Vehicle Routing Problem with Capacity limitation and Time window

Chun Lin, Chien

Introduction – Nodes, Routes, Distances, Demands

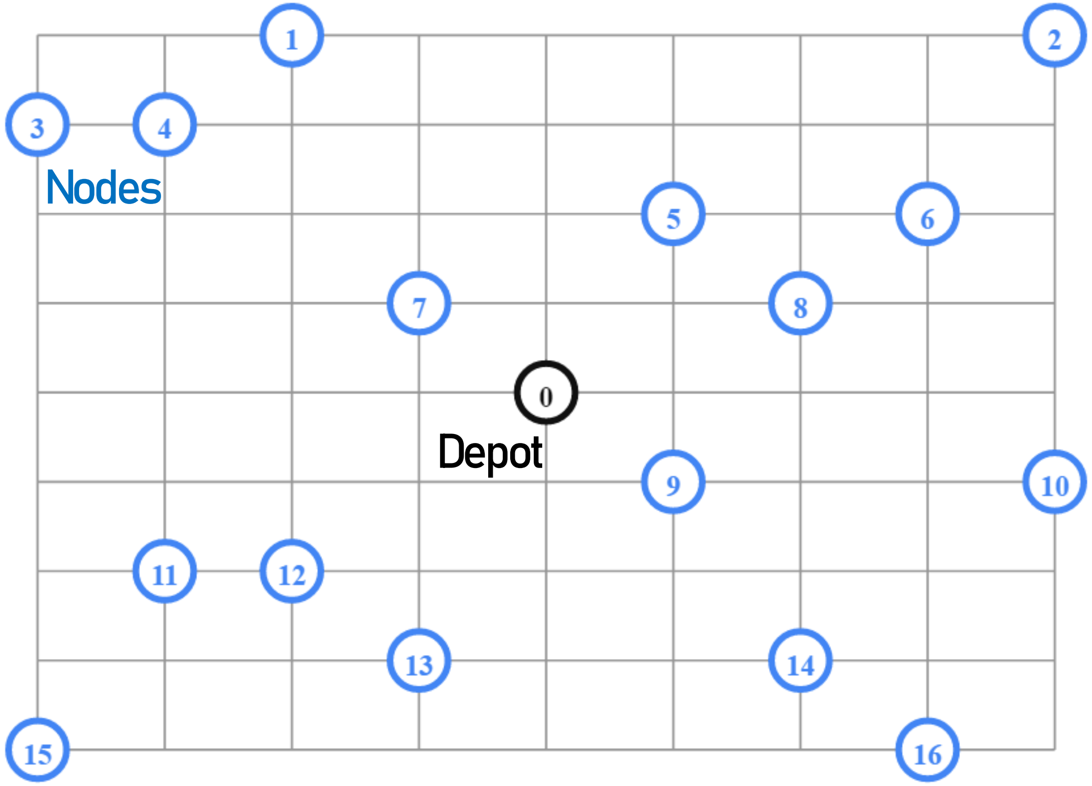

Image resource: Google OR-Tools

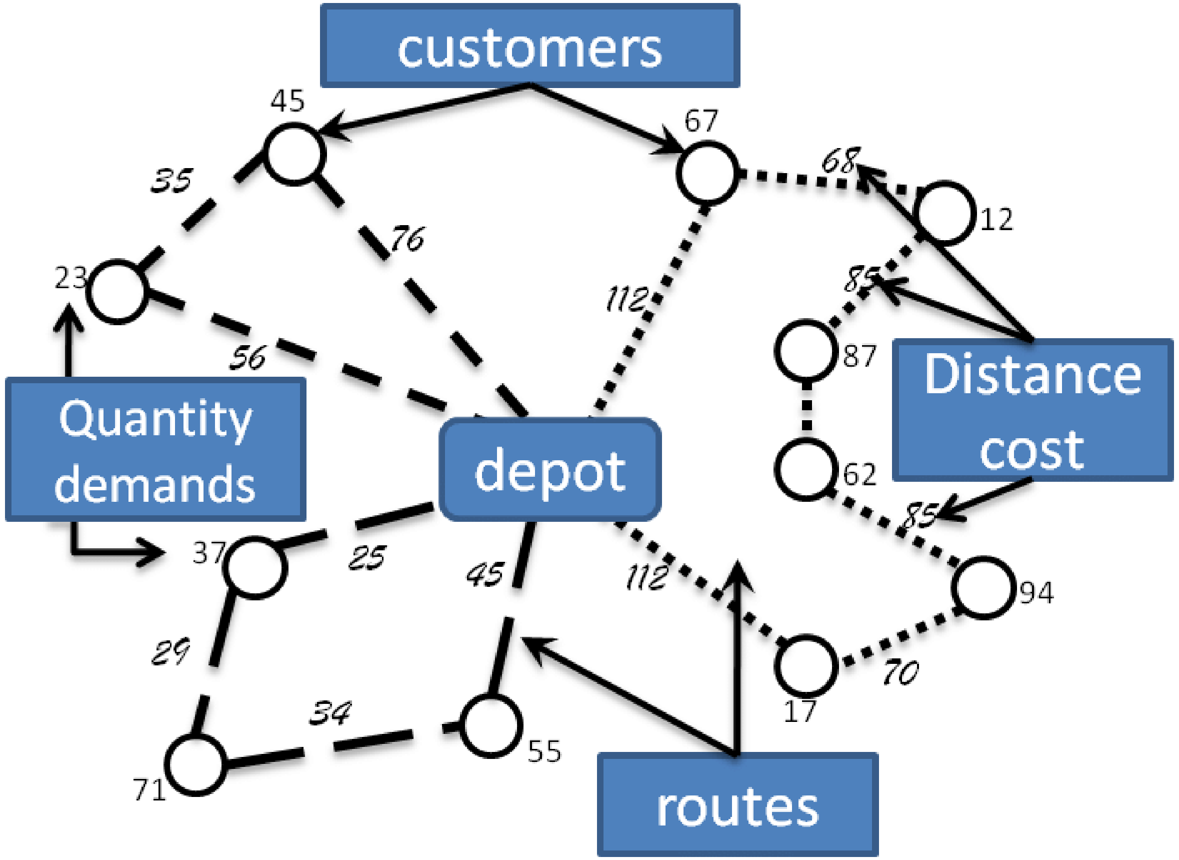

Image resource: Konstantinidis et al. (2014)

Introduction – Capacity Limitation

Without capacity limitation

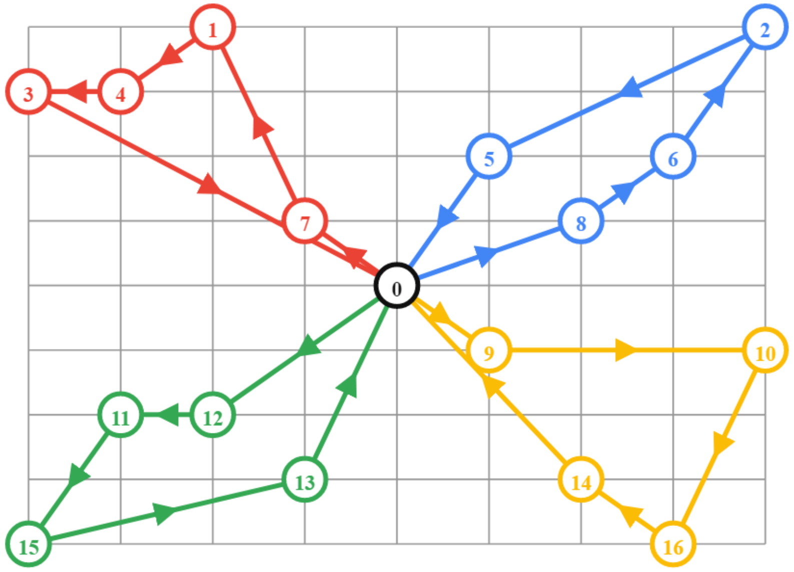

Image resource: [Google OR-Tools]

- **Distance of Vehicle 1:** 1552 m
- **Distance of Vehicle 2:** 1552 m
- **Distance of Vehicle 3:** 1552 m
- **Distance of Vehicle 4:** 1552 m

➢ **Total Distance:** 6208 m

With capacity limitation

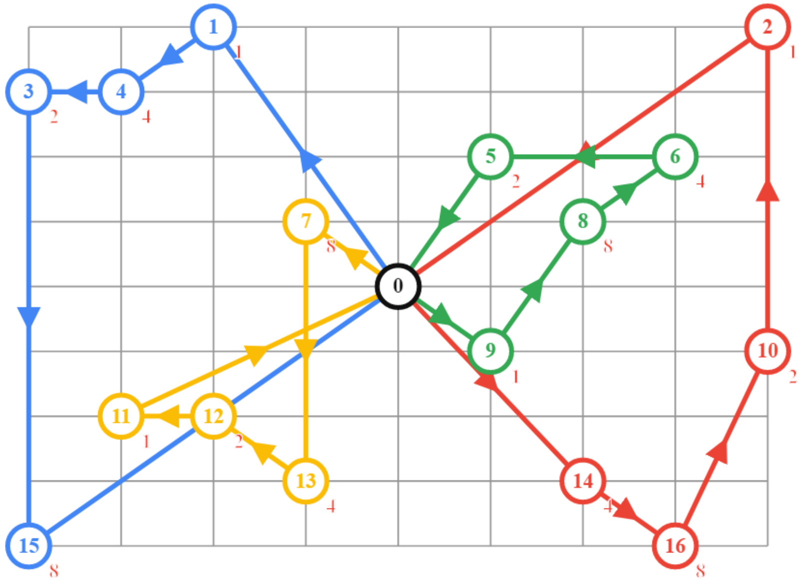

Image resource: [Google OR-Tools]

- **Distance of Vehicle 1:** 2192 m
- **Distance of Vehicle 2:** 2192 m
- **Distance of Vehicle 3:** 1324 m
- **Distance of Vehicle 4:** 1165 m

➢ **Total Distance:** 6872 m

Introduction – Time Window

1. Hard Time Window

   Like this one →

2. Soft Time Window

- Define the accepted range of mismatch arrived time and setup the penalty.

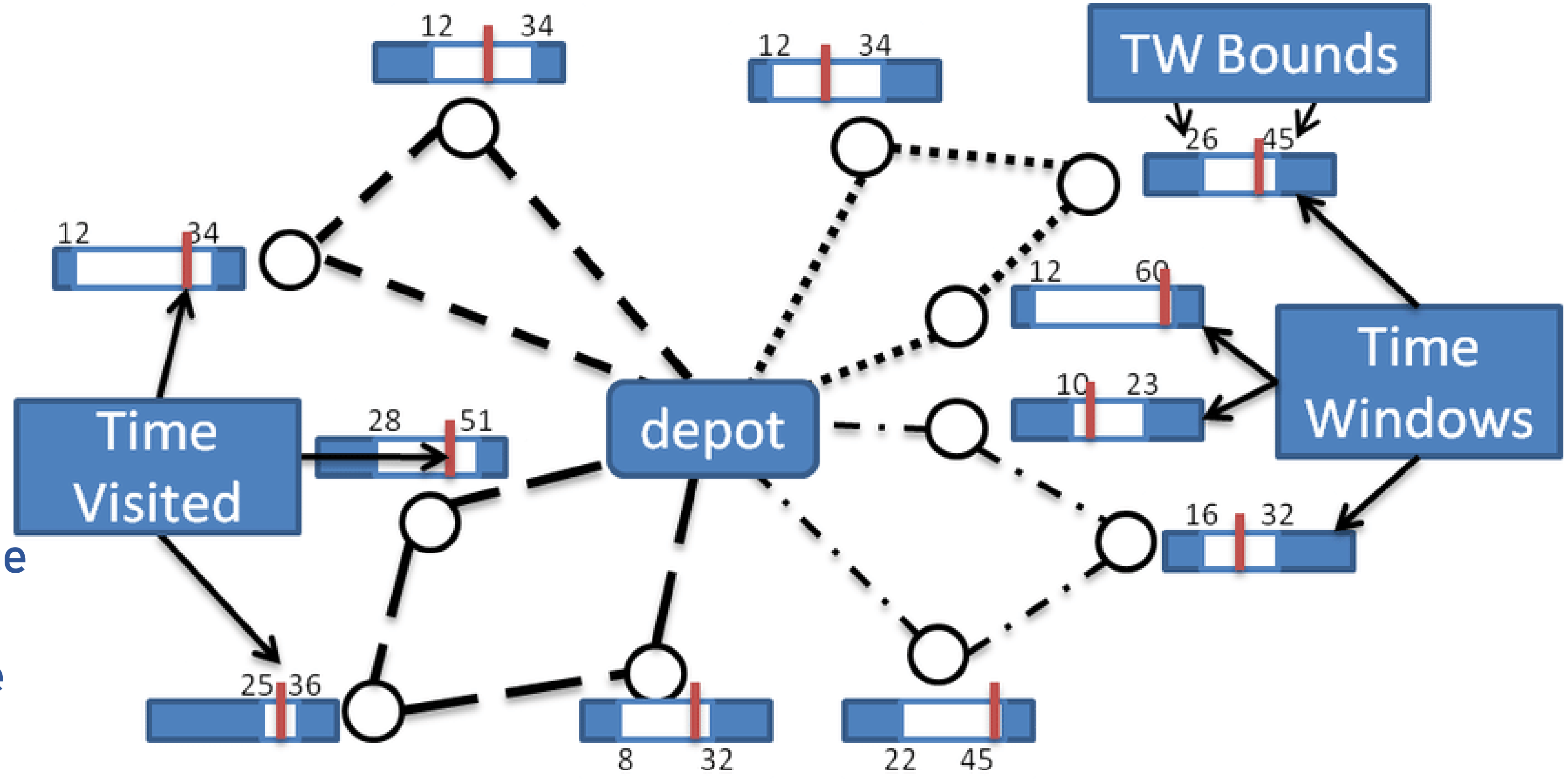

Image resource: Konstantinidis et al. (2014)

Introduction – Python

1. Syntax

● Dynamic Programming Language (High Flexibility)

● Interpreted language (Low speed)

2. Varibale

- String → 'Hello World' -- convert function: str()
- Integer → 12345 -- convert function: int()
- Float → 123.45 -- convert function: float()

3. Data Type

- Tuple, List, Dictionary, Set → (unchangeable), [ ], { 'key' : value }, {no sequence}

4. Comment

Hello ~ This is a comment.

Introduction – PuLP (Python Linear Programming module)

1. Initial Problem

Code: prob = LpProblem(‘Problem name’ , LpMinimize / LpMaximum)

2. Create Variable

Code: x = LpVariable(“x", lowerBound=0, upperBound=1, cat=‘Binary’)

Code: y = LpVariable(“y", lowerBound=0, upperBound=1, cat=‘Binary’)

3. Define Objective Function (expression)

Code: prob += -12*x + 24*y

4. Setup Constraints

Code: prob += x + y <= 1

↓ Additional Solver :

Reference: https://pypi.org/project/PuLP/

Problem – Capacitated Vehicle Routing Problem

- n is the number of clientes
- N is set of clients, with $N = \{1, 2, \ldots, n\}$
- V is set of vetices (or nodes), with $V = \{0\} \cup N$
- A is set of arcs, with $A = \{(i, j) \in V^2 : i \neq j\}$
- c_{ij} is cost of travel over arc $(i, j) \in A$
- Q is the vehicle capacity
- q_i is the amount that has to be delivered to customer $i \in N$

$$
\begin{array}{lll}
\min & \sum_{i,j \in A} c_{ij} x_{ij} & \\
\text{s.t.} & \sum_{j \in V, j \neq i} x_{ij} = 1 & i \in N \\
& \sum_{i \in V, i \neq j} x_{ij} = 1 & j \in N \\
& \text{if } x_{ij} = 1 \Rightarrow \boxed{u_i} + q_j = \boxed{u_j} & i, j \in A : j \neq 0, i \neq 0 \\
& q_i \leq \boxed{u_i} \leq Q & i \in N \\
& x_{ij} \in \{0, 1\} & i, j \in A
\end{array}
$$

Dummy Variable :

- u_i is the accumulated delivers at N_i.

Binary → Go or Not

Reference: CPLEX & Python. CVRP

Problem – (Multi) Capacitated Vehicle Routing Problem

- $Node = \{1, 2, 3, 4, 5, 6, 7, 8\}$ → Set of the index of each city.
- $Mode = \{1, 2, 3, 4, 5\}$ → Set of the index of each vehicle.
- x_{ijm} is the states of route (i, j) served by vehicle m, $\forall\, i \neq j$. (Decision Variable – Go or not)
- C_{ijm} is the cost of route (i, j) served by vehicle m, $\forall\, i \neq j$.
- Q_i is the demand volume of city i.
- V_m is the capacity of vehicle m.
- u_{im} is the accumulated delivers at city i served by vehicle m. (Dummy Variable)

$$Minimize\ Z = \sum_{i,j\,\in\,Node} \sum_{m\,\in Mode} C_{ijm} x_{ijm}$$

s.t.

For speeding up

(1) $\sum_{j\,\in\,Node} \sum_{m\,\in Mode} x_{ijm} \geq 3, \quad \forall\, i = 1$

(2) $\sum_{j\,\in\,Node} x_{ijm} \leq 1, \quad \forall\, i = 1, m \in Mode$

(3) $\sum_{j\,\in\,Node} x_{ijm} - \sum_{j\,\in\,Node} x_{jim} = 0, \quad \forall\, i = 1, m \in Mode$

(4) $\sum_{j\,\in\,Node} \sum_{m\,\in Mode} x_{ijm} = 1, \forall\, i \in Node, m \in Mode, if\ Q_i + Q_j \leq V_m$

(5) $\sum_{i\,\in\,Node} \sum_{m\,\in Mode} x_{ijm} = 1, \forall\, j \in Node, m \in Mode\, , if\ Q_i + Q_j \leq V_m$

(6) $\sum_{m\,\in\,Mode} u_{im} \geq Q_i, \quad \forall\, i \in Node, i \neq 1$

(7) $u_{im} \leq V_m, \quad \forall\, i \in Node, i \neq 1, m \in Mode$

Important !!

(8) $\boldsymbol{u_{jm} - u_{im} + (V_m - (Q_i + Q_j)) \cdot x_{jim} + V_m \cdot x_{ijm} \leq V_m - Q_i,}$
$\boldsymbol{\forall\, i, j \in Node, i, j \neq 1, m \in Mode}$

(9) $u_{jm} + \left(V_m - Q_j\right) \cdot x_{ijm} \leq V_m, \quad \forall\, i = 1, j \in Node, m \in Mode$

(10) $\sum_{m\,\in Mode} u_{jm} - \sum_{i\,\in Node} \sum_{m \in Mode} x_{ijm} \cdot Q_i \geq Q_j,$
$\forall\, i \neq 1, j \in Node$

Problem – CVRP (multi capacity) with Time Window

- $Node = \{1, 2, 3, 4, 5, 6, 7, 8\}$ → Set of the index of each city.
- $Mode = \{1, 2, 3, 4, 5\}$ → Set of the index of each vehicle.
- x_{ijm} is the states of route (i, j) served by vehicle m, $\forall\, i \neq j$. (Decision Variable – Go or not)
- C_{ijm} is the cost of route (i, j) served by vehicle m, $\forall\, i \neq j$.
- Q_i is the demand volume of city i.
- V_m is the capacity of vehicle m.
- u_{im} is the accumulated delivers at city i served by vehicle m. (Dummy Variable)
- E_i is the earliest time window of city i.
- L_i is the latest time window of city i.
- S_i is the visit duration at city i.
- t_i is the accumulated delivery time at city i. (Dummy Variable)

- D_{ij} is the distance between city i and j.
- T_{max} is the maximum drive time(99999).
- Trv_{rate} is the travel rate (1.2).

$$\textit{Minimize } Z = \sum_{i,j \in Node} \sum_{m \in Mode} C_{ijm} x_{ijm}$$

s.t.

(1) – **(10)** are as same as the constraints in the previous version (i.e., (Multi) Capacitated Vehicle Routing Problem).

$$(11) \quad t_j \geq t_i + \left((S_i + Trv_{rate} \cdot D_{ij}) \cdot \sum_{m \in Mode} x_{ijm} - L_i \cdot \left(1 - \sum_{m \in Mode} x_{ijm}\right)\right), \quad \forall\, i, j \in Node, \quad j \neq 1$$

$$(12) \quad t_i \geq E_i, \quad \forall\, i \neq 1 \qquad (13) \quad t_i \leq L_i, \quad \forall\, i \neq 1 \qquad (14) \quad t_i + S_i + Trv_{rate} \cdot D_{ij} \cdot \sum_{i \in Node} \sum_{m \in Mode} x_{ijm} \leq T_{max}, \quad \forall\, j = 1$$