Un nouvel algorithme d'approximation polynomial pour le problème de l'échafaudage.

Annie Chateau[1], Tom Davot[1], Rodolphe Giroudeau[1], Mathias Weller[2]

[1] Université de Montpellier, LIRMM, Montpellier, France
{chateau,davot,giroudeau}@lirmm.fr
[2] CNRS, LIGM, Marne-la-Vallée
mathias.weller@u-pem.fr

Mots-clés : *graphe, échafaudage génomique, approximation.*

1 Introduction

En biologie , le séquençage est une technologie permettant d'extraire l'information génétique de l'ADN. Le séquençage et l'assemblage produisent des séquences appelées *contigs*. Les informations présentes au sein de ces contigs sont utiles pour certaines analyses, comme la présence ou non de certains gènes, mais ne permettent pas d'avoir une vision d'ensemble du génome. Le problème *d'échafaudage* est un problème d'optimisation permettant de reconstituer la séquence d'ADN originale à partir des contigs. Pour résoudre ce problème, nous travaillons sur un *graphe d'échafaudage* (G^*, M^*, ω), défini comme suit : G^* est un graphe simple doté d'un couplage parfait M^* et d'une fonction de poids $\omega : E(G^*) \setminus M^* \to \mathbb{N}$. Les arêtes de couplage représentent les contigs et la fonction de poids ω représente le taux de confiance que deux contigs se suivent dans la séquence génomique. Un chemin ou cycle $(u_1, \ldots, u_k)$ est dit *alterné* si $u_{2i+1}u_{2i+2}$ est une arête de couplage pour tout $i \leq \frac{k}{2}$. Les arêtes aux extrémités d'un chemin alterné appartiennent au couplage. Le problème d'échafaudage consiste à trouver une collection de σ_p chemins alternés et de σ_c cycles alternés (réprésentant les chromosomes linéaires et circulaires) telle que la collection maximise la somme des poids. Les entiers σ_p et σ_c sont donnés en entrée du problème.

2 Algorithme Glouton

Les résultats présentés ici sont une continuation des résultats obtenus dans [1]. Le problème de l'échafaudage est NP-difficile, même pour les graphes peu denses. Pour pouvoir proposer des solutions à ce problème, on peut se tourner vers des algorithmes approchés comme l'algorithme glouton, présenté dans l'Algorithme 1. L'idée générale de cet algorithme est de parcourir les arêtes n'appartenant pas au couplage par ordre décroissant de poids et des les ajouter au fur

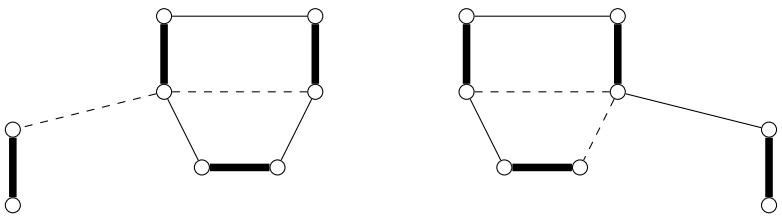

FIG. 1 – Exemple de graphe d'échafaudage et de solution au problème d'échafaudage. Les arêtes en gras appartiennent au couplage et toutes les arêtes n'appartenant pas au couplage sont de poids un. Les arêtes en traits pleins appartiennent à une solution ayant pour entrée $\sigma_p = 2$ et $\sigma_c = 1$.

et à mesure dans la solution lorsque c'est possible. La difficulté majeure est de développer une fonction de *Faisabilité*. Cette fonction indique s'il est possible de construire une solution dans un graphe (G^*, M^*, ω) à partir d'une solution partielle S. Il n'existe pas de fonction de faisabilité en temps polynomial dans le cas général. Une première version de l'algorithme glouton a été étudiée dans les graphes complets, aboutissant au résultat suivant :

Théorème 1 ([1]) *Dans les graphes complets, l'algorithme glouton est une 3-approximation.*

Les instances réelles sont cependant assez éloignées des graphes complets. Pour pouvoir utiliser l'algorithme glouton sur des instances réelles, il est nécessaire d'ajouter des arêtes de poids nul à celles-ci. Toutefois, une solution renvoyée par l'algorithme glouton peut contenir des arêtes n'existant pas dans le graphe d'origine et dans ce cas, la solution ne peut pas être applicable à l'instance réelles. Nous avons étudié des fonctions de faisabilité sur d'autres classes de graphes en essayant de répondre à l'hypothèse suivante : est-ce l'algorithme glouton renvoie une meilleure solution si on utilise une fonction de faisabilité plus proche de l'instance d'origine ?

Algorithm 1: Algorithme Glouton

```
   // Phase d'initialisation.
1  S ← M*;
2  E ← E(G*) \ M*;
3  trier E par ordre décroissant de poids;
4  Si ¬Faisabilité(G*, M*, S, σ_p, σ_c) alors retourner Faux;
   // boucle principale
5  Pour tous e ∈ E faire
6  |  Si Faisabilité(G*, M*, S ∪ {e}, σ_p, σ_c) alors
7  |  |  S ← S ∪ {e};
8  retourner S;
```

3 Graphes de cluster connectés

Un *graphe de cluster connecté* G est un graphe pour lequel il existe une ensemble d'arêtes B tel que chaque arête de B est un isthme et chaque composante connexe de $G \setminus B$ est une clique. Nous avons développé une fonction de faisabilité sur les graphes de cluster connectés reposant sur un algorithme dynamique, amenant au résultat suivant :

Théorème 2 *Dans les graphes de cluster connectés, l'algorithme glouton est une 5-approximation.*

Bien que la borne théorique soit moins bonne pour les graphes de cluster connectés que pour les graphes complets, les tests effectués sur des instances réelles tendent à montrer que la version sur les graphes de cluster connectés renvoit une solution au moins aussi bonne en terme de qualité.

4 Conclusion et perspectives

Les tests effectués tendent à confirmer notre hypothèse de départ. Toutefois l'amélioration apportée par la nouvelle version de l'algorithme glouton n'est pas significative. On peut supposer que les graphes de cluster connectés sont trop denses pour apporter une grande amélioration. Il pourrait être intéressant de continuer ce travail en étudiant des classes de graphes contenant des instances moins denses ou d'utiliser une formulation SAT pour créer une fonction de faisabilité dans le cas général.

Références

[1] A. Chateau and R. Giroudeau, *A complexity and approximation framework for the maximization scaffolding problem*, Theoretical Computer Science, (2015), 92–106.