

INFO-F310 - ALGORITHMIQUE ET RECHERCHE OPÉRATIONNELLE

Dimitrios Papadimitriou
dimitrios.papadimitriou@ulb.be

Hugo Callebaut
hugo.callebaut@ulb.be

1 Problème

Le problème que l'on vous demande d'analyser est celui du transport multi-articles avec nœuds intermédiaires. Son énoncé étend le problème de transport que vous avez vu en cours.

Le problème de transport de base considère le déplacement de marchandises entre m sources (producteurs) et n destinations (consommateurs) ($m, n \in \mathbb{N}_0$), avec pour objectif de minimiser le coût de transport. Dans ce cadre, il n'y a qu'un seul type d'article et chaque source est directement connectée à une ou plusieurs destinations.

Dans le problème de transport multi-articles avec nœuds intermédiaires, la complexité est accrue. Ce problème consiste à minimiser le coût de transport de plusieurs types d'articles indicés par l'entier $k \geq 1$ depuis m_k sources (producteurs) vers n_k destinations (consommateur) ($m_k, n_k \in \mathbb{N}_0$), avec potentiellement la présence d'un ou plusieurs nœuds intermédiaires qui agissent chacun comme point de transfert entre les sources et les destinations. Les sources ne sont donc plus nécessairement connectées directement à chaque destination et il peut y avoir plusieurs chemins d'une source vers une destination.

L'objectif de ce projet est d'explorer ces différences et de développer des méthodes efficaces pour résoudre ce problème de transport multi-articles avec nœuds intermédiaires, et ce en tenant compte de toutes les contraintes et objectif spécifiques à ce contexte.

De plus, nous vous demandons de comparer analytiquement deux approches différentes pour résoudre le problème de transport multi-articles avec nœuds intermédiaires : l'approche agrégée et l'approche désagrégée.

- L'approche agrégée consiste à considérer les différents types d'articles comme une seule entité. Cela correspond donc au problème de transport avec nœuds intermédiaires sans la partie multi-articles. Pour la formulation de ce problème, il vous est demandé de sommer la capacité des sources sur chaque article pour obtenir une unique capacité par source. De même, pour obtenir une seule demande par destination vous devrez sommer les demandes sur chaque article. Finalement, pour le coût de transport par arc, prenez le médian des coûts de chaque objet sur chaque arc.
- L'approche désagrégée prend en compte les caractéristiques spécifiques de chaque type d'article lors de la planification de leur transport. Cela correspond donc exactement au problème tel que spécifié initialement.

2 Instances

Un répertoire contenant des instances `n_i_instance.txt` du problème vous est fourni. Les fichiers sont au format suivant (le template est disponible à la Figure 1) :

- `n` et `i` dans le nom du fichier correspondent respectivement au nombre de sommets et d'objets dans le problème.
- Les instances sont générées de sorte à avoir un degré moyen par nœud proche de 6.
- Chaque ligne dans la section **NODES** correspond à un nœud (ceci comprend les sources, destinations et nœuds intermédiaires), les IDs sont toujours numérotés de 0 à `n` dans l'ordre dans lequel ils apparaissent.
- Chaque ligne dans la section **EDGES** correspond à un arc entre deux nœuds, ces arcs sont dirigés, le coût correspond au coût unitaire de passage de l'objet en question sur cet arc. `id_start_node` et `id_end_node` correspondent respectivement aux nœuds de départ et d'arrivée de cet arc.

```

ITEMS nb_items

NODES nb_nodes
ID x y
id_node_1 x_pos y_pos
id_node_2 x_pos y_pos
...
EDGES nb_edges
ID START END COST_ITEM_0 ... COST_ITEM_I
id_edge_1 id_start_node id_end_node cost_item_0 ... cost_item_i
id_edge_2 id_start_node id_end_node cost_item_0 ... cost_item_i
...
SOURCES nb_sources
ID CAPACITY_ITEM_0 ... CAPACITY_ITEM_I
id_source_node_1 capacity_item_0 ... capacity_item_i
id_source_node_2 capacity_item_0 ... capacity_item_i
...
DESTINATIONS nb_destinations
ID DEMAND_ITEM_0 ... DEMAND_ITEM_I
id_destination_node_1 demand_item_0 ... demand_item_i
id_destination_node_2 demand_item_0 ... demand_item_i
...

```

FIGURE 1 – Template des fichiers d'instance

- Chaque ligne dans la section **SOURCES** correspond à une source qui produit certains objets, l'id de la source correspond au nœud avec le même id dans la section **NODES**.
- La section **DESTINATIONS** est similaire à la section **SOURCES**, les lignes correspondent ici à une destination qui a une certaine demande pour un ou plusieurs objets.
- Il y a une ligne vide qui sépare chaque section

Notez que les arêtes sont dirigées (**start_node** vers **end_node**) et qu'il n'y a aucune contrainte sur celles-ci. Toute arête a un coût unitaire par objet (**COST_ITEM_k**) qui peut être négatif. Une arête peut aller directement d'une source (producteur) vers une destination (consommateur) sans passer par un nœud intermédiaire. Il peut y avoir des arêtes entre producteurs (**start_node** et **end_node** sont des noeuds source), entre consommateurs (**start_node** et **end_node** sont des noeuds destination) et entre noeuds intermédiaires. Une paire donnée de noeuds peut être reliée par plusieurs arêtes parallèles. Les arêtes n'ont pas d'attribut de capacité.

Pour vous faciliter la tâche, vous pouvez partir du principe qu'un nœud sera producteur, consommateur ou aucun des deux (nœud intermédiaire) mais jamais les deux en même temps. Tous les fichiers d'instance fournis respectent cette règle. Un exemple est décrit par la Figure 2 donnant la représentation de l'instance `20_2_nonvalidly.txt` : les sommets coloriés en verts désignent les sources, en bleu les noeuds intermédiaires et en rouge les destinations.

3 Documents à remettre

1. Génération du modèle :

Un script python3 nommé `generate_model.py` prenant en paramètres en ligne de commande le nom d'une instance dans le même dossier et un paramètre p égal à 0 ou 1 qui génère un programme linéaire en nombre entiers de cette instance au format CPLEX LP vu en TP. La valeur de p indique quel modèle générer :

- si $p = 0$, générer le modèle agrégé
- si $p = 1$, générer le modèle désagrégé

Ce programme doit être sauvé dans un fichier `model_instance.lp` où p est le paramètre passé en ligne de commande. Le script appelé sur l'instance `20_2_nonvalidly.txt` via la commande

4 Consignes de remise

Ce travail est à réaliser par groupe de 2 personnes. Vous êtes invités à communiquer votre groupe pour le **15 avril 2024** en envoyant un email reprenant les noms et prénoms des membres du groupe à l'adresse email `hugo.callebaut@ulb.be`.

Si vous ne trouvez personne, envoyez aussi un email avant cette date. Si vous ne vous manifestez pas avant cette date, nous ne pourrons vous garantir de trouver un partenaire pour le projet.

Pour remettre votre projet sur l'UV, nous vous demandons de

- créer localement sur votre machine un répertoire intitulé `NOM1_NOM2` (exemple :`PAPADIMITRIOU_CALLEBAUT`, sans espace) dans lequel vous mettez les fichiers demandés (sans y inclure de fichier de données).
- compresser ce répertoire via un utilitaire d'archivage produisant un `.zip` (aucun autre format de compression n'est accepté)
- de soumettre le fichier archive `.zip`, et uniquement ce fichier, sur l'UV (une seule remise par binôme)

Le projet est à remettre pour le **17 mai 2024 à 23h59** sur l'UV. Tout manquement aux consignes ou retard sera sanctionné directement d'une note **0/20** pour le projet.