Recherche Opérationnelle

Livrable 1 : Modélisation

Tuteur Projet : Maël Guiraud

Membres du groupe :

- Julien LAUDICINA
- Sonia REGHIS
- Claude AJAVON
- Clément FAUX

I. Présentation du projet

Le projet, porté par CesiCDP dans le cadre d'un appel à manifestation d'intérêt lancé par l'ADEME, vise à concevoir et expérimenter une solution de mobilité durable et intelligente pour le transport de marchandises. Notre équipe d'ingénieurs est déterminée et fera le nécessaire afin de proposer une solution convenable tout en respectant les contraintes imposées.

Notre priorité est de limiter autant que possible les déplacements et la consommation des véhicules qu'il s'agisse de véhicules électriques tout en prenant en compte les contraintes réelles : horaires, énergie, coûts.

Dans ce premier livrable, nous allons donc formaliser le problème pour répondre aux attentes de l'ADEME ; il s'agit de construire une représentation formelle du problème tout en intégrant ces contraintes et objectifs, afin de pouvoir ensuite l'optimiser.

II. Contraintes supplémentaires

Après analyse du problème avec l'équipe projet, nous avons ajouté deux contraintes supplémentaires :

- **Coût de certaines routes :** Certaines liaisons imposent un péage (Frais de traversée fluviale ou maritime, écotaxe poids lourd, zone à circulation restreinte ...) en échange d'un temps de trajet plus court.
- **Contraintes de recharge des camions électriques :** La tournée s'effectue avec des camions électriques qu'il faut recharger après un certain temps d'utilisation, pendant une pause de durée fixée. Or toutes les villes ne disposent pas de bornes de recharge ; le temps d'arrêt prévu dans chaque ville doit donc tenir compte de cette contrainte.

III. Rappel des classes de complexité

Les classes de complexité permettent de catégoriser des problèmes informatiques en fonction de leur niveau de difficulté, le temps et/ou l'espace qu'il faut afin de les résoudre. Il existe 4 classes de complexité, allant de la classe P qui est la moins complexe à la classe NP-Hard ou NP Difficile qui est la classe la plus complexe. Nous allons maintenant expliquer plus en détail chaque classe.

- **Classe P**

La classe P pour "Polynomial" contient tous les problèmes solvables en un temps polynomial, on peut donc les résoudre rapidement. D'un point de vue algorithme, ils sont faciles à faire.
Exemple : Trier une liste

- **Classe NP**

La classe NP pour "Non déterministe Polynomial" contient les problèmes dont la solution peut être vérifiée rapidement (en temps polynomial), même si on ne sait pas forcément la trouver rapidement.
Exemple : Problème du voyageur du commerce, sudoku

- **Classe NP-Complet**

Cette classe regroupe les problèmes les plus difficiles de la classe NP, elle représente la frontière entre la classe NP et la classe NP Hard. La frontière entre les problèmes qu'on peut vérifier vite et ceux qu'on ne peut pas forcément résoudre vite.

- **Classe NP-Hard**

Un problème appartenant à la classe NP-Hard est un problème qu'on ne peut pas forcément vérifier vite, mais qui est au minimum aussi dur que NP.
Exemple : Échecs

IV. Formulation formelle du projet

Formulation formelle du problème de base

Comme décrit précédemment, notre problème est de proposer une tournée permettant de relier entre elles un ensemble de villes, puis de revenir à son point de départ, de manière à minimiser la durée totale de la tournée.

L'enjeu de cette partie est donc de formuler formellement le problème à travers une représentation mathématique.

Tout d'abord, nous faisons le choix d'utiliser la théorie des graphes afin de représenter notre réseau routier. En effet, nous pourrons facilement représenter l'ensemble de nos villes par des sommets et l'ensemble des routes reliant les villes par des arêtes. De plus, chaque route étant à double sens, notre graphe sera non-orienté. Enfin, pour minimiser le temps de la tournée, il est sous-entendu que chaque route possède un temps de trajet spécifique. Pour cela, nous pondérons nos arêtes avec un nombre positif exprimant le temps de trajet entre deux villes.

Ainsi, nous utilisons un graphe **G** comportant un ensemble de sommets **V** et d'arêtes **E** tel que :

- $G = (V, E)$
- $V = \{v_0, v_1, ..., v_n\}, n \in N$
- $E = \{(v_i, v_j)\}, (i, j) \in N^2 \ et \ i \neq j$

De plus, pour chaque arête $(v_i, v_j) \in E$, a une pondération (un temps) $t_{ij} >= 0$, qui représente le temps nécessaire pour réaliser le trajet du sommet v_i au sommet v_j. Puisque notre graphe est non orienté, on considère $t_{ij} = t_{ji}$.

Nous définissons T_max, le temps maximal autorisé pour la tournée.

Afin de se rapprocher au plus de la réalité, nous instaurons un temps **T**, correspondant au temps constant passé par les camions de la tournée dans une ville. En effet, sans cette valeur, notre problème ne prendrait pas en compte les temps d'arrêt dans les villes ainsi, le temps total du trajet serait incomplet.

Contrainte d'autonomie du véhicule

Notre première contrainte est l'autonomie des véhicules. Nous faisons le choix que nos camions soient entièrement électriques. Ils ont donc un temps de trajet possible avant de devoir se recharger durant un certain temps T_r. Le véhicule a donc une valeur A représentant l'autonomie maximale du véhicule. De plus, nous devons assigner à certaines villes des bornes de recharges électriques. Cet ensemble de ville correspondra à R avec $R \subseteq V$. Afin d'identifier les villes appartenant à R, nous mettons en place un système d'attribution binaire à nos villes. Ainsi, les villes équipées d'une borne de recharge auront comme valeur 1 tandis que celles qui n'en n'ont pas à 0.

L'ajout de cette contrainte entraîne une condition supplémentaire à un cycle pour qu'il soit valide. De cette manière, à tout moment, la somme des temps de trajet effectués depuis la dernière recharge ne doit pas dépasser A.

Autrement dit, entre deux sommets appartenant à R, le véhicule peut effectuer un chemin $P = v_i, v_k, ..., v_j$ si et seulement si :

$$\sum_{(u,v) \in P} t_{uv} \leq A$$

Le véhicule commence sa course pleinement chargé et peut être rechargé uniquement en atteignant un sommet $v \in R$, ce qui réinitialise la somme des temps de trajet.

Contrainte de coût

Certaines arêtes du graphe représentent des liaisons payantes : tunnels, bacs, écotaxes, zones de congestion …
La contrainte impose que la somme des frais associés aux arêtes effectivement empruntées ne dépasse pas le budget autorisé.

Pour chaque arête payante (u, v) on note :

- $c_{uv} > 0$: montant du péage (en €).

L'ensemble des routes payantes est noté $P \subseteq E$.

Une fois l'itinéraire défini, le **coût financier global** des péages est

$$C_{\text{tour}} = \sum_{(u,v) \in P_{\text{parcouru}}} c_{uv}$$

où $P_{\text{parcouru}} \subseteq P$ est l'ensemble des liaisons payantes effectivement empruntées. De plus, C_{tour} doit être inférieur à $C_{\max}$, avec $C_{\max}$ le montant maximum autorisé pour le coût de la tournée.

Propriétés théoriques

Puisque nous cherchons un cycle passant par chaque sommet exactement une seule fois, nous remarquons que notre problème se rapproche de la définition d'un cycle hamiltonien, c'est-à-dire, une chaîne qui passe une fois et une seule par chaque sommet du graphe en formant un cycle. Ici, nous parlons de chaîne car nous plaçons notre problème dans un graphe non orienté. Dans un graphe orienté, nous parlerions de circuit.

Le problème du chemin hamiltonien est un problème de décision consistant à répondre par oui ou non à la question suivante : "Etant donné un graphe, admet-il un cycle hamiltonien ?". Ce problème a été prouvé NP-Complet par Richard Karp dans son article *Reducibility Among Combinatorial Problems* en 1972. Comme décrit plus tôt, cette propriété implique qu'on peut vérifier une solution en un temps polynomial nonobstant, résoudre le problème prendra certainement un temps déraisonnable.

En plus de devoir chercher un cycle hamiltonien, nous cherchons également à minimiser la durée totale de la tournée. A travers cette nouvelle contrainte, nous instaurons un problème d'optimisation. Nous souhaitons minimiser la somme des poids des arêtes.

Considérons, un cycle hamiltonien C, une séquence de villes qui commence et se termine à la même ville, en passant par chaque autre ville une seule fois. On peut le noter comme suit :

$C = (v_1, v_2, \ldots, v_n, v_1)$

Le temps total de ce cycle hamiltonien C est la somme des temps de trajet entre les villes successives dans le cycle :

$\text{Temps}(C) = w_{1,2} + w_{2,3} + \cdots + w_{n-1,n} + w_{n,1} = \sum_{k=1}^{n-1} w_{k,k+1} + w_{n,1}$

Notre objectif est de trouver le cycle hamiltonien T^* qui a le temps total le plus petit possible. On peut le noter ainsi :

$T^* = argmin_{C \in H}\{Temps(C)\}$, avec H représentant l'ensemble de tous les cycles hamiltoniens possibles dans le graphe.

Désormais,le problème est à la fois de trouver un chemin hamiltonien mais également de minimiser la somme des poids. En ajoutant la contrainte d'optimisation, on en déduit que **notre problème est au moins aussi difficile que de trouver un cycle hamiltonien.**

On remarque également que ce premier problème se rapproche du problème du voyageur de commerce. En effet, ce dernier est un problème d'optimisation qui consiste à déterminer, étant donné un ensemble de villes, le plus court cycle passant par chaque ville une seule fois.

Le problème du voyageur, ou *Traveling salesman problem (TSP)* est NP-complet car pour un ensemble de n points, il existe au total $n!$ chemins possibles. Le point de départ ne change pas la longueur du chemin et les chemins peuvent être parcourus dans les deux sens, on obtient alors $\frac{(n-1)!}{2}$ chemins possibles. Cette explosion combinatoire devient alors impossible à résoudre dans un temps raisonnable.
De ce fait, une approche de résolution naïve en parcourant tous les chemins possibles est irréalisable dans notre cas à cause du temps que ca prendrait. A la place, nous utiliserons des algorithmes heuristiques. Ceux-ci seront abordés plus loin.

V. Problème de décision

Existe-t-il un cycle hamiltonien C = (v_1, v_2, ..., v_n, v_1) dans G avec $Temps(C) <= T_{lmax}$ et avec $C_{tour} <= C_{max}$ tel que :

Entre deux sommets consécutifs du cycle où le véhicule se recharge (sommets de R), la somme des temps de trajet ne dépasse pas l'autonomie Q :

$\forall (v_a, v_{a+1}, \ldots, v_b)$ sous-chemin de C tel que $r(v_a) = r(v_b) = 1$ et $r(v_j) = 0$ pour $a < j < b$: $\sum_{j=a}^{b-1} c_{v_j v_{j+1}} \leq A$

VI. Problème d'optimisation

Quelle est la plus petite valeur de $Temps(C)$ et la plus petite valeur de C_{tour} pour un cycle hamiltonien C = (v_1, v_2, ..., v_n, v_1) dans G ?

VII. Preuve de la NP-complétude de notre problème

Supposons que le problème temps est NP-Complet. Il faut prouver que notre problème l'est aussi. Pour cela, il faut montrer que notre problème est dans NP (donc que la validité d'une solution peut être vérifiée en temps polynomial), et qu'il est NP-Difficile (au moins aussi difficile que n'importe quel problème de NP). Puisqu'il est parmi les problèmes les plus durs de NP, il est NP-Complet.

1. Formalisation précise du problème de décision

En reprenant la définition formelle de notre problème ainsi que les contraintes, nous l'exprimons de la manière suivante :

- $G = (V, E)$, un graphe non orienté
- $V = \{v_0, v_1, \ldots, v_n\}, n \in N$, l'ensemble de nos sommets (villes)
- $E = \{(v_i, v_j)\}, (i, j) \in N^2 \ et \ i \neq j$, l'ensemble des arêtes du graphe (routes)
- $\forall (v_i, v_j) \in E, \exists t_{ij} \geqslant 0$ représentant le temps nécessaire pour réaliser le trajet du sommet v_i au sommet v_j.
- $\forall (v_i, v_j) \in E, \exists c_{ij} \geqslant 0$ représentant le coût pour réaliser le trajet du sommet v_i au sommet v_j.
- $R \subseteq V$: sous-ensemble de sommets équipés de bornes de recharge
- $A \in \mathbb{N}$: autonomie maximale des véhicules (en unités de temps/distance)
- $K \in \mathbb{N}$: constante représentant le coût/temps maximal acceptable pour le cycle

Pour tout sommet $v \in V$, nous définissons une fonction binaire $r(v)$ telle que :

$r(v) = 1$ si $v \in R$ (ville équipée d'une borne de recharge)
$r(v) = 0$ sinon

2. Prouver que le problème est dans NP

Afin de prouver que notre problème est dans NP, nous devons montrer qu'il existe un algorithme qui vérifie en temps polynomial si un certificat donné est une solution valide.

Problème de décision : Existe-t-il un cycle hamiltonien C = (v_1, v_2, ..., v_n, v_1) dans G tel que :

La coût total du parcours est inférieure ou égale à K : $\sum_{i=1}^{n} t_{v_i v_{i+1}} \leq K \text{ (avec } v_{n+1} = v_1)$

Entre deux sommets consécutifs du cycle où le véhicule se recharge (sommets de R), la somme des temps de trajet ne dépasse pas l'autonomie Q :$\forall (v_a, v_{a+1}, ..., v_b) \text{ sous-chemin de } C \text{ tel que } r(v_a) = r(v_b) = 1 \text{ et } r(v_j) = 0 \text{ pour } a < j < b$: $\sum_{j=a}^{b-1} t_{v_j v_{j+1}} \leq A$

Certificat :

Notre donnée d'entrée est un cycle $C = (v_0, v_1, ..., v_n)$, représentant une séquence de sommets à visiter. Considérons alors cet algorithme :

1. **Vérifier si** $u_n = u_0$ (retour au sommet initial)
   - Cette vérification peut être spécifiée en $O(1)$
2. **Vérifier que chaque sommet n'apparaisse qu'une unique fois**
3. **Vérifier que pour chaque paire consécutive** (v_i, v_{i+1}) **dans** C**, l'arête** $(v_i, v_{i+1}) \in E$
   - Mathématiquement : $\forall i \in \{1, 2, ..., n\}$, $(v_i, v_{i+1}) \in E$ (avec $v_{n+1} = v_1$)
   - Cette vérification peut être réalisée en $O(n)$ où n est le nombre de sommets
4. **Vérifier si le coût total du parcours est** $\leq K$:
   - Calculer la somme $S = \sum_{i=1}^{n} t_{v_i, v_{i+1}}$
   - Vérifier que $S \leq K$
   - Mathématiquement : $\sum_{i=1}^{n} t_{v_i, v_{i+1}} \leq K$ (avec $v_{n+1} = v_1$)
   - Cette vérification peut être réalisée en $O(n)$
5. **Vérifier le respect de la contrainte d'autonomie** :
   - Parcourir le cycle C et, pour chaque segment entre deux sommets consécutifs équipés de bornes de recharge, calculer la somme des temps de trajet
   - Vérifier que cette somme ne dépasse pas A pour chaque segment
   - Mathématiquement : $\forall (v_a, v_{a+1}, ..., v_b)$ sous-chemin de C tel que $r(v_a) = r(v_b) = 1$ et $r(v_j) = 0$ pour $a < j < b$: $\sum_{j=a}^{b-1} t_{v_j, v_{j+1}} \leq A$
   - Cette vérification peut être réalisée en $O(n)$

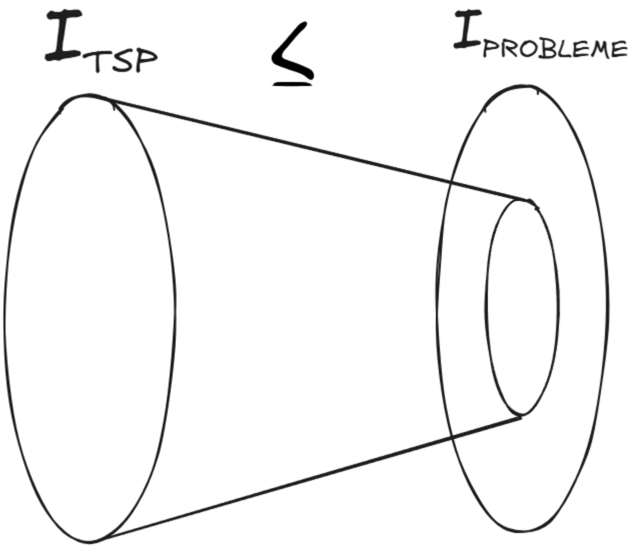
Puisque toutes les étapes de vérification s'exécutent en temps polynomial ($O(n)$), notre problème est bien dans NP : étant donné un certificat, nous pouvons vérifier en temps polynomial s'il constitue une solution valide à notre problème.

3. Prouver que le problème est NP-difficile

Afin de prouver que notre problème est NP-difficile, nous allons établir une réduction polynomiale depuis le problème du voyageur de commerce (TSP), qui est connu pour être NP-complet.

Rappel du problème TSP standard

Le problème du voyageur de commerce consiste à déterminer, dans un graphe complet pondéré $G = (V, E)$, s'il existe un cycle hamiltonien dont la somme des poids est inférieure ou égale à une valeur donnée K.

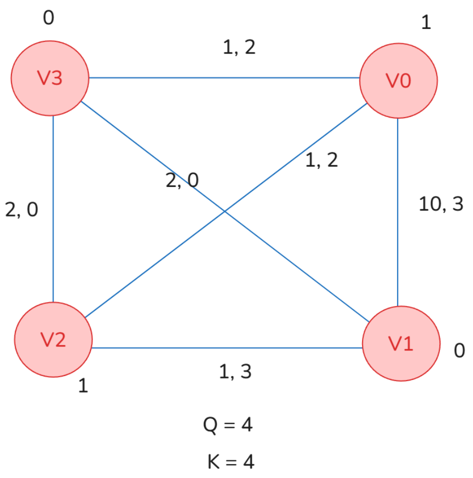


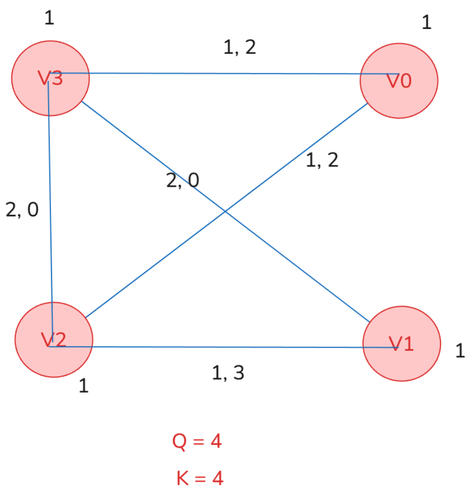
Réduction polynomiale

Soit $I\_TSP = (G, K)$ une instance du problème TSP avec $G = (V, E)$ un graphe complet pondéré et K la borne supérieure du coût.

Nous allons construire une instance $I' = (G', R, A', K')$ de notre problème comme suit :

- $G' = (V', E')$ avec $V' = V$ et $E' = E \backslash \forall (v_i, v_j), c\_i, j > K', t\_ij > A'$ (dans l'instance $I\_probleme$, on supprime les arêtes dont leur temps t_ij est superieur à A et les arêtes dont le coût c_i, j est supérieur à K)
- $R = V$ (toutes les villes sont équipées de bornes de recharge)
- $A' = A$ (même autonomie)
- $K' = K$ (même borne de coût)





4. Preuve de l'équivalence des solutions

1. Hypothèses

Supposons qu'une solution existe pour notre problème $I_{problème}$. Cela signifie qu'il existe un cycle hamiltonien C dans $G' = (V, E')$, avec les conditions suivantes :

- Le coût total du cycle C, soit $C_{tour} = \sum_{(c_i, c_j) \in C} c_{ij}$, est inférieur ou égal à K'.
- Le temps de chaque trajet t_{ij} dans le cycle est inférieur ou égal à A'.
- Si un trajet entre deux villes dépasse A', le véhicule se recharge dans une ville de recharge, ce qui est possible puisque toutes les villes sont équipées de bornes de recharge.

Nous devons maintenant prouver que ce cycle C est également une solution valide pour le problème TSP standard I_{TSP}.

2. Preuve de l'équivalence des solutions

2.1. Coût du cycle

Dans $I_{problème}$, on doit s'assurer que le total des "poids" (qui représentent le coût) des chemins empruntés dans le cycle ne dépasse pas K'. Comme K' est la même chose que K (c'est ce qu'on a décidé avant), cela veut dire que le coût total du cycle dans $I_{problème}$ est bien en dessous ou égal à K, qui est la règle à suivre dans le problème TSP standard (I_{TSP}).

Donc, le chemin qu'on a trouvé dans $I_{problème}$ respecte déjà la règle sur le coût, et c'est pour ça que ce chemin est aussi bon pour I_{TSP}.

2.2. Règle d'autonomie

Dans $I_{problème}$, le chemin C respecte la règle d'autonomie A'. Ça veut dire que pour chaque trajet entre deux villes (v_i et v_j), le temps de voyage t_{ij} est toujours plus petit ou égal à A'. Donc, tous les trajets dans C suivent cette règle.

Maintenant, dans I_{TSP}, il n'y a pas de règle d'autonomie comme ça. Donc, cette condition de $I_{problème}$ ne pose aucun souci pour I_{TSP}. Le chemin C respecte simplement la règle du coût (qui est la seule chose qui compte dans I_{TSP}).

2.3. Recharge dans les villes

Si le chemin dans $I_{problème}$ passe par un trajet où le temps t_{ij} est plus grand que A', alors la voiture doit se recharger dans une ville où il y a une borne. Ça ne change pas le coût total du chemin, car la recharge se fait dans une ville où c'est permis par les règles de $I_{problème}$ (toutes les villes ont des bornes).

Par contre, dans le problème I_{TSP}, il n'y a pas de règle d'autonomie. Donc, le chemin peut simplement être vu comme un chemin qui passe par toutes les villes une seule fois et dont le coût total est respecté. Les recharges qui se font dans $I_{problème}$ n'ont pas d'effet sur le chemin dans I_{TSP}, car ces recharges n'ajoutent pas de coût supplémentaire pour le problème TSP.

2.4. Conclusion sur l'équivalence

Comme le chemin C dans $I_{problème}$ respecte :

- la règle du coût (qui est la même que $K' = K$),
- la règle d'autonomie (qui n'est pas une règle dans I_{TSP}),
- et peut faire des recharges sans changer le coût total (parce que toutes les villes ont des bornes de recharge),

alors le chemin C est aussi une bonne solution pour I_{TSP}.

3. Conclusion

Si on trouve une solution pour notre problème $I_{problème}$ (avec les règles d'autonomie et de bornes de recharge), alors on a aussi une bonne solution pour le problème I_{TSP} (le problème du voyageur de commerce normal). C'est parce que :

- Le coût du chemin respecte la limite K.
- Les règles d'autonomie n'ont pas d'importance pour le problème I_{TSP}.
- Les recharges ne changent pas le coût total du chemin.

Donc, si on résout $I_{problème}$, on a automatiquement une solution pour I_{TSP}, ce qui montre que les solutions des deux problèmes sont liées.

Annexe

1. Bibliographie

- Notre site pour le projet : https://ro.student-project.space/
- Cycle Hamiltonien : https://fr.wikipedia.org/wiki/Graphe_hamiltonien
- Classes de complexité : https://fr.wikipedia.org/wiki/Probl%C3%A8me_NP-complet
- TSP : https://fr.wikipedia.org/wiki/Probl%C3%A8me_du_voyageur_de_commerce

2. Exemples d'applications selon les contraintes choisies