AAGA Projet 1: Dissertation sur les algorithmes présentés dans [Li, Thai, Wang, Yi, Wan, and Du. Wireless Communications and Mobile Computing, 2005].

BERREFAS Khelifa
SOURI Moad

October 2016

1 Introduction

Nos moyens de correspondances ont fortement évolué ces dernières années, notamment grâce au développement de réseaux de télécommunications modernes. Ces réseaux de télécommunications ont contribué au déploiement croissant d'antennes-relais. Il est donc important de pouvoir assurer le transport de l'information entre les différentes antennes, afin de satisfaire les différents acteurs du marché. Nous sommes face à des problématiques de coût et de performances: comment mettre le moins d'antenne relais, pour assurer la transmission de l'information de façon pérenne et efficace. Ce problème s'apparente clairement à celui d'un graphe connexe ou **le coût est représenté par le nombre de noeud utilisés** pour établir un graphe connexe, et **la performance est représentée au temps d'exécution de notre algorithme.** L'article dont nous allons faire l'étude propose une solution expérimentale aux problème de graphe décrit ci dessus.

Ainsi, nous étudierons dans un premier temps, les principes fondamentaux de cet article (ensemble dominant, connexité, etc) pour en déduire une modélisation d'un graphe connexe optimisé. Puis dans un second temps, nous nous intéresseront aux algorithmes d'optimisation décrit dans l'article (et dans un article référencé), pour les comparer ensuite)à notre algorithme de connexité Steiner.

2 Algorithmes

Nous allons commencé par définir les différents graphes/termes liés à la problématique d'un ensemble dominant connexe optimal, puis nous analyserons les algorithmes décrit dans l'article, pour construire une implémentation efficace. Finalement, nous comparerons l'algorithme expérimentale de cet article avec une implémentation de l'arbre de Steiner précédemment utilisé dans nos travaux.

2.1 Ensemble Indépendant Maximum

Un ensemble indépendant maximum d'un graphe est un sous graphe tel que entre chaque paire de point il n'existe pas d'arête les reliant. On remarque que cet ensemble est un ensemble dominant, mais par forcement le plus optimale. On peux construire un tel ensemble de manière gloutonne en ajoutant un quelconque premier point, puis un deuxième point en le choisissant parmi les non adjacents du premier et ainsi de suite jusqu'à ce qu'on puisse plus trouver de point.

Algorithm 1 MIS

Entrée points : liste de sommets du graphe
Sortie I: Ensemble indépendant Maximum

```
for(p:points) p.color=WHITE
s=points.get(0)
s.color=BLACK
I.add(s)
for(u:neighbor(s)) u.setActif(true)
while existWhite(points) do
  actifMax = actif with max white neighbors
  actifMax.color=BLACK
  I.add(actifMax)
  for(u:neighbor(actifMax))
    u.color=GREY
    for(v:neighbor(u))
      v.setActive(true)
end while
return I
```

Dans l'article Li et Al ils utilisent un MIS qui doit avoir la propriété

suivante: *chaque paire de point doit avoir une distance de deux saut c'est-à-dire exactement deux* arrêtes.

2.2 Ensemble Dominant

Un ensemble dominant est un sous ensemble de point tel que chaque point du graphe appartient sois à l'ensemble dominant sois il est voisin d'un des point de l'ensemble dominant. Pour obtenir un ensemble dominant on a utilisé un algorithme glouton qu'on améliore avec une boucle de local Searching. Le glouton c'est de garder les sommets qui on le maximum de voisins, est dans le local searching on améliore le résultat tant qu'on peut avec la règle de remplacer deux points de l'ensemble dominant par un point appartenant à l'ensemble de point de départ et n'appartenant pas à la solution courante.

Algorithm 2 DS-glouton

```
Entrée points : liste de sommets du graphe
Sortie DS: Ensemble Dominant

while points != ∅ do
  u=getMaxConnecte(points)
  DS.add(u)
  points.removeAll(neighbor(u))
  points.remove(u)
end while
return DS
```

Au résultat de l'algorithme 2 on applique un local searching qu'on améliore avec la règle: remplacer deux points de la solution par un seul point.

Algorithm 3 LocalSearching

```
Entrée points : liste de sommets du graphe
Sortie DS: Ensemble Dominant
solOld=DS-glouton(points)
solCurr = regle(glouton)
while solCurr.size() < solOld.size() do
  solOld=solCurr
  solCurr=regle(solOld)
end while
return solCurr
```

2.3 Ensemble Dominant Connexe

Un ensemble dominant connexe (CDS) est un ensemble dominant dans lequel nous pouvons "voyager" de n"importe quel point A vers n'importe quel point B (tous les noeuds de l'ensemble sont connectés). Dans la plupart des méthodes de construction de CDS, on utilise des couleurs pour marquer l'appartenance d'un noeud: noir désignant un noeud appartenant a un ensemble dominant, gris à un noeud dominé (directement voisin d'un point noir) et blanc tous les noeuds qui ne sont ni dominant ni dominé. Dans cet article, les chercheurs Guha et Khuller proposent deux algorithmes de construire un squelette de CDS.

1er Algorithme

Un premier algorithme consisterait à construire le CDS noeud par noeud, on prend arbitrairement un premier noeud, qui sera notre premier point dominant, que l'on colorie en noir, puis on colorie tous les points dominés gris, et on réitère jusqu'à ne plus avoir de points blancs.

2nd Algorithme

Un second algorithme consisterait dans un premier temps à identifier tous les points dominants, puis dans un second à les relier entre eux avec des noeuds intermédiaire pour former un ensemble de point dominants connexe.

Il existe plusieurs façon de construire un CDS, mais quel est l'algorithme le plus rentable ?

Ici, pour évaluer l'efficacité d'un CDS, on utilise le rapport de performance (PR) définis de la sorte : taille du CDS construit / taille du MCDS.

2.4 Principe / Implémentions

S-MIS

L'algorithme S-MIS décrit dans l'article est une nouvelle approche gloutonne du problème. La première étape de cet algorithme consiste à construire un (MIS) ensemble indépendant maximum,**chaque noeud du MIS a au plus 5 noeud voisins indépendant**.Dans le graphe résultant du MIS, chaque point a une distance de deux intervalles avec son points voisin.

Puis, dans la seconde étape, nous connectons tous les noeuds de notre MIS pour finalement former un CDS (par définition un MIS est un ensemble dominant mais pas connexe). L'algorithme permettant de rendre le MIS connexe repose sur une coloration des points; tous les points sont faisant partis du MIS sont initialement noir, tous les autres sont gris, et deviennent

bleu s'ils ont n des voisins noir, voisins qui doivent être issus de différentes composantes connexes (n allant de 5 à 2).

Remarque : Dans l'algorithme ci dessus, nous ne préoccupons que du nombre de voisins noir (dans différents composantes connexe) et pas du nombre de voisins bleu, ce qui a pour effet de ne pas perturber l'algorithme en cours. En effet, si à chaque itération, le nombre de voisins d'un point gris venait à être changer cela compliquerai l'algorithme et donnerai un résultat presque identique.

Nous avons implémenté deux algorithmes le premier celui décrit dans l'article qu'on nommera S-MIS, le deuxième c'est résoudre Steiner avec un ensemble dominant que l'on nommera DS-Steiner.

Ci-dessous les captures d'écrans des résultats des deux algorithmes sur le graphe *input.points*.

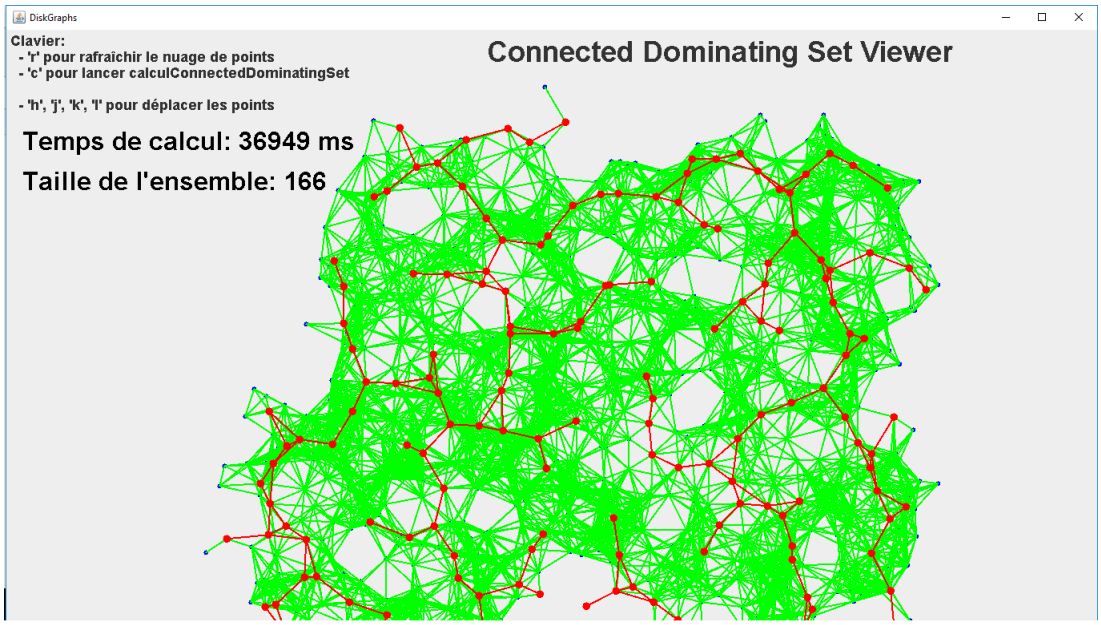

Figure 1: DS-Steiner

Dans cet échantillons, nous remarquons, pour un nombre de 1000 points en entrée, nous arrivons à couvrir l'ensemble des points avec un CDS de taille 166 et un temps de calcul de 36 secondes

L'implémentation de l'algorithme S-MIS ci-dessous, nous montre clairement une différence d'efficacité, avec un temps d'exécution beaucoup plus faible (1 secondes), et un gain du nombre de noeud utilisé.

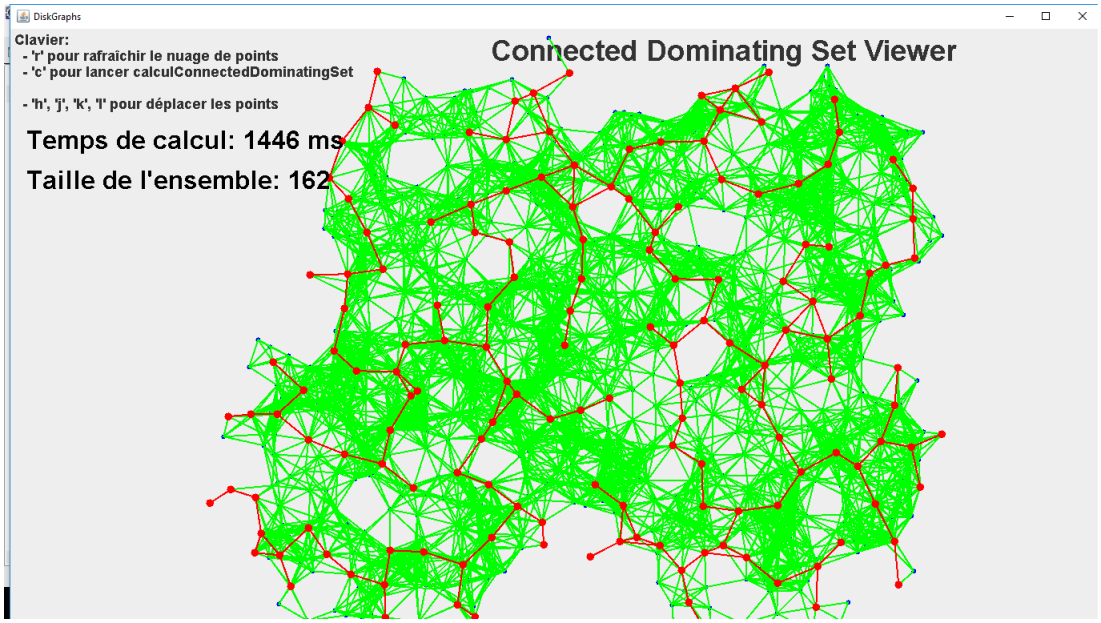

Figure 2: S-MIS

3 Résultats

Nous allons maintenant élargir nos échantillons, pour avoir une base de tests plus pertinente.

L'étude expérimentale pour comparer l'algorithme de l'article et notre algorithme se base aussi sur un jeu de test; nous allons utilisé une base de test de 100 instances de graphe générés aléatoirement chaque instance contient 500 points.*(cf Figure1)*

En ordonnée nous avons le nombre de sommet de notre Ensemble Dominant Connexe (CDS), et en abscisse le numéro de l'instance de test.

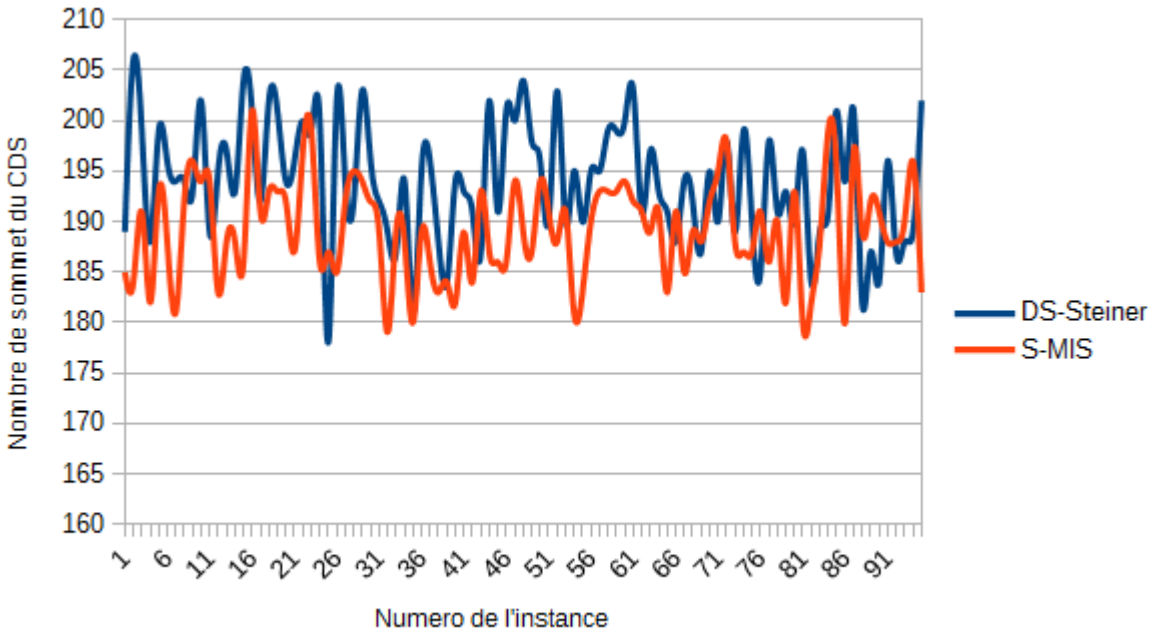

Figure 3: Nombre de noeuds avec S-MIS et DS-Steiner

La courbe rouge (algorithme de l'article) est presque toujours en dessous de la courbe bleu (algorithme Steiner de base), ce qui traduit un CDS plus petit et par conséquent plus performant pour le S-MIS.

D'autre part, la variance ou amplitude de la courbe bleu est plus grande que la courbe rouge, cela explicite le caractère aléatoire de l'algorithme de Steiner; qui peut avoir un CDS de plus de 200 puis un CDS de moins de 180 à l'instance suivante.

A la suite de ces arguments, on peut clairement avancer que l'algorithme S-MIS de l'article est beaucoup plus optimale que l'algorithme Steiner de base.

Remarque : La répartition géographique des noeuds et aussi intéressante à examiner, l'algorithme Steiner présente des noeuds "entremêlés" avec des cycles très appuyés, alors que du coté de l'algorithme S-MIS, cette formation cyclique est beaucoup moins présente, et la répartitions des noeuds sur le graphe est plus "homogène", dans le sens ou chaque noeuds quadrille des voisins différents (en générale).?

4 Conclusion

Dans l'implémentation de l'algorithme S-MIS, nous avons rencontrés des difficultés dans le développement de la première étape, à savoir construire un MIS valide. L'algorithme du MIS et présent dans un autre article, ce qui a prolongé le temps de développement; nous avions pensé que nous auront pu sauter cette étape en remplaçant la première étape par un glouton basique, sur lequel nous aurions pu appliquer la coloration de point, mais **cela ne respectait pas le Lemme 2** indispensable à la mise en pratique du S-MIS.

Par contre, l'algorithme S-MIS est plus simple, et moins long à mettre en place que Steiner. Ce qui est paradoxal puisque celui ci (S-MIS) a un temps d'exécution beaucoup plus rapide que Steiner; cela nous a prouvé qu'a l'issue d'une propriété rigoureusement démontrée, les chercheurs peuvent aboutir à des algorithmes performants sans développements prématurés.

Cependant, avec les avancées technologies dont nous faisons face (au niveau hardware), il est fort probable que la portée de certaines antennes relais soit de plus en plus grande et que leur vitesse de transmission soit plus rapide, ce qui agrandira le champs d'actions de notre graphe rendant ainsi les problématiques de hardware plus importantes que le software. Mais cela n'arrivera pas, puisque même à très grande échelle, nous chercherons toujours à réduire nos coûts en installant le moins de hardware possible; et cela passe par un software intelligent.