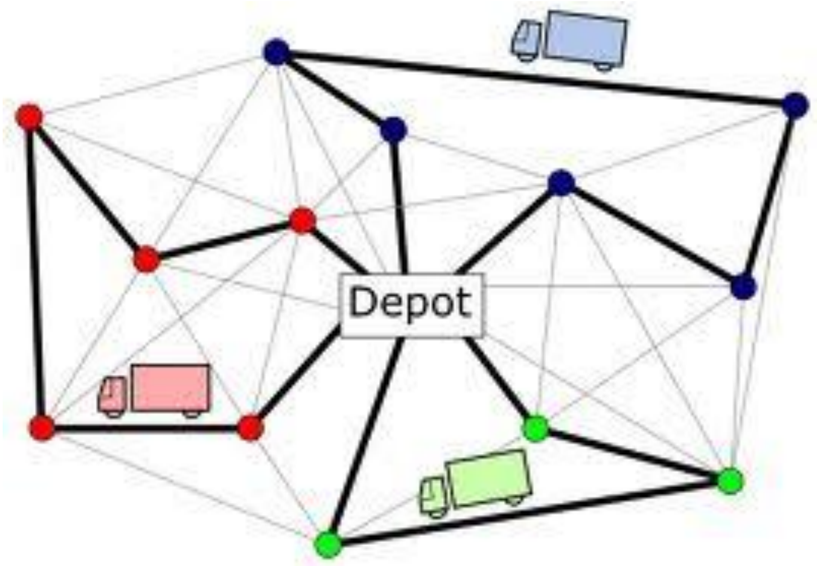
SISTEM TRANSPORTASI DAN DISTRIBUSI BARANG

Model Matematis dan Algoritma Heuristik Permasalahan Perutean Kendaran (VRP)



Muhammad Nashir Ardiansyah, S.T., M.T., Ph.D.

Program Studi S1 Teknik Industri – Telkom University

Image Source: https://www.r-bloggers.com/any-r-packages-to-solve-vehicle-routing-problem/

Penggunaan Solver untuk Penyelesaian Masalah VRP

Penyelesaian Model Matematis VRP

- Model matematis berfungsi untuk mendefinisikan masalah VRP secara matematis.
- Model matematis dapat diselesaikan dengan menggunakan perangkat lunak mathematical solver yang berisi algoritma branch and bound dan berbagai metode cutting plane dan dekomposisi
- Contoh Mathematical Solver:
  - Excel Solver
  - IBM CPLEX
  - Gurobi

Gurobi Solver dan Bahasa Pemrograman Python

- Parameter adalah data yang didefinisikan/tersedia diawal
- Pendefinisian parameter

Parameter:

C_{ij}: cost of transporting from city i to city j.

G_j: load of node j

Node	X	Y	Demand
0	224	168	0
1	260	402	39
2	154	205	30
3	173	315	31
4	299	410	67
5	347	275	83
6	399	133	39
7	29	175	89
8	258	63	89
9	54	66	77
10	61	333	96

```
#Parameter T
T = {}
T[0,0]=9999; T[0,1]=237; T[0,2]=79; T[0,3]=156; T[0,4]=253; T[0,5]=163; T[0,6]=178; T[0,7]=195; T[0,8]=110; T[0,9]=198; T[0,10]=232;
T[1,0]=237; T[1,1]=9999; T[1,2]=224; T[1,3]=123; T[1,4]=40; T[1,5]=154; T[1,6]=303; T[1,7]=324; T[1,8]=339; T[1,9]=394; T[1,10]=211;
T[2,0]=79; T[2,1]=224; T[2,2]=9999; T[2,3]=112; T[2,4]=251; T[2,5]=205; T[2,6]=255; T[2,7]=129; T[2,8]=176; T[2,9]=171; T[2,10]=158;
T[3,0]=156; T[3,1]=123; T[3,2]=112; T[3,3]=9999; T[3,4]=158; T[3,5]=179; T[3,6]=290; T[3,7]=201; T[3,8]=266; T[3,9]=276; T[3,10]=113;
T[4,0]=253; T[4,1]=40; T[4,2]=251; T[4,3]=158; T[4,4]=9999; T[4,5]=143; T[4,6]=294; T[4,7]=358; T[4,8]=349; T[4,9]=422; T[4,10]=250;
T[5,0]=163; T[5,1]=154; T[5,2]=205; T[5,3]=179; T[5,4]=143; T[5,5]=9999; T[5,6]=151; T[5,7]=333; T[5,8]=230; T[5,9]=360; T[5,10]=292;
T[6,0]=178; T[6,1]=303; T[6,2]=255; T[6,3]=290; T[6,4]=294; T[6,5]=151; T[6,6]=9999; T[6,7]=372; T[6,8]=157; T[6,9]=351; T[6,10]=393;
T[7,0]=195; T[7,1]=324; T[7,2]=129; T[7,3]=201; T[7,4]=358; T[7,5]=333; T[7,6]=372; T[7,7]=9999; T[7,8]=255; T[7,9]=112; T[7,10]=161;
T[8,0]=110; T[8,1]=339; T[8,2]=176; T[8,3]=266; T[8,4]=349; T[8,5]=230; T[8,6]=157; T[8,7]=255; T[8,8]=9999; T[8,9]=204; T[8,10]=334;
T[9,0]=198; T[9,1]=394; T[9,2]=171; T[9,3]=276; T[9,4]=422; T[9,5]=360; T[9,6]=351; T[9,7]=112; T[9,8]=204; T[9,9]=9999; T[9,10]=267;
T[10,0]=232; T[10,1]=211; T[10,2]=158; T[10,3]=113; T[10,4]=250; T[10,5]=292; T[10,6]=393; T[10,7]=161; T[10,8]=334; T[10,9]=267; T[10,10]=9999;

#parameter G
G = {'D': 640.0, 'C9': 77.0, 'C8': 89.0, 'C3': 31.0, 'C2': 30.0, 'C1': 39.0, 'C10': 96.0, 'C7': 89.0, 'C6': 39.0, 'C5': 83.0, 'C4': 67.0}
```

Gurobi Solver dan Bahasa Pemrograman Python

- Pendefinisian variable

Variable:
x_{ij}^{v}: binary variable to indicate the path that goes from city i to city j by vehicle v is taken.
g_{j}^{v}: accumulated vehicle load v at city j

```
m = Model("crowdsourching_trans")
#-------------------------------------------------
#VARIABLE VALUE
for i in Nd:
    for j in Nd:
        for v in V:
            x[i,j,v] = m.addVar(vtype="B" , name="x[%s,%s,%s]"%(i,j,v))

for i in Nd:
    for v in V:
        g[i,v] = m.addVar(vtype="C" , name="g[%s,%s]"%(i,v))

m.update()
```

- Pendefinisian fungsi objektif

$$min \sum_{i \in N} \sum_{j \in N} \sum_{v \in V} x_{ij}^{v} C_{ij}$$

```
#-------------Objective Function-------------------------
m.setObjective(
    quicksum(x[i,j,v] * D[i,j] for i in Nd for j in Nd for v in V)
)
m.ModelSense = 1 #MINIMIZATION
```

Gurobi Solver dan Bahasa Pemrograman Python

- Pendefinisian pembatas

$$\sum_{j \in N} \sum_{v \in V} x_{ij}^{v} = 1, \qquad \forall i \in N$$

$$\sum_{i \in N} \sum_{v \in V} x_{ij}^{v} = 1, \qquad \forall j \in N$$

$$\sum_{i \in N} x_{ik}^{v} - \sum_{j \in} x_{kj}^{v} = 0, \forall k \in N \cup M, v \in V$$

```
#2
for i in N:
    m.addConstr(
        quicksum(x[i,j,v] for j in Nd for v in V) == 1
        , "1[%s]"%i)

#3
for j in N:
    m.addConstr(
        quicksum(x[i,j,v] for i in Nd for v in V) == 1
        , "2[%s]"%j)

#4
for k in N:
    for v in V:
        m.addConstr(
            quicksum(x[i,k,v] for i in Nd) - quicksum(x[k,j,v] for j in Nd) == 0
            , "3[%s,%s]"%(k,v))
```

Gurobi Solver dan Bahasa Pemrograman Python

- Pendefinisian pembatas

$$g_j^v \geq g_i^v + \left(x_{ij}^v - 1\right)Z + G_j, \forall v \in V, i, j \in N$$
$$g_j^v \leq H, \forall v \in V, j \in N$$

```
for i in Nd:
    for j in N:
        for v in V:
            m.addConstr(
                g[j,v] >= g[i,v] + G[j] + (x[i,j,v] - 1) * Z
                , "5[%s,%s,%s]"%(i,j,v))

for i in Nd:
    for v in V:
        m.addConstr(
            g[i,v] <= Q
            , "6[%s,%s]"%(i,v))
```