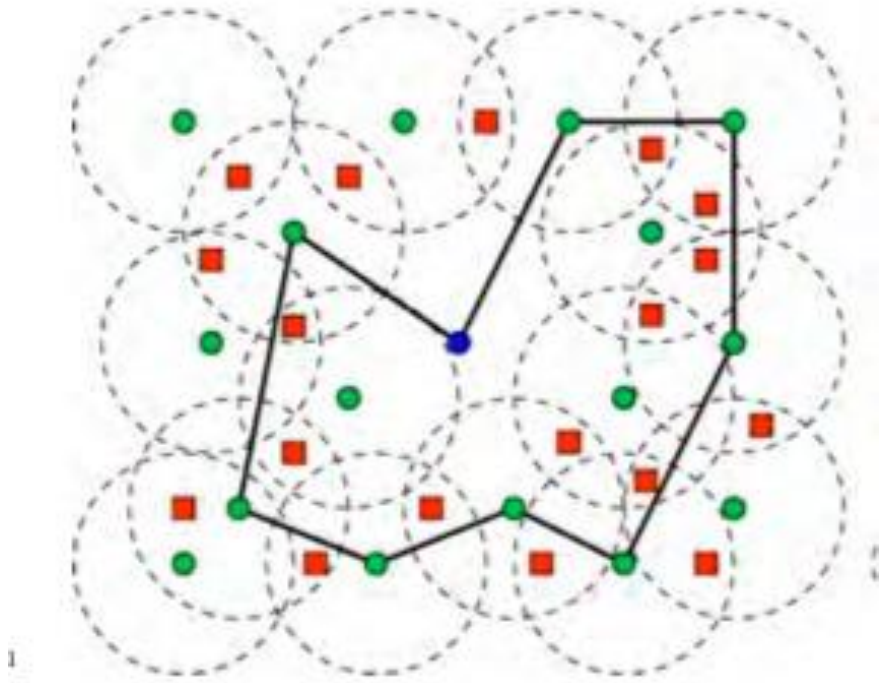


# Rapport – RO

## Maximisation de bénéfice net pour un entreprise de transport

### Extraction de Données - Graphe

RO : RECHERCHE OPERATIONNELLE

REALISES PAR : KARLA ROSAS  ET AWA DIABY
RESPONSABLE TP : VICTOR EPAIN
RESPONSABLE DU MODULE : ROUMEN ANDONOV

Réalisé : Février 2021

ETABLISSEMENT : UNIVERSITE DE RENNES 1
Groupe : 1A
Discipline : Master 1 MIAGE

# Contenu



# Introduction

Pour développer notre projet nous avons étudié le problème classique de <<transbordement>>. Nous avons aussi mis en pratique les aspects vus sur les TP du cours de Recherche opérationnel.

# Objectifs du projet

Ce projet est composé de deux grandes partir que sont : l'extraction des données du dossier « data », ensuite utiliser ses données dans un fichier « truck_pulp.py » pour effectuer les différentes opérations nécessaires pour obtenir le résultat souhaiter.

# Les outils et librairies utilises

Ce projet est codé python

Librairies :

- Pulp
- networkx

# 1. Présentation du programme linéaire

## ➢ Les différentes Données exploitées dans le code

- 1 camion
- 3 dépôts (D1, D2, D3)
- 2 Clients (C1, C2)
- Pour le camion nous avons une limite P de matériel à son bord
- La route à une capacité de GPU qui peut être transportées
- Pour chaque client servi, doit l'être dans la totalité de sa demande et payent ainsi « 1000 euros » pour chaque GPU

## ➢ Modélisation

L'objectifs :

Maximiser le bénéfice net maximal : Ventes – Coûts

Ventes = Quantité de GPU vendus * 1000 euros

Coûts = Somme de Cr, Cr = αr + βr * qr => Ici il s'agit de Minimiser le Coûts de Transports.

## ➢ Variables

Cr = αr + βr * qr (Le coût de chaque traverser en fonction de alpha et beta par rapport à chaque quantité de GPU dans le camion)

αr >= 0 : il s'agit du coût de l'essence

βr >= 0 : Taxe douanière lors de chaque traverser

Dc (i) : Demande client, avec i qui varie de 1 à m

Sd (j) : Stock du dépôt

Cap : Capacité sur la route des GPU transportés

P : Capacité du camion (GPU)

- Contraintes

Ru : Routes = {1 si une route est traversée, 0 sinon}

GPU transportés (Quantité dans le camion) <= Cap

Quantité de GPU dans le camion <= p (limite de matériel à son bord)

Quantité de GPU dans le camion qr >= 0 (on ne peut pas avoir une quantité négative dans le camion)

Quantité GPU dans le camion qr >= demande du client (chaque client servi doit l'être dans la totalité de sa demande)

Dc = 0 (Demande de client supérieur ou égale à zéro)

## 2.Présentation de l'ensemble du programme

<<Le problème de transbordement>>

| Modèle | Interprétation |
|---|---|
| **Unités transportées (qr)** | Quantités de GPU |
| **Origine (m)** | Dépôts |
| **Destinataire (n)** | Clients |
| **Ressource en dépôts (Si)** | Quantités disponibles |
| **Demande du client (dj)** | Quantités demandes |
| **Coûts par unité GPU (Cij)** | Coûts unitaires par unités distribuée du dépôt au client |

Hypothèse pour trouver des solutions : Conservation

$$\sum_{i=1}^{m} Si = \sum_{j=1}^{n} dj$$

$$x(i,j)\ Quantité\ cirlant\ dans\ les\ arcs$$
$$arcs(i,j)$$
$$Sommet\ X_i \geq 0\ (demande)$$
$$Sommet\ X_i \leq 0\ (stock\ disponible)$$

Tableau de paramètres

| | Clients | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | ... | n | Stock |
| 1 | $C_{11}$ | $C_{12}$ | ... | $C_{1n}$ | $S_1$ |
| | $C_{21}$ | $C_{22}$ | ... | $C_{2n}$ | $S_2$ |
| 2 Dépôt | ... | ... | ... | ... | ... |
| ... | $C_{m1}$ | $C_{12}$ | ... | $C_{mn}$ | $S_m$ |
| m | | | | | |
| | $D_1$ | $D_2$ | ... | $D_n$ | |

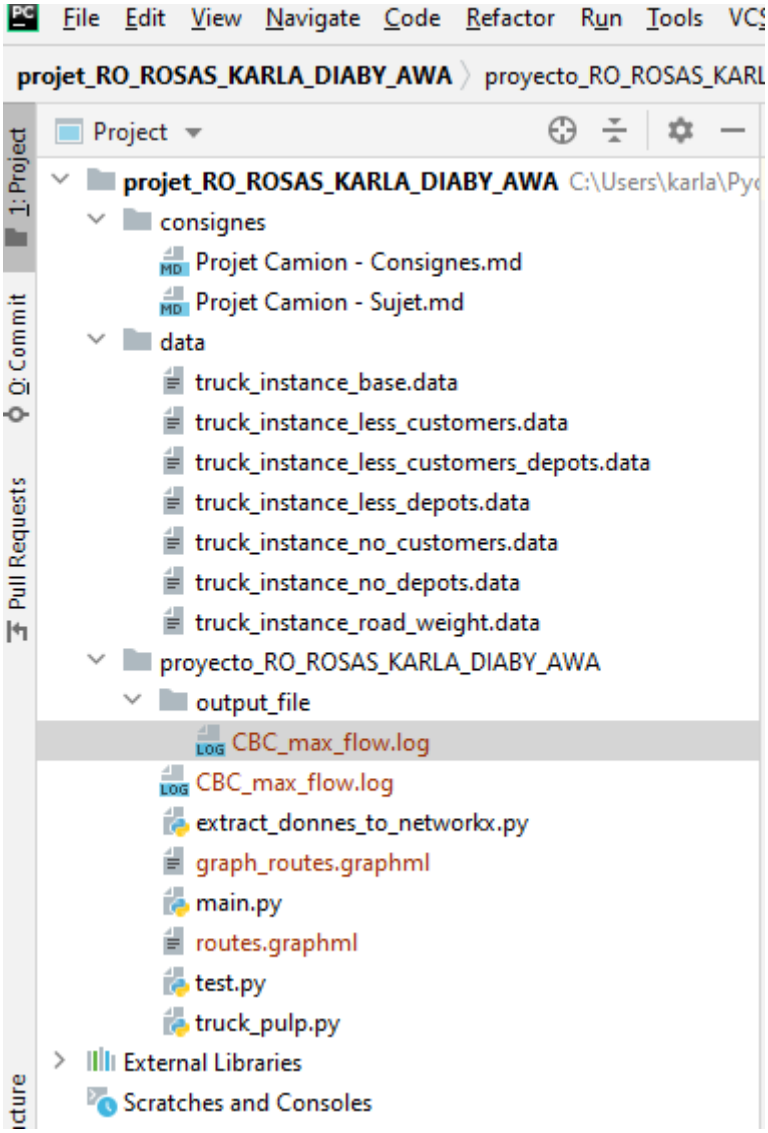
Min Z= C11X11 + C12X12 +.....CmnXmn

**Contraints de l'offert du dépôt :**
X11 + X12 +X1n <= S1
X21 + X22 +X2n <= S2
.....
Xm1 + Xm2 +Xmn <= Sm

**Contraints de la demande du client :**
X11 + X21 +Xm1 <= D1
X21 + X22 +X2n <= D2
.....
X1n + X2n +Xmn <= Dn

**Contraints additionnels décrites dans la section précédente.**

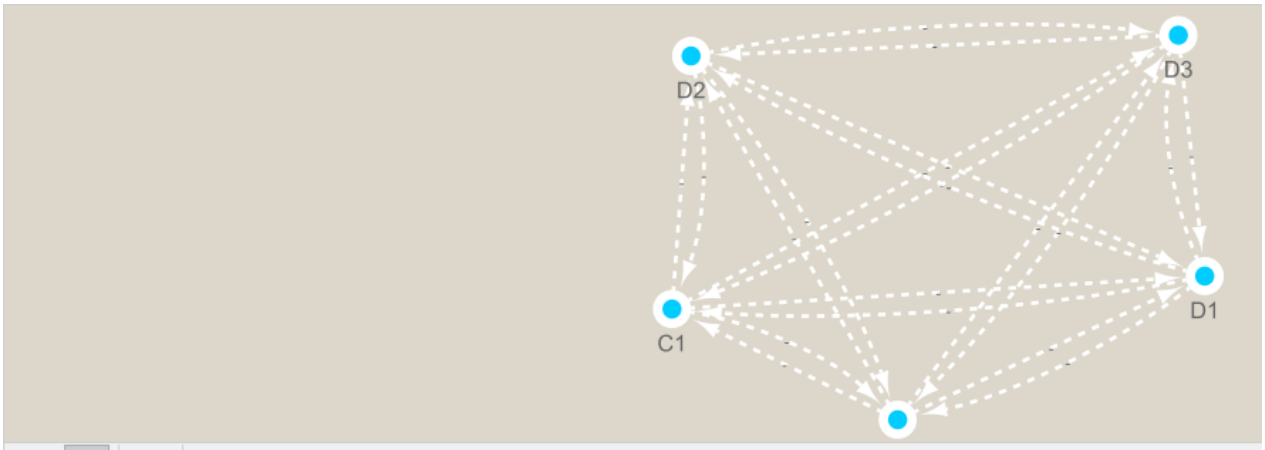
Structure du projet




| Fichier | Description |
|---|---|
| **Extract_donnes_to_networkx.py** | Extraction de donnés depuis le fichier truck_instance_base.data |
| **test.py** | Obtention de résultats finales |
| **Truck_pulp.py** | Calcule de la minimisation de coût et optimisation de la distribution de GPU |
| **CBC_max_flow.log** | Fichier avec les résultats de la fonction objectif |
| **graph_routes.graphml** | Visualisation du graphe obtenu de l'extraction de données |

## 3. Présentation des résultats

À partir de l'extraction des données du fichier 'truck_instance_base.data' nous avons obtenu un graphe avec ses caractéristiques.



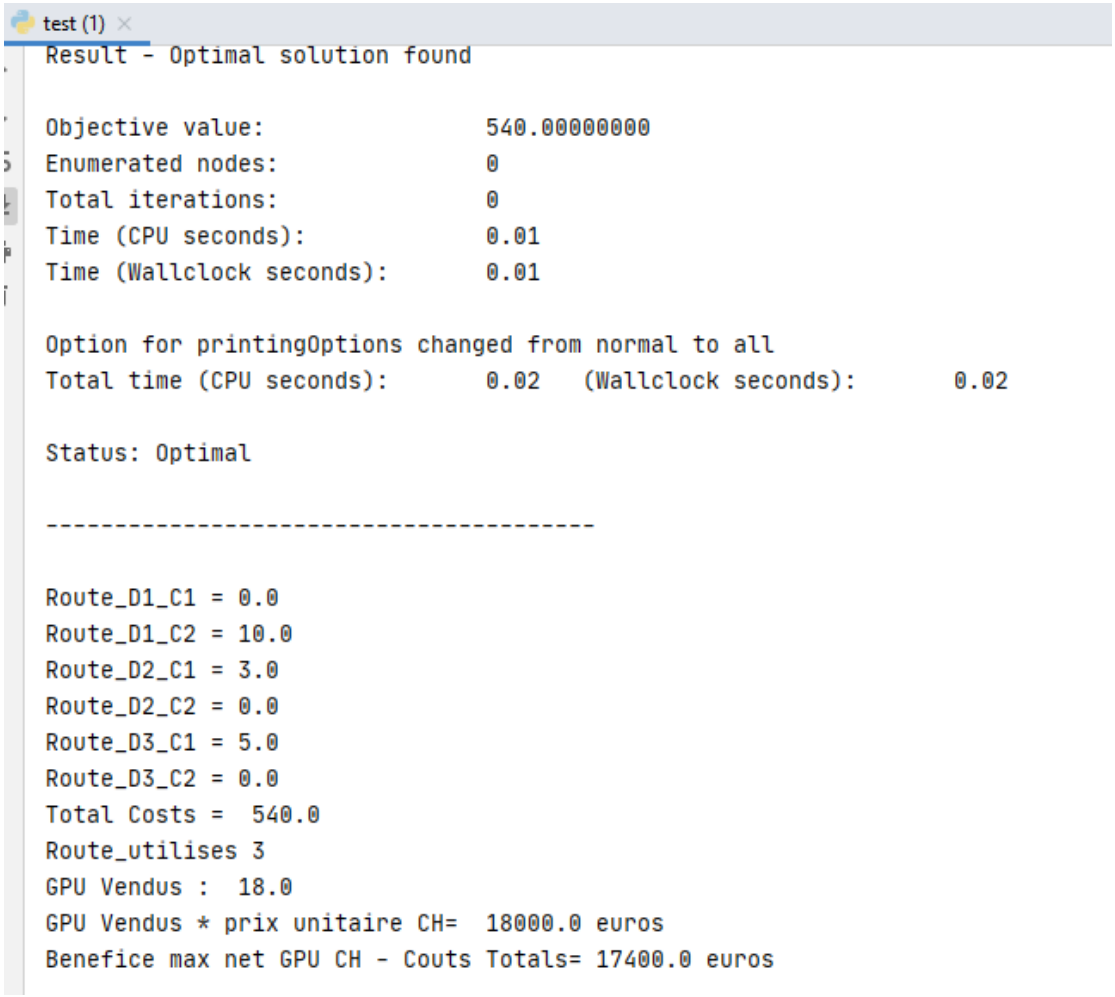

Edge Table

| shared name | shared interaction | cap | essence | tauxdou | name | interaction |
|---|---|---|---|---|---|---|
| D1 (-) D2 | - | 10 | 20 | 30 | D1 (-) D2 | - |
| D1 (-) D3 | - | 10 | 20 | 30 | D1 (-) D3 | - |
| D1 (-) C1 | - | 10 | 20 | 30 | D1 (-) C1 | - |
| D1 (-) C2 | - | 10 | 20 | 30 | D1 (-) C2 | - |
| D2 (-) D1 | - | 10 | 20 | 30 | D2 (-) D1 | - |
| D2 (-) D3 | - | 10 | 20 | 30 | D2 (-) D3 | - |
| D2 (-) C1 | - | 10 | 20 | 30 | D2 (-) C1 | - |
| D2 (-) C2 | - | 10 | 20 | 30 | D2 (-) C2 | - |
| D3 (-) D1 | - | 10 | 20 | 30 | D3 (-) D1 | - |
| D3 (-) D2 | - | 10 | 20 | 30 | D3 (-) D2 | - |
| D3 (-) C1 | - | 10 | 20 | 30 | D3 (-) C1 | - |

Node Table

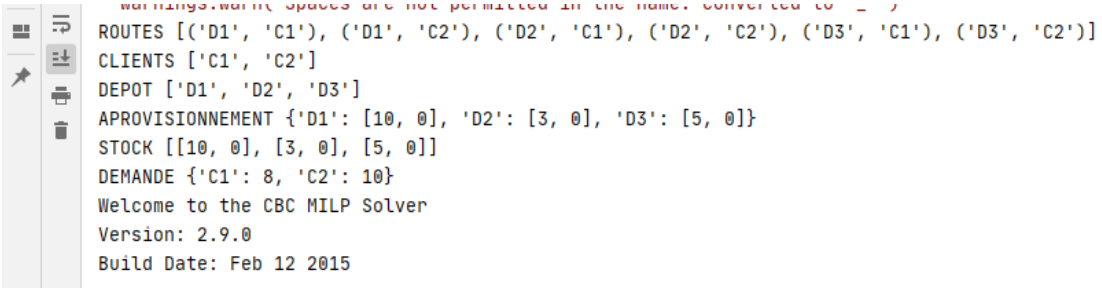
| shared name | type | entity | name | NumChildren | __ViewSettings | __AggregationSettings |
|---|---|---|---|---|---|---|
| D1 | depot | -10 | D1 | | | |
| D2 | depot | -3 | D2 | | | |
| D3 | depot | -5 | D3 | | | |
| C1 | customer | 8 | C1 | | | |
| C2 | customer | 10 | C2 | | | |

Obtention de résultats truck_pulp.py




## 4. L'interprétation des données

Nous avons créé des listes et dictionnaires pour afficher et manipuler les données :



| | |
|---|---|
| **ROUTES** | Liste de différentes routes |
| **CLIENTS** | Liste avec les différents clients |
| **DEPOT** | Liste avec les différents dépôts |
| **APROVISIONNEMENT** | Dictionnaire avec les quantités disponibles en stock |
| **STOCK** | Liste de stock |
| **DEMANDE** | Dictionnaire avec les demandes de chaque client |

Pour obtenir la maximisation du bénéfice net de l'entreprise de transport, nous avons décidé de minimiser le coût de transport. Nous avons considéré comme coût

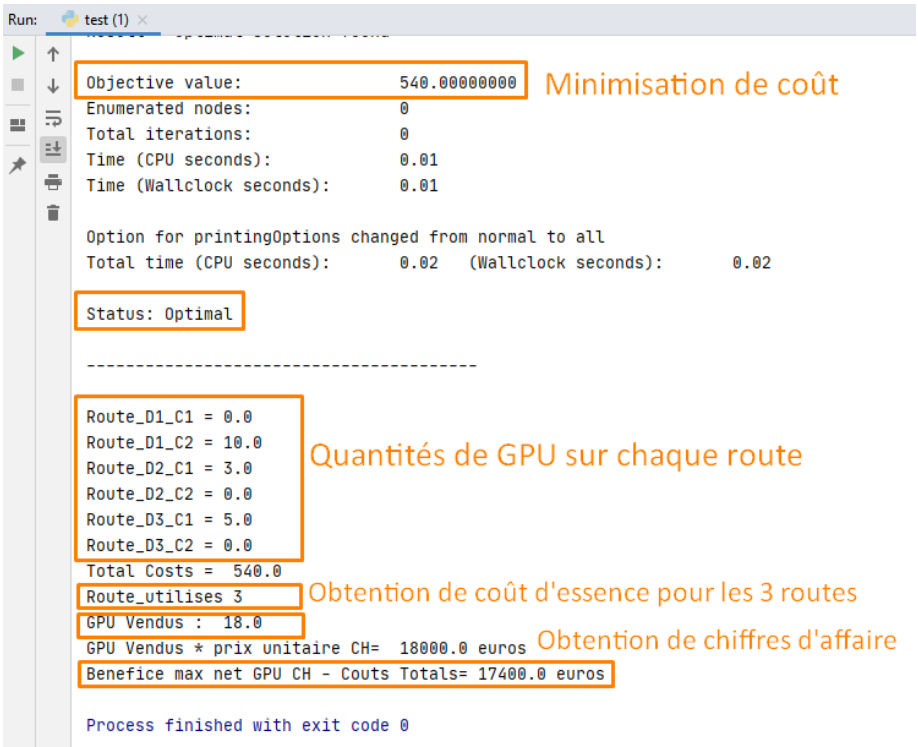
Cr =essence + (taux douanière * quantité de GPU)

Fonction objective avec la librairie Pulp = Minimiser Cr

Les résultats obtenus :

Le coût minimal du transport, les quantités vendus de GPU et les quantités par route.

## 5. L'interprétation des résultats



## Des hypothèses :

Nous avons réalisé l'objectif d'obtenir le bénéfice net max pour l'entreprise de transport, mais nous n'avons pas pu considérer toutes routes.

Cela est dure aux faites que nous n'avions pas suffisamment de temps et aussi nous n'avons pas compris du tout comme procède pour mettre les différentes routes en place.

Pour cela nous avions envisagé :

1. Ajouter toutes les routes possibles que nous avons sur le graph « graph.edges() » dans notre liste ROUTES sur le fichier truck_pulp.py

2. Ajouter les autres variables pour réussir à calculer Cr = essence + taux douanière *qr  sur la fonction objective et pas calculer le prix d'essence de manière indépendant sur le fichier test.py

3. Comprendre mieux la manipulation des différentes structures avec la libraire Pulp.