Projet ECMA - Optimisation Robuste

Problème de tournées de véhicules robuste

Réalisé par :

Mohamed Aziz MHADHBI

Mohamed Iheb KACEM

MPRO

Année universitaire : 2024/2025

Table des matières

Chapitre 1

Description du problème

On considère un problème de tournées de véhicules dans lequel un entrepôt doit livrer des vaccins à un ensemble de clients. Le problème est caractérisé par :

- un ensemble de sommets $V = \{1, \ldots, n\}$ tel que le sommet 1 correspond à l'entrepôt des véhicules où se trouve le stock de vaccins et les autres sommets aux clients qu'il faut livrer. Soit $A = \{ij \in V^2 \mid i \neq j\}$;
- une durée $t_{ij} \in \mathbb{N}$ associée à chaque couple de sommets $(i, j) \in A$, $i \neq j$, correspondant au temps minimal pour qu'un véhicule se rende du sommet i au sommet j ;
- une demande $d_i \in \mathbb{N}$ associée à chaque sommet $i \in V \setminus \{1\}$ correspondant au nombre de vaccins à livrer au client i ;
- une capacité maximale $C \in \mathbb{N}$ correspondant au nombre maximal de vaccins qu'il est possible de stocker dans un véhicule. Cette valeur est identique pour tous les véhicules.

On définit une tournée T comme une suite de sommets $T = \{t_1, t_2, \ldots, t_{|T|}\}$ telle que :

- la tournée débute et termine à l'entrepôt (i.e., $t_1 = t_{|T|} = 1$) ;
- la somme des demandes des clients visités n'excède pas la capacité d'un véhicule (i.e., $\sum_{i=2}^{|T|-1} d_{t_i} \leq C$) ;
- tous les clients visités sont différents (i.e., $t_i \neq t_j \ \forall i, j \in \{2, \ldots, |T| - 1\},\ i \neq j$).

L'objectif du problème statique consiste à déterminer un ensemble de tournées permettant de visiter l'ensemble des clients tout en minimisant la durée totale des trajets.

Précisions :

- Le nombre de véhicules n'est pas limité et n'intervient pas dans la fonction objectif.
- Chaque client doit être visité dans exactement une tournée. Ainsi, même s'il peut parfois être intéressant de visiter plusieurs fois un même client, ceci n'est pas autorisé.

Ensuite, on souhaite résoudre une version robuste du problème statique car il est possible que les valeurs choisies pour les temps de trajets soient sous-évaluées.

Chapitre 2

Modélisation du problème statique

2.1 Modélisation compacte

On considère le programme en nombre entiers compact suivant :

2.1.1 Variables de décision

$x_{i,j} \in \{0,1\}$	$\forall i,j \in \{1,\ldots,n\},$	indique si l'arc (i,j) est utilisé.
$u_i \in \mathbb{Z}_{\geq 0}$	$\forall i \in \{1,\ldots,n\},$	ordre de visite du nœud i.
$q_i \in \mathbb{Z}_{\geq 0}$	$\forall i \in \{1,\ldots,n\},$	quantité transportée disponible au nœud i.

2.1.2 Objectif

Minimiser la distance totale parcourue :

$$\text{Minimiser} \sum_{1 \leq i,j \leq n} t_{i,j} \cdot x_{i,j}.$$

2.1.3 Contraintes

1. Conservation de flux :

$$\sum_{j=1}^{n} x_{i,j} = 1, \quad \forall i \in \{2,\ldots,n\},$$

$$\sum_{j=1}^{n} x_{j,i} = 1, \quad \forall i \in \{2,\ldots,n\}.$$

2. Élimination de cycles (MTZ) : l'entrepôt est le seul sommets qui peut appartenir à des cycles :

$$u_i + x_{i,j} - (n-2)(1 - x_{i,j}) \leq u_j, \quad \forall i \in \{1,\ldots,n\}, j \in \{2,\ldots,n\}.$$

3. Contraintes de capacité :

$$q_i \leq C, \quad \forall i \in \{1, \ldots, n\}.$$

4. Satisfaction de la demande et évolution de la quantité q :

$$q_i >= d_i$$

$$q_j = q_i - d_i \text{ si } x_{i,j} = 1$$

la dernière contrainte se traduit de la manière suivante :

$$q_j >= q_i - d_i - M * (1 - x_{i,j})$$

$$q_j <= q_i - d_i + M * (1 - x_{i,j})$$

On a $q_i \leq C, \forall i$ donc on peut prendre $M = C$.
De plus on ne peut pas diminuer encore M car initialement $q_1 = C$

2.1.4 Programme linéaire résultant

$$\begin{aligned}
\text{Min} \quad & \sum_{1 \leq i,j \leq n} t_{i,j} \cdot x_{i,j} \\
\text{sous les contraintes :} \quad & \sum_{j=1}^{n} x_{i,j} = 1, \quad \forall i \in \{2, \ldots, n\}, \\
& \sum_{j=1}^{n} x_{j,i} = 1, \quad \forall i \in \{2, \ldots, n\}, \\
& u_i + x_{i,j} - (n-2)(1 - x_{i,j}) \leq u_j, \quad \forall i, j, \\
& q_i \leq C, \quad \forall i \in \{1, \ldots, n\}, \\
& q_i \geq d_i, \quad \forall i \in \{1, \ldots, n\}, \\
& q_j \geq q_i - d_i - M(1 - x_{j,i}), \quad \forall i \in \{2, \ldots, n\}, j \in \{1, \ldots, n\}, \\
& q_j \leq q_i - d_i + M(1 - x_{j,i}), \quad \forall i \in \{2, \ldots, n\}, j \in \{1, \ldots, n\}, \\
& x_{i,j} \in \{0, 1\}, \quad u_i \geq 0, \quad q_i \geq 0
\end{aligned}$$

Dans la suite on note :

$$
\begin{aligned}
\mathcal{X} = \{ \quad & x, u, q \quad \text{tq :} \\
& \sum_{j=1}^{n} x_{i,j} = 1, \quad \forall i \in \{2, \dots, n\}, \\
& \sum_{j=1}^{n} x_{j,i} = 1, \quad \forall i \in \{2, \dots, n\}, \\
& u_i + x_{i,j} - (n-2)(1 - x_{i,j}) \leq u_j, \quad \forall i, j, \\
& q_i \leq C, \quad \forall i \in \{1, \dots, n\}, \\
& q_i \geq d_i, \quad \forall i \in \{1, \dots, n\}, \\
& q_j \geq q_i - d_i - M(1 - x_{j,i}), \quad \forall i \in \{2, \dots, n\}, j \in \{1, \dots, n\}, \\
& q_j \leq q_i - d_i + M(1 - x_{j,i}), \quad \forall i \in \{2, \dots, n\}, j \in \{1, \dots, n\}, \\
& x_{i,j} \in \{0, 1\}, \quad u_i \geq 0, \quad q_i \geq 0 \\
& \}
\end{aligned}
$$

Chapitre 3

Problème Robuste

$$\mathcal{U} = \left\{ \{t'_{ij} = t_{ij} + \delta^1_{ij}(\hat{t}_i + \hat{t}_j) + \delta^2_{ij}\hat{t}_i\hat{t}_j\}_{ij\in A} \text{ tel que } \begin{array}{l} \sum_{ij\in A} \delta^1_{ij} \leq T, \\ \sum_{ij\in A} \delta^2_{ij} \leq T^2, \\ \delta^1_{ij} \in [0,1],\ \delta^2_{ij} \in [0,2], \\ \forall ij \in A \end{array} \right\}.$$

Le problème statique est le suivant :

$$\begin{aligned} \text{Min} \quad & \sum_{1\leq i,j\leq n} t_{i,j} \cdot x_{i,j} \\ \text{s.t.} \quad & x, u, q \in \mathcal{X} \end{aligned}$$

On considère le problème robuste :

$$min_{x,u,q\in\mathcal{X}} \quad max_{t'_{i,j}\in\mathcal{U}} \sum_{1\leq i,j\leq n} t'_{i,j} \cdot x_{i,j}$$

3.1 Résolution par plans coupants et LazyCallback

3.1.1 Modification du problème afin que la robustesse n'apparaisse plus dans l'expression de l'objectif mais dans les contraintes

$$\begin{aligned} \min_{x,u,q,z} \quad & z \\ \text{s.t.} \quad & z \geq \sum_{1\leq i,j\leq n} t'_{i,j} \cdot x_{i,j} \quad \forall t'_{i,j} \in \mathcal{U} \\ & x, u, q \in \mathcal{X} \end{aligned}$$

3.1.2 Problème maître

$$\begin{aligned} \min_{x,u,q,z} \quad & z \\ \text{s.t.} \quad & \sum_{1\leq i,j\leq n} t'_{i,j} \cdot x_{i,j} \leq z \quad \forall t'_{i,j} \in \mathcal{U}^* \\ & x, u, q \in \mathcal{X} \end{aligned}$$

On peut considérer initialement l'ensemble

$$\mathcal{U}^* = \left\{ \{t'_{ij} = t_{ij}\}_{ij\in A} \text{ i.e. } \delta^1_{ij} = 0 \; \delta^2_{ij} = 0, \forall ij \in A \right\}.$$

3.1.3 Le sous-problème

Pour procéder à la résolution par plan coupant on considère le sous-problème (SP) suivant pour la génération des coupes :

$$\begin{aligned} \max_{(\delta^1_{ij}),(\delta^2_{ij})} \quad & \sum_{1\leq i,j\leq n} (t_{ij} + \delta^1_{ij}(\hat{t}_i + \hat{t}_j) + \delta^2_{ij}\hat{t}_i\hat{t}_j) \cdot x^*_{i,j} \\ & \sum_{ij\in A} \delta^1_{ij} \leq T, \\ & \sum_{ij\in A} \delta^2_{ij} \leq T^2, \\ & \delta^1_{ij} \in [0,1], \delta^2_{ij} \in [0,2], \forall ij \end{aligned}$$

Conditions d'optimalité

Pour qu'une solution (x^*, z^*) du problème maître soit optimale il faut avoir une solution ${t'_{i,j}}^*$ de SP telle que (aucune contrainte robuste n'est violée) :

$$\sum_{1\leq i,j\leq n} {t'_{i,j}}^* \cdot x^*_{i,j} \leq z^*$$

Expression des coupes ajoutées :

Les coupes ajoutées par le sous-problème sont de la forme :

$$\sum_{1\leq i,j\leq n} {t'_{i,j}}^* \cdot x_{i,j} \leq z$$

3.2 Résolution par dualisation

Pour résoudre le problème robuste par dualisation, on procède selon les étapes suivantes :

- On reformule l'objectif du problème robuste afin d'isoler le terme faisant intervenir les variables δ^1_{ij} et δ^2_{ij} d'ou on obtient :

$$min_{x,u,q\in\mathcal{X}} \quad max_{t'_{i,j}\in\mathcal{U}} \sum_{1\leq i,j\leq n} t'_{i,j}\cdot x_{i,j}$$

$=$

$$min_{x,u,q\in\mathcal{X}} \quad max_{(\delta^1_{ij}),(\delta^2_{ij})} \quad \sum_{1\leq i,j\leq n} (t_{ij}+\delta^1_{ij}(\hat{t}_i+\hat{t}_j)+\delta^2_{ij}\hat{t}_i\hat{t}_j)\cdot x_{i,j}$$

$=$

$$min_{x,u,q\in\mathcal{X}} \sum_{1\leq i,j\leq n} t_{i,j}\cdot x_{i,j} + \quad max_{(\delta^1_{ij}),(\delta^2_{ij})} \quad \sum_{1\leq i,j\leq n} x_{i,j}(\hat{t}_i+\hat{t}_j)\delta^1_{ij}+x_{i,j}\hat{t}_i\hat{t}_j\delta^2_{ij}$$

- Le problème interne lié à ces variables s'écrit :

$$\begin{aligned} max_{(\delta^1_{ij}),(\delta^2_{ij})} \quad & \sum_{1\leq i,j\leq n} x_{i,j}(\hat{t}_i+\hat{t}_j)\delta^1_{ij}+x_{i,j}\hat{t}_i\hat{t}_j\delta^2_{ij} \\ & \sum_{ij\in A}\delta^1_{ij}\leq T, \\ & \sum_{ij\in A}\delta^2_{ij}\leq T^2, \\ & \delta^1_{ij}\in[0,1],\delta^2_{ij}\in[0,2],\forall ij \end{aligned}$$

- On dualise ce problème d'ou on obtient :

$$\begin{aligned} min_{\alpha^1,\alpha^2,\beta^1,\beta^2} \quad & T\cdot\alpha^1+T^2\cdot\alpha^2+\sum_{ij\in A}\beta^1_{ij}+2\beta^2_{ij} \\ & \alpha^1+\beta^1_{ij}\geq x_{i,j}(\hat{t}_i+\hat{t}_j)\forall ij \\ & \alpha^2+\beta^2_{ij}\geq x_{i,j}\hat{t}_i\hat{t}_j\forall ij \\ & \alpha^1\geq 0,\alpha^2\geq 0,\beta^1_{ij}\geq 0,\beta^2_{ij}\geq 0 entiers \end{aligned}$$

- Finalement,le problème robuste s'écrit sous la forme de ce PLNE :

$$\begin{aligned} min_{\alpha^1,\alpha^2,\beta^1,\beta^2} \quad & \sum_{1\leq i,j\leq n}(t_{i,j}\cdot x_{i,j}+\beta^1_{ij}+2\beta^2_{ij})+T\cdot\alpha^1+T^2\cdot\alpha^2 \\ & \alpha^1+\beta^1_{ij}\geq x_{i,j}(\hat{t}_i+\hat{t}_j)\forall ij \\ & \alpha^2+\beta^2_{ij}\geq x_{i,j}\hat{t}_i\hat{t}_j\forall ij \\ & x,u,q\in\mathcal{X} \\ & \alpha^1\geq 0,\alpha^2\geq 0,\beta^1_{ij}\geq 0,\beta^2_{ij}\geq 0 entiers \end{aligned}$$

Chapitre 4

Etude éxpérimentale

En utilisant Julia et Cplex, nous avons implémenté différentes méthodes de résolution exacte pour le problème ainsi qu'une approche heuristique.
Dans ce chapitre, nous allons présenter les approches utilisées ainsi que les résultats trouvés pour chacune ,puis nous allons analyser les performances et faire une étude comparative.

4.1 Description des méthodes

Ci dessous est la description de chaque méthode implémentée.

4.1.1 Méthode de Plans Coupants

- On part d'un problème appelé le problème maître tel que,dans la contrainte surlaquelle porte l'incertitude,l'ensemble des scénarios possible est initialisé à :
$$\mathcal{U}^* = \left\{ \{t'_{ij} = t_{ij}\}_{ij \in A} \text{ i.e. } \delta^1_{ij} = 0 \; \delta^2_{ij} = 0, \forall ij \in A \right\}.$$
- On se munit d'un sous-ensemble initial de coupes qui est initialement vide, il s'agit d'une manière de sauvegarder à chaque itération les coupes générées.
- On a implémenté une boucle,qui résout à chaque itération le problème maître et le sous-problème associé puis génère la coupe associée,celle-ci sera enregistrée et dans le cas où elle est violée, elle sera alors ajoutée au problème maître et une autre itération de la boucle sera lancée.
- La boucle s'arrête dès qu'on trouve une coupe non violée.
- Remarque : L'arbre de branchement est refait à partir de sa racine à chaque fois qu'on trouve une coupe violée d'ou les coupes sont ajoutées à chaque itération et pas au niveau des noeuds.

4.1.2 Méthode de Branch and Cut

- On utilise les mêmes coupes que la méthode de plans coupants.
- Cette fois çi, on n'a plus à refaire tout l'arbre de branchement du problème maître en ajoutant les coupes violées à la racine comme on l'a fait pour les plans coupants.

- La résolution du sous-problème ainsi que l'ajout des coupes violées se fait à chaque noeud en utilisant le lazycallback.
- Avantage : Ne pas repartir de 0 après chaque ajout de contrainte.

4.1.3 Méthode de Dualisation

- On a implémenté le modèle PLNE dualisé élaboré dans le chapitre précédent.
- C'est une méthode exacte de résolution.

4.1.4 Méthode Heuristique

Introduction

Dans cette partie nous considérons une heuristique simple qui consiste à partir de l'entrepôt et choisir le plus proche voisin selon un critère donnée. On considère donc une matrice de temps ajustée, notée t', pour prendre en compte la robustesse.

Modification de la matrice de temps

La matrice de temps ajustée t' est obtenue en modifiant la matrice originale t en fonction des temps $\hat{t}$ et du paramètre global T. Cette transformation est définie comme suit :

$$t'_{ij} = t_{ij} + \frac{T}{n^2}(\hat{t}_i + \hat{t}_j) + \frac{T^2}{n^2}(\hat{t}_i \cdot \hat{t}_j), \quad \forall i \neq j. \tag{4.1}$$

Cette modification permet d'approcher au plus le pire des cas qui peut se produire dans l'ensemble d'incertitude.

Heuristique de sélection des clients

L'algorithme de résolution repose sur une heuristique basée sur deux critères :
— **Proximité** : sélection du client le plus proche en termes de distance t' ;
— **Ratio demande/distance** : sélection du client maximisant le rapport $\frac{d_i}{t'_{current,i}}$.
— **Critère demande - distance** : sélection du client minimisant le facteur $d_i.t'_{current,i}$.
À chaque itération, on choisit le prochain client i qui optimise l'un des facteurs.

Résolution du VRP avec t'

L'algorithme suit les étapes suivantes :

1. Initialisation des tournées et des variables de suivi (position actuelle, charge, clients restants).
2. Sélection des clients admissibles respectant la contrainte de capacité.
3. Choix du client suivant en fonction des critères définis.
4. Mise à jour des variables et poursuite de la tournée.
5. Retour à l'entrepôt lorsque plus aucun client ne peut être ajouté.

Le processus est répété jusqu'à ce que tous les clients soient servis, générant ainsi une ou plusieurs tournées minimisant la valeur objective.

Comparaison des variantes

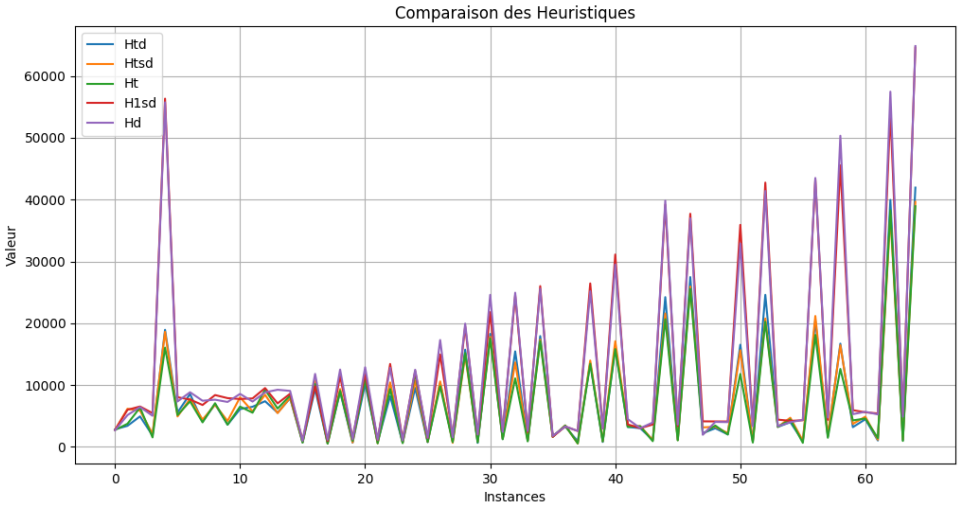

FIGURE 4.1 – Comparaison des variantes des heuristiques

On remarque que l'heuristique la plus performante est celle qui se base sur le ratio du plus proche voisin et celle qui minimise le ratio $d_i.t'_{current,i}$ i.e. choisir a chaque itération le client le plus proche et qui minimise la demande si la capacité le permette.

4.2 Comparaison des performances

Dans cette partie, on va comparer les 4 approches implémentées pour résoudre notre problème. En effet, ce tableau illustre le temps d'exécution de chaque méthode ,le prix de robustesse ainsi que le GAP pour la méthode heuristique.

TABLE 4.1 – Résultats des différentes méthodes sur les instances euclidiennes et non euclidiennes.

Instance	**PR**	**Statique**	**B&C**	**Dual.**	**P. coupants**	**Heuristique**	
		Time	Time	Time	Time	Time	gap (%)
Instances non euclidiennes							
$n = 6$	29.48%	0.025	1.112	0.936	1.836	0	0.18
$n = 7$	11.25%	0.020	0.116	0.027	0.065	0	0.43
$n = 8$	40.04%	0.034	0.160	0.062	0.206	0	0.08
$n = 9$	15.34%	0.055	0.225	0.117	0.318	0	0.23
$n = 10$	22.56%	0.058	0.643	0.096	0.883	0	1.97
Instances euclidiennes							
$n = 6$	2.15%	0.022	0.205	0.038	0.162	0	0.15
$n = 7$	4.99%	0.056	0.312	0.074	0.727	0	0.29
$n = 8$	5.65%	0.060	0.250	0.062	0.518	0	0.62
$n = 9$	18.48%	0.042	0.655	0.075	4.831	0	0.80
$n = 10$	12.16%	0.093	0.577	0.183	4.918	0	1.29

Observations

- La méthode statique donne une solution optimale rapidement.
- De point de vue temps de résolution, la méthode de dualisation est la plus rapide par rapport aux autres méthodes robustes exactes.
- Le prix de la robustesse est plus élevé pour les instances non euclidiennes.
- L'approche heuristique génère des résultats performants puisqu'elle donne un gap faible pour ces instances et en un temps très négligeable (presque nul).

Diagramme de performances

Pour avoir ce diagramme, nous avons fixé un temps limite d'exécution par instance égal à 30 secondes pour voir le nombre d'instances résolus pendant ce temps là.

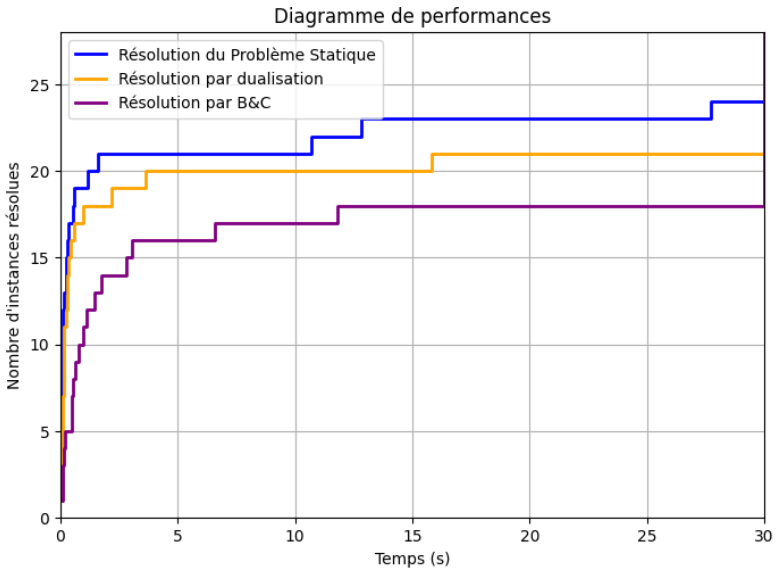

FIGURE 4.2 – Diagramme de performances

Nous remarquons que la méthode statique résout le plus grand nombre d'instances vu qu'elle est moins complexe que les méthodes robustes.
Pour les approches robustes, la méthode de dualisation résout plus d'instances que les autres (21 instances parmi 64), cependant la méthode Branch and Cut par exemple a pu résoudre 18 instances.
D'autre part, la méthode de plans coupants nécessite le plus de temps pour toutes les instances presque, et on peut justifier cela par le fait que le problème est résolu dès le début à chaque fois qu'on ajoute une coupe.

Chapitre 5

Annexe

Les solutions optimales obtenues pour le problème robuste :

Instance	**Tournées**	**Valeur optimale**
n_10-euclidean_false.txt	$1 \to 2 \to 8 \to 3 \to 1$ $1 \to 6 \to 7 \to 5 \to 1$ $1 \to 10 \to 4 \to 9 \to 1$	2684.0
n_10-euclidean_true.txt	$1 \to 2 \to 6 \to 5 \to 7 \to 1$ $1 \to 3 \to 4 \to 8 \to 1$ $1 \to 10 \to 9 \to 1$	5423.0
n_5-euclidean_false.txt.txt	$1 \to 4 \to 3 \to 2 \to 1$ $1 \to 5 \to 1$	1767.0
n_5-euclidean_true.txt.txt	$1 \to 2 \to 5 \to 1$ $1 \to 4 \to 3 \to 1$	3283.0
n_6-euclidean_false.txt.txt	$1 \to 2 \to 1$ $1 \to 4 \to 3 \to 1$ $1 \to 6 \to 5 \to 1$	3307.0
n_6-euclidean_true.txt.txt	$1 \to 2 \to 6 \to 5 \to 1$ $1 \to 3 \to 4 \to 1$	3087.0
n_7-euclidean_false.txt.txt	$1 \to 5 \to 7 \to 4 \to 1$ $1 \to 6 \to 2 \to 3 \to 1$	2066.0
n_7-euclidean_true.txt.txt	$1 \to 5 \to 2 \to 1$ $1 \to 7 \to 4 \to 3 \to 6 \to 1$	3132.0
n_8-euclidean_false.txt	$1 \to 3 \to 5 \to 1$ $1 \to 6 \to 4 \to 8 \to 1$ $1 \to 7 \to 2 \to 1$	3270.0
n_8-euclidean_true.txt	$1 \to 3 \to 1$ $1 \to 5 \to 2 \to 1$ $1 \to 7 \to 4 \to 6 \to 8 \to 1$	3776.0
n_9-euclidean_false.txt	$1 \to 4 \to 2 \to 1$ $1 \to 5 \to 8 \to 9 \to 1$ $1 \to 7 \to 3 \to 6 \to 1$	3113.0

Instance	Tournées	Valeur optimale
n_9-euclidean_true.txt	$1 \to 4 \to 8 \to 6 \to 2 \to 1$ $1 \to 5 \to 1$ $1 \to 7 \to 3 \to 1$ $1 \to 9 \to 1$	4181.0
n_11-euclidean_false.txt	$1 \to 3 \to 6 \to 8 \to 9 \to 1$ $1 \to 4 \to 2 \to 7 \to 1$ $1 \to 5 \to 10 \to 11 \to 1$	3899.0
n_11-euclidean_true.txt	$1 \to 3 \to 7 \to 6 \to 10 \to 1$ $1 \to 4 \to 2 \to 1$ $1 \to 9 \to 1$ $1 \to 11 \to 8 \to 5 \to 1$	7142.0
n_12-euclidean_false.txt	$1 \to 2 \to 11 \to 1$ $1 \to 3 \to 10 \to 5 \to 6 \to 1$ $1 \to 4 \to 9 \to 8 \to 7 \to 12 \to 1$	3389.0
n_12-euclidean_true.txt	$1 \to 3 \to 1$ $1 \to 5 \to 2 \to 1$ $1 \to 7 \to 4 \to 6 \to 8 \to 1$	3776.0
n_13-euclidean_false.txt	$1 \to 6 \to 11 \to 13 \to 5 \to 7 \to 1$ $1 \to 9 \to 8 \to 4 \to 3 \to 1$ $1 \to 10 \to 2 \to 12 \to 1$	2682.0
n_13-euclidean_true.txt	$1 \to 2 \to 5 \to 1$ $1 \to 3 \to 11 \to 1$ $1 \to 12 \to 8 \to 4 \to 1$ $1 \to 13 \to 7 \to 10 \to 9 \to 6 \to 1$	5971.0
n_14-euclidean_false.txt	$1 \to 5 \to 3 \to 1$ $1 \to 11 \to 6 \to 8 \to 1$ $1 \to 12 \to 4 \to 9 \to 2 \to 7 \to 1$ $1 \to 13 \to 10 \to 14 \to 1$	5090.0
n_16-euclidean_false.txt	$1 \to 4 \to 12 \to 1$ $1 \to 5 \to 15 \to 6 \to 1$ $1 \to 7 \to 2 \to 16 \to 10 \to 9 \to 1$ $1 \to 13 \to 11 \to 3 \to 14 \to 8 \to 1$	849.0
n_16-euclidean_true.txt	$1 \to 2 \to 15 \to 8 \to 1$ $1 \to 4 \to 9 \to 11 \to 1$ $1 \to 6 \to 5 \to 1$ $1 \to 10 \to 14 \to 1$ $1 \to 12 \to 7 \to 3 \to 1$ $1 \to 16 \to 13 \to 1$	9144.0
n_17-euclidean_false.txt	$1 \to 2 \to 15 \to 1$ $1 \to 6 \to 16 \to 14 \to 1$ $1 \to 7 \to 9 \to 17 \to 3 \to 5 \to 1$ $1 \to 8 \to 12 \to 4 \to 1$ $1 \to 11 \to 1$ $1 \to 13 \to 10 \to 1$	619.0

Instance	Tournées	Valeur optimale
n_17-euclidean_true.txt	$1 \to 2 \to 5 \to 1$ $1 \to 3 \to 11 \to 1$ $1 \to 12 \to 8 \to 4 \to 1$ $1 \to 13 \to 7 \to 10 \to 9 \to 6 \to 1$	6712.0
n_18-euclidean_false.txt	$1 \to 5 \to 3 \to 1$ $1 \to 11 \to 6 \to 8 \to 1$ $1 \to 12 \to 4 \to 9 \to 2 \to 7 \to 1$ $1 \to 13 \to 10 \to 14 \to 1$	1316.0
n_18-euclidean_true.txt	$1 \to 2 \to 15 \to 8 \to 1$ $1 \to 4 \to 9 \to 11 \to 1$ $1 \to 6 \to 5 \to 1$ $1 \to 10 \to 14 \to 1$ $1 \to 12 \to 7 \to 3 \to 1$ $1 \to 16 \to 13 \to 1$	10154.0
n_19-euclidean_false.txt	$1 \to 4 \to 8 \to 13 \to 1$ $1 \to 5 \to 18 \to 1$ $1 \to 14 \to 11 \to 19 \to 9 \to 6 \to 17 \to 1$ $1 \to 15 \to 10 \to 2 \to 1$ $1 \to 16 \to 3 \to 12 \to 7 \to 1$	1182.0
n_19-euclidean_true.txt	$1 \to 2 \to 5 \to 1$ $1 \to 3 \to 11 \to 1$ $1 \to 12 \to 8 \to 4 \to 1$ $1 \to 13 \to 7 \to 10 \to 9 \to 6 \to 1$	6987.0
n_20-euclidean_false.txt	$1 \to 4 \to 12 \to 1$ $1 \to 5 \to 15 \to 6 \to 1$ $1 \to 7 \to 2 \to 16 \to 10 \to 9 \to 1$ $1 \to 13 \to 11 \to 3 \to 14 \to 8 \to 1$	676.0
n_20-euclidean_true.txt	$1 \to 2 \to 15 \to 8 \to 1$ $1 \to 4 \to 9 \to 11 \to 1$ $1 \to 6 \to 5 \to 1$ $1 \to 10 \to 14 \to 1$ $1 \to 12 \to 7 \to 3 \to 1$ $1 \to 16 \to 13 \to 1$	9098.0