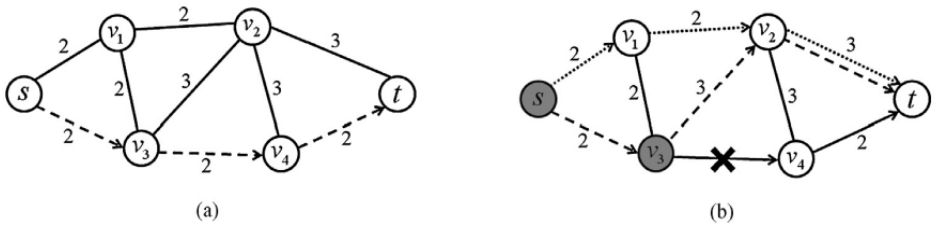
Projet RP

14 Mars, 2025

---

1 Introduction

La motivation de ce projet vient de la nécessité de planifier des itinéraires en ligne dans les réseaux routiers. Bien que les systèmes de navigation actuels utilisent des informations sur les distances routières et les limitations de vitesse pour trouver les itinéraires les plus rapides, les conditions de trafic sont souvent imprévisibles et varient énormément. Par conséquent, les stratégies de prise de décision en ligne jouent un rôle important dans la résolution des problèmes de congestion du trafic. Le problème du voyageur canadien (CTP) consiste à trouver le chemin le plus court entre une source et une destination, compte tenu d'informations incomplètes acquises de manière dynamique. Considérons un réseau routier $G = (V, E)$ représenté par un ensemble V de sommets reliés par des routes, où chaque route $e \in E$ est associée à une durée (coût positif) lorsqu'un voyageur la traverse. Le voyageur connaît à l'avance le réseau G. Cependant, certaines routes peuvent être bloquées par la neige ou par des accidents, et un blocage ne devient connu que lorsque le voyageur atteint un sommet adjacent à la route bloquée. La figure 1 illustre le problème CTP : Le voyageur souhaite aller du sommet s au sommet t. Initialement, comme le réseau routier est connu, il calcule une plus courte chaîne de s à t : $s - v_3 - v_4 - t$ de durée six. Ensuite, il commence en suivant l'itinéraire prévu, mais une fois arrivé au sommet v_3, il se rend compte qu'un accident s'est produit sur la route (v_3, v_4). Ainsi le voyageur doit ajuster son itinéraire en prenant en compte du blocage de (v_3, v_4). Finalement, le voyageur effectue l'itinéraire $s - v_3 - v_2 - t$ pour une durée de huit. Cet exemple montre comment les accidents ou des évènements imprévisibles affectent les itinéraires planifiés. Notons aussi qu'a posteriori l'itinéraire optimal pour cet exemple en tenant compte du blocage de la route (v_3, v_4) est l'itinéraire $s - v_1 - v_2 - t$ de durée sept.



Pour évaluer la qualité d'une solution on utilise le *rapport de compétitivité*. Le rapport de compétitivité pour un algorithme $\mathcal{A}$ représente le pire rapport entre la durée de l'itinéraire emprunté par le voyageur en suivant $\mathcal{A}$ et la durée de la plus courte chaîne a posteriori, où tous les blocages de routes sont connus avant le début de l'itinéraire du voyageur.

Nous allons nous intéresser à une généralisation du CTP, appelée le *problème du voyageur canadien couvrant* (CCTP) : Étant donné un graphe complet $G = (V, E)$ avec un point d'origine $s \in V$, un voyageur connaît toute la structure de G et souhaite commencer en s, visiter tous les autres sommets de V et revenir en s le plus rapidement possible. Cependant, le voyageur découvre en ligne que certaines arêtes sont bloquées. Précisément, le voyageur parcourt le graphe avec la

même incertitude que le CTP, c'est-à-dire qu'il ne découvre une arête bloquée que lorsqu'il atteint un sommet adjacent à l'arête bloquée. En outre, nous faisons deux hypothèses : l'une est que l'état d'une route ne change pas après que le voyageur ait appris qu'elle était bloquée ou pas. Ainsi, une fois qu'une route est bloquée par un accident, le voyageur ne peut jamais passer par cette route. L'autre hypothèse est que le graphe complet G reste connecté même si les arêtes bloquées sont éliminées. Notons que le problème CCTP que nous allons considérer est une variante du problème de voyageur de commerce (TSP) étudié en cours.

Ce projet considère le k-CCTP avec au plus k blocages $k < n-1$ où n est le nombre de sommets du graphe V. Nous supposons également que les durées (coûts) sur les routes satisfont l'inégalité triangulaire.

2 Objectif du projet

L'objectif de ce projet est d'implanter deux algorithmes dont l'idée principale sera présentée par la suite et les comparer en mettant en place un jeu de tests adéquat.

Nous allons maintenant présenter les idées des deux algorithmes que vous aurez à implanter.

3 L'idée de l'algorithme CR (routage cyclique)

L'algorithme de routage cyclique (CR) repose sur le raisonnement suivant : l'itinéraire complet peut être décomposé en plusieurs tours ; à chaque tour, le voyageur tente de visiter le plus grand nombre possible de sommets dans un graphe complet G donné, en suivant l'ordre de visite de la tournée dérivée de l'algorithme de Christophides pour le problème du voyageur de commerce (vu en cours). Soit $P : s = v_1 - v_2 - \cdots - v_n - s$ le tour retourné par l'algorithme de Christophides sur le graphe G. Afin de présenter l'idée de l'algorithme nous avons besoin d'introduire les notations suivantes : soit V_m l'ensemble de sommets non visités au début de l'itération m. Au début, $V_0 = V$ et $V_1 = V \setminus \{s\}$.

A la première itération ($m = 1$), l'algorithme CR tente de visiter les sommets non encore visités en suivant P dans le sens des aiguilles d'une montre. A chaque fois qu'une route (v_i, v_{i+1}) sur P se trouve bloquée, le voyageur tente un raccourci de v_i vers le sommet v_{i+2}. Si la route (v_i, v_{i+2}) est bloquée elle aussi, il tente un raccourci vers le sommet v_{i+3} et continue ainsi jusqu'au sommet d'indice maximum $k \leq n$ qui peut être atteint de cette manière.

Dans la figure (a), nous présentons un exemple pour consolider les notations. Soit $P : v_1 - v_2 - \cdots - v_{16} - v_1$ le tour obtenu par l'algorithme de Christophides. Le voyageur commence par $s = v_1$ et il visite v_2 et arrive en v_3 en suivant l'ordre du tour P puisqu'il n'y a pas de blocage sur les routes (v_1, v_2) et (v_2, v_3). Une fois arrivé à v_3, il découvre que les routes (v_3, v_4) et (v_3, v_5) sont bloquées, mais que la route (v_3, v_6) est disponible. Il continue ainsi son itinéraire en empruntant la route (v_3, v_6), ensuite (v_6, v_7) et ainsi de suite. Sur la figure (a), les lignes continues correspondent au tour de Christophides et l'orientation des arcs correspond à un parcours de sommets dans le sens des aiguilles d'une montre. Les croix sur certains arcs indiquent que les routes en question sont bloquées et les routes en pointillé indiquent les raccourcis effectués par le voyageur après la découverte de différents blocages. L'ensemble des routes bloquées après la première itération est $E'_1 = \{(v_3, v_4), (v_3, v_5), (v_7, v_8), (v_9, v_{10}, (v_{12}, v_{13}), (v_{12}, v_{14}))\}$. L'itinéraire obtenu par la procédure de raccourci est $P_1^{cr} = v_1 - v_2 - v_3 - v_6 - v_7 - v_9 - v_{11} - v_{12} - v_{15} - v_{16}$. On note $v_{1,0} = v_1$ et

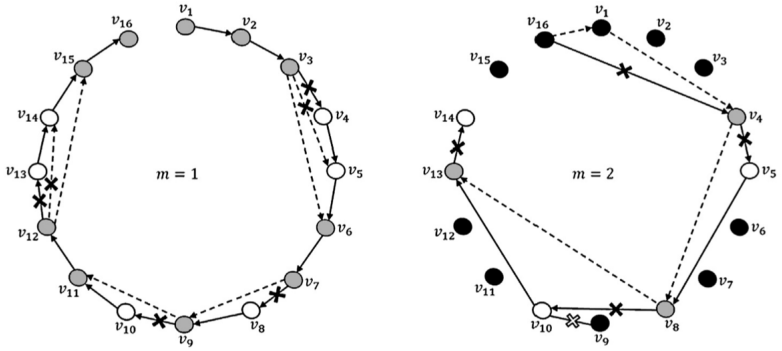
$v_{1,|V_1|} = v_{16}$. On note aussi que le coût du tour raccourci P_1^{cr} est au plus égal au coût de la tournée P en raison de l'inégalité triangulaire.

Après cette première itération, l'ensemble des sommets V est partitionné en deux sous-ensembles : l'ensemble V_2 contenant les sommets non-visités (coloriés en blanc) et l'ensemble $V \setminus V_2$ contenant les sommets visités (coloriés en gris) lors de cette première itération. Ainsi, $V_2 = \{v_4, v_5, v_8, v_{10}, v_{13}, v_{14}\}$ est l'ensemble de sommets non-visités. A partir de V_2, comme $v_{2,0} = v_{1,|V_1|} = v_{16}$, on définit le chemin $P_2 = v_{16} - v_4 - v_5 - v_8 - v_{10} - v_{13} - v_{14}$ qui commence avec le dernier sommet de P_1^{cr} qui ne fait pas parti de V_2 et qui propose d'emprunter les routes directes entre les sommets consécutifs de V_2 en suivant P dans le même sens que lors de l'itération précédente. Dans la figure (b), le chemin P_2 correspond aux arcs pleins et les sommets déjà visités sont noirs. Le voyageur tente de se déplacer le long des arcs pleins et, si nécessaire, emprunte des voies de raccourci en utilisant les sommets internes déjà visités (c'est-à-dire les lignes en pointillé). L'itinéraire obtenu lors de cette deuxième itération est $P_2^{cr} = v_{16} - v_1 - v_4 - v_8 - v_{13}$. Notons que lorsque le voyageur passe de v_8 à v_{10}, la procédure de raccourci n'utilise pas le sommet intermédiaire déjà visité v_9 car l'arête bloquée (v_9, v_{10}) a été révélée à l'itération précédente. Notons également que grâce à l'inégalité triangulaire le coût de P_2^{cr} est inférieur ou égal à celui du tour initial P. L'ensemble des routes bloquées découvertes lors de la deuxième itération est $E_2' = \{(v_{16}, v_4), (v_4, v_5), (v_8, v_{10}), (v_{13}, v_{14})\}$.

Maintenant, l'ensemble des sommets non visités au début de la troisième itération est $V_3 = \{v_5, v_{10}, v_{14}\}$. Comme $v_{3,0} = v_{13} \neq v_{2,|V_2|} = v_{14}$, le voyageur se déplace dans le sens opposé des aiguilles d'une montre. Donc $P_3 = v_{13} - v_{10} - v_5 - v_{14}$. Ceci est illustré dans la figure (c). Comme précédemment, à chaque fois qu'une route directe de P_3 est bloquée, le voyageur essaye de déterminer une route passant par un sommet intermédiaire qui est déjà visité. Dans ce cas on obtient $P_3^{cr} = v_{13} - v_{11} - v_{10} - v_7 - v_5 - v_2 - v_{14}$. Comme tous les sommets sont déjà visités, il suffit de passer de v_{14} à $s = v_1$ ce qui peut se faire en utilisant encore une fois un sommet intermédiaire car la route (v_{14}, v_1) est bloquée, ce qui donne $P_4^{cr} = v_{14} - v_{15} - v_1$ (voir la figure (d)).

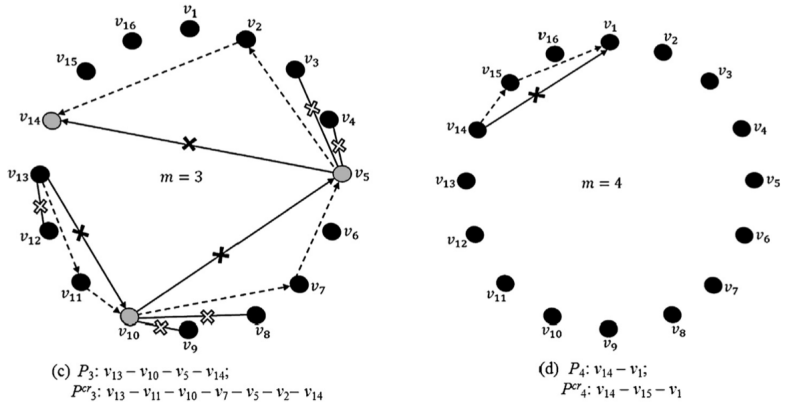
Une chose importante lors de l'exécution de l'algorithme est le sens dans la procédure de raccourci lors de l'itération $m > 1$. Si $v_{m,0} = v_{m-1,|V_{m-1}|}$ alors on suit l'ordre de l'itération $(m-1)$ et dans le cas où $V_{m+1} = V_m$, alors on effectue les raccourcis dans le sens opposé. Enfin, si $v_{m,0} \neq v_{m-1,|V_{m-1}|}$, alors on suit l'ordre opposé de celui suivi dans l'itération $(m-1)$.

L'itinéraire global du voyageur sera donc la concaténation des itinéraires P_i^{cr}, pour $i = 1, 2, 3, 4$.



(a) P_1: $v_1 - v_2 - v_3 - v_4 - v_5 - v_6 - v_7 - v_8 - v_9 - v_{10} - v_{11} - v_{12} - v_{13} - v_{14} - v_{15} - v_{16}$;
P^{cr}_1: $v_1 - v_2 - v_3 - v_6 - v_7 - v_9 - v_{11} - v_{12} - v_{15} - v_{16}$

(b) P_2: $v_{16} - v_4 - v_5 - v_8 - v_{10} - v_{13} - v_{14}$;
P^{cr}_2: $v_{16} - v_1 - v_4 - v_8 - v_{13}$



(c) P_3: $v_{13} - v_{10} - v_5 - v_{14}$;
$P^{cr}{}_3$: $v_{13} - v_{11} - v_{10} - v_7 - v_5 - v_2 - v_{14}$

(d) P_4: $v_{14} - v_1$;
$P^{cr}{}_4$: $v_{14} - v_{15} - v_1$

4 L'algorithme CNN

Pour présenter ce deuxième algorithme, nous devons d'abord définir un nouveau problème, le *problème d'exploration de graphes en ligne*. Le problème peut être formalisé comme suit. Soit $G = (V, E)$ un graphe pondéré, non orienté, avec $|V| = n$ sommets. Le voyageur commence à un sommet $s \in V$ et doit visiter tous les sommets du graphe et revenir à s. Notons que le voyageur est en train d'explorer tous les sommets du graphe. Le voyageur peut visiter un sommet plus d'une fois. A chaque étape, l'agent est situé à un sommet u et peut choisir de se déplacer vers n'importe lequel des sommets voisins de u. L'agent encourt un coût égal au coût de la route traversée. En arrivant à un sommet v, l'agent apprend toutes les arêtes incidentes à v et leurs coûts. L'objectif est de faire cet itinéraire en minimisant le coût total.

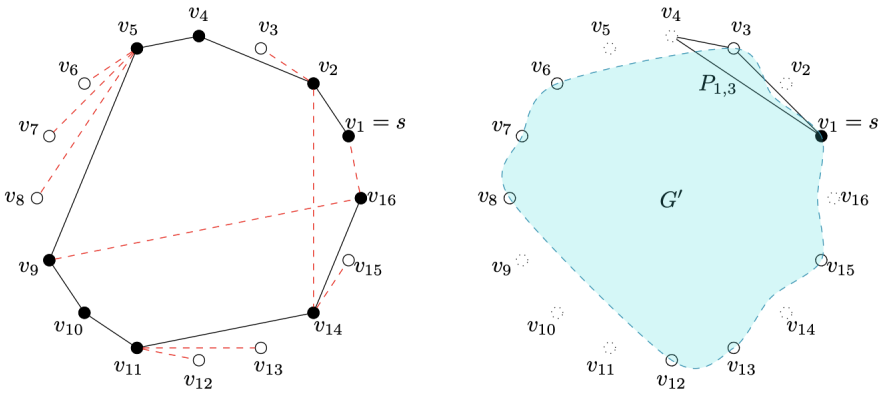
Un algorithme simple et rapide qui résout le problème est l'*algorithme du plus proche voisin* (NN). L'algorithme sélectionne un sommet inexploré qui est le moins cher à atteindre à partir du sommet actuel et le visite, répétant ce processus jusqu'à ce que tous les sommets soient visités.

L'idée générale de l'algorithme CNN est la suivante : tout d'abord, nous allons calculer un tour en utilisant l'algorithme de Christophides. Ensuite, nous essayons de suivre ce tour, en utilisant des raccourcis (sautant des sommets) lorsque des routes se révèlent bloquées comme nous avons fait dans l'algorithme CR. Après cela, nous retournons au sommet de départ. De cette façon, nous visitons au moins $n - k$ sommets de G. Comme ils ne doivent pas être visités à nouveau, nous pouvons utiliser les informations recueillies pour réduire le nombre de sommets du graphe sur lequel nous devons nous concentrer et appliquer l'algorithme d'exploration de graphe.

Plus précisément l'algorithme CNN suit les étapes suivantes : tout d'abord, l'algorithme exécute l'algorithme de Christophides sur le graphe d'entrée G afin de calculer un tour P. Par souci de simplicité, nous réétiquetons les sommets en fonction de la tournée, c'est-à-dire que nous supposons que la tournée P a l'ordre $s = v_1 - v_2 - \ldots - v_n - v_1$. Si une arête (v_i, v_j) est bloquée, le voyageur essaie d'aller au sommet suivant dans l'ordre déterminé par P, c'est-à-dire v_{j+1}, ou v_1 pour $j = n$. Si le voyageur atteint le sommet s, alors cette étape se termine. Si s n'est pas atteignable directement à cause d'une arête bloquée, le voyageur retourne à s en revenant sur ses pas.

Le voyageur apprend l'état de toutes les arêtes (bloquées ou non-bloquées) qui sont adjacentes aux sommets visités au cours de l'étape précédente. Au cours de la procédure, toutes les arêtes dont on découvre qu'elles sont bloquées sont rassemblées dans l'ensemble E_b. Ainsi, le voyageur connaît l'ensemble du graphe (avec tous les blocages) à l'exception du sous-graphe induit formé par

les sommets non visités U qui est un graphe complet. Ensuite, le voyageur, se trouvant en s, doit visiter les sommets en U. Puisque les vraies arêtes du graphe, sauf celles du sous-graphe induit formé par les sommets de U, sont connues, il suffit de ne considérer que les sommets de l'ensemble $U_s = U \cup \{s\}$. Bien que les sommets de $V \setminus U_s$ eux-mêmes ne sont pas nécessaires, un plus court chemin entre deux sommets $x, y \in U_s$ peut inclure des sommets de l'ensemble $V \setminus U_s$ comme sommets intermédiaires. Cela peut se produire lorsque des routes inconnues entre des sommets non visités sont bloquées.



Pour chaque paire de sommets $x, y \in U_s$, on crée une nouvelle arête $P_{x,y}$ représentant le plus court chemin entre x et y, de telle sorte que le chemin ne soit constitué que d'arêtes dont on sait qu'elles ne sont pas bloquées, c'est-à-dire d'arêtes dont au moins un sommet a déjà été visité auparavant - si un tel plus court chemin existe. Ainsi, la procédure crée un multigraphe G' qui se compose de l'ensemble de sommets U_s, des arêtes initiales qui relient ces sommets et du "plus court chemin" représenté par une nouvelle arête dans G'.

Plus formellement, soit E' le sous-ensemble des arêtes (x, y) avec $x, y \in U_s$. Le graphe G' est initialement défini comme $G' = (U_s, E')$. Nous définissons également un graphe auxiliaire $H = (V, E \setminus E')$, c'est-à-dire H comprend tous les sommets du graphe initial G et de toutes les arêtes dont l'état est connu, puisque chaque arête a au moins un sommet qui a déjà été visité. Soit $U_s = \{v'_1, v'_2, \ldots, v'_{|Us|}\}$. On détermine un plus court chemin $P_{i,j}$ de v'_i à v'_j dans H, et soit $c_{i,j}$ le coût total de $P_{i,j}$. On ajoute une arête (v'_i, v'_j) avec un coût $c_{i,j}$ à G'.

Enfin, l'algorithme exécute NN sur G' et visite les sommets restants. Chaque fois que le voyageur visite un sommet, il apprend toutes les routes incidentes. Cela inclut les routes du "chemin le plus court" nouvellement ajoutées, dont nous savons qu'elles sont réalisables. Si le voyageur utilise une telle route de "chemin le plus court", nous la "développons" dans la tournée finale calculée dans le graphe original, ce qui signifie que nous utilisons le chemin réel qui correspond à cette arête.

A la fin, on concatène les deux chemins déterminés, c'est-à-dire on visite les sommets selon le chemin déterminé après la phase de raccourci, puis selon le chemin déterminé par l'algorithme NN dans G'.

5 Modalités de travail et délivrables

Le travail est à effectuer en binôme constitué de deux personnes. Les projets seront déposés au plus tard le 27 avril 2025 à minuit sur le site moodle de RP. Votre livraison sera constituée d'une

archive zip (pas de .tar ou .targz etc) nommée RP-nom1-nom2.zip qui comportera les sources du programme, un fichier README détaillant comment exécuter le programme, et un rapport rédigé (un fichier au format pdf nommé RP-nom1-nom2.pdf) qui présentera le travail effectué. Le plan du rapport suivra le plan du sujet. Il est fortement recommandé de rédiger son rapport en LaTeX. Les projets rendus feront l'objet d'une brève soutenance sur machine en salle tme lors de la dernière séance.

6 Bibliographie

Les descriptions des algorithmes et les schémas sont tirés de deux articles suivants dans lesquels vous aurez également des pseudocodes qui devraient vous permettre de mieux comprendre.

1. Chung-Shou Liao and Yamming Huang. The covering canadian traveller problem. Theor. Comp. Sci., 530:80–88, 2014.
2. Niklas Hahn, Michalis Xefteris, The Covering Canadian Traveller Problem Revisited, MFCS 2023.