Formulation linéaire en nombres entiers pour le problème de la distance d'édition entre graphes

Mostafa Darwiche, Donatello Conte, Romain Raveaux, Vincent t'Kindt

Formulation linéaire en nombres entiers pour le problème de la distance d'édition entre graphes

Mostafa Darwiche[1,2], Donatello Conte[1], Romain Raveaux[1], Vincent T'kindt[2]

[1] Laboratoire d'Informatique (EA 6300), Université François Rabelais Tours, France
[2] Laboratoire d'Informatique (EA 6300), ERL-CNRS 6305, Université François Rabelais Tours, France
{mostafa.darwiche,romain.raveaux,donatello.conte,tkindt}@univ-tours.fr

Mots-clés : *problème de la distance d'édition, appariement de graphes, programmation linéaire en nombres entiers.*

1 Introduction

Les graphes permettent une représentation structurelle très intéressante des objets. Un graphe est défini par deux ensembles : les nœuds qui représentent les composants principaux de l'objet, et les arêtes qui tracent les relations entre les composants. Les nœuds et les arêtes de ces graphes peuvent posséder plusieurs attributs (des valeurs numériques, catégories, ...) qui ajoutent des informations et caractéristiques à cette représentation. Ce type de graphes apparait dans plusieurs domaines, comme la vision par ordinateur, pour effectuer des tâches comme : la reconnaissance de formes ou le suivi d'objets dans des vidéos [5]. En chimie et en biologie on peut comparer des molécules chimiques, des protéines ou des enzymes [4]. Toutes ces applications exigent d'avoir un moyen pour apparier les graphes ou une mesure de (dis)similarité entre graphes. Le problème de la distance d'édition entre graphes (DEG) fournit une mesure de dissimilarité entre deux graphes. Considérant deux graphes G et G', l'idée est de transformer le premier graphe pour obtenir le second en effectuant un ensemble d'opérations d'édition. Les opérations d'édition possibles sont : substitutions, suppressions et insertions de nœuds ou d'arêtes, dont un coût est associé à chaque opération. Résoudre le problème DEG revient alors à trouver l'ensemble des opérations d'édition dont le coût total est minimum. Ce problème est un problème d'optimisation NP-difficile [6]. Dans la littérature, plusieurs méthodes heuristiques sont conçues pour résoudre le problème DEG. Par contre dans le contexte exacte il existe seulement trois formulations mathématiques : un modèle quadratique qui est proposé dans [1] et deux modèles linéaires en nombres entiers dans [2, 3].

Dans ce travail, nous proposons une formulation étendue sous forme d'un programme linéaire en nombres entiers pour ce problème.

2 Modèles mathématiques pour le problème DEG

Soient deux graphes $G = (V, E, \mu, \zeta)$ comme source et $G' = (V', E', \mu', \zeta')$ comme cible, avec μ (resp. ζ) la fonction qui affecte les attributs aux nœuds (resp. arêtes). La distance d'édition $d_{min}(G, G')$, entre G et G', est définie par :

$$d_{min}(G, G') = \min_{\lambda \in \Gamma(G,G')} \sum_{e_i \in \lambda(G,G')} \ell(e_i) \tag{1}$$

où $\Gamma(G, G')$ est l'ensemble de tous les ensembles d'opérations d'édition possibles pour transformer G en G'. Pour un ensemble $\lambda = \{e_1, ..., e_n\} \in \Gamma(G, G')$, ℓ est la fonction qui calcule le coût des opérations.

L'idée de la formulation proposée est basée sur la notion d'objets qui sont de la forme : $o_p = (u_i, u_j, v_k, v_l)$, tel que $(u_i, u_j) \in \overline{V} \times \overline{V}$ et $(v_k, v_l) \in \overline{V}' \times \overline{V}'$, $\overline{V} = V \cup \{N\}$ et $\overline{V}' = V' \cup \{N\}$. N est un nœud fictif pour représenter les opérations de suppression et d'insertion. Par exemple, si $u_i \in V$ est apparié avec le nœud N ($u_i \to N$) cela signifie que u_i est supprimé de G. Alors, chaque objet o_p représente les opérations suivantes : $u_i \to v_k$, $u_j \to v_l$ et l'appariement des arêtes est induit. Si $(u_i, u_j) \in E$ et $(v_k, v_l) \in E'$ donc les arêtes sont appariées. Par contre, si $(u_i, u_j) \in E$ et $(v_k, v_l) \notin E'$, l'arête (u_i, u_j) est supprimée de G (la contraire est l'insertion de l'arête dans G). Ensuite, on crée l'ensemble $\mathcal{O}$ qui contient tous les objets possibles et modélise toutes les opérations qui peuvent s'appliquer. Pour le moment, on considère le cas où les graphes ne sont pas orientés, mais le modèle peut s'adapter pour tels graphes. La fonction objective à minimiser est :

$$\min_{x,y} \sum_{o_p \in \mathcal{O}} C_p^o x_p + \sum_{\forall u_i \in \overline{V}, \forall v_k \in \overline{V}'} C_{i,k}^v y_{i,k} \tag{1}$$

avec x_p des variables booléennes qui représentent les objets o_p, et $y_{i,k}$ des variables booléennes qui représentent l'appariement des deux nœuds $u_i \in \overline{V}, v_k \in \overline{V}'$. C_p^o et $C_{i,k}^v$ sont les coûts associés.

Pour les contraintes, on a besoin d'une contrainte qui vérifie qu'une arête $(u_i, u_j) \in E$ est appariée une seule fois. La même contrainte est donc ajoutée pour les arêtes $(v_k, v_l) \in E'$. En outre, une contrainte est nécessaire pour interdire les appariements d'un nœud avec plusieurs nœuds, e.g. $o_p = (u_i, u_j, v_k, v_l)$ et $o'_p = (u_i, u'_j, v'_k, v'_l)$ ne peuvent pas être sélectionnés car u_i ne peut pas être apparié en même temps avec v_k et v'_k. Donc, la troisième contrainte (contrainte de disjonction) est construite en cherchant tous les objets de la forme o_p et o'_p et disant que seulement un des deux peut être sélectionné dans la solution optimale. Le problème d'une telle approche est qu'il y a un grand nombre de contraintes disjonctives. Nous proposons donc un algorithme pour les fusionner. Pour cela, nous construisons un graphe dont les nœuds sont les objets et les arêtes représentent les disjonctions entre eux. Pour générer les contraintes du PLNE, il suffit de résoudre le problème de calcul des cliques maximales.

Cette formulation est intéressante parce qu'au contraire d'autres formulations, elle raisonne sur les appariements des arêtes avant les nœuds. Elle est dans sa phase d'implémentation et des résultats seront présentés.

Références

[1] Sébastien Bougleux, Luc Brun, Vincenzo Carletti, Pasquale Foggia, Benoît Gaüzère, and Mario Vento. Graph edit distance as a quadratic assignment problem. *Pattern Recognition Letters*, pages –, 2016.

[2] Derek Justice and Alfred Hero. A binary linear programming formulation of the graph edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8) :1200–1214, Aug 2006.

[3] Julien Lerouge, Zeina Abu-Aisheh, Romain Raveaux, Pierre Héroux, and Sébastien Adam. New binary linear programming formulation to compute the graph edit distance. *Pattern Recognition*, 72 :254–265, 2017.

[4] John W Raymond and Peter Willett. Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *Journal of computer-aided molecular design*, 16(7) :521–533, 2002.

[5] Alberto Sanfeliu, René Alquézar, J Andrade, Joan Climent, Francesc Serratosa, and J Vergés. Graph-based representations and techniques for image processing and image analysis. *Pattern recognition*, 35(3) :639–650, 2002.

[6] Zhiping Zeng, Anthony KH Tung, Jianyong Wang, Jianhua Feng, and Lizhu Zhou. Comparing stars : on approximating graph edit distance. *Proceedings of the VLDB Endowment*, 2(1) :25–36, 2009.