Combinatorial Optimization Problem: Model and Implementation

Dang Xuan Vuong, Le Anh Tuan, Le Van Cuong, Pham Quang Dung

December 14, 2019

Contents

Chapter 1

Introduction

This document describes combinatorial optimization problems including models (mixed integer programming, constraint programming) and its implementation by MIP solver, CP solver, CBLS solver, etc.

Chapter 2

Case study

2.1 Capacitated Vehicle Routing Time Windows Problem

2.1.1 Problem description

The are N delivery requests. The *i-th* request require us to ship an item with a weight of *w(i)* to location *i* in period of time beetween *e(i)* and *l(i)*. Unloading item *i* require a period of time equal to *d(i)*. We use K trucks for delivering. The *k-th* truck has capacity *c(k)*, start working time at *e(k)* and finish working time at *l(k)*. Travel distance and travel time between location *i* and location *j* respectively are *d(i, j)* and *t(i, j)*. We need to establish a route for the trucks to meet all the requests so that the total travel distance of the trucks is minimal.

2.1.2 General Model

Graph $G = (V, E)$. Set of vertices $V = \{0, 1, 2, \ldots, N\}$ represents depot (vertex 0) and locations to delivery (vertex $1, 2, \ldots, N$). Each vertex in $\{1, 2, \ldots, N\}$ has:

- $w(i)$: weight of item.

- $e(i)$: the earliest delivery time.
- $l(i)$: the latest delivery time.
- $d(i)$: delivering duration.

Set of edges $E = \{(i, j) \mid i, j \in V\}$ represents paths between 2 locations (or depot and a location). Each edge $(i, j) \in E$ has:

- $d(i, j)$: travel distance between location (or depot) i and location (or depot) j.
- $t(i, j)$: travel time between location (or depot) i and location (or depot) j.

There are K trucks for delivering. Truck k ($k = 1, 2, \ldots, K$) has

- $c(k)$: capacity of the truck (total weight of items on the truck cannot exceed its capacity).
- $e(k)$: start working time.
- $l(k)$: end working time.

We need to find a route for trucks on graph G that satisfies the conditions:

- The *i-th* item is delivered to location i from $e(i)$ to $l(i)$, including unloading time $d(i)$.
- The *k-th* truck start working from vertex 0 after $e(k)$ and go back vertex 0 before $l(e)$.
- The *k-th* truck never carry over $c(k)$ of item weight.
- Total travel distance of the trucks is minimal.

2.1.3 Simplify the problem

Before going into specific solving models, we will simplify the problem by adding some hypotheses.

- Each request i is served by only one truck k. So, we can't split item into multiple pieces and the trucks can't exchange item in working period.
- Each truck k leave depot only once after starting working time and go back before finishing working time.
- Direct path between location i and location j is the best choice (in both sense of distance and time travel) if we want to travel from i to j.

2.1.4 Constrained Programming Model

To create this below Constrained Programming Model formulations, we need to include hypotheses at section *2.1.3*:

Variables

- $X(k,i,j) \in \{0,1\}$ in which $X(k,i,j) = 1$ indicates that truck k traverse from point i to point j $(\forall k \in \{1,\dots,K\}, i,j \in \{0,1,\dots,N\})$.
- $Z(k,i) \in \{0,1\}$ in which $Z(k,i) = 1$ indicates that truck k serve $i-th$ request $(\forall k \in \{1,\dots,K\}, i,j \in \{0,1,\dots,N\})$.
- $Y_a(k,i)$: the time when truck k arrives location i.
- $Y_s(k,i)$: the time when truck k truck start delivering at location i.
- $Y_d(k,i)$: the time when truck k truck leaves location i.

Constraints

$$X(k,i,j) = 1 \implies Y_d(k,i) + t(i,j) = Y_a(k,j) \tag{2.1}$$

$$Z(k,i) = 1 \implies Y_s(k,i) + d(i) = Y_d(k,i) \tag{2.2}$$

$$Z(k,i) = 1 \implies Y_s(k,i) \geq Y_a(k,i) \tag{2.3}$$

$$Z(k,i) = 1 \implies Y_s(k,i) \geq e(i) \tag{2.4}$$

$$Z(k,i) = 1 \implies Y_a(k,i) \leq l(i) \tag{2.5}$$

$$\sum_i X(k,i,j) = Z(k,j), \forall k,j \tag{2.6}$$

$$\sum_k Z(k,j) = 1, \forall j \tag{2.7}$$

$$\sum_i X(k,i,j) = \sum_i (K,j,i), \forall j,k \tag{2.8}$$

$$\sum_i X(k,i,0) \leq K, \forall k \tag{2.9}$$

$$Y_d(k,0) \geq e(k) \tag{2.10}$$

$$X(k,i,0) = 1 \implies Y_d(k,i) + t(i,0) \leq l(k), \forall k,i \tag{2.11}$$

$$\sum_i Z(k,i) \times w(i) \leq c(k), \forall k \tag{2.12}$$

Objective functions

$$\sum_{k,i,j} X(k,i,j) \times d(i,j) \rightarrow min \tag{2.13}$$

2.1.5 Mixed Integer Programming Model

We also use Constrained Programming Model formulations at section *2.1.4* with transforming inferred relation constraints to create below Mixed Integer Programming Model formulations.

Constraints

- Suppose we has a very large constant M and an infinitesimal positive constant ϵ.
- Constraint *2.1* is equivalent with 2 below constraints:

$$Y_d(k,i) + t(i,j) + M \times (X(k,i,j) - 1) \geq Y_a(k,j) \tag{2.14}$$

$$Y_d(k,i) + t(i,j) + M \times (X(k,i,j) - 1) \leq Y_a(k,j) \tag{2.15}$$

- Constraint *2.2* is equivalent with 2 below constraints:

$$Y_s(k,i) + d(i) + M \times (X(k,i,j)) \geq Y_d(k,i) \tag{2.16}$$

$$Y_s(k,i) + d(i) + M \times (X(k,i,j)) \leq Y_d(k,i) \tag{2.17}$$

- Constraint *2.3* is equivalent with below constraint:

$$Y_s(k,i) \times Z(k,i) \geq Y_a(k,i) \times Z(k,i) \tag{2.18}$$

- Constraint *2.4* is equivalent with below constraint:

$$Y_s(k,i) \times Z(k,i) \geq e(i) \times Z(k,i) \tag{2.19}$$

- Constraint *2.5* is equivalent with below constraint:

$$Y_a(k,i) \times Z(k,i) \leq l(i) \times Z(k,i) \tag{2.20}$$

- Constraint *2.11* is equivalent with below constraint:

$$(Y_d(k,i) + t(i,0)) \times X(k,i,0) \leq l(k) \times X(k,i,0) \tag{2.21}$$