Techniques de filtrage pour le problème d'isomorphisme de sous-graphes : comparaison des approches PPC et PLNE

Etienne de Gastines, Arnaud Knippel

INSA Rouen Normandie, Laboratoire de Mathématiques, Rouen, France
{etienne.mace_de_gastines, arnaud.knippel}@insa-rouen.fr

Résumé : *Nous étendons la notion de filtrage de la programmation par contraintes pour certaines formulations de programmation linéaire en nombres entiers de façon à pouvoir comparer la force de filtrage des deux approches dans le cas du problème de l'isomorphisme de sous-graphes. Nous montrons qu'une des formulations PLNE que nous proposons possède un filtrage plus fort que celui de la procédure de filtrage des degrés itérés (Iterative Label Filtering - ILF). Nous en profitons pour affiner l'estimation de la complexité de la procédure ILF. Les résultats numériques montrent la supériorité de l'approche de programmation par contraintes sur ce problème, mais la comparaison des filtrages laisse augurer des améliorations possibles.*

Mots-clés : *PLNE, PPC, Isomorphisme de sous-graphes, filtrage*

1 Introduction

Avant d'étendre la notion de filtrage dans le contexte de la programmation linéaire en nombres entiers, nous introduisons le problème de l'isomorphisme de sous-graphes pour lequel nous allons comparer l'approche de PLNE et l'approche de programmation par contraintes (PPC). Le choix d'un problème de décision pure avantage à priori la programmation par contraintes par rapport à la programmation linéaire en nombres entiers mais illustre bien notre démarche de comparaison de la force des filtrages.

Le problème d'isomorphisme de sous-graphe consiste à déterminer, étant donné un graphe motif et un graphe cible, si le graphe motif est contenu dans le graphe cible. Ce problème se rencontre dans de nombreuses applications, comme la comparaison de composés chimiques ou la recherche de motifs dans des bases de données. Ce problème se décline en deux variantes non-équivalentes selon que l'on considère des sous-graphes induits ou non-induits. Nous nous intéresserons ici uniquement au problème de l'isomorphisme de sous-graphes non-induit. Plus formellement, la notion d'isomorphisme de sous-graphes se formule de la manière suivante :

Définition 1 (Isomorphisme de sous-graphes non-induits) *Soit $G(V_G, E_G)$ et $H(V_H, E_H)$ deux graphes. Le problème de l'isomorphisme de sous-graphes non-induits consiste à déterminer s'il existe un graphe partiel de H isomorphe à G. Cela revient à déterminer s'il existe une injection $\varphi : V_G \to V_H$ telle que pour tout $u_1, u_2 \in V_G$, $u_1u_2 \in E_G \implies \varphi(u_1)\varphi(u_2) \in E_H$.*

Dans la suite, nous ne considérons que des graphes non orientés. Nous notons V_G et E_G l'ensemble des sommets et des arêtes d'un graphe G. Nous notons $N(u)$ l'ensemble des sommets adjacents à u et $d(u) = |N(u)|$ le degré de u. Nous définissons également le voisinage d'un ensemble $A \subseteq V_G$ par $N(A) = \bigcup_{u \in A} N(u)$.

La notion de filtrage est couramment utilisée dans le contexte de la programmation par contraintes. Pour comparer la capacité d'un modèle de programmation par contrainte à éliminer des branches, nous proposons de comparer la force de leurs filtrages :

Définition 2 *Soit M_1 et M_2 deux modèles de programmation par contraintes reposant sur les mêmes variables. Un modèle filtre une valeur v du domaine d'une variable x si la propagation des contraintes permet d'établir qu'il n'existe aucune solution telle que $x = v$. Le modèle M_1*

exerce un filtrage supérieur au modèle M_2 s'il maintient une cohérence plus stricte de ces domaines de variables, c'est à dire qu'une fois la propagation des contraintes réalisées, pour toute variable x et pour toute valeur v du domaine $D(x)$ de x, si $v \in D(x)$ pour le modèle M_1, alors $v \in D(x)$ pour le modèle M_2.

Pour comparer la capacité de formulations PLNE à éliminer des branches dans l'arborescence d'énumération, nous proposons de comparer leurs relaxations :

Définition 3 *Soit F_1 et F_2 deux formulations PLNE reposant sur les mêmes variables. La formulation F_1 exerce un filtrage supérieur à la formulation F_2 si toute solution de la formulation relaxée de F_1 est également une solution de la formulation relaxée de F_2. Cela revient à étudier l'inclusion des polyèdres des solutions représentées par la relaxation de ces formulations.*

Il serait intéressant de pouvoir comparer la capacité d'élagage d'un modèle de programmation par contraintes par rapport à un modèle PLNE. Nous proposons d'étendre la notion de domaine au cas de certains PLNE.

Définition 4 *Soit x une variable prenant ses valeurs dans un domaine $D(x)$ fini. Soit F une formulation PLNE modélisant cette variable x par un ensemble de variables binaires $(x_v)_{v \in D(x)}$ indiquant si la valeur prise par x est la valeur v. Ces variables sont reliées par la relation $\sum_{v \in D(x)} x_v = 1$ dans la formulation F. On définit $D_F(x)$ l'ensemble des valeurs $v \in D(x)$ telle qu'il existe une solution de la formulation relaxée F pour laquelle $x_v > 0$.*

Cette définition fait sens puisque pour tout $v \notin D_F(x)$, on aura à tous les noeuds de l'arborescence du Branch and Cut $x_v = 0$. On peut alors considérer que la valeur v à été filtrée par la formulation.

On peut donc désormais comparer un modèle de programmation par contraintes M et une formulation PLNE F modélisant les mêmes variables en comparant les domaines après propagation des contraintes du modèle M avec les domaines D_F de la formulation F.

Dans la suite nous allons appliquer ces idées sur le problème de l'isomorphisme de sous-graphes. La section 2 présente la formulation linéaire en nombre 0-1 que nous considérons et que nous améliorerons. Nous présentons en section 3 la méthode de filtrage des degrés itérés en programmation par contraintes, en nous appuyant sur [7].

Les filtrages obtenus sont comparés dans la section 4 et nous concluons avec des résultats numériques.

2 Formulations PLNE

Deux formulations de PLNE sont proposées dans [1]. Nous utilisons la seconde qui fournit une meilleure relaxation continue et la notons **LBK**, en écartant la fonction objectif qui concernait une variante numérique du problème.

$$\text{Trouver} \quad x, y \tag{1}$$

$$\text{t.q.} \quad \sum_{v \in V_T} x_{u,v} = 1, \quad \forall u \in V_P \tag{2}$$

$$\sum_{u \in V_P} x_{u,v} \leq 1, \quad \forall v \in V_T \tag{3}$$

$$\sum_{v_2 \in N(v_1)} y_{u_1u_2,v_1v_2} = x_{u_1,v_1} \quad \forall u_1u_2 \in E_P, v_1 \in V_T \tag{4}$$

$$x_{u,v} \in \{0,1\} \quad \forall u \in V_P, \forall v \in V_T \tag{5}$$

$$y_{u_1u_2,v_1v_2} \in \{0,1\} \quad \forall u_1u_2 \in E_P, \forall v_1v_2 \in E_T \tag{6}$$

Nous avons proposé dans [2] une nouvelle formulation plus forte en ajoutant une nouvelle famille de contraintes.

En s'inspirant des contraintes du problème, on peut remarquer que la famille d'inégalités suivante est valide :

$$\sum_{u_2 \in N(u_1)} y_{u_1u_2,v_1v_2} \leq x_{u_1,v_1} \quad \forall u_1 \in V_P, v_1v_2 \in E_T \tag{7}$$

Il est à noter qu'il n'y a plus cette fois-ci d'égalité puisque l'on n'impose pas que toutes les arêtes de T soient associés à des arêtes de P.

Nous appellerons **LBK+** la formulation **LBK**, augmentée de ces inégalités.

Nous introduisons une formulation compacte que nous nommons **FC** :

$$\begin{aligned}
\text{Trouver} \quad & x & (8)\\
\text{t.q.} \quad & (2), (3) & \\
& \sum_{u_1 \in N(u_2)} x_{u_1,v_1} \leq \sum_{v_2 \in N(v_1)} x_{u_2,v_2}, \quad \forall u_2 \in V_P, v_1 \in V_T & (9)\\
& x_{u,v} \in \{0,1\} \quad \forall u \in V_P, \forall v \in V_T & (10)
\end{aligned}$$

La formulation **FCmult** est composée des contraintes de la formulation **FC** et de contraintes suivantes :

$$\sum_{v_1 \in K} x_{u_1,v_1} \leq \sum_{v_2 \in N(K)} x_{u_2,v_2} \ \ \forall u_1u_2 \in E_P, K \subset V_T. \tag{11}$$

La formulation **LBKC** est composée des contraintes de la formulation **LBK** et des contraintes (9).

Il est démontré dans [2] que le polyèdre de la relaxation continue de **FCmult** est la projection du polyèdre de la relaxation continue de **LBKC** sur l'espace des variables x.

Nous utiliserons la formulation **FC** qui est plus compacte que **LBKC** pour nos expériences numériques, mais la formulations **LBKC** interviendra dans la preuve du théorème 2 de la section 4.

3 Filtrage des degrés itérés

Nous présentons dans cette section la technique de filtrage des degrés itérés (Iterated Label Filtering - **ILF**), introduite par les auteurs de [7]. Nous simplifions en ne considérant pas d'affinement d'étiquetage ni de comparaison de multi-ensembles de degrés. La prise en compte de graphes étiquetés se généralise naturellement aux travaux présentés dans cette section, et la comparaison des multi-ensembles n'est pas critique dans cet algorithme dans le sens où elle n'améliore pas la complexité asymptotique.

Pour chaque sommet u de P, nous introduisons une variable x_u, de domaine $D_u \subseteq V_T$ correspondant à l'ensemble des sommets de T pouvant être associés à u. Pour chaque paire de sommets $(u, v) \in V_P \times V_T$:

$$v \notin D_u \implies \text{il n'existe pas d'isomorphisme de sous-graphes associant } u \text{ à } v$$

On dira que u et v sont compatibles si $v \in D_u$, et incompatibles sinon.

ILF se base sur l'observation suivante : Si le degré d'un sommet $u \in V_P$ est supérieur au degré de $v \in V_T$, alors u et v ne sont pas compatibles. La procédure **ILF** cherche à généraliser cette observation. Elle commence en construisant un graphe de valeurs pour chaque paire de sommets $(u, v) \in V_P \times V_T$:

Définition 5 (graphe de valeurs) *Soit P et T deux graphes. pour tout $(u, v) \in V_P \times V_T$, on définit $G_{u,v}(N(u), N(v), E_G)$ le graphe biparti complet avec l'ensemble d'arêtes $E_G = \{(u', v') \in N(u) \times N(v) | v' \in D_{u'}\}$.*

On appelle couplage complet tout couplage de $G_{u,v}$ couvrant tous les sommets $N(u)$ dans $G_{u,v}$. S'il n'existe pas de couplage complet dans $G_{u,v}$, cela signifie que l'on ne peut pas associer tous les voisins de u à des voisins de v, et que l'on peut donc retirer v du domaine D_u.

Notons que cela généralise bien la comparaison des degrés. Si initialement $D_u = V_T$ pour tous les sommets $u \in V_P$, alors $G_{u,v}$ est un graphe biparti complet et il existe un couplage complet de $G_{u,v}$ si et seulement si le degré de u est supérieur au degré de v.

On peut maintenant définir la procédure **ILF**.

Définition 6 (procédure de filtrage des degrés itérés) *Soit P et T deux graphes avec des domaines initiaux D_u pour chaque sommet $u \in V_P$. La procédure **ILF** consiste à filtrer les domaines en utilisant les graphes de valeurs jusqu'à obtenir un point fixe.*

Remarquons qu'il est toujours possible de démarrer cet algorithme en initialisant pour tous les sommets $u \in V_P$ le domaine correspondant à $D_u = V_T$, mais il est également possible de commencer à partir de domaines arbitraires, par exemple en définissant comme incompatibles deux sommets qui ne partagent pas la même étiquette. Il est également possible de filtrer initialement des valeurs en utilisant d'autres techniques de filtrage. Il est par exemple possible de comparer deux sommets en comparant le nombre de triangles adjacents à ces sommets.

Les auteurs de [7] ont analysé la complexité de la procédure de filtrage des degrés itérés, et ont montré que la complexité d'une étape de filtrage est de $\mathcal{O}(|V_P||V_T|d^{5/2})$ avec d le degré maximal du graphe. Cette procédure est équivalente au filtrage de LAD [6]. Les auteurs montrent que la complexité globale de la procédure de filtrage est en $\mathcal{O}(|V_P||V_T|\Delta_P^2\Delta_T^2)$ avec Δ_P et Δ_T les degrés maximaux des graphes P et T.

Nous présentons une analyse plus fine de la complexité globale de la procédure de filtrage :

Proposition 1 *La procédure de filtrage des degrés itérés peut être exécutée avec une complexité $\mathcal{O}(|E_P||E_T|\Delta_P\Delta_T)$ avec Δ_P et Δ_T les degrés maximaux des graphes P et T.*

Preuve : Il n'est pas nécessaire de ré-exécuter un problème de couplage à chaque étape de la procédure de filtrage. On peut stocker pour chaque paire de sommets $(u, v) \in V_P \times V_T$ le couplage dans le graphe $G_{u,v}$ trouvé en dernier. Ce couplage ne doit être mis à jour que si l'une des paires de sommets $(u_2, v_2) \in V_P \times V_T$ avec $u_2 \in N(u)$ et $v_2 \in N(v)$ est filtrée. Cela se produit au plus $|E_{G_{u,v}}| \leq d(u)d(v)$ fois au cours de la procédure de filtrage. A chaque fois que cela arrive, il suffit seulement de trouver un chemin augmentant pour trouver un nouveau couplage complet (ou prouver qu'il n'en existe pas). La complexité pour trouver un chemin augmentant est en $\mathcal{O}(|E_{G_{u,v}}|) = \mathcal{O}(d(u)d(v))$. La complexité totale sur la procédure de filtrage pour le calcul de couplages dans le graphe $G_{u,v}$ est donc en $\mathcal{O}(d(u)^2d(v)^2)$. La complexité du calcul des couplages complets lors de la première étape est dominée par la mise à jour des chemins augmentants. La complexité globale de la procédure de filtrage se trouve donc en sommant sur toutes les paires de sommets $(u, v) \in V_P \times V_T$. On obtient :

$$\mathcal{O}\left(\sum_{(u,v)\in V_P\times V_T} d(u)^2d(v)^2\right) = \mathcal{O}(|E_P||E_T|\Delta_P\Delta_T)$$

□

Il est à noter que cette complexité est obtenue dans le pire des cas, et qu'en pratique, un point fixe est obtenu beaucoup plus rapidement. De plus, si les graphes ne sont pas initialement étiquetés et que l'on n'initialise pas préalablement les domaines avec des valeurs déjà filtrées, la première étape est beaucoup plus rapide à calculer car elle consiste seulement à comparer les degrés des sommets pour établir leur compatibilité.

4 Comparaison de ILF avec les modèles PLNE

Nous allons comparer les filtrages produits par la procédure **ILF** et par les formulations PLNE. Nous montrons d'abord que le filtrage réalisé par **ILF** est plus faible que celui réalisé par **LBK+**.

Théorème 1 *Soit P et T deux graphes. Soit $u \in V_P, v \in V_T$ tels que $v \notin D_u$ à l'issue de la procédure de filtrage des degrés itérés. Alors pour tout point (x, y) appartenant au polytope de la formulation **LBK+**, on a $x_{u,v} = 0$.*

Preuve : Nous montrons par récurrence sur les étapes de la procédure de filtrage des degrés itérés que pour toute paire de sommets $(u, v) \in V_P \times V_T$, $v \notin D_u \implies x_{u,v} = 0$.

A l'étape initiale de l'algorithme, pour tous sommets $u \in V_P$, nous avons $D_u = V_T$ donc la propriété est trivialement vérifiée.

Soit $(u_1, v_1) \in V_P \times V_T$ et $G_{u,v}(N(u_1), N(v_1), E_G)$ le graphe biparti tel que $E_G = \{(u_2, v_2) \in N(u_1) \times N(v_1) | v_2 \in D_{u_2}\}$ (le même que celui décrit dans la procédure de filtrage des degrés itérés). Le couple (u_1, v_1) est filtré s'il n'existe pas de couplage complet dans G_{u_1,v_1}. D'après le théorème de Hall [3], il existe un sous-ensemble $S \subseteq N(u_1)$ tel que $|S| > |N_{G_{u_1,v_1}}(S)|$.

On a d'une part, en utilisant (4) :

$$\sum_{u_2 \in S} \sum_{v_2 \in N(v_1)} y_{u_1u_2,v_1v_2} = \sum_{u_2 \in S} x_{u_1,v_1} = |S| x_{u_1,v_1}.$$

D'autre part, comme on sait que si $x_{u_2,v_2} = 0$, alors $y_{u_1u_2,v_1v_2} = 0$, en utilisant (7) :

$$\sum_{v_2 \in N(v_1)} \sum_{u_2 \in S} y_{u_1u_2,v_1v_2} = \sum_{v_2 \in N_{G_{u_1,v_1}}(S)} \sum_{u_2 \in S} y_{u_1u_2,v_1v_2} \leq \sum_{v_2 \in N_{G_{u_1,v_1}}(S)} x_{u_1,v_1} = |N_{G_{u_1,v_1}}(S)| x_{u_1,v_1}.$$

On a donc $|S| x_{u_1,v_1} \leq |N_{G_{u_1,v_1}}(S)| x_{u_1,v_1}$. Or $|S| > |N_{G_{u_1,v_1}}(S)|$, on en déduit que $x_{u_1,v_1} = 0$. Ainsi, toutes les nouvelles paires de sommets filtrés correspondent bien à une variable nulle pour la formulation **LBK+**.

Par le principe de récurrence, à la fin de la procédure, toutes les paires de sommets filtrés correspondent à une variable nulle pour la formulation **LBK+**. □

Nous allons maintenant montrer que **ILF** obtient un filtrage plus fort que celui de la formulation **LBKC**, et donc aussi **FCmult** et **FC**.

Théorème 2 *Soit P et T deux graphes sans sommets isolés. Alors il existe un point (x, y) telle que toutes les variables de la formulation **LBKC** sont strictement positives.*

Preuve : On pose $y_{u_1u_2,v_1v_2} = \frac{1}{2|E_T|}$ pour toutes les variables y et $x_{u,v} = \frac{d(v)}{2|E_T|}$ pour toutes les variables x. □

5 Résultats numériques

Nous comparons maintenant les performances de l'approche PLNE en utilisant le solveur Gurobi avec l'approche de programmation par contraintes en utilisant le solveur Glasgow [5].

Les instances et les résultats détaillés se trouvent à l'adresse suivante :

https ://github.com/etiennedeg

Le premier diagramme de performance FIG. 1 concerne des instances difficiles qui se trouvent dans la transition de phase [4] et montrent que la formulation **FC** est la plus efficace car elle permet de résoudre plus rapidement un plus grand nombre d'instances, malgré le filtrage plus fort de **LBK+**. Cela montre l'importance de la compacité de la formulation pour ce problème. Le second diagramme de performance FIG. 2 compare **FC** avec **ILF** pour des instances de plus grandes tailles (décrites dans [2]) et montre la supériorité de l'approche **ILF**.

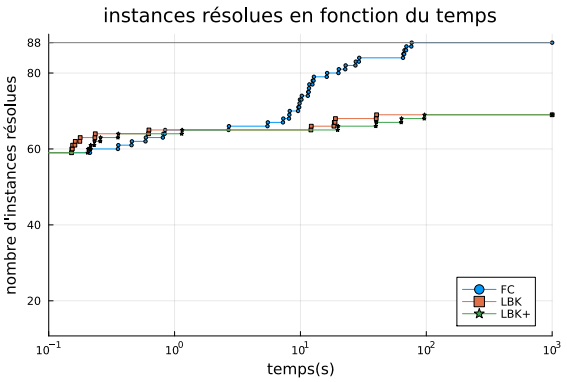

FIG. 1 – Nombre cumulé d'instances résolues en fonction du temps(s) sur les formulations PLNE

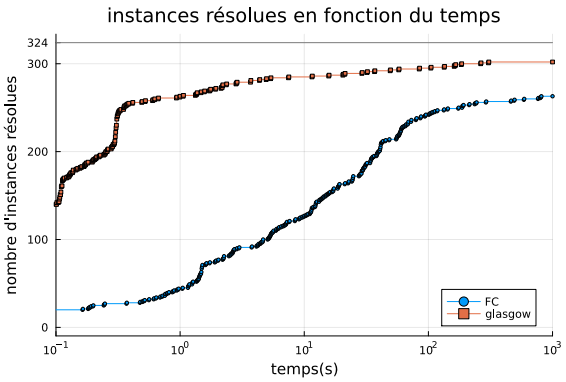

FIG. 2 – Nombre cumulé d'instances résolues en fonction du temps(s) pour le solveur Glasgow et la formulation **FC**

6 Conclusion

L'extension de la notion de filtrage au contexte de la programmation linéaire en nombres entiers nous donne un outil d'analyse et de comparaison intéressant. Dans le cas de l'isomorphisme de sous-graphes, la force de filtrage de la formulation **LBK+** ne lui permet pas de prendre l'avantage sur l'approche **ILF**, mais elle ouvre la perspective d'améliorer encore les résultats sur cette classe de problèmes particulièrement difficile. La procédure **ILF**, dont nous avons amélioré l'estimation de la complexité, se montre efficace en pratique. Du côté de la PLNE, nous avons renforcé une formulation de la littérature et on peut bien sûr espérer trouver des formulations encore plus fortes, en ayant à l'esprit que sur ce type de problèmes de décision, la compacité des formulations joue un rôle important. Notons enfin que les résultats de cet article peuvent s'étendre aux graphes orientés.

Références

[1] P. Le Bodic, *Formulations linéaires en nombres entiers pour des problèmes d'isomorphisme exact et inexact*, projet de fin d'étude, supervisé par Arnaud Knippel, INSA Rouen Normandie (2008)

[2] E. de Gastines, *Modélisation, approche polyédrale et approche arborescente pour des problèmes de comparaison de graphes*, thèse de doctorat, INSA Rouen Normandie et Normandie Université, (2023)

[3] P. Hall, *On Representatives of Subsets*, Journal of the London Mathematical Society, **s1-10**, (1935), 26–30

[4] C. McCreesh, P. Prosser, C. Solnon et J. Trimble, *When Subgraph Isomorphism is Really Hard, and Why This Matters for Graph Databases*, Journal of Artificial Intelligence Research, **61**, (2018), 723–759

[5] C. McCreesh, P. Prosser et J. Trimble, *The Glasgow Subgraph Solver : Using Constraint Programming to Tackle Hard Subgraph Isomorphism Problem Variants*, Graph Transformation, **9255**, (2020), 316–324

[6] C. Solnon, *AllDifferent-based filtering for subgraph isomorphism*, Artificial Intelligence, **174**, (2010), 850–864

[7] S. Zampelli, Y. Deville, C. Solnon, S. Sorlin et P. Dupont, *Filtering for Subgraph Isomorphism*, Principles and Practice of Constraint Programming – CP 2007, **4741**, (2007), 728–742