M2 Androide - Sorbonne Université

Module MAOA

Projet 2025-2026

Traveling Thief Problem (TTP)

On s'intéresse à une version plus concrète du célèbre problème du voyageur de commerce, Traveling Salesman Problem (TSP). En effet, le TSP peut-être vu comme un problème assez éloigné du réel si la personne, le robot ou la personne qui effectue un tour sur une carte le fait avec l'objectif de collecter des objets: ce qui induit qu'on doit prendre en compte un poids maximal qui est transporté, ainsi que le fait que la vitesse dépend du poids transporté. Une telle problématique intervient dans plusieurs contextes: celui de la collecte par un véhicule ou un robot: ramassage scolaire, collecte d'ordure, bras robot en usine...

1 Formalisation de deux problèmes

On va considérer deux versions du problème de voyageur de commerce dépendant du temps et du poids, Time and Weight dependent TSP. Ces deux versions consistent à déterminer un tour passant par toutes les villes et la décision de prendre ou non un objet dans chaque ville. En voici les données et les caractéristiques communes.

Input

- un ensemble $N = \{1, \ldots, n\}$ villes avec les distances $d_{i,j}$ entre chaque paire ij de villes
- une ville $i \in \{1, \ldots, n\}$ contient un ensemble $M_i = \{1, \ldots, m_i\}$ d'items. Chaque item $k \in M_i$ a un profit p_{ik} en euro et un poids w_{ik} en kilo.
- un véhicule pouvant transporter au plus un poids W

Output

- un cycle hamiltonien $(t_1, \ldots, t_n)$, appelé tour, i.e. un cycle passant exactement une fois pas ville
- un plan de collecte (packing plan) encodé par la variable binaire y_{ik} indiquant si l'item i et collecté dans la ville i.
- sous la contrainte de sac-à-dos que le véhicule ne peut pas transporter plus de C kilos, i.e. $\sum_{i=1}^{n} \sum_{k=1}^{m_i} w_{ik} \leq W$.

Il est à noter que pour beaucoup d'instances du problème, il n'y a souvent qu'un seul item par ville, i.e. $m_i = 1$, $i \in \{1, \ldots, n\}$.

1.1 Cumulative TSP avec knapsack constraint

Une première version du Time and Weigth dependent problem est parfois appelé *Cumulative TSP with knapsack constraint* (KCTSP). Il a une donnée supplémentaire.

Input

- K_W coût unitaire par kilomètre du transport d'un kilogramme transporté par le véhicule.

Ainsi pour cette variante du problème, la fonction objective est linéaire.
Objective function

- Le poids transporté à la ville i est alors $W_i = \sum_{k=1}^{i} \sum_{j=1}^{m_i} w_{t_k j} y_{t_k j}$.
- Max $\sum_{i=1}^{n} \sum_{j=1}^{m_i} p_{ij} y_{ij} - K_W \left(d_{t_n t_1} W_n + \sum_{i=1}^{n-1} d_{t_i t_{i+1}} W_i \right)$

Etat de l'art: Ce problème ne semble pas être présent dans la littérature. Un version sans contrainte de sac-à-dos existe dans la littérature sous différents noms. Comme la fonction objective est linéaire, il est possible d'utiliser des outils de la PLNE, e.g. [2, 4, 3, 1].

1.2 Traveling Thief

Une deuxième version du Time and Weigth dependent problem est très clairement nommée *Traveling Thief problem* (TTP). Il a deux données supplémentaires.
Input

- v_{min} (resp. v_{max}) vitesse maximale (resp. minimale) à laquelle peut aller le véhicule.
- K_d coût unitaire de transport par seconde où le véhicule est utilisé.

Objective function

- La vitesse du véhicule dépend du poids transporté. Pour écrire cette vitesse, on pose une constante $v = \frac{v_{max} - v_{min}}{W}$. La vitesse pour le trajet entre t_i et t_{i+1} est alors $v_{max} - W_i$. En se rappelant que le temps est la distance sur la vitesse, on obtient la fonction objective non-linéaire suivante:
- Max $\sum_{i=1}^{n} \sum_{j=1}^{m_i} p_{t_i j} y_{t_i j} - K_d \left(\frac{d_{t_n t_1}}{v_{max} - v W_n} + \sum_{i=1}^{n-1} \frac{d_{t_i t_{i+1}}}{v_{max} - v W_i} \right)$

Etat de l'art: Cette fonction non-linéaire semble être non-linéarisable compte-tenu de l'absence de résultat à ce propos dans la littérature.

2 Manipulation des instances

Pour l'étude du TTP, nous utilisons en priorité le jeu de benchmarks publié lors de la compétition CEC 2014 — « Instances and Code » — hébergé par le groupe OptLog de l'université d'Adelaide[1].

2.1 Jeux d'instances officiels

Le site fournit un ensemble complet d'instances combinant un problème TSP (villes + distances euclidiennes) et un problème de sac à dos (items dans chaque ville, poids, profit).

Quelques caractéristiques des instances disponibles :

- Des instances de petite taille (280 villes), de taille moyenne (4 461 villes), jusqu'à des instances très grandes (33 810 villes)
- Plusieurs variantes : 1 item par ville, 5 items par ville, 10 items par ville — avec différentes corrélations poids/profit

[1] `https://cs.adelaide.edu.au/~optlog/CEC2014Comp/`

- Pour chaque instance, les données comprennent :
  - coordonnées des villes (TSP)
  - liste des items par ville (poids, profit)
  - capacité du sac, vitesse maximale/minimale, paramètres de location (renting rate) selon la variante

Ces instances permettent de tester vos heuristiques, vos formulations PLNE, et de comparer vos résultats avec la littérature existante.

2.2 Recommandations d'usage

- Commencer les tests avec des instances officielles de petite taille — pour débogage, validation des fonctions, compréhension du comportement
- Passer ensuite aux instances officielles de taille moyenne ou élevée — pour évaluer la performance, les temps, la scalabilité
- Utiliser des nstances générées (par vos soins si vous voulez) pour explorer l'impact des paramètres (capacité, distribution poids/profit, corrélation, nombre d'items)
- Documenter clairement dans votre rapport la source de chaque instance utilisée, les transformations éventuelles, et les paramètres choisis

3 Travail demandé pour ce projet

L'idée générale est la suivante:
"Etudier les deux variantes en utlisant plusieurs méthodes de la littérature. Approfondissez au moins une des méthodes avec un investissement s'appuyant sur la littérature."

Pour ce projet, **il est demandé:**

- une visualisation des données et des solutions pour les deux versions
- une heuristique simple (gloutonne) pour les deux problèmes
- une heuristique avancée issue de la littérature avec paramétrage automatique des paramètres pour les deux problèmes
- une formulation compacte PLNE pour la version KCTSP résolu par un outil comme Gurobi
- un approfondissement d'une des méthodes évoquées pour l'une ou les deux variantes
- une comparaison des résultats obtenus pour les 3 méthodes et les 2 problèmes: qualité du résultat, temps de calcul et tailles résolues; avec une testr des méthodes sur un lot suffisant d'instances de tailles croissantes: déterminant ainsi les tailles maximales pouvant être résolues versus la qualité des solutions.
- un rendu sous forme d'un mini-rapport décrivant vos idées, vos expérimentations (tableaux, courbes) avec une analyse critique
- un rendu des archives de votre code
- une mini-soutenance en janvier pour présenter vos idées et résultats (quelques slides issus de votre rapport pendant 5 à 10 min).

L'évaluation de ce projet sera sur plusieurs bases d'évaluation:
- la quantité de travail fait dans la liste ci-dessus et en particuliers le travail effectué sur la méthode que vous avez approfondie.
- la qualité des solutions obtenues
- la taille des instances résolues
- mais aussi l'originalité de la méthode proposée, l'investissement en lecture d'articles etc.

Il est à noter qu'il n'y a pas de méthodes imposées dans ce projet, car il s'agit d'une question de recherche: en effet, les différents critères à mesurer demande de tester différentes méthodes pour savoir lesquelles amènent à de bons résultats. Par exemple, il peut être également intéressante de se demander si les deux critères qui différencient les deux méthodes amènent à des solution réllement différentes.
Ainsi, un objectif plus global de l'ensemble des projets réalisés par les différents monômes et binômes est de répondre de différentes façons à ces deux variantes: C'est pourquoi une démarche originale (mais basée sur une réflexion construite) sera valorisée dans ce projet.

References

[1] Abeledo, H., Fukasawa, R., Pessoa, A., Uchoa, E. The Time Dependent Traveling Salesman Problem: Polyhedra and Branch-Cut-and-Price Algorithm. *Experimental Algorithms. SEA 2010. LNCS*, 6049 (2010).

[2] Jean-Claude Picard, Maurice Queyranne. The Time-Dependent Traveling Salesman Problem and Its Application to the Tardiness Problem in One-Machine Scheduling. *Operations Research*, 26-1, pp. 86-110 (1978).

[3] Jean-François Cordeau, Gianpaolo Ghiani, Emanuela Guerriero. Analysis and Branch-and-Cut Algorithm for the Time-Dependent Travelling Salesman Problem *Transportation Science*, 48-1 pp. 46-58 (2014).

[4] Time Dependent Vehicle Routing Problems: Formulations, Properties and Heuristic Algorithms. Chryssi Malandraki, Mark S. Daskin. *Transportation Science*, 26-3, pp. 185-200 (1992)