CROS Julien
LE HENAFF Tanguy
NEGRE Clement
ROMANO Nicolas

Rapport de projet

Apprentissage Par Projet
Eléments de Recherche Opérationnelle

Responsable:
DUDIN Bashar

EPITA Paris
ING1

Synthèse ERO

Nous nous sommes d'abord penchés sur le sens de notre mission :

- Proposer un trajet minimal du drone lors du survol du réseau routier de Montréal
- Proposer un trajet minimal d'un appareil de déblaiement dans une partie de Montréal
- Proposer un modèle de coût pour les opérations de déblaiement dans Montréal

Caractère eulérien du problème

Identification du problème

Nous comprenions donc qu'il fallait parcourir chaque route de Montréal et que parcourir une seule fois chaque route était suffisant (il faut distinguer cela des routes à double sens qui ne sont pas qu'une seule route et qu'il faut donc parcourir dans les deux sens).

Le trajet du drone qui n'est pas impacté par le code de la route serait donc associé à l'étude d'un trajet optimal dans un graphe non-orienté. Le trajet d'une déneigeuse dans une zone de Montréal serait lui associé à l'étude d'un graphe orienté. Finalement, nous comprenions qu'il était question de trouver des chemins eulériens dans nos graphes.

Mise en place de solutions et problématiques

Le premier problème lié à la pratique était donc le suivant : les graphes associés à la ville de Montréal ou à un sous graphe de Montréal n'étaient pas forcément eulériens. Nous ne pourrions donc pas chercher de chemin eulérien, ce qui aurait été une solution optimale pour le parcours du drone ainsi que pour le parcours d'un appareil de déblaiement. Il fallait donc faire en sorte que les graphes le soient (nous ne pouvions évidemment pas modifier le réseau routier de Montréal).

- **Cas du drone**

Par chance OSMNX permet d'"eulerialiser" un graphe non-orienté facilement et de même pour trouver un chemin eulérien dans ce graphe alors eulérien. Le problème étant que ces algorithmes prennent beaucoup de temps mais nous avons estimé que cela n'est pas gênant dans le cadre de la solution étant donné que le trajet du drone n'est à déterminer qu'une seule fois : celui-ci sera toujours le même (hormis modification du réseau routier de la ville).

- **Cas d'un appareil de déblaiement**

Si OSMnx permettait de rendre un graphe non orienté eulérien très facilement, ce n'était pas le cas pour un graphe orienté. Nous devions le faire nous-même et c'est ce qui fut la principale difficulté de cette partie.

Pour qu'un graphe orienté soit eulérien, les deux seules conditions sont : qu'il soit fortement connexe et que chacun des sommets aient autant d'arêtes entrantes que sortantes. Grâce à OSMnx il est aisé d'obtenir un graphe fortement connexe, toutefois ils ne

sont évidemment pas eulériens. Pour rectifier cela nous commençons par évaluer toutes les entrées et sorties de chaque sommet pour trouver ceux qui ne respectent pas la seconde règle précédemment énoncée. On trouve ainsi nos futures origines et arrivées pour de nouveaux chemins. Ensuite, on prend chacun des nœuds “origine” et on trouve le sommet d’arrivée le plus proche grâce à l'algorithme de plus court chemin de Dijkstra. Il est inutile de calculer la distance de tous les autres nœuds, il suffit simplement de s'arrêter au premier nœud “arrivée” dont on est certain de la distance. Pour finir, nous dupliquons toutes les arêtes qu’il a fallu parcourir pour atteindre ce dernier. Nous appliquons ensuite l’algorithme trouvant un circuit eulérien de Hierholzer. Ce circuit garantit donc de passer au moins une fois par toutes les routes dans chacun des sens, et il ne passe que par des routes existantes sur le graphe d’entrée.

- **Adaptation aux contraintes du client**

Pour adapter notre solution au problème donné, nous avons décidé de proposer un découpage du parcours de la ville divisé en un nombre donné de déneigeuses. Pour cela nous prenons le parcours qu'aurait fait une seule déneigeuse et nous recréons plusieurs parcours à partir de celui-ci. Nous calculons dans un premier temps, la somme de tous les poids du graph qui donne la distance totale à parcourir. Divisé par le nombre de déneigeuses, nous avons la distance que doit parcourir une seule déneigeuse. Elles vont donc se voir attribuer les morceaux du parcours pour une seule déneigeuse successivement correspondant au bon nombre de kilomètres. Nous affichons ensuite le parcours de chaque déneigeuse les uns après les autres. Le parcours affiché est celui à déneiger. Parfois une déneigeuse passe par un segment qu’elle n’a pas à déneigé et donc celui-ci ne se colore pas, ce qui donne l’impression que la déneigeuse “saute” certains segments du graphe. Celui-ci aura en fait été déneigé par une autre machine.

Améliorations possibles :

- **Cas du drone**

Comme cela a été mentionné précédemment, notre solution pour le calcul du trajet du drone n’est pas très rapide. Nous aurions pu améliorer cela, principalement en utilisant les algorithmes que nous avons implémentés nous même. Néanmoins l’utilisation des solutions qu’utilise OSMnx, NetworkX simplifie grandement la manipulation des graphes.

- **Cas d’un appareil de déblaiement**

Nous aurions pu améliorer notre ajout de nouvelles arêtes pour que ce soit mieux adapté aux graphes NetworkX et ainsi permettre d’afficher ou non les routes sur lesquelles les déneigeuses doivent passer mais sans déblayer.

- **Adaptation aux contraintes du client**

C’est dans cette partie qu’il nous manque le plus d’améliorations. Nous aurions pu mettre en place une estimation des coûts aussi bien en carburant qu’en main d'œuvre. De manière générale, la solution manque d’une interface de qualité, ce qui pourrait nuire à l’image de l’entreprise que nous représentons.