Rapport de projet: Géométrie Algorithmique

Paul-Alexis Rodriguez Charlopin

March 2025

1 Introduction

Dans le cadre du cours de géométrie algorithmique de troisième année à l'École Polytechnique, il nous a été demandé de concevoir un algorithme permettant de résoudre le problème du Minimum Dominating Set. Nous pouvions soit résoudre le problème de manière exacte, soit proposer un algorithme d'approximation rapide. Nous avons choisi de proposer une solution au problème, basée sur un algorithme d'approximation rapide, dont nous donnerons les détails dans ce rapport. Nous présenterons tout d'abord le pipeline mis en place, en détaillant les différentes étapes de celui-ci, puis une étude des performances de ce pipeline, enfin quelques résultats obtenus sur des graphes contenant de 20 à 311 sommets.

2 Le pipeline

Pour résoudre le problème, nous avons décidé d'utiliser l'heuristique suivante. On suppose que les graphes sont quasi-planaires et sparse. Le pipeline est le suivant : nous appliquons d'abord l'algorithme de Fruchterman-Reingold afin d'obtenir un embedding planaire (ou quasi-planaire) du graphe. Cet embedding nous permet ensuite d'obtenir facilement un séparateur géométrique du graphe, calculé à l'aide de la médiane des points par rapport à un axe (plus de détails seront donnés). Une fois le séparateur calculé, nous pouvons récursivement décomposer notre graphe en sous-graphes de petite taille afin de résoudre le problème sur ces instances. Enfin, nous fusionnons les solutions obtenues sur chacune des sous-parties.

2.1 Embedding planaire

Nous voulions d'abord utiliser l'algorithme de Tutte pour obtenir l'embedding planaire. Cependant, la nécessité de fournir une face externe nous a menés à choisir un autre algorithme, car nous ne pouvons fournir aucune garantie sur l'optimalité du choix de la face externe. Par ailleurs, nous voulions un algorithme persistant, qui ne dépende pas d'un paramètre structurel du graphe, comme c'est le cas des faces. Étant donné que les graphes nous sont fournis sous forme combinatoire, le calcul d'une face est coûteux et justifie pleinement

ce changement de méthode.

Nous avons donc opté pour l'algorithme de Fruchterman-Reingold. Cependant, celui-ci reste coûteux. Nous avons donc choisi d'ajouter l'approximation de Barnes-Hut, dont la précision est réglable dans notre code. L'avantage de cette approximation est qu'elle permet d'accélérer considérablement la vitesse de calcul de l'embedding. Au lieu de $O(n^2)$, on obtient plutôt $O(n \cdot log(n))$, la complexité dépend toutefois de la précision choisie. De plus, en contrôlant la précision de l'approximation, nous pouvons trouver un bon compromis entre la performance et la qualité de l'embedding. Cette dernière s'avère cruciale dans le calcul du séparateur puisqu'il ne repose que sur la géométrie induite du graphe.

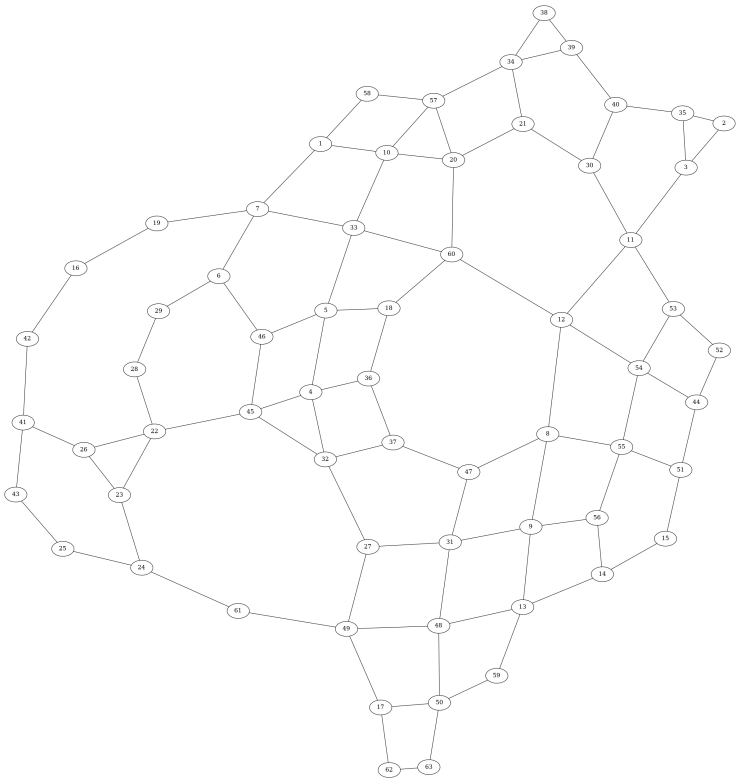

Figure 1: Embedding obtenu pour un graphe à 63 sommets

2.2 Calcul du séparateur

Une fois l'embedding obtenu, nous pouvons désormais calculer le séparateur. Deux options s'offraient à nous. La première était de calculer un séparateur spectral du graphe, en utilisant la matrice Laplacienne du graphe ainsi que le vecteur de Fiedler de celle-ci. La deuxième option profite pleinement de

l'embedding calculé à l'étape 1, en utilisant la médiane par rapport à un axe choisi. Nous avons décidé d'utiliser la deuxième méthode, puisque, selon notre heuristique (graphes quasi-planaires et sparse), la qualité des séparateurs obtenus est très bonne. Nous avons également constaté expérimentalement que les performances des séparateurs géométriques étaient meilleures que celles des séparateurs spectraux.

Pour l'implémentation de cette deuxième méthode, nous avons procédé comme suit. Nous choisissons un axe, puis nous calculons la médiane des points par rapport à cet axe. Les arêtes sécantes à cette médiane sont considérées comme appartenant au séparateur : nous en ajoutons donc les extrémités au séparateur. Les points situés du côté droit de la médiane sont ajoutés au sous-graphe droit, et ceux du côté gauche au sous-graphe gauche. Pour le choix de l'axe, nous testons 12 directions différentes : les axes x et y systématiquement, ainsi que 10 axes aléatoires. Le séparateur conservé est celui qui maximise un score prenant en compte à la fois l'équilibre entre les tailles des deux sous-graphes et la taille du séparateur. Plus précisément, ce score est défini comme $\frac{\min(|L|,|R|)}{(|L|+|R|)(|S|+1)}$, où L est le sous-graphe gauche, R le sous-graphe droit et S le séparateur.

2.3 Résolution de MDS

Une fois que nous avons les séparateurs, nous sommes prêts à résoudre le problème du Minimum Dominating Set (MDS). Nous commençons par diviser récursivement le graphe à l'aide des séparateurs précédemment présentés. Nous obtenons ainsi un arbre de décomposition du graphe original, dont les feuilles sont des sous-graphes de taille bornée, et les nœuds internes contiennent les séparateurs. Par souci de praticité, chaque nœud contient également les arêtes partagées entre le séparateur et ses sous-graphes : l'interface de gauche, composée des arêtes entre le séparateur et le sous-graphe gauche, et l'interface de droite, entre le séparateur et le sous-graphe droit.

Nous voulons désormais résoudre les feuilles et propager les solutions. Pour résoudre les feuilles, deux approches s'offraient à nous. La première consistait à faire un encodage SAT des contraintes de domination du graphe, puis à résoudre le problème par énumération : on calcule un dominating set à l'aide du SAT, on le conserve, on l'exclut de la formule, et on répète jusqu'à obtenir toutes les solutions possibles, dont on retient celle de cardinalité minimale. La deuxième approche consiste à ajouter une contrainte de cardinalité à la formule SAT, puis à effectuer une recherche dichotomique sur la taille de la solution. Cette méthode permet un gain significatif en performance, notamment en réduisant fortement le nombre d'appels au solveur. Nous avons donc naturellement retenu la seconde approche, la première s'avérant exponentiellement plus lente dans la pratique. Nous avons également choisi de travailler avec le solveur SMT Z3, l'un des solveurs les plus efficaces actuellement.

Une fois la résolution des feuilles faite, nous pouvons passer à la propagation.

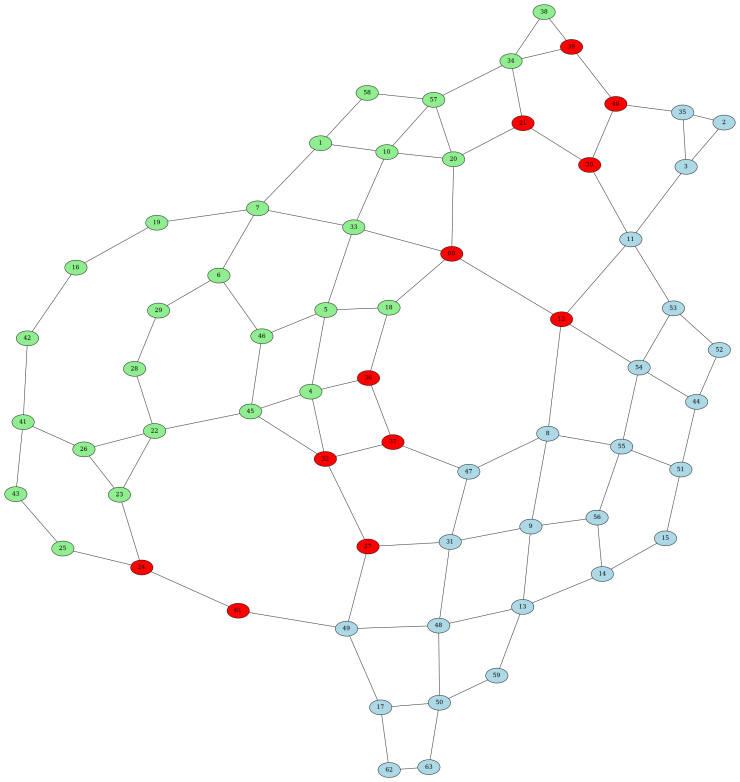

Figure 2: Séparateur obtenu avec la méthode 2. En rouge le séparateur, en vert le sous-graphe gauche et en bleu le sous-graphe droit

Pour cela, nous récupérons les solutions des sous-arbres gauche et droit, nous résolvons le problème pour le graphe induit par le séparateur, puis nous faisons l'union des trois solutions. Ensuite, nous optimisons localement la solution en l'élaguant au niveau des sommets du séparateur, en tentant de supprimer les sommets devenus inutiles sans compromettre la validité de la solution. Nous calculons ensuite le ratio entre la taille de la solution élaguée et celle du sous-graphe associé au nœud. Si ce ratio $r < 2$ et que la taille du sous-graphe est suffisamment petite, nous relançons le solveur SAT sur ce sous-graphe. Cette nouvelle résolution utilise une recherche dichotomique, bornée par la taille de la solution élaguée, ce qui permet de limiter encore davantage le nombre d'appels au solveur, puisque l'on sait que $OPT \leq Sol_{pruned}$.

Une fois que nous avons obtenu une approximation globale, nous appliquons quelques optimisations globales. Nous commençons par tenter d'améliorer le dominating set obtenu. Pour cela, nous utilisons un algorithme de point fixe basé sur des opérations de switch de sommets. Un switch consiste à prendre un sommet qui ne fait pas partie du dominating set, à l'ajouter, puis à retirer tous

ses voisins qui en faisaient partie, à condition que la validité de la solution soit préservée. Nous répétons ce processus jusqu'à ce qu'aucun switch ne modifie plus la solution. Cette étape permet de capturer des changements globalement optimaux mais localement sous-optimaux, qui ont échappé à la résolution récursive dans l'arbre. Enfin, nous appliquons un élagage global à la solution.

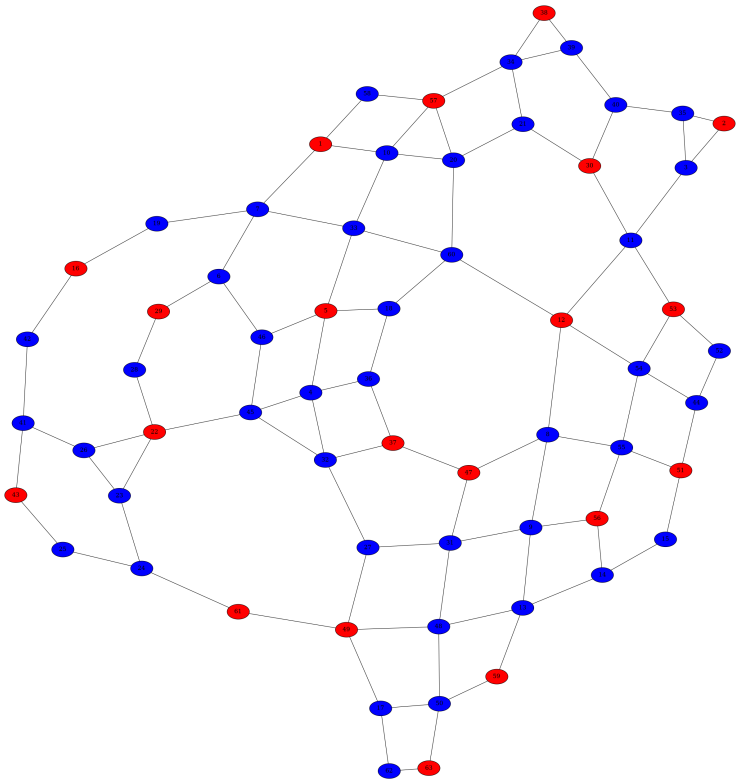

Figure 3: Approximation de MDS

2.4 Résultats

Nombre de sommets	**Taille de la solution**	**Taille optimale**
20	10	10
50	20	17
100	35	30
150	47	43
200	64	58
250	88	75
300	98	85

Table 1: Comparaison entre la solution approximative et la solution optimale selon la taille du graphe (ces résultats correspondent à une random seed fixée)