlusion management techniques. In *Proceedings of the IEEE Conference on Virtual Reality 2007*, pages 51–58, 2007.

[Fek04] Jean-Daniel Fekete. The InfoVis Toolkit. In *Proceedings of the 2004 IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 167–174. IEEE Computer Society, 2004.

[Hea96] Christopher G. Healey. Choosing effective colours for data visualization. In *Proceedings of the IEEE Conference on Visualization*, pages 263–270, 1996.

[HL00] Jason I. Hong and James A. Landay. SATIN: A toolkit for informal ink-based applications. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 63–72, New York, NY, USA, 2000. ACM Press.

[HMS01] David J. Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. MIT Press, Cambridge, MA, 2001.

[HVW07] Jeffrey Heer, Fernanda B. Viégas, and Martin Wattenberg. Voyagers and voyeurs: Supporting asynchronous collaborative information visualization. In *ACM Human Factors in Computing System (CHI'07)*, 2007.

[IEE04] IEEE InfoVis 2004 Contest. available at `www.cs.umd.edu/hcil/iv04contest`, 2004.

[LP05] Sharon Laskowski and Catherine Plaisant. Evaluation methodologies for visual analytics. In *(section 6.1, Illuminating the Path, the Research and Development Agenda for Visual Analytics*, pages 150–157. Thomas, J., Cook, K. (Eds), IEEE Press, 2005.

[MK93] Michael J. Muller and Sarah Kuhn. Participatory design. *Commun. ACM*, 36(6):24–28, 1993.

[Nor06] Chris North. Toward measuring visualization insight. *IEEE Comput. Graph. Appl.*, 26(3):6–9, 2006.

[PF06] Raquel M. Pillat and Carla M. D. S. Freitas. Coordinating views in the infovis toolkit. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 496–499, New York, NY, USA, 2006. ACM.

[PFG08] Catherine Plaisant, Jean-Daniel Fekete, and Georges Grinstein. Promoting insight-based evaluation of visualizations: From contest to benchmark repository. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):120–134, 2008.

[Pla04] Catherine Plaisant. The challenge of information visualization evaluation. In *Proceedings of the AVI Conference*, pages 109–116, Gallipoli, Italy, 2004. ACM Press.

[PS06] Zachary Pousman and John Stasko. A taxonomy of ambient information systems: Four patterns of design. In *Advanced Visual Interfaces*, pages 67–74, may 2006.

[PSM07] Zachary Pousman, John Stasko, and Michael Mataas. Casual information visualization: Depictions of data in ever yday life. In *IEEE Information Visualization conference (InfoVis'07)*, november 2007.

[Shn96] Ben Shneiderman. The Eyes Have It: A Task by Data Taxonomy for Information Visualization. *Visual Languages*, pages 336–343, 1996.

[SP96] Alistair G. Sutcliffe and Uma Pater. 3d or not 3d: is it nobler to the mind? In *British Human Computer Interaction Conference*, pages 79–94. Cambridge University Press, 1996.

[SP04] Ben Shneiderman and Catherine Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction (4th Edition)*. Pearson Addison Wesley, 2004.

[SP06] Ben Shneiderman and Catherine Plaisant. Strategies for evaluating information visualization tools: multi-dimensional in-depth long-term case studies. In *BELIV '06: Proceedings of the 2006 AVI workshop on BEyond time and errors*, pages 1–7, New York, NY, USA, 2006. ACM.

[TC05] James J. Thomas and Kritin A. Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Ctr, 2005.

[Tre85] Anne Treisman. Preattentive processing in vision. *Computer Vision, Graphics, and Image Processing*, 31(2):156–177, 1985.

[Tuf83] Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut, 1983.

[Tuk77] John Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[VWvH$^+$07] Fernanda B. Viégas, Martin Wattenberg, Frank van Ham, Jesse Kriss, and Matt McKeon. Many eyes: A site for visualization at internet scale. In *IEEE Information Visualization Conference (InfoVis'07)*, 2007.

[War04] Colin Ware. *Information Visualization: Perception for Design (2nd Edition)*. Morgan Kauffman, 2004.

[WF96] Colin Ware and Glenn Franck. Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Transactions on Graphics*, 15(2):121–139, 1996.

[WTP$^+$95] James A. Wise, James J. Thomas, Kelly Pennock, David Lantrip, Marc Pottier, Anne Schur, and Vern Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 51–58, Washington, DC, USA, 1995. IEEE Computer Society.

Network visualization

[ACJM03] David Auber, Yves Chiricota, Fabien Jourdan, and Guy Melançon. Multiscale visualization of small world networks. In *Proceedings of the 2003 IEEE Symposium on Information Visualization*, pages 75–81. IEEE Press, 2003.

[Ada06] Eytan Adar. Guess: a language and interface for graph exploration. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 791–800, New York, NY, USA, 2006. ACM Press.

[AK02] James Abello and Jeffrey Korn. MGV: A system for visualizing massive multidigraphs. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):21–38, 2002.

[Aub02] David Auber. *Visualization tools for large data structures*. PhD thesis, University Bordeaux I, 2002.

[Aub03] David Auber. Tulip : A huge graph visualisation framework. In *Graph Drawing Software*, pages 105–126. Springer-Verlag, 2003.

[AvH04] James Abello and Frank van Ham. Matrix zoom: A visual interface to semi-external graphs. In *Proceedings of the 2004 IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 183–190, Austin, Texas, 2004. IEEE Computer Society.

[AvHK06] James Abello, Frank van Ham, and Neeraj Krishnan. Ask-graphview: A large scale graph visualization system. *IEEE TVCG journal*, 12(5):669–676, 2006.

[BEW95] Richard A. Becker, Stephen G. Eick, and Allan R. Wilks. Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28, 1995.

[BP03] Danah Boyd and Jeff Potter. An interactive tool for exploring digital social connections. *Sketch at SIGGRAPH 2003*, July 27-31 2003.

[BW04] Ulrik Brandes and Dorothea Wagner. VisOne - analysis and visualization of social networks. In Michael Jünger and Petra Mutzel, editors, *Graph Drawing Software*, pages 321–340, 2004.

[CF] Commentflow, based on semaspace. `http://web.media.mit.edu/~dietmar/myspace.html`. Accessed March 2008.

[CM00] Andy Cockburn and Bruce McKenzie. An evaluation of cone trees. *People and Computers XV (Proceedings of the 2000 British Computer Society Conference on HumanComputer Interaction.*, 2000.

[CSP$^+$06] Stuart K. Card, Bongwon Suh, Bryan Pendleton, Jeffrey Heer, and John W. Bodnar. Timetree: Exploring time changing hierarchies. *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, 2006.

[Don95] Judith S. Donath. Visual who: animating the affinities and activities of an electronic community. In *MULTIMEDIA '95: Proceedings of the third ACM international conference on Multimedia*, pages 99–107, New York, NY, USA, 1995. ACM.

[EHK$^+$03] Cesim Erten, Phillip J. Harding, Stephen G. Kobourov, Kevin Wampler, and Gary Yee. Graphael: Graph animations with evolving layouts. In *Symposium on Graph Drawing GD'03*, pages 98–110, 2003.

[ERG02] Peter Eklund, Nataliya Roberts, and Steve Green. OntoRama: Browsing RDF ontologies using a hyperbolic-style browser. In *CW '02: Proceedings of the First International Symposium on Cyber Worlds (CW'02)*, page 0405, Washington, DC, USA, 2002. IEEE Computer Society.

[FED$^+$08] Jean-Daniel Fekete, Niklas Elmqvist, Thanh-Nghi Do, Howard Goodell, and Nathalie Henry. Navigating wikipedia with the zoomable adjacency matrix explorer. *Proceedings of Pacific Visualization conference*, 2008.

[FG] Flickr graph. `http://www.marumushi.com/apps/flickrgraph/`. Accessed November 2007.

[Fle] Thomas Fletcher. Facebook friend's wheel. `http://thomas-fletcher.com/friendwheel/`. Accessed March 2008.

[FOS$^+$05] Danyel Fisher, Joshua O'madadhain, Padhraic Smyth, Scott White, and Yan-Biao Boey. Analysis and visualization of network data using jung. *Journal of Statistical Software*, 2005.

[FWDP03] Jean-Daniel Fekete, David Wang, Niem Dang, and Catherine Plaisant. Overlaying graph links on treemaps. IEEE Symposium on Information Visualization Conference Compendium (demonstration), October 2003.

[Gen] GenoPro. `http://www.genopro.com/`.

[GFC05] Mohammad Ghoniem, Jean-Daniel Fekete, and Philippe Castagliola. On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization*, 4(2):114–135, 2005.

[GPB02] Jesse Grosjean, Catherine Plaisant, and Ben B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. *IEEE Symposium on Information Visualization (InfoVis'02)*, pages 57–64, 2002.

[Ham05] Frank Van Ham. *Interactive Visualization of Large Graphs*. PhD thesis, Technische Universiteit Eindhoven, 2005.

[HB05] Jeffrey Heer and Danah Boyd. Vizster: Visualizing Online Social Networks. In *Proceedings of the IEEE Symposium on Information Visualization*, page 5, 2005.

[HBF08] Nathalie Henry, Anastasia Bezerianos, and Jean-Daniel Fekete. Improving the readability of clustered social networks by node duplication. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of Visualization/Information Visualization 2008)*, 14(6), 2008.

[HF06a] Nathalie Henry and Jean-Daniel Fekete. MatrixExplorer: a Dual-Representation System to Explore Social Networks. *IEEE TVCG (Infovis'06 proceedings)*, 12(5):677–684, 2006.

[HF06b] Nathalie Henry and Jean-Daniel Fekete. Matrixexplorer: Un systéme pour l'analyse exploratoire de réseaux sociaux. *Proceedings of IHM2006, International Conference Proceedings Series*, pages 67–74, September 2006.

[HF07] Nathalie Henry and Jean-Daniel Fekete. Matlink: Enhanced matrix visualization for analyzing social networks. In *Lecture Notes in Computer Science (Proceedings of the 13th IFIP TC13 International Conference on Human-Computer Interaction, INTERACT'07)*, volume 4663, pages 288–302, 2007.

[HFM07] Nathalie Henry, Jean-Daniel Fekete, and Michael McGuffin. Nodetrix: Hybrid representation for analyzing social networks. *TVCG (Proceedings of IEEE Information Visualization conference)*, 13(6):1302–1309, 2007.

[HGEF07] Nathalie Henry, Howard Goodell, Niklas Elmqvist, and Jean-Daniel Fekete. 20 years of 4 hci conferences: a visual exploration. *International Journal of Human Computer Interaction Ñ Reflections on Human-Computer Interaction, A Special Issue in Honor of Ben Shneiderman's 60th Birthday*, 23(3):239–285, 2007.

[HMM00] Ivan Herman, Guy Melançon, and Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE TVCG*, 6(1):24–43, 2000.

[Hol06] Danny Holten. Hierarchical edge bundles: Visualizaiton of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[KEC06] René Keller, Claudia M. Eckert, and P. John Clarkson. Matrices or node-link diagrams: which visual representation is better for visualising connectivity models? *Information Visualization*, 5(1):62–76, 2006.

[KPLB06] Hyunmo Kang, Catherine Plaisant, Bongshin Lee, and Benjamin B. Bederson. Netlens: Iterative exploration of content-actor network data. *Proc. of IEEE VAST*, pages 91–98, 2006.

[KSS97] Henry Kautz, Bart Selman, and Mehul Shah. Referral Web: Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.

[LCRB04] Bongshin Lee, Mary Czerwinski, George Robertson, and Benjamin B. Bederson. Understanding eight years of infovis conferences using paperlens. In *INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04)*, page 216.3, Washington, DC, USA, 2004.

[Lee06] Bongshin Lee. *Interactive Visualization for Trees and Graphs*. PhD thesis, University of Maryland, 2006.

[LPP$^+$06] Bongshin Lee, Cynthia S. Parr, Catherine Plaisant, Benjamin B. Bederson, Vladislav D. Veksler, Wayne D. Gray, and Christopher Kotfila. Treeplus: Interactive exploration of networks with enhanced tree layouts. *IEEE TVCG (Infovis'06 proceedings)*, 12(6):1414–1426, 2006.

[LR96] John Lamping and Ramana Rao. The Hyperbolic Browser: A focus + context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–35, 1996.

[Mun97] Tamara Munzner. H3: Laying out large directed graphs in 3d hyperbolic space. *Symposium on Information Visualization (InfoVis Õ97)*, pages 2–10, 1997.

[Mun98a] Tamara Munzner. Drawing large graphs with h3viewer and site manager. *Symposium on Graph Drawing (GD Õ98), Lecture Notes in Computer Science*, 1547:384–393, 1998.

[Mun98b] Tamara Munzner. Exploring large graphs in 3d hyperbolic space. *IEEE Computer Graphics and Applications*, 18(4):18–23, 1998.

[Mun00] Tamara Munzner. *Interactive Visualization of Large Graphs and Networks*. PhD thesis, Stanford University, 2000.

[PLP$^+$06] Catherine Plaisant, Bongshin Lee, Cynthia Sims Parr, Jean-Daniel Fekete, and Nathalie Henry. Task taxonomy for graph visualization. In *BELIV'06 workshop*, pages 82–86, Venice, Italy, 2006. ACM Press.

[PS06] Adam Perer and Ben Shneiderman. Balancing Systematic and Flexible Exploration of Social Networks. *IEEE TVCG (Infovis'06 proceedings)*, 12(5):693–700, 2006.

[RCMC00] Kirsten Risden, Mary Czerwinski, Tamara Munzner, and Dan Cook. An initial examination of ease of use for 2D and 3D information visualizations of web content. *International Journal of Human Computer Studies*, 53(5):695–714, November 2000.

[RMC91] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: Animated 3D visualizations of hierarchical information. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 189–194, New York, NY, USA, 1991. ACM Press.

[SA06] Ben Shneiderman and Alex Aris. Network visualization by semantic substrates. *IEEE TVCG (Infovis'06 proceedings)*, 12(5), 2006.

[SGLS07] John Stasko, Carsten Gorg, Zhicheng Liu, and Kanupriya Singhal. Jigsaw: Supporting investigative analysis through interactive visualization. *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, pages 131–138, 2007.

[Shn92] Ben Shneiderman. Tree visualization with tree-maps: A 2-D space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, January 1992.

[SM07] Zeqian Shen and Kwan-Liu Ma. Path visualization for adjacency matrices. In *Proceedings of the Eurographics/IEEE VGTC Symposium on Visualization*, pages 83–90, 2007.

[TG] Touchgraph llc. facebook touchgraph. `http://www.touchgraph.com/TGFacebookBrowser.html`. Accessed March 2008.

[TSS] True sparrow system ltd. facebook interactive friends graph. `http://www.facebook.com/apps/application.php?id=4079090761&b&ref=pd`. Accessed March 2008.

[VBN$^+$04] Fernanda B. Viégas, Danah Boyd, David H. Nguyen, Jeffrey Potter, and Judith Donath. Digital artifacts for remembering and storytelling: Posthistory and social network fragments. In *Hawaii International Conference on System Sciences*, 2004.

[VC] Visual complexity. `http://www.visualcomplexity.com/`. Accessed November 2007.

[vH03] Frank van Ham. Using multilevel call matrices in large software projects. In *Proceedings of the 2003 IEEE Symposium on Information Visualization*, pages 227–232, Seattle, WA, USA, 2003. IEEE Press.

[vHvW04] Frank van Ham and Jarke J. van Wijk. Interactive visualization of small world graphs. In *INFOVIS '04: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS'04)*, pages 199–206, Washington, DC, USA, 2004. IEEE Computer Society.

[Wat02] Martin Wattenberg. Arc diagrams: Visualizing structure in strings. In *IEEE Symposium of Information Visualization (InfoVis'02)*, 2002.

[Wat06] Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the CHI conference*, pages 811–819, Montréal, Québec, Canada, 2006. ACM Press.

[Wil98] Graham J. Wills. Niche works Ñ interactive visualization of very large graphs. *Symposium on Graph Drawing GD Õ97*, pages 403–415, 1998.

[Wil99] Graham J. Wills. Niche works Ñ interactive visualization of very large graphs. *Journal of Computational and Graphical Statistics*, 8(3):190–212, june 1999.

[ZKB02] Jurgen Ziegler, Christoph Kunz, and Veit Botsch. Matrix browser: Visualizing and exploring large networked information spaces. In *Extended Abstracts of Human Factors in Computing Systems (CHI'02)*, pages 602–603, 2002.

[ZMC05] Shengdong Zhao, Michael J. McGuffin, and Mark H. Chignell. Elastic hierarchies: Combining treemaps and node-link diagrams. In *Proceedings of IEEE Symposium on Information Visualization (InfoVis)*, pages 57–64, October 2005.

Graph layout, clustering and matrix reordering algorithms

[ABH99] Jonathan E. Atkins, Erik G. Boman, and Bruce Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM J. Comput.*, 28(1):297–310, 1999.

[AMA07] Daniel Archambault, Tamara Munzner, and David Auber. Topolayout: Multilevel graph layout by topological features. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):305–317, 2007.

[Ber02] Pavel Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

[BJDG$^+$02] Ziv Bar-Joseph, Erik D. Demaine, David K. Gifford, Angèle M. Hamel, Tommi Jaakkola, and Nathan Srebro. K-ary clustering with optimal leaf ordering for gene expression data. In *WABI '02: Proceedings of the Second International Workshop on Algorithms in Bioinformatics*, pages 506–520, London, UK, 2002. Springer-Verlag.

[BK77] William H. Benson and Bernard Kitous. Interactive analysis and display of tabular data. In *SIGGRAPH '77: Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, pages 48–53, New York, NY, USA, 1977. ACM Press.

[BMK95] Jim Blythe, Cathleen McGrath, and David Krackhardt. The effect of graph layout on inference from social network data. *Proceedings of the graph drawing conference (GD'95)*, pages 40–51, 1995.

[Bra07] Ulrik Brandes. Optimal leaf ordering of complete binary trees. *J. of Discrete Algorithms*, 5(3):546–552, 2007.

[BYEFN01] Reuven Bar-Yehuda, Guy Even, Jon Feldman, and Joseph (Seffi) Naor. Computing an optimal orientation of a balanced decomposition tree for linear arrangement problems. *Journal of Graph Algorithms and Applications*, 5(4):1–27, 2001.

[CHHGWJ$^+$04] Chen Chun-Houh, Hwu Hai-Gwo, Jang Wen-Jung, Kao Chiun-How, Tien Yin-Jing, Tzeng ShengLi, and Wu Han-Ming. Matrix visualization and information mining. In *Proceedings in Computational Statistics 2004 (Compstat 2004)*, pages 85–100. Physika Verlag, Heidelberg, 2004.

[Chu97] Fan R. K. Chung. *Spectral Graph Theory*. AMS Publication, 1997.

[CP05] Gilles Caraux and Sylvie Pinloche. Permutmatrix: A graphical environment to arrange gene expression profiles in optimal linear order. *Bioinformatics*, 21:1280–1281, 2005.

[CR95] Jean-Hugues Chauchat and Alban Risson. Amado, a new method and a software integrating jacques bertin's graphics and multidimensional data analysis methods. In *International Conference on VIsualization of Categorical Data*, Köln, R.F.A, 1995.

[CZ04] Sharlee Climer and Weixiong Zhang. Take a walk and cluster genes: a tsp-based approach to optimal rearrangement clustering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 22, New York, NY, USA, 2004. ACM Press.

[DBF05a] Patrick Doreian, Vladimir Batagelj, and Anuska Ferligoj. *Generalized Blockmodeling*. Structural Analysis in the Social Sciences. Cambridge Univ. Press, 2005.

[DETT98] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, 1998.

[DH04] Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. *International conference on machine learning (ICML 2004)*, pages 225–232, 2004.

[DLR77] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society*, 39(1):1–38, 1977.

[DMM97] Ugur Dogrusöz, Brendan Madden, and Patrick Madden. Circular layout in the graph layout toolkit. In *GD '96: Proceedings of the Symposium on Graph Drawing*, pages 92–100, London, UK, 1997. Springer-Verlag.

[Don00] Stijn .M. Van Dongen. *Graph Clustering By Flow Simulation*. PhD thesis, Universiteit Utrecht, 2000.

[DPS02] Josep Díaz, Jordi Petit, and Maria Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, 2002.

[Ead84] Peter Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.

[Ead92] Peter Eades. Drawing free trees. In *Bulletin of the Institute for Combinatorics and its Applications*, pages 10–36, 1992.

[EdM95] Peter Eades and Candido F. X. de Mendonca. Vertex splitting and tension-free layout. *Proc. of Graph Drawing'95, LNCS*, 1027:202–211, 1995.

[Eig03] Markus Eiglsperger. *Automatic layout of UML class diagrams: a topology-shape-metrics approach*. PhD thesis, Eberhart-Karls-Universität Tübingen, 2003.

[EKS03] Markus Eiglsperger, Michael Kaufmann, and Martin Siebenhaller. A topology-shape-metrics approach for the automatic layout of uml class diagrams. In *SoftVis '03: Proceedings of the 2003 ACM symposium on Software visualization*, pages 189–ff, New York, NY, USA, 2003. ACM.

[Els97] Ulrich Elsner. *Graph partitioning: a survey*. Technische Universitat Chemnitz, 1997.

[Eng02] Andries Engelbrecht. *Computational Intelligence: An Introduction*. Wiley & Sons, 2002.

[ESBB98] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 95:14863–14868, 1998.

[Fau88] K. Faust. Comparison of methods for positional analysis: Structural and. general equivalences. *Social networks*, 10:313–341, 1988.

[FB04] Mountaz Hascoët François Boutin. Cluster validity indices for graph partitioning. *Proceedings of IV 2004*, 2004.

[Fja98] Per-Olof Fjallstrom. Algorithms for graph partitioning: A survey. *Linkoping Electronic Articles in Computer and Information Science*, 3, 1998.

[FK46] E. Forsyth and L. Katz. A matrix approach to the analysis of sociometric data: a preliminary report. *Sociometry*, 9(4):340–347, 1946.

[FR91] Thomas M.J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software - Practice and Experience*, 21(11):1129–1164, 1991.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completness*. W. H. Freeman, San Francisco, 1979.

[GK01] Pawel Gajer and Stephen G. Kobourov. Grip: Graph drawing with intelligent placement. *Symposium of Graph Drawing (GD'00), Lecture Notes in Computer Science*, 1984:222–228, 2001.

[GN02] Michelle Girvan and Mark E.J. Newman. Community structure in social and biological networks. *National academy of science*, 99:7821–7826, 2002.

[Hal70] K. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17(3), 1970.

[Har75] John A. Hartigan. Clustering algorithms. *Wiley*, 1975.

[Hel00] Keld Helsgaun. An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000.

[HF06] Nathalie Henry and Jean-Daniel Fekete. Evaluating visual table data understanding. In *BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV'06)*, Venice, Italy, 2006. ACM Press.

[HJ04] Stefan Hachul and Michael Junger. Drawing large graphs with a potential-field-based multilevel algorithm (extended abstract). *Symposium of Graph Drawing (GD'04), Lecture Notes in Computer Science*, 3383:285–295, 2004.

[HJ06] Stefan Hachul and Michael Jünger. An experimental comparison of fast algorithms for drawing general large graphs. In Patrick Healy and Nikola S. Nikolov, editors, *Graph Drawing, Limerick, Ireland, September 12-14, 2005*, pages pp. 235–250. Springer, 2006.

[HK00a] David Harel and Yehuda Koren. A fast multi-scale method for drawing large graphs. *Symposium of Graph Drawing (GD'00), Lecture Notes in Computer Science*, 1984:183–196, 2000.

[HK00b] David Harel and Yehuda Koren. Graph drawing by high-dimensional embedding. *Symposium on Graph Drawing (GD Õ02), Lecture Notes in Computer Science*, 2528:207–219, 2000.

[HK01] David Harel and Yehuda Koren. Clustering spatial data using random walks. In *Knowledge Discovery and Data Mining (KDD'01)*, pages 281–286, 2001.

[HMdRD99] Ivan Herman, Guy Melançon, Maurice M. de Ruiter, and Maylis Delest. Latour — A tree visualisation system. *Proceedings of the Graph Drawing symposium (GDÕ99), LNCS*, 1731:392–399, 1999.

[JMF99] Anil K. Jain, M. Narasimha Murty, and Patrick J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[KCH03] Yehuda Koren, Liran Carmel, and David Harel. Drawing huge graphs by algebraic multigrid optimization. *Multiscale Modeling and Simulation*, 1(4):645–673, 2003.

[KH02] Yehuda Koren and David Harel. A multi-scale algorithm for the linear arrangement problem. In *WG '02: Revised Papers from the 28th International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 296–309, London, UK, 2002. Springer-Verlag.

[KK89] Tomohisa Kamada and Satoru Kawai. An algorithm for drawing general undirected graphs. In *Information Processing Letters*, volume 31, pages 7–15, 1989.

[KL70] Brian W. Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graph. *Bell Syst. Tech. Journal*, 49(2):291–307, 1970.

[KMN$^+$02] Tapas Kanungo, David M. Mount, Nathan Netanyahu, Christine Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892, 2002.

[Kor05] Yehuda Koren. Drawing graphs by eigenvectors: Theory and practice. *Computers and Mathematics with Applications*, 49(11–12):1867–1888, 2005.

[KST89] Shojiro Tagawa Kozo Sugiyama and Mitshuhiko Toda. Methods for visual understanding of hierarchical systems structures. *IEEE Transactions on systems, man and cybernetics*, SMC-11(2):109–125, 1989.

[Len74] J. K. Lenstra. Clustering a data array and the traveling salesman problem. *Operations Research Letters*, 22:412–414, 1974.

[MBK97] Cathleen McGrath, Jim Blythe, and Davis Krackhardt. The effect of spatial arrangement on judgments and errors in interpreting graphs. *Social Networks*, 19(3):223–242, 1997.

[Mel06] Guy Melançon. Just how dense are dense graphs in the real world?: a methodological note. In *BELIV '06: Proceedings of the 2006 AVI workshop on BEyond time and errors*, pages 1–7, New York, NY, USA, 2006. ACM.

[MH00] Guy Melançon and Ivan Herman. DAG drawing from an information visualization perspective. In R. van Liere, editor, *Data Visualization'00*, Amsterdam, The Netherlands, 2000. Springer-Verlag.

[MML07] Christopher Mueller, Benjamin Martin, and Andrew Lumsdaine. A comparison of vertex ordering algorithms for large graph visualization. In *Asia-Pacific Symposium on Visualization (APVIS'07)*, February 2007.

[MS05] Erkki Mäkinen and Harri Siirtola. The barycenter heuristic and the reorderable matrix. *Informatica*, page to appear, 2005.

[NJW01] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Proceedings of Advances in Neural Information Processing Systems*, 14, 2001.

[Noa05] Andreas Noack. Energy-based clustering of graphs with nonuniform degrees. In Patrick Healy and Nikola S. Nikolov, editors, *Proceedings of the 13th International Symposium on Graph Drawing (GD 2005)*, pages 309–320, Limerick, Ireland, 2005. Springer-Verlag.

[Ols95] Clark F. Olson. Parallel algorithms for hierarchical clustering. *Parallel Computing*, 21:1313–1325, 1995.

[PCJ95] Helen Purchase, Robert F. Cohen, and Murray James. Validating graph drawing aesthetics. *Proceedings of Graph Drawing'95*, pages 435–446, 1995.

[Pet01] J. Petit. Experiments on the minimum linear arrangement problem, 2001.

[PHG06] Helen Purchase, Eve E. Hoggan, and Carsten Görg. How important is the 'mental map'? - an empirical investigation of a dynamic graph layout algorithm. *Proceedings of Graph Drawing'06*, pages 184–195, 2006.

[PL06] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. *Journal of Graph Algorithms and Applications (JGAA)*, 10(2):191–218, 2006.

[Pur97] Helen Purchase. Which aesthetic has the greatest effect on human understanding? *Proceedings of Graph Drawing'97*, pages 248–261, 1997.

[RC94] Ramana Rao and Stuart K. Card. The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322, New York, NY, USA, 1994. ACM Press.

[Rob51] W. S. Robinson. A method for chronologically ordering archaeological deposits. *American Antiquity*, 16:293–301, 1951.

[RT81] Edward M. Reingold and John S. Tilford. Tidier drawing of trees. *IEEE Transactions on Software Engineering*, SE-7(2):223–228, 1981.

[SBB96] Michael Spenke, Christian Beilken, and Thomas Berlage. Focus: the interactive table for product comparison and selection. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 41–50, New York, NY, USA, 1996. ACM Press.

[Sii99] Harri Siirtola. Interaction with the reorderable matrix. In E. Banissi, F. Khosrowshahi, M. Sarfraz, E. Tatham, and A. Ursyn, editors, *Proceedings of the International Conference on Information Visualization IV'99*, pages 272 –277, London, 1999. IEEE Computer Society.

[SM05] Harri Siirtola and Erkki Mäkinen. Constructing and reconstructing the reorderable matrix. *Information Visualization*, 4(1):32–48, 2005.

[Tam87] Roberto Tamassia. On embedding a graph in the grid with the minimum number of bends. *SIAM Journal on Computing*, 16(3):421–444, 1987.

[TY85] Michel Tenenhaus and F. W. Young. An analysis and synthesis of multiple correspondence analysis, optimal scaling, dual scaling, homogeneity analysis, and other methods of quantifying categorical multivariate data. *Psychometrika*, 50(1):91–119, 1985.

[War63] Joe H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):235–244, 1963.

[WPCM02] Colin Ware, Helen Purchase, Linda Colpoys, and Matthiew McGill. Cognitive measurements of graph aesthetics. *Journal of information visualization*, 1(2):248–261, 2002.

[Zah71] Charles T. Zahn. Graph theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, C-20(1):68–86, 1971.

Interaction techniques and animation

[AF06] Caroline Appert and Jean-Daniel Fekete. Orthozoom scroller: 1d multi-scale navigation. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 21–30, New York, NY, USA, 2006. ACM.

[AWS92] Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman. Dynamic queries for information exploration: An implementation and evaluation. In *Proceedings of the ACM CHI'92 Conference on Human Factors in Computing Systems*, pages 619–626, New York, NY, USA, 1992. ACM Press.

[Bar97a] Lyn Bartram. Can motion increase user interface bandwidth? In *Proc. IEEE Conference on Systems, Man, and Cybernetics'97*, pages 1686–1692, 1997.

[BH94] Ben B. Bederson and James Hollan. Pad++: A zooming graphical interface for exploring alternative interface physics. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 17–26, New York, NY, USA, 1994. ACM Press.

[BLH04a] Patrick Baudisch, Bongshin Lee, and Libby Hanna. Fishnet: a fisheye web browser with search term popouts. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 133–140, New York, NY, USA, 2004. ACM Press.

[BS90a] Ronald M. Baecker and Ian Small. Animation at the interface, 1990. A chapter (pp. 251–267) in Brenda Laurel, editor, The Art of Human-Computer Interface Design, Addison-Wesley.

[BSP$^+$93] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony DeRose. Toolglass and Magic Lenses: The see-through interface. In *Computer Graphics (SIGGRAPH '93 Proceedings)*, pages 73–80, New York, NY, USA, 1993. ACM Press.

[BW97] Ulrik Brandes and Dorthea Wagner. A bayesian paradigm for dynamic graph layout. In *Symposium on Graph Drawing GD'97*, pages 236–247. Springer-Verlag, 1997.

[BW02] Lyn Bartram and Colin Ware. Filtering and brushing with motion. *Information Visualization*, 1(1):66–79, 2002.

[BWC03] Lyn Bartram, Colin Ware, and Tom Calvert. Moticons: detection, distraction and task. *Int. J. Hum.-Comput. Stud.*, 58(5):515–545, 2003.

[Car99] M. Sheelagh T. Carpendale. *RA Framework for Elastic Presentation Space*. PhD thesis, Simon Fraser University, 1999.

[CCF95] M. Sheelagh T. Carpendale, David J. Cowperthwaite, and F. David Fracchia. 3-dimensional pliable surfaces: for the effective presentation of visual information. In *UIST '95: Proceedings of the 8th annual ACM symposium on User interface and software technology*, pages 217–226, New York, NY, USA, 1995. ACM.

[CM01] M. Sheelagh T. Carpendale and Catherine Montagnese. A framework for unifying presentation space. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 61–70, 2001.

[CRM91] Stuart K. Card, George G. Robertson, and Jock D. Mackinlay. The information visualizer, an information workspace. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 181–188, New York, NY, USA, 1991. ACM Press.

[EHRF08] Niklas Elmqvist, Nathalie Henry, Yann Riche, and Jean-Daniel Fekete. Mélange: space-folding for multi-focus interaction. In *CHI '08: Proceedings of the SIGCHI conference on Human Factors in computing systems*, New York, NY, USA, 2008. ACM.

[Emm08] Emmanuel Pietriga and Caroline Appert. Sigma Lenses: Focus-Context Transitions Combining Space, Time and Translucence. In *Proceedings of ACM CHI 2008 Conference on Human Factors and Computing Systems*, pages 1343–1352. ACM Press, April 2008.

[FP99] Jean-Daniel Fekete and Catherine Plaisant. Excentric labeling: dynamic neighborhood labeling for data visualization. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 512–519, 1999.

[Fur86] George W. Furnas. Generalized fisheye views. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, pages 16–23, New York, NY, USA, 1986. ACM Press.

[Fur06] George W. Furnas. A fisheye follow-up: further reflections on focus + context. In *Proceedings of the ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 999–1008, 2006.

[HEW98] Mao Lin Huang, Peter Eades, and Junhu Wang. Online animated graph drawing using a modified spring algorithm. *Journal of visual languages and computing,*, 9(6), 1998.

[IGY06] Pourang Irani, Carl Gutwin, and Xing Dong Yang. Improving selection of off-screen targets with hopping. In *Proceedings of the ACM CHI 2006 Conference on Human Factors in Computing Systems*, pages 299–308, 2006.

[MGT+03] Tamara Munzner, François Guimbretière, Serdar Tasiran, Li Zhang, and Yunhong Zhou. TreeJuxtaposer: scalable tree comparison using focus+context with guaranteed visibility. In *Proceedings of ACM SIGGRAPH 2003*, pages 453–462, 2003.

[MRC91a] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The perspective wall: Detail and context smoothly integrated. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 173–179, New York, NY, USA, 1991. ACM Press.

[MRC91b] Jock D. Mackinlay, George G. Robertson, and Stuart K. Card. The Perspective Wall: Detail and context smoothly integrated. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, pages 173–179, 1991.

[NBM+06] Dmitry Nekrasovski, Adam Bodnar, Joanna McGrenere, François Guimbretière, and Tamara Munzner. An evaluation of pan & zoom and rubber sheet navigation with and without an overview. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 11–20, New York, NY, USA, 2006. ACM.

[Nor95] Stephen C. North. Incremental layout in dynadag. In *Symposium on Graph Drawing GD'95*, pages 409–418. Springer-Verlag, 1995.

[PF93] Ken Perlin and David Fox. Pad: An alternative approach to the computer interface. In *Proceedings of Computer Graphics (SIGGRAPH 93)*, pages 57–64, New York, NY, USA, 1993. ACM Press.

[RM93] George G. Robertson and Jock D. Mackinlay. The document lens. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 101–108, 1993.

[SG07] Garth Shoemaker and Carl Gutwin. Supporting multi-point interaction in visual workspaces. In *Proceedings of the ACM 2007 Conference on Human Factors in Computing Systems*, pages 999–1008, 2007.

[Shn83] Ben Shneiderman. Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57–69, 1983.

[SSTR93] Manojit Sarkar, Scott S. Snibbe, Oren J. Tversky, and Steven P. Reiss. Stretching the rubber sheet: A metaphor for visualizing large layouts on small screens. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 81–91, 1993.

[vWN03] Jarke J. van Wijk and Wim A. A. Nuij. Smooth and efficient zooming and panning. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 15–22, 2003.

[WCG03] Nelson Wong, M. Sheelagh T. Carpendale, and Saul Greenberg. EdgeLens: An interactive method for managing edge congestion in graphs. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 51–58, 2003.

[Woo84a] David D. Woods. Visual momentum: a concept to improve the cognitive coupling of person and computer. *International Journal of Man-Machine Studies*, 21:229–244, 1984.

[ZMG$^+$03a] Polle T. Zellweger, Jock D. Mackinlay, Lance Good, Mark Stefik, and Patrick Baudisch. City lights: contextual views in minimal space. In *Extended Abstracts of the ACM CHI 2003 Conference on Human Factors in Computing Systems*, pages 838–839, 2003.

Social networks analysis

[Alb07] Jim Albert. *Bayesian Computation with R*. Springer, New York, 2007. ISBN 978-0-387-71384-7.

[BA99] Albert-Lásló. Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[BEF99] Stephen Borgatti, Martin Everett, and Linton C. Freeman. *UCINET V user's guide*. Analytic Technologies, Natick, MA, 1999.

[BLGS06] Mustafa Bilgic, Louis Licamele, Lise Getoor, and Ben Shneiderman. D-Dupe: An interactive tool for entity resolution in social networks. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 43–50, New York, NY, USA, 2006. IEEE Press.

[Bon72] Phillip Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2:113–120, 1972.

[BP98] Sergey Brin and Lawrence Page. Integration and radiality: measuring the extent of an individual's connectedness and reachability in a network. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[Bra01] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(2):163–177, 2001.

[BW76] Scott A. Boorman and Harrison C. White. American journal of sociology. *Social structure from multiple networks. II. Roles structures.*, 81:1384–1449, 1976.

[dNMB05] Wouter de Nooy, Andrej Mrvar, and Vladimir Batagelj. *Exploratory Social Network Analysis with Pajek.* Structural Analysis in the Social Sciences. Cambridge Univ. Press, 2005.

[ER59] P. Erdos and A. Renyi. On random graphs i. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[Ex] An example of coauthorship network visualization. http://www.mpi-fg-koeln.mpg.de/ lk/netvis/Huge.html. Accessed March 2008.

[Exc] Excel. Microsoft office excel, microsoft corp.

[Fra77] Ove Frank. Journal of statistical planning and inference. *Survey sampling in graphs*, 1:235–264, 1977.

[Fra78] Ove Frank. Social networks. *Sampling and estimation in large social networks*, 1:91–101, 1978.

[Fra80] Ove Frank. International statistical review. *Sampling and inference in a population graph*, 48:33–41, 1980.

[Fra88] Ove Frank. Mathematiques, informatique et sciences humaines. *Random sampling and social networks: a survey of various approaches*, 26:19–33, 1988.

[Fre70] Linton C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1(3):215–239, 1970.

[Fre77] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, 1977.

[Fre00] Linton C. Freeman. Visualizing social networks. *Journal of social structure*, 1(1), 2000.

[GN02] Michelle Girvan and Mark E.J. Newman. Community structure in social and biological networks. *Proc Natl Acad Sci U S A*, 99(12):7821–7826, June 2002.

[HD05] Mark Huisman and Marijtje A. J. Duijn. Software for social network analysis (chapter 13). In V. Batagelj W. de Nooy, A. Mrvar, editor, *Exploratory Social Network Analysis with Pajek*, 2005.

[HFB$^+$04] Daniel B. Horn, Thomas A. Finholt, Jeremy P. Birnholtz, Dheeraj Motwani, and Swapnaa Jayaraman. Six degrees of Jonathan Grudin: a social network analysis of the evolution and impact of CSCW research. In *CSCW '04: Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 582–591, New York, NY, USA, 2004. ACM Press.

[HH95] Per Hage and Frank Harary. Eccentricity and centrality in networks. *Social Networks*, 17:57–63, 1995.

[HL81] Paul Holland and Samuel W. Leinhardt. An exponential family of probability distributions for directed graphs (with discussion). *Journal of the American Statistical Association*, 76:33–65, 1981.

[HLL83] Paul W. Holland, Kathryn B. Laskey, and Samuel Leinhardt. Social networks. *Stochastic blockmodels: some first step*, 5:109–137, 1983.

[HvD03] Mark Huisman and Marijtje A.J. van Duijn. Stocnet: Software for the statistical analysis of social networks. *Connections*, 25(1):7–26, 2003.

[INS] INSNA. International network for social network analysis. `http://www.insna.org/`. Accessed November 2007.

[JSGF$^+$06] Michal Jacovi, Vladimir Soroka, Gail Gilboa-Freedman, Sigalit Ur, Elad Shahar, and Natalia Marmasse. The chasms of CSCW: a citation graph analysis of the CSCW conference. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 289–298, New York, NY, USA, 2006. ACM Press.

[Kle99] Jon M. Kleinberg. Auhoritative sources in a hyperlinked environment. *Journal of the Association for Computing Machinery*, 46(5):604–632, 1999.

[Kre94] Hildrun Kretschmer. Coauthorship networks of invisible college and institutionalized communities. *Scientometrics*, 30:363–369, 1994.

[Lyn07] Scott M. Lynch. *Introduction to Applied Bayesian Statistics and Estimation for Social Scientists*. Springer, New York, 2007. ISBN 978-0-387-71264-2.

[Mat] Matman package for Excel. http://www.tracksys.co.uk/product-details.php?id=17. Accessed March 2008.

[Mor34] Jacob Levy Moreno. *Who Shall Survive?* Nervous and Mental Disease Publishing Company, Washington, DC, 1934.

[MP96] Goran Melin and Olle Persson. Studying research collaboration using coauthorships. *Scientometrics*, 36:363–377, 1996.

[New01b] M.E.J. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. *Phys. Rev.*, 64:016131–016132, 2001.

[New04a] Mark E. J. Newman. Coauthorship networks and patterns of scientific collaboration. In *Proceedings of the National Academy of Science*, volume USA 101, pages 5200–5305, Berlin, 2004.

[Newar] Mark E. J. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. In E. Ben-Naim, H. Frauenfelder, and Z. Toroczkai, editors, *Complex Networks*, Berlin, To appear. Springer.

[R D06] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.

[RPKL07] Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. An introduction to exponential random graph (p*) models for social networks. *Social Networks*, 29:173–191, 2007.

[Sab66] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31:581–603, 1966.

[Sea05a] Andrew J. Seary. *MultiNet: An interactive program for analysing and visualizing complex networks*. PhD thesis, Simon Fraser University, Irvine, 2005.

[Sea05b] Andrew J. Seary. Multinet manual., 2005.

[SNA] Sna package for R. http://erzuli.ss.uci.edu/R.stuff/. Accessed March 2008.

[vDSZ04] Marijtje .A.J. van Duijn, Tom .A.B. Snijders, and Bonne H. Zijlstra. p2: a random effects model with covariates for directed graphs. *Statistica Neerlandica*, 58:234–254, 2004.

[VF98] Thomas W. Valente and Robert K. Foreman. Integration and radiality: measuring the extent of an individual's connectedness and reachability in a network. *Social networks*, 20(1):89–105, 1998.

[WA87] Stanley Wasserman and Carolyn Anderson. Social networks. *Stochastic a posteriori blockmodels: construction and assessment*, 9:1–36, 1987.

[WBB76] Harrison C. White, Scott A. Boorman, and Ronald L. Breiger. American journal of sociology. *Social structure from multiple networks. I. Blockmodels of roles and positions.*, 81:730–779, 1976.

[WF94] Stanley Wasserman and Katherine Faust. *Social Network Analysis*. Cambridge Univ. Press, 1994.

[WS98] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440 – 442, 1998.

Exploring social networks with matrix-based representations

With the increasing use of Internet technologies, social scientists have more data to analyze. Online communities such as FaceBook or Flickr provide rich information on how people communicate and how their social network evolves. To analyze this data, social scientists require robust tools that can handle large and complex networks and allow a flexible analysis from overviews of the entire dataset to detailed analysis of important sections. As human brain is particularly effective at processing visual information, we propose to support the exploration of social networks by information visualization. Previous tools for network visualization are mostly based on node-link diagrams, suffering of readability issues (node overlapping or edge crossing) for either large or dense networks. In this thesis, we investigate alternative representations based on adjacency matrices.

Following participatory design principles, we involved social scientists into the design of three interactive visual systems: MatrixExplorer, MatLink and NodeTrix. MatrixExplorer combines matrices and node-link diagrams. Both representations are coordinated and a set of interactive tools allows their manipulation. MatLink is an augmented matrix, providing interactive links on its border to help performing several connectivity tasks. Finally, NodeTrix represents networks as node-link diagrams, using matrices for dense sub-parts. NodeTrix is particularly suited for small-world networks, globally sparse but locally dense. This dissertation presents the design and evaluation of these three systems including a case study analyzing 20 years of publications data in Human-Computer Interaction.

Keywords : Human-computer interface; Information visualization; Interaction; Social networks; Graphs; Node-link diagram; Adjacency Matrix

Explorer les réseaux sociaux avec des représentations matricielles de graphe

Grâce à la banalisation des outils de communication électroniques, les chercheurs en sciences sociales disposent de nombreuses données. Les communautés en ligne telles que FaceBook et Flickr par exemple, fournissent de riches informations sur la façon dont les personnes communiquent et l'évolution de leurs réseaux sociaux. Pour analyser ces données, les sociologues ont besoin d'outils robustes, pouvant manipuler de grandes quantités de données, et flexibles, permettant l'exploration à de multiples niveaux: de la vue d'ensemble à l'analyse détaillée d'une sous partie. Nous proposons de visualiser les données car le cerveau humain est très efficace pour traiter les informations visuelles. La majorité des outils de visualisation de réseaux est basée sur des diagrammes nœud-lien. Ces représentations ayant d'importants problèmes de lisibilités, nous explorons dans cette thèse des représentations alternatives basées sur les matrices d'adjacence.

Nous présentons trois systèmes interactifs conçus avec des chercheurs en sciences sociales: MatrixExplorer, MatLink et NodeTrix. MatrixExplorer combine matrices et nœuds-liens. Les deux représentations sont synchronisées et un panel d'outils interactifs permet de les manipuler. MatLink est une matrice augmentée de liens interactifs, qui aident à mieux percevoir la connectivité du réseau. Enfin NodeTrix représente les réseaux par des diagrammes nœud-lien et leurs parties denses par des matrices, ce qui se révèle particulièrement efficace pour représenter les réseaux petit-monde. Cette thèse présente la conception et l'évaluation de ces trois systèmes ainsi qu'une étude de cas analysant 20 ans de publications en Interaction Homme-Machine.

Mots clés : Interface homme-machine; Visualisation d'information; Interaction; Réseaux sociaux; Graphes; Diagramme nœud-lien; Matrice d'adjacence