Formulations pour le problème de plus grand graphe partiel commun

Etienne de Gastines, Arnaud Knippel

INSA Rouen Normandie, Laboratoire de Mathématiques, Rouen, France

{etienne.mace_de_gastines, arnaud.knippel}@insa-rouen.fr

Mots-clés : *plus grand graphe partiel commun, polyèdres, programmation linéaire 0-1*

1 Introduction

Le problème du plus grand graphe partiel commun (Maximum Common Edge Subgraph - MCES) a été introduit en premier par Bokhari [5], pour résoudre le problème d'affectation de tâches à des processeurs tout en maximisant les demandes de communications entre tâches. Ce problème donne également une mesure de similarité entre deux graphes et est utilisé comme outil pour comparer des molécules.

Il peut être énoncé comme suit :

Soit $G = (V_G, E_G)$ et $H = (V_H, E_H)$ deux graphes. $S = (V_S, E_S)$ est un graphe partiel de G si $V_S \subseteq V_G$ et $E_S \subseteq V_G$. Si un graphe partiel de G est isomorphe à un graphe partiel de H, alors on le qualifie de graphe partiel commun à G et H. On recherche un graphe partiel commun à G et H maximal en terme de nombre d'arêtes. Nous considérons ici les graphes G et H comme étant simples et non-orientés.

Le problème est $\mathcal{NP}$-difficile et généralise les problèmes d'isomorphisme de sous-graphe, de clique maximale et de chaînes hamiltoniennes.

La première formulation en nombres entiers a été proposée par Almohamad et Duffuaa [1] (AD) dans le cadre du problème plus général de graph matching. Elle introduit les variables $x_{u,v}$ $\forall u \in V_G, v \in V_H$ valant 1 si le sommet u est associé au sommet v, 0 sinon, et des variables comptant le nombre d'arêtes n'appartenant pas au graphe partiel commun. Une autre formulation en nombres entiers fut proposée indépendemment par Marenco et Loiseau [7], [8] (ML), qui utilise les mêmes variables x et introduit les variables y_{u_1,u_2} $\forall u_1u_2 \in E_G$ valant 1 si l'arête u_1u_2 appartient au sous graphe commun, 0 sinon. Une troisième formulation a été proposée plus tard par Bahiense, Manić, Piva, et de Souza [3], [4] (BMPS), qui utilise les mêmes variables x, mais introduit pour sa part les variables $c_{u_1u_2,v_1v_2}$ $\forall u_1u_2 \in E_G, v_1v_2 \in E_H$, valant 1 si l'arête u_1u_2 est associée à l'arête v_1v_2.

2 Nouvelles formulations

Nous proposons cinq nouvelles formulations : une version symétrisée de la formulation proposée par [3] (sym. BMPS) avec une meilleure relaxation, deux versions réduites (R1 et R2) diminuant progressivement le nombre de variables de cette formulation. Nous proposons également une reformulation de la formulation proposée par [1] avec moins de variables, et montrons qu'elle possède la même relaxation que notre formulation réduite la plus compacte. Nous proposons une autre formulation (D) encore plus compacte qui introduit des variables sur le degré des sommets du graphe partiel commun.

3 Résultats numériques

Nous avons testé numériquement les nouvelles formulations obtenues avec les formulations de la littérature sur deux bases de données, une base de données de graphes issus de problèmes rassemblés par Marenco et Loiseau ([8]) et provenant essentiellement d'applications pratiques, et une partie de la base de donnés ARG ([6], [2]) composées de graphes générés aléatoirement selon divers modèles. Nous observons que les performances varient selon la structure et la densité des instances. De nos tests numériques, il ressort que R2 est la plus performante sur les instances les plus denses, et domine R1. Les formulations BMPS et sym. BMPS ont des performances comparables, et sont plus performantes sur certaines des instances les plus creuses. Bien qu'elle ne les résolvent pas, sym. BMPS obtient les meilleurs gaps sur les instances composées de mailles les plus difficiles. Sur le premier jeu de données, la formulation ML est plus rapide sur certaines instances de taille petite et moyennes. La formulation D est plus performante uniquement sur certaines petites instances. De manière générale la formulation R2 est la plus rapide en moyenne et résout le plus d'instances. De plus, elle semble avoir le meilleur comportement sur les instances les plus difficiles et obtient souvent le meilleur gap sur les instances non résolues.

TAB. 1 – Taille asymptotique des formulations et nombre d'instances résolues.

	n. variables	n. cons	instances résolues
AD	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	140
ML	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	$\mathcal{O}(\lvert E_G\rvert\lvert V_H\rvert)$	137
BMPS	$\mathcal{O}(\lvert E_G\rvert\lvert E_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert E_H\rvert + \lvert V_H\rvert\lvert E_G\rvert)$	134
sym. BMPS	$\mathcal{O}(\lvert E_G\rvert\lvert E_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert E_H\rvert + \lvert V_H\rvert\lvert E_G\rvert)$	136
R1	$\mathcal{O}(\lvert V_G\rvert\lvert E_H\rvert + \lvert E_G\rvert\lvert V_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	143
R2	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	146
D	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	$\mathcal{O}(\lvert V_G\rvert\lvert V_H\rvert)$	122

Références

[1] H. A. Almohamad and S. O. Duffuaa, *A Linear Programming Approach for the Weighted Graph Matching Problem*, IEEE transactions on pattern analysis and machine intelligence, **15**, (1993), 522–525

[2] ARG Database, https ://mivia.unisa.it/datasets/graph-database/arg-database/

[3] Bahiense, L. and Piva, B. and de Souza, C., *A branch&cut algorithm for the maximum common edge subgraph problem*, Electronic Notes in Discrete Mathematics **35** (2009), 47–52.

[4] Bahiense, L. and Manić, G and Piva, B. and de Souza, C., *The maximum common edge subgraph problem : A polyhedral investigation*, Discrete Applied Mathematics **160** (2012), 2523–2541.

[5] Bokhari, *On the Mapping Problem*, IEEE Transactions on Computers **3** (1981), 207–214.

[6] De Santo, M., Foggia, P., Sansone, C. and Vento, M., *A large database of graphs and its use for benchmarking graph isomorphism algorithms*, Pattern Recognition Letters **24** (2003), 1067–1079.

[7] Marenco, Javier and Loiseau, Irene, "A branch&cut algorithm for a problem arising in parallel programming environments ", Universidade de Buenos Aires, Departamento de Computación,2000.

[8] Marenco, Javier and Loiseau, Irene, "Un algoritmo branch-and-cut para el problema de mapping ", Master thesis, Universidade de Buenos Aires, Departamento de Computación,1999.