

Documentation Technique

CVRP Viewer

27.05.2020

Stefano Cirieco

**CFPT**

**IFA-P3B**

**V1.0**

Table des matières

[1 Introduction 2](#_Toc41496056)

[2 Rappel de l'énoncé 2](#_Toc41496057)

[2.1 Objectifs du projet 2](#_Toc41496058)

[2.2 Inventaire du matériel 2](#_Toc41496059)

[2.3 Inventaire des logiciels 2](#_Toc41496060)

[3 Analyse fonctionnelle 2](#_Toc41496061)

[3.1 Fonctionnement vu par l’utilisateur 2](#_Toc41496062)

[3.2 Description de l’utilisation et des fonctionnalités 2](#_Toc41496063)

[4 Analyse organique 3](#_Toc41496064)

[4.1 Fonctionnement interne 3](#_Toc41496065)

[4.1.1 Classes 3](#_Toc41496066)

[4.1.2 Format des fichiers 3](#_Toc41496067)

[4.2 Architecture 4](#_Toc41496068)

[4.3 Fenêtre de l’application 4](#_Toc41496069)

[5 Tests 4](#_Toc41496070)

[5.1 Conditions de tests 4](#_Toc41496071)

[5.2 Plan de test 4](#_Toc41496072)

[5.3 Rapport de tests 4](#_Toc41496073)

[6 Conclusion 4](#_Toc41496074)

[6.1 Difficultés rencontrées 4](#_Toc41496075)

[6.2 Améliorations possibles 4](#_Toc41496076)

[6.3 Bilan personnel 4](#_Toc41496077)

[7 Annexes 4](#_Toc41496078)

[7.1 Planning prévisionnel 4](#_Toc41496079)

[7.2 Planning réel 4](#_Toc41496080)

# Introduction

Ce document présente les différents aspects de la conception du projet CVRP Viewer. Ce projet a été développé pendant TPI (Travail pratique individuel) du 25 mai au 9 juin 2020.

CVRP Viewer est une application Windows Form qui visualise et résolue le « Capacitated Vehicle Routing Problem ». Qu’est-ce qu’un CVRP ? C’est un problème contenant un dépôt (point de départ), une liste de clients à différentes positions et des camions avec une capacitée maximale. Pour résoudre ce problème il faut trouver les chemins les moins couteux, (sachant qu’une unité de distance est égale à un franc).

# Rappel de l'énoncé

## Objectifs du projet

L’objectif du projet est de pouvoir parser un fichier de type .vrp ou .dat, importer les données dans un objet puis trouver les chemins optimaux en affichant chaque étapes sur la fenêtre et afficher le coût total des trajets optimaux.

## Inventaire du matériel

* Un ordinateur

## Inventaire des logiciels

* Windows 10
* Microsoft Word (Documentation Technique, Manuel Utilisateur & Rapport TPI)
* Microsoft Excel (Planning)
* C# avec Visual Studio 2019

# Analyse fonctionnelle

## Fonctionnement vu par l’utilisateur

L’utilisateur, à l’ouverture du programme, est accueilli par une fenêtre avec des cercles vides pour représenter les clients, un cercle rempli pour le dépôt et des tournées déjà existant créées aléatoirement. Chaque client d’une tournée est relié par un trait avec son précédent et son prochain, sauf entre le dépôt et le premier et dernier client de la tournée. Il y aura aussi un bouton « Start » pour débuter l’optimisation.

## Description de l’utilisation et des fonctionnalités

Quand l’utilisateur appuiera sur le bouton « Start » l’optimisation commencera et il pourra voir un rafraichissement des tournées à chaque étape de cette procédure, jusqu’à la fin du programme. Le coût total de la meilleure solution trouvée sera affiché dans la barre de titre.

# Analyse organique

## Fonctionnement interne

### Classes

### Format des fichiers

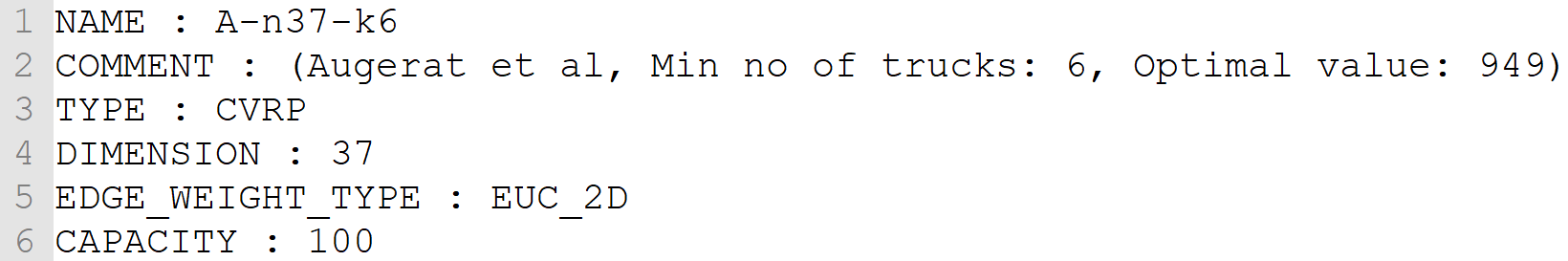
Dans ce projet il y a deux types de fichier (.vrp et .dat) à parser chaque un avec son propre format. Quand la méthode reçoit le fichier, il regarde son extension puis continue avec le parser dont il a besoin.

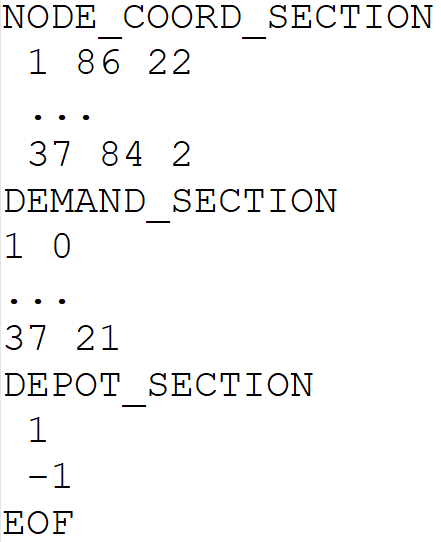
#### Fichier .vrp

Les fichiers .vrp débutes avec les spécifications puis les données.

Les spécifications sont sous la forme *<keyword> : <value>*.

1. NAME : contient le nom du fichier.
2. COMMENT : contient l’auteur du problème ou de la solution, le nombre de camions utilisé et la valeur optimale de la solution.
3. TYPE : spécifie le type de donnée (dans mon cas cela sera toujours *CVRP* ).
4. DIMENSION : dans le cas du *CVRP* cela indique le nombre de nodes avec le dépôt.
5. EDGE\_WEIGHT\_TYPE : cela nous indique comment calculer les distances (dans mon cas je peux simplement utiliser le théorème de Pythagore).
6. CAPACITY : ce nombre indique combien de marchandise les camions peuvent transporter.



Fig. 1 Exemple de spécification pour un fichier .vrp

Les données sont composées de trois sections :

1. NODE\_COORD\_SECTION : chaque ligne de cette section est composée de l’index du node puis ses coordonnés X et Y.
2. DEMAND\_SECTION : cette section est composée de lignes contenant l’index du node et la quantité demandée.
3. DEPOT\_SECTION : cela indique quelles nodes peuvent être le dépôt, cette liste est terminée par le -1.

La fin de la section de données est définie par *EOF.*

#### Fichier .dat

Fig. 2 Exemple de données dans un fichier .vrp

Les fichiers .dat sont composés d’une seule section. Chaque ligne est composée de l’index du node, suivi par les coordonnées X et Y, puis quatre nombres désignant les fenêtres de livraison (ces données sont inutiles car le temps n’est pris en compte dans mon cas), ensuite la quantité demandée et finalement le temps qu’il prend pour servir le node et si on doit livrer (=0) ou ramasser (=1).

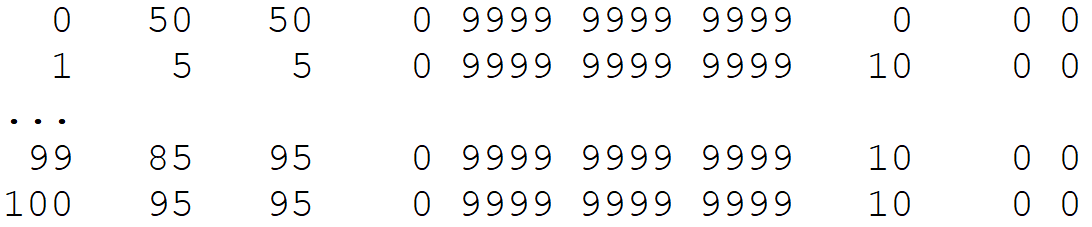


Fig. 3 Exemple de données dans un fichier .dat

## Architecture

## Fenêtre de l’application

# Tests

## Conditions de tests

## Plan de test

## Rapport de tests

# Conclusion

## Difficultés rencontrées

## Améliorations possibles

## Bilan personnel

# Annexes

## Planning prévisionnel

## Planning réel