Projet de MOGPL : Algorithme Bellman Ford et Glouton Fas

Thomas Roig N°28708645 – Andy Torres N° 21304450
08/12/2023

1 Table des matières

2 Introduction

Au cours de ce projet de l'UE MOGPL, on s'intéresse à l'introduction d'un nouvel algorithme, Glouton Fas et son utilisation potentielle dans le cadre de l'algorithme de Bellman Ford afin d'optimiser la recherche de plus court chemin.

3 Organisation du projet

Le projet se découpe en trois fichiers différents :

- projet.py, qui regroupe les fonctions principales du projet comme les algorithmes étudiés, les fonctions nécessaires à la génération d'instances aléatoires et des tests de comparaison

- toolbox.py, qui contient des fonctions utilitaires comme l'initialisation de Bellman Ford, le calcul de chemin ou encore la suppression de sommet dans un graphe pour Glouton Fas.

- testprojet.py, qui sert de fichier de test où l'on exécute notre code.

Les graphes seront modélisés sous la forme suivante :

$$(liste\ de\ Sommets, listes\ d'arcs)$$

La liste de sommets est constituée des divers sommets qui composent le graphe.

La liste d'arc comprend les arcs au format suivant :

$$(départ, arrivée, coût)$$

Le tuple est donc composé du sommet de départ, d'arrivée et du coût lié à l'arc.

4 Implémentation des Algorithmes

4.1 Bellman Ford

Bellman Ford est ici implémenté en récursif. A chaque itération, on calcule le nouveau tableau des distances et l'on vérifie si ce dernier est identique à celui qui est présent précédemment. La fonction renvoie alors le tuple suivant :

$$(nombre\ Itérations, chemin\ plus\ court, distance)$$

Dans le cas où la distance n'est pas la même, on passe à l'itération suivante. Si jamais on atteint un nombre d'itération qui correspond au nombre de sommets on quitte la fonction car cela indique que l'algorithme n'a pas convergé et que l'on en déduit la présence d'un circuit absorbant. La fonction renvoie alors $False$.

4.2 Glouton Fas

Pour implémenter glouton Fas, on applique l'algorithme donné dans le sujet en n'oubliant pas de mettre à jour la liste des sources puis celle des puits à chaque suppression de sommets dans le graphe.

5 Tests et performances

5.1 Génération de Graphes

Pour la génération de graphe, on va utiliser les trois fonctions suivantes :

- $generateGraphe(nbSommets,\ probabilite)$, qui va générer un graphe non pondéré composé du nombre de sommets souhaité avec la probabilité d'apparition des arcs spécifiés. La fonction renvoie un tuple composée de la liste des sommets et de la liste des arcs.

- $genOriginGraphe(nbSommets, probabilite)$, génère le graphe de test qui ne doit pas comporter de circuit absorbant et dont on peut atteindre depuis la source au moins la moitié des sommets du graphe. On utilise pour cela la fonction pour changer la pondération des arcs, puis on applique l'algorithme de Bellman Ford pour détecter les circuits absorbants.

- $genGraphes(nbGraphes, origin\_S, origin\_A)$ qui va construire une liste de $nbGraphes$ à partir de la liste des sommets et des arcs du graphe d'origine. On vérifie aussi qu'après modification des coûts qu'un circuit absorbant n'est pas apparu avec Bellman Ford.

Ces fonctions sont utilisées pour générer des instances aléatoires qui serviront à tester l'efficacité de Bellman Ford. La randomisation de la pondération des arcs s'effectue par tirage aléatoire sur l'intervalle $[-3,10]$.

5.2 Test d'union

Pour comparer les performances de ce que pourrait nous donner Glouton Fas avec un ordre aléatoire, on implémente la fonction suivante :

$$testUnion(grapheOrigin, l\_Graphes, nbfront, Affiche)$$

Cette fonction va procéder de la manière suivante :

1. Calcul de l'union des arborescences de $nbfront$ graphes contenus dans $l\_Graphes$.
2. Formatage de l'union des arborescences, qui supprime les arcs en double.
3. Application de Glouton Fas sur la liste de sommets du graphe d'origine et l'union des arborescences formatée.
4. Application de Bellman Ford sur l'ordre retourné par Glouton Fas et la liste des arcs du sommet d'origine.
5. Application de Bellman Ford sur un ordre aléatoire et la liste des arcs du sommet d'origine.
6. Comparaison des deux valeurs

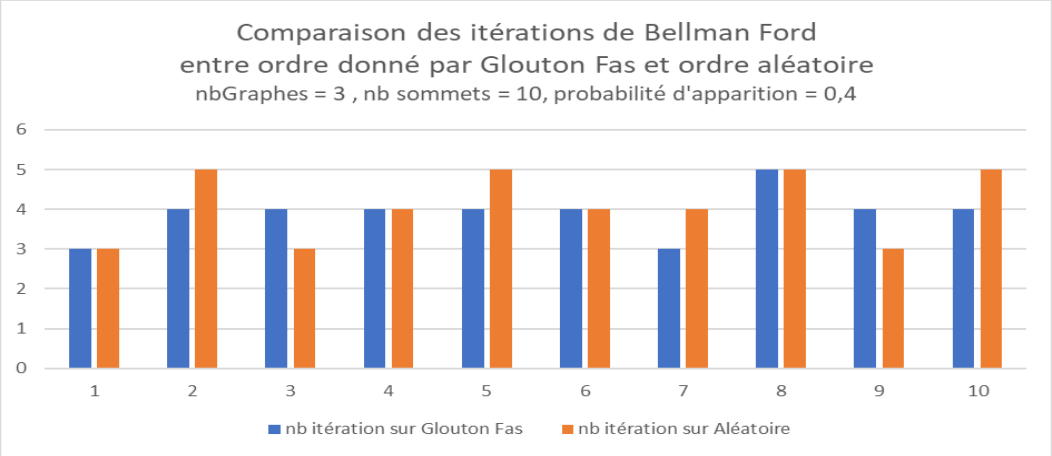
Cela permet de tester si Glouton Fas permet d'améliorer Bellman Ford.

5.3 Performances via Glouton Fas

5.3.1 Différence entre Glouton Fas et l'aléatoire

Pour déterminer si Glouton Fas permet d'améliorer les performances de Bellman Ford on va générer des instances de graphes et tester après apprentissage si le nombre d'itérations avec pré-traitement est plus faible qu'avec un ordre aléatoire.

On obtient les résultats suivants après voir généré trois graphes G_1, G_2, G_3 et comparé les résultats de l'ordre renvoyé par Glouton Fas $<_{tot}$ et un ordre aléatoire sur un autre graphe H :



On observe que bien que dans certaines instances, on peut constater une amélioration, dans la plupart des cas, Glouton Fas ne permet pas une réduction certaine du nombre d'itérations sur trois graphes appris.

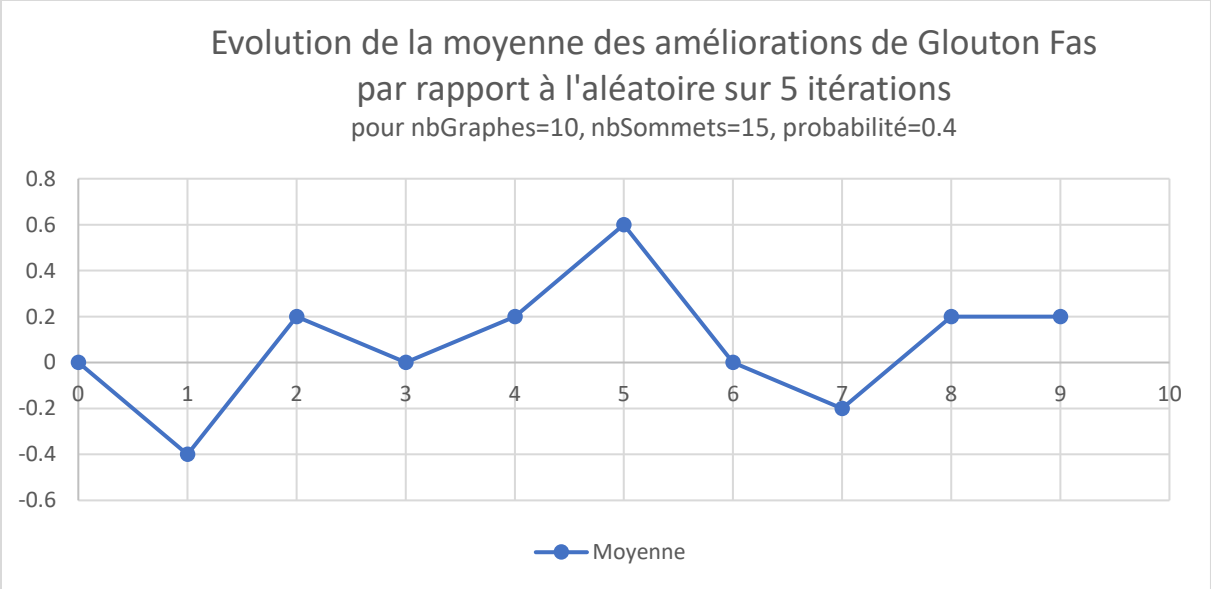
5.3.2 Performances selon le nombre de graphes appris

Après avoir observé que, dans le cadre de trois graphes appris, Glouton Fas n'améliore pas forcément Bellman Ford, on peut se demander si le nombre de graphes appris influence les performances de Bellman Ford.

Pour effectuer cela, on va utiliser la fonction suivante :

$$testUnionGraduel(grapheOrigin, l\_Graphes, nbfront, Affiche, AfficheDetail)$$

Cette fonction va procéder à l'appel du test d'union avec un nombre de graphes appris croissant afin d'afficher l'amélioration apportée par Glouton Fas en fonction du nombre de graphes appris. On obtient les performances suivantes :



On remarque que l'évolution de la moyenne des améliorations ne permet pas de dégager une influence certaine entre le nombre de graphes appris et le nombre d'itérations de Bellman Ford.

6 Conclusion et traitement de la question 11

Dans le cadre d'un graphe par niveau comportant 4 sommets par niveau et 2500 niveaux, il n'y a pas d'intérêt à utiliser Glouton Fas car les sommets d'un niveau j seraient les sources des sommets du niveau $j + 1$. Ainsi, Glouton Fas ne ferait que lister les sommets dans l'ordre croissant des sommets sans pour autant améliorer les performances de Bellman Ford.

On a vu ici que l'implémentation de l'algorithme de Glouton Fas ne permettait pas d'obtenir une amélioration certaine du nombre d'itération de Bellman Ford peu importe le nombre de graphes appris.