Université de Mons
Faculté des sciences
Département d'Informatique

Title

Rapport de stage d'initiation à la recherche

Professeur :
Hadrien Mélot
Superviseur :
Sébastien Bonte

Auteur :
William Karpinski

Année académique 2024-2025

Table des matières

1 Introduction

Ce rapport s'inscrit dans le cadre d'un stage d'initiation à la recherche dans le service d'algorithmie dirigé par M.Hadrien Mélot. Dans ce stage, supervisé par Sébastien Bonte, j'ai été ammené à travailler sur plusieurs invariants de graphes, et plus spécifiquement sur le tree width de graphes simple non-orienté.

2 Notions de base

Dans ce rapport, nous allons définir G, un graphe simple non-orienté comme étant composé de V, un ensemble de sommets, et E, un ensemble d'arêtes. On définit également l'ordre de G, comme étant le nombre de sommets dans V et sa taille, comme le nombre d'arêtes dans E.

En théorie des graphes, une décomposition en arbre d'un graphe G est un arbre, tel que chaque noeud de celui-ci contient un sous-ensemble de V et possède les propriétés suivantes. Soient T, une décomposition en arbre de G, $X_1, ..., X_t$, les sous-ensembles de V dans les noeuds de T, nous avons que :

- Tous les sommets de G doivent être dans au moins un noeud de T
- Si

$$et$$

3 Tree Width

Explication du tree width Tree Decomp Tree Width = minimum de toutes les width des tree decomp dun graph

3.1 Méthode Naïve

Enumération de toutes les possibilités de tree decomp Extremememnt lent O(n!) Ajout d'un lower bound => /4 le temps mais tjrs pas suffisant

3.2 Notions Complémentaires

Prochains algos viennent d'un papier qui introduit quelques notions supplementaire Probleme d'ordo linéaire TW() Q, R, P

3.3 Meilleur Algorithme

TW(S) = min maxTW(S-v),|Q(S-v,v)| Basé la desus

3.4 Algorithme Récursif

TWR(L,S) = min max Q(L U i(<,v),v), TWR([],V) = TW[V] = tw(G) => Calcule de TWR sur [],V pour trouver tw(G)

3.5 Algorithme Récursif Amélioré

dit si tw(G) au plus = k se base sur les composantes connexes du graphes et si leur tw <= k

4 Autres Invariants

Variance des degres,Proximity/Remoteness,Girth

5 Comparaison Python/Rust

Rust mieux mais assez bizarrement pas pour proxi et remote

6 Conclusion