Visualisation hybride des liens hiérarchiques incorporant des treemaps dans une matrice d'adjacence

Sébastien Rufiange, Michael J. McGuffin, Christopher Fuhrman

École de technologie supérieure
Département de génie logiciel et des TI
1100, Notre-Dame ouest
Montréal, Canada, H3C 1K3
sebastien@rufiange.com, { michael.mcguffin, christopher.fuhrman }@etsmtl.ca

RESUME
Les graphes composés («compound graphs») sont généralement efficaces afin de modéliser les composants des systèmes hiérarchiques. Alors que les diagrammes noeuds-liens peuvent servir à visualiser de tels graphes, les matrices d'adjacence contournent les problèmes d'occlusion des arêtes, souvent répandus dans les réseaux denses. Néanmoins, les matrices ne permettent pas de voir les hiérarchies des graphes composés. Afin de résoudre ce problème, nous proposons une technique de visualisation novatrice combinant une matrice d'adjacence et des treemaps montrant les hiérarchies de certains éléments. Un prototype a été construit pour illustrer les idées de cette visualisation hybride permettant de représenter des graphes orientés et pondérés. Des techniques d'interaction aident l'utilisateur à réorganiser les regroupements hiérarchiques des éléments, en facilitant l'exploration de liens au sein du réseau. Par exemple, en génie logiciel, le prototype peut servir à étudier les liens entre les modules afin de découvrir graduellement l'architecture d'un logiciel.

MOTS CLES : visualisation d'information, graphes composés, réseaux, interaction, agrégation.

ABSTRACT
Compound graphs are often useful for modeling components of hierarchical systems. While node-link diagrams can be used to visualize these graphs, adjacency matrices have the advantage of eliminating occlusion between edges, even in dense networks. However, matrices do not reveal the hierarchy in a compound graph. To address this issue, we propose a novel hybrid visualization technique that combines an adjacency matrix with treemaps displaying portions of the hierarchy. A prototype is presented that illustrates these ideas and supports visualization of digraphs with weighted edges. Interaction techniques allow the user to reorganize the hierarchical groupings of elements and facilitates the exploration of links within the network. For example, in software engineering, the prototype enables browsing of links between source code modules to help discover the architecture of the software.

CATEGORIES AND SUBJECT DESCRIPTORS: H.5.2. Information interfaces and presentation : User Interfaces.

GENERAL TERMS: Design.

KEYWORDS: information visualization, compound graphs, networks, interaction, aggregation.

INTRODUCTION
La visualisation des liens entre les éléments d'un réseau permet de révéler la structure de leurs interactions. Ces éléments sont souvent organisés en hiérarchie pouvant être modélisée par un graphe composé. Il peut s'avérer difficile de visualiser ces vastes réseaux d'interactions, étant donné leur complexité. La technique de visualisation proposée dans cet article aide à retrouver l'organisation hiérarchique des éléments au sein de réseaux denses grâce à l'agrégation. En général, notre approche permet de visualiser des graphes orientés contenant des arêtes pondérées. L'exploration des interactions à l'intérieur d'un système permet d'identifier des goulots d'étranglement, des relations scabreuses ou des regroupements déséquilibrés. En réingénierie des logiciels, retrouver une architecture à partir du code aide à mieux comprendre les rôles des composants d'un système, alors que la documentation n'est pas continuellement mise à jour. Les graphes composés peuvent aussi représenter les interactions entre les protéines d'un réseau biologique. Ainsi, ces protéines sont regroupées en hiérarchie selon leur emplacement physique dans les composants cellulaires. En outre, les réseaux sociaux peuvent être modélisés par des graphes composés.

Dans cet article, nous comparons diverses approches servant à visualiser les interactions internes d'un système et proposons une nouvelle technique hybride. Un prototype a

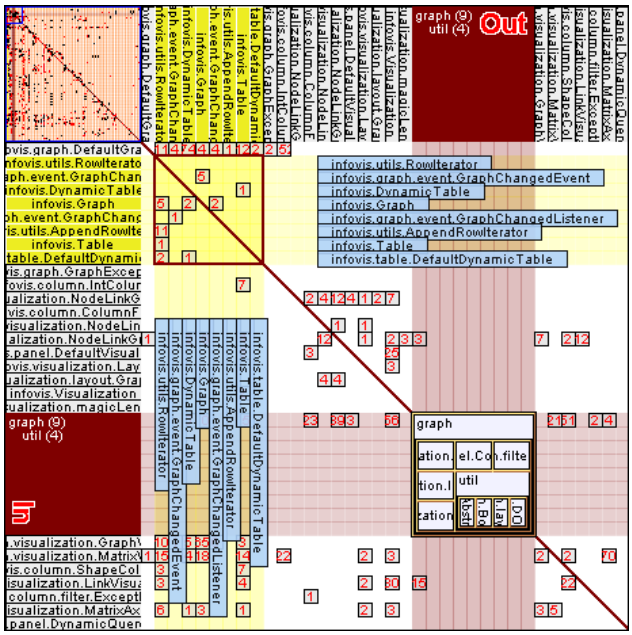

Figure 1 : Vue d'ensemble du prototype. L'utilisateur sélectionne des noeuds et leurs arêtes (en jaune) avant de les rassembler dans un nouveau groupe.

été conçu pour illustrer les idées de cette visualisation novatrice qui combine les matrices d'adjacence [4] et les treemaps [8] (Figure 1).

Afin de visualiser les liens entre les composants d'un système complexe, un diagramme noeuds-liens est un candidat naturel. Dans ce cas, chaque noeud représente un composant du système, alors que les liens forment des associations entre les noeuds. Cette approche a l'inconvénient d'être rapidement illisible lorsque le nombre de noeuds et de liens augmente. Une autre manière de représenter un graphe est d'utiliser une matrice d'adjacence, ce qui résout le problème d'occlusion des arêtes dans les réseaux denses. Toutefois, il demeure difficile de montrer la hiérarchie d'un graphe composé. En revanche, les treemaps peuvent représenter des hiérarchies en utilisant peu d'espace, mais ne montrent pas clairement les relations entre les éléments.

L'originalité de cet article est de proposer une visualisation hybride combinant les avantages des matrices et des treemaps afin de montrer les liens hiérarchiques au sein de réseaux denses. Cette nouvelle approche utilise efficacement l'espace disponible et n'est pas affectée par le problème de chevauchement des liens.

TRAVAUX ANTÉRIEURS

Il existe un nombre croissant de visualisations hybrides faisant usage de deux vues ou de deux représentations et celles-ci peuvent être utilisées pour montrer diverses informations. Certaines de ces techniques sont applicables aux arborescences [11], aux graphes non-composés [5, 6] ou aux graphes composés [3, 7, 10]. Dans ce dernier cas, les noeuds sont alors regroupés de façon hiérarchique.

Zhao et al. [11] se sont servis de treemaps pour représenter des parties d'une hiérarchie à l'intérieur d'un diagramme noeuds-liens, mais cette méthode n'est valable que pour des arborescences. Le système de Henry et Fekete [5] présente deux vues synchronisées d'un réseau social, soit un diagramme noeuds-liens et une matrice. Henry et al. [6] ont ensuite proposé une vue intégrée remplaçant certains éléments d'un diagramme noeuds-liens par des matrices, tout en conservant les associations. Chaque matrice peut alors regrouper les noeuds d'une communauté ou d'un sous-graphe dense. En revanche, il n'est pas possible d'obtenir plusieurs niveaux de regroupements à l'aide de cette technique, que l'on retrouve dans les graphes composés.

Certains travaux [3, 7] proposent de montrer les liens d'un graphe sur un treemap. Ainsi, Holten [7] a proposé de rassembler en liasses des arêtes. Ces stratégies peuvent être efficaces pour obtenir une vue d'ensemble d'un système, mais le discernement des liens devient ardu pour des graphes plus élaborés. Sangal et al. [10] ont utilisé une matrice d'adjacence pour représenter les relations entre les composants d'un logiciel. Une vue séparée montre l'organisation hiérarchique des éléments du système. En comparaison, notre visualisation hybride intègre les informations concernant les nœuds dans une vue unifiée et utilise moins d'espace, mais n'indique pas toutes les relations internes.

En résumé, la technique présentée permet de visualiser des graphes orientés comprenant des arcs pondérés et montre une vue d'ensemble d'un réseau hiérarchique. En outre, notre prototype peut montrer des réseaux denses sans être affecté par le problème d'occlusion des arêtes, présent dans [3, 7]. Enfin, l'utilisateur peut réorganiser la

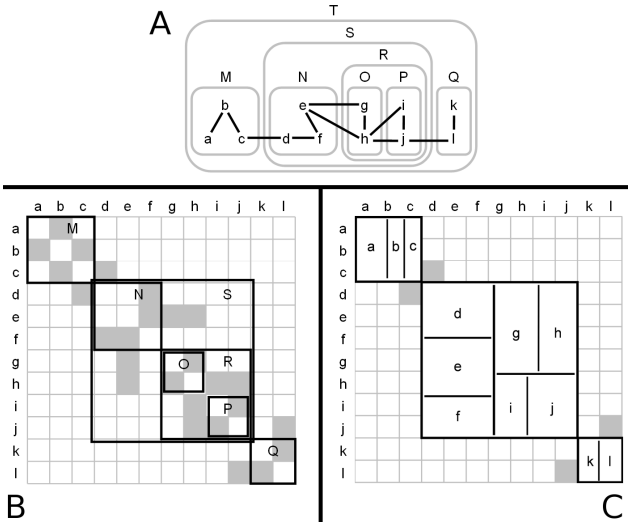

Figure 2 : Visualisation d'un graphe composé de trois manières différentes. A) Diagramme noeuds-liens. B) Matrice d'adjacence sur laquelle les regroupements sont encadrés. C) Notre technique hybride. Dans ce cas, les treemaps des groupes M, S et Q sont placés à l'intérieur de la matrice.

hiérarchie des éléments, ce qui est utile notamment pour découvrir et interpréter l'architecture d'un système.

CONCEPTION

Dans cette section, nous comparerons tout d'abord des techniques de visualisation des graphes composés. Ensuite, pour mieux illustrer les fonctionnalités supportées par le prototype, nous présenterons des scénarios d'utilisation en réingénierie des logiciels.

Visualisation de graphes composés

Des techniques pouvant être utilisées pour visualiser une hiérarchie d'éléments d'un graphe sont présentées dans la figure 2. En 2A, les arêtes et les feuilles, à savoir les noeuds en lettres minuscules, forment un graphe. Pour bien montrer l'organisation hiérarchique des éléments, des rectangles gris délimitent les noeuds regroupés, en lettres majuscules. Ce même graphe est également visualisé à l'aide d'une matrice d'adjacence à la figure 2B et l'arborescence est mise en évidence par des bordures noires. Les rangées et les colonnes de la matrice représentent les noeuds du graphe, alors que les cellules grises montrent plutôt les arêtes pondérées reliant ces noeuds. Le groupe **S** comprend les noeuds **d** à **j** et occupe une aire de 7×7=49 cases. Parmi celles-ci, seulement sept sont réservées aux noeuds à l'intérieur du regroupement **S**, soit les cellules situées le long de la diagonale. Les cases restantes servent à représenter les arêtes. Enfin, la visualisation présentée en 2C incorpore notamment le treemap de l'arborescence **S** dans la matrice. En négligeant les marges, l'aire du treemap consacre sept fois plus d'espace aux noeuds qu'à la figure 2B. Des attributs peuvent par exemple être associés aux étiquettes, couleurs et aires des noeuds **d** à **j**. Les treemaps disposés en carré (« squarified treemaps ») [1] utilisent bien l'espace [9] et s'insèrent naturellement dans une matrice. Pour ces raisons, cette approche a été retenue pour notre prototype.

Le graphe illustré à la figure 2A est non orienté ; les matrices associées (figures 2B et 2C) sont donc symétriques. Néanmoins, notre système est capable de visualiser des graphes orientés comprenant des arêtes pondérées. En supposant qu'un graphe représente l'architecture d'un logiciel, les arêtes désigneraient alors les dépendances entre les modules, pondérées selon une mesure de couplage. Une propriété intéressante de la matrice d'adjacence est sa capacité à montrer un grand nombre de liens, sans nuire à la lisibilité [4]. De plus, les régions de la matrice comportant davantage d'interactions sont plus faciles à déceler, ce qui suggère la formation de groupes. Par contre, d'autres techniques sont requises afin de faciliter la navigation et le repérage des liens à l'intérieur des réseaux denses.

Présentation du prototype

Une tâche fondamentale de notre application est d'explorer l'organisation hiérarchique des composants du système observé. Notre prototype supporte plusieurs modes qui gèrent l'activation des fonctionnalités : création de groupes, sélection de noeuds, translation panoramique et zoom. Un curseur approprié indique à l'utilisateur le mode actuel et la touche « SHIFT » permet de changer d'outil.

L'agrégation des noeuds et des arêtes permet à l'utilisateur de rassembler des éléments interdépendants, en fonction des interactions internes. La figure 1 montre le processus de création d'un groupe. Des étiquettes (en bleu) permettent d'identifier plus clairement les noeuds sélectionnés et l'utilisateur peut ainsi se concentrer sur la tâche de regroupement qui survient lorsque la touche « CTRL » est enfoncée. Il est possible de renommer un groupe en double-cliquant dessus ou de déplacer un élément à l'aide des touches directionnelles.

Bien que la création de groupes soit une technique utile pour analyser l'architecture d'un système, ces groupes communiquent également entre eux à différents niveaux hiérarchiques. Les cellules de la matrice indiquent le degré de la relation unifiant les noeuds. Dans le cas présenté, il s'agit du nombre de liens entre les composants, mais d'autres heuristiques pourraient être utilisées. Pour un groupe, cette mesure est l'ensemble des liens entre les éléments à l'intérieur de la hiérarchie et les autres noeuds.

En mode de création, l'outil de sélection respecte l'apparence carrée d'une matrice d'adjacence (Figure 1). Afin de tenir compte de l'orientation des relations, le côté gauche de la matrice renferme les noeuds entrants et la région supérieure inclut les noeuds sortants. Ainsi, l'utilisateur sélectionne à la fois les rangées et les co-

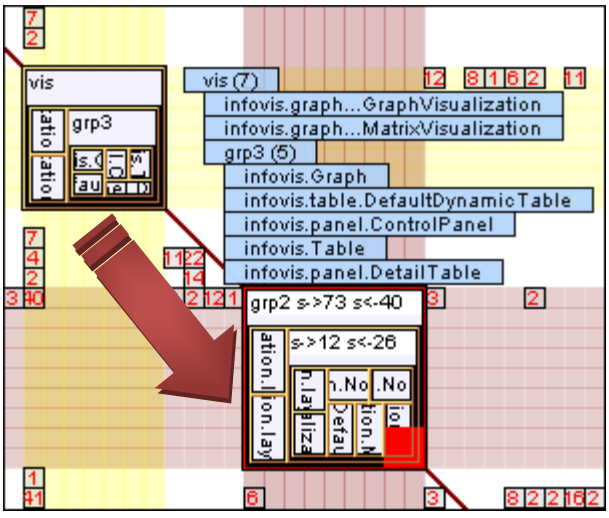

Figure 3 : L'utilisateur peut glisser-déposer un noeud dans un autre, tel que montré par la flèche, afin de réorganiser les hiérarchies. Le chiffre à droite de l'étiquette « s-> » indique le nombre de liens de l'élément déplacé vers la hiérarchie de destination. La marque « s<- » dénote le sens inverse (de la destination vers la source). Dans cet exemple, le composant « vis » utilise 73 éléments de « grp2 ».

lonnes associées aux arêtes choisies. Le mode de sélection montre, près du curseur, une vue en arborescence où les noms des noeuds sont indentés selon le niveau de profondeur de l'élément (en bleu à la figure 3). En plus de maintenir le centre d'attention de l'utilisateur, cette approche améliore la lisibilité des étiquettes. Enfin, le troisième mode offre une vue panoramique et un outil zoom qui sont ajustables par des glissements de la souris. L'analyste peut donc obtenir une vue d'ensemble du système ou se concentrer sur une région spécifique.

Au fur et à mesure que les éléments s'organisent en hiérarchies, l'analyste peut déplacer des noeuds en les glissant déposant à l'intérieur d'une autre hiérarchie. Le déplacement d'un sous-groupe sur une case vide de la matrice aura pour effet de le détacher de sa hiérarchie parente. Afin d'aider l'utilisateur à repérer la cible adéquate, des étiquettes sont affichées sur les hiérarchies visibles (Figure 3). Par exemple, si un groupe en utilise un autre plusieurs fois, mais que l'inverse n'est pas vrai, cela suggère que le premier groupe est à un niveau d'abstraction plus élevé que le second. Dans ce cas, le premier groupe serait le parent du deuxième. Le processus peut être répété afin de déduire l'architecture du système étudié.

IMPLEMENTATION

Le prototype fonctionne à l'aide de la plateforme Java et intègre la boîte à outils InfoVis [2]. Le support de la visualisation hybride et des techniques d'interaction associées est notre principale contribution. Les données présentées dans cet article ont été extraites à partir du code source et du code compilé de cette même boîte à outils, étant donné sa taille et sa complexité typiques.

CONCLUSION ET TRAVAUX FUTURS

L'objectif de cette recherche était de proposer une technique de visualisation appropriée pour les liens entre les éléments hiérarchiques d'un système. Nos travaux ont montré que les treemaps utilisent plus efficacement l'espace afin de caractériser les noeuds que d'autres visualisations. Cet article a proposé une visualisation hybride combinant des matrices et des treemaps. Un prototype a été conçu pour illustrer les techniques d'interaction supportant l'analyste dans son exploration des interdépendances d'un système. Des études devront être menées pour comparer l'approche à d'autres méthodes servant à montrer les liens hiérarchiques. Une analyse des besoins et des expériences auprès d'utilisateurs contribueront à suggérer de nouvelles techniques d'interaction et des fonctions. L'évaluation d'algorithmes d'ordonnancement s'avère une piste intéressante afin de suggérer des regroupements dans la matrice. La visualisation proposée est utile pour découvrir l'organisation d'une version donnée, mais d'autres travaux devront être effectués pour comparer des hiérarchies et visualiser leur évolution.

BIBLIOGRAPHIE

1. Bruls, M., Huizing, K., and Van Wijk, J.J. Squarified treemaps. In *Proc. of the Joint EUROGRAPHICS and IEEE TCVG Symposium on Visualization*, pages 33-42, 2000. Springer-Verlag/Wien.

2. Fekete, J.-D. The infovis toolkit. In *Proc. of the 2004 IEEE Symposium on InfoVis*, 2004.

3. Fekete, J.-D., Wang, D., Dang, N., Aris, A., and Plaisant, C. Overlaying graph links on treemaps. In *Proc. of the 2003 IEEE Symposium on InfoVis*, pages 82–83, 2003.

4. Ghoniem, M., Fekete, J.-D., and Castagliola, P. Comparaison de la lisibilité des graphes en représentation noeuds-liens et matricielle. In *Proc. of the 16th conference on Association Francophone d'Interaction Homme-Machine*, 2004. ACM.

5. Henry, N., and Fekete, J.-D. Matrixexplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, *12*(5):677-684, 2006.

6. Henry, N., Fekete, J.-D., and McGuffin, M.J. Nodetrix: A hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, *13*(6):1302-1309, 2007.

7. Holten, D. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, *12*(5):741-748, 2006.

8. Johnson, B., and Shneiderman, B. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proc. of the 2nd conference on Visualization '91*, 1991. IEEE.

9. McGuffin, M.J., and Robert, J.-M. Quantifying the space-efficiency of 2D graphical representations of trees. *Information Visualization*, 2009.

10. Sangal, N., Jordan, E., Sinha, V., and Jackson, D. Using dependency models to manage complex software architecture. In *Proc. of the 20th annual ACM SIGPLAN conference on OOPSLA*, pages 167-176, 2005. ACM.

11. Zhao, S., McGuffin, M.J., and Chignell, M.H. Elastic hierarchies: Combining treemaps and node-link diagrams. In *Proc. of the 2005 IEEE Symposium on InfoVis*, pages 57-64, 2005.