Analyse de graphes dynamiques issus d'un processus de réécriture de graphes

Bruno Pinaud

Analyse de graphes dynamiques issus d'un processus de réécriture de graphes

Bruno Pinaud
Université Bordeaux 1, LaBRI
Bât A30, 351 Cours de la Libération
33405 Talence Cedex, France
bruno.pinaud@labri.fr

INTRODUCTION

Je suis membre de l'équipe Modèles et Algorithmes pour la Bioinformatique et la Visualisation (MaBioVis) du Laboratoire Bordelais de Recherche en Informatique (LaBRI) et plus précisément du thème "Visualisation de grandes masses de données". La vision scientifique de l'équipe s'inspire de la formidable collaboration qui se met en place aujourd'hui entre les sciences de l'information, les sciences de la vie, les sciences sociales et les sciences économiques. La combinaison de techniques expérimentales à haut-débit et des méthodes computationelles avancées permet de traiter des problèmes de taille sans précédent, et a donné naissance à des questions émergentes, par exemple en biologie ou en géographie quantitative. Dans les différents domaines que nous étudions, les données peuvent se modéliser par des systèmes complexes, eux mêmes définis par un ensemble constitué d'un grand nombre d'entités en interaction.

Je m'intéresse plus particulièrement à la modélisation, la simulation et l'analyse de systèmes complexes obtenus par de la réécriture de graphes. Imaginez un jeu dans lequel des règles de transformation sont successivement appliquées sur un graphe jusqu'à atteindre une condition d'arrêt. Une règle décrit un motif local (un sous-graphe) qui doit être identifié dans le graphe et comment transformer ce motif. Le formalisme de la réécriture de graphes est à la fois très riche et complexe rendant l'étude d'un système utilisant ce formalisme difficile. Par exemple, prédire si une suite de règles est applicable dans n'importe quel ordre est bien souvent un problème difficile. Pour la modélisation de systèmes complexes, les formalismes graphiques ont des avantages certains : ils sont intuitifs et rendent plus facile le raisonnement sur le système. Nous avons donc développé la plate-forme PORGY[1] Figure 1) [3]. Elle est le résultat de trois ans de travail collaboratif avec des experts des systèmes de réécritures. La méthodologie suivie est largement inspirée du travail de Munzner sur le développement et la validation de plate-formes de visualisation [2].

Dans la suite, le terme "graphe" désigne le graphe sur lequel les règles de réécritures sont appliquées. Après avoir rapidement introduit notre plate-forme et la réécriture, je ferais le lien avec les données temporelles et les problèmes posés par le formalisme que nous utilisons.

LES BASES DE LA RÉÉCRITURE DE GRAPHE

Chaque modification autorisée (topologie, attributs des sommets/arêtes) sur un graphe donné G est appelée une *règle de réécriture*. Elles servent à modéliser les connaissances de l'expert sur le système étudié et décrivent comment un (petit) sous-graphe de G doit être modifié ou *réécrit*. La figure 1, partie 2 est un exemple de règle. Le dessin permet de déduire immédiatement l'effet de cette règle : si un graphe contient un sous graphe avec cinq sommets dans la configuration donnée par la règle, alors la connexion entre les sommets bleu et vert est supprimée. Des modifications plus complexes sont réalisées en combinant des séquences de règles avec des opérateurs spécifiques dans une *stratégie* qui décrit qui/quand/où/comment appliquer les règles [1].

LIEN AVEC LES DONNÉES TEMPORELLES

Porgy permet dans un premier temps de concevoir et d'éditer graphiquement des règles. Le challenge principal est alors de savoir si la règle ainsi construite est un modèle correct du système étudié. Porgy permet alors de simuler l'évolution du système après avoir défini une stratégie de réécriture (voir la figure 2). L'historique des calculs est géré par l'arbre de dérivation (figure 1, partie 4). Ses sommets sont les différents états pris par le graphe en cours de réécriture. Une arête noire indique une application de règle alors qu'une arête verte indique le point de départ et l'état final d'une stratégie. Chaque sommet de l'arbre de dérivation peut servir de point de départ pour l'application d'une nouvelle stratégie et ainsi créer une nouvelle branche.

La complexité du système de réécriture est d'une certaine façon capturée par l'arbre de dérivation, qui est donc un objet central pour l'étude d'un système de réécriture. Si on considère le graphe sur lequel les règles sont appliqués comme un graphe dynamique, l'arbre de dérivation regroupe alors l'ensemble des transformations possibles de ce graphe (une sorte de *story-board*). Les branches de

[1] Voir `http://tulip.labri.fr/TulipDrupal/?q=porgy`.

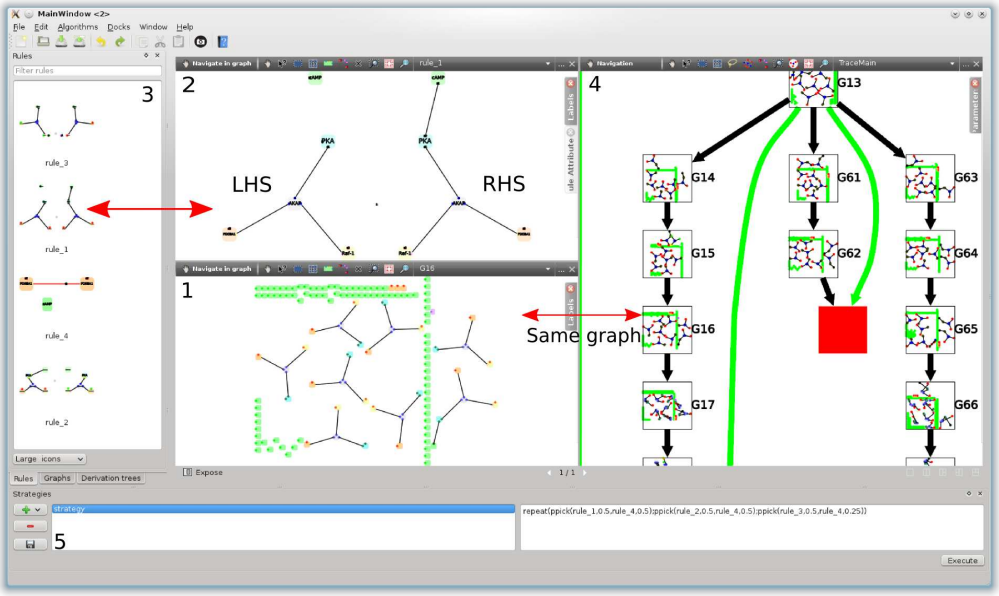
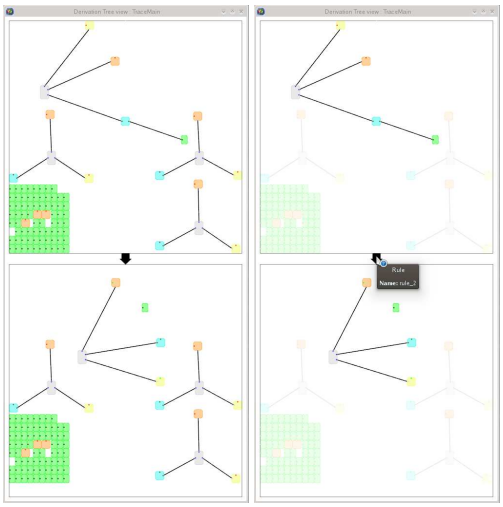


Figure 1. Vue d'ensemble de PORGY: (1) édition d'un état du graphe qui est en cours de réécriture ; (2) édition d'une règle ; (3) les règles de réécriture disponibles ; (4) une partie de l'arbre de dérivation, un historique complet des calculs effectués ; (5) l'éditeur de stratégie.



(a) Vue classique (b) Visualisation des élements modifiés par la réécriture

Figure 2. Une étape de réécriture (règle de la figure 1) avant et après avoir passé le pointeur de la souris sur l'arête qui relie le graphe d'origine (celui du haut) au graphe résultat (bas).

l'arbre (qui peut être infini) sont à considérer comme autant de monde parallèles rendant l'étude et la visualisation du système très difficile.

L'étude d'un système de réécriture nécessite de passer en permanence de vues locales (les règles, les graphes) à une vue globale de l'arbre de dérivation. Le principal enjeu est alors de comprendre comment le comportement global du système est dirigé par les règles qui définissent des modifications uniquement locales et une stratégie qui pilote l'application de ces règles. C'est en regardant précisément les graphes dans l'arbre de dérivation tant globalement que localement qu'un expert du domaine peut apprécier l'adéquation du modèle avec le système.

A notre connaissance, il n'existe aucun algorithme de tracé de graphe qui produise de bons tracés quelle que soit les données utilisées. De plus, cet algorithme de dessin devrait garantir une stabilité relative du dessin entre toutes les branches de l'arbre de dérivation pour conserver au moins à minima la carte mentale de l'utilisation. De plus, l'évolution du graphe est difficile à prédire car les modifications sont pilotées uniquement par l'application des règles.

CONCLUSION

La combinaison à notre connaissance unique de la réécriture de graphe et de la visualisation soulève donc de nombreux problèmes. Il nous reste à traiter notamment le cas de règles de taille importante (problème isomorphisme graphe/sous-graphe) ou encore les problèmes de stabilité dans le tracé des graphes dynamiques. Nous avons surtout utilisé pour le moment des exemples basés sur la bio-informatique mais nous sommes à la recherche de nouveaux domaines d'applications (nous pensons aux sciences sociales notamment).

BIBLIOGRAPHIE

1. Fernandez, M., Kirchner, H., and Namet, O. A strategy language for graph rewriting. In *Logic-Based Program Synthesis and Transformation*, vol. 7225 of *LNCS*, Springer (2012), 173–188.
2. Munzner, T. A nested process model for visualization design and validation. *IEEE Trans. on Visualization and Computer Graphics 15*, 6 (2009), 921–928.
3. Pinaud, B., Melançon, G., and Dubois, J. PORGY: A Visual Graph Rewriting Environment for Complex Systems. *Computer Graphics Forum 31*, 3 (2012), 1265–1274.