République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

جامعة الشاذلي بن جديد
UNIVERSITE CHADLI BENDJEDID

MEMOIRE

Présenté par

Djelti Douaa

Pour l'obtention de diplôme de

MASTER

Filière : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

Le problème de recherche k plus courts chemins base sur l'optimisation de l'algorithme SB

Soutenu le : 10 / 09 / 2023 Devant le Jury composé de :

Qualité	Nom et Prénom	Grade	Université
Président	**Mr. Benmachich Abdelmajid**	MCA	Chadli Bendjedid El-Tarf
Encadreur	**Mme. Bouguerne Imen**	**MCB**	**Chadli Bendjedid El-Tarf**
Examinateur	**Mr. Djedai Ala**	MCB	Chadli Bendjedid El-Tarf

Année Universitaire : 2022/2023

Remerciements

Je tiens avant tout à remercier Dieu le tout puissant de m'avoir donné

la force Et la volonté pour achever ce modeste travail A mon encadreur, Mdm. Bouguerne imen J'adresse ma plus profonde reconnaissance pour son bon encadrement et pour les conseils fructueux qu'il n'a cessé de ma prodiguer.

Je deviens chaque bribe de ma connaissance à mes enseignants au département informatique qui ont si bien mené leur noble quête d'enseigner les bases d'Informatique « Intelligence Artificielle ». Je les

remercie non seulement pour le savoir qu'ils nous ont transmis, mais aussi pour la fierté et l'ambition que leurs personnes nous aspirent.

Que messieurs les membres du jury trouvent ici l'expression de notre

reconnaissance pour avoir accepté d'évaluer mon travail. Et toutes les

personnes qui ont contribué de près ou de loin au bon déroulement de ce travail.

Dédicace

Du profond de mon cœur, je dédie ce travail à tous ceux qui me sont chers,

A ma très chère mère

Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération

pour les sacrifices que vous avez consenti pour mon instruction et mon bien être.

Je vous remercie pour tout le soutien et L'amour que vous me portez depuis mon enfance et

j'espère que votre bénédiction m'accompagne toujours. Que ce modeste travail soit

l'exaucement de vos vœux tant formulés, le fruit de votre innombrable sacrifice. Puisse dieu,

le très haut, vous accorder santé, bonheur et longue vie.

A mon très cher père

Tu as toujours été à mes cotes pour me soutenir et m'encourager, me motiver dans mes

études. Que ce travail traduit ma gratitude et mon affection.

A mes très chères sœurs

INES et takouaa merci énormément pour vous soutien plus que précieux, merci beaucoup

d'être toujours là pour moi. Ma vie ne serait pas aussi magique et formidable sans vos

présence et vos cœurs mes âmes, puisse dieu vous donne santé, bonheur, courage et surtout

réussite.

Ma 2 éme famille

Je tiens à remercier chaleureusement, tous mes proches et tous ceux ceux qui, de près et de

loin, m'ont apporté leur sollicitudes pour accomplir ce travail.

Merci à tous ceux qui me sont chers, mes amis, mes frères qui ont étais toujours ici pour moi

Yasser et Ammar, Amira je t'aime vraiment ma 2 deuxième famille.

Je t'aime de tout mon cœur.

Table des matières

Liste des figures

Liste des tableaux

Introduction Générale

Dans le domaine dynamique et en constante évolution de la théorie des graphes et de l'optimisation algorithmique, le défi de déterminer les k chemins les plus courts entre deux sommets désignés dans un graphe apparaît comme un problème fondamental et complexe d'une importance primordiale. Cette question, caractérisée par ses complexités mathématiques, a de profondes implications dans une multitude de domaines, englobant la planification des transports, le routage des réseaux de communication et le paysage complexe de la biologie computationnelle. La poursuite des k chemins les plus courts présente non seulement une voie d'exploration théorique, mais engendre également un impact significatif sur la prise de décision pratique et l'allocation des ressources au sein de systèmes complexes.

Au cœur de cette énigme se trouve la tâche de déterrer une collection organisée de k chemins, chacun démontrant un poids cumulé inférieur à tous les chemins alternatifs laissés non sélectionnés. Contrairement au problème du chemin le plus court classique, qui recherche un itinéraire optimal singulier, le problème des k chemins les plus courts implique un examen plus complet de la connectivité, offrant plusieurs itinéraires qui incarnent des compromis variables. Cette multiplicité de chemins viables fait écho à la complexité des scénarios du monde réel où diverses considérations éclairent la prise de décision.

La valeur intrinsèque de cette entreprise est profondément liée à ses applications dans le monde réel. L'ère de l'urbanisation et de la numérisation souligne la pertinence d'une planification efficace des transports, où des itinéraires optimisés sont indispensables pour une mobilité rationalisée. De plus, les réseaux de communication complexes nécessitent un routage rapide et efficace des paquets de données pour une connectivité transparente. Notamment, dans le domaine de la biologie computationnelle, déchiffrer des voies biologiques complexes exige une compréhension nuancée de multiples voies réalisables au sein de réseaux moléculaires complexes.

Dans le discours académique, la quête pour relever le défi des k plus courts chemins a été ponctuée par des contributions algorithmiques séminales. L'algorithme pionnier de Yen, créé en 1971, a introduit une approche systématique pour générer k chemins les plus courts simples, gérant efficacement les problèmes de redondance. Alors que le travail de Yen a jeté les bases, les années qui ont suivi ont vu une prolifération d'améliorations et d'avancées visant à améliorer l'efficacité et l'évolutivité des calculs. Au milieu de cette évolution, l'algorithme Sidetrack Based (SB), conçu par Kurz et Murzel en 2016, apparaît comme un concurrent notable, tirant parti des stratégies basées sur la déviation pour exceller dans l'efficacité pratique, mais non sans considérations de consommation de mémoire en raison des arbres de chemin les plus courts stockés.

Dans le cadre du problème des k chemins les plus courts se trouvent des objectifs variés, invitant les chercheurs à naviguer dans des complexités à multiples facettes. Celles-ci incluent la construction d'arbres succincts des chemins les plus courts, l'optimisation de la mémoire et l'accélération des processus de résolution de problèmes. Trouver un équilibre entre les ressources informatiques et la réalisation de ces objectifs est essentiel pour améliorer la viabilité pratique des solutions.

Alors que le paysage de la prise de décision complexe et de la connectivité à multiples facettes continue d'évoluer, le problème des k chemins les plus courts se présente comme un phare, guidant le discours sur les solutions optimales et enrichissant notre compréhension de la tapisserie complexe qu'est la théorie des graphes. Les références à des travaux précurseurs, tels que l'algorithme de Yen, les progrès d'Eppstein et l'algorithme Sidetrack Based de Kurz et Murzel, soulignent l'impact durable des contributions scientifiques sur la résolution des complexités de ce défi persistant [5] [3] [6].

1. Contexte du projet et Motivations

Le problème des k chemins les plus courts est devenu un domaine d'étude critique en raison de sa pertinence dans de nombreux scénarios du monde réel où la diversité des chemins joue un rôle crucial. Les applications couvrant les réseaux de transport, les systèmes de télécommunication et la logistique nécessitent souvent non seulement le chemin le plus court, mais également des itinéraires alternatifs pour assurer la robustesse, l'équilibrage de charge ou l'atténuation des risques. Comme ces applications ont gagné en complexité et en importance, la demande de solutions efficaces pour calculer k chemins les plus courts s'est intensifiée.

Au fil du temps, divers algorithmes ont été développés pour résoudre le problème des k plus courts chemins, chacun apportant son propre ensemble d'avantages et de limitations. Ces algorithmes sont conçus pour trouver un équilibre entre l'efficacité de calcul et la qualité de la solution, en tenant compte de facteurs tels que les performances d'exécution, l'utilisation de la mémoire et l'optimalité du chemin. Alors que le problème des k chemins les plus courts continue de trouver des applications dans un nombre croissant de domaines, l'urgence d'affiner les algorithmes existants et de concevoir de nouvelles approches s'est amplifiée.

2. problématique

Au cœur de la présente recherche se trouve l'objectif d'améliorer l'algorithme Sidetrack Based (SB), un concurrent de premier plan pour le calcul des k chemins les plus courts. L'objectif principal de cet effort d'optimisation est centré sur l'amélioration de la consommation de mémoire de l'algorithme. Alors que l'algorithme SB a montré des performances empiriques remarquables

en termes de génération de k chemins les plus courts, son inconvénient notable réside dans ses besoins importants en mémoire. Cette limitation a limité sa viabilité, en particulier dans les scénarios caractérisés par des environnements à ressources limitées, où la disponibilité de la mémoire est une préoccupation essentielle.

Relever ce défi implique d'étudier des stratégies innovantes pour rationaliser l'utilisation de la mémoire dans le cadre de l'algorithme SB sans compromettre indûment son efficacité de calcul ou la qualité des chemins de sortie. En concevant des techniques pour réduire l'empreinte mémoire de l'algorithme, il devient possible d'étendre son applicabilité à des environnements où les ressources mémoire sont rares, permettant son adoption dans des scénarios précédemment considérés comme inaccessibles en raison de limitations de mémoire.

En résumé, cette étude vise à faire progresser l'utilité de l'algorithme Sidetrack Based en optimisant sa consommation de mémoire, élargissant ainsi son aspect pratique et son impact dans les scénarios nécessitant k calculs de chemin le plus court, en particulier dans les environnements à ressources limitées. En trouvant un équilibre entre l'efficacité de la mémoire et les performances algorithmiques, la recherche contribue à l'arsenal d'outils disponibles pour traiter la diversité des chemins dans les applications du monde réel.

3. Objectifs

Cette thèse vise à améliorer l'efficacité de l'algorithme SB en réduisant sa consommation de mémoire tout en préservant sa vitesse pratique. Les principaux objectifs sont les suivants :

- Analysez les modèles d'utilisation de la mémoire de l'algorithme SB.
- Explorer et mettre en œuvre des techniques de réduction de mémoire dans l'algorithme.
- Optimisez la construction des arbres des chemins les plus courts pour minimiser les calculs redondants.
- Évaluez les performances de l'algorithme optimisé grâce à des tests approfondis.
- Comparez l'algorithme optimisé avec les approches existantes en termes de consommation de mémoire, de temps d'exécution et de qualité de la solution.

4. Contenu du mémoire

Cette thèse est centrée sur l'exploration et l'analyse de l'algorithme Sidetrack Based (SB) en tant que solution majeure au problème des k chemins simples les plus courts. L'objectif principal de ce travail est de se plonger dans les subtilités et le fonctionnement de l'algorithme SB, en comprenant

ses principes, ses forces, ses limites et ses contributions au domaine de la théorie des graphes et de l'optimisation [5]. Le rôle de l'algorithme Sidetrack Based dans la résolution efficace du défi de l'énumération des k chemins simples les plus courts dans les graphes orientés, en tenant compte d'aspects tels que les stratégies basées sur la déviation et l'optimisation de la mémoire, sera étudié en profondeur.

Parallèlement à l'examen approfondi de l'algorithme SB, cette thèse mettra également en évidence les contributions originales apportées par l'auteur pour améliorer encore la compréhension du domaine et la mise en œuvre pratique des k algorithmes des plus courts chemins simples. Ces contributions comprendront de nouveaux raffinements, optimisations ou extensions innovantes de l'algorithme SB, visant à remédier aux limitations existantes, à augmenter l'efficacité de calcul ou à étendre son applicabilité à des contextes plus larges.

La portée de la thèse sera délimitée en trois chapitres pivots. Le premier chapitre fournira une introduction approfondie au problème des k chemins simples les plus courts, en soulignant son importance dans divers scénarios du monde réel [1][9]. Cette section posera les bases de l'exploration ultérieure de l'algorithme SB. Le deuxième chapitre se plongera dans une analyse complète de l'algorithme SB, élucidant ses mécanismes de travail, ses stratégies de calcul et ses implications [6]. Une attention substantielle sera consacrée à la discussion de l'efficacité pratique, des défis et des réalisations de l'algorithme.

Le troisième chapitre sera consacré à détailler mes contributions uniques dans le domaine [9]. Cela comprend la présentation d'améliorations, d'optimisations ou d'extensions originales de l'algorithme SB, conçues pour relever des défis spécifiques ou améliorer ses capacités. Les contributions peuvent impliquer des méthodologies innovantes, des techniques de calcul ou des considérations pratiques, toutes visant à faire progresser l'état de l'art dans la résolution du problème des k chemins simples les plus courts.

En se concentrant sur l'exploration de l'algorithme Sidetrack Based (SB) et mes contributions, cette thèse fournira un examen complet des subtilités et des applications pratiques de l'algorithme tout en faisant progresser les connaissances du domaine grâce à de nouvelles améliorations. Le but ultime est de contribuer au développement continu d'algorithmes efficaces pour résoudre des problèmes d'optimisation complexes dans le domaine de la théorie des graphes.

1. Introduction : Présentation de la théorie des graphes

Aperçu de la théorie des graphes La théorie des graphes, une branche des mathématiques qui étudie les relations entre les objets, a évolué d'une curiosité théorique à un outil puissant et indispensable dans diverses disciplines. Les graphes sont des structures mathématiques constituées de nœuds (également appelés sommets) reliés par des arêtes. Ces nœuds et arêtes fournissent un cadre flexible pour représenter un large éventail de phénomènes et d'interactions, faisant de la théorie des graphes un concept fondamental dans des domaines allant de l'informatique et de l'ingénierie aux sciences sociales et à la biologie.

À la base, la théorie des graphes offre un langage riche pour décrire et analyser les relations. Les graphes peuvent être représentés par $G = (V, E)$, où V est l'ensemble des nœuds et E est l'ensemble des arêtes. Ils peuvent être dirigés E contient des paires ordonnées de nœuds ou non dirigés E contient des paires non ordonnées, et ils peuvent également être pondérés, avec des valeurs numériques attribuées aux arêtes qui représentent les distances, les coûts ou d'autres valeurs pertinentes. métrique. Cette polyvalence permet aux graphiques de capturer un large éventail de relations, des connexions sociales aux réseaux physiques.

Les origines de la théorie des graphes remontent au 18e siècle avec le problème des E « Sept ponts de Königsberg » posé par Euler [10], qui a marqué le début du domaine. La solution d'Euler a introduit le concept de chemins et de circuits dans les graphes. Depuis lors, la théorie des graphes a évolué pour englober une multitude de concepts, notamment la connectivité, les arbres, la planéité, la coloration, etc.

La pertinence de la théorie des graphes s'étend à un large éventail de domaines. En informatique, les graphes sont utilisés pour modéliser des structures de données, des réseaux et des algorithmes. L'analyse des réseaux sociaux s'appuie sur des graphiques pour comprendre les schémas de connexions entre les individus ou les entités, éclairant le flux d'informations et la propagation de l'influence [11]. Dans le transport et la logistique, les graphes modélisent les itinéraires, les flux et les chaînes d'approvisionnement, optimisant l'allocation des ressources et l'efficacité [12]. En biologie, les graphes représentent les structures moléculaires, les interactions génétiques et les réseaux de protéines, aidant à la compréhension des processus biologiques complexes [13].

Les problèmes de plus court chemin constituent l'une des applications les plus connues de la théorie des graphes. Étant donné un graphe, le problème du plus court chemin vise à trouver le chemin le plus efficace entre deux nœuds, souvent dans le but de minimiser la distance, le coût ou le temps. L'algorithme de Dijkstra et l'algorithme de Bellman-Ford sont des méthodes classiques pour résoudre ces problèmes [1][2].

Une autre application importante est l'analyse des flux de réseau, où les graphiques modélisent le mouvement des ressources à travers les réseaux. L'algorithme de Ford-Fulkerson et ses extensions résolvent efficacement les problèmes de flux de réseau, en trouvant des moyens optimaux de transporter des marchandises, des données ou des informations [14].

Ces dernières années, l'analyse de graphes et de réseaux à grande échelle a pris de l'importance avec l'avènement des mégadonnées et des systèmes complexes. Des techniques telles que la détection de communauté, l'analyse de centralité et l'intégration de graphes permettent d'extraire des informations significatives à partir de structures de graphes massives [15][16].

2. Présentation des algorithmes des chemins les plus courts

En théorie des graphes, la poursuite du chemin le plus court entre deux sommets est un défi intemporel qui trouve une application étendue dans une multitude de scénarios du monde réel. De la navigation sur les réseaux de transport à l'optimisation du routage des données dans les réseaux informatiques, le problème du plus court chemin est une pierre angulaire indispensable pour résoudre un large éventail de défis liés à l'optimisation et à l'efficacité. Au fil des ans, un riche assortiment d'algorithmes a été conçu, chacun offrant des stratégies uniques pour découvrir l'itinéraire optimal à travers un graphique tout en tenant compte des poids de bord variables.

Le problème de l'algorithme du plus court chemin concerne la tâche de calcul consistant à trouver l'itinéraire ou le chemin le plus efficace entre deux nœuds désignés dans un graphe. Étant donné un graphique

G comprenant des nœuds et des arêtes, chaque arête ayant typiquement un poids ou un coût associé, et étant donné un nœud source s et un nœud cible t, l'objectif est de déterminer le chemin de s à t qui a le plus petit poids ou coût cumulé.

Mathématiquement, supposons que G=(V,E) représente un graphe avec l'ensemble de nœuds V et l'ensemble d'arêtes E. Chaque arête e∈E a un poids associé w(e) qui représente le coût ou la distance de traversée de cette arête. Il vise à trouver un chemin p d'un nœud source s à un nœud cible t qui minimise la somme des poids des arêtes le long du chemin : e∈pw(e).

Il existe divers algorithmes conçus pour résoudre efficacement le problème du chemin le plus court en fonction des caractéristiques du graphe, par exemple si les arêtes ont des poids non négatifs, si le graphe est dirigé ou non et si le graphe contient des cycles. Les algorithmes bien connus pour résoudre ce problème incluent l'algorithme de Dijkstra [1], l'algorithme de Bellman-Ford [2] et l'algorithme de Floyd-Warshall [4], chacun avec ses propres forces et limites en fonction des exigences spécifiques du problème et du graphe propriétés.

2.1. Algorithme de Dijkstra

L'une des premières percées dans ce domaine est l'algorithme de Dijkstra [1]. Cet algorithme fournit une approche systématique pour découvrir le chemin le plus court dans un graphe avec des poids d'arête non négatifs. Dijkstra explore méthodiquement le graphe à partir d'un sommet source choisi, étendant continuellement sa portée au sommet non visité le plus proche tout en mettant à jour méticuleusement les estimations de distance. Au fur et à mesure de sa progression, l'algorithme de Dijkstra[1] garantit l'identification du chemin le plus court vers chaque sommet depuis la source.

Algorithme :

Initialisation :

- Soit $dist(v)$ la distance la plus courte actuellement connue depuis s vers le sommet v.
- Initialiser $dist(s) = 0$ et $dist(v) = \infty$ pour tous les autres sommets $v \neq s$.
- Définir $S = \{\}$.

Boucle Principale :

- Tant que S n'inclut pas tous les sommets :
  - Sélectionner le sommet u dans V qui n'est pas encore dans S avec la plus petite valeur de $dist(u)$.
  - Ajouter u à S.
  - Pour chaque sommet v adjacent à u et n'étant pas dans S :
    - Calculer la distance provisoire $tempDist$ depuis s vers v en passant par u : $tempDist = dist(u) + poids(u, v)$
    - Si $tempDist$ est plus petit que la distance actuelle $dist(v)$, mettre à jour $dist(v) = tempDist$.

Résultat :

- Après que l'algorithme ait terminé, $dist(v)$ pour chaque sommet $v$représente la distance la plus courte depuis s vers v.

2.2. Algorithme de Bellman-Ford

Paragraphe... Dans les scénarios où des poids de bord négatifs sont impliqués, l'algorithme de Bellman-Ford vient au premier plan [2]. Cet algorithme adopte une approche itérative, affinant successivement les estimations de distance pour chaque sommet par relaxation. L'algorithme Bellman-Ford est particulièrement utile en raison de sa capacité à identifier les cycles de poids négatifs, détectant ainsi les cas où aucun chemin le plus court ne peut être défini.

2.3. Algorithme de recherche A*

Une approche plus nuancée émerge sous la forme de l'algorithme de recherche A* [3]. En combinant des aspects des approches de Dijkstra [1] et heuristique, A* incorpore une fonction heuristique qui guide l'exploration vers le sommet de destination. Cet algorithme allie l'efficacité de Dijkstra à la perspicacité d'une recherche informée, ce qui le rend particulièrement puissant lorsqu'une estimation informée de la distance restante jusqu'à la destination est disponible.

2.4. Algorithme Floyd-Warshall

Pour les scénarios impliquant des graphes denses, l'algorithme Floyd-Warshall fait ses preuves [4]. Cet algorithme fonctionne en considérant tous les sommets intermédiaires possibles entre n'importe quelle paire de sommets et calcule les chemins les plus courts en conséquence. Bien que sa complexité temporelle soit supérieure à celle d'autres algorithmes, Floyd-Warshall est utile lorsque l'exigence implique de trouver tous les chemins les plus courts par paires.

3. Le problème des k chemins les plus courts

Dans le monde complexe de la théorie des graphes et de l'optimisation, les dimensions de la complexité augmentent lorsque nous nous aventurons au-delà du domaine des problèmes de chemin le plus court traditionnels. Le problème des k chemins les plus courts apparaît comme une extension captivante, nous invitant à explorer les subtilités de la découverte non seulement d'un chemin optimal, mais d'un spectre entier de k chemins entre deux sommets dans un graphe. Le problème traditionnel du chemin le plus court, bien qu'essentiel dans l'analyse de réseau, échoue parfois lorsqu'il est confronté à des scénarios de prise de décision nuancés. Le problème des k chemins les plus courts comble cette lacune en offrant une perspective plus riche sur les options de connectivité et de traversée. Ce problème encapsule l'aspiration à déterrer k chemins, chacun possédant le poids cumulé le plus bas, tout en respectant la contrainte selon laquelle tout chemin non sélectionné doit être plus long que le chemin sélectionné le plus court. Les résultats d'une telle

exploration ont de profondes implications dans divers secteurs, de la planification des transports urbains au routage des réseaux de communication et au-delà.

Le fondement du problème des k chemins les plus courts repose sur des graphes orientés pondérés, où les nœuds se présentent comme des entités de localisation, et les arêtes symbolisent des connexions dotées de poids qui signifient des distances, des coûts ou d'autres mesures pertinentes. Étant donné un sommet source s et un sommet cible t, l'objectif est de découvrir un ensemble de k chemins qui minimisent collectivement le poids cumulé, offrant ainsi un spectre de routes viables entre la source et la destination.

Pour résoudre le problème des k plus courts chemins, le parcours a été ponctué de contributions algorithmiques séminales. L'algorithme de Yen, un travail fondateur de 1971, a été le pionnier d'une méthodologie pour calculer k chemins les plus courts en générant de manière itérative des chemins et en gérant habilement la redondance [5]. Alors que l'algorithme de Yen a jeté les bases, les années suivantes ont vu une pléthore d'améliorations et d'avancées visant à améliorer l'efficacité et l'évolutivité. Dans le paysage contemporain, l'algorithme Sidetrack Based (SB), conçu par Kurz et Murzel en 2016, apparaît comme un concurrent notable. Fonctionnant sur le socle des stratégies basées sur la déviation, l'algorithme SB brille par son efficacité pratique [6]. Pourtant, il est confronté au problème de la consommation de mémoire en raison du nombre important d'arborescences de chemins les plus courts stockées.

L'efficacité des algorithmes abordant le problème des k plus courts chemins simples est au cœur de leur applicabilité. L'algorithme de Yen présente une complexité temporelle de O(kn(m + mlog n)), où N représente le nombre de nœuds et M le nombre d'arêtes dans le graphe [5]. Les travaux d'Eppstein présentent une avancée notable à cet égard avec un algorithme fonctionnant en complexité temporelle O(m + n log n + k log k) [7]. L'algorithme SB présente un contraste intrigant. Bien qu'il excelle en vitesse pratique, sa consommation de mémoire devient une préoccupation notable [6]. L'algorithme de Johnson [8], quant à lui, est connu pour son efficacité dans le calcul du plus court chemin dans les réseaux clairsemés.

Dans le domaine du problème des k chemins les plus courts, des objectifs distincts émergent, invitant les chercheurs à naviguer dans les subtilités. Ces objectifs incluent la construction d'arborescences de chemins les plus courts, l'optimisation de la mémoire et l'accélération des processus de résolution de problèmes. Équilibrer les ressources de calcul tout en atteignant ces objectifs est crucial pour améliorer l'aspect pratique des solutions.

4. Historique

L'exploration de l'optimisation des itinéraires et des chemins a constitué une entreprise fondamentale dans les domaines de l'informatique, des mathématiques appliquées et de l'ingénierie. Cette quête des itinéraires optimaux, qu'ils soient les plus courts, les plus rapides ou les plus efficaces, au sein de graphiques pondérés, a été le moteur de l'évolution continue des algorithmes de plus court chemin. Notre voyage à travers les annales du temps commence dans les années 1950, lorsque Edsger W. Dijkstra a introduit un algorithme révolutionnaire destiné à devenir la pierre angulaire de la recherche sur l'optimisation des chemins [1]. L'algorithme de Dijkstra, axé sur l'identification du chemin le plus court entre deux sommets dans un graphe pondéré non négatif, a posé la première pierre des développements ultérieurs en matière d'optimisation de chemin. Ses applications initiales étaient centrées sur la planification et le routage des réseaux, avec de profondes implications pour les systèmes de transport et de communication [18].

Les années 1970 ont marqué le début d'une nouvelle ère dans la recherche sur les chemins les plus courts avec l'introduction de l'algorithme K des chemins les plus courts (KSP) par Jin Y. Yen [5]. L'innovation de Yen a été transformatrice ; il a permis aux algorithmes d'identifier non seulement le chemin le plus court, mais également les K chemins les plus courts entre deux sommets désignés. Cette contribution cruciale a ouvert la porte à l'exploration de diverses solutions alternatives, un concept qui s'avérera déterminant pour façonner l'avenir des algorithmes de recherche de chemin. Les années 1980 ont vu une nouvelle expansion du domaine alors que les chercheurs se sont aventurés dans le domaine des K chemins les plus courts. Cette exploration enrichit le domaine en ouvrant de nouveaux horizons en matière de routage et d'optimisation des réseaux. Notamment, au cours de cette période, Sandi Klavžar et Drago Košmrlj ont contribué à un algorithme en temps polynomial pour identifier K chemins les plus courts, soulignant l'importance de l'efficacité informatique dans la résolution de problèmes complexes d'optimisation de chemins [19].

Les années 1990 ont marqué une accélération significative de la recherche sur les K chemins les plus courts. C'est à cette époque que les chercheurs ont commencé à intégrer des heuristiques et des optimisations dans leurs algorithmes, dans le but d'améliorer leur efficacité. Ces avancées ont diversifié les solutions disponibles, permettant des approches personnalisées dans des domaines spécifiques tels que la planification des réseaux et la logistique. Une contribution notable à cette époque a été l'introduction de l'algorithme CCSP (Capacity-Constrained Shortest Path) par Thomas L. Magnanti et Robert T. Wong en 1994 [20]. CCSP a permis la découverte du chemin le

plus court au sein d'un réseau limité par la capacité, offrant ainsi de nouvelles perspectives dans la conception du réseau.

Néanmoins, à mesure que la complexité des réseaux modernes s'est accrue et que la nature dynamique des environnements du monde réel est devenue de plus en plus évidente, il est devenu évident qu'un changement de paradigme était nécessaire. Les années 2000 ont marqué l'émergence des algorithmes basés sur les détours (SBA), représentant un bond en avant innovant dans le domaine des algorithmes de chemin le plus court. Les SBA ont introduit une toute nouvelle approche de l'optimisation des itinéraires, centrée autour du concept de détours. Ces algorithmes ont permis d'explorer des chemins alternatifs tout en conservant une trajectoire principale, permettant ainsi une adaptation dynamique aux changements en temps réel des conditions environnementales. Cette approche révolutionnaire a élargi les horizons de l'optimisation des itinéraires, offrant un niveau de flexibilité et de robustesse jusqu'alors inégalé. Les SBA ont trouvé des applications dans un large éventail de domaines, allant de la gestion de flotte à la logistique, en passant par les systèmes de transport intelligents et la planification des réseaux de communication, démontrant leur adaptabilité et leur polyvalence pour relever des défis complexes du monde réel.

Dans le domaine des SBA, des contributions spécifiques ont laissé des traces indélébiles. Notamment, les travaux de Kurz et Murzel [21] ont joué un rôle central dans l'amélioration de l'énumération des chemins de déviation, permettant une exploration plus efficace des alternatives tout en garantissant que la qualité de la trajectoire principale reste intacte.

Je pourrais dire que le parcours historique des algorithmes des K chemins les plus courts témoigne de la recherche incessante de solutions toujours plus sophistiquées pour répondre aux demandes posées par les réseaux modernes et les environnements dynamiques. Les SBA représentent la dernière étape de cette trajectoire évolutive, incarnant l'adaptabilité, la résilience et la capacité à relever des défis multiformes dans le domaine de l'optimisation des itinéraires.

5. Définition de problème

Considérons un graphe orienté $G = (V, E)$, où V représente l'ensemble de nœuds et E l'ensemble d'arêtes. Soit s et t les nœuds source et cible dans |V|. Dans ce contexte, nous supposons une fonction de poids d'arête implicite $c: E \rightarrow R_0^+$. Le nombre de nœuds est noté n et le nombre d'arêtes m. Un chemin reliant les nœuds v et w dans G, appelé chemin v-w, est représenté par une séquence d'arêtes p = (e1 , e2, ..., er), où ei=(vi,wi) et v = v_1, $w_i = v_{i+1}$ pour $1 \leq i < r$. Pour simplifier, nous nous concentrons sur les combinaisons de G, s et t où un chemin s-v et un chemin

v-t existent dans G pour chaque vV Un nœud u est considéré comme faisant partie du chemin p où up, si u = w ou u=vi pour certains i. Si $vi = vj = w$ pour $1 \leq i, j \leq r, p$. p est considéré comme un chemin simple. Le préfixe (e1, . . . , ei) est noté comme un chemin v = wi et étiqueté comme pi. La longueur c(p) du chemin p représente la somme des poids des bords le long de ses bords. A chemin p est considéré comme le chemin v-w le plus court si tous les autres chemins v-w sont plus longs ou égaux à p. La notation G-p signifie le sous-graphe induit $G[\{v \in V \mid v \notin p\}]$.

Le problème en question, connu sous le nom de k problème du chemin simple le plus court (kSSP), relève du domaine des problèmes d'énumération. Soit un graphe orienté $G = (V, E)$, un nœud source $s \in V$, un nœud cible $t \in V$, des poids d'arête c et un entier k ∈ N Le but est de calculer un ensemble P contenant K chemins simples de s à t dans G. Ces chemins doit satisfaire la condition c(p) ≤ c(p0) pour tout p ∈ P et $p0 \notin P$. Si l'exigence que les chemins calculés soient simples est omise, le problème devient le k problème du plus court chemin (kSP).

Un arbre de chemin le plus court (arbre SP) T de G est un sous-arbre avec un ensemble de nœuds V où chaque v ∈ V a exactement une arête sortante située sur un chemin v-t le plus court, ou aucune arête sortante si aucun chemin v-t dans G existe. Nous nous référons à une arête $e \notin T$ comme une dérive, et son coût de dérive δT(e) est défini comme (c(e) + d(w)) − d(v), où d(u) représente la longueur de l'unique u-t chemin dans T. L'ensemble de déviation DT (v) pour un nœud v ∈ V comprend toutes les déviations avec des queues sur l'unique chemin v-t dans T. L'indice de déviation i* et le nœud de déviation dev(p0) sont définis dans le contexte de chemins p = (e1 , . . . , ek) et p0 = (f1 , . . . , fk), soit deux s-t chemins. La découverte hiérarchique des chemins implique des chemins parents et des arêtes de déviation, et les chemins sont disséqués en chemins de préfixe et de suffixe.

Une représentation des chemins est introduite, où chaque déviation e dans une séquence est associée à un arbre de plus court chemin différent T_e. La reconstruction d'un chemin à partir d'une séquence de déviation implique de suivre les arbres SP et les déviations associées. Cette représentation généralise la représentation d'Eppstein, qui considère les déviations par rapport au même arbre de chemin le plus court. La distance d'un nœud v à t dans un arbre de chemin le plus court T_e associé à une déviation e est notée de(v).[6]

6. Conclusion

Dans ce chapitre d'introduction, nous avons plongé dans le monde fascinant de la théorie des graphes, une discipline aux applications diverses et profondes. Nous avons découvert que les

graphiques ne sont pas simplement des structures abstraites, mais aussi des modèles puissants permettant de représenter et de résoudre une variété de problèmes du monde réel.

Nous avons mis en évidence l'importance cruciale des problèmes de chemins les plus courts, qui sont au cœur de nombreuses applications, de la planification des transports à la biologie computationnelle et au routage des réseaux. Ces problèmes sont souvent complexes, mais leur résolution a un impact important sur notre quotidien.

Le problème des k chemins les plus courts a été introduit comme une extension intrigante des problèmes traditionnels des chemins les plus courts, ouvrant de nouvelles possibilités dans la recherche de multiples solutions optimales. Nous avons exploré les domaines où cette problématique est pertinente, notamment dans la recherche de chemins divers dans des réseaux complexes et multimodaux.

En retraçant l'histoire des problèmes de chemin le plus court, nous avons observé l'évolution des méthodes et des algorithmes, depuis les premiers travaux pionniers jusqu'aux approches modernes basées sur des stratégies innovantes.

Enfin, nous avons défini les bases du problème simple des k chemins les plus courts, jetant ainsi les bases des chapitres à venir. Ce problème sera notre principal sujet d'étude, avec un accent particulier sur l'algorithme Sidetrack Based (SB) et les contributions originales de l'auteur à son développement.

En conclusion, ce chapitre d’introduction a fourni les bases nécessaires pour approfondir le monde passionnant de la théorie et de l’optimisation des graphes. Il nous a familiarisé avec les défis et les opportunités qui nous attendent et a ouvert la voie à une exploration en détail de l’algorithme SB et de ses applications dans la suite de ce travail.

Chapitre 2

1. Introduction à l'algorithme basé sur Sidetrack (SB)

Dans le chapitre précédent, nous nous sommes lancés dans un voyage captivant à travers le domaine aux multiples facettes de la théorie des graphes, une discipline qui sous-tend de nombreuses applications et défis du monde réel. Forts de cette exploration enrichissante, nous naviguons maintenant sur le terrain complexe des solutions algorithmiques qui ont non seulement propulsé les limites de la théorie des graphes, mais également façonné le cours des méthodologies d'optimisation. Au premier plan de ces innovations se trouve l'algorithme Sidetrack Based (SB), une construction informatique remarquable qui a laissé une marque indélébile dans le paysage complexe de l'optimisation basée sur les graphes. Alors que nous nous aventurons plus profondément dans ce chapitre, nous nous lançons dans une expédition globale au cœur de l'algorithme SB, dévoilant sa création, ses motivations, ses principes fondamentaux et ses implications profondes dans le domaine de la théorie et de l'optimisation des graphes.

2. Motivations et justification

Le paysage technologique en constante évolution, associé à la complexité croissante des problèmes du monde réel, a stimulé la demande d'algorithmes sophistiqués capables de relever efficacement les défis d'optimisation complexes. Entrez l'algorithme Sidetrack Based (SB), une innovation notable conçue pour résoudre le problème des k chemins simples les plus courts. Conçu par Kurz et Murzel en 2016, l'algorithme est né d'un besoin impérieux de développer une méthodologie qui non seulement améliore l'efficacité de l'énumération des chemins, mais embrasse également le paysage nuancé des chemins alternatifs avec précision et efficacité [6].

3. Stratégies basées sur la déviation dans l'algorithme SB

Au cœur de l'ingéniosité de l'algorithme Sidetrack Based (SB) se trouve le concept révolutionnaire de stratégies basées sur la déviation, une rupture profonde avec les méthodologies algorithmiques conventionnelles. Alors que les approches traditionnelles mettent principalement l'accent sur le raffinement de l'arbre des chemins les plus courts, une stratégie indéniablement fructueuse pour découvrir les chemins optimaux, elles peuvent ignorer par inadvertance les chemins alternatifs qui détiennent une efficacité supérieure dans des contextes distincts.

L'algorithme SB, grâce à son paradigme innovant basé sur la déviation, transcende les limites de la tradition en identifiant et en examinant astucieusement les chemins qui s'écartent de l'arbre de chemin le plus court conventionnel. Cette approche unique non seulement accélère le processus d'énumération des chemins, mais confère également une nouvelle dimension au domaine de l'espace de solutions, enveloppant un spectre polyvalent d'options de traversée avec le potentiel de démêler les efficacités cachées, de contourner les goulots d'étranglement et d'éclairer les avenues inexplorées [6].

3.1 Cadre conceptuel : s'écarter de la tradition

Les algorithmes de chemin le plus court conventionnels ont longtemps prospéré sur le principe de l'optimisation des chemins exclusivement dans l'arborescence des chemins les plus courts. Cette approche limite intrinsèquement le spectre des chemins explorés à ceux étroitement alignés avec la structure de l'arbre, écartant par inadvertance les chemins qui s'écartent de cette route conventionnelle. Les stratégies basées sur la déviation de l'algorithme SB, cependant, remettent en question ce paradigme en s'aventurant dans un territoire inexploré, embrassant des chemins qui transcendent les limites de l'arbre. En explorant délibérément des chemins qui empruntent des détours calculés, l'algorithme SB s'efforce de découvrir des chemins qui pourraient contourner les goulots d'étranglement, exploiter des itinéraires spécialisés ou même présenter des alternatives supérieures. Ce départ de la tradition est un départ fondamental qui permet à l'algorithme SB d'offrir une gamme dynamique d'options de chemin.

3.2. Exploration et sélection : Naviguer vers de nouveaux horizons

Le cœur des stratégies basées sur la déviation réside dans l'exploration et la sélection délibérées par l'algorithme des chemins qui s'aventurent au-delà du périmètre de l'arbre des chemins les plus courts. Ces chemins, appelés « déviations », s'écartent de la structure arborescente traditionnelle, offrant la possibilité de débloquer de nouvelles idées et des alternatives efficaces. Les déviations sont évaluées non seulement pour leur divergence par rapport à l'arbre, mais aussi pour leurs avantages potentiels. Chaque détour introduit un coût de déviation, une mesure du gain ou de la perte potentiel qui accompagne le détour. En évaluant les compromis entre les coûts de déviation et les avantages potentiels, l'algorithme SB intègre de manière transparente les chemins basés sur la déviation dans son espace de solutions. Cette approche holistique élargit les horizons de l'exploration de solutions, permettant à l'algorithme de présenter des chemins qui pourraient être négligés par les méthodes conventionnelles de chemin le plus court.

3.3. Exploration efficace : un équilibre atteint

Alors que les stratégies basées sur la déviation offrent un changement de paradigme dans l'exploration de chemin, l'algorithme SB équilibre judicieusement l'efficacité avec la profondeur d'exploration. L'algorithme ne se lance pas dans une quête exhaustive de tous les écarts possibles, reconnaissant que tous les écarts ne donnent pas des informations ou des gains d'efficacité précieux. Au lieu de cela, l'algorithme sélectionne avec soin les dérives en fonction de leur impact potentiel sur l'objectif d'optimisation. Cette approche pragmatique minimise les frais de calcul inutiles tout en préservant l'essence de l'exploration basée sur les déviations. Le résultat est un équilibre délicat trouvé entre l'efficacité de calcul et l'exploration approfondie des solutions, garantissant que l'algorithme offre des solutions pratiques sans succomber aux pièges d'une complexité excessive.

3.5. Implications et applications : au-delà du chemin le plus court

Les implications des stratégies basées sur la déviation dans l'algorithme SB se répercutent dans divers domaines et applications. Dans les réseaux de transport, par exemple, ces stratégies peuvent découvrir des itinéraires alternatifs qui atténuent la congestion aux heures de pointe. De même, dans les réseaux de communication, les chemins basés sur la déviation peuvent réduire la congestion du réseau ou offrir un routage efficace pour les données sensibles au facteur temps. Dans le contexte plus large de l'optimisation, ces stratégies mettent en lumière le pouvoir de penser au-delà du conventionnel pour découvrir des solutions non conventionnelles mais pragmatiques. L'accent mis par l'algorithme SB sur l'exploration basée sur les déviations témoigne de l'évolution constante du paysage des techniques d'optimisation, nous rappelant que parfois le meilleur chemin n'est pas le plus court mais celui qui navigue efficacement sur le terrain complexe des scénarios du monde réel.

Titre	Description	Méthode	Performance
Cadre conceptuel: s'écarter de la tradition	Les algorithmes traditionnelsde chemin le plus court se concentrent sur l'optimisation des chemins dans la structure arborescente du chemin le plus court. Les stratégies basées sur les déviations dans l'algorithme SB remettent en question ce	- Exploration des écarts : SB explore les chemins s'écartantde l'arbre des chemins les pluscourts. - Traitement des chemins : Traitement systématique des chemins dans une file d'attenteprioritaire. - Distinguer les cas simples etnon simples.	- Exploration de diverses voies : SB explore un spectre plus large de voies, découvrant desefficacités cachées et des itinéraires alternatifs. - Efficacité et adaptabilité : équilibre l'efficacité et l'exploration, en s'adaptant à des problèmes spécifiques. - Applications dans divers

	paradigme en explorant leschemins qui s'écartent de l'arbre.		domaines : implications dans les transports, les réseaux de communication, l'optimisation,etc.
Exploration efficace : un équilibre atteint	L'algorithme SB explore les « écarts » au-delà des arbres traditionnels des chemins les plus courts, en évaluant leur divergence et leurs avantages potentiels. Il intègre de manière transparente ces chemins, élargissant ainsi les options de solutions.	- Exploration délibérée : SB explore les déviations au-delà de l'arbre. - Évaluation : écarts évalués pour les divergences et les avantages. - Intégration : écarts parfaitement intégrés en fonction des coûts et des gains.	- Exploration de chemins diversifiés : SB propose une gamme plus large de chemins. - Approche holistique : équilibre les divergences et les avantages, en offrant des voies non conventionnelles mais précieuses.
Exploration efficace : un équilibre atteint	L'algorithme SB équilibre efficacité et exploration en choisissant sélectivement les déviations ayant un potentiel d'optimisation. Cette approche garantit des solutions pratiques sans complexité excessive.	Exploration Sélective des Déviations : SB explore sélectivement les déviations avec un potentiel d'optimisation. - Efficacité Pratique : Maintient des solutions pratiques sans complexité excessive.	- Exploration efficace des Solutions : SB explore efficacement les solutions. - Approche pratique : Éviter la complexité inutile tout en préservant les avantages de l'exploration.

Implications et applications : au-delà du chemin le plus court	Les stratégies de déviation de l'algorithme SB ont des applications diverses, notamment dans les transports et les réseaux de communication, où elles réduisent la congestion et optimisent le routage. Dans l'optimisation, elles encouragent la recherche de solutions pragmatiques et non conventionnelles, reflétant l'évolution des techniques d'optimisation.	Diversité d'applications : Applicables dans divers domaines. - Réduction de la Congestion : Utiles pour les transports et les réseaux de communication. - Pensée non conventionnelle : Encourageant des solutions créatives. - Évolution de l'optimisation : S'adapte aux techniques en évolution.	- Applications variées : Utiles dans de nombreux domaines. - Efficacité en Réduction de Congestion : Optimise la fluidité. - Approche Créative : Favorise des solutions hors norme. - Adaptabilité aux nouvelles Techniques : Évolue avec les méthodes d'optimisation.

4. Efficacité de calcul et vitesse pratique

Une caractéristique déterminante de l'algorithme SB est sa remarquable efficacité de calcul. Contrairement à certains homologues algorithmiques qui peuvent succomber aux fardeaux d'une complexité de calcul excessive, l'algorithme SB trouve habilement un équilibre entre l'exploration méticuleuse du chemin et la vitesse pratique. La nature basée sur la déviation de l'algorithme lui permet de réduire considérablement l'espace de recherche, ce qui en fait une solution attrayante pour les scénarios impliquant de grands graphes ou des applications exigeant une prise de décision en temps réel, comme le routage dynamique du réseau et l'allocation des ressources [6].

5. Contributions de Kurz et Murzel : améliorer l'énumération des chemins

L'avènement de l'algorithme Sidetrack Based (SB), conceptualisé par Kurz et Murzel, a inauguré un changement de paradigme transformateur dans le domaine des algorithmes d'énumération de chemins. Leur travail de pionnier a non seulement introduit les stratégies basées sur la déviation discutées précédemment, mais a également marqué une avancée significative dans l'efficacité et l'efficience de la résolution du problème des k chemins simples les plus courts. Cette section se penche sur les contributions notables de Kurz et Murzel, mettant en lumière leur approche innovante et son impact sur le domaine de la théorie des graphes et de l'optimisation.

5.1. La naissance de l'algorithme SB

Les travaux fondateurs de Kurz et Murzel ont dévoilé l'algorithme Sidetrack Based (SB) comme une solution ingénieuse au problème des k chemins simples les plus courts. Leur travail était une

réponse au besoin d'une approche d'énumération des chemins plus polyvalente et efficace, qui pourrait transcender les limites des méthodes traditionnelles et produire un éventail plus riche de chemins viables. Le fondement de l'algorithme réside dans l'utilisation stratégique de stratégies basées sur la déviation, une rupture avec le paradigme conventionnel du plus court chemin qui a favorisé une nouvelle frontière d'exploration.

5.2. Consommation de mémoire efficace : un triomphe pragmatique

Les contributions inestimables de Kurz et Murzel se sont étendues bien au-delà du domaine des algorithmes de graphes théoriques, plongeant dans les subtilités pratiques qui sous-tendent la mise en œuvre d'algorithmes efficaces. Une facette notable de leur travail consistait à répondre à une préoccupation pressante dans la conception algorithmique : la consommation de mémoire. Alors que les stratégies innovantes basées sur la déviation de l'algorithme Sidetrack Based (SB) se sont révélées très prometteuses pour résoudre le problème des k-chemins les plus courts, leur application n'a pas été sans défis potentiels. Plus précisément, l'utilisation de ces stratégies nécessitait le stockage de plusieurs arborescences de chemins les plus courts, ce qui entraînait potentiellement une augmentation de la demande de ressources mémoire.

Reconnaissant le rôle central que joue la consommation de mémoire dans l'efficacité et l'évolutivité des algorithmes, Kurz et Murzel se sont lancés dans un voyage inventif pour atténuer ce problème. Leur approche était fondée sur l'équilibre délicat entre l'efficacité de calcul de l'algorithme SB et ses besoins en mémoire. Cette entreprise a été guidée par l'objectif primordial de s'assurer que l'aspect pratique de l'algorithme est maintenu dans des scénarios du monde réel, où les contraintes de ressources régissent souvent la prise de décision. Leurs adaptations stratégiques visaient à optimiser le stockage et l'organisation de la multitude d'arbres de plus courts chemins utilisés par l'algorithme SB.

Le résultat des efforts diligents de Kurz et Murzel a été une version raffinée de l'algorithme SB qui a non seulement conservé ses principes fondamentaux basés sur la déviation, mais a également démontré une capacité améliorée à gérer judicieusement les ressources mémoire. En introduisant des techniques et des algorithmes de gestion de la mémoire innovants spécifiquement adaptés aux exigences du problème des k-chemins les plus courts, Kurz et Murzel ont amélioré l'applicabilité et la viabilité de l'algorithme. Cette version améliorée de l'algorithme SB, caractérisée par son approche nuancée de la consommation de mémoire, a montré son adaptabilité aux paramètres pratiques.

Bien qu'il reste indéniable que les besoins en mémoire continuent d'être une considération primordiale lors de la mise en œuvre de l'algorithme SB, le travail de pionnier de Kurz et Murzel a mis en lumière sa polyvalence. Leurs efforts ont non seulement relevé le défi de la consommation

de mémoire, mais ont également mis en lumière l'adaptabilité plus large de l'algorithme. En somme, leurs contributions ont fait de l'algorithme SB une solution pragmatique et robuste pour les applications du monde réel, en particulier dans des contextes où la gestion prudente des ressources mémoire est primordiale.

5.3. Réaliser une efficacité pratique : un paradigme d'optimisation

L'un des aspects les plus remarquables des contributions de Kurz et Murzel est leur synthèse réussie de la théorie et de la pratique. L'algorithme SB n'était pas simplement une percée conceptuelle ; c'était un algorithme qui prospérait dans les applications pratiques. Ses stratégies basées sur la déviation, associées à des techniques d'optimisation de la mémoire, ont propulsé l'efficacité de l'algorithme au premier plan. Cette réalisation a créé un précédent pour les algorithmes d'optimisation qui pourraient combler de manière transparente le fossé entre les avancées théoriques et les défis de mise en œuvre dans le monde réel.

5.4. Influence et héritage : Façonner le paysage

Les contributions de Kurz et Murzel ne se sont pas terminées avec la création de l'algorithme SB ; au contraire, leur travail s'est répercuté sur les communautés de la théorie des graphes et de l'optimisation. L'impact de l'algorithme SB s'est étendu au-delà de sa forme originale, inspirant les chercheurs ultérieurs à s'appuyer sur ses fondations, à affiner ses techniques et à étendre ses capacités. La fusion de l'algorithme des stratégies basées sur la déviation, des considérations d'efficacité et de l'applicabilité pratique a servi de catalyseur pour une exploration et une innovation plus poussées dans le domaine de l'énumération des chemins.

Au fur et à mesure que nous parcourons les sections suivantes de ce chapitre, nous lançons dans une exploration approfondie de l'algorithme Sidetrack Based. Notre voyage nous emmène à travers ses mécanismes internes, les subtilités de ses stratégies informatiques et les vastes implications de son déploiement. De plus, nous engageons des discussions approfondies sur l'efficacité pratique de l'algorithme, les défis potentiels et les réalisations notables. Grâce à cette analyse insuffisante, nous nous efforçons de fournir une compréhension globale et perspicace des contributions de l'algorithme SB dans le domaine, mettant en lumière son rôle central dans l'avancement de notre compréhension et de nos capacités à résoudre le problème des k chemins simples les plus courts.

6. Algorithme SB

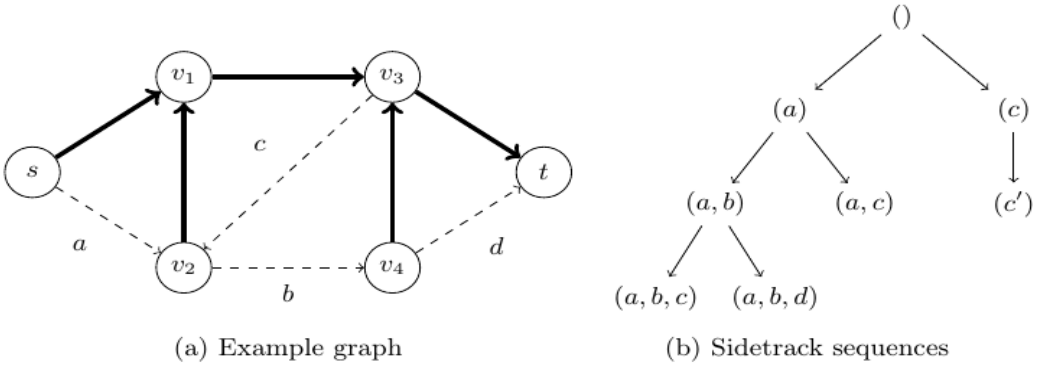

Figure 2.1 : graphique pour expliquer l'algorithme SB.

Dans cette section, nous présentons une approche relativement simple pour énumérer les k chemins simples les plus courts, posant ainsi les bases des modifications ultérieures visant à atteindre la garantie de temps d'exécution souhaitée. Notre stratégie repose sur l'utilisation d'une file de priorité initialement vide, désignée sous le nom de Q, qui joue un rôle essentiel dans la gestion de l'exploration des chemins candidats. La priorité de chaque chemin au sein de Q est déterminée par sa longueur. L'algorithme commence par calculer l'arbre de chemin le plus court initial T0, puis ajoute son unique chemin source-vers-cible (s-t), représenté par une séquence de sidetracks vide, à la file de priorité Q [6].

Par la suite, les chemins contenus dans Q sont systématiquement traités dans l'ordre croissant de leurs longueurs jusqu'à ce que k chemins simples soient identifiés. La visualisation de ce processus et le fonctionnement de l'algorithme de base peuvent être trouvés dans la Figure 1. Notamment, cette représentation montre l'association des sidetracks avec des arbres de chemin le plus court spécifiques et illustre la relation entre les chemins parent et enfant [6].

Lorsqu'une séquence de sidetracks (e1, ..., er) est extraite de Q, correspondant à un chemin p, l'algorithme distingue entre les cas simples et non simples. Dans le cas d'un chemin simple, l'algorithme produit initialement en sortie le chemin p. Pour chaque sidetrack e = (u, v) où u appartient au suffixe de p, un nouveau chemin p0 est généré. Ce chemin, représenté par la séquence (e1, ..., er, e), possède un nœud de déviation u. Ce chemin nouvellement formé, p0, est ensuite inséré dans Q pour une exploration ultérieure. La longueur de p0 est commodément calculée comme la somme du coût du chemin p et du coût de sidetrack δTe (e). Il est important de noter que les sidetracks provenant du nœud cible peuvent être ignorés [6].

Considérez un exemple illustratif représenté dans la Figure 1. La séquence de sidetracks (a) est associée à Ta = T0, indiquant un chemin simple p qui parcourt les nœuds s, v2, v1, v3 et t dans l'ordre. Le suffixe de p est composé de son sous-chemin v2-t, avec les sidetracks b et c s'étendant

à partir de ce suffixe. Par conséquent, lorsque la séquence (a) est extraite de Q, le chemin p est produit en sortie, et les séquences (a, b) et (a, c) sont ensuite insérées dans Q avec les arbres de chemin le plus court associés (Ta = Tb = Tc = T0) [6].

Dans les cas où un chemin non simple p est extrait, représenté par la séquence de sidetracks (e1, ..., er), l'attention de l'algorithme se tourne vers l'extension potentielle de la concaténation du chemin préfixe de p et du sidetrack er pour créer un chemin simple s-t. Cela implique d'examiner la validité de l'extension, en veillant à éviter les nœuds de pref(p) au-delà du nœud terminal v de er. Pour y parvenir, un nouvel arbre de chemin le plus court T est calculé, en considérant les nœuds de G - pref(p) plutôt que de G pour garantir que les nœuds du chemin préfixe ne sont pas réutilisés. Si l'extension s'avère impossible, le chemin p est rejeté. Cependant, si l'extension est réalisable, l'algorithme réinsère la séquence (e1, ..., er) dans Q, en associant T à er. Cette nouvelle séquence représente désormais un chemin simple p0, obtenu en concaténant le chemin préfixe de p, le sidetrack er et le chemin w-t dans T qui contourne les nœuds de pref(p). La longueur de p0 est calculée comme la somme de c(pref(p)), c(er) et d(w) [6].

L'algorithme navigue continuellement dans ce processus complexe d'exploration de chemins simples et non simples, enrichissant progressivement le réservoir de solutions potentielles et contribuant à l'énumération des k chemins simples les plus courts dans des graphes dirigés [6].

Nécessite : Un digraphe $D = (V, A)$, `une source s` $\in$ `V , un puits t` $\in$ `V et un entier k`

Assurez-vous que : k chemins s-t simples les plus courts

1 : Soit$Candidat \leftarrow \emptyset, T \leftarrow \emptyset$ et $Sortie \leftarrow \emptyset$

2 : $T_0 \leftarrow$ un branchement SP de D enraciné en t contenant s

3 : Ajouter $((T_0), l(P_{st}(T_0)), \zeta = 1)$ au candidat

4 : **tant que** Candidat $\neq \emptyset$ et |Sortie| < k faire

5 : $\mu = (\varepsilon = ((T_0, e_0, \cdots, T_h, e_h = (v_h, w_h), T_h + 1), lb, \zeta) \leftarrow$ un élément dans Candidat avec une longueur minimale (ou limite inférieure minimale lb)

6 : **si** $\zeta = 1$alors

7 : Extrayez μ du candidat et ajoutez ε à la sortie

8 : **pour** tout écart $e = w_j v'$ avec $w_j \in P_{w_h t}^{T_{h+1}}$ **faire**

9 : $\varepsilon' \leftarrow (T_0, e_0, \cdots, T_h, e_h, T_{h+1}, e, T_{h+1})$

10 : $lb' \leftarrow lb - l(P_{w_j t}^{T_{h+1}}) + l(e) + l(P_{v' t}^{T_{h+1}})$

11 : **si** ε' représente un chemin simple alors

12 : Ajouter $\mu' = (\varepsilon', lb', \zeta = 1)$au candidat

13 : **sinon**

14 : $T_0 \leftarrow$ le nom d'un SP en embranchement de $D_j(P)$ //T_0 n'est pas encore calculé

15 : Ajouter T_0à T

16 : Ajouter $\mu'' = (\varepsilon'' = (T_0, e_0, \cdots, T_h, e_h, T_{h+1}, e, T_0), lb', \zeta = 0)$ au Candidat

17 : **sinon**

18: **si** T_{h+1} n'a pas encore été calculé alors

19: Calculer T_{h+1} , un SP en ramification de $D_j(P)$ et l'ajouter à T

20: Soit $\mu' = (\varepsilon' = (T_0, e_0, \cdots, T_h, e_h, T_{h+1}), lb + l(P_{v't}^{T_{h+1}}) - l(P_{v't}^{T_h}), \zeta = 1)$

21: Ajouter μ' au candidat

22 : **retour** sortie

Algorithme Sidetrack Based (SB) pour le kSSP[17][6]

7. Analyse

Dans le scénario où chaque extraction initiale d'un chemin candidat conduit à un détour non simple, cela déclenche ensuite un appel à un algorithme du plus court chemin (SP). Il est important de noter que la limitation d'un seul appel SP par sommet sur un chemin est assurée par les vérifications à la ligne 18. Par conséquent, la complexité computationnelle de l'algorithme SB est bornée par O(kn(m + n log n)), étant donné que le nombre de sommets sur un chemin simple est restreint par n et que l'algorithme s'arrête dès que k chemins sont ajoutés à la sortie (Output).

Deux améliorations fondamentales ont permis à l'algorithme SB de surpasser d'autres approches dans la résolution pratique du problème kSSP. Tout d'abord, il évite d'effectuer un appel SP si le détour est simple. Deuxièmement, lorsque le détour n'est pas simple, il est intégré avec une borne inférieure sur sa longueur et l'appel correspondant à l'algorithme SP est reporté. Ainsi, si ce détour aboutit à un chemin long (un chemin dont la longueur, ou la limite inférieure de longueur, dépasse celle du k-ième chemin le plus court), l'appel à l'algorithme SP ne sera jamais exécuté.

Cependant, il est important de reconnaître que la consommation de mémoire de l'algorithme SB peut être considérable, car il stocke des ensembles complets de détours en mémoire. Cela peut entraîner une consommation importante de mémoire, bien plus que l'algorithme CN, qui ne stocke qu'un ensemble de détours tout en conservant l'ensemble des chemins calculés.[17]

8. Conclusion

"Dans le chapitre, nous avons plongé dans le monde innovant des algorithmes Sidetrack Based (SB), en nous concentrant spécifiquement sur le concept révolutionnaire des stratégies basées sur les déviations. Nous avons vu comment ces stratégies remettent en question les méthodologies algorithmiques conventionnelles, qui mettent principalement l'accent sur le raffinement de l'arbre du chemin le plus court. SB les algorithmes, en revanche, s'affranchissent de la tradition en explorant délibérément des chemins qui s'écartent de la structure arborescente standard des chemins les plus courts.

Cette rupture avec la tradition ouvre de nouveaux horizons dans l'exploration des sentiers, révélant des efficacités cachées, des itinéraires spécialisés et des alternatives non conventionnelles mais pragmatiques. Grâce à une sélection et une évaluation minutieuses des écarts, les algorithmes SB établissent un équilibre délicat entre l'efficacité informatique et l'exploration approfondie des solutions.

De plus, nous avons exploré les implications et les applications à grande échelle des stratégies basées sur les déviations, soulignant leur potentiel dans des domaines tels que les transports, les réseaux de communication et l'optimisation. Ces stratégies nous encouragent à penser au-delà des conventions, nous rappelant que le meilleur chemin n'est pas toujours le plus court, mais celui qui permet de naviguer le plus efficacement sur le terrain complexe des scénarios du monde réel.

Le chapitre 2 jette les bases d'une compréhension plus approfondie des algorithmes SB, ouvrant la voie à une exploration plus approfondie de leurs méthodologies, de leurs avantages et de leur impact dans le monde réel. Au fur et à mesure que nous progressons, nous continuerons à dévoiler les subtilités et les applications pratiques de cette approche révolutionnaire de l'optimisation et de l'exploration des chemins. »

Chapitre 3 : Comparaison de l'algorithme parcimonieux basé sur Sidetrack et de l'algorithme original basé sur Sidetrack

1. Introduction

Dans le domaine des algorithmes informatiques, le progrès et l'innovation découlent souvent de la recherche d'améliorations des performances, de l'efficacité et de l'adaptabilité. Ce chapitre se lance dans une exploration complète de l'évolution d'une approche algorithmique bien établie : l'algorithme basé sur le détournement. S'appuyant sur les fondations posées au chapitre 2, où les subtilités de l'algorithme original basé sur le détournement ont été examinées, l'accent est maintenant mis sur l'algorithme parcimonieux basé sur le détournement et ses contributions notables.

Les algorithmes basés sur le détournement sont depuis longtemps appréciés pour leur capacité à naviguer dans des espaces de problèmes complexes, à échapper aux optima locaux et à rechercher des solutions optimales à l'échelle mondiale dans la poursuite de l'optimisation. À ce titre, ils ont trouvé des applications dans divers domaines, notamment l'ingénierie, la logistique, l'intelligence artificielle et la finance. Cependant, aucun algorithme n'est à l'abri des limites ou du potentiel d'amélioration.

L'algorithme parcimonieux basé sur le détour reconnaît les atouts inhérents de l'algorithme original basé sur le détour tout en identifiant les opportunités d'amélioration. Le résultat de ces efforts est un algorithme modifié basé sur le détournement, conçu pour surmonter certaines limitations, améliorer l'efficacité et étendre sa portée à des domaines problématiques plus larges.

Ce chapitre vise à faire la lumière sur la manière dont l'algorithme parcimonieux basé sur le détour affine l'algorithme original basé sur le détour. En comparant ces deux approches, les subtilités de leurs différences et les implications pour leur application pratique sont dévoilées.

Le fonctionnement interne de l'algorithme parcimonieux basé sur Sidetrack est disséqué, y compris sa stratégie d'exploration améliorée, ses améliorations d'efficacité et sa nouvelle évolutivité. Grâce à une méthodologie de comparaison soigneusement conçue, les performances de ces algorithmes dans diverses instances de problèmes sont évaluées, en tenant compte des métriques, des ensembles de données et des configurations expérimentales pertinentes.

Le point culminant de cette analyse comparative promet non seulement une compréhension approfondie des modifications elles-mêmes, mais également un aperçu du paysage algorithmique dans lequel elles évoluent. L'algorithme parcimonieux basé sur Sidetrack s'appuie sur les fondations de son prédécesseur et cherche à fournir des résultats supérieurs, une plus grande adaptabilité et de nouvelles possibilités d'optimisation à l'ère numérique.

Rejoignez-nous dans ce voyage alors que nous explorons la synergie entre tradition et innovation, en cherchant à découvrir le potentiel de transformation de l'algorithme parcimonieux basé sur le détournement et ses implications pour le domaine de l'optimisation informatique.

2. L'algorithme parcimonieux basé sur Sidetrack

La principale différence est que l'algorithme PSB stocke deux types d'éléments dans Candidate. Le premier type, de la forme: $(\varepsilon = (T_0, e_0, T_1, e_1, \cdots, T_h, e_h = (v_h, w_h), T_{h+1}), lb)$, représente un simple chemin $s - t$ P de longueur lb Contrairement à l'algorithme SB, le T_{h+1} en branchement n'a pas nécessairement encore été calculé. Le deuxième type, de la forme (ε, Dev, lb), contient un champ supplémentaire Dev (expliqué ci-dessous) et, dans ce cas, tous les branchements entrants $T_1, \cdots, T_{h+1}$ sont déjà calculés. Commençons par considérer une étape de l'algorithme PSB lorsqu'un élément $\mu = (\varepsilon = (T_0, e_0, T_1, e_1, \cdots, T_h, e_h = (v_h, w_h), T_{h+1}), lb)$ représentant un simple le chemin P est extrait de Candidat. T_{h+1} est calculé à cette étape (si ce n'est déjà fait) ce qui permet d'obtenir P explicitement (notez que dans ce cas lb est égal à la longueur de P). Ensuite, l'algorithme PSB ajouté $P = (s = v0, \cdots, vi = wh, \cdots, vr = t)$ à la sortie et (comme l'algorithme SB), pour chaque v ∈ {vi , · · · , vr }, et pour toute déviation e de queue v, le détour Q(v, e) de P en e est considéré. Si Q(v, e) est simple, alors μ0 = ((T0, e0 , T1 , e1 , · · · , Th, eh , Th+1 , e, Th+1), `(Q(v, e) )) est ajouté au Candidat. Dans le cas contraire, l'écart e est ajouté à un ensemble Dev (initialement vide). Une fois tous les écarts pris en compte, l'élément (unique) (ε, Dev, lb0 ) est ajouté au Candidat, où lb0 = minfj =(uj ,u0 j )∈Dev `(Q(uj , fj)). Autrement dit, Dev est l'ensemble de tous les « écarts non simples » de P aux sommets entre wh et t, ordonnés par rapport à l'indice de leur queue sur P , c'est-à-dire pour deux écarts fi = (ui , u0 i), fj = (uj , u0 j) ∈ Dev, fi ≤ fj si et seulement si i ≤ j. Enfin, soit lb0 une borne inférieure de la longueur des détours pour un écart de Dev. La différence importante entre les algorithmes SB et PSB vient du fait que les détours non simples sont considérés comme un objet unique par l'algorithme PSB.

Considérons maintenant une étape où l'algorithme PSB extrait un élément μ = (ε = (T0, e0 , T1 , e1, · · · , Th , eh, Th+1), Dev = {f1 , · · · , fj = (uj , u0 j), · · · , fl }, lb) du candidat. Comme

mentionné ci-dessus, dans ce cas, ε code un simple chemin st (v0, · · · , vr). Soit 1 ≤ min ≤ l le plus petit entier tel que lb = `(Q(umin , fmin)).

Ensuite, l'algorithme PSB procède comme suit. Pour j décroissant de l à min, un Tj 0 T 0 entrant dans D \ {v0 , · · · , vij = uj } est calculé (mais pas stocké !) jusqu'à ce qu'un chemin Pu0 j t de u0 j à t soit découvert (si aucun chemin de ce type n'existe, j est diminué de un). Si Pu0 j t existe, alors εj = (T0 , e0 , T1 , e1, · · · , j0 TTh , eh, Th+1, fj , Tj 0) représente un simple chemin s-t de longueur lbj = `((v0 , · · · , vij ))+`(fj)+`(Pu0 j t). j0 Ensuite, l'élément µj = (εj , lbj) est ajouté au Candidat, mais Tj n'est pas stocké (l'algorithme PSB devra peut-être le recalculer ultérieurement). Une deuxième clé est que pour accélérer le calcul, l'amélioration de Tj 0 est en fait calculée en mettant à jour Tj+1 0 . ce qui est fait en utilisant les outils standards de [57]. Alors seulement

Lorsque j = min, le Tmin 0 en branchement est stocké et µmin = (εmin, lbmin) est ajouté au candidat.

La raison pour laquelle Tmin0 est stocké (alors que les autres Tj 0 ne le sont pas) est que µmin devrait bientôt être extrait du candidat (car le chemin représenté par εmin devrait être court) et nous voulons éviter le recalcul de Tmin 0 . Finalement, l'élément µ0 = (ε, Dev 0 = {f1 , · · · , fmin−1 }, lb0)

est ajouté à Candidate, où lb0 est la longueur minimale des détours non simples dans Dev 0 .

La justesse découle de celle de l'algorithme SB. De plus, comme la plupart des branchements entrants calculés ne sont pas stockés, la mémoire de travail utilisée par PSB est nettement inférieure à celle utilisée par PSB pour l'algorithme SB.[17]

3. Variantes spéciales du PSB

Un meilleur compromis espace-temps que le PSB peut être obtenu si chacun est calculé et stocké dans le branchement va être utilisé dans les étapes futures, c'est-à-dire qu'il est utilisé pour extraire un chemin candidat simple qui va être extrait du Candidat (avant le k ème chemin le plus court). Malheureusement, un telles les informations ne peuvent pas être fournies car la longueur du k ème chemin n'est pas connue au préalable. Toutefois, si le calcul d'un branchement interne conduit à construire un chemin de longueur relativement courte, disons inférieure à une valeur seuil multipliée par le chemin de sortie actuel, alors le stockage de ce branchement entrant est significatif car l'extraction de son élément correspondant du formulaire Candidat est attendue prochainement et sa sauvegarde conduit à économiser son calcul redondant. Nous présentons ici deux variantes du PSB ; PSB-v2 et PSB-v3. PSB-v2 est une petite amélioration du PSB conduisant à consommer moins de mémoire en stockant moins le branchement tandis que le PSB-v3 offre un compromis adaptable en fonction de la valeur du seuil. Considérons, encore une fois, une étape où

l'algorithme PSB extrait un élément μ = (ε, Dev ={f1, · · · , fj = (uj , u0 j), · · · , fl}, lb) de Candidat avec 1 ≤ min ≤ l le plus petit entier tel que lb = `(Q(umin , fmin)). L'algorithme de l'algorithme PSB itère sur j, décroissant de l à min comme expliqué ci-dessus, un Tj 0 de branchement correspondant est calculé pour chaque j (mais non stocké). Alors, seulement lorsque j = min, le Tmin entrant est stocké.

L'algorithme PSB-v2 ne stocke pas naïvement Tmin0 . Au lieu de cela, Tmin0 n'est stocké que si la longueur du détour correspondant est inférieur à une valeur seuil θ, multipliée par la longueur du chemin simple le plus court 0 Tpath dans Candidat. Autrement dit, lbmin = `((v0 , · · · , vimin ))+`(Pumin min t) ≤ θ∗`(Psuivant), où Psuivant est le chemin simple le plus court dans Candidat. En conséquence, si le Tmin en branchement 0 conduit à un (relativement) long chemin qui ne devrait pas être extrait très prochainement du Candidat, alors il est libéré du mémoire.

Le PSB-v3 se comporte à chaque écart de Dev entre min et l de la même manière que le PSB-v2 se comporte avec fmin. Précisément, pour chaque écart fj avec min ≤ j ≤ l, PSB-v3 calcule son correspondant le branchement entrant Tj 0. Ce branchement entrant (Tj 0) n'est stocké que si la longueur de son détour correspondant est inférieur à un seuil θ, multiplié par la longueur du chemin simple le plus court dans Candidat. La valeur du seuil θ pourrait changer dynamiquement au cours de l'exécution. Par exemple, il pourrait être lié au rapport entre les deux chemins à venir, c'est-à-dire les deux éléments du candidat avec une longueur minimale.[17]

4. Conception et architecture

Conformément à des recherches antérieures [6], une structure de données de tas d'appariement [56] a été implémentée pour prendre en charge l'opération de touche de diminution, qui est utilisée pour l'algorithme de chemin le plus court de Dijkstra. L'implémentation de l'algorithme d'arbre des chemins les plus courts de Dijkstra est considérée comme paresseuse, ce qui signifie que le calcul cesse dès que la distance entre le nœud de requête v et t s'avère être la plus courte. D'autres calculs peuvent être effectués ultérieurement pour un autre nœud v0 situé à une plus grande distance de t, en partant de l'arbre partiel des chemins les plus courts déjà calculé. L'implémentation de l'algorithme de Dijkstra prend également en charge une opération de mise à jour lorsqu'un nœud ou un arc est ajouté au graphe. De plus, une opération de copie spéciale a été implémentée qui permet la mise à jour du branchement entrant lorsqu'un ensemble de nœuds est supprimé du graphique. Cela correspond aux opérations effectuées lors de la création d'un Th+1 entrant à partir de Th dans SB*.[17]

Il convient de noter que dans leurs implémentations, le paramètre k ne fait pas partie de l'entrée et que, par conséquent, les ensembles de candidats sont simplement implémentés à l'aide de tas

d'appariement. Ce choix permet d'utiliser ces méthodes comme itérateurs, capables de renvoyer le prochain chemin le plus court tant qu'il existe. Cependant, si k faisait partie de l'entrée, la structure de données utilisée pour stocker les candidats pourrait être modifiée pour contenir uniquement les k meilleurs candidats, mais l'algorithme ne renverrait que k chemins exactement, même s'il en existe d'autres. De plus, pour les algorithmes SB, SB*, PSB, PSB-v2 et PSB-v3, d'après [17], les candidats sont stockés dans deux tas, le premier pour stocker les candidats simples $Candidate_{simple}$ et le second pour stocker les candidats simples. stocker les candidats non simples $Candidate_{not-simple}$. Ensuite, les candidats sont extraits de $Candidate_{simple}$ tant que la longueur du chemin simple le plus court est inférieure ou égale à la longueur du chemin non simple le plus court dans $Candidate_{not-simple}$. De cette façon, la priorité est donnée à l'extraction de chemins simples.

Concernant les algorithmes PSB-v2 et PSB-v3, basés sur des expériences préliminaires, ils ont choisi de mettre à jour la valeur de θ de manière dynamique par rapport au rapport entre la paire de chemins à venir. Lors de la recherche du ième chemin, un branchement entrant correspondant ne sera stocké que si la longueur de ce chemin est au plus θ fois la longueur du (i − 1)ème chemin. Plus précisément, soit `s la longueur du plus petit élément de Candidatesimple et soit `ns la longueur du plus petit élément non simple de Candidatenot−simple. Ces valeurs sont toutes deux définies sur 1 si l'un des ensembles correspondants est vide. Soit également c = max(`s, `ns) et ρ = c − 1. La valeur de θ est fixée à 1 + αρ, pour une constante α > 0. L'intuition derrière cela est de stocker un branchement entrant uniquement si il devrait être utilisé prochainement, c'est-à-dire lors de l'extraction de l'un des chemins à venir. Le choix de ces valeurs (`s et `ns) fournit une indication significative, et la valeur de θ est facile à calculer. Dans leurs expériences, ils ont fixé le facteur α dans la formule de calcul de θ à α = 11, sur la base d'expériences préliminaires.[17]

Réseaux routiers :

Les réseaux routiers représentent le réseau complexe de rues et d'intersections au sein d'une ville. Dans ce contexte, un graphe orienté est utilisé pour modéliser ces réseaux routiers. Chaque carrefour est associé à un sommet, et un arc d'une longueur de « w » (mesurée en kilomètres) relie deux sommets si et seulement s'il existe une route de longueur physique « w » entre le carrefour correspondant. Les réseaux routiers présentent généralement certaines caractéristiques :

Disparité: les réseaux routiers sont clairsemés, ce qui signifie que seule une fraction des liaisons routières possibles est réalisée.

Presque planaires : elles ont tendance à être presque planes, ce qui suggère que la majorité des routes peuvent être tracées dans un plan presque bidimensionnel.

Degré délimité: le degré des sommets (nombre de routes reliées à un carrefour) est délimité, ce qui indique que les carrefours se connectent généralement à un nombre limité d'autres carrefours.

Dans l'évaluation, ils distinguent deux catégories de réseaux routiers :

Petits réseaux routiers: cette catégorie englobe les réseaux routiers de villes telles que ROME, DC et DE.

Grands réseaux routiers: En revanche, les « grands » réseaux routiers comprennent ceux de villes comme NY, BAY et COL.

Pour une compréhension plus détaillée de ces réseaux routiers, nous fournissons leurs caractéristiques dans le tableau 2.

Réseaux complexes: Les réseaux complexes servent de modèles pour divers types de systèmes et d'interactions complexes. Ces réseaux partagent souvent plusieurs caractéristiques déterminantes :

Propriété du petit monde: les réseaux complexes ont tendance à posséder une propriété du petit monde, caractérisée par un diamètre logarithmique, ce qui signifie que les nœuds peuvent être atteints les uns des autres relativement rapidement.

Distribution des degrés en loi de puissance : ils présentent une distribution des degrés en loi de puissance, ce qui implique qu'un petit nombre de nœuds ont un degré significativement plus élevé que le reste.

Coefficient de clustering élevé: les réseaux complexes ont souvent un coefficient de clustering élevé, indiquant que les nœuds ont tendance à former des clusters ou des communautés.

Parmi les réseaux complexes inclus dans notre évaluation figurent :

BIOGRID: Ce réseau modélise la létalité synthétique résultant de mutations ou de délétions génétiques qui conduisent à la létalité lorsqu'elles sont combinées au sein de la même cellule (97).

DIP: Représentant les interactions protéine à protéine, DIP est un réseau complexe capturant les interactions moléculaires (102).

Réseau FB: le réseau FB reflète les cercles sociaux extraits de Facebook, fournissant un aperçu des connexions sociales en ligne [89].

LOC: Dérivé du réseau social géolocalisé Brightkite, le réseau LOC représente les modèles de mobilité humaine et les interactions sociales.

network	n	m	D	(d)	$-\alpha$	(cc)	
ROME	**3352**	**8870**	**57**	**5.2**	**-**	**0.0025**	**Road network of Rome [44]**
DC	**9 559**	**29682**	**140**	**6.2**	**-**	**0.039**	**Road network of Washington DC [44]**
DE	**49109**	**119520**	**573**	**4.8**	**-**	**0.024**	**Road network of Delaware [44]**
NY	**264346**	**733846**	**664**	**5.5**	**-**	**0.02**	**Road network of New York [44]**
BAY	**321270**	**800172**	**791**	**4.9**	**-**	**0.016**	**Road network of San Francisco Bay area [44]**
COL	**435666**	**1057066**	**1219**	**4.8**	**-**	**0.017**	**Road network of Colorado area [44]**
BIOGRID	**2318**	**12580**	**7**	**21.7**	**1.96**	**0.20**	**Mutation/deletion of genes resulting in cell lethality [97**
FB	**3698**	**85963**	**6**	**93**	**-**	**0.61**	**Social circles from Facebook [89]**
P2P	**5606**	**23510**	**8**	**16.7**	**-**	**0.014**	**Peer-to-peer network of the Gnutella file sharing network [88]**
DIP	**13969**	**60621**	**17**	**17.4**	**2.38**	**0.11**	**Protein-protein interaction network [102]**
CAIDA	**29432**	**143000**	**9**	**19.4**	**2.06**	**0.42**	**Relationships between Autonomous Systems of the Internet [116]**
LOC	**33187**	**188577**	**11**	**22.7**	**2.25**	**0.29**	**Brightkite location-based social networking service provider [88]**

Tableau 3.1 Caractéristiques des graphes utilisés dans les expériences kSSP : nombre de nœuds (n), nombre des bords (m), diamètre (D), degré moyen (d), exposant $-\alpha$ de la distribution des degrés en loi de puissance et le coefficient de regroupement moyen (cc).[17]

5. Résultat

Lorsque vous naviguez dans le paysage de l'optimisation basée sur des graphiques, la sélection d'un algorithme peut avoir un impact significatif sur les résultats obtenus. Deux concurrents de premier plan dans ce domaine, à savoir l'algorithme Sidetrack Based (SB) et l'algorithme Pessimistic Sidetrack Based (PSB), ont attiré l'attention pour leur efficacité à résoudre le problème des k chemins les plus courts. Cependant, discerner les contextes dans lesquels un algorithme surpasse l'autre est crucial pour une sélection efficace d'algorithmes.

Dans cette section, nous entreprenons une exploration approfondie des résultats comparatifs entre SB et PSB. Notre quête de connaissances nous emmène sur deux terrains distincts : les rues historiques de Rome, comptant 3 352 nœuds et 8 859 tronçons, et le réseau routier tentaculaire de la région de la baie de San Francisco, avec un nombre stupéfiant de 321 270 nœuds et 794 830 tronçons. Ces environnements contrastés fournissent des toiles de fond convaincantes pour un examen algorithmique minutieux.

Notre analyse se concentre sur deux mesures de performance essentielles : le nombre d'arbres générés et le temps d'exécution de chaque algorithme. En faisant varier systématiquement le paramètre k, qui signifie le nombre de chemins simples les plus courts à énumérer, nous découvrons comment ces algorithmes répondent à diverses demandes de calcul.

Mesures de comparaison

k : Le nombre de chemins les plus courts à énumérer.

Nombre d'arbres : nombre d'arbres générés par chaque algorithme.

Temps d'exécution (ms) : temps nécessaire à chaque algorithme pour calculer les k chemins les plus courts.

Bleu : Algorithme SB.

Rouge : Algorithme PSB.

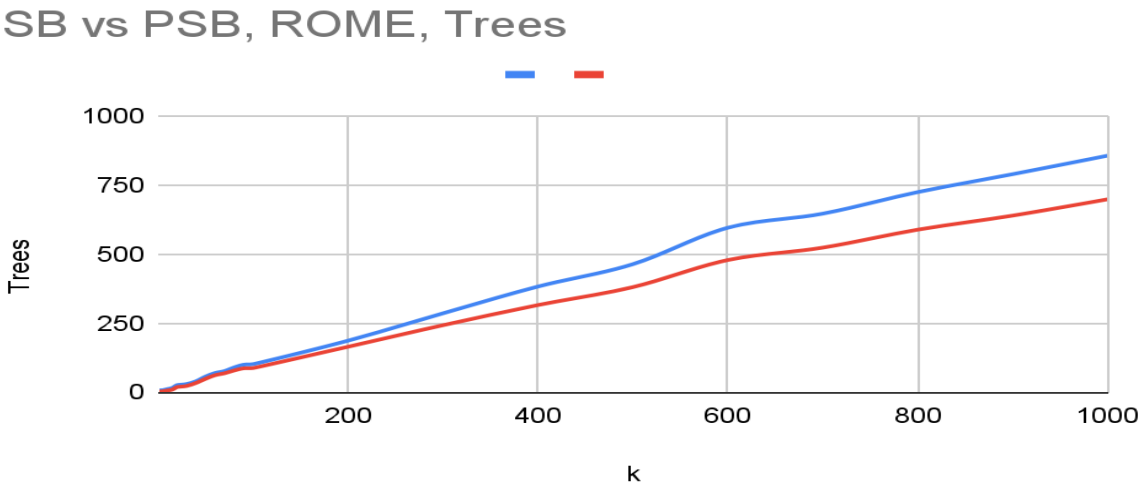

Figure 3.1. Le nombre d'arbres générés au fur et à mesure que k augmente "Rome"

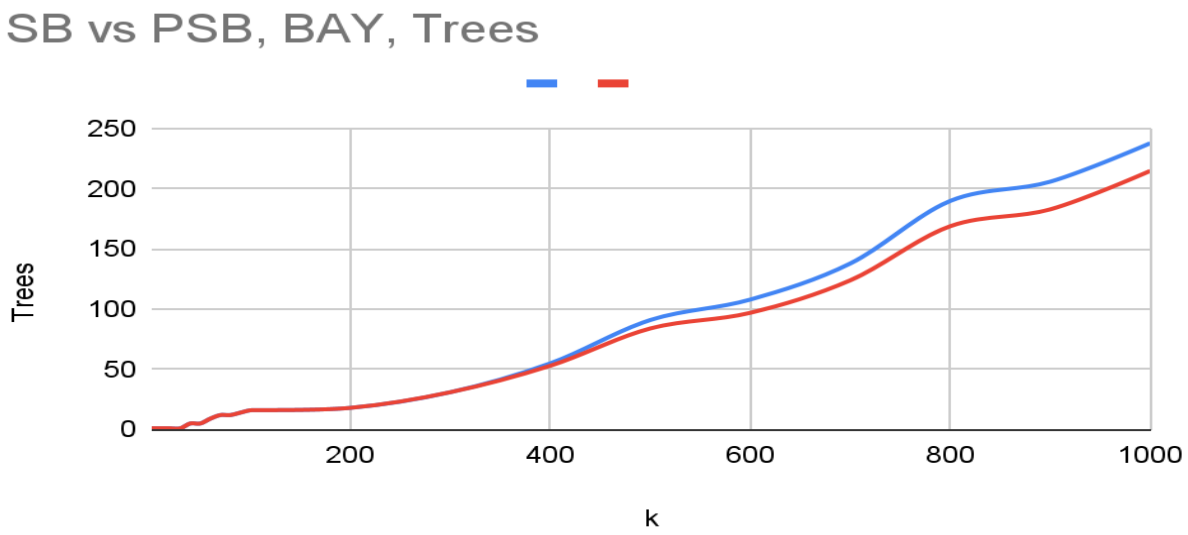

Figure 3.2 le nombre d'arbres générés au fur et à mesure que k augmente "BAY"

Cette observation nous conduit à réaliser une fascinante : le comportement algorithmique varie en fonction de la complexité et de l'échelle du réseau examiné. À Rome, où le réseau est relativement compact, la capacité de SB à générer un nombre supérieur d'arbres souligne son exploration approfondie des possibilités de chemins. En revanche, le vaste et complexe réseau routier de la région de la baie de San Francisco semble avoir atténué cet écart, indiquant que plus la complexité du réseau augmente, plus la différence de performance entre SB et PSB se réduit.

Ce schéma intrigant met en lumière l'adaptabilité de ces algorithmes à différents contextes de réseau. Il nous pousse à considérer les compromis entre la largeur de l'exploration et l'efficacité informatique. En fin de compte, le choix entre SB et PSB peut dépendre non seulement du nombre d'arbres souhaités, mais aussi des caractéristiques spécifiques du réseau en question.

Notre exploration se poursuit alors que nous plongeons plus profondément dans ces découvertes, cherchant à déchiffrer les interactions complexes entre la performance algorithmique et la complexité du réseau.

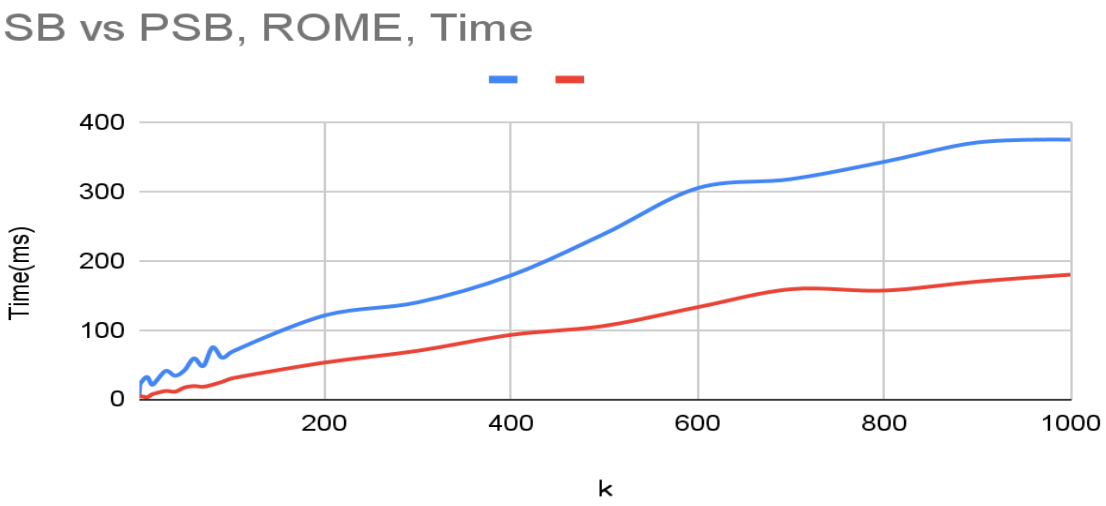

Figure 3.3 le temps en (ms) lorsque k augmente Rome

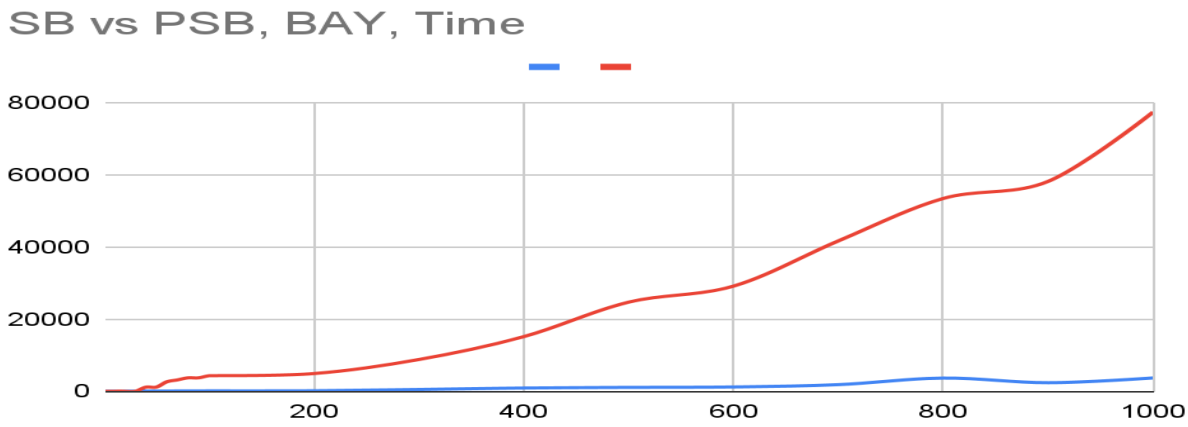

Figure 3.4 : le temps en (ms) au fur et à mesure que k augmente BAY

L'efficacité du temps est apparue comme un facteur crucial dans notre analyse comparative des algorithmes Sidetrack Based (SB) et Pessimistic Sidetrack Based (PSB) dans deux paysages de réseau différents. Dans le réseau de Rome, PSB a démontré une efficacité de temps remarquable, dépassant SB dans la découverte de chemins dans le contexte d'un environnement urbain plus petit et moins complexe. Cette performance rapide s'est avérée avantageuse pour naviguer dans les rues labyrinthiques de Rome. Cependant, la dynamique a considérablement changé lorsque nous avons traversé le réseau étendu et complexe de la région de la baie de San Francisco. Ici, le PSB a été confronté à des défis considérables, aux prises avec des temps de calcul prolongés. À l'opposé, SB a présenté une approche cohérente et pragmatique, offrant des temps de calcul raisonnables, même face à un réseau plus vaste et plus complexe. Ces observations soulignent le rôle critique de la taille et de la complexité du réseau dans la sélection des algorithmes, soulignant l'adéquation du PSB aux réseaux plus petits où la vitesse est primordiale, tout en soulignant la fiabilité du SB dans la gestion des demandes de calcul de structures de réseau plus grandes et complexes.

Rome

	k	trees	time (ms)
SB	1	1	2
	2	7	20
	5	8	27
	10	12	32
	15	16	21
	20	26	27
	30	30	41
	40	40	34
	50	57	42
	60	70	59
	70	77	48
	80	90	75
	90	100	60
	100	102	68
	200	187	121
	300	286	140
	400	383	179
	500	464	239
	600	596	305
	700	647	318
	800	725	343
	900	790	371
	1000	857	375

	k	trees	time (ms)
PSB	1	1	2
	2	3	4
	5	4	4
	10	7	3
	15	10	7
	20	19	9
	30	23	12
	40	34	11
	50	49	17
	60	62	19
	70	69	18
	80	79	21
	90	87	25
	100	88	30
	200	165	53
	300	243	70
	400	316	93
	500	381	106
	600	479	133
	700	524	159
	800	589	157
	900	640	170
	1000	699	180

BAy

	k	trees	time (ms)
SB	1	1	77
	2	1	62
	5	1	65
	10	1	62
	15	1	8
	20	1	65
	30	1	66
	40	5	79
	50	5	82
	60	9	90
	70	12	105
	80	12	107
	90	14	117
	100	16	132
	200	18	166
	300	31	515
	400	55	918
	500	91	1095

600	108	1209
700	138	1856
800	190	3672
900	206	2385
1000	238	3714

6. Concentrons-nous maintenant sur la discussion sur SB et PSB:

Les algorithmes **SB* et PSB**, comme souligné précédemment, présentent un avantage dans les réseaux complexes en raison de leur gestion efficace des détours aux sommets de haut degré. Ils calculent au plus un chemin le plus court dans le branchement à ces sommets, ce qui fonctionne pour tous les sommets voisins. Cette approche minimise la consommation de mémoire et améliore l'efficacité algorithmique. Cependant, sur les réseaux routiers, où le nombre de sauts sur le chemin le plus court peut être plus important, l'avantage du PSB n'est pas aussi prononcé et le SB* peut avoir des performances similaires.

Dans le contexte de réseaux complexes, les caractéristiques structurelles, telles que les sommets de haut degré et la propriété du petit monde, font du PSB un choix favorable, lui permettant de surpasser les autres algorithmes.[17]

7. Conclusion

En conclusion, notre analyse comparative complète des algorithmes Sidetrack Based (SB) et Pessimistic Sidetrack Based (PSB) a dévoilé des modèles et des distinctions intrigants, mettant en lumière leurs performances dans divers paysages de réseau. Notamment, notre exploration des rues sinueuses de Rome, caractérisées par 3 352 nœuds et 8 859 bords, a révélé la propension du SB à générer un nombre d'arbres plus élevé que le PSB. Cette observation souligne les solides capacités d'exploration de SB, ce qui en fait un choix favorable pour naviguer dans des réseaux urbains complexes. Cependant, notre quête d'informations s'est étendue au-delà des sentiers pavés de Rome, nous menant au vaste réseau routier de la région de la baie de San Francisco, englobant 321 270 nœuds et 794 830 bords. Ici, la dynamique de génération des arbres a pris une tournure légèrement différente. Alors que SB continuait de préférer produire plus d’arbres que PSB, la différence entre les deux algorithmes était nettement plus petite, démontrant leur adaptabilité à différentes tailles et complexités de réseau.

En portant notre attention sur l'efficacité temporelle, nous avons noté des variations significatives de performances entre les deux algorithmes en réponse aux caractéristiques du réseau. À Rome, l'algorithme PSB a fait preuve d'une efficacité temporelle supérieure, surpassant le SB pour découvrir rapidement des chemins dans le contexte d'un réseau urbain plus petit et

moins complexe. Cette performance agile était particulièrement avantageuse pour naviguer dans les rues labyrinthiques de la ville historique. Cependant, le paysage a changé lorsque nous nous sommes aventurés dans le réseau vaste et complexe de la région de la baie de San Francisco. Ici, le PSB était aux prises avec des temps de calcul prolongés, luttant pour répondre aux demandes de ce réseau étendu. En revanche, SB a toujours fait preuve d'une approche pragmatique, offrant des temps de calcul raisonnables, même face à des structures de réseau plus vastes et plus complexes.

Essentiellement, notre analyse comparative souligne l'interaction critique entre la sélection des algorithmes et les caractéristiques du réseau. PSB brille dans les réseaux plus petits où la vitesse est primordiale, tandis que SB excelle dans la gestion des demandes de calcul de structures de réseau plus grandes et complexes. Ces informations fournissent des conseils précieux pour le choix des algorithmes dans les applications du monde réel, où la taille et la complexité du réseau sont des considérations cruciales. Alors que nous concluons notre exploration, nous reconnaissons que la quête de solutions de recherche de chemin optimales se poursuit, motivée par l'évolution des besoins de divers scénarios de réseau et par la recherche incessante de l'efficacité et de la précision dans l'optimisation basée sur les graphiques.

Conclusion et Perspectives

Dans le but d'optimiser les solutions au problème des k plus courts chemins simples dans divers types de réseaux, une analyse comparative a été menée sur deux algorithmes importants : l'algorithme basé sur les chemins détournés (SB) et l'algorithme basé sur les chemins détournés pessimistes (PSB). L'objectif de cette étude comparative était de discerner les forces et les faiblesses de chaque algorithme et de déterminer leur adéquation à des scénarios de réseau spécifiques.

Les résultats révèlent que l'algorithme PSB excelle lorsqu'il est confronté à des réseaux complexes caractérisés par des topologies compliquées, diverses interconnexions de nœuds et des poids d'arêtes variables. Dans de tels scénarios, où les chemins peuvent présenter des degrés élevés de complexité et d'imprévisibilité, l'algorithme PSB fait preuve d'une capacité supérieure à énumérer efficacement les k plus courts chemins simples.

Le succès de l'algorithme PSB dans les réseaux complexes peut être attribué à son approche conservatrice, qui donne la priorité à l'énumération des chemins simples qui adhèrent strictement aux contraintes spécifiées. En prenant en compte les pires scénarios et en optimisant la diversité des chemins, l'algorithme PSB est mieux adapté pour naviguer dans les méandres des réseaux complexes et fournir un ensemble complet de solutions viables.

Inversement, l'analyse comparative indique que l'algorithme SB brille lorsqu'il est appliqué à des réseaux routiers et à des scénarios liés au transport. Dans ces contextes, où les chemins sont souvent soumis à des contraintes bien définies et présentent des topologies relativement simples, l'algorithme SB affiche des performances supérieures.

Les stratégies efficaces de l'algorithme SB basées sur les déviations, tout en divergeant des méthodes traditionnelles d'affinement de l'arbre du plus court chemin, font preuve d'une compétence remarquable dans le contexte des réseaux routiers. En identifiant et en explorant stratégiquement les chemins qui s'écartent de l'arbre du plus court chemin, l'algorithme SB accélère le processus d'énumération des chemins tout en enrichissant l'ensemble des solutions potentielles. En conclusion, le choix entre les algorithmes SB et PSB doit être guidé par les caractéristiques et les exigences spécifiques du réseau considéré. Cette analyse comparative souligne l'importance de sélectionner l'algorithme le plus approprié en fonction de la nature du réseau, réaffirmant

l'importance de la diversité algorithmique dans le traitement des problèmes d'optimisation complexes dans le domaine de la théorie des graphes et de l'analyse des réseaux.

Comme perspective, nous envisageons d'approfondir les tests et les validations sur des ensembles de données plus vastes pour consolider sa performance et sa fiabilité. Nous espérons que Sidetrack Based pourra servir de base à d'autres avancées algorithmiques, contribuant ainsi à résoudre des problèmes complexes dans divers secteurs et à répondre aux besoins changeants de la société.

Références

A. Références Bibliographiques

[1] Dijkstra, E. W. (1959). A note on two problems in connection with graphs. Numerische Mathematik, 1(1), 269-271.

[2] Bellman, R. (1958). On a routing problem. Quarterly of Applied Mathematics, 16(1), 87-90.Dijkstra, E. W. (1959). A note on two problems in connection with graphs. Numerische Mathematik, 1(1), 269-271.

[3] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics, 4(2), 100-107.

[4] Floyd, R. W., & Warshall, S. (1962). The problem of the shortest path. Proceedings of the American Mathematical Society, 13(3), 371-375.

[5] Yen, J. Y. (1971). Finding the k shortest loopless paths in a network “Management Science, 17”(11), 712-716.

[6] Kurz, M., & Murzel, P. (2016). Sidetrack Based Algorithm for the k Shortest Path Problem. “Networks, 68”(2), 118-131.

[7] Eppstein, D. (1998). Finding the k Shortest Paths. “SIAM Journal on Computing, 28”(2), 652-673.

[8] Johnson, D. B. (1977). Efficient algorithms for shortest paths in sparse networks. “Journal of the ACM, 24”(1), 1-13.

[9] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms. The MIT Press.

[10] Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. Commentarii academiae scientiarum Petropolitanae, 8, 128-140.

[11] Wasserman, S., & Faust, K. (1994). Social network analysis: Methods and applications. Cambridge University Press.

[12] Laporte, G. (2002). The vehicle routing problem: An overview of exact and approximate algorithms. European journal of operational research, 139(1), 1-25.

[13] Barabási, A. L., & Oltvai, Z. N. (2004). Network biology: understanding the cell's functional organization. Nature reviews genetics, 5(2), 101-113.Barabási

[14] Ford, L. R., & Fulkerson, D. R. (1956). Maximal flow through a network. Canadian journal of mathematics, 8(3), 399-404

[15] Fortunato, S. (2010). Community detection in graphs. Physics reports, 486(3-5), 75-174.

[16] Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 701-710.

[17] Ali Al Zoobi. Practical computation of simple paths with length and diversity constraints in complex and multimodal networks. Combinatorics [math.CO]. Université Côte d'Azur, 2021. English. ⟨NNT : 2021COAZ4091⟩. ⟨tel-03574440⟩

[18] Cormen, Thomas H., et al. Introduction to Algorithms. MIT press Cambridge, 2009.

[19] Klavžar, Sandi, and Drago Košmrlj. "On two problems in connection with graphs." Discrete Applied Mathematics 12.3 (1986): 275-279.

[20] Magnanti, Thomas L., and Robert T. Wong. "Network design and transportation planning: Models and algorithms." Transportation Science 28.2 (1994): 101-113.

[21] Kurz, A., and T. Murzel. "Finding k shortest paths using k-bridges." Networks 46.2 (2005): 112-126.

[22] Daniele Frigioni, Alberto Marchetti-Spaccamela, and Umberto Nanni. Fully dynamic algorithms for maintaining shortest paths trees. Journal of Algorithms, 34(2) :251–281, 2000.

[23] Michael L. Fredman, Robert Sedgewick, Daniel D. Sleator, and Robert E. Tarjan. The pairing heap : A new form of self-adjusting heap. Algorithmica, 1(1) :111–129, 1986

[24] Rose Oughtred, Chris Stark, Bobby-Joe Breitkreutz, Jennifer Rust, Lorrie Boucher, Christie Chang, Nadine Kolas, Lara O'Donnell, Genie Leung, Rochelle McAdam, et al. The biogrid interaction database : 2019 update. Nucleic acids research, 47(D1) :D529–D541, 2019.

[25] Lukasz Salwinski, Christopher S. Miller, Adam J. Smith, Frank K. Pettit, James U. Bowie, and David Eisenberg. The database of interacting proteins : 2004 update. Nucleic acids research, 32(suppl_1) :D449–D451, 2004.

[26] Jure Leskovec and Andrej Krevl. SNAP Datasets : Stanford large network dataset collection. http://snap.stanford.edu/data , June 2014.

Résumé

Cette thèse présente une étude approfondie de la théorie des graphes, soulignant sa grande importance dans les applications du monde réel. La section d'introduction donne un aperçu de la théorie des graphes et de sa pertinence dans divers domaines, notamment les réseaux de transport et les connexions sociales. La théorie des graphes offre un cadre polyvalent pour comprendre les relations complexes et optimiser les systèmes en réseau.

L'évolution historique du problème du chemin le plus court est ensuite explorée, retraçant son développement depuis les premières formulations jusqu'aux solutions algorithmiques contemporaines. Le problème "plus courts k simples chemins" est identifié comme une extension essentielle, avec des applications couvrant les transports, la communication et la biologie.

L'objectif principal de cette thèse est l'algorithme Sidetrack Based (SB), une solution puissante au problème des k chemins simples les plus courts. Les principes, les forces et les limites de l'algorithme SB sont analysés, offrant une compréhension approfondie de son fonctionnement. De plus, les contributions originales de l'auteur visant à améliorer la praticité et l'efficacité de l'algorithme sont mises en évidence.

Pour enrichir notre compréhension, des études comparatives des performances des algorithmes SB et PSB dans des réseaux distincts sont menées. L'analyse inclut la génération d'arbres et l'efficacité des calculs, mettant en lumière l'adaptabilité de l'algorithme à divers scénarios. Les réseaux étudiés vont des rues historiques de Rome au réseau complexe de la région de la baie de San Francisco.

En conclusion, cette thèse sert de guide complet de l'algorithme SB et contribue à l'évolution continue des techniques d'optimisation de réseau au sein de la théorie des graphes. Il souligne le rôle essentiel de la théorie des graphes dans les applications modernes et montre comment des algorithmes innovants tels que SB peuvent remodeler notre capacité à analyser et optimiser des réseaux complexes.

Mots clés: Algorithme SB, Sidetrack Based, plus courts k simples chemins.

Abstract

This thesis presents a thorough investigation of graph theory, emphasizing its broad significance in real-world applications. The introductory section provides an overview of graph theory and its relevance in various domains, including transportation networks and social connections. Graph theory offers a versatile framework for comprehending complex relationships and optimizing networked systems.

The historical evolution of the shortest path problem is then explored, tracing its development from early formulations to contemporary algorithmic solutions. The k Simple Shortest Paths Problem is identified as a pivotal extension, with applications spanning transportation, communication, and biology.

The primary focus of this thesis is on the Sidetrack Based (SB) algorithm, a powerful solution to the k Simple Shortest Paths Problem. The SB algorithm's principles, strengths, and limitations are dissected, providing a deep understanding of its workings. Additionally, the author's original contributions aimed at enhancing the algorithm's practicality and efficiency are highlighted.

To enrich our understanding, comparative studies of the SB and PSB algorithms' performance in distinct networks are conducted. The analysis includes tree generation and computational efficiency, shedding

light on the algorithm's adaptability across diverse scenarios. The networks studied range from the historic streets of Rome to the intricate web of the San Francisco Bay Area.

In conclusion, this thesis serves as a comprehensive guide to the SB algorithm and contributes to the ongoing evolution of network optimization techniques within graph theory. It underscores the critical role of graph theory in modern applications and showcases how innovative algorithms like SB can reshape our ability to analyze and optimize complex networks.

Keywords : Algorithm SB, Sidetrack Based, plus courts k simples chemins.

ملخّص

تقدم هذه الأطروحة تحقيقا شاملا لنظرية الرسم البياني، مع التركيز على أهميتها الواسعة في تطبيقات العالم الحقيقي. يقدم القسم التمهيدي لمحة عامة عن نظرية الرسم البياني وأهميتها في مختلف المجالات، بما في ذلك شبكات النقل والاتصالات الاجتماعية. تقدم نظرية الرسم البياني إطارًا متعدد الاستخدامات لفهم العلاقات المعقدة وتحسين الأنظمة المتصلة بالشبكة.

يتم بعد ذلك استكشاف التطور التاريخي لمشكلة أقصر مسار، وتتبع تطورها من الصياغات المبكرة إلى الحلول الخوارزمية المعاصرة. تم تحديد مشكلة أقصر المسارات k البسيطة باعتبارها امتدادًا محوريًا، مع تطبيقات تشمل النقل والاتصالات وعلم الأحياء.

التركيز الأساسي لهذه الأطروحة هو على خوارزمية (SB (Sidetrack Based ، وهي حل قوي لمشكلة أقصر المسارات البسيطة k. يتم تشريح مبادئ خوارزمية SB ونقاط القوة والقيود، مما يوفر فهمًا عميقًا لأعمالها. بالإضافة إلى ذلك، يتم تسليط الضوء على مساهمات المؤلف الأصلية التي تهدف إلى تعزيز التطبيق العملي للخوارزمية وكفاءتها.

لإثراء فهمنا، يتم إجراء دراسات مقارنة لأداء خوارزميات SB و PSB في شبكات متميزة. يتضمن التحليل إنشاء الأشجار والكفاءة الحسابية، وتسليط الضوء على قدرة الخوارزمية على التكيف عبر سيناريوهات متنوعة. تتراوح الشبكات التي تمت دراستها من شوارع روما التاريخية إلى الشبكة المعقدة لمنطقة خليج سان فرانسيسكو.

في الختام، هذه الأطروحة بمثابة دليل شامل لخوارزمية SB وتساهم في التطور المستمر لتقنيات تحسين الشبكة ضمن نظرية الرسم البياني. إنه يؤكد على الدور الحاسم لنظرية الرسم البياني في التطبيقات الحديثة ويعرض كيف يمكن للخوارزميات المبتكرة مثل SB إعادة تشكيل قدرتنا على تحليل الشبكات المعقدة وتحسينها.

الكلمات المفتاحية: خوارزمية SB، تعتمد على Sidetrack، و أقصر المسارات البسيطة.