INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Département Génie Mathématique

Stage de 3GM

Optimisation du damage d'une station de ski

11 juin au 31 juillet 2019

Théo Guyard

theo.guyard@insa-rennes.fr

Organisme d'accueil : **INSA Rennes et IRMAR**

Maître de stage : **Jeremy Omer**

Enseignant référent : **Ayse Nur Arslan**

Résumé

Chaque soir dans les stations de ski, des dameuses partent pour remettre en état la station. Le but est de damer les pistes qui ont été utilisées la journée précédente. L'objectif de ce stage est de trouver une manière optimale d'effectuer le damage d'une station de ski à partir d'un plan de la station et d'un nombre de véhicules disponibles. L'approche utilisée est la modélisation sous forme de graphe de la station. Cette approche permet de lier le problème de damage aux problèmes classiques de routage. Suivant les paramètres du problème, on sera amené à résoudre un type de problème de routage en particulier. Pour les problèmes les plus simples, une modélisation par PLNE sera décrite puis implémentée pour être résolue par un solver. Dans le cas le plus complexe, un algorithme de Branch-and-Price sera proposé. L'objectif de ce stage est également de se familiariser avec le monde de la recherche, les publications scientifiques, d'apprendre à effectuer des recherches bibliographiques et de commencer à faire de la recherche sur un problème encore ouvert. Il a été effectué au sein du département Génie Mathématique de l'INSA Rennes.

Mots clés : Graphes, Programmes linéaires, PLNE, Problème du postier Chinois, CARP, Branch-and-Price

Remerciements

Je tiens tout particulièrement à remercier M. Omer et Mme Arslan de m'avoir proposé ce stage et guidé durant sa réalisation ainsi que M. Gilbert pour avoir contribué à la relecture de ce rapport.

Rapport complet

Ce rapport respecte une contrainte de nombre de pages pour qu'il puisse être évalué. Un rapport plus détaillé est disponible à l'adresse suivante : https://github.com/TheoGuyard/Groomer-Optimization-Problem.

Table des matières

1 Introduction

1.1 Problème

Pour attirer des clients, une station de ski se doit d'avoir des pistes avec une bonne neige. Pour entretenir les pistes, les stations sont damées chaque soir. Des dameuses passent sur les pistes afin de ré-étaler et renouveler la neige. Cette opération est effectuée sur une partie ou sur la totalité des pistes de la station. Le problème qui va être traité lors de ce stage consiste à trouver une manière optimale de damer les pistes d'une station de ski. Pour cela, on aura à disposition un plan des pistes et un nombre de véhicules disponibles. On connaîtra également certaines informations quand au coût des opérations de damage sur les pistes et quand aux caractéristiques des dameuses. L'objectif sera de trouver le chemin que doit effectuer chaque dameuse pour la prochaine session de damage. Il n'est pas obligatoire de toutes les utiliser. Par la suite, ce problème sera appelé GOP (Groomer Optimisation Problem).

Dans une station de ski, toutes les dameuses sont entreposées dans des hangars appelés "dépôts" desquels elles partent et auxquels elles doivent revenir en fin de service. On suppose qu'une dameuse est toujours affectée à un unique dépôt. Toutes les pistes et les chemins praticables par les dameuses sont connectés entre eux par au moins une jonction de telle manière qu'on puisse atteindre n'importe quel endroit de la station depuis tous les dépôts. Néanmoins, certaines pistes ne sont pas praticables dans les deux sens pour les dameuses.

Dans les petites stations possédant une unique dameuse, celle-ci doit être capable de damer toutes les pistes en une seule session de damage. Dans les stations possédant plusieurs dameuses, il faut assigner un chemin pour chaque dameuse afin qu'au terme d'une session de damage, toutes les pistes qui avaient besoin d'entretien aient été damées. Pour éviter qu'une dameuse soit surchargée de travail par rapport aux autres, chaque dameuse possède une charge maximale de travail qu'elle ne peut pas dépasser.

Pour définir l'optimalité du GOP, nous considéreront qu'on peut définir pour chaque piste un demande, un coût de passage sur la piste et un coût de damage de la piste. Si la demande d'une piste est positive, alors elle doit être damée. On pourra donc associer un coût à chaque chemin effectué par les dameuses en fonction des pistes qu'elles dament ou non. Le damage optimal sera celui qui minimise ces coûts pour toutes les dameuses et qui permet que toutes le pistes avec une demande positive soient damées.

1.2 Formalisation du problème

1.2.1 Modélisation

Nous allons modéliser le GOP sous forme de graphe. L'ensemble des pistes formera les arêtes du graphe et les jonctions entre les pistes formeront ses sommets. Chaque arête aura une demande ainsi qu'un coût de passage et un coût de damage associé.

On note $G = (S, A' \cup E)$ le graphe associé aux pistes où S est l'ensemble des jonctions sur les pistes et $A' \cup E$ est l'ensemble des pistes ou chemins praticables pour les dameuses. Une piste sera représentée par un arc (de A') si la piste est praticable dans un seul sens ou par une arête (de E) si elle est praticable dans les deux sens. On dira d'une piste qu'elle est "requise" si elle a une demande positive et "servie" si elle est damée par une dameuse. Il sera possible que plusieurs dameuses empruntent la même piste lors d'une session de damage. Par conséquent, une dameuse pourra passer sur une piste sans la damer. Comme les dameuses devront partir et revenir au dépôt qui leur est associé, elles devront effectuer des cycles sur G. Dans le cas avec plusieurs dameuses, on leur assignera une capacité maximale. La somme des demandes sur les pistes traitées par une dameuse ne devra pas excéder cette capacité maximale afin de ne pas surcharger la dameuse de travail.

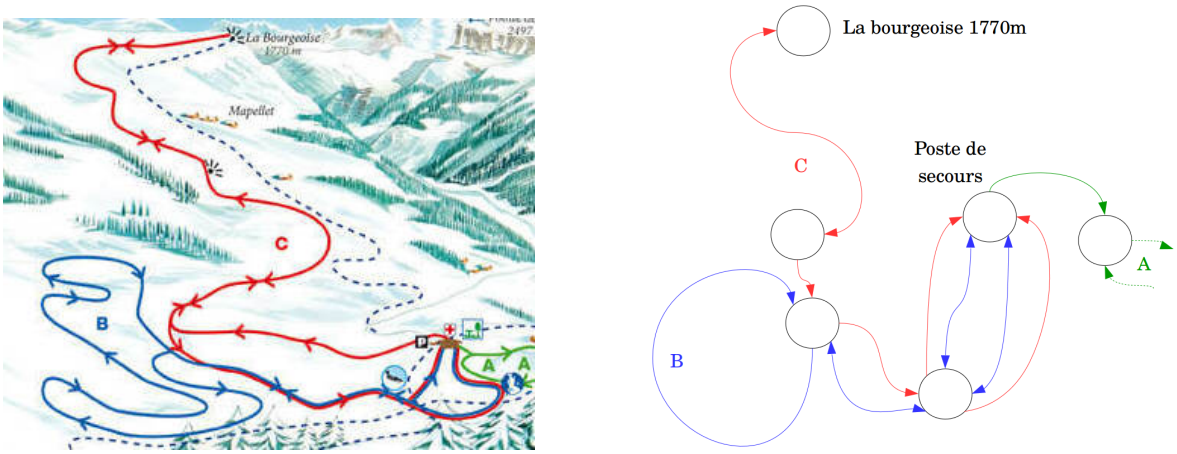

FIGURE 1 – Modélisation de pistes sous forme de graphe (ici un graphe mixte)

1.2.2 Gestion des coûts et de la demande

Dans le modèle, on suppose qu'on peut associer une demande ainsi qu'un coût de damage et un coût de passage sur chaque piste. Nous proposons ici une manière d'estimer ces valeurs pour une piste.

Chaque soir, la station fait un point sur la qualité des pistes et détermine celles qui sont à damer. La charge de travail n'est pas forcément égale pour toutes les pistes. Celle-ci peut dépendre de la quantité de neige à traiter, de la qualité de la neige, etc... Dans notre modèle, on effectuera une somme pondérée de ces différentes charges de travail qu'on regroupera sous une unique valeur : la demande de la piste. On pourra également se baser sur des caractéristiques de la piste (longueur, pente, etc...) ainsi que sur le prix du carburant ou le prix d'entretien d'une dameuse pour calculer les coûts de damage et de passage sur une piste.

Exemple :

Prenons l'exemple d'une piste très pentue sur laquelle on doit damer $50m^3$ de neige. Les équipes de la station ont calculé que la consommation de carburant est de 1 L pour $10m^3$ de neige. Cette piste mesure 2 km et le carburant coûte 5 € par litre. De plus, on ne peut que monter la piste avec une dameuse donc la consommation de carburant sera multipliée par 1.5. Enfin, la consommation pour le passage sur une piste est de 1 L par kilomètre.

Demande :

$$d = 50\ m^3$$

Coût de passage :

$$p = 2\ km \times\ 1.5 \times 1\ \frac{L}{km}\ \times 5\frac{€}{L} = 15€$$

Coût de damage :

$$c = p + \frac{1\ L}{10\ m^3} \times d \times 5\frac{€}{L} = 40€$$

1.2.3 Caractéristiques propres au problème

Deux caractéristiques propres au GOP pourront justifier les choix qui seront faits par la suite pour le résoudre.

Taille
Une des principales caractéristiques du problème est la taille. En effet, en restant dans le cadre des stations de ski, on sera quasiment sûr que le graphe obtenu en modélisant les pistes de dépassera pas une certaine taille. Il sera rare d'avoir plus de 100 sommets en modélisant le graphe.

Coûts
Le damage des stations de ski est assez coûteux. Rajouter une dameuse ou effectuer un chemin qui n'est pas optimal peut être très pénalisant. Donner une solution approchée du problème n'est donc pas une bonne solution en pratique. On cherchera toujours une solution exacte et, si possible, une qui minimise le nombre de dameuses utilisées.

1.3 Lien avec les problèmes de routage

La modélisation sous forme de graphe du GOP permet de faire un lien avec les problèmes classiques de routage : le problème du postier Chinois (CPP, DCPP, RCPP), le problème du postier Chinois rural (RCPP, DRCPP, MRCPP), le problème de routage avec capacité (CARP) ou le problème de routage avec capacité multi-dépôt (MD-CARP) [19].

Dans le cas du GOP avec une unique dameuse, on suppose que la dameuse peut s'occuper de la totalité des pistes toute seule et on ne prend donc pas en compte sa capacité. Si l'ensemble des pistes est requis, on sera dans le cas du CPP, DCPP ou MCPP suivant le type des pistes. Si seulement une partie des pistes est requise, on sera dans le cas du RCPP, DRCPP ou MRCPP. Dans le cas d'un GOP avec plusieurs dameuses, on peut leur associer à chacune une capacité maximale afin qu'une dameuse ne soit pas surchargée par rapport aux autres. Le problème reviendra donc à un problème de CARP. On traitera tous les problèmes avec plus d'une dameuse comme des problèmes de MCARP (CARP pour un graphe mixte) car dans la plupart des cas, le graphe obtenu lors de la modélisation est mixte. Enfin, si la station possède plusieurs dépôts de dameuse, on sera dans le cas du MD-MCARP.

1.4 Méthode de résolution

Lors de la résolution du GOP, on cherchera dans un premier temps à trouver à quel problème de routage on peut s'apparenter. Les problèmes de postier Chinois (non ruraux) sont P-durs alors que le MCARP et le MD-MCARP sont NP-durs. Choisir le bon type de problème permettra de gagner du temps de calcul. On va ensuite résoudre un programme linéaire de manière optimale. Celui-ci permettra de trouver combien de fois chaque véhicule doit passer par chaque arc/arête et lesquels il doit servir. On pourra ensuite reconstruire le chemin de chaque dameuse à partir des arcs/arêtes sur lesquels elle passe. Le choix de modéliser ce problème sous forme de programme linéaire est motivé par le fait qu'on veut pas de solution approchée. Les résolutions de programmes linéaires peuvent prendre beaucoup de temps lorsqu'il y a un nombre important de variables. Dans notre cas, celles-ci seront liées au nombre d'arcs/arêtes dans le graphe qui ne sera pas de taille très importante. On peut donc espérer des temps de calcul convenable.

Pour les cas avec un seul dépôt, on pourra utiliser une interface de programmation mathématique qui nous permettra de décrire le PLNE puis on utilisera un solver pour le résoudre. Pour les cas de problèmes du postier Chinois ruraux, on ne formulera pas un PLNE spécifique. On utilisera la modélisation du MCARP et un seul véhicule sera donné en paramètre. Les problèmes du postier Chinois ruraux et le MCARP étant tous deux NP-durs, on ne gagnerait pas beaucoup de temps de calcul en modélisant différemment les problèmes de postier Chinois ruraux, surtout si on doit inclure la contrainte de passage par le dépôt. Le cas avec plusieurs dépôts de dameuses est plus compliqué. Dans ce cas, on proposera un algorithme de Branch-and-Price pour résoudre le GOP.

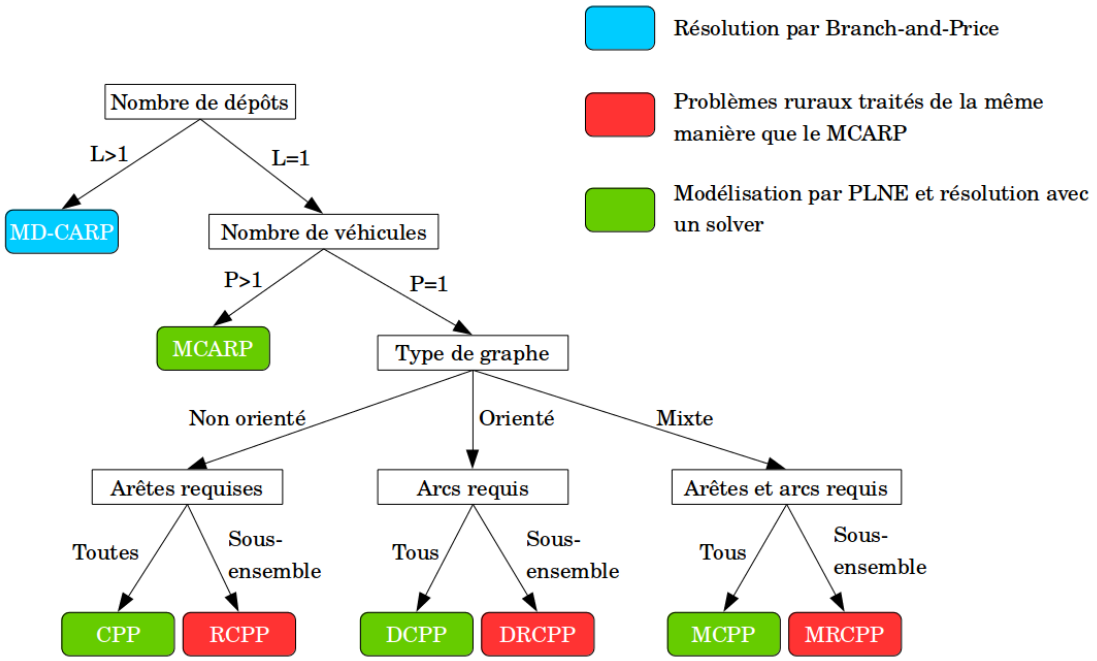

FIGURE 2 – Classification du problème de damage et méthodes de résolution

2 Recherches bibliographiques

Les problèmes de routage sont assez classiques et on retrouve cette modélisation pour de nombreux problèmes réels. Le premier problème de routage est lié au problème des sept ponts de Königsberg [7]. Ce problème consiste à trouver un chemin passant une unique fois par chacun des sept ponts de la ville de Königsberg. Euler a démontré en 1736 que ce problème n'avait pas de solution, mais que si on permettait de passer plusieurs fois par les ponts, on pouvait trouver un chemin de longueur minimale pour ce problème.

En 1962, Kwan Mei-Ko a été le premier à proposer une solution au problème plus général de couverture par les arêtes dans un graphe connexe non-orienté [18]. C'est à ce mathématicien Chinois que le CPP doit son nom. Kwan Mei-Ko montre que ce problème revient à trouver un couplage parfait de poids minimum entre les sommets de degrés impair afin de rajouter des arêtes au graphe pour le rendre Eulérien. Il faudra ensuite trouver un chemin Eulérien dans le nouveau graphe, à l'aide par exemple de l'algorithme de Hierholzer [2]. À cette époque, il n'existe pas d'algorithme polynomial pour résoudre le problème de couplage parfait de poids minimum.

J. Edmonds propose en 1965 l'algorithme du "Blossom" [5] qui permet de résoudre un problème de couplage maximal en un temps polynomial. C'est le premier algorithme polynomial permettant de résoudre un problème de couplage. Par la suite, des versions plus élaborées de cet algorithme sont développées pour résoudre tout type de problème de couplage. On peut notamment citer l'algorithme "Blossom V" [12] développé en 2010 par Kolmogorov qui est la dernière version permettant de résoudre efficacement le problème du couplage parfait de poids minimum. À partir des années 1970, le CPP est donc classé comme un problème P-dur.

Par la suite, de nombreuses variantes au CPP ont été étudiées. Le DCPP ou le MCPP sont des dérivés du CPP sur d'autres types de graphes. Les problèmes de type "rural" sont des CPP où on doit couvrir simplement un sous-ensemble des arêtes. Les premiers algorithmes de résolution exacte, introduits par J. Edmonds en 1973, traitent le DCPP par un problème de flot et transforment le MCPP pour pouvoir le traiter de la même manière [6] [16] [9]. Les problèmes de type "rural" sont, quand à eux, de type NP-dur. Il faut attendre 1981 pour avoir les premières formulations exactes sous forme de programme linéaire de ces problèmes [4]. Celles-ci utilisent un nombre exponentiel de contraintes.

Les problèmes dans lesquels on doit trouver une couverture par les arêtes d'un graphe pour un nombre fixé de chemins sont aussi de classe NP-dur. Des heuristiques sont alors développées à partir des années 2000 pour résoudre ce type de problème en un temps raisonnable. Il existe différentes approches. On peut partir d'une solution du CPP puis découper le cycle en sous-cycles ou alors utiliser des méthodes de cluster [20].

Au début des années 2000, le taille grandissante des réseaux de distribution ou des réseaux de communication font émerger des problèmes de type CARP [10]. Ces problèmes consistent à trouver sur un graphe un ensemble de chemins couvrant un sous-ensemble des arêtes du graphe tout en respectant certaines contraintes sur chaque chemin. Les réseaux pouvant très simplement être modélisés sous forme de graphe, ces problèmes sont encore aujourd'hui au cœur de nombreux sujets de recherche. Wøhlk donne une liste et un historique des problèmes et des solutions liées au CARP[22].

Le CARP est NP-dur, on peut le relier au problème du voyageur de commerce en transformant le graphe [8]. La résolution par programme linéaire en nombres entiers est quasiment systématiquement utilisée lorsqu'il faut trouver une solution exacte. Une première formulation est faite en 1981 par Golden et Wong mais nécessite un nombre exponentiel de contraintes [10]. Belenguer et Benavent ont beaucoup travaillé sur les formulations du CARP pour les simplifier. Ils proposent en 2003 un modèle avec une seule variable de décision par arc, mais avec un nombre de contraintes toujours exponentiel [3] en se basant sur les méthodes des plans coupants. Le problème a été étendu par la suite à tout type de graphes sous la dénomination de MCARP et la première formulation compacte de ce problème a été publiée en 2010 [14]. Celle-ci introduit une nouvelle variable de flot. Ceci permet d'avoir un nombre

polynomial de variables. C'est la première formulation qui donne des résultats satisfaisants sur des jeux de données de taille moyenne [21]. Cette formulation compacte est étendue au MD-CARP (plusieurs dépôts disponibles) en 2015 [13]. Des algorithmes de Branch-and-Price ont également été développés spécifiquement pour le CARP afin de résoudre ce problème sur de plus grosses instances [17].

En parallèle, des heuristiques sont développées pour les cas de graphes de grande dimension et où une solution exacte n'est pas forcément nécessaire. Ces problèmes sont souvent liés à l'industrie ou aux problèmes de réseaux de distribution/collecte dans les grandes villes. L'exemple de la collecte de déchets dans les villes est d'ailleurs l'exemple le plus courant du MCARP/MD-CARP. Ces heuristiques sont basées sur plusieurs concepts comme le clustering [11], les recherches avec tabou [1] ou les algorithmes de colonies de fourmis [15]. Une bibliographie très complète des problèmes de routage a été proposée en 2017 par M. Mourao [19]. Elle détaille les nombreux types de problèmes de routage ainsi que les solutions utilisées pour les résoudre.

3 Formulations des PLNE pour un unique dépôt

3.1 Notations

Les notations seront similaires pour les différentes formulations mais elles ne seront pas forcément toutes utilisées. On note :

- G le graphe du problème (non-orienté, orienté ou mixte)
- S les sommets de G
- A' l'ensemble des arcs de G
- A_R l'ensemble des arcs requis de G
- E l'ensemble des arêtes de G
- E_R l'ensemble des arêtes requises de G
- $A = A' \cup \{(i,j),(i,j) \in E\} \cup \{(j,i),(i,j) \in E\}$ l'ensemble des arcs et des arêtes de G où chaque arête a été remplacée par deux arcs opposés (en conservant les caractéristiques de l'arête)
- $\Gamma = (S, A)$ le graphe orienté formé à partir de A
- R l'ensemble des arcs requis dans Γ ($|R| = 2|E_R| + |A_R|$)
- q_{ij} la demande sur l'arc $(i,j) \in R$
- d_{ij} le coût de passage sur l'arc $(i,j) \in A$
- c_{ij} le coût de service sur l'arc $(i,j) \in R$
- Q_T la demande totale sur G
- W la capacité maximale de chaque véhicule
- 0 le nœud de Γ correspondant au dépôt
- P le nombre de véhicules disponibles

3.2 CPP

On a $P = 1,\ A = A_R = \emptyset$ et $E = E_R$ (un seul véhicule, pistes praticables dans les deux sens et qui doivent toutes êtres damées).

Variables

- $y_{ij},\ (i,j) \in E$ est le nombre de fois qu'on passe par l'arête (i,j) sans la servir
- $k_i,\ i \in S$

Le nombre de passages sur une arête est $y_{ij} + 1$.

Formulation

$$\text{Minimize :} \quad \sum_{(i,j)\in E} c_{ij} + \sum_{(i,j)\in E} y_{ij} d_{ij} \tag{1}$$

$$\text{Subject to :} \quad \sum_{(i,j)\in E} (1 + y_{ij}) = 2k_i \qquad \forall i \in S \tag{2}$$

$$y_{ij} \in \mathbb{N} \qquad \forall (i,j) \in E \tag{3}$$

$$k_i \in \mathbb{N} \qquad \forall i \in S \tag{4}$$

Commentaires

L'équation (1) traduit le fait qu'il faille minimiser les coûts de service sur toutes les arêtes (car on les sert toutes une unique fois) ainsi que les coûts de passage sur les arêtes sur lequel on passe sans service. La première somme de (1) est constante et n'est donc pas utile dans cette formulation, mais pour plus de clarté, on la conserve. Les équations (2) et (4) permettent d'être sûr qu'on forme un chemin continu. En effet, si on rentre dans un nœud i, on doit pouvoir en sortir donc le nombre d'arêtes adjacentes à un nœud est pair. Enfin, (3) nous permet d'avoir un nombre de passage entier par arête.

3.3 DCPP

On a $P = 1,\ E = E_R = \emptyset$ et $A = A_R$ (un seul véhicule, pistes praticables dans un seul sens et qui doivent toutes êtres damées).

Variables

- y_{ij}, $(i,j) \in A$ est le nombre de fois qu'on passe par l'arc (i,j) sans le servir

Le nombre de passages sur un arc est $y_{ij} + 1$.

Formulation

$$\text{Minimize :} \quad \sum_{(i,j)\in A} c_{ij} + \sum_{(i,j)\in A} y_{ij} d_{ij} \tag{1}$$

$$\text{Subject to :} \quad \sum_{(i,j)\in A} (1 + y_{ij}) = \sum_{(j,i)\in A} (1 + y_{ji}) \quad \forall i \in S \tag{2}$$

$$y_{ij} \in \mathbb{N} \qquad \forall (i,j) \in A \tag{3}$$

Commentaires

Comme précédemment, la première somme de (1) n'est pas utile, mais elle sert à clarifier les notations. On souhaite minimiser le coût de service sur tous les arcs (car on doit tous les servir un unique fois) et les coûts de passage sur les arcs empruntés sans service. L'équation (2) permet de s'assurer que le chemin construit est continu . En effet, on doit pourvoir sortir autant de fois que l'on entre dans chaque sommet de S pour former un chemin. Enfin, (3) permet d'avoir un nombre de passage entier par arc.

3.4 MCPP

On a $P = 1$, $E = E_R$ et $A' = A_R$ donc $R = A$ (un seul véhicule, pistes praticables dans un seul ou dans les deux sens et qui doivent toutes être damées).

Variables

- $x_{ij} = \begin{cases} 1 \text{ si on sert l'arc } (i,j) \in R \\ 0 \text{ sinon} \end{cases}$
- y_{ij}, $(i,j) \in A$ est le nombre de fois qu'on passe par l'arc (i,j) sans service

Le nombre de passages sur un arc est $x_{ij} + y_{ij}$.

Formulation

$$\text{Minimize :} \quad \sum_{(i,j)\in A} x_{ij} c_{ij} + \sum_{(i,j)\in A} y_{ij} d_{ij} \tag{1}$$

$$\text{Subject to :} \quad \sum_{(i,j)\in A} (x_{ij} + y_{ij}) = \sum_{(j,i)\in A} (x_{ji} + y_{ji}) \quad \forall i \in S \tag{2}$$

$$x_{ij} = 1 \qquad \forall (i,j) \in A_R \tag{3}$$

$$x_{ij} + x_{ji} = 1 \qquad \forall (i,j) \in E_R \tag{4}$$

$$y_{ij} \in \mathbb{N} \qquad \forall (i,j) \in A \tag{5}$$

Commentaires

L'équation (1) traduit le fait qu'on veuille minimiser les coûts de service sur les arcs servis et les coûts de passage sur les arcs empruntés sans service. (2) permet d'assurer la continuité des chemins formés comme expliqué précédemment. (3) nous assure que chaque arc requis est servi et (4) nous assure que chaque arête requise de E_R est servie une seule fois (on peut y passer dans les deux sens, mais en la servant qu'une seule fois). Enfin, (5) permet d'avoir un nombre de passage entier par arc.

3.5 MCARP

Variables

- $x_{ij}^p = \begin{cases} 1 \text{ si le véhicule } p \text{ sert l'arc } (i,j) \in R \\ 0 \text{ sinon} \end{cases}$
- y_{ij}^p, $(i,j) \in A$ est le nombre de fois que le véhicule p passe par l'arc (i,j) sans le servir

- $f_{ij}^p,\ (i,j) \in A$ est le flot sur l'arc $(i,j) \in A$ correspondant à la demande restante dans le chemin effectué par le véhicule p

Le nombre de fois que le véhicule p passe sur un arc est $x_{ij}^p + y_{ij}^p$.

Formulation

$$\text{Minimize :} \quad \sum_{p=1}^{P} \left[\sum_{(i,j)\in R} x_{ij}^p c_{ij} + \sum_{(i,j)\in A} y_{ij}^p d_{ij} \right] \tag{1}$$

$$\text{Subject to :} \quad \sum_{(i,j)\in R} x_{ij}^p + \sum_{(i,j)\in A} y_{ij}^p = \sum_{(j,i)\in R} x_{ji}^p + \sum_{(j,i)\in A} y_{ji}^p \qquad \forall i \in S \qquad \forall p \in \{1,...,P\} \tag{2}$$

$$\sum_{p=1}^{P} x_{ij}^p = 1 \qquad \forall (i,j) \in A_R \tag{3}$$

$$\sum_{p=1}^{P} (x_{ij}^p + x_{ji}^p) = 1 \qquad \forall (i,j) \in E_R \tag{4}$$

$$\sum_{(0,j)\in A} y_{0j}^p + \sum_{(0,j)\in R} x_{0j}^p \leq 1 \tag{5}$$

$$\sum_{(j,i)\in A} f_{ji}^p - \sum_{(i,j)\in A} f_{ij}^p = \sum_{(j,i)\in R} x_{ji}^p q_{ji} \qquad \forall i \in S \setminus 0 \qquad \forall p \in \{1,...,P\} \tag{6}$$

$$\sum_{(0,j)\in A} f_{0j}^p = \sum_{(i,j)\in R} x_{ij}^p q_{ij} \qquad \forall p \in \{1,...,P\} \tag{7}$$

$$\sum_{(i,0)\in A} f_{i0}^p = \sum_{(i,0)\in R} x_{i0}^p q_{i0} \qquad \forall p \in \{1,...,P\} \tag{8}$$

$$f_{ij}^p \leq W(x_{ij}^p + y_{ij}^p) \qquad \forall (i,j) \in A \qquad \forall p \in \{1,...,P\} \tag{9}$$

$$f_{ij}^p \geq 0 \qquad \forall (i,j) \in A, \qquad \forall p \in \{1,...,P\} \tag{10}$$

$$y_{ij}^p \in \mathbb{N} \qquad \forall (i,j) \in A, \qquad \forall p \in \{1,...,P\} \tag{11}$$

Commentaires

L'équation (1) traduit le fait qu'on veuille minimiser la somme du coût de service sur les arcs servis ainsi que le coût de passage sur les arcs sur lesquels on passe et ce pour tous les véhicules. (2) permet d'avoir des chemins continus pour chaque véhicule comme dans le cas du MCPP. L'équation (3) (resp. (4)) nous assure qu'au moins un véhicule sert chaque arc requis (resp. sert chaque arête requise dans un seul sens). (5)-(9) sont les contraintes liées au flot qui permettent d'éviter des sous-tours. Une explication est faite au paragraphe suivant. (9) permet de garantir le respect de la capacité de chaque véhicule (on peut prouver cela en sommant sur les acrs de A puis en utilisant (5) et (7), on obtient alors $\sum_{(i,j)\in R} q_{ij} x_{ij}^p \leq W$). Enfin, (10)-(11) permettent d'avoir un flot positif et un nombre de passage entier. On peut remarquer que (5) impose qu'un tour se termine dès lors qu'il repasse au dépôt. Il se peut alors qu'un véhicule soit utilisé pour un tout petit cycle. Dans ce cas, on peut augmenter le nombre de véhicules puis assigner un véhicule à plusieurs petites tournées, tout en respectant sa capacité.

Variables de flot

Les variables de flots permettent de supprimer les sous-tours et elles représentent la demande restante dans un chemin pour chaque arc. L'équation (7) permet d'initialiser le flot sur le premier arc du chemin

avec la valeur de la demande totale du chemin. Pour chaque arc, le flot sera le flot de l'arc précédent moins la demande de l'arc qu'on vient de traverser (6). Enfin, la contrainte (8) permet de vérifier que sur le dernier arc, le flot correspond bien à la demande de l'arc. Dans le cas d'un sous-tour, on n'aurait pas pu décrémenter le flot des demandes des arcs compris dans le sous-tour et le flot sur l'arc terminant dans le dépôt serait donc supérieur à la demande de l'arc, (8) ne serait donc pas vérifiée.

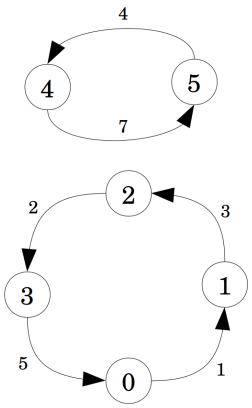

FIGURE 3 – Chemin avec un sous-tour (vérifiant quand même (2))

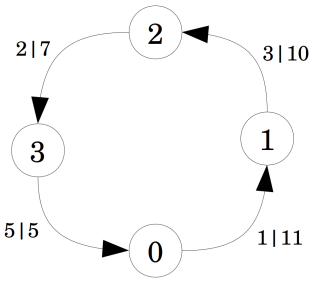

FIGURE 4 – Chemin valide avec demande et flot sur chaque arc

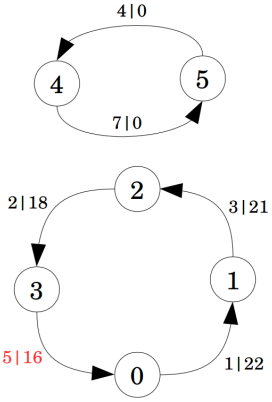

FIGURE 5 – Chemin non-valide avec demande et flot sur chaque arc

Contraintes valides supplémentaires pour accélérer la résolution

$$\sum_{p=1}^{P}\left(\sum_{(0,j)\in A} y_{0j}^{p} + \sum_{(0,j)\in R} x_{0j}^{p}\right) \geq \left\lceil \frac{Q_T}{W} \right\rceil \tag{12}$$

$$f_{ij}^{p} \geq x_{ij}^{p} q_{ij} \qquad \forall (i,j) \in R \qquad \forall p \in \{1,...,P\} \tag{13}$$

$$f_{ij}^{p} \geq y_{ij} - 1 \qquad \forall (i,j) \in A \setminus R \qquad \forall p \in \{1,...,P\} \tag{14}$$

$$\sum_{(0,j)\in A} y_{0j}^{p} + \sum_{(0,j)\in R} x_{0j}^{p} \geq \sum_{(0,j)\in A} y_{0j}^{p+1} + \sum_{(0,j)\in R} x_{0j}^{p+1} \qquad \forall p \in \{1,...,P-1\} \tag{15}$$

Les équations (12)-(15) ne sont pas nécessaires, mais elles permettent d'accélérer la résolution en fixant le nombre minimum de véhicules à avoir en fonction de leur capacité et de la demande totale pour (12), en imposant des bornes inférieures au flot pour (13)-(14) et en brisant les symétries l'association chemin/véhicule pour (15).

4 Branch-and-Price pour le cas avec plusieurs dépôts

Lorsque plusieurs dépôts sont disponibles on se trouve dans le cas du MD-MCARP. On peut avoir envie d'adapter la formulation du MCARP en ajoutant une indexation correspondant au dépôt associé au chemin de la dameuse. Malheureusement, les contraintes de flot sont difficilement adaptables pour ce cas. Il faudrait donc revenir sur une formulation avec un nombre exponentiel de contraintes. De plus, on devrait également introduire des variables d'association entre véhicule et dépôt qui seraient elles aussi en nombre exponentiel. Une modélisation et une résolution par solver prendrait beaucoup trop de temps. On propose donc un algorithme de Branch-and-Price pour résoudre ce problème. En effet, on peut facilement remarquer qu'on peut séparer le GOP en sous-problèmes traitants chacun d'une seule tournée associée à un seul dépôt. Au moment de l'écriture de ce rapport, il ne semble qu'aucun article de recherche ne traite ce problème de cette manière.

4.1 Notations supplémentaires

On reprend en partie les notations du MCARP et on introduit des notations supplémentaires :

- $S_D \subset S$ l'ensemble des dépôts de G
- T^{dp} la tournée du véhicule p associé au dépôt $d \in S_D$
- $\mathcal{T} = \cup_{d,p}\, T^{dp}$ l'ensemble des tournées introduites dans le problème maître réduit
- $\mathcal{B}$ l'ensemble des branchements réalisés dans la branche courante
- c^{dp} le coût de T^{dp}
- $\tilde{c}^{dp}$ le coût réduit de T^{dp}

Variables

- α^{dp} le pourcentage d'importance de la tournée T^{dp} dans la solution du problème maître réduit
- $X^{dp} = \left[x_{ij}^{dp}\right]_{(i,j)\in R} \in \{0,1\}^{|R|}$ les arcs servis ou non par T^{dp}
- $Y^{dp} = \left[y_{ij}^{dp}\right]_{(i,j)\in A} \in \mathbb{N}^{|A|}$ le nombre de fois que T^{dp} passe par un arc sans le servir
- $F^{dp} = \left[f_{ij}^{dp}\right]_{(i,j)\in A} \in \mathbb{R}_+^{|A|}$ le flot sur l'arc $(i,j) \in A$ correspondant à la demande restante dans le chemin effectué par le véhicule p associé au dépôt d

Le nombre de fois que le véhicule de la tournée T^{dp} passe sur l'arc (i,j) sera donc $x_{ij}^{dp} + y_{ij}^{dp}$.

4.2 Principe de l'algorithme (BnP-opt)

On va décomposer le GOP en d'un côté un problème maître regroupant les contraintes de service des arcs/arêtes requis ainsi que la contrainte du nombre de véhicules et d'un autre côté des problèmes esclaves. Les problèmes esclaves P_e^d correspondront à un problème où, pour un dépôt d fixé, on cherche une tournée d'un véhicule partant et arrivant au dépôt tout en respectant les contraintes de continuité de la tournée et les contraintes de capacité du véhicule. Dans le problème maître réduit P_{mr}, on prendra en compte un sous-ensemble des tournées construites dans les problèmes esclaves et on cherchera à savoir lesquelles utiliser pour minimiser le coût total. Chaque tournée T^{dp} aura un coût c^{dp} calculé de la même manière que pour le MCARP ainsi qu'un coût réduit $\tilde{c}^{dp}$ tenant compte des valeurs duales du problème maître auquel il est relié. Le calcul de ces coûts sera détaillé dans la formulation des problèmes maîtres et esclaves.

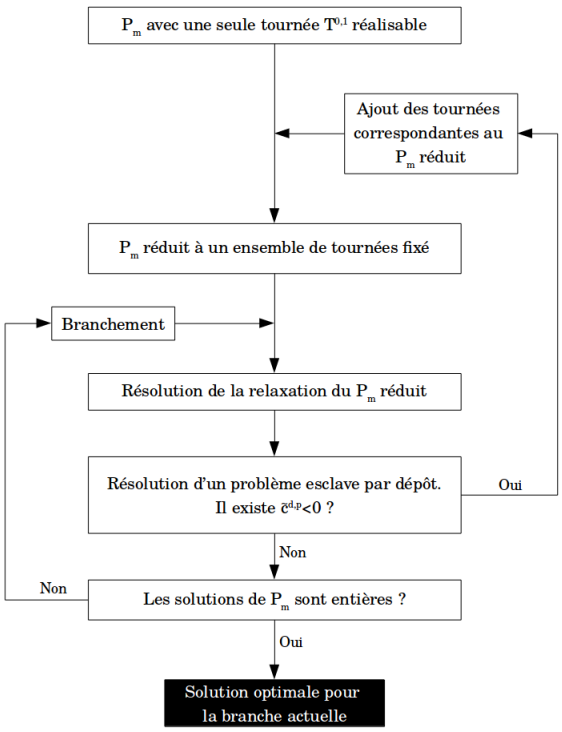

FIGURE 6 – Schéma de l'algorithme de Branch-and-Price

4.3 Formulations des problèmes maîtres et esclaves

4.3.1 Problème maître réduit (P_{mr})

On considère l'ensemble des tournées $T^{dp} \in \mathcal{T}$ introduites dans le problème maître réduit à une certaine itération de l'algorithme.

Formulation

$$\text{Minimize :} \quad \sum_{T^{dp} \in \mathcal{T}} \alpha^{dp} c^{dp} \tag{1}$$

$$\text{Subject to :} \quad \sum_{T^{dp} \in \mathcal{T}} \alpha^{dp} \leq P \tag{2}$$

$$\sum_{T^{dp} \in \mathcal{T}} \alpha^{dp} x_{ij}^{dp} \geq 1 \qquad \forall (i,j) \in A_R \tag{3}$$

$$\sum_{T^{dp} \in \mathcal{T}} \alpha^{dp} (x_{ij}^{dp} + x_{ji}^{dp}) \geq 1 \qquad \forall (i,j) \in E_R \tag{4}$$

$$\alpha^{dp} \geq 0 \qquad \forall\, T^{dp} \in \mathcal{T} \tag{5}$$

Commentaires

L'équation (1) traduit le fait qu'on veuille minimiser le coût total de toutes les tournées qu'on utilise. (2)

permet de respecter le nombre de véhicules disponibles. (3) et (4) sont les contraintes de couverture des arcs/arêtes requis. Enfin, (5) correspond aux contraintes sur les variables de choix d'utilisation d'une certaine tournée. À chaque P_{mr}, on devra ajouter les contraintes de branchements de la branche courante.

Variables duales
On récupère les différentes variables duales du problème maître afin de les utiliser dans les problèmes esclaves :

- π^p pour la contrainte sur le nombre de véhicules
- $\pi^{A_R} = \left[\pi_{ij}^{A_R}\right]_{(i,j)\in A_R}$ pour la contrainte de couverture des arcs requis
- $\pi^{E_R} = \left[\pi_{ij}^{E_R}\right]_{(i,j)\in E_R}$ pour la contrainte de couverture des arêtes requises
- $\pi^B = \left[\pi_b^B\right]_{b\in\mathcal{B}}$ pour les contraintes des branchements réalisés dans la branche courante

Le détail des branchements est expliqué par la suite.

4.3.2 Problèmes esclaves (P_e^d)

On définit un problème esclave par dépôt et pour chaque problème esclave, on cherche à trouver une tournée partant de ce dépôt qui est continue et qui respecte la capacité du véhicule. Ici, p et d seront donc fixés pour chaque (P_e^d). Le coût d'une tournée sera :

$$c^{dp} = \sum_{(i,j)\in R} x_{ij}^{dp} c_{ij} + \sum_{(i,j)\in A} y_{ij}^{dp} d_{ij}$$

On aura le coût réduit de la tournée T^{dp} associée à (P_e^d) suivant :

$$\begin{aligned}
\tilde{c}^{dp} &= c^{dp} && (1)\\
&- \pi_p && (2)\\
&- \sum_{(i,j)\in A_R} \pi_{ij}^{A_R}(i,j) x_{ij}^{dp} && (3)\\
&- \sum_{(i,j)\in E_R} \pi_{ij}^{E_R}(x_{ij}^{dp} + x_{ji}^{dp}) && (4)\\
&- \sum_{\{(i,j),d'\}\in\mathcal{B} \ \setminus \ d'=d} \pi^B_{\{(i,j),d'\}} x_{ij}^{dp} && (5)
\end{aligned}$$

Le coût réduit prend en compte le coût de la tournée auquel on soustrait des termes correspondants aux valeurs duales. (2) correspond à la contrainte du nombre de véhicules dans P_{mr}, (3) et (4) correspondent aux contraintes de service d'arcs/arêtes dans P_{mr}. Enfin, (5) correspond aux contraintes introduites lors des branchements qui portent sur le dépôt traité dans P_e^d. Les signes négatifs devant les termes des valeurs duales sont à titre indicatifs. En effet, les valeurs duales peuvent être positives ou négatives suivant le sens de l'inégalité de la contrainte correspondante dans P_{mr}. On cherchera toujours à ce que les termes des valeurs duales soient négatifs.

Formulation

$$\text{Minimize :} \quad \tilde{c}^{dp} \tag{1}$$

$$\text{Subject to :} \quad \sum_{(i,j)\in R} x_{ij}^{dp} + \sum_{(i,j)\in A} y_{ij}^{dp} = \sum_{(j,i)\in R} x_{ji}^{dp} + \sum_{(j,i)\in A} y_{ji}^{dp} \qquad \forall i \in S \tag{2}$$

$$\sum_{(0,j)\in A} y_{0j}^{dp} + \sum_{(0,j)\in R} x_{0j}^{dp} \leq 1 \tag{3}$$

$$\sum_{(j,i)\in A} f_{ji}^{dp} - \sum_{(i,j)\in A} f_{ij}^{dp} = \sum_{(j,i)\in R} x_{ji}^{dp} q_{ji} \qquad \forall i \in S \setminus d \tag{4}$$

$$\sum_{(d,j)\in A} f_{dj}^{dp} = \sum_{(i,j)\in R} x_{ij}^{dp} q_{ij} \tag{5}$$

$$\sum_{(i,d)\in A} f_{id}^{dp} = \sum_{(i,d)\in R} x_{id}^{dp} q_{id} \tag{6}$$

$$f_{ij}^{dp} \leq W(x_{ij}^{dp} + y_{ij}^{dp}) \qquad \forall (i,j) \in A \tag{7}$$

Commentaires
Dans (1), on traduit le fait qu'il faille minimiser le coût réduit de la tournée. Les contraintes (2) - (6) permettent de créer des tournées qui partent d'un dépôt, qui y reviennent, qui ne créent pas de sous-tours et qui respectent la capacité de chaque véhicule comme pour le MCARP.

4.4 Étapes importantes de l'algorithme

4.4.1 Tournée initiale

À l'initialisation de l'algorithme, on crée une tournée fictive associée à aucun dépôt, passant par tous les arcs/arêtes requis et avec un coût très élevé (10^3 fois la somme de tous les coûts de damage). Cette tournée permettra de toujours avoir une solution au problème maître. Néanmoins, si cette solution est choisie par le problème maître, c'est qu'il n'existe pas d'autres combinaisons de tournées permettant de passer par tous les arcs/arêtes requis. Le coût lié au nœud ayant pour solution cette tournée initiale ne sera jamais la solution optimale car son coût est trop élevé. Si on obtient une solution comprenant cette tournée, c'est qu'il n'existe pas de solution au GOP et qu'il faut augmenter le nombre de véhicules ou leur capacité.

4.4.2 Intégrité des solutions

Lors de la résolution du P_{mr}, la variable α^{dp} est réelle. Néanmoins, on ne peut pas avoir une tournée "non-entière". La relaxation de cette variable permet d'explorer plus vite l'arbre de branchement. Lorsque la résolution du P_{mr} conduit à des α^{dp} qui sont tous soit 0, soit 1 (les α^{dp} ne dépasseront pas 1 car avoir un $\alpha^{dp} > 1$ ne permet pas de minimiser), on aura la solution optimale pour la branche actuelle. La solution optimale du GOP sera la meilleure des solutions optimales des branches lorsqu'on ne pourra plus explorer aucun nœud.

Lorsqu'on obtient une solution non-entière au problème maître du nœud N, on trouve alors une borne inférieure $\underline{z}^N$ aux solutions dans la branche courante. Si la solution est entière, on trouve une borne supérieure globale $\bar{z}$ au problème. $\bar{z}$ sera mis à jour à chaque fois qu'on trouve des solutions entières qui améliorent le précédent $\bar{z}$. Si on trouve dans un nœud $\underline{z}^N > \bar{z}$, alors c'est qu'on ne peut pas améliorer les solutions dans cette branche et elle sera donc coupée.

4.4.3 Variables de branchement

Lors d'un branchement, nous allons créer deux nouveaux nœuds. Dans un nœud, on empêchera les tournées partant d'un dépôt de servir un arc et dans l'autre nœud, on obligera au moins un véhicule de ce dépôt de servir cet arc (en tenant compte du nombre de véhicules disponibles). Le dépôt et l'arc seront choisis à l'avance.
Dans un nœud, la contrainte introduite dans P_{mr} sera

$$\sum_p \alpha^{dp} x^{dp}_{i_B j_B} = 0 \qquad \text{avec } d \in S_D \text{ et } (i_B, j_B) \in R \text{ fixés}$$

Dans l'autre nœud, on introduira dans dans P_{mr} les contraintes

$$\begin{cases} \sum_p \alpha^{dp} x^{dp}_{i_B j_B} \geq 1 \\ \sum_p \alpha^{dp} x^{dp}_{i_B j_B} \leq P \end{cases} \quad \text{avec } d \in S_D \text{ et } (i_B, j_B) \in R \text{ fixés}$$

Pour choisir l'arc et le dépôt du branchement, on construira le vecteur $\Delta^d = \left(\sum_p \alpha^{dp} x^{dp}_{ij} \right)_{(i,j)\in R}$ pour chaque $d \in S_D$. On trouvera ensuite quelle coordonnée de quel vecteur est la plus proche de 0.5. On obtiendra alors le dépôt d et l'arc $(i_B, j_B) := (i, j)$ utilisés pour le branchement. Il faudra tout de même vérifier de ne pas ajouter deux fois le même branchement dans une branche. Un branchement $b \in \mathcal{B}$ sera donc $b = \{(i_B, j_B), d\}, \quad (i_B, j_B) \in R, \quad d \in S_D$.

Lorsqu'on introduit les contraintes $\sum_p \alpha^{dp} x^{dp}_{i_B j_B} \geq 1$, on peut être amené à rentrer en conflit avec la contrainte $\sum_p \alpha^{dp} \leq P$. Dans ce cas, le problème maître devient infaisable et la tournée initiale sera incluse dans la solution, donc l'exploration de la branche se terminera.

4.4.4 Descente dans l'arbre et complexité de résolution

À chaque nœud, on rajoute un certain nombre de tournées à celles déjà existantes qu'on transmet aux nœuds fils. Le nombre de tournées augmente donc lorsqu'on descend dans l'arbre de recherche. Comme le nombre de variables du problème maître réduit correspond au nombre de tournées du nœud, les problèmes maîtres sont donc de plus en plus longs à résoudre. De plus, plus le graphe est grand, plus l'arbre de recherche sera profond. On ne peut néanmoins pas retirer des tournées d'un nœud car celle-ci pourrait être utile dans la solution optimale. Une bonne technique d'exploration de l'arbre serait donc d'explorer quelques fois en profondeur pour fixer de bonnes bornes supérieures au départ puis d'explorer en largeur, car la résolution des problèmes maîtres prend moins de temps si on est haut dans l'arbre. On pourrait également imaginer une heuristique qui nous permettrait de retirer une tournée si elle est inactive dans les solutions de la relaxation pour un certain nombre de nœuds, quitte à devoir la réintroduire plus tard dans l'arbre. Ceci permettrait de minimiser le nombre de variables de chaque problème maître réduit. Une autre méthode d'accélération de convergence pourrait également la fixation la borne supérieure initiale en résolvant le problème dans le cas d'un unique dépôt. En effet, en ajoutant des dépôts, on ne peut pas avoir un coût plus élevé que pour un seul dépôt.

4.5 Algorithme dérivé et solution approchée (BnP-approx)

En plus de l'algorithme de Branch-and-Price, on propose un algorithme plus rapide permettant d'obtenir une bonne approximation de la solution. Le principe est de commencer comme l'algorithme de Branch-and-Price, mais de s'arrêter après la résolution de la relaxation de la racine. On aura donc un certain nombre de tournées ajoutées dans la racine, qui ne sont potentiellement pas retenues dans la solution de la relaxation ($\alpha^{dp} = 0$). On prendra alors l'ensemble de ces tournées et on résoudra le P_{mr} en nombre entiers ($\alpha^{dp} \in \{0, 1\} \quad \forall T^{dp} \in \mathcal{T}$) pour les tournées ajoutées dans la racine. On peut

espérer qu'avec le petit nombre de tournées ajoutées, la résolution soit assez rapide. Le choix de cette méthode est motivé par le fait qu'on retrouve souvent des tournées de la solution optimale qui sont valorisés (α^{dp} proches de 1) dans la solution de la relaxation de la racine. Dès la première relaxation, on peut déjà identifier certaines bonnes tournées et on espère donc avoir une bonne solution en résolvant le PLNE directement pour la racine.

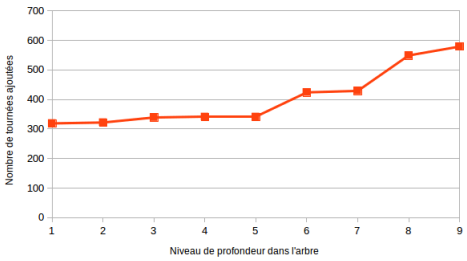

FIGURE 7 – Nombre de tournées dans les nœuds en fonction de leur profondeur dans l'abre de branchement sur un petit graphe

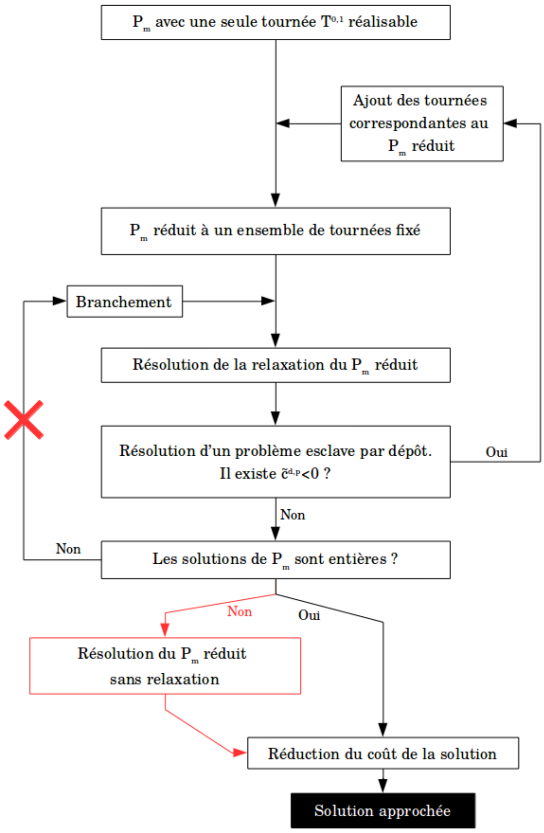

FIGURE 8 – Schéma de l'algorithme pour une solution approchée

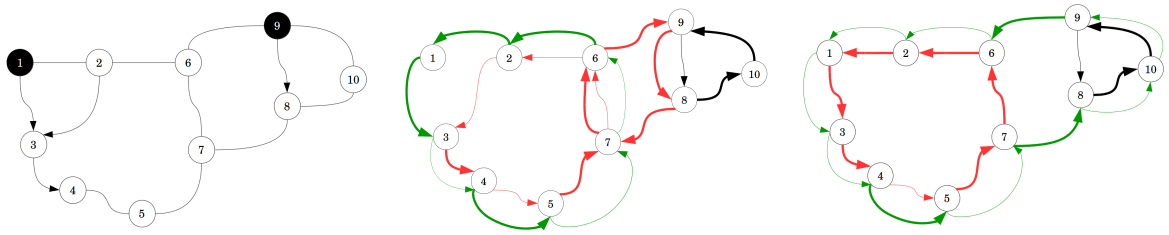

FIGURE 9 – Solution du Branch-and-Price (milieu) et solution de l'algorithme approché (droite) pour un petit jeu de données (gauche). On retrouve la tournée noire en commun. Les arcs servis sont en épais.

Comme on a une solution approchée, il se peut que plusieurs tournées servent le même arc. On rajoute donc une procédure à la fin de l'algorithme qui, si plusieurs tournées servent le même arc, assigne simplement la tournée le moins coûteuse au service de cet arc. Les autres tournées passeront simplement sur l'arc sans le servir et on pourra donc réduire un petit peu le coût global de la solution. Dans certains cas, la solution approchée requerra également plus de véhicules que la solution optimale.

5 Résultats

Trois différents types de jeux de données sont utilisés. Le premier (`"small-..."`) fournit 6 petits jeux sur lesquels les algorithmes peuvent être testés. Trois stations ont aussi étés modélisées sous forme de graphe (`devoluy-...`, `ceuze-...` et `greoliere-...`) avec plusieurs configurations différentes (unique/multi dépôt, totalité/partie des pistes ...). On utilise également des jeux de données plus classiques du CARP (`mval-...`, `lpr-...`) [21].

Dans le cas d'une résolution optimale pour un ou plusieurs dépôts, on notera c_{tot} le coût total de damage et t le temps de résolution en secondes. Pour les tests sur les jeux de données classiques de CARP (`mval-...` et `Lpr-...`) [21], on notera en gras le coût total si il correspond à la valeur optimale prouvée pour le jeu de données. Dans le cas d'une résolution approchées par l'algorithme de Branch-and-Price, on notera $\tilde{c}_{tot}$ le coût avant l'application de la procédure de réduction de coût et c_{tot} le coût après l'application de cette procédure. On notera également GAP l'erreur relative de la solution approchée par rapport à la solution optimale. Les jeux de données dont le temps de résolution trop élevé ne sont pas affichés dans les résultats.

Jeu de donnée	$\|S\|$	$\|E_R\|$	$\|E\|$	$\|A_R\|$	$\|A\|$	P	Algo	c_{tot}	t
small-undirected	7	10	10	0	0	1	CPP	38	0.028
small-undirected-rural	7	3	10	0	0	1	MCARP	22	0.027
small-directed	7	0	0	10	10	1	DCPP	62	0.001
small-directed-rural	7	0	0	6	10	1	MCARP	28	0.001
small-mixed	7	5	5	5	5	1	MCPP	47	0.057
small-mixed-rural	7	3	5	3	5	1	MCARP	28	0.003
devoluy-1	7	12	12	0	0	2	MCARP	60.5	0.033
devoluy-2	13	10	21	0	0	2	MCARP	67	0.068
devoluy-3	13	21	21	0	0	2	MCARP	106.5	0.111
ceuze-1	10	9	9	4	4	2	MCARP	59	0.029
ceuze-2	22	17	26	5	7	2	MCARP	122	0.051
ceuze-3	22	26	26	7	7	2	MCARP	177.5	0.199
greoliere-1	38	19	30	3	5	2	MCARP	148	0.068
greoliere-2	38	37	60	3	5	2	MCARP	239	0.218
greoliere-3	38	60	60	5	5	3	MCARP	384	15.613
mval-IF-3L-01A	24	20	20	35	35	4	MCARP	**230**	1.163
mval-IF-3L-01B	24	13	13	38	38	5	MCARP	**261**	2.583
mval-IF-3L-02A	24	16	16	28	28	4	MCARP	**324**	0.216
mval-IF-3L-02B	24	12	12	40	40	5	MCARP	**395**	6.634
mval-IF-3L-03A	24	15	15	33	33	4	MCARP	**115**	0.404
mval-IF-3L-03B	24	16	16	29	29	5	MCARP	**142**	3.641
mval-IF-3L-04A	41	26	26	69	69	5	MCARP	**580**	3.959
mval-IF-3L-05A	34	22	22	74	74	5	MCARP	**597**	13.253
mval-IF-3L-05C	34	17	17	81	81	7	MCARP	**697**	41.649
mval-IF-3L-06A	31	22	22	47	47	5	MCARP	**326**	4.234
mval-IF-3L-06B	31	22	22	44	44	6	MCARP	**317**	17.998
mval-IF-3L-07B	40	25	25	66	66	6	MCARP	**412**	3.708
mval-IF-3L-08A	30	20	20	76	76	5	MCARP	**581**	3.551
mval-IF-3L-08B	30	27	27	64	64	6	MCARP	**531**	18.248
mval-IF-3L-10A	50	32	32	106	106	5	MCARP	**634**	5.754
Lpr-IF-a-01	28	0	0	52	94	2	MCARP	**12884**	0.045
Lpr-IF-a-02	53	5	5	99	164	3	MCARP	**27152**	1.002
Lpr-IF-b-01	28	5	5	45	58	2	MCARP	**14235**	0.017
Lpr-IF-b-02	53	5	5	99	115	2	MCARP	**27754**	0.443
Lpr-IF-c-01	28	39	39	11	13	2	MCARP	**18039**	0.332

FIGURE 10 – Résolution pour un unique dépôt

Avec les différents tests sur les jeux de données, on peut observer que la résolution est beaucoup plus rapide dans le cas d'un unique dépôt. Il faut tout de même prendre en compte que le solver utilisé dans ce cas a été développé par des équipes d'experts, des années durant. L'algorithme de Branch-and-Price quant à lui a été développé en quelques semaines et n'est pas codé de telle manière à être optimisé. En effet, il y a de nombreux affichages au cours de l'algorithme qui ont servis à débugger et à comprendre ce qu'il se passait. Il y a donc une nette marge de progression quant à la vitesse de l'algorithme de Branch-and-Price. On peut noter que dans certains cas, la solution approchée obtenue

Jeu de donnée	$\|S\|$	$\|S_D\|$	$\|E_R\|$	$\|E\|$	$\|A_R\|$	$\|A\|$	P	Algo	c_{tot}	t
devoluy-1	8	2	12	12	0	0	2	BnP-opt	57	12.185
devoluy-2	13	2	10	21	0	0	2	BnP-opt	61	9.558
devoluy-3	13	2	21	21	0	0	3	BnP-opt	107.5	236.006
ceuze-1	10	2	9	9	4	4	2	BnP-opt	59	20.445
ceuze-2	22	2	17	26	5	7	2	BnP-opt	114	179.202
ceuze-3	22	2	26	26	7	7	2	BnP-opt	159.5	12684.146
greoliere-1	38	2	19	30	3	5	2	BnP-opt	148	154.679

FIGURE 11 – Cas avec plusieurs dépôts résolus par Branch-and-Price de manière optimale

Jeu de donnée	$\|S\|$	$\|S_D\|$	$\|E_R\|$	$\|E\|$	$\|A_R\|$	$\|A\|$	P	Algo	c_{tot}	$\tilde{c}_{tot}$	GAP	t
devoluy-1	8	2	12	12	0	0	2	BnP-approx	60.5	60.5	5.8%	5.651
devoluy-2	13	2	10	21	0	0	3	BnP-approx	61	61	**0%**	6.088
devoluy-3	13	2	21	21	0	0	3	BnP-approx	127.5	119.5	10.0%	12.171
ceuze-1	10	2	9	9	4	4	3	BnP-approx	64	62	4.7%	5.053
ceuze-2	22	2	17	26	5	7	2	BnP-approx	116	114	**0%**	26.314
ceuze-3	22	2	26	26	7	7	3	BnP-approx	182.5	169.5	5.9%	37.210
greoliere-1	38	2	19	30	3	5	3	BnP-approx	163	162	8.6%	18.767

FIGURE 12 – Cas avec plusieurs dépôts résolus par Branch-and-Price de manière approchée

par l'algorithme de Branch-and-Price modifié donne une assez bonne borne supérieure au GOP avec des temps de calcul très convenables. Parfois, on obtient même la solution exacte. Néanmoins, il est impossible de savoir quelle sera la qualité de cette approximation.

Bien que la résolution par solver soit assez rapide, elle est sensible à l'augmentation du nombre de véhicules. En effet, doubler le nombre de véhicules signifie doubler le nombre de variables et de contraintes. Dans le cas de la résolution par Branch-and-Price, l'ajout de véhicules signifie simplement la modification de certaines contraintes. Ajouter des véhicules pour le Branch-and-Price aura pour effet de moins contraindre le problème maître et donc potentiellement d'avoir une solution plus rapidement. Augmenter le nombre de dépôts signifie résoudre un problème esclave de plus dans le Branch-and-Price. Ces problèmes ne nécessitant pas beaucoup de temps de calcul, l'ajout d'un dépôt affectera peu le temps de résolution du Branch-and-Price.

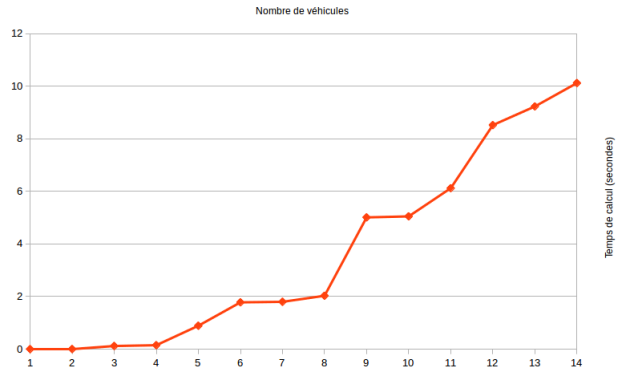

FIGURE 13 – Impact du nombre de véhicules pour ceuze-2 (dépôt unique, résolution par solver)

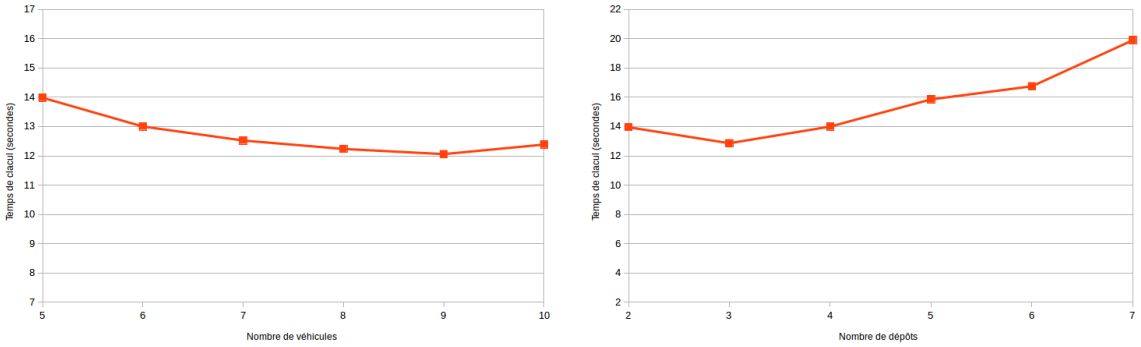

FIGURE 14 – Impact du nombre de véhicules (avec 2 dépôts, résolution par BnP-approx) et impact du nombre de dépôts (avec 3 véhicules, résolution par BnP-approx) pour devoluy-3

La plupart des stations de ski n'ont qu'un seul dépôt de dameuses. La résolution par la modélisation d'un PLNE puis la résolution par un solver permet d'avoir une solution exacte avec un temps de calcul acceptable. Lorsque le nombre de dameuses devient trop grand, il se peut que la résolution prenne trop de temps. Dans ce cas, on peut donc se tourner vers l'algorithme de Branch-and-Price qui fonctionnera également si on lui donne un seul dépôt en paramètre. Bien que celui-ci soit assez lent, un approfondissement dans l'écriture et l'optimisation du code pourrait s'avérer très intéressant, car cela permettrait d'avoir un algorithme donnant une solution optimale en un temps acceptable et qui serait peu sensible aux paramètres donnés en entrée. Cet algorithme serait donc bien adapté à la résolution du GOP dans tout type de situations.

6 Conclusion

Lors de ce stage de 8 semaines, j'ai été amené à résoudre un problème de damage dans une stations de ski. Les premières semaines ont été quasiment exclusivement consacrées aux recherches bibliographiques. Celle-ci ont également continuées tout au long du stage. Grâce à cela, j'ai pu trouver une modélisation adéquat au problème et justifiée par les différentes contraintes pratiques qui étaient imposées. J'ai aussi pu identifier des problèmes déjà existants qui ont aiguillé la modélisation de telle sorte à se rapprocher de ces problèmes. J'ai ensuite été amené à comprendre, mettre en place et implémenter plusieurs méthodes de résolution à base de programmes linéaires pour résoudre les différents types de problèmes liés au GOP. Pour cela, j'ai dû me renseigner sur les méthodes déjà existantes et les comprendre. Devoir coder l'algorithme de Branch-and-Price sans aide de packages extérieurs m'a aussi permis de bien comprendre chaque étape de l'algorithme ainsi que son fonctionnement. Cela a également été l'occasion d'apprendre un nouveau langage de programmation dédié principalement aux mathématiques. Le GOP m'a permis de découvrir un assez large panel de ce qui existe en terme de problème de routage, les méthodes de résolution développées et m'a également permis de me familiariser avec certains outils utilisés dans la recherche ou dans l'industrie comme par exemple le solver `CPLEX`. Le fait de travailler sur un problème concret et de lire des articles de recherche très récents a grandement participé à la conservation de ma motivation tout au long du stage. Ces deux mois de stage sont donc, pour moi, une très bonne première approche du monde de la recherche appliquée en mathématique.

7 Annexes

7.1 Jeux de données small

Sur les graphes suivantes, le nœud correspondant au dépôt est le cercle noir. Les pistes à damer sont représentées en rouge et les autres pistes en noir. Les pistes praticables dans les deux sens sont représentées par des traits et celles praticables dans un seul sens par des flèches.

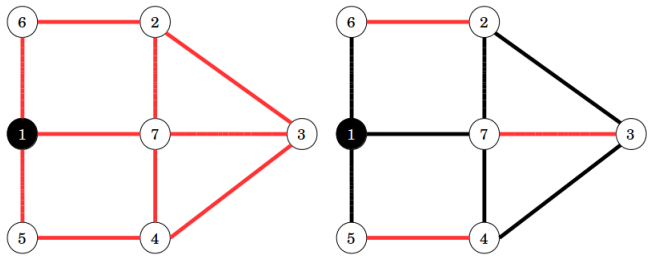

FIGURE 15 – small-undirected et small-undirected-rural

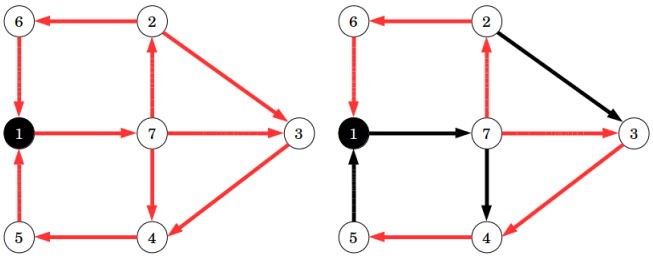

FIGURE 16 – small-directed et small-directed-rural

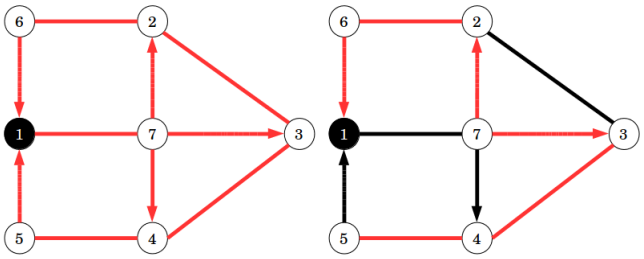

FIGURE 17 – small-mixed et small-mixed-rural

7.2 Modélisation des stations de ski

Pour les trois stations de ski étudiées, les pistes sont représentées avec la couleur qui correspond à leur difficulté afin de mieux se repérer. Les pistes requises ne sont pas repérées d'une manière particulière car ce ne sont pas les mêmes pour les trois jeux de données de chaque station. Les pistes

sont représentées par une flèche si elles sont praticables dans un seul sens et par un trait si elles le sont dans les deux sens. Le choix des pistes praticables dans un seul sens est arbitraire, il a été décidé que toutes les pistes noires (qui sont donc pentues) et certaines pistes rouges sont praticables uniquement en descendant à l'exception près de la station de Dévoluy qui est une station de ski de fond (les pentes ne sont donc pas importantes). Un unique dépôt est représenté. Le second dépôt est le sommet 13 pour la station du Dévoluy, le sommet 9 pour la station de Céüze et le sommet 16 pour la station de Gréolière.

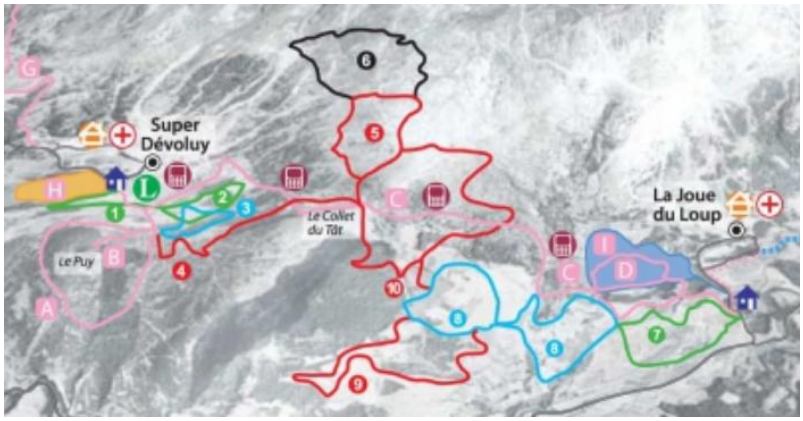

FIGURE 18 – Station de ski du fond du Dévoluy

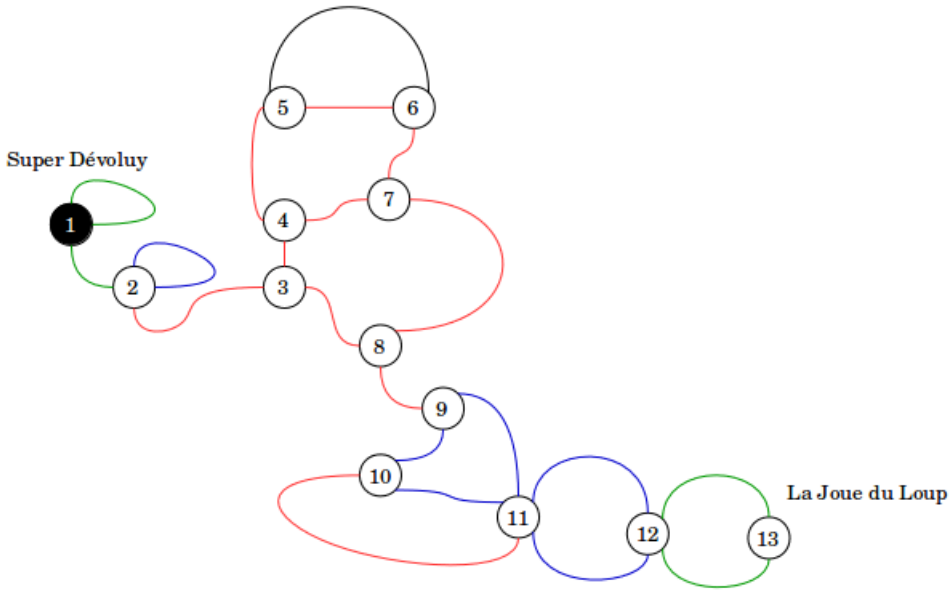

FIGURE 19 – Station de ski du fond du Dévoluy sous forme de graphe

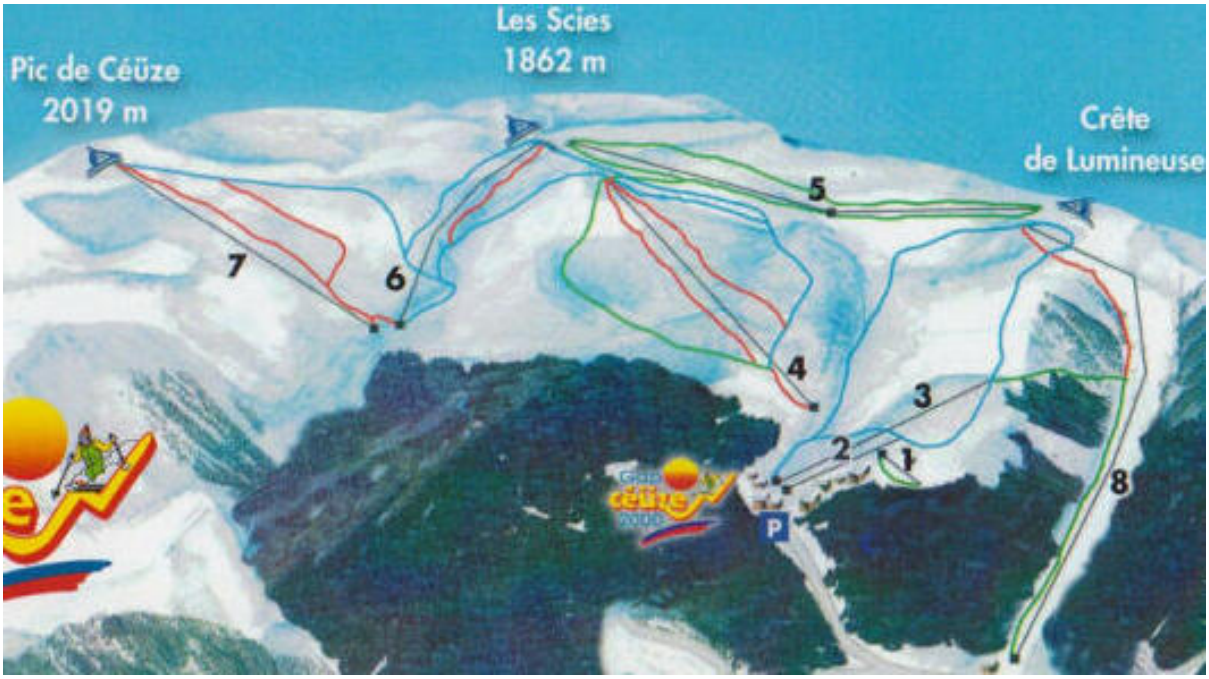

FIGURE 20 – Station de ski de Céüze

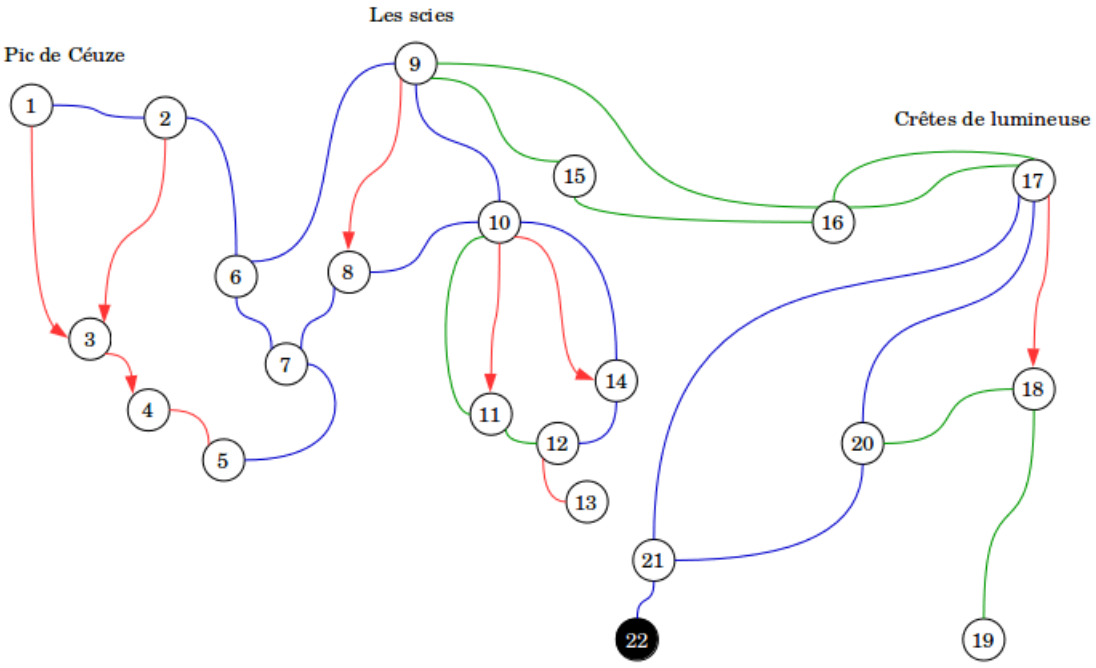

FIGURE 21 – Station de ski de Céüze sous forme de graphe

FIGURE 22 – Station de ski de Gréolière

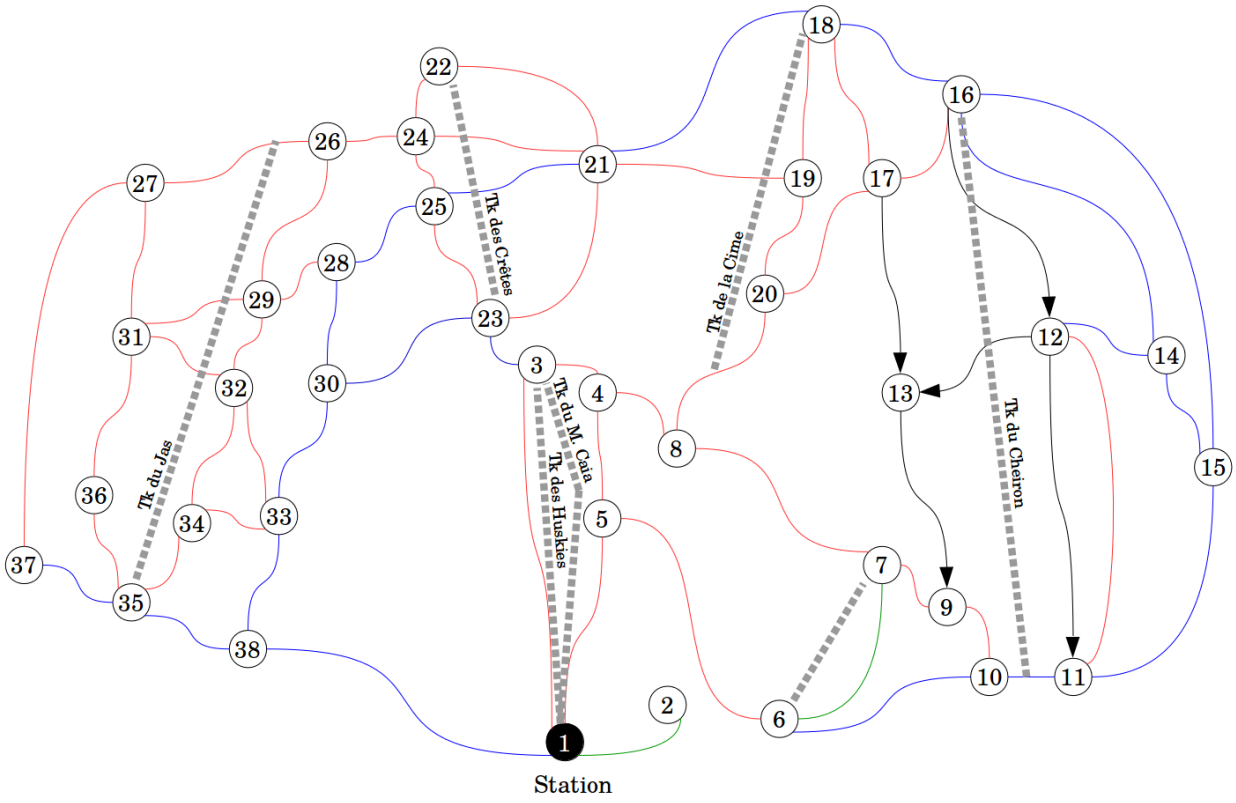

FIGURE 23 – Station de ski de Gréolière sous forme de graphe

Références

[1] M. Mittaz A. Hertz, G. Laporte. A tabu search heuristic for the capacitated arc routing problem. *Operations Research*, 48 :129–135, 2000.

[2] Anonymous. Eulerian path. https://en.wikipedia.org/wiki/Eulerian_path#Hierholzer's_algorithm, 2018.

[3] José M. Belenguer and Enrique Benavent. A cutting plane algorithm for the capacitated arc routing problem. *Computers and Operations Research*, 30(5) :705 – 728, 2003.

[4] Nicos Christofides, Vicente Campos, Angel Corberán, and E Mota. An algorithm for the rural postman problem. 1981.

[5] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 1965.

[6] Jack Edmonds and Ellis L. Johnson. Matching, euler tours and the chinese postman. *Mathematical Programming*, 5(1) :88–124, Dec 1973.

[7] L. Euler. *Solutio problematis ad geometriam situs pertinentis*. 1759.

[8] Les Foulds, Humberto Longo, and Jean Martins. A compact transformation of arc routing problems into node routing problems. *Annals of Operations Research*, 226(1) :177–200, Mar 2015.

[9] D.R. Fulkerson. An out-of-kilter method for solving minimal cost flow problems. *Journal of the Society for Industrial and Applied Mathemetics*, 1961.

[10] Bruce L. Golden and Richard T. Wong. Capacitated arc routing problems. *Networks*, 11(3) :305–315.

[11] Byung-In Kim, Seongbae Kim, and Surya Sahoo. Waste collection vehicle routing problem with time windows. *Computers and Operations Research*, 33(12) :3624 – 3642, 2006. Part Special Issue : Recent Algorithmic Advances for Arc Routing Problems.

[12] Vladimir Kolmogorov. Blossom v : A new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 2010.

[13] Dmitry Krushinsky and Tom Van Woensel. An approach to the asymmetric multi-depot capacitated arc routing problem. *European Journal of Operational Research*, 244(1) :100 – 109, 2015.

[14] L. Santiago Pinto L. Gouveia, M. Candida Mourao. Lower bounds for the mixed capacited arc routing problem. *Computers and Operations Research*, pages 692–699.

[15] Philippe Lacomme, Christian Prins, and Alain Tanguy. First competitive ant colony scheme for the carp. In Marco Dorigo, Mauro Birattari, Christian Blum, Luca Maria Gambardella, Francesco Mondada, and Thomas Stützle, editors, *Ant Colony Optimization and Swarm Intelligence*, pages 426–427, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[16] M. Minoux M. Gondran. *Graphes et algorithmes*, chapter 8, page 328. 1995.

[17] Rafael Martinelli, Diego Pecin, Marcus Poggi, and Humberto Longo. A branch-cut-and-price algorithm for the capacitated arc routing problem. In Panos M. Pardalos and Steffen Rebennack, editors, *Experimental Algorithms*, pages 315–326, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[18] Kwan Mei-Ko. Graphic programming using odd or even points. *Acta Mathematica Sinica (in Chinese)*, (10) :263–266.

[19] M. Cândida Mourão and Leonor S. Pinto. An updated annotated bibliography on arc routing problems. *Networks*, 70(3) :144–194.

[20] Wen Lea Pearn. Solvable cases of the k-person chinese postman problem. *Operations Research Letters*, 16(4) :241 – 244, 1994.

[21] Elias J. Willemse and Johan W. Joubert. Benchmark dataset for undirected and mixed capacitated arc routing problems under time restrictions with intermediate facilities. *Data in Brief*, 8 :972 – 977, 2016.

[22] Sanne Wøhlk. *A Decade of Capacitated Arc Routing*, pages 29–48. Springer US, Boston, MA, 2008.