CS 524: Introduction to Optimization
Lecture 23 : Traveling salesman

Michael Ferris

Computer Sciences Department
University of Wisconsin-Madison

October 27, 2023

Traveling Salesman

- A traveling salesman must visit all his cities at minimum cost
- Must also start and end at home

Given

- Set of cities N
- Distances between cities c_{ij}

- This is the most famous combinatorial optimization problem
- There are lots of applications

$$x_{ij} = \begin{cases} 1 & \text{We travel from city } i \text{ to city } j \\ 0 & \text{Otherwise} \end{cases}$$

A Formulation for TSP?

- Consider the following

$$
\begin{aligned}
\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} & \\
\sum_{i \in N} x_{ij} = 1 \quad & \forall j \in N \\
\sum_{j \in N} x_{ij} = 1 \quad & \forall i \in N \\
x_{ij} \in \{0, 1\} &
\end{aligned}
\tag{1}
$$

- Is this a valid TSP formulation?

10 City USA Instance

```
table      dist(i,j)      "distances"
            Atlanta Chicago   Denver Houston LosAngeles Miami NewYork SanFrancisco Seattle WashingtonDC
Atlanta       0       587     1212    701       1936     604     748      2139        2182     543
Chicago      587       0       920    940       1745     1188    713      1858        1737     597
Denver       1212     920       0     879        831     1726    1631      949        1021     1494
Houston      701      940      879     0        1372     968     1420     1645        1891     1220
LosAngeles   1936     1745     831    1374        0      2339    2451      347         959     2300
Miami        604      1188     1726   968       2339     0       1092     2594        2734     923
NewYork      748      713      1631   1420      2451     1092     0        2571        2408     205
SanFrancisco 2139     1858     949    1645       347     2594    2571       0          678     2442
Seattle      2182     1737     1021   1891       959     2734    2408      678          0      2329
WashingtonDC  543     597      1494   1220      2300     923      205      2442        2329     0 ;
```

Subtours

- How do we get rid of those pesky subtours?
- For every set S of cities except the whole set, we can use at most $|S| - 1$ edges.

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \quad \forall S \subseteq N, 2 \leq |S| \leq |N| - 1$$

- There are exponentially many "subtour elimination" constraints

Quiz Time!

1018517988167243043134222844204689080525734196832968125318070224677190649881668353091698688.

- Is this...
  - (a) The number of gifts that I have bought my wife?
  - (b) The number of subatomic particles in the universe?
  - (c) The number of "subtour elimination constraints" when $|N| = 299$.
  - (d) All of the above?
  - (e) None of the above?

Answer Time!

- The answer is (e). (a)–(c) are all too small (as far as I know) `:-)`. (It is (c), for $|N| = 300$).
- "Exponential" is **really** big.
- Yet people have solved TSP's with $|N| > 80,000$!
- How? You ask...
- Branch and cut! (`www.tsp.gatech.edu`)
  - Solve the problem without all of those subtours, and then just add them as we need them
- Finding the "add if we need them" subtours is called the *separation problem*
- Let's finish solving our example...
  - See tsp1.gms in GAMS model library for fancier code: doing it dynamically (Other examples: tsp2,tsp3,tsp4,tsp5,swath) or the MIRO tsp app.

Miller Tucker Zemlin

- There is a weaker, but more compact formulation that should suffice to eliminate subtours for smallish instances.
- We can derive it using the "Turn on constraint' idea'.
- Idea: Let $u_i, i \in N$ be the relative positive of node i in the tour. Choose node 1 to be first (arbitrarily) $u_1 = 1$.
- $2 \leq u_i \leq n = |N| \quad \forall i \in N \setminus \{1\}$
- Just model implication (assuming $x_{ii} = 0$ is fixed):

$$x_{ij} = 1 \Rightarrow u_j \geq u_i + 1 \quad (i.e. u_i - u_j + 1 \leq 0) \quad \forall i \neq 1, \forall j \neq 1$$

- Since $g(u) = u_i - u_j + 1$, it is clear that $M = n - 2 + 1 = n - 1$
- This gives inequalities

$$u_i - u_j + 1 \leq (n-1)(1 - x_{ij}) \quad \forall i \neq 1, \forall j \neq 1$$

The formulation (tsp10.gms)

To exclude subtours, use extra variables $u_i (i = 1, \ldots, n)$, and constraints

$$\begin{aligned} & u_1 = 1, \\ & 2 \leq u_i \leq n, \quad \forall i \neq 1, \\ & u_i - u_j + 1 \leq (n-1)(1 - x_{ij}), \quad \forall i \neq 1, \forall j \neq 1. \end{aligned} \tag{2}$$

The formulation consisting of (1) and (2) is called the MillerTuckerZemlin (MTZ) formulation of the TSP. It indeed excludes subtours, as (a) constraint (2) for (i, j) forces $u_j \geq u_i + 1$, when $x_{ij} = 1$; (b) if a feasible solution of (1)-(2) is contained more than one subtour, then at least one of these would not contain node 1, and along this subtour the u_i values would have to increase to infinity. This argument, with the bounds on the u_i variables, also implies that the only feasible value of u_i is the position of node i in the tour.

SOS type 1 variables

Some solvers implement `SOS1` variables, and these simplify the model and allow a solver to perform different branching strategies.

- A set of variables $\{\delta_1, \delta_2, \ldots \delta_m\}$ is called a special ordered set (SOS) of type 1 variables if at most 1 of them is nonzero.

We often use `SOS1` in conjunction with a normalizing constraint that fixes the nonzero value (e.g. $\sum_i \delta_i = 1$ is a popular choice).

Modeling a Restricted Set of Values

- Want variable x to only take on values in $\{a_1, \ldots, a_m\}$.
- We introduce m binary variables $\delta_j, j = 1, \ldots, m$ and the constraints

$$x = \sum_{j=1}^{m} a_j \delta_j,$$

$$\sum_{j=1}^{m} \delta_j = 1, \quad \delta_j \in \{0, 1\}, j = 1, \ldots, m$$

Alternatively, we could replace the last statement:

```
binary variables delta(J);
```

with

```
sos1 variables delta(J);
```

This results in the same model, but solvers may process these variables differently (branching strategy)

Example—Building a warehouse

- Suppose we are modeling a facility location problem in which we must decide on the size of a warehouse to build.
- The choices of sizes and their associated cost are shown below:

Size	Cost
10	100
20	180
40	320
60	450
80	600

Warehouse sizes and costs

Warehouse Modeling (`warehouse.gms`)

- Using binary decision variables $\delta_1, \delta_2, \ldots, \delta_5$, we can model the cost of building the warehouse as

$$\text{COST} \equiv 100\delta_1 + 180\delta_2 + 320\delta_3 + 450\delta_4 + 600\delta_5.$$

- The warehouse will have size (or $\text{SIZE} \geq 35$)

$$\text{SIZE} \equiv 10\delta_1 + 20\delta_2 + 40\delta_3 + 60\delta_4 + 80\delta_5,$$

- and we have the SOS constraint

$$\delta_1 + \delta_2 + \delta_3 + \delta_4 + \delta_5 = 1.$$