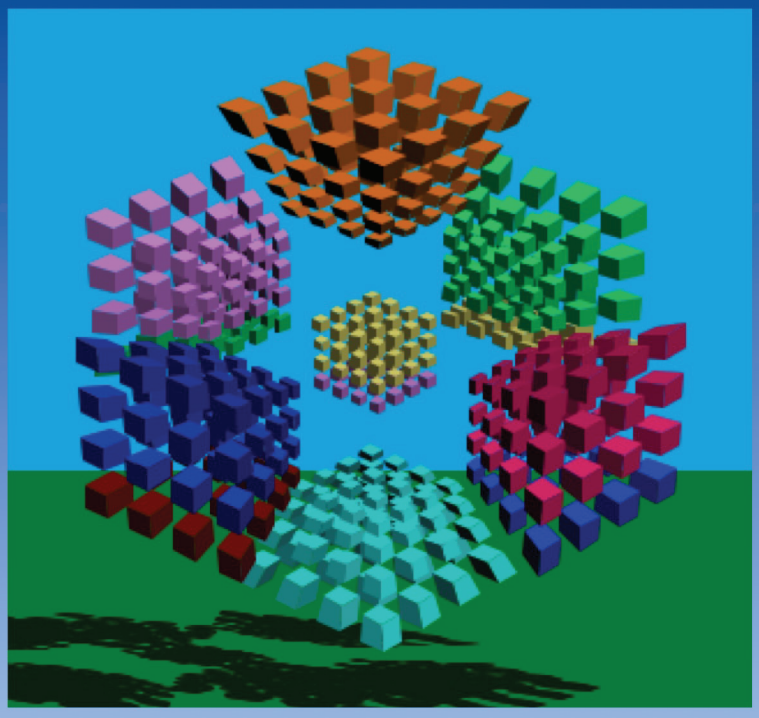
EGC'2013

13e Conférence Francophone sur l'Extraction et la Gestion de Connaissances

Université Paul Sabatier – IRIT – Toulouse

29 janvier 2013 – Journée Ateliers/Tutoriels



Visualisation d'informations, interaction et fouille de données (VIF)

Université de Toulouse

Université Paul Sabatier TOULOUSE III

IRIT CNRS - INPT - UPS - UT1 - UTM

EGC

Atelier Visualisation d'informations, interactions et fouille de données (VIF)

Organisateurs : Hanene Azzag (LIPN, Université de Paris 13)
Fatma Bouali (LI, Université François-Rabelais Tours et Université de Lille2)
Fabien Picarougne (LINA, Université de Nantes)
Bruno Pinaud (LABRI, Université de Bordeaux)

PRÉFACE

L'édition 2013 de l'atelier "*Visualisation d'informations, interaction et fouille de données*" a pour but de fournir une présentation de méthodes nouvelles, d'axes de recherche, de développements et d'applications dans les domaines de la visualisation d'informations, de la fouille visuelle de données et plus généralement des approches visuelles et interactives pour la représentation et l'extraction d'informations et de connaissances.

Cet atelier prend pour support le groupe de travail *Visualisation d'informations, interaction et fouille de données* (GT-VIF). Notre ambition est de permettre aux participants d'aborder tous les thèmes de la visualisation et concerne aussi bien les chercheurs du monde académique que ceux du secteur industriel, et autant les notions conceptuelles que les applications. Les thèmes abordés en visualisation sont très nombreux et nous nous intéresserons principalement aux questions et les problématiques liées aux grands volumes de données (les "*Big Data*"), aux aspects de méthodologie et d'évaluation, à la représentation d'informations topologiques (cartes, graphes de voisinage, etc), au "Business Intelligence", l'OLAP et autres bases de données. Nous débattrons également sur les succès et échecs récemment rencontrés.

TABLE DES MATIÈRES

Évaluation des interfaces visuelles

Guy Melançon*, Monique Noirhomme-Fraiture** et Bruno Pinaud*

*Université de Bordeaux, UMR CNRS 5800 LaBRI
351 Cours de la Libération, 33405 Talence Cedex, France
{guy.melancon, bruno.pinaud}@labri.fr,
http://www.labri.fr
**Université de Namur, Faculté d'Informatique
21 rue Grandgagnage, 5000 Namur, Belgique
mno@info.fundp.ac.be
http://www.info.fundp.ac.be

Résumé. La présente contribution est une synthèse de l'atelier VIF qui a eu lieu le 16 octobre 2012 à Bidart (France, 64), dans le cadre de la conférence Ergo IHM 2012. Le thème de l'atelier était l'évaluation des interfaces visuelles en analyse de données sous la forme d'une table ronde. Pendant l'atelier, les aspects suivants du contraignant, long et complexe processus d'évaluation ont été abordés : précisions de quelques concepts ; état des lieux des modes d'évaluation des interfaces dans les communautés IHM et analyse de données ; bonnes pratiques d'expérimentation utilisateurs ; problèmes de l'évaluation de systèmes complexes. Les organisateurs, encouragés par les résultats de cet atelier et les discussions avec les participants ont souhaité rédiger un document pour le partager avec un plus grand nombre de chercheurs.

1 Introduction

La présente contribution est une synthèse de l'atelier VIF [1] qui a eu lieu le 16 octobre 2012 à Bidart (France, 64), dans le cadre de la conférence Ergo IHM 2012. Le thème de l'atelier était l'évaluation des interfaces visuelles en analyse de données sous la forme d'une table ronde. Les organisateurs, encouragés par les résultats de cet atelier et les discussions avec les participants ont souhaité rédiger un document pour le partager avec un plus grand nombre de chercheurs.

Pendant l'atelier, les présentations et la discussion ont porté sur les aspects suivants : précisions de quelques concepts du contraignant, long et complexe processus d'évaluation ; état des lieux des modes d'évaluation des interfaces dans les communautés IHM et analyse de données ; bonnes pratiques d'expérimentation utilisateurs ; problèmes de l'évaluation de systèmes complexes. Nous suivrons ce plan pour le présent document.

Les constatations reprises dans les paragraphes 3 et 4 proviennent principalement de notre expérience comme relecteur pour des conférences et journaux, en particulier EGC, IHM, INTERACT, VisWeek (et plus généralement IEEE TVCG), EuroVis, PacificVis et Graph Drawing

1. Groupe de travail commun des associations EGC et AFIHM sur la visualisation d'informations, l'interaction et la fouille de données.

(GD). Il ne nous est donc pas possible de fournir des références précises. Néanmoins, l'évaluation devient un sujet de plus en plus présent et l'ouvrage récent d'Helen Purchase (Purchase, 2012) témoigne de cette prise de conscience de la communauté et des efforts de structuration engagés notamment depuis (Plaisant, 2004) et (Thomas et Cook, 2005).

2 Différentes méthodes d'évaluation

Nous nous concentrons ici sur des méthodes qui incluent des utilisateurs finaux, par opposition aux méthodes à base d'experts. En effet, en IHM, il est fréquent que l'on fasse appel au concours d'un ou plusieurs experts en ergonomie. Ceux-ci utilisent le logiciel en se mettant à la place d'un utilisateur et décèlent les défauts de l'interface. Leur bonne connaissance des règles et principes d'ergonomie ainsi que l'usage d'heuristiques leur permettent de mettre en évidence des défauts majeurs. Cependant, même en faisant appel à plusieurs experts, on ne peut déceler qu'une partie des problèmes rencontrés par les utilisateurs. Il a été montré qu'il n'était pas intéressant d'utiliser plus de cinq experts car au-delà de ce nombre, les nouveaux experts n'identifient plus de nouveaux défauts (Nielsen et Molich, 1990). Ajoutons qu'il est aussi possible de tester automatiquement les interfaces sur base de règles. De nombreux systèmes ont été créés pour évaluer l'accessibilité de pages Web comme par exemple DESTINE (Beirekdar et al., 2002, 2005). Comme ces systèmes utilisent le code HTML des pages, ils ne couvrent que 70% des règles. Toutes les règles de type sémantique sont alors ignorées.

Les méthodes basées sur des utilisateurs peuvent être réparties en trois classes : les études de cas, les enquêtes et les expérimentations.

Les études de cas consistent à décrire en détail l'usage de l'interface par quelques utilisateurs finaux bien choisis. Sans qu'on puisse réellement généraliser les informations obtenues à l'ensemble de la population, les études peuvent révéler des difficultés et des comportements d'utilisation.

Les enquêtes à plus grande échelle sont maintenant facilitées par l'usage d'Internet (comme c'est le cas aux USA avec le site "Mechanical Turk" d'Amazon [2], par exemple). Des participants, choisis à l'avance ou volontaires, sont invités à se connecter sur un site et à tester un logiciel en ligne. Habituellement, un scénario leur est proposé. Après le test, ils remplissent un questionnaire portant, d'une part, sur des réponses correspondant au scénario et, d'autre part, sur leurs impressions personnelles. Des questions permettant d'identifier leur profil sont également ajoutées (sexe, âge, profession, formation, ...). Cette méthode est assez rapide à mettre en place car elle ne demande pas d'observation directe. Elle permet d'avoir plus facilement un grand échantillon. Toutefois, beaucoup de biais peuvent se glisser dans l'étude étant donné le manque de contrôle (surtout par Internet). En particulier, on risque des réponses peu fiables et un biais dans le profil des répondants. Des corrections sont toutefois possibles. Lorsqu'il apparaît que le questionnaire a été rempli "au hasard", l'enquêteur a le loisir de l'écarter. Comme dans une enquête par téléphone, il peut aussi rétablir un équilibre dans le profil des participants en relançant l'enquête auprès des profils manquants. Enfin, on peut espérer qu'un grand nombre de réponses lisse les questionnaires erronés.

Étant donné qu'en IHM et en analyse des données, la plupart des études portent sur l'expérimentation, nous nous concentrerons dans la suite sur cette technique.

2. `https://www.mturk.com/mturk/welcome`

3 État des lieux en visualisation des données

Dans la suite, nous entendons par “visualisation des données” également “visual analytics” qui est une méthode de plus en plus populaire d'analyse utilisant le visuel comme support. Il ne faut pas confondre avec la visualisation en analyse de données qui est utilisée pour visualiser le résultat de l'analyse. L'IHM est alors l'interface/interactions entre homme et machine.

Dans ce domaine, on rencontre très peu d'expérimentations avec des utilisateurs et bien sûr pas d'enquête de grande envergure.

Suivant la pratique des chercheurs en analyse de données, les méthodes sont bien validées sur plusieurs jeux de données (simulées ou réelles). Les résultats sont comparés avec des méthodes connues. Mais toutes les manipulations sont effectuées par les auteurs des méthodes. Il n'y a donc aucun retour de la part des utilisateurs potentiels. Cette étape est cependant essentielle. Ce n'est pas parce qu'une méthode performante permet à son auteur de découvrir des propriétés intéressantes dans les données qu'elle convient à tout le monde. Si l'utilisateur potentiel n'est pas capable de s'en servir, la méthode ne sera jamais utilisée. C'est ce qui est arrivé à bon nombre de logiciels, abandonnés parce que leur interface n'était pas adaptée au profil des utilisateurs.

4 État des lieux en IHM

Il y a une douzaine d'années, les publications scientifiques ne relataient pas ou très peu d'expérimentations. Tout au plus, les nouvelles interfaces étaient-elles testées de manière informelle avec quelques cobayes, proches du développeur. Mais principalement depuis 2004 grâce aux travaux de C. Plaisant (Plaisant, 2004), la communauté internationale de visualisation d'informations a commencé à traiter le problème de l'évaluation en proposant de hierarchiser les différents types d'évaluation possibles en fonction des objectifs recherchés (cf. Thomas et Cook, 2005, chap. 6). L'objectif général des chercheurs de cette communauté est de permettre à des utilisateurs experts de leurs données de les manipuler par des métaphores visuelles. On comprend donc, l'intérêt de vérifier précisément que les solutions fournies répondent aux problèmes des utilisateurs. Néanmoins, il n'existe aucune méthode générique d'évaluation. Un article récent, résultat d'un travail gigantesque d'analyse de 850 publications dans le domaine de la visualisation d'informations recense 7 scénarios possibles d'évaluation (Lam et al., 2012). Chaque scénario définit un objectif précis pour l'évaluation et la méthode pour y parvenir.

De nos jours, grâce à ces travaux et sous la pression des relecteurs des conférences internationales, la situation s'est améliorée : des expérimentations sont soigneusement décrites, mais souvent avec un petit nombre de participants. Les hypothèses sont bien posées, le processus est reproductible. Néanmoins, on rencontre très souvent le terme “évaluation” mais ce dernier nous semble sur-estimé. Le protocole employé ressemble plus à “une validation” de la solution proposée. De notre point de vue, une évaluation doit faire l'objet d'un article entier alors qu'une validation est un processus plus rapide à effectuer et peut donc n'être qu'une partie d'un article. Une validation consiste bien souvent à prendre 4 à 5 cobayes qui effectuent quelques tâches simples pour montrer que la solution fonctionne. Cette méthode ne permet évidemment pas d'effectuer une analyse statistique des résultats (bien que certains auteurs s'y essaient). Plus globalement, du côté de l'analyse statistique, la situation reste malheureusement critiquable :

- dans de nombreux cas, les auteurs se contentent de publier des boîtes à moustaches, sans test et donc sans possibilité d'inférence ;
- l'échantillon, peu nombreux, est fortement biaisé car constitué, en général, de jeunes informaticiens âgés de 20 à 35 ans ;
- si un test statistique est utilisé, il s'agit généralement de l'ANOVA car ce test est à la mode dans certaines communautés. Il n'y a bien souvent pas de vérification des indispensables hypothèses de normalité, ni d'égalité des variances. L'ANOVA est parfois même utilisée pour une comparaison entre deux moyennes de données appariées alors que le test t dit de "student" (équivalent à l'ANOVA dans ce cas) est plus simple à mettre en place. Il semble que de nombreux chercheurs se contentent d'une analyse automatique type "presse bouton", sans bien comprendre la méthode utilisée ;
- Lorsque les hypothèses de normalité et d'égalité des variances ne sont pas satisfaites, les tests non paramétriques (test de Wilcoxon par exemple), applicables avec peu de données, sont le plus souvent ignorés ;
- il arrive même que le test utilisé soit erroné. Par exemple, prenons une expérimentation qui consiste à tester différentes interfaces et manipulations par les mêmes cobayes. Il faut impérativement utiliser une variante de l'ANOVA dite "within" ou "mixte" ou "inter-participants". Autre exemple, la correction de Bonferroni nécessaire lors de l'analyse répétée d'un même jeu de données est très rarement employée. Ces problèmes de méthodologie peuvent conduire à des résultats complètement erronés qui peuvent alors remettre en question les conclusions de l'expérimentation.

Notons que pour un relecteur, il n'est pas aisé d'identifier les erreurs car les articles fournissent bien souvent peu d'information : ils ne citent pas le test utilisé (et parfois même pas le seuil de décision α) et se contentent de donner la "p-value". Lors de l'utilisation d'une ANOVA, ils donnent une valeur de F sans donner les nombres de degrés de liberté (*idem* pour le t de student), *etc*. Parfois, un p anormalement petit, vu le nombre d'observations, peut laisser suspecter une erreur grossière mais le relecteur n'a aucune certitude sur ce point. D'autre part, certains relecteurs, incompétents en statistiques, n'identifient pas les erreurs ou, *a contrario*, font des remarques tout à fait incorrectes ou incohérentes.

5 Bonnes pratiques en évaluation

La conduite d'une évaluation est une tâche difficile et chronophage sachant qu'il n'est souvent pas possible (par manque de temps) de tester toutes les combinaisons possibles des différents paramètres. L'évaluation doit donc suivre un protocole précis et rigoureux afin d'éviter d'être mise en doute. Tous les choix et les nécessaires compromis effectués doivent être clairement justifiés. L'expérimentation doit aussi être juste. Elle ne doit pas favoriser une méthode par rapport à une autre sur n'importe quel aspect (bonne/mauvaise implémentation, tâche orientée vers une méthode plutôt que l'autre, ...). Il faut être conscient que la généralisation des résultats finaux ne peut de toute façon qu'être limitée aux conditions de l'expérimentation. Les différentes bonnes pratiques énoncées dans cette partie sont issues de travaux d'évaluation (fructueux ou non) menés en collaboration avec Helen Purchase de l'université de Glasgow. Elle a récemment publié un ouvrage détaillé sur ce sujet (Purchase, 2012). Les paragraphes suivants ne sont en aucun cas exhaustif (se référer à l'ouvrage de H. Purchase pour plus de détails). Nous citons ci-dessous les principaux points stratégiques pour la bonne conduite d'une

expérimentation qui permettent de prévenir les problèmes et donc limiter les biais qui auront une influence néfaste sur la qualité des données recueillies.

En amont de l'expérimentation, il est impératif d'identifier clairement les objectifs en les formulant sous la forme d'une ou plusieurs questions de recherche et comment y répondre. Ces éléments vont ensuite servir à formuler les tâches que les participants auront à résoudre et les mesures à effectuer (bien souvent le taux de mauvaise réponse et le temps de réponse). Au sujet de la difficulté des tâches, elles doivent être discriminantes entre les méthodes tout en évitant les effets planchers et plafonds : si une tâche est trop facile, on n'observe aucune différence entre les méthodes (effet plancher) et à l'inverse, si une tâche est trop difficile (effet plafond), le participant se décourage et n'aboutit pas (les réponses sont alors quasiment toutes mauvaises). Pour éviter ces phénomènes désagréables, il convient de toujours tester l'expérimentation sur un petit nombre de cobayes afin de la calibrer : difficulté (raisonnable) des tâches, rédaction du questionnaire final (validation), durée (raisonnable) de l'expérimentation, *etc*. Ces tests sont appelés des expériences pilotes. Une fois que le protocole d'expérimentation est clairement défini, il ne doit plus être modifié. Chaque participant doit subir exactement la même expérimentation sous peine de devoir supprimer toutes les données recueillies et recommencer l'expérimentation depuis le début (en trouvant de nouveaux participants).

Pendant l'expérimentation, il est impératif de s'assurer en permanence de la qualité des données recueillies. Par exemple, les participants doivent tous passer l'expérimentation dans les mêmes conditions expérimentales, l'expérimentateur doit s'assurer que le cobaye répond en toute bonne foi aux questions, la plate-forme d'évaluation doit être robuste, *etc*. L'expérimentateur doit donc rester avec le participant jusqu'à la fin de l'expérimentation (et être disponible à tout moment si besoin) pour s'assurer que le processus d'évaluation se déroule correctement. Pour conserver la pleine et entière collaboration des participants, il faut aussi prendre en compte leur inévitable fatigue. L'expérimentation doit donc comporter des temps de repos régulièrement positionnés. La fatigue peut aussi être minimisée en regroupant les tâches équivalentes pour minimiser les changements de contexte cognitif. La charge cognitive des participants doit aussi être prise en compte. Ils doivent se concentrer au maximum sur la tâche à résoudre plutôt que de perdre du temps sur le fonctionnement de l'interface d'évaluation (qui doit donc être la plus simple possible) ou la réponse aux questions (pas de texte libre notamment). De plus, si les participants sont motivés par une récompense, les résultats sont bien souvent de meilleure qualité.

Après l'expérimentation, il est souhaitable de recueillir des informations qualitative de la part des participants et leurs impressions sur l'ensemble du processus. Ces nouvelles données vont servir à pondérer les résultats récupérés et affiner leur analyse.

6 Problèmes de l'évaluation de systèmes complexes

La nature des expériences qui sont menées, et l'analyse statistique qui est ensuite effectuée sur ses résultats, exige de pouvoir "mesurer" les temps d'accomplissement des tâches et le taux de réussite (ou plus souvent le nombre d'erreurs) de l'utilisateur. Cela conduit souvent à définir des tâches de bas niveau, en terme de manipulation des données. Or, l'objectif ultime de la visualisation analytique est de produire des interfaces capables de seconder l'utilisateur dans la formulation d'hypothèses sur le problème étudié, voire dans la découverte de nouvelles connaissances (van Wijk, 2005). Cet objectif est difficilement compatible avec les exigences

des expériences contrôlées. Munzner (2009) propose un modèle explicitant la conception et la validation des systèmes ou des techniques de visualisation sur quatre couches conceptuelles imbriquées (Figure 1). Il est utile de le décrire ici afin de bien cerner la portée des expérimentations contrôlées lorsque vient le moment d'affirmer l' efficacité et l'utilisabilité d'une technique ou d'un algorithme.

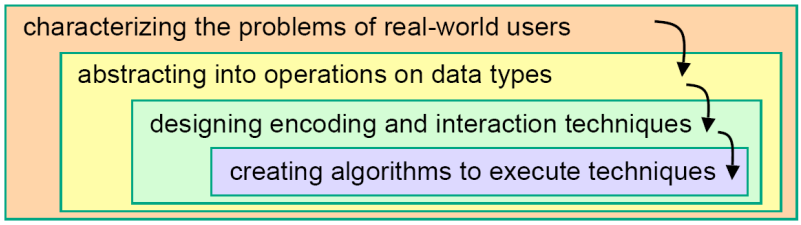


FIG. 1 – *Le modèle en couches imbriquées de Munzner (Munzner, 2009).*

La couche de plus haut niveau exige une bonne définition du domaine de référence, des questions qu'il s'agit d'étudier et des réponses attendues. Dans un contexte de visualisation de données bio-informatique, il pourrait s'agir de questions relatives à l'identification de familles de gènes comme bio-marqueurs d'une pathologie, par exemple. Dans un contexte de supervision d'un réseau informatique hautement sécurisé, on pourrait chercher à identifier des utilisateurs faisant transiter du contenu de manière illégale (en le communiquant à des tiers non autorisés), par exemple [3].

De cette couche doit pouvoir émerger la couche inférieure donnant lieu à la spécification de tâches à effectuer sur un jeu de données. Il ne s'agit pas encore ici de tâches décrites en termes de manipulation d'une interface (qui reste à spécifier et concevoir), mais bien de tâches d'analyse à conduire sur un jeu de données lui-même spécifié par les questions formulées au niveau le plus haut. Dans les exemples données au paragraphe précédent, ces tâches deviennent par exemple "Identifier les familles de gènes dont les niveaux d'expression sont très similaires dans un certains nombres de conditions expérimentales" ou "Identifier les acteurs dont l'usage du réseau paraît anormal, tant au niveau des débits, des destinations que des plages horaires où ces irrégularités se produisent".

C'est ici que le concepteur doit traduire ces spécifications en termes de manipulations au niveau de l'interface. Cela exige aussi de préciser les représentations visuelles et les indices visuels qui seront adoptés. Les tâches prennent à ce (troisième) niveau la forme d'une série de manipulation qu'il est possible de faire pour répondre aux questions posées. C'est ici que l'évaluation par expériences contrôlées intervient, puisqu'il est nécessaire de s'assurer de proposer à l'utilisateur la/les représentation/s et les interactions les plus performantes pour les tâches qu'il a à accomplir.

Munzner ajoute encore un quatrième niveau où entre en jeu la performance des algorithmes. En effet, il est primordial d'implémenter des algorithmes performants, tant en temps qu'en justesse des résultats obtenus si on veut espérer proposer une interface de visualisation efficace.

3. Cet exemple s'inspire du concours IEEE VAST 2009. Voir `http://hcil.cs.umd.edu/localphp/hcil/vast/index.php`.

Le modèle de Munzner est clair sur un point qui nous intéresse ici : la performance seule d'un algorithme n'assure en rien de l'efficacité de la technique qui l'exploite. De même, il met en garde contre toute affirmation du caractère universelle d'une technique en vertu de sa supériorité démontrée dans le cadre d'une expérience contrôlée. L'utilisabilité tient aussi au contexte, du domaine d'application, des données qui forment le support de l'exploration et/ou de l'analyse et des manipulations –des tâches– qu'il convient d'y mener. Une technique performante pour certaines tâches pourraient bien s'avérer inutile ou inefficace dans un contexte différent.

7 Défis majeurs et conclusion

L'évaluation des interfaces et interactions est un problème difficile qui n'admet pas de solution universelle. Une expérimentation doit suivre un protocole particulier afin que les résultats obtenus puissent être analysés. Pour aller plus loin, la conception des outils doit-elle aussi répondre à un protocole précis afin de bien cerner quelles parties concernent l'utilisateur et donc les points à évaluer. Nous relevons quelques défis majeurs pour améliorer les bonnes pratiques pour l'expérimentation des interfaces. Les problèmes cités ci-dessous peuvent en partie être résolu par l'organisation de séances de tutorats pour expliquer les bonnes pratiques en évaluation :

Améliorer la qualité des échantillons : taille suffisante, échantillon représentatif de la population cible (personnes âgées, tous genres confondus, jeunes, ...) ;

En visualisation de données, mieux identifier l'utilisateur : Est-ce un statisticien ou le propriétaire des données ? S'il s'agit de cette dernière catégorie, quelle formation a-t-il ou quel effort est-il prêt à faire pour utiliser le logiciel ? ;

Mieux sensibiliser les chercheurs aux évaluations : si un chercheur met au point une solution inutilisable et incompréhensible pour l'utilisateur spécialiste des données, le résultat est un temps précieux gâché pour tout le monde ;

Améliorer la formation des informaticiens en statistique : l'analyse des résultats et la présence (ou pas) de différence significative ne peut être validé que par des tests statistiques correctement effectuées pour montrer que les différences sont significatives (ou pas) ;

Pour aller plus loin, les évaluations standards de type essais-erreurs (comptage des mauvaises réponses et du temps de réponse) commencent à montrer leurs limites. La communauté travaille à la mise au point de nouvelles méthodes d'évaluation. Le colloque bi-annuel BELIV [4] est une première réponse pour faire émerger de nouvelles techniques. On peut notamment citer l'utilisation d'un occulomètre (eye-tracker). En effet, le cobaye n'exploite peut-être pas l'interface en cours d'évaluation comme l'avait pensé son concepteur. Une étude des données du regard de l'utilisateur semble donc s'avérer fort judicieuse. Néanmoins, les occulomètres délivrent un flot de données extrêmement important. Il faut donc avoir recours à des techniques d'analyse et de fouilles de données pour exploiter ces résultats.

Remerciements

Ce travail a été en partie financé par le projet ANR EVIDEN ANR 2010-JCJC-0201-01.

4. Beyond Time and Errors : Novel Evaluation Method for Visualization (BELIV), `http://www.beliv.org`

Références

Beirekdar, A., M. Keita, M. Noirhomme, F. Randolet, J. Vanderdonckt, et C. Mariage (2005). Flexible reporting for automated usability and accessibility evaluation of web sites. In *Proc. of the 2005 IFIP TC13 Int. Conf. on Human-Computer Interaction*, INTERACT'05, pp. 281–294.

Beirekdar, A., J. Vanderdonckt, et M. Noirhomme-Fraiture (2002). A framework and a language for usability automatic evaluation of web sites by static analysis of html source code. In *Proc. of the* 4^{th} *Int. Conf. on Computer-Aided Design of User Interfaces, CADUI'02*, pp. 337–348.

Lam, H., E. Bertini, P. Isenberg, C. Plaisant, et S. Carpendale (2012). Empirical studies in information visualization : Seven scenarios. *IEEE Trans. on Visualization and Computer Graphics 18*(9), 1520–1536.

Munzner, T. (2009). A nested process model for visualization design and validation. *IEEE Trans. on Visualization and Computer Graphics 15*, 921–928.

Nielsen, J. et R. Molich (1990). Heuristic evaluation of user interfaces. In *CHI'90. Proc. of the SIGCHI Conf. on Human Factors in Computing*, pp. 249–256.

Plaisant, C. (2004). The challenge of information visualization evaluation. In *Proc. of the Working Conf. on Advanced Visual Interfaces*, AVI '04, pp. 109–116. ACM.

Purchase, H. C. (2012). *Experimental Human-Computer Interaction. A practical Guide with Visual Examples*. Cambridge University Press.

Thomas, J. J. et K. A. Cook (Eds.) (2005). *Illuminating the Path : The Research and Development Agenda for Visual Analytics*. IEEE Computer Society.

van Wijk, J. J. (2005). The value of visualization. In *Proc. of IEEE Visualization*, pp. 79–86.

Summary

This contribution is a synthesis of the VIF panel discussion of the Ergo IHM 2012 conference (October 16^{th}, 2012 in Bidart, France). The panel covered the evaluation of visual interface in data analysis. The following themes of the long and difficult evaluation process have been addressed: precisions of some concepts, state of the art of the different evaluation modes in HCI and Data Analysis; good practices for user evaluations; complex systems evaluations. The organizers wrote this document to share the interesting results of the panel with a broader range of researchers.

Proposition de kernel semi-supervisé, et application au clustering visuel interactif

Pierrick Bruneau*, Benoît Otjacques*

*CRP Gabriel Lippmann - Département Informatique
41 rue du Brill, L-4422 Belvaux (Luxembourg)
bruneau,otjacque@lippmann.lu,
http://www.lippmann.lu

Résumé. Cet article décrit une nouvelle procédure de transformation de fonction noyau (i.e. *kernel*). La procédure vise à incorporer la supervision d'un utilisateur directement dans les valeurs de similarité entre objets. En utilisant ces similarités modifiées, l'ensemble d'objets est projeté en 2D grâce à une ACP à noyaux (i.e. *kernel PCA*). Un compromis est ainsi établi entre les données originales et l'expertise d'un utilisateur, tout en offrant un moyen naturel de visualisation et d'interaction. Ces projections semi-supervisées sont évaluées sur des données réelles et synthétiques, dans un contexte simulant une tâche de clustering visuel interactif. L'action d'un utilisateur est reproduite en sélectionnant aléatoirement un sous-ensemble d'objets étiquetés a priori. Les résultats expérimentaux démontrent l'efficacité de la méthode, un seul élément étiqueté pour chaque classe réelle suffisant à introduire des effets manifestes sur la visualisation.

1 Introduction

Le clustering est une tâche cruciale dans le contexte d'une analyse de données visuelle, e.g. en simplifiant la visualisation de jeux de données volumineux (Keim et al., 2008). Le cluster est un objet très parlant visuellement (Ware, 2004), et par conséquent un candidat naturel en tant que point d'entrée d'une analyse de données visuelle. Toutefois, un ensemble de clusters doit au préalable être projeté dans un espace à faible dimension (préférentiellement 2D) pour devenir accessible visuellement. La définition d'un système de clustering visuel n'est donc pas triviale, car les données réelles sont souvent associées à une haute dimensionalité. Dans cet article, nous proposons une nouvelle procédure de construction de kernel, combinant les similarités originales entre éléments avec des étiquettes de classes spécifiées a priori. La projection 2D de ce kernel modifié par une transformation kernel PCA permet alors de combiner de manière consistante la topologie intrinsèque des données avec des contraintes spécifiées par un utilisateur. En traitant ces données projetées, un algorithme de clustering peut ainsi prendre en compte le compromis de manière implicite.

En pratique, les données sans étiquette sont souvent les plus abondantes : l'étiquetage reflète souvent une vérité terrain établie manuellement (e.g. par un expert du domaine), donc

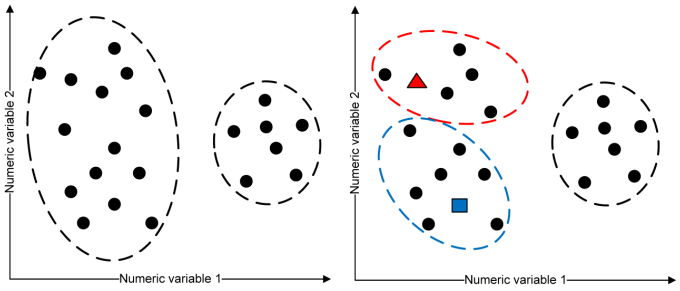


FIG. 1 – À gauche *: résultat potentiel d'un algorithme de clustering.*
À droite *: une autre solution semble privilégiée avec l'ajout de deux exemples étiquetés.*

coûteuse. Dans ce contexte, une tâche d'apprentissage semi-supervisé peut être comprise de deux manières, non-exclusives mais conceptuellement différentes :

– comme une tâche d'apprentissage supervisé (i.e. classification) avec un ensemble d'apprentissage de taille très réduite. Cette configuration interdit habituellement l'utilisation de la plupart des algorithmes d'apprentissage supervisé. Cependant, certains auteurs ont proposé d'exploiter la densité des données sans étiquette (i.e. disponibles en grande quantité) pour dépasser cette limitation (Chapelle et al., 2003).
– comme une tâche d'apprentissage non-supervisé aidé par quelques exemples étiquetés, en vue d'incorporer de la connaissance experte (voir figure 1). Cette connaissance peut ne pas être en accord avec la direction imposée par le critère de la procédure d'apprentissage ; le but de la méthode semi-supervisée est alors de traiter ce conflit de manière consistante. Cela peut se faire en utilisant les exemples étiquetés pour l'initialisation du modèle de clustering, et, de manière complémentaire, en forçant leur classe d'appartenance selon cette initialisation tout au long du processus (Basu et al., 2002). Dans le contexte des modèles probabilistes, certains auteurs ont transformé un ensemble d'étiquettes en contraintes probabilistes (i.e. *must-link* et *must-not-link*), et ont proposé un algorithme maximisant la vraisemblance de ce modèle (Law et al., 2005).

Les approches de clustering semi-supervisé existantes souffrent des limites suivantes :

– tous les travaux mentionnés ci-dessus se basent sur des transformations linéaires, et des clusters à forme gaussienne, ce qui est parfois trop restrictif dans des situations réelles (e.g. données suivant des variétés non-linéaires, non-gaussiennes),
– certains travaux ont essayé de relâcher l'hypothèse sur la forme des clusters, en autorisant l'association de plusieurs composantes gaussiennes à chaque cluster (Miller et Uyar, 1996). Mais le réglage de l'algorithme résultant s'avère délicat, et semble dépendant du domaine des données.

Notre travail n'entend pas forcer le respect de contraintes de manière explicite, comme cela est fait dans la littérature commentée ci-dessus. Au lieu de cela, nous cherchons plutôt à injecter un compromis dans une projection 2D, qui prenne en compte les similarités originales et un ensemble d'étiquettes spécifiées par un expert. En d'autres termes, notre projection 2D suit la topologie originale des données autant que le permet une information fournie a priori. N'importe quel algorithme de clustering, comme k-means ou EM pour le mélange de

gaussiennes (Bishop, 2006) peut ensuite opérer sur ces données numériques continues à faible dimension.

La visualisation de données à grande dimension en utilisant des projections 2D, et les artéfacts de distorsion qui en résultent généralement, sont un sujet d'étude à part entière dans la littérature (Aupetit, 2007), encore actif. Notre contribution peut être vue comme un complément à ce domaine : d'après la terminologie définie dans (Aupetit, 2007), la technique proposée ici pourrait être qualifiée de projection continue non-linéaire.

Dans la section 2, dans un souci d'exhaustivité nous présentons brièvement la transformation kernel PCA, en soulignant l'utilisation de cette technique pour le calcul de projections 2D. Ensuite, dans la section 3, nous proposons une nouvelle procédure de transformation de kernel. Elle permet d'obtenir un compromis entre similarités originales, et étiquetage a priori.

Ce kernel peut s'inscrire dans la tâche de clustering visuel suivante :

1. réaliser une projection 2D avec la transformation kernel PCA, en utilisant notre kernel modifié,
2. effectuer le clustering de ces données projetées.

Dans ce contexte, la semi-supervision serait construite à partir d'interactions avec un utilisateur (e.g. en cliquant et étiquetant des éléments directement sur la visualisation 2D). Dans cet article, l'aspect purement interactif est volontairement laissé de côté, pour se concentrer sur une évaluation expérimentale la plus objective possible du comportement de notre kernel. Nous avons ainsi choisi de le confronter à des sous-ensembles d'éléments étiquetés, sélectionnés aléatoirement dans des jeux de données ayant une vérité terrain connue. Par ce choix, nous entendons identifier les propriétés intrinsèques de notre kernel, et le contraster avec une approche existante. Un kernel classique, non-supervisé, sert de groupe de contrôle pour cette comparaison. Après une discussion critique de nos résultats expérimentaux, nous concluons avec quelques perspectives ouvertes par ce travail dans le domaine de la fouille de données visuelle.

2 Projection 2D par transformation kernel PCA

Considérons un ensemble d'éléments $\mathbf{X} = \{\mathbf{x}_i\}_{i \in 1 \dots N}$, prenant ses valeurs dans un domaine $\mathcal{X}$ (appelé *espace original* ci-après), et une transformation non-linéaire ϕ projetant un élément $\mathbf{x}_i \in \mathcal{X}$ sur un point $\phi(\mathbf{x}_i) \in \mathbb{R}^M$ (appelé *espace transformé* par la suite). Sous l'hypothèse $\sum_{i=1}^{N} \phi(\mathbf{x}_i) = \mathbf{0}$, la matrice de covariance empirique de l'image de $\mathbf{X}$ dans l'espace transformé est donnée par :

$$\mathbf{C} = \frac{1}{N} \sum_{i=1}^{N} \phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^T,$$

les vecteurs propres associés à cette matrice vérifiant alors :

$$\mathbf{C}\mathbf{v}_m = \lambda_m \mathbf{v}_m, \; m = 1 \dots M.$$

Suivant en cela les travaux de (Schölkopf et al., 1998) et (Bishop, 2006), ce calcul peut être transformé en :

$$\mathbf{K}\mathbf{a}_m = \lambda_m N \mathbf{a}_m,\ m = 1 \dots M, \tag{1}$$

avec $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$ un kernel, $\mathbf{K}$ la matrice $N \times N$ telle que $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ (nommée *matrice de kernel* ci-après), et $\mathbf{a}_m$ un vecteur dans $\mathbb{R}^N$. Après avoir résolu (1), i.e. trouvé ses vecteurs et valeurs propres, un ensemble de M fonctions de projection peut être défini comme suit :

$$y_m(\mathbf{x}) = \sum_{i=1}^{N} a_{mi} k(\mathbf{x}, \mathbf{x}_i). \tag{2}$$

En supposant les valeurs propres ordonnées de manière décroissante, la projection 2D qui capture le maximum de variance dans l'espace transformé est alors construite avec y_1 et y_2. L'hypothèse $\sum_{i=1}^{N} \phi(\mathbf{x}_i) = \mathbf{0}$ peut être relâchée en utilisant l'expression modifiée suivante en lieu de matrice du kernel (Bishop, 2006) :

$$\tilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_N \mathbf{K} - \mathbf{K}\mathbf{1}_N + \mathbf{1}_N \mathbf{K} \mathbf{1}_N,$$

avec $\mathbf{1}_N$ la matrice $N \times N$ dont toutes les cellules égalent $\frac{1}{N}$. L'application ϕ n'a en général pas à être explicitement définie : en effet, toute matrice semi-définie positive $\mathbf{K}$ a été démontrée comme résultant de produits scalaires dans un espace transformé, possiblement à dimension infinie (Bishop, 2006). Ainsi, en pratique des fonctions kernel sont définies directement, en s'assurant simplement que les matrices de kernel induites sont bien semi-définies positives.

Le kernel gaussien vérifie cette propriété, et est défini comme suit :

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{\sigma^2}\right).$$

Remarquons que cette expression requiert un calcul de norme euclidienne, allouant implicitement $\mathbb{R}^d$ à $\mathcal{X}$. Ce kernel a été largement utilisé dans la littérature ; toutefois, expérimentalement nous l'avons trouvé inadapté pour le traitement de données à grande dimension ($d > 100$). Ce problème a déjà été identifié dans la littérature (François et al., 2005). En tant qu'alternative, les auteurs proposent la fonction p-gaussienne :

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{d_{L2}(\mathbf{x}, \mathbf{x}')^p}{\sigma^p}\right), \tag{3}$$

avec $d_{L2}(.,.)$ la distance euclidienne. Les paramètres p et σ sont estimés par des formules empiriques, calibrées de sorte que la distribution des valeurs de kernel s'accorde avec celle des distances dans l'espace original, indépendamment de sa dimension :

$$p = \frac{\ln(\frac{\ln 0.05}{\ln 0.95})}{\ln \frac{d_{L2}^{5\%}}{d_{L2}^{95\%}}}, \quad \sigma = \frac{d_{L2}^{95\%}}{(-\ln 0.05)^{\frac{1}{p}}} = \frac{d_{L2}^{5\%}}{(-\ln 0.95)^{\frac{1}{p}}}, \tag{4}$$

avec $d_{L2}^{5\%}$ (respectivement $d_{L2}^{95\%}$) le quantile à 5% (respectivement 95%) de la distribution cumulée de d_{L2} [1]. Dans le reste de cet article, le kernel (3) sera utilisée en tant que base non-supervisée.

1. Dans le papier référencé, $d_{L2}^{5\%}$ et $d_{L2}^{95\%}$ ont été échangés par erreur dans les expressions de σ. Une version corrigée est apportée ici.

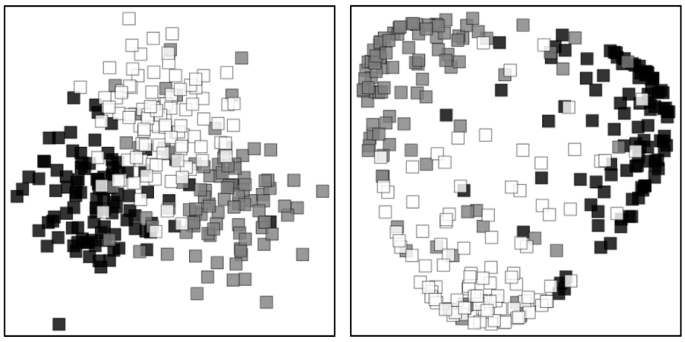


FIG. 2 – À gauche *: Jeu de données 2D original. La composante gaussienne à l'origine de chaque élément est identifiée par une teinte de gris caractéristique.*
À droite *: projection 2D par transformation kernel PCA de ce jeu de données, utilisant la fonction p-gaussienne.*

Sur la figure 2, un échantillon de données généré par 3 gaussiennes 2D se chevauchant partiellement est illustré, dans son espace original, et selon sa projection utilisant les équations (4), (3), and (2). Dans cet exemple, les données semblent "gonflées" par la transformation : la distribution des distances reste semblable après transformation, mais la topologie intrinsèque (i.e. clusters gaussiens) est maintenant exacerbée.

3 Proposition de kernel semi-supervisé

Dans cette section, les valeurs retournées par le kernel sont supposées appartenir à $[0, 1]$. Cette hypothèse est assez conventionnelle (François et al., 2005), et est respectée par la fonction p-gaussienne. Une tâche de clustering revient en partie à affecter des étiquettes (inconnues a priori) à une collection d'éléments. Le but recherché est alors d'obtenir un étiquetage le plus proche possible d'une vérité terrain. Formellement, pour un échantillon de données $\mathbf{X}$ tel que défini dans la section précédente, nous introduisons une fonction d'étiquetage associant chaque élément à une classe parmi R :

$$\begin{aligned} l \ : \mathbf{X} &\rightarrow \{1, \ldots, R\} \\ \mathbf{x} &\rightarrow l(\mathbf{x}). \end{aligned}$$

Dans cet article nous supposons un contexte semi-supervisé, i.e. avec un étiquetage potentiellement incomplet : seul un ensemble $\mathbf{X}_L \in \mathbf{X}$ est associé à une étiquette. l ne sera donc utilisée le plus souvent qu'au travers de sa restriction $l' = l|_{\mathbf{X}_L}$. Notons que l' peut définir n'importe quel niveau de supervision, d'une totale absence d'étiquetage (i.e. $\mathbf{X}_L = \emptyset$), à un contexte complètement supervisé (i.e. $\mathbf{X}_L = \mathbf{X}$), en passant par toutes les situations intermédiaires. Notre intuition est de transformer un kernel selon les plus proches voisins étiquetés de

ses arguments respectifs. La fonction suivante implémente en partie cette intuition, en retournant le plus proche élément étiqueté de n'importe quel élément dans $\mathbf{X}$:

$$\begin{aligned} s \ &: \mathbf{X} \to \mathbf{X}_L \\ &\mathbf{x} \to s(\mathbf{x}) = \begin{cases} \emptyset \quad \text{if } \mathbf{X}_L = \emptyset \\ \arg\max_{\mathbf{x}' \in \mathbf{X}_L} k(\mathbf{x}, \mathbf{x}') \text{ sinon.} \end{cases} \end{aligned}$$

l et s sont utilisées pour transformer k comme suit :

$$k'(\mathbf{x}, \mathbf{x}') = \begin{cases} k(\mathbf{x}, \mathbf{x}') & \text{si } |\text{Im}(l')| \leq 1 \\ k(\mathbf{x}, \mathbf{x}')^{\frac{1}{\alpha}} & \text{si } |\text{Im}(l')| > 1 \wedge l'\big(s(\mathbf{x})\big) \neq l'\big(s(\mathbf{x}')\big) \\ k(\mathbf{x}, \mathbf{x}')^{\alpha} & \text{si } |\text{Im}(l')| > 1 \wedge l'\big(s(\mathbf{x})\big) = l'\big(s(\mathbf{x}')\big), \end{cases} \tag{5}$$

avec $\alpha \in \mathbb{N}^*$, et Im(l') l'image de l' [2]. Intuitivement, avec k retournant des valeurs dans $[0, 1]$ comme requis, et $\alpha > 1$, transformer k en k' revient à augmenter (respectivement diminuer) la similarité entre éléments ayant la même image (respectivement une image différente) par $l' \circ s$, tout en restant dans l'intervalle voulu. Les illustrations et expériences de cet article utilisent la fonction p-gaussienne (voir l'équation (3)), mais remarquons que l'expression (5) pourrait être appliquée à n'importe quel kernel semi-défini positif prenant ses valeurs dans $[0, 1]$, ceci sans perte de généralité. L'inspection stricte des règles de construction de fonctions kernel valides (voir e.g. (Bishop, 2006)) semble indiquer que la fonction définie par l'expression (5) (voire même la fonction p-gaussienne) n'est pas un kernel valide (i.e. n'implique pas nécessairement des matrices de kernel semi-définies positives). Toutefois, des fonctions kernel invalides ont déjà été utilisées avec succès dans la littérature (Vapnik, 1995). De plus, dans ce travail nous n'utilisons que les deux premières dimensions de l'espace propre (i.e. la projection 2D), les valeurs propres desquelles ne seraient pas toutes deux réelles et significativement positives que pour des données extrêmement dégénérées.

4 Protocole expérimental

4.1 Description de la tâche

La procédure de transformation de kernel proposée dans la section précédente est incluse dans la tâche de clustering visuel interactif suivante :

1. une projection 2D initiale (équation (2)) est calculée avec la matrice de kernel p-gaussienne (équation (3)),
2. des étiquettes (i.e. valeurs de classe) sont associées à tous les éléments par un algorithme de clustering,
3. l'utilisateur met à jour ces étiquettes, ainsi que la sémantique associée, selon ses préférences,
4. les étiquettes modifiées par l'utilisateur sont utilisées pour transformer la matrice de kernel initiale (équation (5)),

2. La modification du kernel n'est sensée que si l'image de l' contient plus d'une valeur d'étiquette : la condition $\mathbf{X}_L = \emptyset$ n'est donc pas assez forte.

5. cette nouvelle matrice de kernel est utilisée pour mettre à jour la projection 2D via une transformation kernel PCA,
6. retour à l'étape 2, à moins que l'utilisateur ne soit satisfait de la projection et du clustering courants.

Dans cet article, nous laissons l'aspect purement interactif de côté, pour nous concentrer sur une évaluation approfondie du comportement de notre kernel transformé, en le confrontant à des sous-ensembles aléatoires d'éléments étiquetés a priori. Pour une meilleure estimation des effets de notre transformation de kernel, nous la contrastons avec les deux alternatives suivantes :

– la matrice de kernel issue de la fonction p-gaussienne, sans supervision (qui jouera le rôle de groupe de contrôle),
– une méthode de clustering semi-supervisé existante (Basu et al., 2002) transposée sur un kernel. En quelques mots, cette approche revient originellement à contraindre l'appartenance des éléments étiquetés par l'utilisateur, en biaisant ensuite l'algorithme de clustering avec ces affectations statiques. Dans les termes du présent article, nous implémentons ce principe en utilisant l'équation (5) sans la fonction de voisinage, i.e. la valeur de kernel $k(\mathbf{x}, \mathbf{x}')$ est transformée ssi $\mathbf{x}$ et $\mathbf{x}'$ sont tous deux dans $\mathbf{X}_L$.

4.2 Mesure de la qualité des résultats

La performance de ces méthodes est évaluée avec les mesures suivantes :
– le nombre de classes inféré par l'algorithme de clustering (**nclass** dans la table 1)
– la pureté des clusters (**purity** dans la table 1).

Ainsi qu'évoqué en introduction, le présent travail est à relater avec la littérature traitant de visualisation et de projections 2D. Dans ce contexte, nous mesurons également les distorsions engendrées par les projections, afin de dénoter des compressions et étirements (**compress** et **stretch** dans la table 1) relatifs aux distances dans l'espace original. Ces mesures de distorsion sont normalisées dans l'intervalle $[0, 1]$, 1 indiquant la distorsion maximale. Des détails au sujet du calcul de ces mesures peuvent être trouvés dans (Aupetit, 2007). En rapport à cette référence, remarquons que contrairement à ce qui y est préconisé en cas de données à grande dimension, nous n'avons pas employé de mesures basées sur le rang : nous avons en effet déjà traité ce problème en employant un kernel y étant peu sensible.

4.3 Jeux de données choisis, et utilisation

Un jeu de données synthétique et deux jeux de données réels issus du dépôt UCI ont été utilisés pour nos expériences. Celles-ci ont été implémentées avec R.

– **Gaussian** : 3000 points générés selon 3 gaussiennes 2D se recouvrant partiellement. 1000 éléments sont échantillonnés selon chaque composante. Un sous-échantillon de ce jeu de données a déjà été illustré sur la figure 2.
– **Pima** : ce jeu de données a été établi à partir d'enregistrements médicaux de patients venant de la tribu Indienne Pima. Il est défini sur 8 variables numériques, et une variable de classe binaire (i.e. présence ou absence du diabète). Il contient 500 exemples négatifs, et 268 exemples positifs.

– **Isolet** : ce jeu de données a été créé à partir d'enregistrements audio de personnes prononçant des lettres isolées. Chaque enregistrement est décrit par 617 variables numériques. Nous avons extrait les enregistrements de voyelles : cela revient donc à considérer 5 classes, avec 300 exemples dans chacune d'entre elles.

Chaque expérience consiste tout d'abord à tirer un sous-échantillon, sans remplacement, dans un de ces jeux de données. 100 éléments sont pris dans chaque classe (exception faite des exemples négatifs de *Pima*, parmi lesquels nous tirons 200 éléments, ceci afin de reproduire au mieux l'équilibre du jeu de données original). La vérité terrain est ignorée pour tous les exemples du sous-échantillon, sauf pour un nombre donné n_{lab} d'entre eux pour chaque classe, ceci afin de simuler l'interaction avec l'utilisateur. Une expérience est paramétrée par α (voir équation (5)), et n_{lab}. Nous autorisons $\alpha \in \{2, 3, 5, 10\}$, et $n_{\text{lab}} \in \{1, 2, 5, 10\}$. Notons que le pourcentage de supervision associé prend ses valeurs dans $[1\%, 10\%]$.

Une expérience est aussi paramétrée par une transformation de kernel, parmi :

– **unsupervised** : la fonction p-gaussienne, sans supervision,
– **simple** : l'approche semi-supervisée de référence (Basu et al., 2002),
– **neighbors** : notre approche de kernel semi-supervisé sensible au voisinage étiqueté (équation (5)).

Les mesures de compression et d'étirement sont calculées pour chaque expérience. Afin de ne produire qu'une mesure pour chaque expérience, nous conservons la médiane des mesures de compression (respectivement étirement) spécifiques à chaque expérience. Les données projetées sont ensuite fournies à un algorithme de clustering, de manière non-supervisée. Nous utilisons l'algorithme EM bayésien implémenté dans le package VBmix (Bruneau, 2012) pour obtenir un mélange de gausiennes représentant notre clustering. Le nombre de composantes trouvé a posteriori sert d'estimateur pour notre mesure de qualité basée sur le nombre de classes. Le mélange de gaussiennes est utilisé pour inférer l'étiquette de chaque élément, et leur comparaison à la vérité terrain sert à estimer la pureté des clusters obtenus. In fine, une condition expérimentale est caractérisée par un tuple (jeu de données, transformation, α, n_{lab}). Pour chaque condition, nous réalisons 20 expériences. L'algorithme de clustering utilisé est connu pour souffrir de problèmes de minima locaux. Pour contourner ce problème, 10 exécutions en sont réalisées, et un critère pseudo-BIC permet de sélectionner le meilleur parmi ce pool.

5 Résultats et discussion

Un test ANOVA *three-way independent* est effectué sur les résultats de nos expériences. Les trois variables indépendantes identifiées sont ordonnées comme suit : *transformation*, α, et n_{lab}. Pour *transformation*, nous définissons le contraste de *contrôle* entre *unsupervised* et les méthodes semi-supervisées, ainsi que le contraste *expérimental* entre *simple* et *neighbors*. Un contraste polynomial est appliqué à α and n_{lab}. Beaucoup de conditions expérimentales sont associées à des distributions de mesures pour lesquelles l'hypothèse gaussienne est inacceptable, ou provoquent l'échec du test de Levene pour l'homogénéité de la variance. Toutefois, nous avons choisi d'effectuer le même nombre d'expériences (i.e. 20) dans toutes les conditions expérimentales, ce qui assure la robustesse du test ANOVA (Donaldson, 1968).

Le test a été exécuté indépendamment pour chaque jeu de données et mesure de qualité : les résultats en sont résumés dans la table 1. Les conclusions suivantes peuvent en être tirées :

compress	– Le contraste expérimental est très significatif ($p < 10^{-10}$, avec $p < 0.01$ seulement pour *Isolet*). – α induit très significativement une tendance linéaire ($p < 10^{-10}$). – n_{lab} induit plus faiblement une tendance linéaire ($p \simeq 10^{-3}$). – Ces tendances sur α et n_{lab} interagissent presque exclusivement avec le contraste expérimental.
stretch	– Le contraste expérimental est très significatif ($p < 10^{-10}$). – α induit très significativement une tendance linéaire ($p < 10^{-10}$). – n_{lab} induit plus faiblement une tendance linéaire ($p \simeq 10^{-3}$), plus fortement avec *Gaussian* ($p < 10^{-10}$). – Ces tendances sur α et n_{lab} interagissent presque exclusivement avec le contraste expérimental.
purity	– Les deux contrastes sur la transformation sont significatifs ($p \simeq 10^{-2}$, seulement le contraste expérimental pour *Isolet*). – α induit modérément une tendance linéaire ($p < 10^{-3}$). – n_{lab} induit significativement une tendance linéaire ($p < 10^{-5}$), plus faiblement pour *Pima* ($p < 0.1$). – Selon le jeu de données, il peut y avoir une interaction entre une tendance linéaire sur α et le contraste de contrôle ($p < 0.1$ pour tous les jeux de données), ou le contraste expérimental ($p < 10^{-5}$ pour *Gaussian* et *Isolet*). – Excepté avec *Pima* (interaction faible, $p < 0.1$), une forte interaction entre le contraste expérimental et une tendance linéaire selon n_{lab} a été relevée ($p < 10^{-6}$).
nclass	– Le contraste expérimental est très significatif ($p < 10^{-10}$). Le contraste de controle est plus faiblement significatif, et seulement avec *Gaussian* et *Isolet* ($p < 0.1$). – α induit très significativement une tendance linéaire ($p < 10^{-10}$), plus modérément pour *Isolet*. – n_{lab} n'a pas d'influence si considéré isolément. – Une interaction significative entre le contraste expérimental et une tendance linéaire selon α a été relevée ($p < 10^{-3}$).

TAB. 1 – *Résultats des tests ANOVA, agrégés par mesure de qualité.*

- L'influence de la transformation *simple* sur la topologie des données projetées est généralement insignifiante.
- L'analyse des mesures de distorsion montre que la proposition de kernel semi-supervisé entraîne des modifications drastiques de la projection 2D obtenue. Cette influence est très perceptible avec ne serait-ce qu'un seul élément étiqueté par classe, et un supplément d'effet modéré pour de plus grands échantillons étiquetés (voir la figure 3). Cette propriété est plutôt conforme à ce qu'un utilisateur est susceptible d'attendre, en rendant ses actions rapidement tangibles.
- Les distorsions sont très fortement influencées par la variation de α. Même pour une valeur faible, les artéfacts de projection inhérents à la fonction p-gaussienne sont soit allégés (i.e. dans le cas de l'étirement), soit accentués (i.e. dans le cas de la compression). Cette tendance suit fortement une tendance linéaire, ce qui souligne davantage le rôle essentiel de α en tant que paramètre ajustable.
- Une augmentation de α tend à diminuer le nombre de clusters inférés, avec parfois des conséquences négatives en termes de pureté de clusters. Remarquons tout de même que de manière mécanique, une meilleure pureté est plus facile à obtenir en utilisant un plus grand nombre de clusters. De manière plus générale, une amélioration de la pureté devrait raisonnablement être attendue avec une méthode semi-supervisée : toutefois, la sélection aléatoire des éléments étiquetés de nos expériences a considérablement dégradé les performances de notre méthode de ce point de vue.
- Utiliser davantage d'éléments étiquetés a une influence visible seulement pour la transformation *neighbors*. Cette influence est plutôt négative sur la pureté des clusters pour des valeurs faibles de n_{lab} : ce handicap est toutefois rattrapé et dépassé rapidement avec l'augmentation du nombre d'éléments étiquetés. Cela confirme accessoirement l'influence tangible qu'auraient des interactions avec l'utilisateur en employant notre kernel semi-supervisé.

6 Conclusion

Dans cet article, une nouvelle transformation de kernel semi-supervisée a été décrite et évaluée. Celle-ci est essentiellement basée sur le voisinage des données étiquetées. Comme nos expériences le montrent, très peu d'éléments étiquetés a priori sont suffisants pour influencer fortement une projection kernel PCA subséquente, tout en préservant la topologie originale des données. Cette réactivité permet de minimiser le nombre d'interactions avec l'utilisateur, tout en lui fournissant un retour tangible dans le contexte d'une visualisation. La séparation en clusters est également renforcée par notre méthode, une caractéristique plutôt intéressante pour faciliter la caractérisation visuelle de ces objets.

Les expériences ont également permis de montrer l'importance du paramètre ajustable α dans le contexte de notre kernel semi-supervisé. Son augmentation tend a diminuer de nombre de clusters estimés sur les données projetées, avec une augmentation linéaire des artéfacts de projection en contrepartie, et pas d'avantage apparent sur le plan de la pureté des clusters.

Aucun paramétrage n'est donc optimal de manière évidente : dans un contexte interactif, il est donc préférable de proposer une valeur intermédiaire par défaut (e.g. 3), tout en permettant à l'utilisateur de l'ajuster ensuite selon sa préférence. Nous avons démontré la grande sensibilité de notre proposition aux éléments étiquetés a priori. Cette propriété a un revers : si un seul

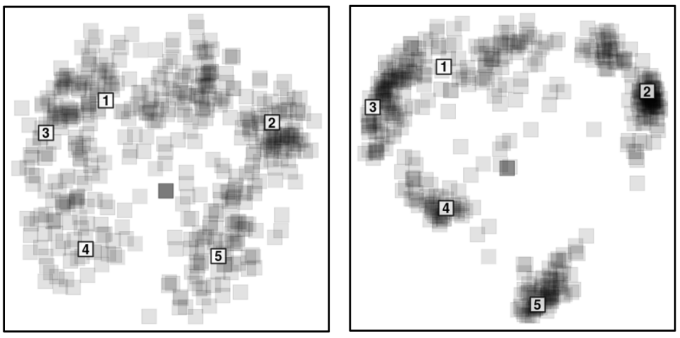


FIG. 3 – À gauche *: projection d'un sous-ensemble d'*Isolet *avec la transformation* unsupervised*. La teinte de gris indique la densité de données dans la projection, et un élément de chaque classe est surligné avec un chiffre distinctif.*
À droite *: projection du même sous-ensemble avec la transformation* neighbors*, utilisant l'ensemble surligné en tant que* $\boldsymbol{X}_L$*, et* $\alpha = 3$ *(voir équation* (5)*). Ceci accentue la séparation de l'échantillon en groupes, d'où résulte une plus grande compression de ses éléments par la projection.*

élément par classe est fourni, et est également mal choisi (e.g. outlier de sa classe), notre proposition a des conséquences plutôt négatives sur la pureté des clusters estimés par la suite. Toutefois, l'augmentation du nombre d'éléments étiquetés compense rapidement ce handicap initial.

Au travers de ce travail, nous avons voulu décrire et évaluer en détail notre méthode de projection semi-supervisée. Nous voulons inscrire celle-ci dans la construction d'un système de clustering visuel interactif, mais nous avons d'abord voulu étudier ses propriétés indépendamment de considérations liées aux interactions avec un utilisateur. Nous avons cependant tracé les contours d'une potentielle implémentation de ce système dans la section 4.1. Dans ce contexte, la vérité terrain serait l'expertise que l'utilisateur possède sur les données, et la performance de la méthode serait idéalement mesurée selon sa faculté à s'approcher efficacement d'un clustering respectant au mieux la vérité terrain spécifique de l'utilisateur. L'idée générale derrière un tel système serait de permettre à un utilisateur d'étiqueter les éléments de manière interactive, directement au travers de la projection 2D, puis d'adapter celle-ci de manière dynamique à ces actions. Au-delà de la définition de cinématiques adéquates pour ces interactions, nous tenons à souligner que notre travail n'est pas adaptable tel quel à ce contexte interactif. En effet, chaque interaction transforme la matrice de kernel de manière non-linéaire. En considérant une approche naïve, le calcul de la projection modifiée requiert $O(N^3)$ opérations en pratique. Des optimisations sont déjà possibles du fait que seules les deux premiers vecteurs et valeurs propres sont nécessaires ; mais il doit exister un algorithme (ou au moins une heuristique) permettant de mettre à jour *en ligne* la projection définie par l'équation (2) selon le différentiel de chaque interaction, minimisant ainsi le coût calculatoire.

Références

Aupetit, M. (2007). Visualizing distortions and recovering topology in continuous projection techniques. *Neurocomputing*, 1304–1330.

Basu, S., A. Banerjee, et R. Mooney (2002). Semi-supervised clustering by seeding. *Proceedings of 19th International Conference on Machine Learning*.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bruneau, P. (2012). VBmix : a R package for Variational-Bayes mixture learning. Technical report, LINA (CNRS UMR 6241).

Chapelle, O., J. Weston, et B. Schölkopf (2003). Cluster kernels for semi-supervised learning. *Advances in Neural Information Processing Systems 15*, 585–592.

Donaldson, T. S. (1968). Robustness of the F-test to errors of both kinds and the correlation between the numerator and denominator of the F-ratio. *Journal of the American Statistical Association*, 660–676.

François, D., V. Wertz, et M. Verleysen (2005). About the locality of kernels in high-dimensional spaces. *International Symposium on Applied Stochastic Models and Data Analysis*, 238–245.

Keim, D., G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, et G. Melançon (2008). Visual analytics : Definition, process, and challenges. In *Information Visualization*, pp. 154–175. Springer.

Law, M. H. C., A. Topchy, et A. K. Jain (2005). Model-based clustering with probabilistic constraints. *Proceedings of SIAM Data Mining*.

Miller, D. J. et D. J. Uyar (1996). A mixture of experts classifier with learning based on both labelled and unlabelled data. *Advances in Neural Information Processing Systems 9*, 571–577.

Schölkopf, B., A. Smola, et K.-R. Müller (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 1299–1319.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer.

Ware, C. (2004). *Information Visualization : Perception for Design*. Elsevier.

Summary

In this paper, a new kernel transformation procedure is described. It aims at incorporating a degree of supervision directly in the original pairwise similarities of a data set. The modified similarities can then be projected using a 2D kernel PCA, so as to reflect the compromise between genuine data and user knowledge, while being affordable for visualization and interaction. Such semi-supervised projections are evaluated with synthetic and real data, in the context of a simulated visual clustering task. Randomly selected subsets of elements are chosen to hold a label, thus reproducing actual user interactions. The results show the effectiveness of the method, with as few as one labelled element per class inducing tangible effects.

Improved Cluster Tracking for Visualization of Large Dynamic Graphs

Chris Muelder*, Arnaud Sallaberry**, Kwan-Liu Ma*

* VIDi group - University of California, Davis
One Shields Avenue
Davis, CA 95616-8562, USA
muelder@cs.ucdavis.edu, ma@cs.ucdavis.edu
http://vis.cs.ucdavis.edu/~muelder/, http://www.cs.ucdavis.edu/~ma/

**LIRMM - Université Montpellier 3
UMR 5506 - CC 477
161, rue Ada
34095 Montpellier Cedex 5, France
arnaud.sallaberry@lirmm.fr
http://www2.lirmm.fr/~sallaberry/

Abstract. Analysis and visualization of dynamic graphs is a challenging problem. Clustering can be applied to dynamic graphs in order to generate interactive visualizations with both high stability and good layout quality. However, the existing implementation is naïve and unoptimized. Here we present new algorithms to improve both the temporal clustering results and the efficiency of the cluster tracking calculation, and evaluate the results and performance.

1 Introduction

In recent years, the domain of network visualization has yielded many techniques for exploring large graphs. Most of these deal with static networks and are based on graph drawing algorithms (see for example Hachul and Jünger (2006) or Muelder and Ma (2008)), clustering techniques (see Schaeffer (2007) for an introduction), or exploratory methods (see for example van Ham and van Wijk (2004), Abello et al. (2006) or Archambault et al. (2008)). In contrast, fewer works have been devoted to the exploration of dynamic graphs. A dynamic graph is an evolving graph where vertices and edges are added and removed over time. Examples of such graphs include social networks, dependency graphs in software engineering, website hyperlinks, router networks, collaboration networks, *etc.*

When creating a node-link diagram for a dynamic graph, not only does the layout need to consider graph topology, but also the stability between time-steps. This generally forces a trade-off between layout quality and stability, as a perfectly stable layout would sacrifice layout quality, and naively calculating ideal layouts would not offer stability. While there are a number of existing methods for creating these layouts, they have not been shown to scale well to large dynamic graphs.

A dynamic clustering based approach for visualization of dynamic graphs can provide an overview of the entire dynamic graph over time, yield high quality layouts for every time-step, minimize node motion between time steps to provide stability and preserve the user's mental map, and allow for interactive exploration even under random access patterns. One existing approach consists of first clustering each time-step independently to guarantee good locality for every time-step, then tracking the clusters between time-steps, and finally arranging the clusters and nodes such that nodes that are consistant are stationary and transitional node motion is minimized Sallaberry et al. (2013). This produces a temporal arrangement which we directly visualize as a timeline, and which is used to define layouts for a node link diagram for each time-step which both meets general layout criteria (namely cluster co-location and short average edge length) and where node motion is minimized between time-steps.

However, there are some limitations to this approach. First, by only considering pairwise timesteps, it is impossible to track clusters that disappear for some amount of time before reforming, which results in the formation of extraneous dynamic clusters. Second, the association between timesteps was previously cacluated very naïvely, with pairwise comparison and Jaccard calculation for each possible association.

Here, we describe improvements to this approach that resolve both of these issues. The first issue is addressed by an algorithm for preserving unused dynamic clusters from old time steps and allowing them to be reincorporated into new time steps. And the second issue is addressed through a heavy optimization of the association algorithm that not only reduces the number of clusters that have to be compared but which also enables direct calculation of the Jaccard index. The result of the combination of these improvements is a more compact, robust, dynamic clustering that can be computed much more efficiently.

2 Related Works

A common method for visualizing dynamic graphs is to animate the transitions between time-steps (North, 1996; Diehl and Görg, 2002; Erten et al., 2004; Görg et al., 2004; Boitmanis et al., 2008; Frishman and Tal, 2008). This approach yields dynamic visualization with nodes appearing, disappearing and moving to produce readable layout for each time-step. Alternatively, multiple time-steps can be staticly placed next to each other using “Small Multiples" Tufte (1990). This eases the comparison of distant time-steps but the area devoted for each time-step is small and this reduces the readability of each graph. An empirical study to compare the advantages and drawbacks of these approaches ("Animation" vs. "Small Multiples") has been performed by Archambault et al. (2011). A major issue for both methods is to ensure the stability of the layout (Kumar and Garland, 2006; Frishman and Tal, 2008; Brandes and Mader, 1012; Hu et al., 2012). A stable layout helps preserve the user's mental map as there is less movement between time-steps, but sacrifices quality in terms of readability for later time-steps as their layout depends on previous time-steps. Many experiments has been proposed to examine the effect of preserving the mental map in dynamic graphs visualization (Purchase et al., 2007; Saffrey and Purchase, 2008; Purchase and Samra, 2008). The results of Purchase and Samra (2008) were quite surprising because the most effective visualizations were the extreme ones, *i.e.* the ones with very low or high mental map preservation: visualizations with medium preservation performed less well. The approach described in this paper aims to achieve high mental map preservation.

An interesting visualization dealing with dynamic large directed graphs has been proposed by Burch et al. (2011). Vertices are ordered and positioned on several vertical parallel lines, and directed edges connect these vertices from left to right. Each time-step's graph is thus displayed between two consecutive vertical axes. Hu et al. (2012) proposed a method based on a geographical metaphor to visualize clustered dynamic graphs. However, their approach requires one global clustering over time, while ours allows nodes to be transferred in order to create better local clusterings and to capture the evolutions of the communities.

Finding a partition of the nodes of a static graph according to its structure is a well studied problem; Schaeffer (2007) has published a good overview of graph clustering methods. But clustering a dynamic graph is a less studied problem.

3 Clustering

A dynamic graph can be defined formally as an agglomerate graph $G = (V, E)$ and an ordered sequence of subgraphs $S = \{G_1 = (V_1, E_1), G_2 = (V_2, E_2), ..., G_k = (V_k, E_k)\}$ where each G_t is the subgraph of G at time t. $V, V_1, V_2, ..., V_k$ are finite and non-disjointed sets of nodes, $E, E_1, E_2, ..., E_k$ are finite and non-disjointed sets of edges such that $V = V_1 \cup V_2 \cup ... \cup V_k$ and $E = E_1 \cup E_2 \cup ... \cup E_k$. What we need is to create a time-varying clustering, *i.e.* a set of clusters evolving over time. The clustering method we describe here is a two step algorithm. The first step consists of partitioning the nodes for each time step independently. Then, we associate these clusters through time to derive time-varying clusters.

3.1 Time-step Clusterings

To calculate a dynamic clustering, we first find a partition for each time step, *i.e.* a set of clusterings $C = \{C_1, C_2, ..., C_k\}$ where $C_t = \{c_1^t, c_2^t, ..., c_{lt}^t\}$ is a partition of the nodes V_t of G_t. In this paper, we call each C_t a "time-step clustering" where c_i^t is the "time-step cluster" i at time t, and $c_i^t \subseteq V_t$ for each $i \in [\![1, lt]\!]$, $V_t = c_1^t \cup c_2^t \cup ...c_l^t$ and $c_i^t \cap c_j^t = \emptyset$ for each pair $(i, j) \in [\![1, lt]\!]^2$.

Our algorithm is based on the so-called modularity function of Newman and Girvan (2004). It represents the sum of the number of edges linking nodes of the same clusters minus the expected such sum if edges were distributed at random. For a graph $G_t = (V_t, E_t)$ and a partition C_t of its nodes, the modularity $Q(C_t)$ is defined by:

$$Q(C_t) = \frac{1}{2|E_t|} \sum_{u,v \in V_t} \left[A_{uv} - \frac{k_u k_v}{2|E_t|} \right] \delta(c^t(u), c^t(v))$$

where $|E_t|$ is the number of edges, A_{uv} is 1 if there is an edge between u and v and 0 otherwise, $k_u = \sum_v A_{uv}$ is the number of edges attached to u, $c^t(u)$ is the time-step cluster of C_t containing u, $\delta(c^t(u), c^t(v))$ is 1 if $c^t(u) = c^t(v)$ and 0 otherwise.

A partition that maximizes this function helps to discover clusters of densely connected communities. Moreover, as shown by Noack (2008), optimizing the modularity is the same as optimizing an energy function in graph layout. This equivalence implies that our layout based on such a clustering algorithm yields a good representation of the graph.

The problem of finding a partition that maximizes the modularity is hard, and the corresponding decision problem is NP-complete (Brandes et al., 2006). We use the heuristic proposed by Blondel et al. (2008), which works well in terms of both the quality of the results and the computation time. Initially, each node belongs to its own cluster. Then pairs of clusters are recursively merged such that the modularity of the partitioning increases. If two possible merges involve the same cluster, the merge that improves the modularity the most is performed.

3.2 Time-varying Clustering

3.2.1 Overview

The previous approach (Sallaberry et al., 2013) was to compare each time-step cluster in the current time-step pairwise with the time-step clusters of the previous time-step according to the Jaccard index, and then to itratively and greedily associate the time-step clusters that most closely match into the same time-varying cluster, halting when all clusters are assigned or the similarity falls below a user-defined threshold. If there are any remaining new clusters that do not have a match, they are considered new clusters, and so they start new time-varying clusters. And any remaining time-varying clusters present in the previous time-step that were not assigned a cluster in the current time-step were discarded, as there was no match.

Our approach here operates fairly similarly, with two key differences. The first improvement is that rather than discarding time-varying clusters after they fail to find a match, we retain them in the matching algorithm so that they can return if that portion of the network returns to a similar enough state, even if there are many intervening time-steps. This reduces the overall number of time-varying clusters, and can aid in identifying certain patterns (such as periodicity, split/merges, clustering instability, etc...). The second improvement is to reduce the number of pairwise clusters to evaluate, by only comparing clusters that share at least one vertex. This optimization method has the added bonus that it can be used to calculate the Jaccard index very efficiently.

3.2.2 Details

We define a time-varying clustering of a dynamic graph G as a set of time-varying clusters $VC = \{VC_1, VC_2, ..., VC_l\}$. Each of these time-varying clusters is an ordered sequence $VC_i = \{vc_i^1, vc_i^2, ..., vc_i^k\}$ where k is the number of time steps and each vc_i^t is a subset of the vertices V_t at time t. That is, each time-varying cluster VC_i is a cluster whose membership can evolve over time, where vc_i^t represents the set of nodes in the cluster i at time t. As the number of time-step clusters can change between timesteps, not every time-varying cluster is populated at every timestep, and the total number of time-varying clusters l can be larger than the number of time-step clusters at any time step.

We start from an empty set VC of time-varying clusters and we create a time-varying cluster VC_i for each time-step cluster c_i^1 of the first time-step clustering C_1. The set of nodes of these time-varying clusters VC_i at time 1 are initialized with the time-step clusters c_i^1: $vc_i^1 \leftarrow c_i^1$. Then, for each subsequent timestep t, we want to compute similarities between each time-step cluster $c_i^t \in C_t$ and potential time-varying clusters VC_i. In our implementation, we use the Jaccard index to compute the similarities. For two clusters c_i^{t-1} and c_j^t, this is defined by the equation $|c_i^{t-1} \cap c_j^t| / |c_i^{t-1} \cup c_j^t|$. There are two main advantages in using this metric.

First it takes into account the number of shared nodes as well as the total number of nodes, which guarantees homogeneity between consecutive steps of a time-varying cluster. Secondly it returns a value normalized between 0 and 1 which is helpful for empirically defining a $threshold$.

In the original algorithm, the association step was performed by comparing each time-step cluster $c_i^t \in C_t$ to every time-step cluster $c_j^{t-1} \in C_{t-1}$ to create a similarity matrix, which costs $O(|C_t| * |C_{t-1}|)$ times the cost of the similarity calculation. Associations are then performed greedily, starting with the largest matrix value, and stopping when either one set of clusters is exhausted or the remaining matrix values are less than the threshold. Then, any remaining clusters in C_t were assigned new time-varying clusters in VC. And any remaining clusters in C_{t-1} were discarded, and their corresponding time-varying cluster in VC was terminated for the remainder of the execution.

However, we found that this process led to the creation of many new time-varying clusters, because sometimes the network might revert to a clustering previously encountered where the corresponding time-varying cluster was already terminated, so the method would create a new time-varying cluster. To resolve this, we preserve the most recent time-step cluster c_k^u for each time-varing clusters in VC that would have been terminated, and compare against those as well, where $0 < u < t - 1$. If an older c_k^u is more similar than any time-step cluster $c_j^{t-1} \in C_{t-1}$, then the system will select and revive the time-varying cluster of c_k^u instead of that of one of active time-varying clusters VC_i. This adds additional complexity to the algorithm, but produces more robust and compact results, as time-varying clusters are allowed to reform instead being terminated.

Another issue is that many of the time-step clusters are disjoint sets of nodes, and thus we do not need to compute their similarity. In our improved version, rather than computing the entire matrix, we consider only the clusters that share at least one node. We do this by computing a list of candidate clusters CC_i^t for each time-step cluster c_i^t, which we define as $CC_i^t = \{(vc_a^*, |c_i^t \cap vc_a^*|), (vc_b^*, |c_i^t \cap vc_b^*|), ...\}$, where each vc_j^* is the most recent timestep of VC_j. This can be computed relatively efficiently by iterating over each node $n \in c_i^t$: for each n we take any vc_j^* that contains n, then either add vc_j^* to CC_i^t with a paired value of 1, or if it is already in the list, we simply increment its paired value. To make this process even more efficient, we use a lookup table to map each node n to its existing vc_j^* clusters. In the original process, this would be trivial to do, as each node n could only exist in one previous cluster vc_j^*. However, since we are now preserving terminated clusters, it is possible for n to have multiple previous clusters. So we build a hash to map each node n to its prior clusters by iterating over each vc_j^* and inserting a pointer to vc_j^* for each node $m \in vc_j^*$. This elimates the need for any searching. While this process appears to add some computational overhead as it iterates over every node instead of working with the clusters, this is entirely offset in the calculation of the Jaccard index, as we have already computed the size of each intersection $|c_i^t \cap vc_j^*|$. From this, we can calculate the Jaccard index in $O(1)$ time as $J(c_i^t, vc_j^*) = \frac{|c_i^t \cap vc_j^*|}{|c_i^t| + |vc_j^*| - |c_i^t \cap vc_j^*|}$. So while this optimization does add additional memory overhead, the computation is much more efficient.

4 Ordering

Our visualization is based not just on a clustering, but on an ordering of the time-varying clusters (in the next sections, we use the word “cluster" instead of “time-varying cluster" to simplify the notation). Then, nodes are also ordered within each cluster. A node that moves from a cluster VC_a to a cluster VC_b is involved in the node ordering of both VC_a and VC_b, *e.g.* it can be the 6th node of VC_a and the 3rd node of VC_b.

4.1 Ordering clusters

The stability of the layout is one of the main goals of our method: we want to easily see the evolution of the clusters and also be able to follow nodes that move between clusters. As the layout depends on the ordering, clusters need to be ordered in such a way that two clusters exchanging many nodes are close to each other.

We do this by first creating a weighted quotient graph $QG = (V_{QG}, E_{QG}, \omega)$ defined by the relationships between the time-varying clusters VC of G. Each node of V_{QG} represents a cluster of VC, *i.e.* $V_{QG} \leftarrow VC$. There is an edge in E_{QG} between VC_i and VC_j if and only if there is at least one node in the sets of VC_i that is also in a set of VC_j. The weight function ω is a function $\omega : E_{QG} \to \mathbb{N}$ defined for each edge $e = (VC_i, VC_j)$ as the number of transferred of nodes between sets of VC_i and sets of VC_j.

Next we need to find an ordering of these clusters, *i.e.* a permutation $\phi : V_{QG} \to \{1, 2, ..., |V_{QG}|\}$ that minimizes the function:

$$LA_\phi(QG) = \sum_{\substack{uv \in E_{QG} \\ u,v \in V_{QG}}} \omega(uv) \cdot |\phi(u) - \phi(v)|$$

This function is called the *Linear arrangement function* (LA) and finding an ordering that minimizes it is known as the *Minimum Linear Arrangement Problem*, MinLA (Petit, 2001). MinLA is NP-hard and the corresponding decision problem is NP-complete (Garey and Johnson, 1979). Many heuristics have been proposed to find a satisfying solution. A list of these methods and an experiment has been proposed by Petit (2001). More recently, Koren and Harel (2002) have proposed a new heuristic that is a good compromise between computation time and quality of the results.

4.2 Ordering nodes

The second ordering step consists in finding a permutation of the nodes within each cluster VC_i of VC. Since we want to maximize stability, we calculate this permutation over all time, so that nodes will not move within a cluster, even if this leaves gaps at some time-steps. As with the clusters, this is another MinLA problem. Let vc_i be the set of nodes of VC_i: $vc_i = \bigcup_{1 \le t \le k} vc_i^t$. Then the permutation is defined as $\varphi_i : vc_i \to \{1, 2, ..., |vc_i|\}$. This ordering needs to take into account the ordering of the clusters computed previously: for example, if a node v moves only once from a cluster VC_a to a cluster VC_b and if $\phi(VC_a) < \phi(VC_b)$, then v should lie at the upper extremity of VC_a (high $\varphi_a(v)$) and at the lower extremity of VC_b (low $\varphi_b(v)$). To find the permutation φ_i, we first compute for each node v of vc_i the median

of the clusters VC_i v belongs to:

$$median_i(v) = \frac{\sum_{VC_j; v \in vc_j} \phi(VC_j)}{|\{VC_j; v \in vc_j\}|}$$

Then, the permutation φ_i is the ordering obtained by sorting the nodes of vc_i according to their median value: $median_i(v) < median_i(u) \Leftrightarrow \varphi_i(v) < \varphi_i(u)$.

5 Visualization

The visualization methods we employ focus on representing the evolving clusters in dynamic graphs. We employ two views: a time-line inspired by Ogawa and Ma (2010); Tanahashi and Ma (2012) that provides an overview of the entire dynamic graph, and a more traditional node-link view for individual time-steps. Both of these views are derived from the clustering and ordering methods described earlier. Moreover, since the clustering and ordering are computed as a preprocessing step, the computation times of the visualizations are linear, which makes it possible to obtain real-time, interactive navigation of the dynamic graph.

5.1 Time-line view

The time-line view depicts an overview of the nodes' arrangement into clusters and of the nodes motion between clusters. Each node is represented as a line where the x-position is time and the y-positions corresponds the cluster the nodes belong to at each given time and its position within the cluster. Figure 1(a) shows an example. For reference purposes, in this diagram the time-steps are represented with vertical grey lines (from t=1 to 5) positioned along the x-axis that represents time. There are 8 plotted lines (including black, blue and green ones), which correspond to 8 nodes. There are four clusters on the y-axis, and a horizontal line is in front of one of them when the corresponding node belongs to it. Clusters are positioned according to the ordering ϕ computed by the pre-processing algorithm, from bottom to top (*e.g.* the cluster labelled 4 is the cluster VC_a with a such that $4 = \phi(VC_a)$).

As an example of reading this plot, consider the blue line. The corresponding node v belongs to the cluster 4 at times 1 and 2, and it belongs to the cluster 3 at times 3 and 4, since the blue line moves from cluster 4 to cluster 3 at time 3. Also, the blue node is no longer in the graph at the time-step 5 and that the green node appears in the graph at the time-step 2.

Lines in the clusters are positioned according to the orderings φ_i computed during the pre-processing step. In this way, a node v that moves from a cluster VC_a to a cluster VC_b with $\phi(VC_a) < \phi(VC_b)$ is likely to be positioned at the upper extremity of VC_a (high $\varphi_a(v)$) and at the lower extremity of VC_b (low $\varphi_b(v)$). This technique reduces edge crossings and improves the readability of the view.

Clusters are separated by a constant gap to clarify their distinctions. The height of a cluster VC_i corresponds to the size the set vc_i of all the nodes that belong to it at least for one time-step. As an example, $|vc_b| = 4$ (see the red circle 1) and $|vc_c| = 3$ with c such that $2 = \phi(VC_c)$ (see the red circles 2 and 3). Thanks to this, a node has always the same position in the same cluster, so there will be no bends when a node remains in the same cluster and the area devoted to a cluster remains the same.

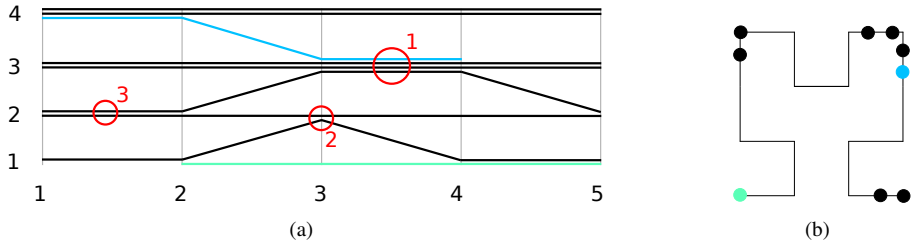


FIG. 1: (a) The time-line gives an overview of the clusters and of the nodes moving from clusters to clusters. Each horizontal or bent line is a node. Vertical grey lines represent time-steps, from 1 to 5. Y-axis represents clusters, *e.g.* the blue line near the cluster 4 at time-steps 1 and 2 stands for a node v that belongs to VC_a with a such that $4 = \phi(VC_a)$, at the time 1 and at the time 2 (it belongs to vc_a^1 and vc_a^2 but not to vc_a^3, vc_a^4 and vc_a^5). (b) Time-step view of the graph used in the example of Figure 1(a). It shows the graph at the 3rd time-step. We don't display the edges here. Nodes represented as disks are positioned along a Hilbert's curve, represented by the black bended line.

5.2 Time-step view

The second view is a node-link diagram that shows the graph at any selected time-step. The layout is based on the technique of Muelder and Ma (2008), which maps a 1-D ordering of nodes to a space-filling curve to define the layout.

Since we have already computed a stable ordering of nodes, it is sensible to map this same ordering onto the space-filling curve. In the timeline, the height of each line at any time step corresponds to the pre-computed ordering. So, we can reuse these y-positions as a 1-D layout for that time-step, then map the nodes directly to a space-filling curve by placing the nodes at the corresponding distance along the curve. This is done by normalizing both the 1-D layout and the length of the curve, then calculating the position of each node by recursively mapping it to the curve in constant time, as in the original paper (Muelder and Ma, 2008). Figure 1(b) shows an example of this node positioning on the same example as the one presented in Figure 1(a) for time-step 3. In this diagram, we use a Hilbert curve, but we also use Peano curve, a Gosper curve, and an H-curve, and the user can switch between these curves as desired (see Haverkort and van Walderveen (2010) for a summary of well-known space-filling curves).

One interesting property of a space-filling curve is known as the *Worst-Case Locality* (Haverkort and van Walderveen, 2010). This property guarantees that the euclidean distances between nodes in the layout are bounded by the distances of the same nodes in the one-dimensional layout. So, the proximities of elements (nodes/clusters) depend directly on the ordering. As the ordering is based primarily on the connectivity of the networks, this guarantees layout quality metrics, such as tightly connected groups of nodes being placed close together with a good aspect ratio, and short average edge lengths.

Since a node has always the same position in the time-line when it is in the same cluster and the area devoted to a cluster remains the same, its placement in the layout will also be constant. This ensures the stability of the layout. Even the distance that nodes move is minimized, as the ordering is such that clusters that exchange many nodes are placed closer together.

As the layout itself runs in linear time, the visualization can be updated interactively by the user and we can even easily play the sequence of graphs and animate the transitions with graphs of tens of thousands nodes/edges (see our previous paper (Sallaberry et al., 2013) for more details).

As we use a clustering hierarchy, we can also employ the hierarchical edge bundling technique of Holten (2006) which improves the readability of the graph. Control points of the spline linking a node v and a node u are defined by the path through the clustering hierarchy, and placed according to the clusters' centroids.

5.3 Interaction and navigation

One of the most useful features of our approach is that any time-step can be laid out quickly and directly, without needing to iterate over the other time-steps. The benefit of this is that it enables random access. That is, users can find interesting time-steps in the time-line and skip between them directly. We enable this form of interaction by letting the user simply click in the time-line on the time-step that they want to load. We also include the more traditional approach of simply animating over the entire dynamic graph. In either case, the positions of nodes that move are interpolated between time-steps so that the user can follow their motion. Within the node-link diagram itself, we can also allow for traditional graph interaction, such as selection, or focus+context zooming.

6 Discussion

We have described two algorithmic improvements over the previous approach. First, our new approach is tuned to reduce the total number of persistant time-varying clusters, which will improve the space utilization of the resulting visualization. Second, we have substantially optimized the cluster association step, which greatly reduces the time it takes to process the data. Here, we present some case studies and quantitatively evaluate both improvements.

To test our improvements, we ran both improvements as well as the original approach on several datasets. First, we ran them on a social network dataset collected from the Rimzu social networking site, as used by Frishman and Tal (2008). While quite small and straightforward, this dataset makes a good baseline for comparison against existing works. Next, we evaluated the MIT Reality dataset (Eagle and Pentland, 2005). In this dataset, there is a small, strong core of about 80 people, but hundreds of periferal nodes that do persist through the data and which generally only have one connection. Due to their transitivity, they do not contriubte much to the overall structure of the network, so we evaluate both the entire network and just the core network without the external nodes. Finally, we evaluate the improvement on a dataset of the autonomous systems of the internet, derived from data collected by the Oregon Route Views project and as used in several existing works (Muelder, 2011; Sallaberry et al., 2013).

In each case, the results followed our expectation, as is shown in Table 1. The inclusion of additional comparisons against dead clusters succeeds in reducing the total number of dynamic time-varying clusters, but requires more comparisons, and hence takes more computation time. However, this is entirely offset by the optimization, which reduces the computation time by around a factor of 20 or more.

	Original		Improved		
	N Clusters	Time (ms)	N Clusters	Time (ms)	Opt. time (ms)
Social network	392	3050	391	3150	160
MIT reality	518	120	457	518	80
MIT reality (core)	136	30	78	50	<1
Internet (16 steps)	162	7220	86	9640	400

TAB. 1: Results of our improvements. We found that our methods reduce the number of time-varying clusters by up to half, and reduce computation time by a factor of about 20 or more.

7 Conclusion and Future Work

This paper presents an incremental improvement to a previous work. While the visual representation remains the same, the underlying clustering algorithm has been greatly improved and optimized. The evaluation follows accordingly, and demonstrates our improvements quantitatively.

While the results described in this paper are promising, they are still further ways they can be improved. Most importantly, the cluster and node ordering steps are still extremely slow and we are investigating ways to improve upon this. One such method is to assign more localized orderings, where we find the actual time range used by each dynamic cluster, and more compactly arrange the clusters accordingly, similar to the work of Tanahashi and Ma (2012). However, this algorithm is also very time consuming, and we are investigating heuristic improvements.

References

Abello, J., F. van Ham, and N. Krishnan (2006). ASK-GraphView: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics 12*(5), 669–676.

Archambault, D., T. Munzner, and D. Auber (2008). GrouseFlocks: Steerable exploration of graph hierarchy space. *IEEE Transactions on Visualization and Computer Graphics 14*(4), 900–913.

Archambault, D., H. C. Purchase, and B. Pinaud (2011). Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics 17*(4), 539–552.

Blondel, V., J. Guillaume, R. Lambiotte, and E. Lefebvre (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*.

Boitmanis, K., U. Brandes, and C. Pich (2008). Visualizing internet evolution on the autonomous systems level. In *Proceedings of the International Symposium on Graph Drawing (GD'07)*, Volume 4875 of *LNCS*, pp. 365–376. Springer.

Brandes, U., D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner (2006). Maximizing modularity is hard. `arxiv.org/abs/physics/0608255`.

Brandes, U. and M. Mader (1012). A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In *Proceedings of the International Symposium on Graph Drawing (GD'11)*, Volume 7034 of *LNCS*, pp. 99–110. Springer.

Burch, M., C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf (2011). Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics 17*(12), 2344–2353.

Diehl, S. and C. Görg (2002). Graphs, they are changing. In *Proceedings of the International Symposium on Graph Drawing (GD'02)*, Volume 2528 of *LNCS*, pp. 23–30. Springer.

Eagle, N. and A. S. Pentland (2005). CRAWDAD data set mit/reality (v. 2005-07-01). Downloaded from http://crawdad.cs.dartmouth.edu/mit/reality.

Erten, C., P. J. Harding, S. G. Kobourov, K. Wampler, and G. V. Yee (2004). GraphAEL: Graph animations with evolving layouts. In *Proceedings of the International Symposium on Graph Drawing (GD'03)*, Volume 2912 of *LNCS*, pp. 98–110. Springer.

Frishman, Y. and A. Tal (2008). Online dynamic graph drawing. *IEEE Transactions on Visualization and Computer Graphics 14*(4), 727–740.

Garey, M. R. and D. S. Johnson (1979). *Computers and Intractability: a Guide to the Theory of NP-Completeness*. W. H. Freeman.

Görg, C., P. Birke, M. Pohl, and S. Diehl (2004). Dynamic graph drawing of sequences of orthogonal and hierarchical graphs. In *Proceedings of the International Symposium on Graph Drawing (GD'04)*, Volume 3383 of *LNCS*, pp. 228–238. Springer.

Hachul, S. and M. Jünger (2006). An experimental comparison of fast algorithms for drawing general large graphs. In *Proceedings of the International Symposium on Graph Drawing (GD'05)*, Volume 3843 of *LNCS*, pp. 235–250. Springer.

Haverkort, H. J. and F. van Walderveen (2010). Locality and bounding-box quality of two-dimensional space-filling curves. *Computational Geometry, Theory and Applications 43*(2), 131–147.

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

Hu, Y., S. G. Kobourov, and S. Veeramoni (2012). Embedding, clustering and coloring for dynamic maps. In *Proceedings of the 5th IEEE Pacific Visualization Symposium (PacificVis 2012)*, pp. 33–40.

Koren, Y. and D. Harel (2002). A multi-scale algorithm for the linear arrangement problem. In *28th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2002)*, Volume 2573 of *LNCS*, pp. 296–309. Springer.

Kumar, G. and M. Garland (2006). Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics 12*(5), 805–812.

Muelder, C. (2011). *Advanced Visualization Techniques for Abstract Graphs and Computer Networks* . Dissertation, University of California, Davis.

Muelder, C. and K.-L. Ma (2008). Rapid graph layout using space filling curves. *IEEE Transactions on Visualization and Computer Graphics 14*(6), 1301–1308.

Newman, M. E. J. and M. Girvan (2004). Graph clustering. *Physical Review E 69*(026113).

Noack, A. (2008). Modularity clustering is force-directed layout. *CoRR abs/0807.4052*.

North, S. C. (1996). Incremental layout in DynaDAG. In *Proceedings of the International Symposium on Graph Drawing (GD'95)*, Volume 1027 of *LNCS*, pp. 409–418. Springer.

Ogawa, M. and K.-L. Ma (2010). Software evolution storylines. In *Proceedings of the ACM 2010 Symposium on Software Visualization (SoftVis'10)*, pp. 35–42.

Petit, J. (2001). Experiments on the minimum linear arrangement problem. Technical Report LSI-01-7-R, Universitat Politecnica de Catalunya, Departament de Llenguatges i Sistemes Informatics.

Purchase, H., E. Hoggan, and C. Görg (2007). How important is the "mental map"? - an empirical investigation of a dynamic graph layout algorithm. In *Proceedings of the International Symposium on Graph Drawing (GD'06)*, Volume 4372 of *LNCS*, pp. 184–195. Springer.

Purchase, H. and A. Samra (2008). Extremes are better: Investigating mental map preservation in dynamic graphs. In *Proceedings of the 5th International Conference on Diagrammatic Representation and Inference (Diagrams 2008)*, Volume 5223 of *LNCS*, pp. 60–73. Springer.

Saffrey, P. and H. Purchase (2008). The "mental map" versus "static aesthetic" compromise in dynamic graphs: A user study. In *Proceedings of the 9th Australasian User Interface Conference (AUIC2008)*, pp. 85–93.

Sallaberry, A., C. W. Muelder, and K.-L. Ma (2013). Clustering, visualizing, and navigating for large dynamic graphs. In *Proceedings of the International Symposium on Graph Drawing (GD'12)*, LNCS. Springer (to appear).

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review 1*(1), 27–64.

Tanahashi, Y. and K.-L. Ma (2012). Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics 18*(12), 2679–2688.

Tufte, E. R. (1990). *Envisionning Information*. Graphics Press.

van Ham, F. and J. J. van Wijk (2004). Interactive visualization of small world graphs. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'04)*, pp. 199–206.

Résumé

L'analyse et la visualisation de graphes dynamiques est un problème difficile. Une méthode de clustering que nous avons développée lors d'un précédent travail peut être appliquée à de tels graphes afin de générer des visualisations interactives à la fois stables et de bonne qualité. Cependant, l'implémentation existante est naïve et non optimisée. Dans cet article, nous présentons de nouveaux algorithmes pour améliorer à la fois les résultats du clustering dynamique et la rapidité des calculs. Nous comparons les résultats et le rendement par rapport à la méthode précédente.

Les dendro-matrices : une alternative aux dendrogrammes pour visualiser les résultats d'une classification ascendante hiérarchique

Renaud Blanch*, Rémy Dautriche* et Gilles Bisson**

*UJF-Grenoble 1; UPMF-Grenoble 2; Grenoble-INP; **CNRS; INRIA
Laboratoire d'Informatique de Grenoble, UMR 5217 F-38041, Grenoble, France
blanch@imag.fr, dautriche@imag.fr, bisson@imag.fr

Résumé. Le résultat d'une classification ascendante hiérarchique est classiquement présenté sous la forme d'un dendrogramme. Cette représentation fournit toute l'information disponible sur les classes mais occulte partiellement celle sur les individus qui ne peuvent être associés qu'à une seule classe élémentaire. Nous proposons une alternative au dendrogramme, qui dans le même espace présente comme lui la hiérarchie des classes, leur dissimilarité, mais permet de plus la comparaison d'individu à individu ; d'individu à classe ; et de classe à classe. Cette visualisation est un hybride entre un arbre (le dendrogramme) et une matrice (la matrice de distance). Nous présentons un ensemble de techniques d'interaction associées à cette visualisation.

1 Introduction

La classification ascendante hiérarchique (CAH, Ward Jr., 1963) est souvent utilisée pour construire des classes à partir d'individus dès lors que l'on dispose d'une mesure de similarité entre individus. Elle doit son succès en partie à la facilité qu'ont ses utilisateurs à comprendre la manière dont la classification est établie et au contrôle qu'ils ont a posteriori sur le nombre de classes souhaité pour établir une partition des individus. Cette compréhension s'appuie sur la représentation canonique utilisée pour représenter la hiérarchie de classes produite par la CAH : le dendrogramme ou hiérarchie indexée.

Cette représentation donne en effet l'arbre binaire des classes construites (par la structure nœuds/liens) mais aussi une information sur l'homogénéité de chaque classe (par la hauteur du palier qui la représente). On peut ainsi visuellement se rendre compte s'il existe une rupture dans la répartition des paliers qui permettra alors de choisir d'un niveau de coupure du dendrogramme pour établir in fine les classes significatives de la partition.

Une information est néanmoins perdue par la représentation sous forme de dendrogramme : il s'agit des détails sur les individus. On peut en restaurer un partie en juxtaposant au dendrogramme une *heat map* (Wilkinson et Friendly, 2009) représentant les données brutes, mais cette visualisation ne donne pas directement d'information sur le degré de similarité entre les individus. Elle ne permettra donc pas de répondre à des questions sur la classification du type : « pourquoi cet individu a été classé ici plutôt que là ? »

Nous proposons donc une représentation alternative au dendrogramme qui, dans le même espace, représente la hiérarchie de classes, leur homogénéité, mais également l'information de distance entre les individus classifiés. Nous présentons une série de techniques d'interaction qui permet d'exploiter cette visualisation pour mieux comprendre les résultats de la CAH.

Du point de vue de la classification, nous n'utilisons que des techniques courantes. Du point de vue de la visualisation et de l'interaction, nos travaux peuvent être comparés à d'autres visualisations hybrides proposées ces dernières années. Par exemple, les représentations d'arbres à l'aide d'hybrides nœuds/liens et emboîtements à la *treemaps* ont été étudiées par McGuffin et Robert (2010), mais leur travaux ne concernent que les arbres. D'un autre côté, des représentations hybrides de graphes nœuds/liens et matrice d'adjacence ont été étudiées par Henry et al. (2007), Henry et Fekete (2007) ou Rufiange et al. (2012). Mais là encore, elles ne s'intéressent qu'à des structures de graphe et ne superposent pas deux types de données comme nous le faisons.

2 Dendro-matrice

Nous avons nommé cette visualisation *dendro-matrice* car il s'agit d'une représentation hybride combinant l'arbre binaire de la hiérarchie des classes obtenue par la CAH et la matrice de similarité des individus. La Figure 1 (a) montre un jeu de données (cinq points du plan) utilisé dans la suite pour illustrer le principe de construction des dendro-matrices. La Figure 1 (b) montre le résultat d'une CAH sur ce jeu de données en utilisant la distance euclidienne pour mesurer la similarité entre individus et le minimum de cette distance entre les éléments de deux classes pour mesurer la similarité entre classes (i.e., *single linkage*).

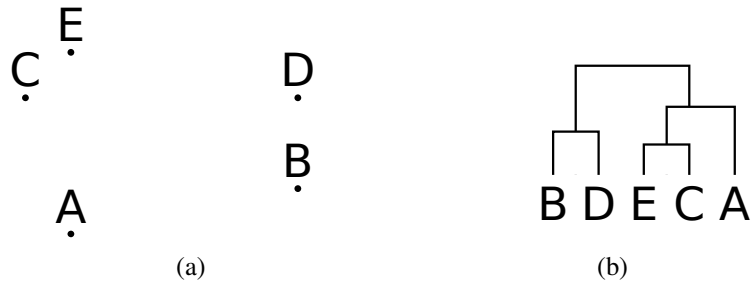


FIG. 1 – *Données utilisées pour l'illustration : (a) points du plan ; et (b) dendrogramme issu de la CAH utilisant la distance euclidienne et le minimum des distances entre individus comme distance de groupe.*

2.1 Visualisation

En partant de la hiérarchie des classes et de la matrice de similarité, on construit la dendro-matrice :

1. en encodant visuellement l'information contenue dans la matrice de similarité (ici par la taille des points) —Figure 2 (a) ;

2. en réordonnant les lignes et colonnes de la matrice avec un ordre des individus issu de la traversée en largeur des feuilles de la hiérarchie fournie par la CAH —Figure 2 (b) ;
3. en surlignant les blocs formés par les classes (qui sont contigus de par l'ordre choisi à l'étape précédente) et en encodant leur homogénéité dans leur couleur de fond (ici, plus le fond est foncé, plus la classe est homogène) —Figure 2 (c) ; et
4. en ne conservant que la moitié supérieure de la matrice (étant symétrique, une moitié en contient toute l'information), et en la basculant de 45° (pour placer la racine de l'arbre en haut) —Figure 2 (d).

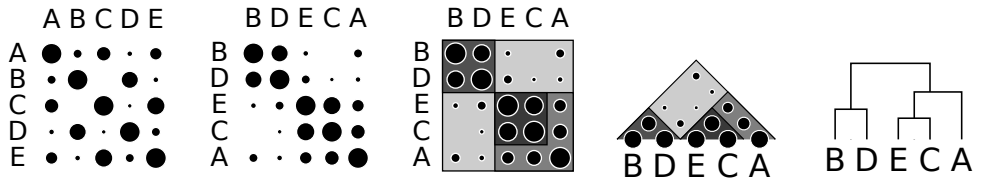


FIG. 2 – *De la matrice de similarité à la dendro-matrice : (a) encodage graphique de la similarité ; (b) matrice réordonnée ; (c) encodage graphique des classes ; et (d) la dendro-matrice comparée au (e) dendrogramme présentant le même résultat de CAH.*

Pour la première étape, la similarité est normalisée en divisant toutes les valeurs par le maximum de similarité observé. Le rayon des cercles utilise la racine carrée de cette valeur normalisée pour que leur surface soit proportionnelle à la similarité.

Pour la seconde étape, nous utilisons un ordre optimal du dendrogramme, c'est-à-dire un ordre qui minimise les différences entre individus voisins sommées sur le parcours de cet ordre. L'espace de recherche de ce minimum est donné par l'ensemble des permutations de l'arbre binaire des classes, mais un algorithme efficace d'exploration de cet espace est donné par Bar-Joseph et al. (2003).

Pour la troisième étape, l'homogénéité des classes est encodée par le niveau de gris du fond : le noir correspond à une classe dont les individus sont identiques, alors que le blanc correspond à une classe la moins homogène possible.

Pour la dernière étape, outre la rotation de la demi-matrice, on peut ajouter une déformation qui compresse la partie éloignée de la diagonale, car si l'ordre est bien choisi, elle ne présente que peu d'information. On peut aussi placer les étiquettes verticalement pour pouvoir densifier la représentation. Ces dernières transformations sont illustrées sur la Figure 8 qui présente la dendro-matrice d'un jeu de données comportant 143 individus : les auteurs ayant publié au moins 2 articles à la conférence InfoVis entre 2006 et 2011. Ces derniers sont caractérisés par un vecteur donnant pour tous les coauteurs possibles le nombre d'articles publiés ensemble. La distance utilisée pour effectuer la classification est le cosinus de l'angle de ces vecteurs.

2.2 Techniques d'interaction

Les dendro-matrices sont interactives : un ensemble de techniques d'interaction permettent de les manipuler.

2.2.1 Comparaison d'individus

Le survol par le curseur de la souris permet d'obtenir des information sur les individus. Lorsque le curseur se trouve sur une étiquette, celle-ci est mise en évidence, tout comme la ligne et la colonne les concernant (Figure 3 (a), mise en évidence de George Robertson). Par ailleurs, la similarité de cet individu avec les autres individus est rappelée en niveaux de gris par la ligne de rectangles située entre les étiquettes et la dendro-matrice.

Lorsque le curseur survole la dendro-matrice, sa position désigne deux individus dont les étiquettes ainsi que les lignes et colonnes les concernant sont mises en évidence (Figure 3 (b), mise en évidence de George Robertson et Jeffrey Heer). La ligne de rectangles située entre les étiquettes et la dendro-matrice est cette fois utilisée pour montrer la similarité des deux individus par le produit de leurs similarités avec les autres individus. Enfin, la classe la plus spécifique contenant ces deux individus est mise en évidence par le renforcement visuel de son pourtour et une mise en évidence des étiquettes de tous les individus de cette classe. Cette classe est alors la classe courante, et elle peut devenir l'objets d'autres interactions décrites ci-dessous.

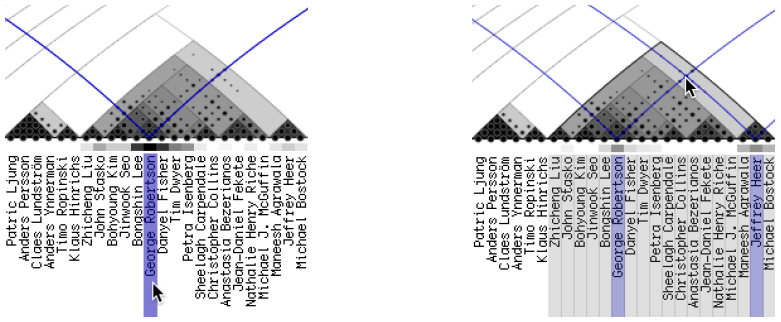
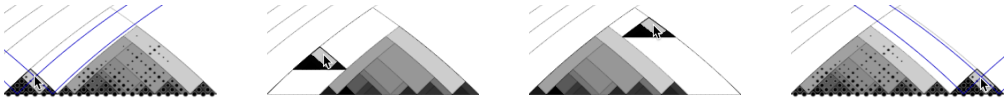


FIG. 3 – *Mise en évidence (a) d'un individu ; et (b) d'un couple d'individus.*

2.2.2 Modification de l'ordre

Une interaction de glisser-déposer permet de modifier l'ordre des individus utilisé pour la matrice en déplaçant la classe courante (Figure 4). L'ordre devant correspondre à une permutation de l'arbre issu de la CAH, le glisser-déposer est contraint : si la classe manipulée est le fils gauche (resp. droit) de sa classe parente, son glisser vers la droite (resp. gauche) est simple, il correspond à une permutation avec son frère. Dans le cas contraire, il faut remonter dans l'arbre pour trouver la première classe ancêtre pour laquelle la classe courante est incluse dans le fils gauche (resp. droit) et permuter alors les fils de celle-ci.

Durant l'interaction (Figure 4 (b) et (c)), les points donnant l'information de similarité individuelle ne sont pas affichés pour des raisons de performance. La position de la classe manipulée est contrainte pour rester dans son parent et ne pas empiéter sur son frère. Son déplacement prend donc la forme d'un saut : elle remonte vers la racine de son parent ; quand elle l'atteint, son frère passe au-dessous ; et enfin, elle peut redescendre de l'autre côté. Si l'utilisateur interrompt son glisser-déposer avant son terme, la fin du déplacement est animée :



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

FIG. 4 – *Interaction de glisser-déposer pour permuter des classes, (a) état initial, (b) et (c) états intermédiaires, (d) état final après permutation.*

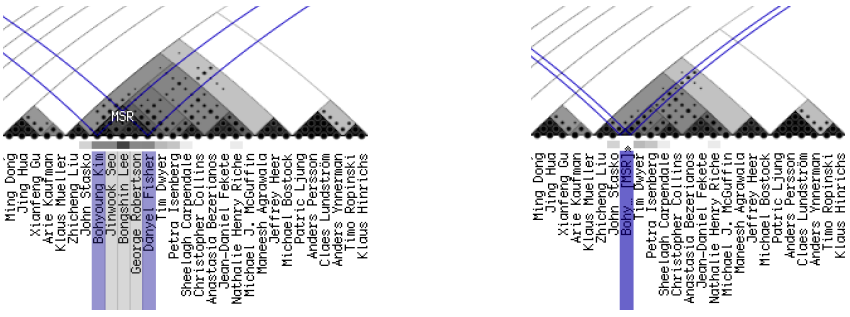
la classe revient à son point de départ si elle n'était pas encore passée par dessus son frère (Figure 4 (b)) ; sinon (Figure 4 (c)), elle poursuit son chemin de l'autre côté.

2.2.3 Agrégation de classes

Les classes peuvent être annotées : l'appui sur la touche entrée passe en mode édition de la classe courante, ce qui permet d'entrer ou d'éditer une étiquette pour cette classe. Cette étiquette est affichée à proximité de la racine de la classe (Figure 5 (a), la classe courante de 5 individus a été étiquetée "MSR"). Lorsqu'on clique sur une classe, elle est repliée et l'étiquette est alors utilisée pour la montrer parmi les individus (Figure 5 (b)). Les lignes et colonnes correspondantes sont alors remplacées par une information agrégée : celle de la distance aux individus en utilisant la méthode d'agrégation utilisée lors de la CAH.

Il faut noter que les classes, une fois repliées, se comportent comme des individus du point de vue des techniques d'interaction décrites précédemment : on peut donc comparer un individu à une classe repliée, ou deux classes repliées entre-elles.

Si on clique à nouveau sur un classe pliée, elle se déplie pour en révéler le détail. Ces transitions sont animées continûment pour que l'utilisateur puisse les comprendre. Enfin, l'étiquetage de classe comme le pliage/dépliage fonctionne récursivement : on peut plier/déplier/étiqueter des classes dont des descendants ont déjà été pliés/étiquetés.



(a) (b)

FIG. 5 – *Agrégation de classes : (a) étiquettage d'une classe ; et (b) repliement de cette classe.*

2.2.4 Sélection d'une partition

Un graphique montrant la correspondance entre nombre de classes et homogénéité celles-ci (qui correspond au niveau de coupure exprimée en terme de similarité dans une hiérarchie indexée classique) est affiché à côté de la visualisation (Figure 6, en haut à gauche). Il est interactif et permet de choisir par glisser-déposer soit une homogénéité (axe vertical), soit un nombre de classes (axe horizontal). La coupure résultante est matérialisée par une bordure plus épaisse doublée d'un halo qui sépare les classes plus homogènes que la coupure –situées sous la bordure– de celles moins homogènes –situées au-dessus (Figure 6).

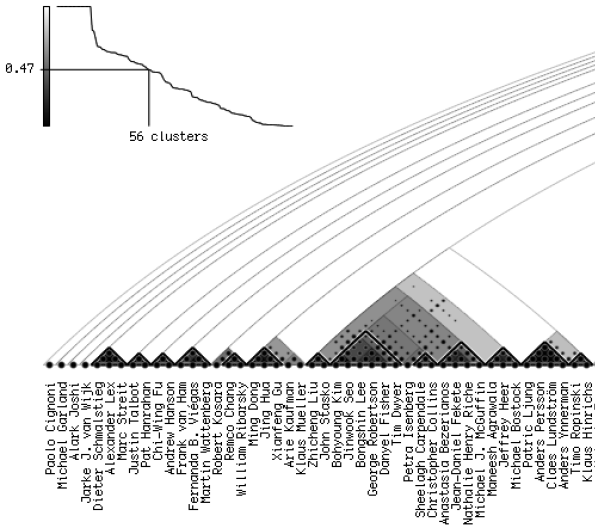


FIG. 6 – *Réalisation d'une coupure.*

2.3 Réalisation

Le prototype présenté est réalisé dans le langage Python [1]. La CAH est réalisés à l'aide de l'extension C hcluster [2]. Le calcul de l'ordre optimal est réalisé par nos soins en Python en utilisant l'algorithme de Bar-Joseph et al. (2003). Bien que celui-ci troque de la complexité en temps contre de la complexité en espace, il reste quadratique, ce qui le rend utilisable uniquement pour des jeux de données pas trop gros (le calcul de l'ordre optimal pour le jeu de données des 143 auteurs de la Figure 8 prend par exemple 1.2 secondes, alors qu'il ne prend que 0.36 secondes pour un jeu de données de taille 109). Cette complexité étant du même type que celle de la CAH, une réalisation dans un langage plus efficace repoussera la limite sur la taille des jeux de données, pour atteindre celle imposée par la CAH. Entre-temps, la dendro-matrice est utilisable avec n'importe quel ordre compatible avec la classification, et rien n'oblige à utiliser un ordre optimal.

1. Python, http://python.org/.
2. hcluster, http://pypi.python.org/pypi/hcluster.

Pour la partie graphique, le programme interactif utilise la bibliothèque OpenGL [3] et son interface pour Python PyOpenGL [4]. Les interactions sont réalisées à l'aide de GLUT [5], une boîte à outils portable de construction d'application OpenGL.

Les motifs de points utilisent une texture procédurale (*fragment shader*) appliquée à des quadrilatères pour obtenir le rendu le plus net possible. Les déformations non-linéaires de la matrice (compression des parties éloignées de la diagonale et repliement hiérarchique) sont réalisées à l'aide de programmes OpenGL (*vertex shaders*). La compression réalise la transformation suivante :

$$y' = \beta \log\left(1 + \frac{y}{\beta}\right) \tag{1}$$

où y (resp. y') est l'éloignement à la diagonale avant (resp. après) compression, et β un paramètre réglable interactivement (β petit (resp. grand) comprime beaucoup (resp. peu) la matrice).

Le repliement hiérarchique utilise une table qui associe à chaque ligne/colonne de la matrice un coefficient de repliement qui est établi après chaque interaction de dépliement/repliement : les individus d'une classe ayant 5 membres verront ce coefficient multiplié (resp. divisé) par 5 lorsque leur classe est repliée (resp. dépliée). Par accumulation, ce coefficient donne le facteur par lequel il faudra diviser la largeur de la ligne/colonne pour l'affichage.

Le programme permet également de générer une sortie SVG [6] grâce à laquelle ont été réalisées les figures de cet article.

3 Conclusion

Nous avons présenté les dendro-matrices, une visualisation interactive permettant de présenter et d'explorer les résultats de CAH. Par rapport à la visualisation classique à l'aide de dendrogrammes, les dendro-matrices conservent l'information de similaritié entre individus. Cette information peut permettre de mieux comprendre les algorithmes de classification en montrant par exemple les anomalies (individus similaires non classés ensembles).

Les dendro-matrices, tout comme les dendrogrammes, peuvent être juxtaposées à une *heatmap*. Leur manipulation permet alors de réordonner cette dernière. La Figure 7 montre cette juxtaposition pour un jeu de données d'identification de protéines à partir de peptides obtenues par spectrométrie de masse. Cette application est utile dans le domaine de la protéomique.

Nous comptons poursuivre nos travaux en appliquant les dendro-matrices à la co-classification, en nous appuyant sur les travaux de Bisson et Grimal (2011). Comme ci-dessus, l'idée est d'adjoindre des dendro-matrices sur les bords de la matrice de co-similarité pour manipuler ses lignes et ses colonnes. Nous comptons tester cette approche pour l'exploration des systèmes de gestion de version distribuée dans lesquels les contributeurs à un logiciel et les fichiers le constituant peuvent être mis en correspondance par les modifications atomiques stockées dans le système de gestion de version. Pour cela, nous comptons développer une version des dendro-matrices s'intégrant dans les navigateurs web afin de s'intégrer au mieux avec les outils existant dans ces domaines.

3. OpenGL, http://opengl.org.
4. PyOpenGL, http://pyopengl.sourceforge.net.
5. *The OpenGL Utility Toolkit* (GLUT), http://www.opengl.org/resources/libraries/glut/.
6. *Scalable Vector Graphics* (SVG), http://www.w3.org/TR/SVG/.

Par ailleurs, nous comptons enrichir les interactions proposées par les dendro-matrices, notamment en permettant de définir des zones d'intérêt et en faisant alors dépendre la place consacrée aux individus de ce degré d'intérêt.

Nous comptons enfin évaluer les bénéfices des dendro-matrices par rapport à la représentation classique des dendrogrammes par des expériences menées avec des utilisateurs experts d'un domaine utilisant la CAH.

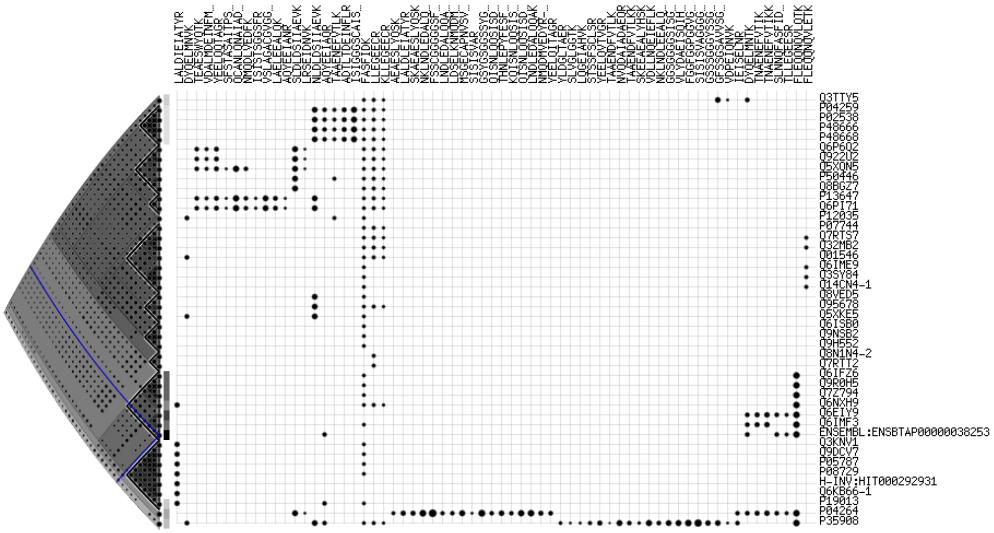


FIG. 7 – *Dendro-matrice appliquée à la classification d'identification de protéines par des peptides.*

References

Bar-Joseph, Z., E. D. Demaine, D. K. Gifford, A. M. Hamel, T. S. Jaakkola, et N. Srebro (2003). K-ary clustering with optimal leaf ordering for gene expression data. *Bioinformatics 19*(9), 1070–8.

Bisson, G. et C. Grimal (2011). Un cadre général pour les mesures de co-similarité. In *Actes de la conférence de la Société Francophone de Classification (SFC)*, Orléans, France, pp. 141–145.

Henry, N. et J.-D. Fekete (2007). Matlink: Enhanced matrix visualization for analyzing social networks. In *Proc. Interact'07*, pp. 288–302.

Henry, N., J.-D. Fekete, et M. J. McGuffin (2007). Nodetrix: Hybrid representation for analyzing social networks. *IEEE Trans. on Visualization and Comp. Graphics (Proc. InfoVis'07) 13*(6), 1302–9.

McGuffin, M. J. et J.-M. Robert (2010). Quantifying the space-efficiency of 2d graphical representations of trees. *Information Visualization (IVS) 9*(2), 115–140.

Rufiange, S., M. J. McGuffin, et C. P. Fuhrman (2012). Treematrix: A hybrid visualization of compound graphs. *Computer Graphics Forum (CGF) 31*(1), 89–101.

Ward Jr., J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association 58*(301), 236–244.

Wilkinson, L. et M. Friendly (2009). The history of the cluster heat map. *The American Statistician 63*(2), 179–184.

Summary

Clustering is often a first step when trying to make sense of a large data set. A wide family of cluster analysis algorithms, namely *hierarchical clustering* algorithms, does not provide a partition of the data set but a hierarchy of clusters organized in a binary tree, known as a *dendrogram*. The dendrogram has a classical node-link representation used by experts for various tasks.

We present *Dendrogramix*, a hybrid tree-matrix interactive visualization of dendrograms that superimposes the relationship between individual objects on to the hierarchy of clusters. *Dendrogramix* enables users to do tasks which involve both clusters and individual objects that are impracticable with the classical representation, like: to explain why a particular objects belongs to a particular cluster; to elicit and understand uncommon patterns (e.g., objects that could have been classified in a totally different cluster); etc.

Those sensemaking tasks are supported by a consistent set of interaction techniques that facilitates the exploration of large clustering results.

Visualisation alternative de classification ascendante hiérarchique

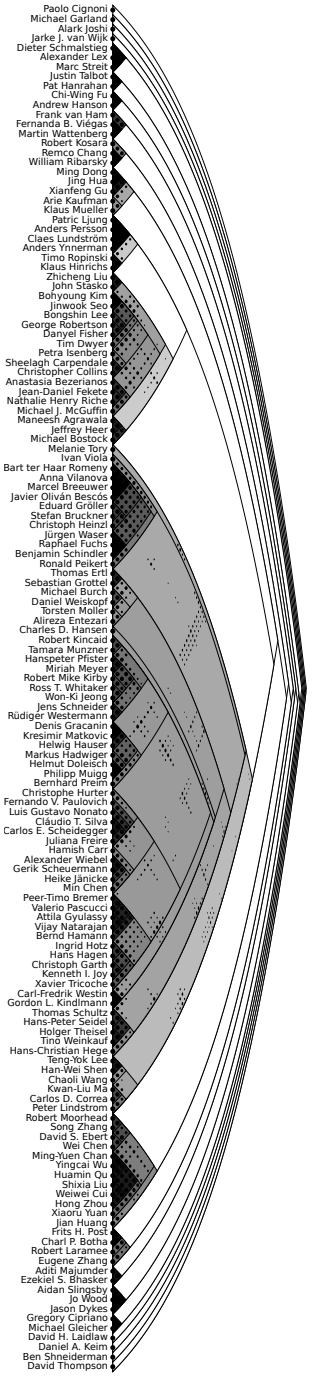


FIG. 8 – *143 auteurs ayant publiés plus de 2 articles à InfoVis entre 2006 et 2011, classés par similarité de coauteurs.*

Une interface 3D pour OLAP en réalité virtuelle (démonstration)

Sébastien Lafon*, Fatma Bouali**,*, Christiane Guinot*, Gilles Venturini*

* Université François-Rabelais de Tours, Laboratoire d'Informatique
64 avenue Jean Portalis, 37200 Tours, France
venturini@univ-tours.fr
sebastien.lafon@etu.univ-tours.fr
http://www.antsearch.univ-tours.fr
** Université de Lille2, IUT, Dpt STID
25-27 Rue du Maréchal Foch, 59100 Roubaix, France
fatma.bouali@univ-lille2.fr

Résumé. Nous présentons dans cet article une nouvelle interface visuelle et interactive pour explorer des cubes OLAP en réalité virtuelle. En premier lieu nous introduirons un état de l'art des visualisations en 3D de cubes OLAP où nous détaillons leurs avantages et leurs points faibles. Ensuite, nous détaillons notre approche, avec en particulier la représentation de plusieurs mesures et l'utilisation d'opérateurs OLAP directement dans la représentation 3D. Enfin nous exposons les résultats d'une évaluation utilisateur sur un ensemble de tâches ainsi que les conclusions qui en ont été tirées.

1 Introduction

OLAP (Online Analytical Processing), décrit notamment dans (Codd et al., 1993) (Chaudhuri et Dayal, 1997), est un ensemble d'outils permettant de réaliser une analyse multidimensionnelle de données volumineuses. Pour cela les données sont représentées selon plusieurs dimensions, chacune d'entres elles étant divisée en membres. Ces données sont généralement symbolisées par un cube ou hypercube OLAP subdivisé en cellules représentant une mesure au croisement de chaque dimension. OLAP met à disposition plusieurs opérateurs permettant d'interagir avec l'hypercube pour pouvoir ainsi préciser l'analyse. Ces opérateurs permettent de modifier l'apparence de l'hypercube pour l'adapter à ce que l'on souhaite visualiser, de naviguer à travers les hiérarchies des dimensions pour afficher plus ou moins de détails, et d'extraire des données utiles.

La majorité des représentations OLAP se font à l'aide de tableaux croisés dynamiques. En effet, comme précisé dans (Codd et al., 1993), avant OLAP la majorité des outils de visualisation de données utilisaient des tableaux, les utilisateurs sont donc habitués à ce type de représentation. C'est pourquoi beaucoup d'outils OLAP utilisent des tableaux croisés auxquels sont rajoutées des fonctionnalités spécifiques à OLAP (opérateurs, ...). Cependant cette représentation sous forme de tableaux n'est pas adaptée pour les données multidimensionnelles.

	roll-up	drill-down	slice	dice	switch	autre op.	Nb Mes.	Réal. virt.
DBMINER	ext.	ext.	ext.	ext.	non		2	non
DIVE-ON	int.	int.	int.	int.	non		1	oui
DIVA	ext.	ext.	int.	int.	non		1	non
Miner3D	ext.	ext.	ext.	ext.	non		1	non
VR4OLAP	int.	int.	int.	int.	int.	réorg.	2	oui

TAB. 1 – *Caractéristiques des interfaces 3D pour OLAP. Pour chaque opérateur, nous précisons s'il peut être externe à la visualisation 3D ("ext."), intégré à la visualisation 3D ("int.") ou non implémenté ("non"). Nous précisons également si d'autres opérateurs existent (comme la réorganisation par exemple), le nombre de mesures affichées simultanément ("Nb Mes.") ou encore si la réalité virtuelle est utilisée ("Réal. virt.").*

Comme décrit dans (Cuzzocrea et Mansmann, 2009), le problème de représentation de ce type de données vient justement de leur caractère multidimensionnel. En effet, OLAP permet d'effectuer une analyse en observant une ou plusieurs mesures représentées selon une ou plusieurs dimensions. La visualisation à l'aide de tableaux est adaptée lorsque les mesures sont définies par une ou deux dimensions. Dès que ce nombre de dimensions est supérieur, il est nécessaire de regrouper des dimensions sur les lignes ou les colonnes du tableau, ce qui rend l'analyse beaucoup plus complexe lorsque les données sont volumineuses.

D'autres interfaces ont donc été développées pour OLAP, et notre objectif est de contribuer à ce domaine en nous concentrant plus spécialement sur des visualisations 3D utilisant la réalité virtuelle (écran stéréoscopique, matériel d'interaction) à l'instar des travaux de (Ammoura et al., 2001). Nous avons cherché à savoir notamment si les développements matériels récents dans le domaine de la 3D stéréoscopique peuvent contribuer à ce type d'interfaces. La suite de cet article est organisée comme suit : la section 2 introduit un état de l'art des visualisations en 3D de cubes OLAP. La section 3 présente la visualisation OLAP que nous proposons, VR4OLAP. La section 4 aborde l'évaluation utilisateur réalisée pour tester notre application. Enfin la section 5 conclut sur les perspectives de ce travail.

2 Etat de l'art

Il existe plusieurs visualisations de cubes OLAP en dehors des tableaux croisés (voir par exemple (Mansmann et Scholl, 2008)), et nous nous intéressons ici à celles mettant en oeuvre des visualisations 3D (voir table 1).

Dans (Han et al., 1997) est décrit le système DBMiner qui propose un large panel de fonctions de fouille de données, dont OLAP. Les données OLAP dans DBMiner sont affichées en 3D dans une représentation à 3 axes avec des cubes de tailles différentes espacés les uns des autres afin de permettre une meilleure lisibilité des données se trouvant au centre du cube. Les membres de chaque dimension sont également affichés dans la visualisation. On peut choisir les dimensions sur chaque axe et jusqu'à deux mesures via une interface extérieure à la visualisation. L'une des deux mesures sera représentée par la couleur et l'autre par la taille du cube

affiché. Lorsque la souris reste assez longtemps sur un cube, les membres associés changent de couleur pour permettre à l'utilisateur de les situer, et une légende avec les valeurs du cube apparaît en haut de la visualisation. Si un clic est effectué sur un cube, un bandeau apparaît détaillant les valeurs du cube. Les opérateurs OLAP disponibles sont le roll-up, le drill- down, le slice, et le dice, exécutables via une fenêtre externe à la visualisation.

Dans (Ammoura et al., 2001), un logiciel de visualisation de données est introduit et celui-ci a la particularité de présenter les données dans un environnement en réalité virtuelle. Ce système se nomme DIVE-ON (Datamining in an Immersed Virtual Environment Over a Network) et permet d'immerger l'utilisateur dans la visualisation grâce à la projection de celle-ci tout autour de l'utilisateur en stéréoscopie. Cet environnement de réalité virtuelle utilisé s'appelle CAVE (Cave Automatic Virtual Environment), il place ainsi l'utilisateur entre 3 écrans (d'environs trois mètres sur trois) disposé en face de lui, à sa gauche et à sa droite. La réalité virtuelle permet à l'utilisateur d'avoir une activité sensori-motrice et cognitive dans un monde artificiel, ici c'est le déplacement dans le cube 3D et la possibilité d'exécuter des opérations pour modifier le cube. Ces actions se font à l'aide d'un gant permettant de suivre les mouvements de l'utilisateur, et d'un casque qui permet au logiciel de savoir où l'utilisateur regarde. Les données sont représentées sous forme de cubes ou de sphères espacés, de tailles et de couleurs différentes suivant les valeurs ou la nature des données. Une grille représentant la structure externe du cube est représentée afin de ne pas perdre l'orientation des données. Pour utiliser les opérateurs OLAP, l'utilisateur dispose d'un menu latéral à partir duquel il peut effectuer les opérateurs drill-down, roll-up, slice et dice. Ce menu latéral apparaît lorsque l'utilisateur appuie sur l'un des boutons du gant. De plus, grâce au casque que ce dernier porte, le menu est toujours affiché face à lui. L'utilisateur peut également pointer un objet et afficher grâce à un second bouton sur le gant une boîte de dialogue affichant des informations sur l'objet pointé. Un troisième bouton sur le gant permet de se déplacer dans la visualisation selon deux modes de déplacement : un mode fournissant une carte montrant à l'utilisateur où il se trouve et lui permettant de cibler sa destination, et un mode permettant à l' utilisateur de pointer directement dans la visualisation l'endroit qu'il veut atteindre.

Dans (Bulusu, 2003), l'outils DIVA (Data warehouse Interface for Visual Analysis) est présenté. Cet outil est dédié à la visualisation et l'analyse OLAP. DIVA est intégré à une interface Web, son but étant de fournir à l'utilisateur une interface légère et simple pour exécuter les requêtes et opérateurs OLAP de manière transparente. L'avantage de cette visualisation en 3D est que l'on peut se déplacer librement dans la scène pour analyser les données sous n'importe quel angle de vue. Les valeurs sont inscrites sur les cubes et l'effet de transparence sur les cubes permet une meilleure analyse. Cependant, l'intérieur du cube est invisible et la couleur des cubes est insignifiante. On peut noter l'effet de plan qui permet de rappeler l'orientation du cube. Les opérateurs OLAP disponibles sont le drill-down, le roll-up, le slice et le dice. Alors que les opérateurs slice et dice sont intégrés à la visualisation, le drill-down et le roll-up sont exécutés via un panneau en dehors de la visualisation. Ainsi par exemple, pour effectuer un roll-up, il faut choisir la dimension sur laquelle on veut agréger les données. Une fois la dimension choisie, dans la liste "FROM" du panneau on sélectionne de quelle hiérarchie on part, et on indique dans la liste "TO" le niveau de la hiérarchie que l'on souhaite atteindre. Il reste ensuite à cliquer sur le bouton "Roll Up" pour valider. L'exécution des opérateurs roll-up et drill-down n'est donc pas des plus intuitives.

Une autre solution intéressante est Miner3D (Miner3D, 2010), ce logiciel propriétaire per-

met de visualiser en 3D les données d'un dataWarehouse. Le cube peut être construit selon différentes visualisations : une visualisation bars chart, qui représente les données sous forme de rectangles dont la taille varie en fonction de la valeur de la mesure, une visualisation représentant les données sous forme de cubes ou sous forme de sphères dont la taille et la couleur varient en fonction de la valeur,... Il est possible d'afficher jusqu'à 5 dimensions et de visualiser dynamiquement les mesures au cours d'une période. Le logiciel va ainsi afficher à intervalle régulier le cube dont les valeurs des mesures correspondent à l'intervalle actuel. Lorsque l'on passe la souris sur une donnée, la valeur de la mesure correspondante s'affiche. Il est possible d'exécuter les opérateurs drill-down, roll-up, slice et dice.

Pour synthétiser (voir table 1), dans les visualisations présentées ici, il n'y en a aucune qui implémente beaucoup d'opérateurs OLAP, on trouve le plus souvent le slice, le dice et le drill-down. La mise en place de plus d'opérateurs permettrait à l'utilisateur de mieux naviguer dans le cube. De plus les opérateurs gagneraient en intuitivité s'ils étaient directement incorporés dans la visualisation, comme par exemple l'opérateur slice dans l'outil DIVA. Concernant les membres, toutes les visualisations présentées les affichent ce qui permet à l'utilisateur de savoir facilement à quoi correspondent les données qu'il analyse. Il est également intéressant d'afficher les valeurs des mesures lorsque l'on clique sur une donnée comme dans la plupart des visualisations, pour avoir ainsi la valeur précise des données qu'on observe. Dans les outils DBMiner et DIVE-ON, les données affichées sont espacées pour éviter que celles au premier plan n'occluent pas celle aux plans suivants. Enfin, l'affichage en stéréoscopie utilisé dans DIVE-ON est intéressant car il permet à l'utilisateur de mieux percevoir la profondeur. Enfin, à notre connaissance, aucune évaluation utilisateur n'a été réalisée sur ces visualisations pour valider l'efficacité de celles-ci.

Toutes ces remarques nous ont aidées à concevoir une nouvelle visualisation, VR4OLAP (Virtual Reality for OLAP), afin de proposer à l'utilisateur un environnement 3D et interactif le plus complet possible (voir la dernière ligne de la table 1) pour explorer des données OLAP mais aussi les présenter à d'autres personnes.

3 Visualisation OLAP proposée : VR4OLAP

3.1 Choix des données à visualiser

Les données à visualiser dans notre outil sont gérées à l'aide du serveur OLAP Mondrian, un serveur Open Source se présentant sous la forme d'une librairie Java. Mondrian fait partie de la catégorie des serveurs R-OLAP, c'est-à-dire qu'il permet d'accéder à des données contenues dans une base de données relationnelle classique qui est structurée pour réagir comme une base OLAP. Il exécute des requêtes écrites avec le langage MDX pour récupérer les données. Pour fonctionner, il faut fournir au serveur OLAP Mondrian un fichier XML décrivant le schéma multidimensionnel de la base de données sur laquelle le serveur se connecte. C'est dans ce fichier que sont définis les différents cubes disponibles, les dimensions associées ainsi que leur hiérarchie, et les mesures pouvant être visualisées. Mondrian permet de se connecter à de nombreuses bases de données différentes. Actuellement notre application permet de se connecter à des bases Access et MySQL. base de données pour exécuter une requête MDX.
Une fois la connexion à la base réussie, la fenêtre de choix du cube apparaît sous la forme d'une

interface 2D. Cette fenêtre permet à l'utilisateur de choisir les données qui seront affichées en 3D (cube choisi, dimensions pour ce cube et une ou deux mesures).

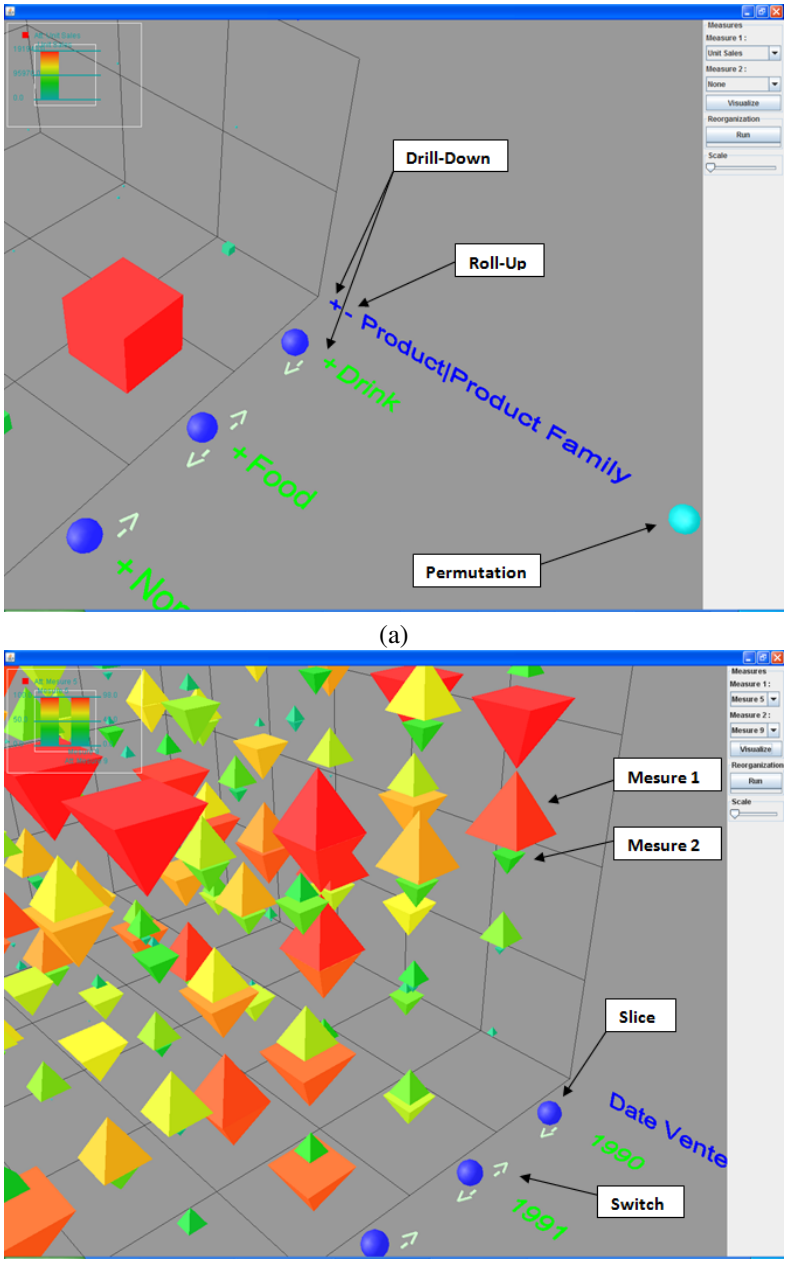
3.2 Définition de la visualisation 3D

La visualisation 3D est définie de la manière suivante (voir figure 1). Trois axes sont utilisés pour représenter les trois dimensions choisies. Chacun de ces axes porte le nom de la dimension qui lui correspond, ainsi que les noms des membres de cette dimension. Les membres affichés pour une dimension dépendent des opérations de développement réalisées sur cette dimension (voir la section suivante). Ensuite, si une seule mesure a été sélectionnée, les valeurs de cette mesure sont représentées par un cube dont la taille et la couleur dépendent de la mesure. Si deux mesures ont été choisies, alors chaque cellule du cube est représentée par un pyramidion (voir figure 1(b)). Cet objet 3D est composé de deux pyramides dont la hauteur et la couleur dépendent de chacune des valeurs des deux mesures choisies. En plus de ces éléments, une grille est affichée pour aider l'utilisateur à mieux situer les données par rapport aux membres des dimensions. Mentionnons également qu'il est possible, via une fenêtre 2D externe à la visualisation, de changer toutes les couleurs de la scène 3D, ce qui permet à l'utilisateur de personnaliser la visualisation de ses données. L'utilisateur peut mettre en route ou non l'affichage stéréoscopique, et d'un point de vue matériel, l'environnement de visualisation stéréoscopique peut être un grand écran 3D immersif ou bien un écran LCD 3D.

3.3 Interactions

Nous avons ensuite représenté les opérateurs OLAP et autres interactions dans cette visualisation à l'aide de signes visuels supplémentaires sur lesquels l'utilisateur peut directement interagir (voir figure 1). Suivant l'interaction sélectionnée, le système génère la requête MDX correspondante et regénère dynamiquement un nouveau cube.

Une des interactions les plus simples consiste, lorsque l'on clique sur une donnée avec la souris, à faire apparaître une fenêtre "pop-up" qui affiche la valeur de la mesure et les membres correspondants à la donnée cliquée. Ensuite viennent les opérateurs OLAP. L'opérateur drill-down est représenté dans la visualisation par le symbole "+". Ce symbole se trouve sous les noms des membres et sous le nom des dimensions. Lorsque l'on clique sur un "+" se trouvant sous un membre, un drill-down est effectué sur ce membre et lorsque l'on clique sur un "+" sous le nom d'une dimension, alors un drill-down est effectué sur chaque membre de cette dimension. De plus, lorsqu'un membre ou une dimension ne permet pas de faire un drill-down alors le symbole est masqué. De manière similaire, l'opérateur roll-up est représenté par un symbole "-" sous les noms des dimensions. Lorsque l'on clique dessus, on remonte d'un cran dans la hiérarchie de la dimension correspondante. De même, lorsqu'une dimension ne permet pas de faire un roll-up alors le symbole est masqué. Les opérateurs de sélection Slice et Dice sont représentés par des sphères se trouvant à côté des noms des membres des axes. Lorsque l'on clique dessus, seules sont affichées les données liées au membre correspondant à la sphère cliquée. Les autres données sont masquées, cependant les noms des autres membres du même axe et les sphères correspondantes sont toujours affichés pour que l'utilisateur puisse également les sélectionner s'il le souhaite. Pour désélectionner un membre, il suffit de recliquer sur la sphère correspondante. L'opérateur de permutation switch est représenté dans la visualisation par les symboles "->" et "<-" à côté des noms des membres. Lorsque l'on clique dessus, le



(a)

(b)

FIG. 1 – *Visualisation de une (a) ou deux (b) mesures*

membre correspondant est permutée avec sa voisine de gauche ou de droite (selon le sens de la flèche). Il est également possible de permuter deux dimensions en cliquant sur la sphère se trouvant à côté du nom de la troisième dimension.

D'autres interactions sont proposées à l'utilisateur. A droite de la visualisation se trouve un bandeau à partir duquel l'utilisateur peut effectuer différentes opérations. Il peut tout d'abord changer les mesures affichées, et le système répond dynamiquement en modifiant la visualisation. L'utilisateur dispose d'un curseur pour faire varier dynamiquement la taille des cubes ou pyramidions afin de mieux les voir s'ils sont trop petits. Ensuite, l'utilisateur peut lancer un algorithme de réorganisation. En effet, dans nos travaux précédents (Sureau et al., 2009) nous nous sommes intéressés à la réorganisation d'une dimension afin de placer côte à côte des membres ayant des valeurs de mesures similaires. Cet algorithme était cependant limité à une réorganisation linéaire des dimensions et ne pouvait tenir compte d'éventuelles hiérarchies. Le nouvel algorithme utilisé dans VR4OLAP (qui par manque de place ne sera pas décrit ici, voir (Lafon et al., 2012)) se sert d'un algorithme génétique pour réorganiser les membres de chaque dimension tout en respectant leurs hiérarchies. Il réalise donc des permutations d'arbres et de sous arbres afin de maximiser une mesure de lisibilité du cube. Plusieurs versions de cet algorithme sont utilisables (réorganiser la hiérarchie telle qu'elle est affichée et développée, réorganiser toute la hiérarchie, réorganiser niveau par niveau en agrégeant les mesures). Son exécution est incrémentale, par pas durant moins de 1 minute. L'utilisateur peut donc cliquer une première fois pour obtenir un résultat et donc un nouveau cube, et s'il souhaite continuer et améliorer ce résultat, il peut cliquer à nouveau. Pendant l'exécution de l'algorithme, il peut continuer à utiliser la visualisation. Ainsi, l'utilisation de l'algorithme génétique ne vient pas rallonger outre mesure l'utilisateur dans son exploration du cube. Les résultats sont visuellement très intéressants (voir l'évaluation utilisateur). Enfin, les déplacements dans la visualisation se font soit à l'aide du clavier et de la souris, soit à l'aide d'un SpacePilot (souris à 6 degrés de liberté).

4 Evaluation utilisateur

4.1 Présentation

Afin de cerner l'efficacité et les voies d'amélioration de notre visualisation, et aussi de la comparer à une approche concurrente, nous avons réalisé une évaluation utilisateur selon le protocole suivant. Tout d'abord nous avons recruté des utilisateurs/testeurs ayant déjà des connaissances en OLAP. 7 utilisateurs ont ainsi été choisis au département STID (STatistique et Informatique Décisionnelle) de l'IUT de l'Université de Lille 2. Nous avons sélectionné alors différentes interfaces à tester : VR4OLAP en 3D-monoscopique, VR4OLAP en 3D-stéréoscopique, VR4OLAP en 3D-stéréoscopique avec réorganisation lorsque cette dernière a un sens, et une méthode classique appelée JPivot. Cet outil 2D est un logiciel open-source disposant d'une interface web et représentant les cubes OLAP à l'aide de tableaux croisés dynamiques. JPivot permet d'effectuer de façon interactive plusieurs opérateurs OLAP (drill-down, roll-up, slice, dice, ...). Il affiche la valeur des mesures directement sous la forme d'un nombre.

Nous avons défini un ensemble de questions/tâches représentatives afin de cerner les points forts et points faibles de ces visualisations en termes de fouille de données. Voici les questions qui ont été posées :

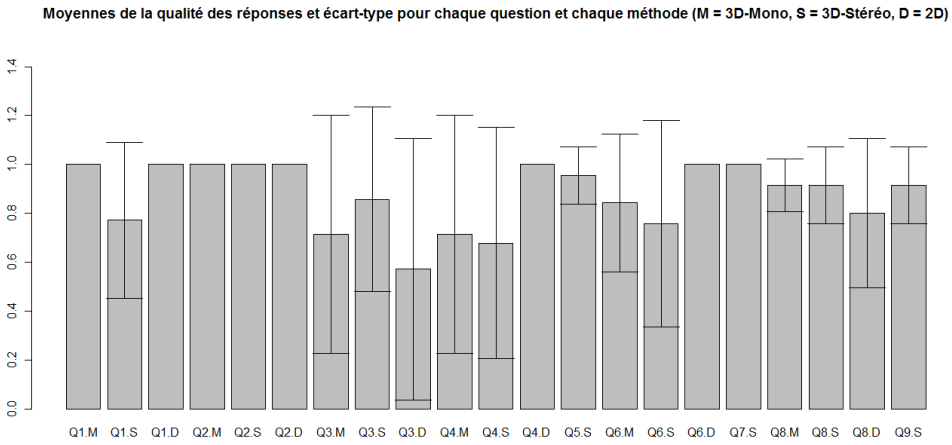
- Q1 : trouver une valeur du cube pour des attributs donnés. Pour cela, l'utilisateur partira d'un cube non développé et devra pour aller chercher une valeur cible en utilisant les opérateurs drill-down et roll-up.
- Q2 : afficher un certain cube à partir d'un autre le contenant (utilisation des opérateurs de sélection slice et dice).
- Q3 : trouver, dans un cube contenant deux mesures, la cellule dans laquelle ces deux mesures sont égales.
- Q4 : trouver dans une dimension du cube deux membres âyant le même comportement vis à vis de la mesure (2 "tranches" du cube identiques).
- Q5 : même question que la précédente (deux membres égaux) mais en s'aidant avec la réorganisation (VR4OLAP stéréo uniquement).
- Q6 : trouver dans un cube un membre d'une dimension plus atypique que les autres (les membres appartiennent à des classes, les membres d'une même classe ont les mêmes valeurs, et le membre atypique est celui n'appartenant à aucune classe).
- Q7 : même question que la précédente (trouver le membre atypique) mais en s'aidant de l'algorithme de réorganisation (VR4OLAP stéréo uniquement).
- Q8 : trouver le nombre de classes des membres d'une dimension.
- Q9 : même question que la précédente (trouver le nombre de classe de membres d'une dimension) mais en s'aidant de la réorganisation (VR4OLAP stéréo uniquement).

Nous avons utilisé plusieurs cubes de données : pour les deux premières questions, la base utilisée est celle fournie avec Mondrian (base Access MondrianFoodMart), et pour les autres questions nous avons utilisé une base Access remplie par nous même spécialement pour l'évaluation. Une randomisation a lieu afin d'éviter les effets d'apprentissage (plusieurs fois le même cube, ou le même ordre de test 2D-3D, ou les questions dans le même ordre, etc).

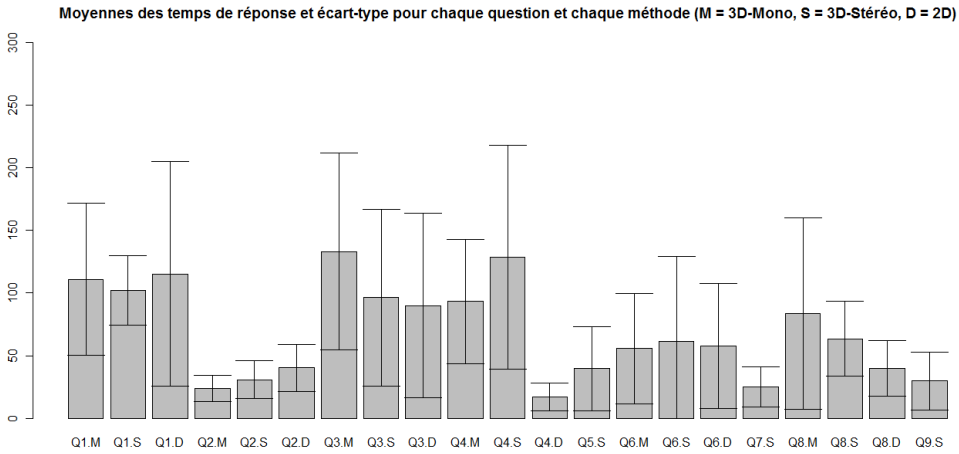
A chaque question et pour chaque méthode testée, la réponse donnée par l'utilisateur est notée pour pouvoir plus tard la comparer avec la valeur attendue et ainsi mesurer la qualité de la réponse sous la forme d'une mesure de similarité avec la bonne réponse ($\in [0, 1]$). De plus, pour chacune des questions, le temps de réponse a été mesuré. Un questionnaire préalable permet de connaître le niveau de la personne en Informatique, en 3D et en OLAP. Un questionnaire final permet de connaître les impressions "à chaud" de l'utilisateur. Avant de répondre aux questions, nous laissons l'utilisateur interagir avec les visualisations dans le but qu'il se familiarise avec elles et obtienne les compétences nécessaires pour répondre aux questions posées par la suite.

4.2 Résultats et discussion

Les résultats des questions Q1 à Q9 sont présentés figure 2. Nous pouvons en retirer les analyses et conclusions suivantes. En ce qui concerne les opérateurs OLAP (questions Q1 et Q2), les utilisateurs répondent avec une qualité comparable pour la 2D et la 3D-mono, mais la qualité baisse un peu en 3D-stereo. Les temps sont un peu plus courts pour la 3D. L'implémentation des opérateurs OLAP dans la représentation 3D semble globalement aussi efficace que celle de la 2D, ce qui est encourageant pour notre approche si l'on considère que les utilisateurs sont souvent plus habitués à la 2D qu'à la 3D.



(a)



(b)

FIG. 2 – *Représentation des réponses des utilisateurs (qualité en haut, temps en bas).*

Question	2D-mono	3D-mono	3D-stéréo
Facile à utiliser	3.5 (0.9)	3.0 (0.8)	2.8 (0.6)
Se repérer dans les données	3.2 (0.4)	3.8 (0.8)	3.4 (1.1)
Attractif	1.7 (0.4)	3.7 (0.4)	4.0 (0.8)
Adapté pour présentation	2.8 (0.8)	3.8 (1.0)	2.8 (1.0)

TAB. 2 – *Réponses des utilisateurs à différentes questions (scores de 1 à 5, moyenne sur 7 utilisateurs, écarts-types entre parenthèses).*

Pour la représentation de deux mesures (question Q3), on note que les meilleurs temps de réponses sont obtenus en 2D, mais par contre la qualité de la réponse est meilleure pour la 3D que la 2D. La représentation de deux mesures sous forme de pyramidion possède donc un avantage par rapport à la représentation 2D sous forme de deux nombres. Un attribut visuel "taille" permet plus facilement la comparaison entre deux valeurs (Mackinlay, 1986).

Pour la question Q4, les résultats sont nettement en faveur de la 2D, aussi bien en qualité que pour le temps de réponse. En effet, nous avons pu observer que le fait de représenter explicitement la valeur sous la forme d'un nombre simplifie cette tâche par rapport à la 3D. Dans la représentation 3D, la comparaison entre valeurs éloignées spatialement les unes des autres est difficile sur la base de la couleur et de la taille des cubes, par rapport à des chiffres qui doivent se mémoriser plus facilement. Egalement, il peut y avoir des occlusions en 3D (mais en principe, nous avons remarqué que celles-ci sont limitées car les cotés des cubes aident beaucoup à résoudre cette tâche).

Pour la détection de membre atypique (Q6), les temps de réponse sont assez comparables pour cette tâche, avec un avantage net en qualité pour la 2D grâce à la lecture directe des valeurs sous forme de nombre. Pour la détection du nombre de classes (Q8), les réponses sont meilleures en 3D qu'en 2D, mais les temps sont plus courts en 2D. Nous avons observé le même phénomène que pour la question Q4 : en 2D, les utilisateurs se contentent d'observer la tête de colonne pour déterminer le nombre de classes. Nous aurions du complexifier cette tâche en créant des classes pas totalement homogènes.

En ce qui concerne la réorganisation (questions Q5, Q7 et Q9), on observe que celle-ci égalise ou améliore les performances de la 3D-stéréo sans réorganisation. La réorganisation augmente la qualité des réponses et diminue le temps nécessaire pour répondre (souvent divisé par 2), même par rapport à la 2D (questions Q7 et Q9). Le fait de faire apparaître des régularités dans la visualisation augmente de beaucoup la faculté d'analyse (un sujet largement débattu en réorganisation linéaire de matrices par exemple (Bertin, 1977)). Les individus atypiques apparaissent nettement mieux, de même que les classes existantes dans les valeurs visualisées.

Enfin, nous rapportons les informations obtenues via le questionnaire informel final (voir table 2). On constate que la 3D et la 2D obtiennent des notes comparables en ce qui concerne le repérage de l'utilisateur au sein des données et la facilité d'utilisation. Compte tenu de la nouveauté de la visualisation 3D, on aurait pu s'attendre à une notation défavorable pour notre approche mais ce n'est pas le cas. En particulier, l'apprentissage de l'utilisation du SpacePilot est difficile au début, même si nous savons par expérience que ce périphérique est très efficace une fois maîtrisé. Plus précisément, dans la phase d'apprentissage "libre", les utilisateurs ont passé trois fois plus de temps à tester le SpacePilot que le clavier (358 secondes en moyenne

contre 156 secondes). Nous avons noté aussi que les déplacements clavier-souris pouvaient être améliorés avec un zoom plus rapide en 3D. Malgré cela, la 2D et la 3D sont notées sur ces points de manière équivalente. On note aussi que les utilisateurs trouvent VR4OLAP beaucoup plus attractif que l'approche 2D, et qu'ils l'utiliseraient plus volontiers que la 2D pour présenter des résultats. Ce point peut être important si l'on considère qu'OLAP est aussi utilisé pour présenter les conclusions d'une analyse à des décideurs. On constate aussi que les utilisateurs ont souvent moins bien noté la 3D-stéréo que la 3D-mono, ce qui traduit les difficultés qu'ils ont pu rencontré (adaptation, fatigue, etc).

En conclusion de cette étude, il faut retenir les points suivants : des performances équivalentes ont été observées entre 2D et 3D, notamment dans la navigation dans le cube avec les opérateurs OLAP, ou encore la visualisation de deux mesures. Ce point ne peut donc prouver la supérioté de la 3D-mono sur la 2D, ou encore la 3D-stereo sur la 2D, ou l'inverse. Néanmoins, il faut rappeler que les utilisateurs sont peu familiers avec la 3D. Avec un apprentissage plus long (plusieurs séances), il serait peut être plus facile de montrer l'apport de la 3D en général (ou inversement), mais cela compliquerait beaucoup le protocole qui ne serait peut être plus acceptable pour des utilisateurs bénévoles. Ensuite, des performances meilleures en 2D sont observées notamment en détection de similarité. Pourtant, la 3D-mono est souvent considérée comme plus attractive que la 2D. Egalement, la réorganisation apporte un gain net dans la résolution des tâches liées à la similarité. Enfin, la 3D-stereo ne s'est pas montré meilleure en performance que la 3D-mono. Nous pensions qu'avec du matériel récent et plus accessible que le CAVE utilisé dans (Ammoura et al., 2001) (et qui n'avait pas été évalué par des utilisateurs) une différence aurait pu apparaître, mais cela n'a pas été le cas.

5 Conclusion

Nous avons présenté dans cet article VR4OLAP, une nouvelle interface pour OLAP. Les principaux éléments qui la caractérisent sont une représentation 3D, l'inclusion d'un grand nombre d'opérateurs OLAP sous la forme d'objets clicquables, l'utilisation d'un algorithme de réorganisation et enfin la visualisation sur un écran stéréoscopique avec du matériel d'interaction. Nous avons aussi présenté les résultats d'une évaluation utilisateur, ce qui est nouveau à notre connaissance pour ce type d'interface. Ces résultats n'ont pas permis de montrer un avantage de la 3D-stéréo sur la 3D-mono. Ils laissent plus d'ouverture sur la comparaison 2D-3D si l'on considère le manque d'expérience des utilisateurs en 3D par rapport à la 2D. Cependant, l'engouement des utilisateurs pour la 3D est motivant et souligne le défi représenté par les interfaces 3D : faire correspondre à cet engouement une facilité d'utilisation et une efficacité dans la résolution des tâches.

Outre les perspectives suggérées par nos testeurs, nous souhaitons développer à la fois le coté visualisation et le coté interaction de VR4OLAP. Dans le premier cas, nous allons ajouter des représentations plus complexes, avec par exemple des images (placées sur les cubes) pour représenter des informations supplémentaires sur les données (par exemple, photos de produits), et nous allons étudier également comment représenter plus de deux mesures. Pour les interactions, nous sommes en train de mettre en place des opérations de sélection de données, afin de proposer ensuite de nouveaux résultats à l'utilisateur (par exemple ne garder que le cube qui correspond aux données sélectionnées, ou encore préciser des informations sur ces données, etc).

Références

Ammoura, A., O. Zaïane, et R. Goebel (2001). Towards a novel olap interface for distributed data warehouses. *Data Warehousing and Knowledge Discovery, Third International Conference, DaWaK 2001 2114*, 174–185.

Bertin, J. (1977). La graphique et le traitement graphique de l'information. *Nouvelle Bibliothèque Scientifique.*.

Bulusu, P. (2003). Diva-data warehouse interface for visual analysis. master thesis. university of florida.

Chaudhuri, Q. et U. Dayal (1997). An overview of data warehousing and olap technology. *ACM SIGMOD Record 26*, 65–74.

Codd, E., S. Codd, et C. Salley (1993). Providing olap to user-analysts: An it mandate. Technical report, E.F. Codd and Associates.

Cuzzocrea, A. et S. Mansmann (2009). Olap visualization: models, issues, and techniques. *Encyclopedia of Data Warehousing and Mining, 2nd ed.*, 1439–1446.

Han, J., J. Chiang, S. Chee, J. Chen, Q. Chen, S. Cheng, W. Gong, M. Kamber, K. Koperski, G. Liu, Y. Lu, N. Stefanovic, L. Winstone, B. Xia, O. R. Zaiane, S. Zhang, et H. Zhu (1997). Dbminer: A system for data mining in relational databases and data warehouses. *CASCON'97: Meeting of Minds*, 249–260.

Lafon, S., F. Bouali, C. Guinot, et G. Venturini (2012). RÃl'organisation hiÃl'rarchique de visualisations dans olap. *Revue des Nouvelles Technologies de l'Information Extraction et Gestion des Connaissances, RNTI-E-23*, 287–298.

Mackinlay, J. (1986). Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics 5*, 110–141.

Mansmann, S. et M. H. Scholl (2008). Visual olap: a new paradigm for exploring multidimensional aggregates. In *IADIS International Conference Computer Graphics and Visualization 2008*, pp. 59–66.

Miner3D (2010). http://www.miner3d.com/products/visual_olap.html.

Sureau, F., F. Bouali, et G. Venturini (2009). Optimisation heuristique et génétique de visualisations 2d et 3d dans olap : premiers résultats. *RNTI, 5ème journées francophones sur les entrepôts de données et l'analyse en ligne (EDA'09)*, 62–75.

Summary

In this paper we present a new visual and interactive interface for OLAP in virtual reality. First of all, we introduce existing approaches for vizualizing OLAP cubes in 3D, their advantages and their drawbacks. Then we present our approach, and especially the representation of several measures and the use of OLAP operators within the 3D representation. Finally, we detail the usability tests of our visualization and the derived conclusions.

La visualisation de traces, support à l'analyse, déverminage et optimisation d'applications de calcul haute performance

Damien Dosimont[*,**], Guillaume Huard[*,***], Jean-Marc Vincent[*,***]

[*]prénom.nom@imag.fr
[**]INRIA
[***]Université Joseph Fourier, Grenoble I

Résumé. L'analyse du comportement d'applications logicielles est une tâche de plus en plus difficile à cause de la complexité croissante des systèmes sur lesquels elles s'exécutent. Alors que l'analyse des systèmes embarqués doit faire face à une pile logicielle complexe, celle des systèmes parallèles doit être capable de s'adapter à l'envergure de leur architecture matérielle et à leur indéterminisme. La visualisation de traces obtenues lors du déroulement des applications s'exécutant sur ces plate-formes s'est répandue dans les outils d'analyse pour faire face à ces problématiques. Il existe aujourd'hui un large éventail de techniques qui se distinguent par la quantité d'informations, l'échelle des systèmes, ou les comportements qu'elles sont capables de représenter. Nous nous proposons d'en faire un état de l'art, en discutant des méthodes de visualisation statistiques, comportementales et structurelles de l'application, et des techniques permettant le passage à l'échelle de l'analyse.

1 Introduction

L'analyse du comportement d'applications logicielles à des fins de compréhension, de déverminage et d'optimisation est une tâche de plus en plus difficile, en raison de la complexité croissante des systèmes sur lesquels elles sont exécutées. Dans le cas des systèmes embarqués et des ordinateurs personnels à processeurs multi-coeurs, évoqué par Herlihy et Shavit (2008), les principales difficultés sont liées à la pile logicielle complexe, car l'analyse concerne aussi bien les aspects bas-niveau proches du matériel, que les couches intergicielles abstrayant celui-ci (Campbell et al., 1999), ou que l'applicatif. Les systèmes parallèles, et en particulier les systèmes dédiés au calcul, se voient quant à eux distribués dans le monde, à travers une hiérarchie complexe (processeur, machine, grappe, site) et de plus en plus souvent hétérogène (Hwang et al., 2012). L'échelle de ces systèmes est contraignante pour les outils d'analyse. Pour tirer au mieux parti des multiples unités de calcul, les applications utilisent des bibliothèques logicielles, comme OpenMP ou MPI (Quinn, 2004) permettant de simplifier leur parallélisation. Cependant, elles se caractérisent par un fort indéterminisme inhérent au paradigme de la programmation parallèle (condition de concurrences d'accès aux ressources, ordre des messages, interblocages), évoqué par Chassin de Kergommeaux et al. (2001), et pour lesquelles les méthodes d'analyse traditionnelles ne sont pas suffisantes.

Parmi les solutions les plus fréquemment employées pour l'analyse d'un système informatique figure le traçage. Tracer consiste à stocker, sous forme de fichiers, le résultat de l'exécution de l'application en enregistrant des informations sur l'état de l'exécution et des mesures sur l'état de la plate-forme issues de la récupération de valeurs contenues dans des compteurs matériels ou fournies par l'applicatif. Pour ce faire, le code de l'application est instrumenté à l'aide de bibliothèques logicielles, d'outils de traçage, ou simplement de fonctions d'affichage ou d'écriture dans des fichiers. Les éléments à tracer (fonction appelée, état d'une entité, commutation de contexte, interruption, valeur d'une variable, événement particulier) sont déterminés par l'analyste ou par l'outil. La trace est alors générée au fur et à mesure de l'exécution, sous la forme d'un format particulier (générique ou standardisé, textuel ou binaire selon les outils utilisés). Parmi les formats de traces les plus répandus, on peut citer OTF (Knüpfer et al., 2006) utilisé par des outils de visualisation de traces comme Vampir (Knüpfer et al., 2008), le format de trace de Tau (Shende, 2006), celui de l'outil de visualisation Pajé (Chassin de Kergommeaux, 2000), ou CTF, spécifié par Desnoyers (2012). Le traçage concerne aussi bien le monde des architectures largement distribuées, avec notamment des infrastructures comme Tau ou des outils comme Score-P (Score-P Project, 2012), que celui des systèmes de plus petite échelle (ordinateurs personnels, systèmes embarqués). On citera par exemple LTTng, patch pour Linux permettant de tracer les événements au niveau noyau et *user-space* (Toupin, 2011; Fournier et al., 2009), ou KPTrace (Prada-Rojas et al., 2009), proposé par STMicroelectronics pour le débogage de ses plate-formes.

Les principales difficultés dues au traçage sont posées par le volume des fichiers obtenus, lié à la durée d'exécution et au nombre d'événements. Cela rend fastidieux le suivi de l'évolution de l'application au cours du temps, nécessaire à cause de son indéterminisme. Différentes méthodes de traitement (filtrage, élimination du bruit, agrégation, fouille de données, reconnaissance de motifs) ont pour but de simplifier la trace et de la structurer, tandis que d'autres (intégration dans une base de données) permettent une gestion optimisée de son stockage. En bout de chaine, des représentations visuelles de la trace sont fréquemment employées. L'émergence de ces outils date des années 1970, pour faire face aux problématiques liées à l'analyse des systèmes parallèles, et ont vu jusqu'à nos jours l'intégration progressive de différentes techniques de visualisation provenant de domaines variés comme les statistiques, la gestion de projet ou encore le stockage de données. Le choix de ces méthodes repose sur la pertinence qu'elles apportent à l'analyste quant à sa compréhension de l'application, aussi bien au niveau de l'évolution de son comportement au cours du temps que de sa structure, et des relations qui lient ses différents composants.

Nous nous proposons, dans ce papier, de faire un état de l'art des méthodes de visualisations appliquées pour l'analyse de traces. Nous suivrons une progression dans le détail qui consiste à commencer par une analyse des caractéristiques générales de l'application à l'aide d'une synthèse globale s'appuyant sur des visualisations statistiques. Elle se poursuit par une analyse détaillée de l'exécution de la trace, à travers des représentations qui font intervenir un axe temporel et montrent les séquences d'événements se produisant dans la trace, mais aussi à travers des représentations structurelles, favorisant l'observation des interactions entre les composants de l'application, leur hiérarchie et leur topologie. Nous évoquerons ensuite les solutions proposées, à travers des extensions des techniques précédentes, afin de permettre le passage à l'échelle de l'analyse. Pour finir, nous conclurons par une synthèse des concepts évoqués, puis par des perspectives en vue d'étendre les fonctionnalités de ces procédés.

2 Synthèse globale

La synthèse globale consiste en une représentation d'informations contenues dans la trace traitées par des opérateurs statistiques. Leur utilisation est une approche pertinente pour débuter une analyse, selon la méthodologie proposée par Shneiderman (1996), par exemple (*overview, zoom and filter, then details on demand*). Parmi les représentations possibles, les formes textuelles sont couramment utilisées, mais des méthodes visuelles faisant appel à des diagrammes, graphiques, sont aussi employées, grâce à la facilité de lecture qu'elles procurent.

2.1 Visualisations statistiques

Graphiques à courbes en deux dimensions Il s'agit d'une représentation simple, constituée par deux axes et une courbe, traduisant la variation de la valeur d'une variable en fonction de la valeur d'un de ses paramètres. L'outil PerfExplorer (Huck et Malony, 2005) offre la possibilité de comparer l'efficacité relative ou le *speed-up* en fonction du nombre de processeurs grâce à ce type de représentations.

Histogrammes, diagrammes circulaires Les histogrammes et diagrammes circulaires sont des représentations statistiques classiques traditionnellement utilisées pour comparer la répartition d'un certain nombre de valeurs. On peut ainsi représenter, par exemple, le nombre de messages reçus par les processus, le taux de mémoire utilisé par les différentes machines. On trouve ce genre de visualisations à partir de ParaGraph (Heath et Etheridge, 1991). Paradyn (Miller et al., 1995) implémente des histogrammes horizontaux contenant plusieurs types de valeurs, chacun possédant des échelles différentes. Les diagrammes circulaires sont présents dans Pajé et permettent de quantifier le temps passé dans les différents états par chaque processus. La possibilité est laissée à l'utilisateur de définir la tranche de temps sur laquelle faire l'analyse. Cette méthode est idéale pour comparer deux processus et isoler des problèmes de performance reliés à un déséquilibre de répartition des tâches, par exemple.

Nuages de points Le nuage de points est une représentation graphique donnant des indications sur le degré de corrélation entre deux ou plusieurs variables liées. Elle permet ainsi d'observer des relations (directes, inverses), des dépendances (fortes ou faibles), des tendances (linéaires, non linéaires) entre les variables, d'avoir un aperçu de l'homogénéité des répartitions ou d'isoler des données aberrantes. Une visualisation de ce type est disponible au sein de l'outil PerfExplorer. Les auteurs se servent de la corrélation pour ignorer des données qui fourniraient des informations redondantes (comme celles fournies par certains compteurs matériels), en vue de simplifier l'analyse.

"Area charts" Ils consistent en une représentation en deux dimensions, basée sur un graphique à courbes, où sont superposées différentes quantités à observer, mises en exergue par des zones de couleurs différentes. L'intérêt est de pouvoir comparer la taille des aires respectives et leur évolution en fonction de l'abscisse. Dans PerfExplorer (figure 1) , cette représentation peut être utilisée afin de comparer l'évolution du temps relatif passé dans différentes fonctions ou états en fonction du nombre de processeurs.

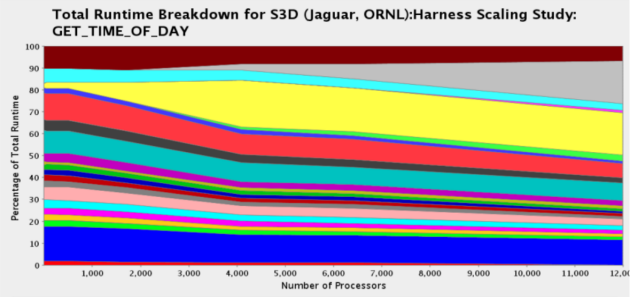


FIG. 1 – *Exemple de représentation statistique issue de PerfExplorer : un "area chart" représentant le pourcentage de temps passé dans différentes fonctions en fonction du nombre de processeurs.*

Représentations en trois dimensions Des représentations en trois dimensions sans axe temporel permettent d'élargir l'analyse en affichant une donnée en fonction de trois variables. ParaProf (Bell et al., 2003) possède ainsi ce type de visualisations pour des nuages de points, des histogrammes, ou à d'autres représentations à base de réseaux de triangles.

2.2 Conclusion sur les outils statistiques

Les visualisations synthétiques basées sur les statistiques donnent à l'analyste une première approche efficace de la trace mais ne suffisent pas à elles seules à permettre une rétro-action sur le code de l'application. Les informations fournies permettent de déceler un comportement général problématique (un processus gourmand en ressources, par exemple) mais les relations de causalité liant une configuration d'événements et d'états et leurs conséquences sur le déroulement de l'application ne sont pas explicités (comme des échéances non respectées, des interblocages). De même, les liens entre les composants de l'application ou leur structure topologique ne sont pas représentés, ce qui rend par exemple difficile l'analyse des communications afin de mettre en évidence des goulets d'étranglement sur le réseau. Les représentations statistiques incitent donc l'utilisateur à affiner son examen de la trace à travers des méthodes de visualisation permettant une analyse temporelle et structurelle détaillée de celle-ci, tout en leur fournissant des opérateurs mathématiques pertinents pour la gestion du passage à l'échelle.

3 Visualisation détaillée de la trace

Les techniques de visualisation présentes au sein des outils d'analyse donnent des informations détaillées sur le comportement de l'application mais aussi sur la structure de la trace. Les plus plébiscitées sont issues de domaines divers, comme la gestion de projet (diagramme de Gantt), le stockage de données (visualisation treemap), l'algèbre (représentations matricielles) ou l'analyse mathématique (graphes d'appels de fonctions). Un certain nombre d'entre-elles ont été catégorisées par Schnorr (2009) durant sa thèse, que nous avons complété en suivant la classification qu'il propose.

3.1 Visualisation du déroulement de l'exécution

Un certain nombre de techniques de visualisation permettent d'étudier le comportement d'une application au cours du temps à travers l'analyse de la séquence des événements, états et modification de la valeur de ses variables. L'objectif est en particulier de déduire les relations de causalité entre les différents événements aboutissant à un état particulier de l'application à un moment donné, pour éventuellement influer sur l'origine d'un comportement indésirable à partir de ces indications (redimensionnement de la plate-forme, modification du code source).

Diagrammes de Gantt Le diagramme de Gantt (Wilson, 2003) est apparu en 1896 et a d'abord été employé dans la gestion de projets. Dans le cas des outils de visualisation de trace, il est composé d'un axe temporel en abscisse tandis que l'axe des ordonnées correspond à l'ensemble des *conteneurs* (par exemple cœurs, processus, threads, fonctions) dont les états vont être représentés au cours du temps. Ces derniers sont disponibles sous la forme de rectangles disposés horizontalement, parfois colorés, dont les extrémités correspondent aux dates de début et de fin. Dans le cas d'événements ponctuels, certains outils se servent de symboles. KPTrace utilise, par exemple, des images de verrous pour indiquer la prise ou la libération d'un mutex ou d'une sémaphore tandis qu'il souligne un changement de contexte ou une préemption par des flèches verticales colorées et orientées. LTTng Eclipse viewer (Linux Tools Project, 2012) possède deux diagrammes de Gantt séparés représentant d'un côté les éléments logiciels (processus et fonctions) et de l'autre les entités matérielles (processeurs, IRQ). A l'inverse, Pajé, utilisant un format de trace générique, laisse la possibilité d'entrelacer la hiérarchie matérielle et logicielle.

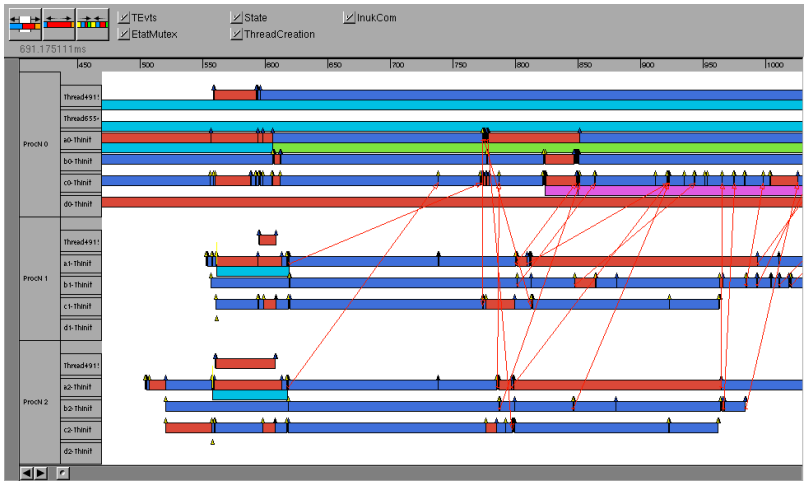


FIG. 2 – *Exemple de diagramme de Gantt issu de l'outil Pajé et montrant l'exécution de trois processus (états et communications)*

Graphiques à courbes, histogrammes, "area charts", avec axe temporel Représentations similaires à celles présentées dans la section 2.1, elles se distinguent ici par la présence d'un axe temporel en abscisse servant à montrer l'évolution de la valeur d'une ou plusieurs variables au cours du temps. Ces variables peuvent être des informations relatives au matériel : la Streamline de l'outil ARM DS5 (ARM Ltd., 2012) possède des représentations de ce type, sous la forme d'histogrammes (utilisation du processeur par le système et l'utilisateur), ou d'*area charts* (comparant deux à deux les interruptions logicielles et matérielles, les *cache hits* et *misses*, les écritures et lectures sur disque, la mémoire utilisée et disponible). A contrario, on peut aussi trouver des informations issues du logiciel : dans Pajé, il est ainsi possible de représenter des variables sous la forme d'une courbe intégrée au diagramme de Gantt principal. Ces techniques peuvent être transposées en trois dimensions, comme la *3D Terrain Visualization* de Paradyn, où les données représentées sont dépendantes de deux variables et du temps.

"3D View" Dans les visualisations comportementales, un axe est utilisé pour le temps. Cela oblige à projeter les ressources sur une dimension, au détriment de la compréhension de leur structure. Une solution proposée par l'auteur de Triva (Schnorr, 2009) est la *3D view* qui combine la structure, à travers une représentation de la topologie, et un axe temporel.

3.2 Visualisation de la structure de l'application

Les techniques de visualisation structurelles se caractérisent par la représentation des composants constituant l'application ainsi que leurs interactions. La dimension temporelle, si elle est présente, n'est pas centrale, comme dans les représentations évoquées dans la section 3.1.

Appels de fonctions Les graphes d'appels de fonction sont des graphes orientés, dans lesquels sont représentés les appels (symbolisés par des liens) entre les différents composants (fonctions ou méthodes représentés par des nœuds). Cette technique de visualisation est efficace pour l'analyse d'applications parallèles, en particulier celles conçues sous la forme d'un graphe de flot de données. Virtue (Shaffer et Reed, 2000) contient un graphe d'appel de fonctions intégré au sein d'un environnement en trois dimensions, permettant à l'utilisateur de manipuler la représentation (rotation, sélection de parties) afin d'obtenir plus d'informations sur les éléments affichés. L'outil Extravis (Cornelissen et al., 2008) propose une représentation circulaire (figure 3) sur le bord de laquelle les différentes fonctions sont indiquées. Un gradient de couleur appliqué sur les liens donne au choix des informations sur le sens des appels, ou sur leur survenue dans le temps.

Communications Les communications entre différents processus à un instant donné peuvent être matérialisées par des matrices. Ces dernières consistent en une représentation à deux dimensions, sous la forme d'une grille. Des couleurs peuvent être employées pour indiquer le type de communications ou la taille des données transmises. Elles sont un outil idéal pour des applications parallèles en permettant la détection de goulets d'étranglement, et figurent dans le choix des visualisations proposées par Vampir. Une méthode alternative pour représenter les communications est le graphe orienté. Les entités sont symbolisées par les nœuds, tandis que les liens indiquent les communications. On en trouve une implémentation dans ParaGraph. Le graphe topologique en est une version étendue où la position relative des entités représente la

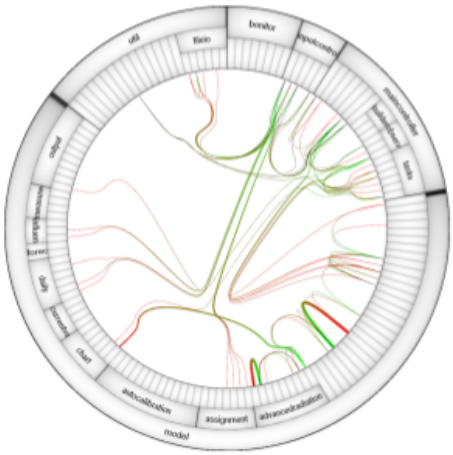


FIG. 3 – *Graphe d'appels de fonctions circulaire d'Extravis. Les liens symbolisent les appels entre les fonctions, le gradient de couleur indiquant leur sens.*

topologie réelle du réseau. Implémenté dans Triva (Schnorr et al., 2011), il possède un mécanisme d'animation temporel qui permet d'analyser au cours du temps la variation des volumes de communications.

Topologie de la plate-forme La hiérarchie des ressources et la topologie d'une plate-forme peuvent être représentés à l'aide d'arbres. Une technique d'arbres particulière est la représentation *treemap* inventée par Shneiderman (1991). Elle consiste à diviser l'espace de visualisation (l'écran) en représentant chacune des entités sous la forme d'un rectangle, dont la surface est proportionnelle à la valeur de la variable que l'on veut mettre en évidence. La treemap est particulièrement utile pour représenter un grand nombre d'entités grâce à la surface disponible qu'elle offre et la possibilité de représenter les niveaux hiérarchiques. Elle est employée dans Triva (Schnorr et al., 2012), dans sa version *squarified*, pour montrer la répartition des tâches sur de larges architectures distribuées. En particulier, les auteurs utilisent le même mécanisme d'animation temporel que pour les graphes topologiques afin de montrer la variation du ratio du temps attribué à l'exécution de deux tâches sur chaque entité, au cours du déroulement de l'application.

3.3 Problématiques liées aux méthodes de visualisation présentées

Les méthodes visuelles présentées ne répondent pas toutes de la même manière à des contraintes comme la gestion de l'échelle (nombre important d'entités produisant les événements, hiérarchie complexe) ou la diversité et de la quantité des informations à analyser (durée de la trace, nombre d'événements, hétérogénéité des durées des événements). Des symptômes comme les pertes de contexte, dues à la nécessité de *scroller* ou dézoomer (par exemple dans le cas du diagramme de Gantt, pour afficher l'intégralité des conteneurs ou se déplacer dans

l'axe du temps), nuisent à la compréhension générale de la trace. En outre, un déplacement dans la trace ou un changement d'échelle sont aussi associés à des phénomènes de crénelage : si la taille d'un élément à afficher après un zoom ou un dézoom est un nombre réel non entier, il ne pourra pas être représenté correctement à cause de la discrétisation de l'écran. Si cet objet est déplacé, la taille affichée sera potentiellement modifiée en fonction de sa position par rapport aux pixels de l'écran, alors qu'il ne sera pas pas censé avoir été redimensionné. Ces artefacts sont susceptibles d'induire l'analyste en erreur, voire de donner une représentation intégralement illisible dans le cas d'un fort dézoom. Certaines représentations dont la taille des éléments est liée à la surface d'affichage disponible (treemaps, matrices) peuvent voir ces dernières devenir trop petites à cause d'un nombre d'entités trop important.

4 Passage à l'échelle de l'analyse

Afin de palier au manque de surface disponible pour afficher les visualisations et répondre aux problématiques liées au passage à l'échelle, différents auteurs ont mis au point des solutions basées sur des mécanismes d'agrégation. La méthodologie employée nécessite de définir trois critères :

- Le choix de la dimension d'agrégation, corrélée avec la technique de visualisation qui va servir de support. On distingue en particulier la dimension temporelle et la dimension spatiale. Cette dernière se fonde sur des critères structurels de l'application liés à l'ensemble de ses ressources.
- Le choix de la métrique de proximité, qui est une condition ou un critère dépendant de l'axe et qui va déclencher l'agrégation. Cette métrique permet de déterminer l'ensemble des éléments sur lesquels effectuer l'opération.
- Le choix de l'opérateur qui va être appliqué sur l'ensemble des données à agréger et ainsi fournir une représentation de cet agrégat.

Exemples d'agrégations Parmi les outils employant des agrégations basées sur l'axe temporel, on peut citer LTTng et Pajé (diagrammes de Gantt), mais aussi Triva (animation temporelle). La métrique de proximité de LTTng est une taille de pixel minimale liée à la représentation des états. Lors d'un dézoom, si la taille de l'objet devient plus petite que le seuil toléré, alors il est visuellement agrégé, ce qui est matérialisé par un point au-dessus de sa position. On retrouve un principe similaire dans Pajé, où la métrique est un groupe d'états contigus. Si l'un d'eux possède, suite à un redimensionnement, une taille inférieure à un certain nombre de pixels, le groupe d'états est agrégé sous la forme d'un état unique barré d'une ligne diagonale. Dans le cas de Triva, la métrique est une tranche de temps, dont l'analyste peut paramétrer la durée, et selon laquelle la trace va être découpée. Tous les événements contenus dans chaque tranche de temps seront alors agrégés, en calculant la moyenne des durées des états ou des valeurs des variables au cours de la tranche de temps, ou la somme des communications ou des événements ponctuels.

Dans le cas d'une agrégation selon l'axe spatial, le diagramme KPTrace permet d'agréger les ressources selon une métrique déterminée par la proximité hiérarchique : tous les conteneurs fils sont masqués et rassemblés au sein de leur conteneur père. Les états et événements associés aux fils sont alors déportés dans la ligne correspondant au père. Pajé permet aussi la même agrégation hiérarchique, bien que les états agrégés correspondant aux fils ne soient

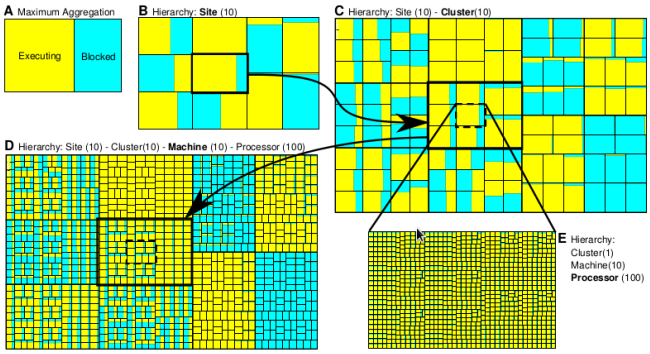


FIG. 4 – *Agrégation hiérarchique de la treemap proposée par Triva. On se déplace progressivement au sein de la hiérarchie en désagrégeant et en se focalisant sur une partie de la représentation.*

représentés que sous la forme d'un état indéterminé (les communications avec des conteneurs externes sont cependant toujours affichées). La même métrique de proximité spatiale est utilisée par Triva au sein de la treemap (figure 4) et du graphe topologique. Les fils ne sont plus représentés, et les valeurs caractérisant le père sont obtenues à l'aide de l'opérateur *somme* appliqué sur l'ensemble des fils. Dans le cas du regroupement par profil proposé par Vampir, la métrique est liée à une proximité "comportementale" : les différentes tâches qui possèdent un profil d'exécution proche, dont la similitude est évaluée à l'aide d'une distance euclidienne appliquée sur un jeu de paramètres (fonctions, durées de celles-ci), sont rassemblées. L'ensemble des profils est ensuite affiché sous la forme d'un histogramme. La durée des états contenu dans chaque profil est la moyenne de celles des processus agrégés.

4.1 Problématiques liées aux méthodes d'agrégation proposées

Les solutions étudiées, si elles possèdent un certain nombre de techniques permettant de répondre aux besoins de passage à l'échelle, amènent toutefois à certains questionnements.

Pertes d'informations liées à l'agrégation de données Les opérateurs utilisés afin d'agréger les données contenues dans la trace font irrémédiablement perdre une certaine quantité d'information : la moyenne efface celles concernant la distribution des données, tandis que des opérateurs de distribution ne donnent des informations que sur une seule des valeurs du groupe agrégé. Dans le cas de l'agrégation temporelle de Triva, par exemple, la valeur moyenne calculée sur tranche de temps ne permet pas de déterminer le nombre d'occurrences de chaque état, leur périodicité, ou encore la variance de leur durée. Pour une analyse détaillée de la distribution et de la régularité, cette agrégation sera alors insuffisante. De même, des méthodes d'agrégation visuelles, comme celles proposées par LTTng Viewer ou Pajé, permettent de savoir qu'un état agrégé est "trop petit" pour être représenté, mais si on compare deux états agrégés, il n'y a plus d'informations sur leur durée relative. La perte de l'information tolérée est reliée à la pertinence de la technique employée pour le type de comportement à visualiser :

un profilage utilisant le regroupement de profil de Vampir se satisfera d'une perte d'information due à l'agrégation, à contrario de l'analyse des interblocages qui aura besoin de plus de précision. Detyniecki (2001) évoque un certain nombre d'opérateurs d'agrégations dont l'utilisation pourrait être envisagée afin de mieux contrôler la perte d'information, mais aussi afin de faire ressortir des caractéristiques comportementales que les opérateurs les plus simples ne permettent pas de mettre en évidence. De même, des efforts peuvent être poursuivis afin d'attribuer plus de sémantique aux agrégats visuels (taille des symboles corrélée à une information perdue par l'agrégation, par exemple).

Phénomènes d'instabilité On peut parler d'instabilité si une légère variation des paramètres d'agrégation peut générer une représentation différente. Cela peut être visible lors d'une agrégation temporelle utilisant une tranche de temps : lors d'un redimensionnement de ces dernières, la distribution des événements va changer, et certains vont passer d'une tranche à l'autre, être répartis sur plusieurs tranches de temps, ou passer d'un état réparti à une tranche unique. Ainsi, pour deux représentations possédant n et $n+1$ tranches de temps, l'amplitude associée aux tranches de temps "correspondantes", de par leur proximité temporelle, pourra être différente. Le phénomène produira deux représentations agrégées différentes et visuellement non cohérentes malgré des paramètres d'agrégation très proches. Pour rétablir la cohérence, on pourrait ajouter des contraintes sur la métrique de proximité (tranches de temps multiples les unes des autres) ou envisager un lissage ou un filtrage, avec toutefois le risque de modifications ou de pertes d'informations.

5 Conclusion

Les méthodes de visualisation statistiques, temporelles et structurelles de traces d'exécution sont des techniques complémentaires répondant en partie aux besoins liés à l'analyse des systèmes informatiques actuels. Elles fournissent des informations générales sur la trace, sur le comportement des applications au cours du temps et les relations entre les différents composants de la plate-forme étudiée. Face aux problématiques de passage à l'échelle dues à des architectures structurellement complexes et un niveau de détail important, des solutions faisant appel à des mécanismes d'agrégation simplifient la visualisation en agissant au niveau des dimensions temporelles et spatiales. Cependant, ces techniques sont limitées par la perte d'informations qu'elles induisent, la fiabilité des représentations qu'elles produisent, ou à cause d'effets d'instabilités. De plus, la pertinence de leur utilisation est fortement corrélée à la structure de la plate-forme, à l'application à analyser, et au type de comportements à discriminer.

Les outils actuels ne laissent pas le choix des opérateurs, de l'ensemble des données ou de la partie de la trace sur lesquels agir. Des agrégations successives, combinées, ou la représentation d'un même objet en utilisant simultanément des agrégations différentes n'ont pas encore été évoquées dans la littérature concernant l'analyse visuelle des traces. Nous pensons qu'étendre les possibilités offertes par ce procédé, à travers l'interaction et la possibilité d'implanter de nouvelles fonctions d'agrégation, est une solution face aux limitations dont il souffre actuellement. En particulier, la multiplication des possibilités d'agrégation serait une solution remédiant aux pertes d'informations en permettant le recoupage des données.

Références

ARM Ltd. (2012). ARM streamline performance analyzer - ARM. http ://www.arm.com/products/tools/software-tools/ds-5/streamline.php.

Bell, R., A. D. Malony, et S. Shende (2003). ParaProf : a portable, extensible, and scalable tool for parallel performance profile analysis. In *Euro-Par 2003 Parallel Processing*, Volume 2790, pp. 17–26. Berlin, Heidelberg : Springer Berlin Heidelberg.

Campbell, A., G. Coulson, et M. Kounavis (1999). Managing complexity : middleware explained. *IT Professional 1*(5), 22 –28.

Chassin de Kergommeaux, J. (2000). Pajé, an interactive visualization tool for tuning multi-threaded parallel applications. *Parallel Computing 26*(10), 1253–1274.

Chassin de Kergommeaux, J., Ã. Maillet, et J.-M. Vincent (2001). *Program Development for Cluster Computing : Methodology, Tools and Integrated Environments*, Chapter Monitoring Parallel Programs for Performance Tuning in Distributed Environments, chapter 6. Nova Science.

Cornelissen, B., A. Zaidman, D. Holten, L. Moonen, A. Van Deursen, et J. van Wijk (2008). Execution trace analysis through massive sequence and circular bundle views. *Journal of Systems and Software 81*(12), 2252–2268.

Desnoyers, M. (2012). Common trace format (CTF) specification (v1.8.2). http ://git.efficios.com/ ?p=ctf.git ;a=blob_plain ;f=common-trace-format-specification.txt.

Detyniecki, M. (2001). Fundamentals on aggregation operators. *This manuscript is based on Detyniecki's doctoral thesis and can be downloaded from*.

Fournier, P. M., M. Desnoyers, et M. R. Dagenais (2009). Combined tracing of the kernel and applications with LTTng. In *Proceedings of the 2009 Linux Symposium*.

Heath, M. T. et J. A. Etheridge (1991). Visualizing the performance of parallel programs. *IEEE Software 8*(5), 29–39.

Herlihy, M. et N. Shavit (2008). *The art of multiprocessor programming*. Amsterdam ; London : Elsevier/Morgan Kaufmann.

Huck, K. A. et A. D. Malony (2005). Perfexplorer : A performance data mining framework for large-scale parallel computing. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, pp. 41.

Hwang, K., J. J. Dongarra, et G. C. Fox (2012). *Distributed and cloud computing*. Amsterdam ; London : Elsevier/Morgan Kaufmann.

Knüpfer, A., R. Brendel, H. Brunst, H. Mix, et W. E. Nagel (2006). Introducing the open trace format (OTF). In *Computational Science – ICCS 2006*, Volume 3992, pp. 526–533. Berlin, Heidelberg : Springer Berlin Heidelberg.

Knüpfer, A., H. Brunst, J. Doleschal, M. Jurenz, M. Lieber, H. Mickler, M. S. Müller, et W. E. Nagel (2008). The vampir performance analysis tool-set. In *Tools for High Performance Computing*, pp. 139–155. Berlin, Heidelberg : Springer Berlin Heidelberg.

Linux Tools Project (2012). Linux tools Project/LTTng2/User guide - eclipsepedia. http ://wiki.eclipse.org/index.php/ Linux_Tools_Project/LTTng2/User_Guide.

Miller, B. P., M. D. Callaghan, J. M. Cargille, J. K. Hollingsworth, R. B. Irvin, K. L. Karavanic,

K. Kunchithapadam, et T. Newhall (1995). The paradyn parallel performance measurement tool. *Computer 28*(11), 37–46.

Prada-Rojas, C., F. Riss, X. Raynaud, S. De-Paoli, et M. Santana (2009). *Observation tools for debugging and performance analysis of embedded linux applications*. September.

Quinn, M. J. (2004). *Parallel programming in C with MPI and OpenMP*. Boston [etc.] : McGraw-Hill.

Schnorr, L. M. (2009). *Quelques Modèles de Visualisation pour l'Analyse des Applications Parallèles*. Ph. D. thesis.

Schnorr, L. M., G. Huard, et P. O. A. Navaux (2012). A hierarchical aggregation model to achieve visualization scalability in the analysis of parallel applications. *Parallel Computing 38*(3), 91–110.

Schnorr, L. M., A. Legrand, et J.-M. Vincent (2011). Detection and analysis of resource usage anomalies in large distributed systems through multi-scale visualization. *Concurrency and Computation : Practice and Experience*, n/a–n/a.

Score-P Project (2012). Score-P – HPC profiling and event tracing infrastructure. http ://www.vi-hps.org/projects/score-p.

Shaffer, E. et D. Reed (2000). Real-time immersive performance visualization and steering. *ACM SIGGRAPH Computer Graphics 34*(2), 11–14.

Shende, S. S. (2006). The tau parallel performance system. *International Journal of High Performance Computing Applications 20*(2), 287–311.

Shneiderman, B. (1991). Tree visualization with tree-maps : A 2-d space-filling approach. *ACM Transactions on Graphics 11*, 92–99.

Shneiderman, B. (1996). The eyes have it : A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pp. 336–343.

Toupin, D. (2011). Using tracing to diagnose or monitor systems. *Software, IEEE 28*(1), 87–91.

Wilson, J. (2003). Gantt charts : A centenary appreciation. *European Journal of Operational Research 149*(2), 430–437.

Summary

Behavior analysis of an application software is a difficult task because of computing system growing complexity. This phenomenon is characterized by complex sofware stacks in embedded and personnal systems case, while parallel systems analysis has to deal with large architectures and strong indeterminism. Execution traces visualization usage related to software analysis has rised because of its ability to solve these issues. Analysis tools propose now a large set of techniques, distinguished by data volume, system scale, or type of software behavior they can represent. We propose to study them accross this survey by discussing about statistics methods, then about behavioral and structural visualisations which give precise details on the applications. We will extend our explanations to techniques capable to solve the issues related to scalability found in most of traditional visualization ways.

Visualisation efficace de folksonomies à base d'« Intersecteurs »

Amira Mouakher*, Sadok Ben Yahia*

*Faculty of Sciences of Tunis, Tunis, Tunisia.
sadok.benyahia@fst.rnu.tn

Résumé. L'essor des sites collaboratifs sur Internet relevant du mouvement participatif que l'on désigne souvent du nom de Web 2.0 a permis la naissance de nouvelles formes d'indexations des contenus du Web créées librement par les usagers et partagés au sein de réseaux sociaux, baptisées sous le nom de folksonomies. Dans cet article, nous introduisons une nouvelle approche pour la visualisation de larges *folksonomies* basées sur les traverses minimales "intersecteurs". L'idée phare de cette approche est basée sur l'extraction d'un ensemble restreint de noeuds à partir duquel l'utilisateur pourrait, par un processus de pliage/dépliage, aisément naviguer dans la folksonomie.

1 Introduction

Actuellement, de plus en plus d'utilisateurs utilisent les folksonomies, à travers les services de tagging, pour partager leur ressources avec d'autres utilisateurs. En effet, les utilisateurs des folksonomies n'ont pas besoin de compétences spécifiques pour pouvoir les utiliser. D'autant plus que les tags sont des mots clés librement choisis et les utilisateurs sont ainsi libres de créer ou de sélectionner les tags de leur choix. Ces folksonomies ouvrent de nombreuses possibilités d'exploitation, pourvu qu'un système de partage efficace des connaissances et des ressources en tire profit. Cependant, les folksonomies mettent en évidence la possibilité de développer un système plus élaboré, auquel les internautes pourront être tentés de participer. Elles ont aussi le mérite d'exister face aux projets de web sémantique dont les applications concrètes sont encore limitées.

Les sites de folksonomies permettent une exploration plus ouverte et hasardeuse du contenu que les moteurs de recherche. De plus, le vocabulaire évolue parfois rapidement et la folksonomie permet de refléter en temps réel cette évolution.

Cette structure spécifique aux folksonomies a rendu leur exploitation d'un grand intérêt pour la recherche d'information dans la mesure où elles permettraient d'identifier et de surveiller l'émergence de nouveaux concepts.

Cependant, la taille et la densité de cette structure présentent souvent des obstacles réels qui peuvent parfois handicaper à une visualisation lisible et compréhensible par l'utilisateur (Lohmann et Díaz, 2012).

L'objectif de cet article est d'introduire une nouvelle approche de visualisation de folksonomies en se basant sur la notion de noeuds utilisateurs. L'idée motrice de cette approche est basée sur la détection efficace d'un ensemble restreint de noeuds utilisateurs qui permettent de

retrouver les autres utilisateurs. Ces noeuds particuliers sont connus dans la littérature comme ambassadeurs, ou diffuseurs (Scripps et al., 2007a,b; Agarwal et al., 2008; Opsahl et Hogan, 2010; Jelassi et al., 2012) et ils ont été largement utilisés dans le marketing viral. Ainsi, l'utilisateur aura a visualiser un ensemble très restreint de noeuds, même pour des graphes de très grande taille. En effet, des résultats expérimentaux (Jelassi et al., 2012) montrent que la taille de ces noeuds est autour d'une vingtaine même pour de très larges folksonomies. De point de vue théorique, il s'avère que ces noeuds ne sont que les traverses minimales les plus petites en termes de taille.

Opérant dans une perspective de pliage/dépliage et en conformité avec la recommandation de Shneiderman (Shneiderman, 1996) "*Overview first, then details on demand*", l'utilisateur ne sera plus submergé par la quantité du noeud du graphe. Par contre, il sera en mesure de "développer" une partie du réseau en cliquant sur les noeuds intersecteurs pour retrouver les noeuds couverts et gérer graduellement la complexité du réseau.

Le reste de l'article est organisé comme suit. Dans la section 2, nous introduisons tout d'abord brièvement quelques notions préliminaires sur les folksonomies. Par la suite, nous allons présenter un survol de l'état de l'art sur la qualité des tags ainsi que les approches de visualisation folksonomies à bases de graphes. La section 3 est dédiée à une présentation détaillée de l'approche de visualisation à base de noeuds intersecteurs. La dernière section conclut cet article et dresse les perspectives de recherche des travaux en cours.

2 État de l'art

Dans cette section, nous allons commencer par présenter les notions préliminaires sur les folksonomies.

2.1 Notions préliminaires sur les Folksonomies

Une folksonomie peut être décrite par une contexte triadique introduit dans ce qui suit :

Définition 1 (CONTEXTE TRIADIQUE) *Selon Ganter et Wille (1999), un contexte triadique d'extraction (ou un contexte d'extraction) est un quadruplet $\mathcal{K} = (\mathcal{E}, \mathcal{I}, \mathcal{C}, \mathcal{Y})$, où $\mathcal{E}$, $\mathcal{I}$ and $\mathcal{C}$ sont des ensembles, et $\mathcal{Y}$ est une relation ternaire entre $\mathcal{E}$, $\mathcal{I}$ et $\mathcal{C}$, i.e., $\mathcal{Y} \subseteq \mathcal{E} \times \mathcal{I} \times \mathcal{C}$. Les éléments de $\mathcal{E}$, $\mathcal{I}$ et $\mathcal{C}$ sont respectivement appelés objets, attributs, et conditions et $(e, i, c) \in \mathcal{Y}$, sous-entend que l'objet e est relatif à l'attribut i relativement à la condition c.*

Un contexte triadique représente exactement la structure d'une *folksonomie* dont la définition est la suivante :

Définition 2 (FOLKSONOMIE) *Hotho et al. (2006) ont défini une folksonomie comme étant un ensemble de tuples $\mathcal{F} = (\mathcal{U}, \mathcal{T}, \mathcal{R}, \mathcal{Y})$ où*

- *$\mathcal{U}$, $\mathcal{T}$ et $\mathcal{R}$ sont des ensembles finis dont les éléments sont appelés utilisateurs, tags et ressources.*
- *$\mathcal{Y} \subseteq \mathcal{U} \times \mathcal{T} \times \mathcal{R}$ représente une relation triadique dont chaque $y \subseteq \mathcal{Y}$ peut être représenté par un triplet :*

$$y = \{(u, t, r) \mid u \in \mathcal{U}, t \in \mathcal{T}, r \in R\}.$$

ce qui signifie que l'utilisateur u a annoté la ressource r par le tag t.

Une *folksonomie* est également appelée *"Social Tagging"*, un processus par lequel de nombreux utilisateurs ajoutent des données sous forme de tags pour partager des ressources. En d'autres termes, il s'agit d'un support du Web 2.0 pour la classification de ressources. L'annotation des ressources par les tags facilite ainsi le partage et la recherche de l'information, e.g., DEL.ICIO.US, FLICKR, CONNOTEA, YOUTUBE, etc.

Exemple 1 *Le Tableau 1 illustre un exemple d'une folksonomie $\mathcal{F}$ avec $\mathcal{U}$ = {u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8}, $\mathcal{T}$ = {t_1, t_2, t_3, t_4, t_5, t_6} et $\mathcal{R}$ = {r_1, r_2, r_3}.*

Notons que chaque "$\times$" représente une relation triadique entre un utilisateur appartenant à $\mathcal{U}$, un tag appartenant à $\mathcal{T}$ et une ressource annotée appartenant à $\mathcal{R}$. Par exemple, l'utilisateur u_1 a taggué la ressource r_1 par le biais des tags t_1 et t_2.

U/R-T	r_1						r_2						r_3					
	t_1	t_2	t_3	t_4	t_5	t_6	t_1	t_2	t_3	t_4	t_5	t_6	t_1	t_2	t_3	t_4	t_5	t_6
u_1	×	×					×	×										
u_2	×	×	×	×			×	×	×	×								
u_3			×	×					×	×			×	×				×
u_4					×								×	×				
u_5	×	×					×	×					×	×				
u_6					×										×	×	×	
u_7			×	×					×	×					×	×	×	×
u_8															×	×	×	

TAB. 1 – *Une folksonomie.*

2.2 Qualité des tags

Plusieurs travaux se sont intéressés aux mesures de qualité des tags dans les folksonomies,*e.g*, (Sen et al., 2007; Krestel et Chen, 2008; Damme et al., 2008; Gu et al., 2011).

Parmi ces travaux, nous avons retenu les métriques qui ont été proposées par Damme et al. (2008) pour mesurer la pertinence des tags. Dans ce qui suit, nous allons expliquer brièvement ces mesures que nous allons utiliser pour notre approche.

1. (FRÉQUENCE DES TAGS) : dans une folksonomie donnée, la fréquence d'apparition d'un tag donné pour une même ressource est considéré comme un facteur pour juger la pertinence de ce dernier. Ainsi, pour chaque ressource tagguée, nous pouvons ordonner les tags en mesurant la fréquence d'apparition de chaque tag distinct. Les tags ayant les fréquences les plus élevées présentent alors une meilleur qualité.
2. (ACCORD DES TAGS) : pour une ressource x, cette métrique se définit comme l'ensemble des tags qui ont été sélectionnés parmi la plupart des utilisateurs qui ont taggué cette ressource. Au début, nous déterminons la fréquence de chaque tag unique. Ensuite, nous calculons le nombre des utilisateurs qui ont taggué chaque ressource. L'accord des tags est par conséquent mesuré par la division de la fréquence des tags par le nombre des utilisateurs qui ont taggué cette ressource. Lorsque tous les utilisateurs sont d'accord sur un certain tag, cette mesure doit être égale à 1. Plus cette valeur est proche de 0, moins les utilisateurs sont d'accord sur tag donné.

$$Acc(t_{x,y}) = \frac{Freq(t_{x,y})}{\sum |u_i|}$$

3. (TF-IRF) : cette métrique est l'une des variante de la mesure statistique "TF-IDF" proposée par Salton et Buckley (1988). Cette dernière est souvent utilisée en recherche d'information et en particulier dans la fouille de textes. Elle permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus.

   Cette mesure nécessite une certaine modification pour l'évaluation des tags parce que sa formule contient des informations textuelles. Cependant, dans une folksonomie, les ressources tagguées ne sont pas toujours textuelles (fichier mp3, image). Ainsi, la formule de TF-IRF proposée par Damme et al. (2008) est la suivante :

$$TF - IRF(t_{x,y}) = \frac{\textit{Freq}(t_{x,y})}{T_y} * \log(\frac{\sum |r_i|}{r_x})$$

   Cette formule est basée sur TF-IDF (T_y = nombre total de tags pour la ressource y et r_x = nombre des ressources qui ont le tag t_x).

Dans ce qui suit, nous allons passer en revue quelques travaux qui ont été mené sur la visualisation de folksonomies ainsi que leurs principales cadres applicatifs.

2.3 Approches de Visualisation, à base de graphes, de Folksonomies

Plusieurs travaux se sont intéressés à la visualisation des folksonomies. Au début, cet hypergraphe tripartite d'utilisateurs, de tags et de ressources a été représenté à travers un *"nuage de tags"*. Dans ce cadre, Lohmann et al. (2009) ont établi une étude comparative sur les performances des différents modèles d'un *"nuage de tags"* pour l'exploration visuelle. Ces derniers ont conclu que le choix du modèle du *"nuage de tags"* intervient pour garantir une meilleure visualisation. Cependant, Kangpyo et al. (2009) ont affirmé que cette représentation ne fournit pas d'informations sur les relations entre les tags. D'où, la nécessité de la visualisation sous la forme d'un graphe.

Dans cette section, nous allons nous consacrer à ce type de visualisation. Il est clair que la taille d'une folksonomie est en plein croissance. Ceci rend sa visualisation un peu difficile voir impossible parfois pour l'utilisateur. (Lohmann et Díaz, 2012) affirment que l'efficacité de cette visualisation doit être accompagnée d'une simplification ou d'une réduction pour garantir son exploitation.

(Lambiotte et Ausloos, 2006) ont essayé de simplifier la représentation de folksonomie en la projetant dans des structures moins complexes. Étant considérée comme un réseau triparti, la folksonomie est projetée à chaque fois dans un réseau d'ordre inférieur jusqu'à l'obtention d'un réseau uni-partie et ceci grâce à la corrélation entre les noeuds de même type.

Plusieurs projets ont été menés pour la visualisation des folksonomies. Nous pouvons citer le projet *TagGraph* [1] en 2007 de F. Oliphant qui propose une visualisation folksonomique de la base de données de photographies en ligne Flickr. Il s'agit d'un navigateur de folksonomie permettant à l'utilisateur ayant saisi un tag donné de voir les photos les plus récentes indexées dans Flickr selon ce tag. Dans le même contexte, nous pouvons citer aussi l'outil *Tag Galaxy* [2]

1. http://taggraph.com/
2. http://taggalaxy.de/

développée en 2008 par le développeur allemand S. Wood. Cette application de visualisation permet d'explorer Flickr comme une galaxie, et naviguer entre les planètes de tags.

(Kangpyo et al., 2009) ont proposé un outil FOLKSOVIZ permettant la dérivation des relations d'équivalence, de subsomption et de similarité entre les tags tout en se basant sur le corpus de Wikipedia. Cet outil gère l'affichage des relations sémantiques entre les tags de façon intuitive pour fournir à l'utilisateur une visualisation efficace de la folksonomie.

(Dattolo et Pitassi, 2011) ont introduit un nouvel outil FOLKVIEW qui permet la représentation dynamique de folksonomie à l'aide d'un système multi-agents. Ainsi, ce système serait capable de fournir des vues personnalisées pour les utilisateurs. D'après les différentes approches que nous avons présentées dans cette sous-section, nous pouvons affirmer que le problème majeur qui émerge dans la visualisation des folksonomies est l'incapacité de visualiser de larges folksonomies. En effet, la taille d'une folksonomie évolue en temps réel et sa visualisation devient sans intérêt si elle n'est pas accompagnée d'une stratégie de réduction (Lohmann et Díaz, 2012). En outre, nous pouvons aussi remarquer que la majorité des travaux se sont intéressés à la visualisation des tags, tandis que la relation entre les utilisateurs a été négligée.

3 VIF : Visualisation de larges folksonomies

Une visualisation efficace doit remplir certaines conditions. Selon Shneiderman (Shneiderman, 1996) "*Le processus de la recherche visuelle des informations consiste à : fournir tout d'abord une vue d'ensemble, puis zoomer, filtrer et enfin détailler à la demande*" [3].

A partir de cette recommandation, il faut tout d'abord fournir une vue d'ensemble du système d'informations. Par conséquent, déduire ses caractéristiques principales afin d'identifier les principales sources d'intérêt de ce système. Suite à la vue d'ensemble, l'utilisateur peut explorer certaines parties du système et obtenir des informations détaillées.
Ainsi, il n'est pas intéressant de voir tous les détails mais seulement ceux demandés par l'utilisateur. L'utilisateur doit pouvoir étudier n'importe quelle partie du système afin de voir toutes les informations qu'il désire. Ainsi, une bonne visualisation doit satisfaire les trois facteurs suivants (Grand, 2001) :

1. Une vue globale du système,
2. Détails à la demande,
3. Interaction avec l'utilisateur.

Dans ce qui suit, nous détaillons la nouvelle approche de visualisation, à base de graphe, d'une folksonomie. Rappelons que l'idée clé de cette nouvelle approche est d'éviter à l'utilisateur d'être submergé par la grande taille du graphe associé à la folksonomie. Ainsi, nous proposons de présenter à l'utilisateur un ensemble de noeuds restreint d'utilisateurs à partir desquels il pourrait retrouver les noeuds. Le mode d'exploration de ce réseau est le pliage/dépliage, i.e., l'utilisateur développe le sous-ensemble de noeuds qui l'intéresse. L'approche "VIF" opère en deux étapes comme le montre la figure 1.

1. **Extraction des tri-concepts** : ces tri-concepts vont constituer une représentation condensée des triplets de la folksonomie.

3. *"The Visual Information Seeking Mantra is : overview first, zoom and filter, then details on demand."*

2. **Extraction des traverses minimales « intersecteurs »** : cette étape opère sur les tri-concepts extraits lors de la première étape. Partant du fait que nous sommes à la recherche d'éléments particuliers qui sont connectés au maximum (ou l'intégralité) de noeuds possibles dans le réseau, la notion de traverse minimale, qui intersecte toutes les hyperarêtes de l'hypergraphe, constitue un cadre idéal pour localiser ces intersecteurs. Ainsi, nous allons effectuer une projection sur la dimension "Utilisateur" pour lui proposer comme entrée pour le processus d'extraction des noeuds intersecteurs.

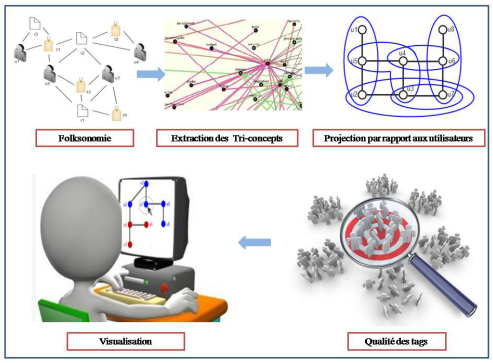


FIG. 1 – *Description de l'approche* VIF.

Les deux étapes de l'approche VIF sont détaillées dans ce qui suit :

3.1 Extraction des tri-concepts

Les tri-concepts, extraits d'une folksonomie, constituent une représentation concise exacte des triplets d'une folksonomie. Un tri-concept est défini comme suit :

Définition 3 (CONCEPT TRIADIQUE)*En adaptant au cas triadique la notion d'*"itemsets fermés fréquents" *introduite par Agrawal et al. (1993), Trabelsi et al. (2012) ont défini un concept triadique (ou un tri-concept) d'une* folksonomie $\mathcal{F} = (\mathcal{U}, \mathcal{T}, \mathcal{R}, \mathcal{Y})$ *comme un triplé* (U, T, R) *où* $U \subseteq \mathcal{U}$, $T \subseteq \mathcal{T}$, *et* $R \subseteq \mathcal{R}$ *avec* $U \times T \times R \subseteq \mathcal{Y}$ *tel que le triplé* (U, T, R) *est maximal, i.e., pour* $U_1 \subseteq U$, $T_1 \subseteq T$ *et* $R_1 \subseteq R$ *où* $U_1 \times T_1 \times R_1 \subseteq \mathcal{Y}$, *les ensembles* $U \subseteq U_1$, $T \subseteq T_1$, *et* $R \subseteq R_1$ *impliquent toujours* $(U, T, R) = (U_1, T_1, R_1)$.

Notons que pour un tri-concept (U, T, R), les ensembles U, R et T sont respectivement appelés ***Extent***, ***Intent***, et **Modus** du tri-concept (U, T, R).

Pour cette tâche, nous avons opté pour l'algorithme TRICONS (Trabelsi et al., 2012), qui opère en largeur d'abord pour localiser les tri-générateurs, i.e., les plus petits éléments d'une classe d'équivalence, pour localiser rapidement les tri-concepts. Cet algorithme a montré ses preuves comparativement aux algorithmes pionniers de la littérature.

Exemple 2 *En se référant à la* folksonomie *du Tableau 1,* TC_1=($\{u_1, u_2, u_5\}$, $\{t_1, t_2\}$, $\{r_1, r_2\}$) *est un tri-concept de* $\mathcal{F}$*, i.e, c'est l'ensemble maximal de tags et de ressources partagées par* u_1, u_2 *et* u_5*. Par ailleurs, les* 5 *autres tri-concepts extraits sont les suivants :*
TC_2=($\{u_2, u_3, u_7\}$, $\{t_3, t_4\}$, $\{r_1, r_2\}$), TC_3=($\{u_4, u_6\}$, $\{t_5\}$, $\{r_1\}$), TC_4=($\{u_3, u_4, u_5\}$, $\{t_1, t_2\}$, $\{r_3\}$), TC_5=($\{u_6, u_7, u_8\}$, $\{t_3, t_4, t_5\}$, $\{r_3\}$) TC_6=($\{u_3, u_7\}$, $\{t_6\}$, $\{r_3\}$).

3.2 Extraction des traverses minimales « intersecteurs »

Un réseau social peut être défini comme un ensemble d'entités interconnectées les unes avec les autres (Wasserman et Faust, 1994). Les entités du réseau sont le plus souvent des personnes ou des organisations mais il peut s'agir aussi de pages web (Watts, 1999), d'articles scientifiques (White et al., 2004), de films, *etc*. Les relations décrivent des interactions entre les entités.

Dans la pratique, on ne connaît pas toujours de façon précise les relations existants entre les entités prises deux à deux. En revanche, il peut exister des sous-groupes formant des communautés au sein du réseau et on peut savoir à quelle(s) communauté(s) appartient chaque entité. C'est dans ce cadre, que nous proposons de définir la notion d'intersecteur au sein du réseau, en faisant appel aux concepts d'hypergraphe et de traverse minimale.

Définition 4 HYPERGRAPHE *(Berge, 1989)*
Soit le couple $H = (\mathcal{X}, \xi)$ *avec* $X = \{x_1, x_2, \ldots, x_n\}$ *un ensemble fini et* $\xi = \{E_1, E_2, \ldots, E_m\}$ *une famille de parties de* $\mathcal{X}$*. H constitue un hypergraphe sur* $\mathcal{X}$ *si :*
- $E_i \neq \emptyset$, $i \in \{1,\ldots, m\}$
- $\bigcup_{i=1,\ldots,m} E_i = \mathcal{X}$

Les éléments x_i de $\mathcal{X}$, appelés sommets de l'hypergraphe, correspondent aux entités du réseau et les éléments de ξ, appelées hyperarêtes de l'hypergraphe, correspondent aux communautés.

Exemple 3 *La figure 2 illustre un hypergraphe* $H = (U, TC)$ *tel que* $U = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$ *et* TC=($\{u_1, u_2, u_5\}$, $\{u_2, u_3, u_7\}$, $\{u_4, u_6\}$, $\{u_3, u_4, u_5\}$, $\{u_6, u_7, u_8\}$, $\{u_3, u_7\}$)

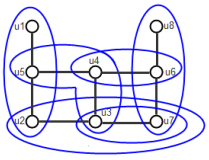


FIG. 2 – *L'hypergraphe associé à l'ensemble des tri-concepts extraits à partir de la folksonomie donnée par la table 1.*

Définition 5 TRAVERSE MINIMALE *(Berge, 1989)*
Soit un hypergraphe $H = (\mathcal{X}, \xi)$ *où* $\xi = \{E_1, E_2, \ldots, E_m\}$ *est l'ensemble des hyperarêtes. L'ensemble des traverses de H, noté* γ_H*, est égal à* $\{T \subset \mathcal{X} | \forall i = 1, \ldots, m, T \bigcap E_i \neq \emptyset\}$.

Une traverse T de γ_H est dite *minimale* si il n'existe pas une autre traverse S de γ_H incluse dans $S : \nexists\, S \in \gamma_H \; s.t. \; S \subset T$.

On notera $\mathcal{M}_H$, l'ensemble des traverses minimales définies sur H. Dans l'exemple illustré par la figure 2, l'ensemble $\mathcal{M}_H$ des traverses minimales de l'hypergraphe est : { (u_1, u_4, u_7), (u_2, u_4, u_7) et (u_4, u_5, u_7).
Dans le contexte d'un hypergraphe, où nous ne disposons que des communautés présentes dans le réseau, nous considérons que les intersecteurs dans ce réseau correspondent à un ensemble de sommets, de taille minimale, capable de représenter l'ensemble des communautés du réseau. Ainsi, il apparaît que la notion de traverse minimale, qui intersecte toutes les hyperarêtes de l'hypergraphe, constitue un cadre idéal pour localiser les « Intersecteurs », définis comme suit :

Définition 6 TRAVERSE MINIMALE INTERSECTEUR
Soit $H = (\mathcal{X}, \xi)$, un hypergraphe et $I \subset \mathcal{X}$, on dit que I est une traverse minimale intersecteur, notée TMI*, si I vérifie les deux conditions suivantes :*

1. *(Condition de minimalité) I est une traverse minimale de $\mathcal{M}_H$ au sens de la cardinalité : $|I| = \tau(H)$ où $\tau(H) = Min\ \{|T|, \forall T \in \mathcal{M}_H\}$.*
2. *(Condition de qualité maximale) :*
   Pertinence(I)=max{Qualité_tag(tag(I)) t.q. $|I| = \tau(H)$}

Dans le même contexte, Jelassi et al. (2012) définissent similairement les « diffuseurs » en privilégiant les traverses minimales avec une connectivité maximale.

Ainsi, un ensemble de sommets constitue une TMI si sa taille est la plus petite possible et s'il maximise la condition de meilleure qualité des tags. Plus précisément, la première condition vise à prendre un ensemble d'intersecteurs le plus petit possible et la deuxième condition, dite de *qualité maximale*, permet, s'il existe plusieurs traverses minimales vérifiant la condition de minimalité, de privilégier les utilisateurs ayant taggué avec une pertinence maximale. Ainsi l'objectif est de représenter la folksonomie à l'aide d'un minimum de sommets du graphe ayant une qualité maximale en termes de tags. Dans ce cadre, la recherche des intersecteurs dans un hypergraphe nécessite l'extraction d'une famille particulière des traverses minimales. Plusieurs algorithmes ont été présentés, dans la littérature, pour l'extraction des traverses minimales dans un hypergraphe, e.g., (Berge, 1989; Kavvadias et Stavropoulos, 2005; Hébert et al., 2007).

Dans notre approche, nous avons utilisé l'algorithme TMD-MINER proposé par (Jelassi et al., 2012). Ce dernier a été montré plus efficace par rapport à ses concurrents. En effet, il détermine intelligemment le degré de transversalité, i.e., le niveau K où est situé les TMIs. Ensuite, les candidats de taille k sont générés parmi lesquels les traverses minimales sont retenues. Comme ces éléments vérifient la première condition (de minimalité) de la définition 6, on calcule le degré de pertinence des tags associés. Enfin, nous retenons le TMI qui présente la qualité de pertinence maximale comme l'exige la deuxième condition.

4 Exemple illustratif

Afin d'illustrer notre approche, nous considérons la folksonomie donnée par le tableau 1. Tout d'abord, nous allons extraire les différents tri-concepts qui se trouvent dans cette folksonomie. Pour se faire, nous allons appliquer l'algorithme TRICONS introduit par (Trabelsi

et al., 2012) avec des $minsup_u = 2$, $minsup_t = 1$ et $minsup_r = 1$. Nous obtenons alors l'hypergraphe donné par la figure 2. Ce dernier peut être aussi représenté par le tableau 2 où les hyperarêtes T_i sont les projections des tri-concepts par rapport aux différents utilisateurs. Nous commençons tout d'abord par rechercher les plus petits itemsets essentiels, dont

	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
T_1	×	×			×			
T_2		×	×				×	
T_3				×		×		
T_4			×	×	×			
T_5						×	×	×
T_6			×				×	

TAB. 2 – *Le contexte d'extraction relatif à l'hypergraphe de la figure 2*

le support est égal au nombre d'hyperarêtes. Il s'agit des traverses vérifiant la condition nécessaire et la condition de qualité. Ces traverses minimales qui remplissent aussi la condition de minimalité au sens de l'inclusion représentent des TMI candidats. Nous pouvons extraire comme candidats les éléments $\{u_1u_4u_7\}$, $\{u_2u_4u_7\}$ et $\{u_4u_5u_7\}$. La recherche des traverses minimales, les plus petites en termes de cardinalité, se justifie par le fait que notre approche cherche à couvrir le réseau de manière efficace, i.e, en utilisant le minimum de noeuds possible. Une fois nous avons extrait ces TMI candidats, la fonction *Pertinence* évalue pour chaque TMI candidat la qualité des tags qui lui sont associées. En effet, chaque utilisateur appartenant à un TMI candidat a taggué certaines ressources dans la folksonomie. Nous allons retenir celui dont l'ensemble des tags est le plus pertinent. L'évaluation des tags s'effectue via les métriques Acc et TF−IRF qui ont été définies dans la section précédente. Ensuite, nous allons calculer la moyenne de ces deux mesures pour chaque TMI candidat et nous allons retenir celui qui maximise cette moyenne. Ainsi, prenons par exemple le cas du candidat $\{u_1u_4u_7\}$. Le noeud u_1 a taggué la ressource r_1 par $\{t_1, t_2\}$ et la ressource r_2 par les tags $\{t_1, t_2\}$. Le noeud u_4 a taggué la ressource r_1 par $\{t_5\}$ et la ressource r_3 par les tags $\{t_1, t_2\}$. Tandis que le noeud u_7 a taggué la ressource r_1 par $\{t_3, t_4\}$, la ressource r_2 par les mêmes tags et la ressource r_3 par les tags $\{t_3, t_4, t_5, t_6\}$. Après avoir calculé la pertinence des tags associés à tous les TMI candidats, nous retenons le TMI le plus pertinent, a savoir $\{u_4, u_5, u_7\}$.

Le tableau 3 regroupe toutes les traverses minimales avec les différentes valeurs de pertinence relatives à leurs tags. La figure 3 représente l'interface de visualisation accessible à

TMI	{Tags}	Per.	TMI	{Tags}	Per.	TMI	{Tags}	Per.
$u_1u_4u_7$	$\{t_1,t_2\}\rightarrow r_1$	0.19	$u_2u_4u_7$	$\{t_1,t_2,t_3,t_4\}\rightarrow r_1$	0.06	$u_4u_5u_7$	$\{t_1,t_2\}\rightarrow r_1$	0.19
	$\{t_1,t_2\}\rightarrow r_2$	0.19		$\{t_1,t_2,t_3,t_4\}\rightarrow r_2$	0.06		$\{t_1,t_2\}\rightarrow r_2$	0.19
	$\{t_5\}\rightarrow r_1$	0.16		$\{t_5\}\rightarrow r_1$	0.16		$\{t_1,t_2\}\rightarrow r_3$	0.19
	$\{t_1,t_2\}\rightarrow r_3$	0.19		$\{t_1,t_2\}\rightarrow r_3$	0.19		$\{t_5\}\rightarrow r_1$	0.16
	$\{t_3,t_4\}\rightarrow r_1$	0.19		$\{t_3,t_4\}\rightarrow r_1$	0.19		$\{t_1,t_2\}\rightarrow r_3$	0.19
	$\{t_3,t_4\}\rightarrow r_2$	0.19		$\{t_3,t_4\}\rightarrow r_2$	0.19		$\{t_3,t_4\}\rightarrow r_1$	0.19
	$\{t_3,t_4,t_5,t_6\}\rightarrow r_3$	0.06		$\{t_3,t_4,t_5,t_6\}\rightarrow r_3$	0.06		$\{t_3,t_4\}\rightarrow r_2$	0.19
							$\{t_3,t_4,t_5,t_6\}\rightarrow r_3$	0.06
		1.17			0.91			**1.36**

TAB. 3 – *Processus de calcul des* TMI.

l'utilisateur après l'extraction du TMI pertinent. Comme le montre la figure 3(a), l'utilisateur peut voir les noeuds constituant le TMI ayant la meilleure qualité de tags $\{u_4, u_5, u_7\}$. Si ce dernier souhaiterait explorer davantage les noeuds qui sont "touchés" par cette traverse, il peut cliquer sur l'un de ces noeuds. Si nous prenons l'exemple du noeud u_5, un click sur ce noeud permet de faire découvrir les noeuds en sorte de "dépliage" du réseau u_2, u_4 et u_1(c.f., figure 3(b)). Ensuite, si l'utilisateur décide d'explorer un autre noeud u_7, les voisins du noeuds

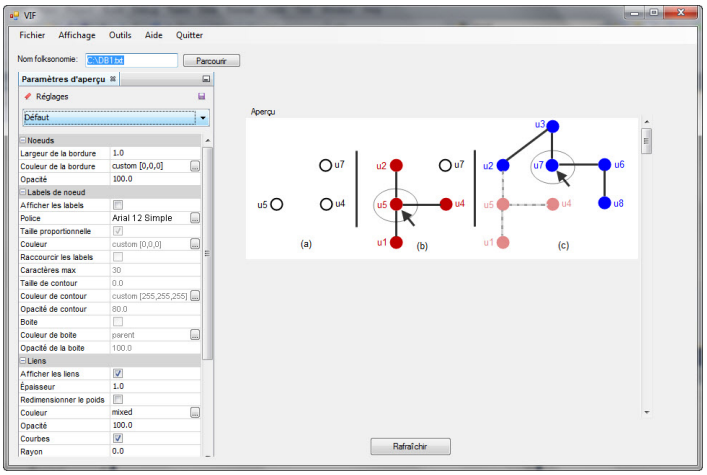


FIG. 3 – *Capture d'écran de l'outil* VIF.

ayant été déjà explorés sont moins visibles et les nouveaux voisins s'affichent, à savoir u_3, u_2, u_6 et u_8(c.f., figure 3(c)). Comme nous pouvons le remarquer, à partir d'un seul TMI pertinent, nous avons pu régénérer tous les noeuds de la folksonomie.

5 Conclusion

Dans cet article, nous avons introduit l'outil VIF, pour la visualisation de larges données folksonomiques, basé sur les traverses minimales intersecteurs. Pour l'implémentation de l'outil de visualisation, nous comptons utiliser l'approche de (Simonetto et Auber, 2009), qui mentionnent que la plupart des hypergraphes ne peuvent pas être dessinés de façon à ce que chaque paire d'hyperarêtes dont l'intersection est vide ne se chevauche pas. Dans ce cadre, les auteurs proposent une méthode permettant d'afficher n'importe quel hypergraphe en créant plusieurs régions pour certaines hyperarêtes. Au delà de l'évaluation de l'interface de visualisation, qui déjà en cours de développement, nous comptons nous intéresser à l'aspect prétraitement des tags entrés par l'utilisateur. Par ailleurs, nous comptons prendre en considération le profil de l'utilisateur, qui reste à définir, dans le choix du TMI pertinent.

Références

Agarwal, N., H. Liu, L. Tang, et P. S. Yu (2008). Identifying the influential bloggers in a community. In *Proceedings of the Int. Conference on Web Search and web Data Mining (WSDM '08)*, Stanford, USA.

Agrawal, R., T. Imielinski, et A. Swami (1993). Mining association rules between sets of items in large databases. In *Proceedings of the ACM-SIGMOD Int. Conference on Management of Data*, Washington D. C., USA, pp. 207–216.

Berge, C. (1989). Hypergraphs : Combinatorics of finite sets. pp. 256.

Damme, C., M. Hepp, et T. Coenen (2008). Quality Metrics for Tags of Broad Folksonomies. In *Proceedings of I-semantics'08*, Graz, Austria.

Dattolo, A. et E. Pitassi (2011). Visualizing and managing folksonomies. In *Proceedings of the Workshop on Semantic adaptive social web of the 19th Int. conference on Advances in User Modeling*, pp. 6–14. Springer-Verlag.

Ganter, B. et R. Wille (1999). *Formal Concept Analysis.* Springer, Heidelberg.

Grand, B. L. (2001). *Extraction d'information et visualisation de systèmes complexes sémantiquement structurés*. Doctorat d'université, Paris, Décembre 2001, Université Pierre et Marie Curie.

Gu, X., X. Wang, R. Li, K. Wen, Y. Yang, et W. Xiao (2011). Measuring social tag confidence : is it a good or bad tag ? In *Proceedings of the 12th Int. conference on Web-age information management*, WAIM'11, Berlin, Heidelberg, pp. 94–105. Springer-Verlag.

Hotho, A., R. Jäschke, C. Schmitz, et G. Stumme (2006). Information retrieval in folksonomies : Search and ranking. In Y. Sure et J. Domingue (Eds.), *Proceedings of The Semantic Web : Research and Applications. LNCS*, Volume 4011, pp. 411–426. Springer, Heidelberg.

Hébert, C., A. Bretto, et B. Crémilleux (2007). A data mining formalization to improve hypergraph minimal transversal computation. *Fundamenta Informaticae. 80*(4), 415–433.

Jelassi, N., C. Largeron, et S. Ben Yahia (2012). TMD-Miner : Une nouvelle approche pour la détection des diffuseurs dans un système communautaire. In *Actes de la 12eme Conférence Int. Francophone sur l'Extraction et la Gestion de Connaissance*, Bordeaux, France.

Kangpyo, L., K. Hyunwoo, S. Hyopil, et K. Hyoung-Joo (2009). Folksoviz : A semantic relation-based folksonomy visualization using the wikipedia corpus. In *Proceedings of the 10th ACIS Int. Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing*, SNPD '09, Washington, DC, USA, pp. 24–29. IEEE Computer Society.

Kavvadias, D. J. et E. C. Stavropoulos (2005). An efficient algorithm for the transversal hypergraph generation. *Journal of Graph Algorithms and Applications 9*(2), 239–264.

Krestel, R. et L. Chen (2008). The art of tagging : Measuring the quality of tags. In *ASWC*, pp. 257–271.

Lambiotte, R. et M. Ausloos (2006). Collaborative tagging as a tripartite network. In *Computational Science, ICCS 2006*, pp. 1114–1117. Springer Berlin / Heidelberg.

Lohmann, S. et P. Díaz (2012). Representing and visualizing folksonomies as graphs : a reference model. In *Proceedings of the Int. Working Conference on Advanced Visual Interfaces*, AVI'12, New York, NY, USA, pp. 729–732. ACM.

Lohmann, S., J. Ziegler, et L. Tetzlaff (2009). Comparison of tag cloud layouts : Task-related performance and visual exploration. In T. Gross, J. Gulliksen, P. Kotzé, L. Oestreicher, P. A. Palanque, O. R. Prates, et M. Winckler (Eds.), *Proceedings of INTERACT (1)*, Volume 5726, pp. 392–404. Springer.

Opsahl, T. et B. Hogan (2010). Growth mechanisms in continuously-observed networks : Communication in a facebook-like community. *CoRR*.

Salton, G. et C. Buckley (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management : an Int. Journal 24*, 513–523.

Scripps, J., P.-N. Tan, et A.-H. Esfahanian (2007a). Exploration of link structure and community-based node roles in network analysis. In *Proceedings of the 7th IEEE Int. Conference on Data Mining (ICDM'07)*, Omaha, USA, pp. 649–654.

Scripps, J., P.-N. Tan, et A.-H. Esfahanian (2007b). Node roles and community structure in networks. In *Proceedings of the 1st Workshop on Web Mining and Social Network Analysis (SNA-KDD'07)*, San José, California, pp. 26–35.

Sen, S., M. F. M. Harper, A. Lapitz, et J. Riedl (2007). The quest for quality tags. In *GROUP '07 : Proc. of the 2007 Int. ACM conference on Supporting group work*, New York, NY, USA, pp. 361–370. ACM.

Shneiderman, B. (1996). The eyes have it : A task by data type taxonomy for information visualization. In I. C. S. Press (Ed.), *Proceedings IEEE Symposium on Visual Languages*, Boulder, Colorado, pp. 336–343.

Simonetto, P. et D. Auber (2009). An heuristic for the construction of intersection graphs. In *Proceedings of the 13th Int. Conference on Information Visualisation*, pp. 673–678.

Trabelsi, C., N. Jelassi, et S. Ben Yahia (2012). Scalable mining of frequent tri-concepts from folksonomies. In *Proceedings of The 16th Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD 2012*, Kuala Lumpur, Malysia, pp. 231–242. Springer-Verlag.

Wasserman, S. et K. Faust (1994). *Social Network Analysis, methods and application* (4th edition : 1998, 1999 ed.).

Watts, D. J. (1999). *Small worlds : The dynamics of networks between order and randomness*.

White, H. D., B. Wellman, et N. Nazer (2004). Does citation reflect social structure ? *Journal of the American Society for Information Science and Technology 55*, 111–126.

Summary

Recently, social bookmarking systems have received an increasing attention in both academic and industrial communities. This success is owe to their easy use that relies on simple intuitive process, allowing their users to label diverse resources with freely chosen keywords aka *tags*. The obtained collections are known under the nickname of *Folksonomy*. In this paper, we introduce a new approach dedicated to visualisation of large folksonomies, based on the "intersecting" minimal transversals. The main thrust of such an approach is the proposal of a reduced set of "key" nodes from which the remaining nodes would be faithfully retrieved. Thus, the user could navigate in the folksonomy through a folding/unfolding process.

Une nouvelle représentation de règles d'association par une métaphore moléculaire

Zohra Ben Said[*,***] Fabrice Guillet[*]
Paul Richard[**], Julien Blanchard[*], Fabien Picarougne [*]

[*]LINA, UMR 6241 CNRS, Université de Nantes, Nantes, France
(fabrice.guillet,julien.blanchard,fabien.picarougne)@univ-nantes.fr,
https://www.lina.univ-nantes.fr
[**]LISA, EA 4094, Université d'Angers, Angers, France
paul.richard@univ-angers.fr
http://www.istia.univ-angers.fr/LISA
[***]LIUM, EA 4023, Université du Maine, Le Mans, France
zohra.ben_said@univ-lemans.fr
http://www-lium.univ-lemans.fr

Résumé. Afin d'extraire des connaissances intéressantes à partir de grandes quantités de résultats générés par des algorithmes d'extraction de règles d'association, des représentations visuelles de règles d'association peuvent être utilisées. Ces représentations aident l'utilisateur à trouver et valider les connaissances intéressantes. Toutes les représentations visuelles proposées ont été développées pour représenter des règles d'association sans prêter attention aux relations entre les items et la contribution de chacun d'eux à la règle. Dans cet article, nous proposons une nouvelle représentation pour la visualisation des règles d'association. Cette représentation permet la visualisation des attributs qui constituent la prémisse et des attributs qui constituent la conclusion, la contribution de chacun à la règle et les corrélations entre chaque paire d'attributs de la prémisse et chaque paire d'attributs de la conclusion. L'introduction de cette représentation dans un outil de fouille visuelle de règles d' association permet de faciliter la tâche d'exploration et d'extraction de règles pertinentes.

1 Introduction

L'extraction de règles d'association consiste à trouver des corrélations entre les attributs présents dans une base de données Agrawal et al. (1993). Une règle d'association est une implication de la forme $X \rightarrow Y$, ou X (prémisse) et Y (conclusion) sont deux ensembles disjoint d'items. Par exemple, *lait, oeufs* $\rightarrow$ *pain* est une règle d'association disant que quand un consommateur achète du lait et des oeufs, il est susceptible d'acheter aussi du pain. A la sortie du processus d'extraction des règles d'association, l'utilisateur doit évaluer et sélectionner les règles intéressantes (Post-traitement des règles d'association). Pour sélectionner les règles intéressantes les algorithmes d'extraction, des contraintes sur les mesures d'intérêt peuvent être

utilisées. Les contraintes les plus utilisées sont le seuil minimal de support et le seuil minimal de confiance.

Le principal inconvénient des algorithmes classiques d'extraction des règles d'association est le volume de règles générées. Le traitement cognitif de milliers de règles prend beaucoup plus de temps que de les générer. Afin de réduire la charge cognitive des utilisateurs, des représentations visuelles de règles d'association peuvent être utilisées pour faciliter et accélérer la compréhension, ainsi que la comparaison des règles. Toutes les techniques proposées pour la visualisation des règles ont été développées pour représenter une règle d'association comme un tout sans prêter attention aux relations entre les attributs qui constituent la prémisse et la conclusion et la contribution de chacun d'eux à la règle, alors que les attributs d'une règle d'association peuvent être plus informatifs que la règle elle-même Freitas (1998). Deux règles possédant les mêmes valeurs des mesures d'intérêt peuvent avoir différents degrés d'intérêt pour l'utilisateur en fonction des attributs composant la prémisse et la conclusion de la règle. De la même façon, les relations entre les items (corrélation) fournit à l'utilisateur une information plus claire par rapport à l'analyse d'une règle.

2 Visualisation des règles d'association

La visualisation peut être très bénéfique pour la fouille de règles d'association (Simoff et al. (2008)). En fait, les techniques de visualisation fournissent un moyen efficace aux utilisateurs en proposant des représentations visuelles significatives au lieu de listes textuelles (par exemple les navigateurs de règle Fule et Roddick (2004)).

Les représentations visuelles peuvent être utilisées :
- en conjonction avec les algorithmes d'extraction de règles (Ertek et Demiriz (2006) en 2D, Beale (2007) en 3D) ;
- comme une méthode de fouille de règles d'association à part entière (Liu et Salvendy (2006) en 2D,Blanchard et al. (2007) en 3D).

La plupart des outils de visualisation dans le post-traitement des règles permettent la visualisation de la syntaxe des règles et ainsi une comparaison facile des différentes règles affichées. Mais, le manque d'intéractivité engendre une saturation rapide de l'espace d'affichage (vu le nombre exorbitant des règles extraites). La majorité des outils (sauf ArVis Blanchard et al. (2007)) souffrent aussi du nombre des mesures d'intérêt affichées trop faible (support et confiance) et de leurs non mise en valeur.

En plus de la visualisation, les méthodes utilisant la visualisation durant le processus de fouille de règles d'association offrent aux utilisateurs des représentations interactives. La participation des utilisateurs dans le processus de fouille de règles d'association (essentiellement dans le choix des items) permet de diminuer considérablement le nombre de règles extraites (l'un des principaux inconvénient des algorithmes de fouille de règles d'association). Très peu d'outils de fouille de règles d'association offrent aux utilisateurs la possibilité d'interagir avec les algorithmes de fouille via des interfaces visuelles et aucun d'entre eux ne propose la visualisation des apports des items et les relations entre chaque paire d'items de la prémisse et

chaque paire des items de la conclusion. Deux types d'interface visuelle peuvent être utilisés : les interfaces 2D et les interfaces 3D.

Dans le tableau 1, nous présentons une comparaison entre la visualisation 2D et la visualisation 3D. En conclusion, une représentation 3D ne doit être utilisée que dans certaines conditions ou les données à afficher sont très riches sémantiquement (Tufte (1983)), ce qui est le cas d'un objet représentant une règle d'association.

	2D	3D
Données riches sémantiquement	Faible	Dense
Temps d'adaptation	Faible	Long
Désorientation	Faible	Grande
Occultation	Faible	Grande
Mémorisation		Augmenté

TAB. 1 – *Comparaison 2D vs 3D.*

3 Importance des items individuels des règles

3.1 Corrélation entre les items

Deux items sont corrélés s'ils ne sont pas indépendants. Deux attributs sont indépendants si le changement de la valeur de l'un n'affecte pas la valeur de l'autre. La mesure $Lift$ peut être utilisée pour calculer la corrélation entre chaque paire d'items de la prémisse et chaque paire d'items de la conclusion.

Le $lift$ a d'abord été défini par Brin et al. (1997) soulignant l'importance de la corrélation entre la prémisse et la conclusion d'une règle d'association.

Le $lift$ est défini comme suit :

$$Lift(X \rightarrow Y) = \frac{P(X,Y)}{P(X)P(Y)} = \frac{supp(X \cup Y)}{supp(X)supp(Y)} = \frac{Confidence(X,Y)}{P(Y)}$$

X et Y sont deux items de la prémisse ou bien deux items de la conclusion.

La corrélation entre deux items représente la quantité d'informations partagée entre ces items. Le $Lift$ permet de déterminer si les deux items sont corrélés positivement (Lift >1) ou négativement (lift <1). La corrélation est considérée comme positive (négative) si la fréquence observée d'exemple satisfaisant à la fois les deux items est supérieur (inférieur) à la fréquence attendue en supposant l'indépendance statistique entre ces deux items. L'étude réalisée par Freitas (2001) a montré que le concept d'interaction entre items peut être bénéfique pour le processus d'extraction de règles d'association et a proposé d'introduire l'interaction entre les items dans la conception de systèmes de fouille de règles.

L'interaction entre les items permet la détection des items surprenant qui ne peuvent être découverts en analysant une règle entière (sans tenir compte des items qui la composent). Les relations exprimées dans une règle globale sont très différentes de celles exprimées en parties séparées de la règle (prémisse et conclusion).

D'autre part, pour découvrir des règles utiles, l'utilisateur a besoin d'obtenir un aperçu des données et de comprendre les relations entre les attributs et leurs propriétés statistiques. L'exploration des items permet d'avoir une idée plus précise des données et apprendre davantage sur le modèle de données. Dans de nombreux cas (contexte biologique ou génétique par exemple) les items de la prémisse ont des relations faibles avec les items de la conclusion. Cependant, ils interagissent de manière complexe pour contrôler la conclusion (Chanda et al. (2010)).

3.2 l'importance de l'apport des items à la règle

Un item peut être important pour l'utilisateur si des régularités sont observées dans un petit ensemble de données, tout en étant inobservables dans l'ensemble des données. Une règle peut être considérée comme une disjonction de règles. La taille de la disjonction de règles est équivalente au nombre d'items qui composent la prémisse et la conclusion de la règle. Par exemple : $r : X1X2X3 \rightarrow Y1Y2$ est une règle d'association. Une disjonction de règles est $r1$: X1$\rightarrow Y1Y2, r2$: X2$\rightarrow$ Y1 Y2, r3 : X3 $\rightarrow$ Y1 Y2, r4 : Y1 $\rightarrow X1X2X3$ et $r5 : Y2 \rightarrow X1X2X3$.

A première vue, il semble que ces règles n'ont pas d'importance, car elles peuvent être considérées comme des règles redondantes. En se basant sur ce point de vue, la quasi-totalité des algorithmes d'extraction ne conservent pas ces règles dans les résultats. Toutefois, ces règles peuvent montrer des relations inattendues dans les données Freitas (1998).

Provost et Aronis (1996) a prouvé que les petites règles ont été jugées intéressantes par les experts dans leurs domaines d'application. En conséquence, il serait bénéfique pour l'utilisateur de visualiser automatiquement ces règles.

Afin d'évaluer la contribution de chaque item à la règle, Freitas (1998) a proposé la mesure du *Gain Informationnel* qui peut être positif ou négatif. Si un $Item \in$ Prémisse a un *Gain Informationnel* positif élevée cela signifie que :$r : Item \rightarrow Conclusion$ est une bonne règle. Si, au contraire, un $Item$ a un *Gain Informationnel* négatif élevé alors la règle est de mauvaise qualité.

En recherchant des règles intéressantes, un item avec un *Gain Informationnel* important peut signaler de nouvelles implications intéressantes inconnues par l'utilisateur. En même temps, une règle inclut les attributs dont le *Gain Informationnel* est faible ou négatif est une règle non pertinentes. Par conséquent, l'utilisateur ne perd pas de temps à chercher ces règles, car il sait déjà qu'elles ne sont pas intéressantes.

4 Nouvelle représentation de règles d'association

Pour trouver des règles intéressantes l'utilisateur à besoin principalement de :

1. Interpréter la sémantique des règles ;
2. Évaluer la qualité des règles.

Pour l'interprétation de la sémantique de la règle, l'utilisateur doit être capable prioritairement de i distinguer dans la représentation de la règle les items qui appartiennent à la prémisse des items qui appartiennent à la conclusion, ii visualiser ces items, et iii leurs intitulé. Concernant la qualité des règles, l'utilisateur a besoin de visualiser i une mesure globale de la règle

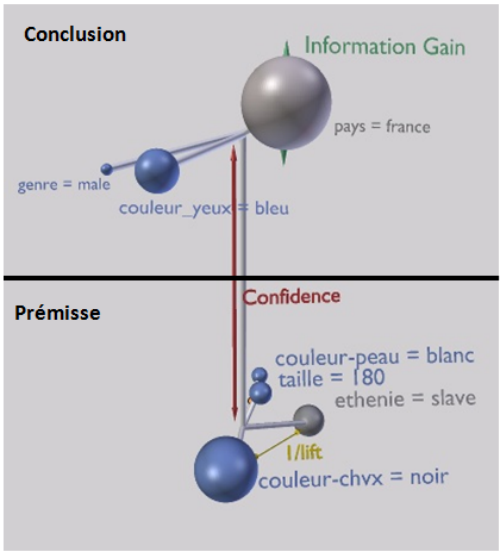


FIG. 1 – *Représentation d'une règle d'association*

afin de comparer facilement différentes règles, ii l'apport de chaque item à la règle, et iii la corrélation entre les items. Cet ordre d'importance sera traduit perceptuellement.

4.1 Encodage graphique

Pour représenter une règle d'association (Fig.1), nous avons choisi une représentation moléculaire qui est une représentation intuitive : une molécule est un ensemble d'atomes qui interagissent entre eux, une règle est un ensemble d'items qui interagissent dans la prémisse et dans la conclusion. Cette représentation permet de voir les items qui composent la règle, leurs apports et les relations entre eux.

La première chose qu'un utilisateur doit visualiser est la prémisse et la conclusion de la règle. Donc, nous utilisons la propriété pré-attentive dominante dans une représentation pour la représentation des variables quantitatives, ici, la distance sur un seul axe (Cleveland et McGill (1984)) pour séparer les items appartenant à la prémisse des items appartenant à la conclusion. Cette propriété représente aussi une mesure globale de la règle : la confiance. La métaphore visuelle met l'accent sur les règles possédant la confiance la plus élevée (2).

Comme dans une molécule ou les atomes sont représentés pas de sphères, les items seront représentés par des sphères et à côté de chaque sphère l'intitulé de l'item est affiché. Comme préconisé par Cleveland et McGill (1984), nous avons choisi un encodage graphique basé sur les positions et les tailles pour mettre en valeur les mesures d'intérêt les plus importantes, à savoir : *Gain informationnel* et corrélation entre les items. L'objet règle d'association est complexe. Afin d'avoir le plus grand degré de liberté pour le placement des sphères, nous avons choisi d'utiliser une représentation 3D.

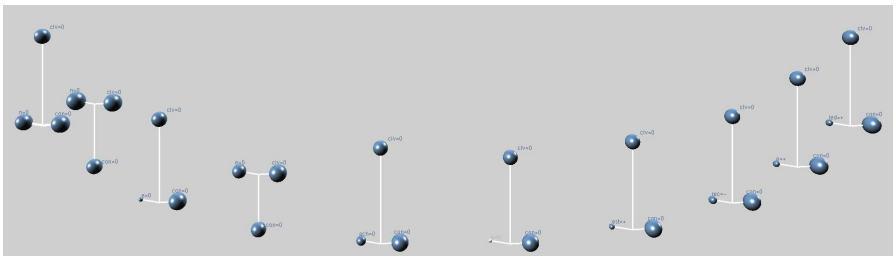


FIG. 2 – *Illustration d'un ensemble de règles d'association. La distance entre la prémisse et la conclusion met en évidence les règles dont la mesure globale est élevée.*

Chaque item de la règle est associé à une variable continue correspondant à son *Gain informationnel*. Dans l'espace 3D, chaque attribut est représenté par une sphère, sa taille est une représentation efficace du *Gain informationnel*. Le *Gain informationnel* possède aussi une caractéristique qualitative : positif ou négatif. Dans ce cas, toujours selon l'étude de Cleveland et McGill (1984), la propriété pré-attentive "couleur" est une bonne représentation de cette caractéristique. Pour le représenter nous utilisons deux couleurs : le bleu sera utilisé pour représenter les items dont le *Gain informationnel* est positif et le blanc sera utilisé pour représenter les items dont le *Gain informationnel* est négatif. Cette représentation met l'accent sur les items importants de la règles (grandes sphères de couleur bleue) et les items qui dégradent la qualité de la règle (grandes sphères de couleur blanche).

Le lift est une mesure positive utilisée pour indiquer le degré de corrélation entre deux items de la prémisse ou deux items de la conclusion. La distance entre deux items est une représentation efficace de cette mesure. Plus les items sont corrélés, plus les sphères sont proches. Dans notre représentation, la prémisse et la conclusion sont représentées par deux graphes distincts dans lesquels les noeuds représentent les items. Pour générer les coordonnées des items dans l'espace 3D, nous utilisons une version modifiée (section 4.2 pour plus d'informations) de l'algorithme basé sur les forces (spring-embedded like algorithm Hendley et al. (1999)). Cet algorithme permet l'auto-organisation des sphères dans l'espace par l'utilisation d'un système de forces en vue de déterminer les positions des sphères. L'utilisation d'un système de forces permet aux items corrélés d'être proches dans l'espace et items indépendants et négativement corrélés d'être éloignés.

4.2 L'algorithme de placement

Le graphe des items de la prémisse et le graphe des items de la conclusion sont projetés dans un espace 3D. Cet algorithme permet l'auto-organisation des sphères dans l'espace de visualisation en utilisant un système de forces pour déterminer les positions des sphères. La représentation graphique est constituée de noeuds et de liens, dont les propriétés représentent les données. Les données affectent les paramètres du graphe tels que la taille des sphères et leurs couleurs, la force et l'élasticité des liens. Dans cette représentation, chaque noeud du graphe représente un item. Les noeuds sont représentés par des sphères. Cet algorithme dynamique utilise un système de forces d'attraction dans le but de trouver un état d'équilibre et de

déterminer la position (les coordonnées) des noeuds (Table 4.2).

Input : Set of item l, Set of Edge E, Force = (x=0,y=0,z=0)
Output : Set of coordinates C

1. **forall** item $l_k \in l$ **do**
5. **forall** Edge $E_m \in E$ connected to l_k**do**
6. Force = Force + HookeAttraction(l_k, E_m)
7. **endfor**
8. C = C + Force
9. **endfor**

TAB. 2 – *Algorithme de placement.*

Les noeuds sont tous liés entre eux (les noeuds de la prémisse et les noeuds de la conclusion ne sont pas liés entre eux). Un lien est considéré comme un ressort entre deux noeuds avec une longueur initiale et un paramètre d'élasticité. La longueur initiale du ressort entre deux items représente la corrélation entre eux (1/lift). Plus les items sont corrélés, plus les liens sont courts et plus les noeuds sont proches. Pour réduire le nombre de contraintes, nous considérons uniquement les liens pertinents (lift est supérieur à 1).

La fonction d'attraction de Hook (F_H) décrit l'élasticité du ressort. Elle est appliquée entre chaque paire de noeuds et tend à les maintenir à une distance définie (1/lift).

La fonction d'attraction de Hook F_H, appliquée à un noeud p_1 de coordonnées C_1 lié à un autre noeud p_2 de coordonnées C_2 où k est la constante de raideur du ressort et R sa longueur initiale.

$$F_H = -k\frac{(|L-R)L}{|L|} \; with \; L = p_2 - p_1$$

Le système produit une approximation physique du mouvement des noeuds qui peut être facilement interprétée par un être humain où les noeuds sont répartis dans l'espace en fonction de la longueur du lien entre les deux noeuds (corrélation entre les deux items).

5 IUCEARVis : Outil de Fouille Visuelle et Interactive de Règles d'Association centré utilisateur

IUCEARVis est un prototype expérimental pour l'extraction et exploration interactive de règles d'association. L'inconvénient des méthodes classiques d'extraction de règles d'association est le nombre très élevé de règles extraites. Pour remédier à ce problème, nous proposons un outil de fouille locale de règles : l'utilisateur ne visualise à la fois qu'un sous-ensemble réduit de règles. Ce sous-ensemble sera proposé par le système en se basant sur des items

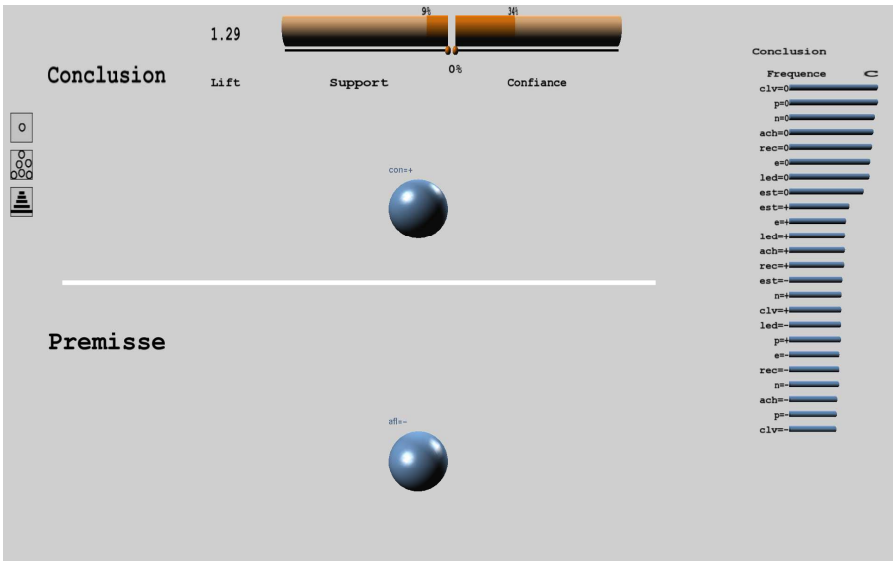


FIG. 3 – *Interface de sélection des items*

sélectionnés au préalable par l'utilisateur. L'utilisateur pourra ensuite naviguer entre les différents sous-ensembles en utilisant différents opérateurs de navigation. Ainsi, l'utilisateur peut effectuer une série d'exploration interactive en fonction de son intérêt aux différentes règles proposées par le système. Ensuite, si l'utilisateur juge certaines règles intéressantes, il peut les garder en mémoire et choisir d'explorer le prochain sous-ensemble. C'est à travers ce processus que la subjectivité de l'utilisateur est exprimée dans le processus d' extraction de règles d'association. Pour réaliser ces différentes fonctionnalités, IUCEARVis est composé de trois interfaces différentes : Interface de sélection des items intéressants, Interface d'extraction et d'exploration interactive de règles extraites et Interface de visualisation de l'historique des règles intéressantes. Plusieurs opérateurs d'interaction permettront à l'utilisateur d'interagir avec i les interfaces graphiques et ii l'algorithme d'extraction de règles d'association.

5.1 Interface de sélection des items intéressants

Via l'interface de sélection des items intéressants (Figure 3)l'utilisateur peut composer une règle d'association de référence (A) qui sera utilisée ensuite par le système afin d'extraire un sous-ensemble de règle susceptible d'intéresser l'utilisateur. Pour la construction de la règle de référence, l'utilisateur doit choisir les items de la prémisse et les items de la conclusion. Plusieurs indicateurs visuels comme par exemple la fréquence d'apparition de l'item dans la base de données ou si l'item améliore ou dégrade la qualité de la règle sont présentés afin d'aider l'utilisateur dans sa sélection d'items.

5.2 Interface d'extraction et d'exploration interactive de règles

En se basant sur la règle de référence construite par l'utilisateur (*A*), des fonctions d'anticipations préforment la transformation de cette règle de référence en sous-ensemble de règles. Les fonctions d'anticipations utilisées sont des fonctions de spécialisation et des fonctions de généralisation. La spécialisation et la généralisation sont, en effet, les deux processus cognitifs fondamentaux permettant de générer de nouvelles règles selon Holland et al. (1986).

Une des caractéristiques des fonctions d'anticipations est l'intégration des mesures d'intérêt les plus importants de règles d'association, le support ($supp$) et la confiance ($conf$). Chaque mesure d'intérêt est associée à un seuil minimum ($minsupp$ et $minconf$) défini par l'utilisateur. Ces seuils peuvent être modifiés par l'utilisateur à tout moment en cours de navigation. Pour définir les fonctions d'anticipation, nous avons regroupé les seuils dans la fonction booléenne *GoodQuality* :

$$\forall r \in R, GoodQuality(r) \Leftrightarrow (supp > minsuppandconf > minconf)$$

IUCEARVis propose six fonctions d'anticipations :

- Si la règle de référence possède un seul item en prémisse ou un seul item en conclusion

- $A_1(x \rightarrow Y) = z \rightarrow Y | z \in I\ (Y)$
- $A_2(X \rightarrow y) = X \rightarrow z | z \in I\ (X)$

- Si la règle de référence possède au moins deux items en prémisse ou deux items en conclusion

- Fonctions de spécialisations : ajouter un item à la prémisse ou à la conclusion ;

  - $A_3(X \rightarrow Y) = X \cup z \rightarrow Y | z \in I\ (X \cup Y) \wedge GoodQuality(X \cup z \rightarrow Y)$
  - $A_4(X \rightarrow Y) = X \rightarrow Y \cup z | z \in I\ (X \cup Y) \wedge GoodQuality(X \rightarrow Y \cup z)$

- Fonctions de généralisations : simplifier la prémisse ou la conclusion.

  - $A_5(X \rightarrow Y) = Xz \rightarrow Y | z \in X \wedge GoodQuality(X\ z \rightarrow Y)$
  - $A_6(X \rightarrow Y) = X \rightarrow Yz | z \in Y \wedge GoodQuality(X \rightarrow Y\ z)$

Selon le nombre d'items présents dans la base de données, le nombre de règles d'association générées peut être très élevé. Afin de limiter la charge cognitive de l'utilisateur, nous proposons un sous-ensemble limité de règles. Seuls les 10 règles les plus importantes seront affichées. Afin de sélectionner ces règles, l'utilisateur doit choisir un des critères de sélection parmi : le support, la confiance et le lift.

Une fois les règles extraites, elles seront affichées via l'interface d'exploration visuelle interactive (Fig.4) composé de trois régions :

1. En haut à droite : la règle de référence.
2. En haut à gauche : un scatterPlot dans lequel les règles sont représentées par des sphères. Chaque sphère est positionnée selon les valeurs de mesures d'intérêt de la règle (support, confiance et lift).
3. En bas : les règles sont affichées sur un demi-cercle en utilisant la nouvelle métaphore de représentation de règles d'associations. Elles sont organisées de gauche à droite selon une mesure d'intérêt choisie par l'utilisateur.

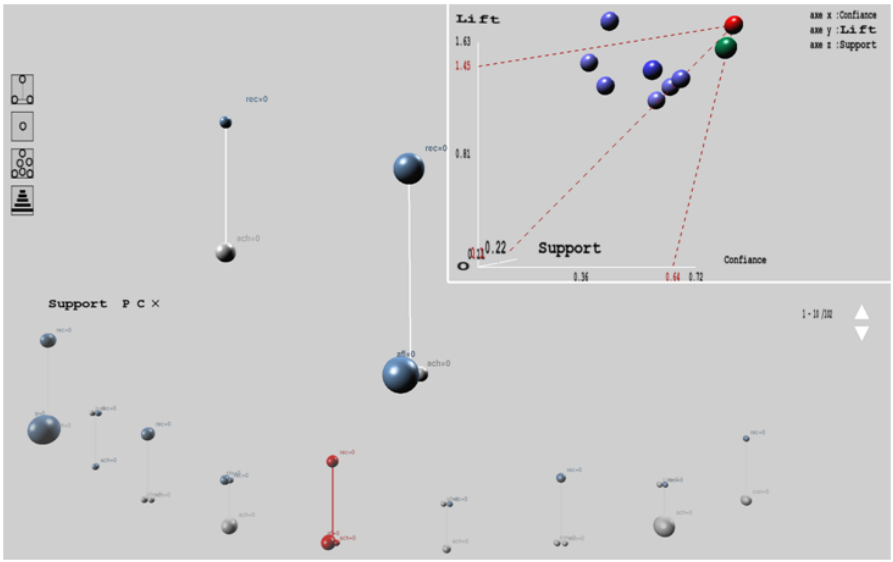


FIG. 4 – *Interface d'exploration visuelle des règles extraites*

5.3 Interface de visualisation de l'historique de navigation

Chaque fois que l'utilisateur juge une règle intéressante, il lui donne une note entre 1 et 3. Ces règles qui peuvent appartenir à différents sous-ensembles peuvent ainsi être visualisées et comparées dans l'interface historique. Les règles sont affichées sur une échelle composée de sept marches, selon une mesure d'intérêt choisie par l'utilisateur (Fig.5). L'affichage sur une échelle permet de mettre l'accent sur les règles de bonne qualité. Plus précisément, une règle placée sur la première marche de l'échelle à une mesure d'intérêt plus élevée qu'une règle placée sur la dernière marche.

Une échelle en miniature permet de visualiser l'ordre d'ajout des règles à l'historique.

6 Conclusion

Dans cette étude, nous avons proposé une nouvelle métaphore de visualisation de règle d'association permettant la visualisation des items qui la composent. En outre, cette métaphore montre les relations entre les items et la contribution de chacun d'entre eux à la règle. Mais, développer seulement une nouvelle métaphore visuelle est rarement suffisant pour faire de nouvelles découvertes. Dans le processus d'extraction de règles d'association, le décideur est submergé par les résultats des algorithmes d'extraction de règles d'association. Représenter ces résultats sous forme d'images statiques limite l'utilité de la visualisation. Cela explique pourquoi l'utilisateur doit être capable d'interagir avec la représentation de règles d'association afin de trouver des connaissances pertinentes. Pour cela, nous avons développé un nouveau outil de fouille visuelle de règles d'association afin d'intégrer l'utilisateur dans le processus

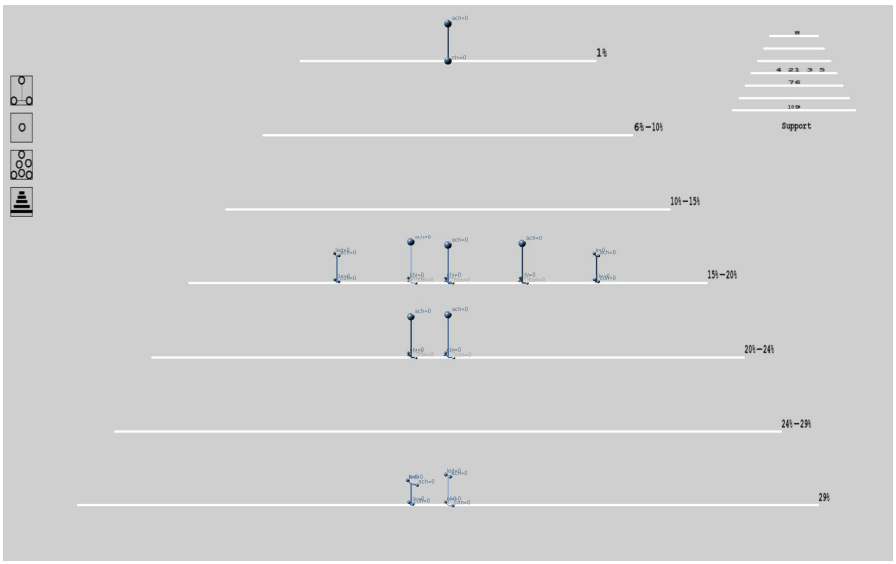


FIG. 5 – *Interface de visualisation de l'historique de navigation*

extraction. L'utilisateur doit être en mesure de manipuler un algorithme local d' extraction de règles et non seulement les représentations graphiques. Cela permet de se concentrer sur la connaissance intéressante du point de vue de l'utilisateur, afin de rendre les méthodes de règles d'association plus utiles pour les utilisateurs. Nos futurs travaux porteront principalement sur la validation fonctionnelle de IUCERVis.

Références

Agrawal, R., T. Imielinski, et A. Swami (1993). Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pp. 207 – 216.

Beale, R. (2007). Supporting serendipity : Using ambient intelligence to augment user exploration for data mining and web browsing. *International Journal of Human-Computer Studies 65*(5), 421–433.

Blanchard, J., B. Pinaud, P. Kuntz, et F. Guillet (2007). A 2d-3d visualization support for human-centered rule mining. In *Computers & Graphics*.

Brin, S., R. Motwani, et C. Silverstein (1997). Beyond market baskets : generalizing association rules to correlations. *ACM SIGMOD Record 26*, 265–276.

Chanda, P., J. Yang, A. Zhang, et M. Ramanathan (2010). On mining statistically significant attribute association information. *SIAM*, 141–152.

Cleveland, W. S. et R. McGill (1984). Graphical perception : Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association 79*(387), 531–554.

Ertek, G. et A. Demiriz (2006). A framework for visualizing association mining results. In *Proceedings of the 21st international conference on Computer and Information Sciences*, ISCIS 2006, pp. 593–602. Springer-Verlag.

Freitas, A. A. (1998). On objective measures of rule surprisingness. In *Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, PKDD '98, London, UK, pp. 1–9. Springer-Verlag.

Freitas, A. A. (2001). Understanding the crucial role of attribute interaction in data mining. *Artificial Intelligence Review 16*(3), 177–199.

Fule, P. et J. F. Roddick (2004). Experiences in building a tool for navigating association rule result sets. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation*, Volume 32, pp. 103 – 108.

Hendley, R. J., N. S. Drew, A. M. Wood, et R. Beale (1999). Narcissus : visualising information. In *Proceedings of the IEEE Symposium on Information Visualization (INFOVIS '95)*, pp. 90–96. Morgan Kaufmann Publishers Inc.

Holland, J. H., K. J. Holyoak, R. E. Nisbett, et P. R. Thagard (1986). *Induction : processes of inference, learning, and discovery*. Cambridge, MA, USA : MIT Press.

Liu, Y. et G. Salvendy (2006). Design and evaluation of visualization support to facilitate association rules modeling. *International Journal of Human-Computer Interaction 21*(1), 15–38.

Provost, F. J. et J. M. Aronis (1996). Scaling up inductive learning with massive parallelism. *Machine Learning 3*(1), 33–46.

Simoff, S. J., M. H. Bohlen, et A. Mazeika (2008). Visual data mining : An introduction and overview. In *Visual Data Mining*, pp. 1–12.

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press.

Summary

In order to discover knowledge from large amount of results generated by the association rules extraction algorithms, visual representations of association rules can be very beneficial to the user. Those representations support the user in finding and validating interesting knowledge. All techniques proposed for association rule visualization have been developed to represent association rule as a whole without paying attention to the relations between attributes and the contribution of each one. In this article, we propose a new visualization metaphor for association rules. This new metaphor represents attributes which make up the antecedent and the consequent, the contribution of each one to the rule, and the correlations between each pair of the antecedent and each pair of consequent. The use of this new representation in an association rules visual exploration tool will facilitate the task of exploration and extraction of relevant rules.

Index

Partenaires :