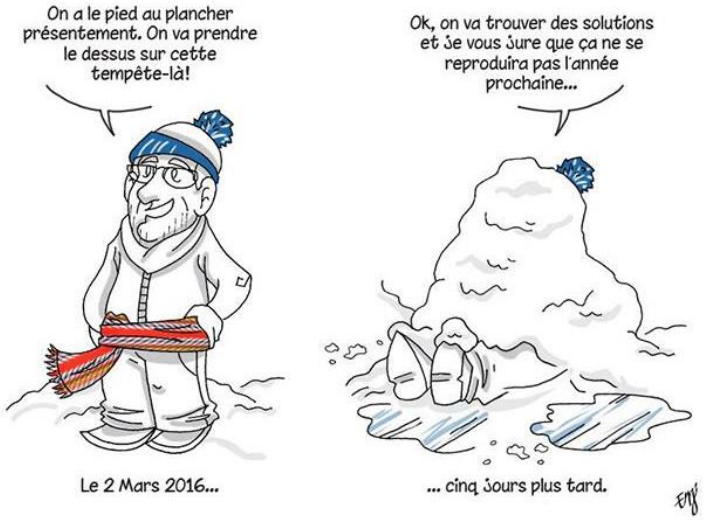
Maessarath Akadiri
Basma Bouharicha
Bastien Duplaix
Aurore Duvernoy
Célia Restes

Eléments de recherche opérationnelle :

Optimisation hivernale

B. Dudin et P. Siarry

Chaque année, la ville de Montréal fait face à des épisodes neigeux. Pour permettre une continuité de l'activité économique, des équipes de déneigement interviennent pour évacuer les chutes de neiges des réseaux routiers et pédestres. Cependant, ces opérations représentent des sommes faramineuses sur le budget de la ville.



L'ancien maire du Québec et sa gestion du déneigement
(Illustration de Marie-Josée Bourgault)

La municipalité nous confie alors la charge d'étudier le moyen de minimiser le trajet des appareils de déblaiement sur le réseau routier de sorte à minimiser le coût des opérations de déblaiement. Notre mission consistera à :

- Déterminer le trajet minimal du drone : faire une analyse aérienne des niveaux neigeux dans le but de limiter les opérations de déblaiement aux routes les plus enneigées ;
- Déterminer le trajet minimal d'une déneigeuse à partir de la carte résultante de l'analyse aérienne ;
- Proposer un modèle de coût pour les opérations de déblaiement sur l'ensemble de la ville en fonction du nombre d'engins disponible.

Pour effectuer au mieux la mission, notre démarche sera itérative : nous procédons par étapes et fixons des objectifs ainsi que des métriques pour chacune d'elles.

1. Recherche du trajet minimal du drone

En partant du principe que le drone ne dépend ni du sens de circulation ni du trafic pour cartographier l'état des routes et trottoirs, nous avons considéré pour cette itération un graphe non orienté. Dans notre analyse, nous nous limitons à un seul drone pour toute la ville.

Périmètre étudié : la ville de Montréal
Métriques à prendre en compte :

- kilométrage effectué par un seul drone
- temps de parcours
- kilométrage total

À l'issue de cette itération, nous saurons s'il est nécessaire d'augmenter le nombre de drones en fonction du temps parcouru. En effet, si ce temps est trop conséquent pour un seul drone, nous pourrons envisager d'en envoyer plusieurs depuis différents points de départ, tout cela en évaluant et ajustant au mieux le coût résultant.

Démarche algorithmique

Avant d'effectuer une analyse d'image aérienne, nous étudions d'abord le trajet minimal du drone dans un graphe pondéré de petite taille dont les nœuds modélisent les intersections, les arêtes, les routes et le poids, les distances ou kilométrages entre les intersections. En plus d'être non orienté, le graphe étudié est fortement connexe : à partir de tout point il existe un itinéraire pour atteindre un autre par les routes.

Notre problème est semblable à celui du voyageur de commerce (*Travelling Salesman Problem*) : cela consiste à trouver le meilleur trajet possible en termes de coût et de temps de sorte que le drone passe exactement une fois par un ensemble donné de destinations intermédiaires, puis revienne à son point de départ à la fin. Ce problème ne s'applique seulement que lorsque les poids vérifient l'inégalité triangulaire mais puisque les distances vérifient cette propriété et qu'elles sont modélisées par les poids, nous pouvons confirmer que cette propriété est bien respectée.

Cependant, il existe une limitation : ce problème s'applique uniquement sur des graphes complets, or dans une ville, les intersections ne sont pas systématiquement liées à toutes les autres.

1. Algorithme de Prim : construire un arbre couvrant de poids minimum

A partir d'une matrice d'adjacence, nous utiliserons l'algorithme de Prim pour construire un arbre couvrant de poids minimal (ACM), un arbre qui connecte tous les nœuds ensemble et dont la somme des poids des arêtes est minimale. De cette façon, nous pouvons construire un réseau routier en minimisant un coût représenté par le kilométrage des routes.

2. Algorithme de recherche en profondeur (*Depth-first search*) : trouver le plus court cycle hamiltonien

Cette itération nous permet de faire le tour de l'ACM à partir d'un nœud donné. Grâce à une recherche en profondeur (DFS) préfixe, nous pouvons déterminer le chemin du drone, dont le poids total ne sera jamais deux fois plus grand que le poids total des arêtes dans l'ACM. Or, ce n'est pas exactement ce que nous voulons puisque faire le tour de l'ACM peut impliquer de passer plusieurs fois sur un même nœud. Pour résoudre cette limitation, nous ne prenons pas compte des nœuds déjà visités lors du DFS, de sorte à obtenir le plus court cycle hamiltonien, c'est-à-dire un cycle qui passe exactement une fois par chaque nœud du graphe.

3. Algorithme de Dijkstra : résoudre la contrainte du graphe complet posée par le voyageur de commerce

Résoudre le problème du voyageur de commerce ne suffit pas. En effet, ce problème part de l'hypothèse que le graphe est complet alors que dans le contexte de la ville de Montréal, cette hypothèse n'est pas vérifiée. Si cette contrainte n'est pas respectée, alors il est possible d'obtenir après l'étape précédente un chemin optimal qui ne soit pas compris dans le graphe initial. Une alternative est alors de transformer le graphe non complet en un graphe facticement complet : s'il n'y a pas d'arêtes entre deux nœuds, nous calculons le plus court chemin, grâce à l'algorithme de Dijkstra, entre ces deux nœuds, puis nous considérons une nouvelle arête de poids la somme des poids des arêtes parcourues.

À l'issue de cette étape, les images du drone seront étudiées et le graphe de la ville sera modifié en fonction du niveau de neige sur les routes. Grâce à ces nouvelles données nous pourrons prioriser les routes à déneiger afin d'être le plus efficace possible, tout en s'assurant de conserver un graphe fortement connexe : en effet, retirer du graphe initial les arêtes qui n'ont pas besoin « d'être déneigées » a pour risque de le diviser en plusieurs sous graphes, ce qui rendrait certaines rues inaccessibles pour les déneigeuses.

2. Recherche du trajet minimal des déneigeuses

Ici, le changement majeur par rapport au parcours du drone est la prise en compte des sens de circulation.

Périmètre étudié : nous avons décidé de considérer dans un premier temps un simple quartier de Montréal : le quartier d'Anjou qui a la particularité d'être assez petit en surface. Le but étant d'ensuite appliquer cette solution à chaque quartier de Montréal.
Limitations : nous avons choisi de ne pas prendre en compte le trafic et les potentiels ralentissements, et de réduire l'impact de ce choix en ajoutant une marge d'erreur sur le temps final de parcours.
Métriques à prendre en compte :

- kilométrage effectué par une seule déneigeuse
- temps de parcours
- kilométrage total

Ces métriques nous permettront, comme pour le drone, d'ajuster au mieux le temps de parcours et les coûts, en ajoutant potentiellement des machines.

Démarche algorithmique

Pour les déneigeuses, nous avons procédé grâce à trois étapes :

1. Parcours d'une seule déneigeuse, dans un graphe eulérien

On considère ici une seule déneigeuse pour parcourir chaque quartier.
Cette itération prend en entrée un graphe eulérien (c'est-à-dire qu'il existe forcément un chemin qui passe par chaque arête, une seule fois, et qui revient au sommet de départ). Le but à la fin de cette étape est de trouver un tel chemin dans le graphe pour constituer l'itinéraire de la déneigeuse.

Ce problème s'apparente au problème du postier chinois sur un graphe orienté. Il nous a fallu ici trouver un cycle eulérien dans le graphe pour obtenir le chemin à parcourir.

2. Parcours d'une seule déneigeuse, graphe eulérien avec poids

Nous considérons toujours un graphe eulérien. Cette fois, nous accordons à chaque arc un poids, qui correspond au nombre de kilomètres de la rue. Ceci va nous permettre d'évaluer précisément la métrique de kilométrage.

Il suffit ici de trouver un cycle eulérien dans le graphe pour obtenir le chemin à parcourir et calculer à partir des poids des arcs le kilométrage total (grâce auquel on pourra calculer les métriques).

3. Parcours d'une seule déneigeuse, dans un graphe pas nécessairement eulérien

On considère toujours ici une seule déneigeuse pour parcourir chaque quartier.
Cette itération prend cette fois en entrée un graphe pas nécessairement eulérien.

Il faut ici effectuer tout d'abord un test pour savoir si le graphe est eulérien ou non, s'il l'est, on cherche un cycle eulérien et cela nous donnera le chemin à parcourir pour la déneigeuse. S'il ne l'est pas, on applique **l'Algorithme du Stepping Stone**, qui consiste à améliorer itérativement une solution de base (non optimale) jusqu'à obtenir une solution non optimisable. Pour trouver la solution de base, il y a plusieurs options, nous avons choisi d'utiliser **l'Algorithme de Balas-Hammer**. À l'issue de l'application de cet algorithme, on peut ajouter des arcs fictifs au graphe initial, et ainsi obtenir un graphe eulérien ; nous avons eulérianisé le graphe. Il suffit ensuite de chercher finalement un cycle eulérien.