Minimize \[ \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij} \]
subject to
\[ \sum_{j=1}^{n} X_{ij} = 1 \text{ for } i=1, \ldots, n \]
\[ \sum_{i=1}^{n} X_{ij} = 1 \text{ for } j=1, \ldots, n \]
\[ X_{ij} \in \{0,1\} \text{ for all } i,j \]

These are the constraints of the assignment problem!
Not all feasible solutions of the assignment problem (AP) are TSP tours!

What constraints can be added to AP in order to eliminate the subtours?

Introduce new variables $u_i, i=1, 2, \ldots n$

For each pair of cities $(i, j), i \neq j$, add the constraint

$$u_i - u_j + nX_{ij} \leq n - 1$$

These constraints will eliminate subtours.

For example: $X_{12} = X_{24} = X_{41} = 1 \Rightarrow$

$$u_1 - u_2 + 5 \leq 4$$
$$u_2 - u_4 + 5 \leq 4$$
$$u_4 - u_1 + 5 \leq 4$$

sum: $15 \leq 12$ \hspace{1cm} infeasible!

These new constraints eliminate subtours of fewer than \( n \) cities, but NOT tours of \( n \) cities:

Let \( u_i = \) sequence \# in which city \( i \) is visited.

So, if \( X_{ij} = 1 \), we have \( u_i - u_j = -1 \)

and so \( u_i - u_j + nX_{ij} = -1 + n \)

i.e., \( u_i - u_j + nX_{ij} \leq n - 1 \) is satisfied!

Dimensions of model:

<table>
<thead>
<tr>
<th>( n(n-1) )</th>
<th>integer variables ( X_{ij} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>continuous variables ( u_i )</td>
</tr>
<tr>
<td>( 2n )</td>
<td>assignment constraints</td>
</tr>
<tr>
<td>( n(n-1) )</td>
<td>subtour elimination constraints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( n ) (#cities)</th>
<th>5</th>
<th>10</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>variables</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0-1 integer</td>
<td>20</td>
<td>90</td>
<td>2450</td>
<td>9900</td>
</tr>
<tr>
<td>continuous</td>
<td>5</td>
<td>10</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>constraints</td>
<td>30</td>
<td>110</td>
<td>2550</td>
<td>10100</td>
</tr>
</tbody>
</table>
Another set of subtour elimination constraints:

Let \( S \) be a nontrivial subset of the cities.

\[
\sum_{i \in S} \sum_{j \in S} X_{ij} \geq 1
\]

insures that there is an edge in the tour which links set \( S \) to the set of cities NOT in \( S \).

If we include such a constraint for every nontrivial subset, we eliminate all subtours.

For example, if \( S \) is the set of cities on a subtour, the constraint is violated!


The subtour elimination constraint is violated, since

\[
X_{41} = X_{51} = X_{61} = X_{71} = X_{42} = X_{52} = X_{62} = X_{72} = X_{43} = X_{53} = X_{63} = X_{73} = 0
\]
Unfortunately, the number of such constraints is exponential in the number of cities \( 2^n - 1 \):

<table>
<thead>
<tr>
<th>n (#cities)</th>
<th>5</th>
<th>10</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>subtour elimination constraints</td>
<td>31</td>
<td>1024</td>
<td>(1.126 \times 10^{15})</td>
<td>(1.268 \times 10^{30})</td>
</tr>
</tbody>
</table>

Another set of subtour elimination constraints

For every nontrivial subset \( S \),

\[
\sum_{i \in S} \sum_{j \in S} X_{ij} \leq |S| - 1
\]

where \( |S| \) is the cardinality of the set \( S \)

The number of such constraints is again \( 2^n - 1 \)
The subtour elimination constraint is violated, since there are three edges in the subtour.

**Summary:** Subtour Elimination Constraints (to be appended to the assignment problem)

\[
\sum_{i \in S} \sum_{j \in S} X_{ij} \geq 1 \quad \text{for all subsets } S \text{ of the nodes}
\]

\[
\sum_{i \in S} \sum_{j \in S} X_{ij} \leq |S| - 1 \quad \text{for all subsets } S \text{ of the nodes}
\]

\[
u_i - u_j + nX_{ij} \leq n - 1 \quad \text{for all edges } (i,j)
\]
Warning!

While subtours are eliminated by any of the three sets of constraints, the smaller set of constraints, i.e.,

\[ u_i - u_j + nX_{ij} \leq n - 1 \]

for all edges \((i, j)\)
yields a much weaker lower bound in an LP relaxation!

One-Tree Constraint

A TSP tour is a special case of a "one-tree", which is a spanning tree with one additional edge included (creating a circuit).
Let $T_1$ denote the set of all one-trees of the network, so that $X \in T_1$ if $X$ is a one-tree. Then the constraint

$$X \in T_1$$

will eliminate subtours, since spanning one-trees are connected!

2-commodity flow model

Suppose that the salesman delivers a full container of some commodity (for example, bottled gas), and picks up an empty container, for each customer.

At all times, he will have a total of $(n-1)$ containers (full plus empty) in his vehicle.
Define **two** sets of **continuous** variables
(one for flow of full containers, and one for flow of empty containers),
plus one set of **binary** variables.

\[
\begin{align*}
Y^p_{ij} &= \text{flow of full containers in edge (i,j)} \\
Y^q_{ij} &= \text{flow of empty containers in edge (i,j)} \\
X_{ij} &= \begin{cases} 1 & \text{if edge (i,j) is on the route} \\ 0 & \text{otherwise} \end{cases}
\end{align*}
\]

**Constraints**

(These guarantee a path from source to each node, & a return path, eliminating subtours.)

\[
\begin{align*}
\sum_{j=1}^{n} Y^p_{ij} - \sum_{k=1}^{n} Y^p_{ki} &= \begin{cases} n-1 & \text{for } i = \text{source} \\ -1 & \text{elsewhere} \end{cases} \\
Y^p_{ij} &\geq 0 \quad \text{conservation of flow for full containers}
\end{align*}
\]

\[
\begin{align*}
\sum_{j=1}^{n} Y^q_{ij} - \sum_{k=1}^{n} Y^q_{ki} &= \begin{cases} -(n-1) & \text{for } i = \text{source} \\ +1 & \text{elsewhere} \end{cases} \\
Y^q_{ij} &\geq 0 \quad \text{conservation of flow for empty containers}
\end{align*}
\]
Since the total flow in each edge of the tour is \( n-1 \), we also have:

\[
Y_{ij}^p + Y_{ij}^Q = (n-1)X_{ij} \quad \text{for each edge } (i,j)
\]

That is, if \( X_{ij} = 0 \), no flow is permitted in edge \((i,j)\) while if \( X_{ij} = 1 \), the total flow is \( n-1 \).

---

Objective

Minimize \( \alpha \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}X_{ij} + \beta \sum_{i=1}^{n-1} \sum_{j=1}^{n} C_{ij} \left( Y_{ij}^p + Y_{ij}^Q \right) \)

where \( \alpha + \beta = 1, \quad \alpha \geq 0 \quad \& \quad \beta \geq 0 \)

For example (\( \alpha = 1, \beta = 0 \)): Minimize \( \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}X_{ij} \)

or (\( \alpha = 0, \beta = 1 \)): Minimize \( \frac{1}{(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} \left( Y_{ij}^p + Y_{ij}^Q \right) \)
This model easily incorporates

**Precedence constraints**

Suppose city \( h \) must precede city \( k \) on the tour. Then the number of "full containers" entering city \( h \) must exceed the number of "full containers" entering city \( k \):

\[
\sum_{i=1}^{n} Y_{ih}^p \geq 1 + \sum_{i=1}^{n} Y_{ik}^p
\]

\[\leftrightarrow \leftrightarrow\]

**(n-1) commodity flow model**

For each city \( k \) (other than the source \( s \)), we define a commodity:

\[Y_{kj}^k = \text{flow in arc } (i,j) \text{ of commodity destined for } k\]
Constraints

Conservation of flow for each commodity $k$:

\[
\begin{align*}
\sum_{j=1}^{n} Y_{sj}^k &= 1 \quad \text{at source } s \\
\sum_{i=1}^{n} Y_{ih}^k &= \sum_{j=1}^{n} Y_{hj}^k \quad \text{at } h \neq s, h \neq k \\
\sum_{i=1}^{n} Y_{ik}^k &= 1 \quad \& \quad \sum_{j=1}^{n} Y_{kj}^k = 0 \quad \text{at destination } k
\end{align*}
\]

Capacity constraints

\[0 \leq Y_{ij}^k \leq X_{ij} \quad \text{for each arc } (i,j)\]

Assignment constraints

\[
\begin{align*}
\sum_{j=1}^{n} X_{ij} &= 1 \quad \text{for all } i \\
\sum_{i=1}^{n} X_{ij} &= 1 \quad \text{for all } j \\
X_{ij} \in \{0,1\} \quad \text{for all } i \& j
\end{align*}
\]
The feasible region of the LP relaxation of the \((n-1)\)-commodity flow model is the same as that of the model with exponentially many subtour elimination constraints!

However, basic feasible solutions of this LP may be fractional, as in the following example.

\[
\begin{array}{c}
1 \\
2 \\
3 \\
4 \\
5 \\
\end{array}
\]

Suppose that \(X_{ij} = \frac{1}{2}\) for each edge below, except for edge \((5,s)\), where \(X_{5,s} = 1\).

Then \(X\) is a basic feasible solution to the LP relaxation of the assignment & subtour elimination constraints.

Can constraints be added to eliminate this solution?
"Mutual Flow Constraints"

The following $n(n-1)/2$ constraints will eliminate the previous solution:

$$\sum_{i=1}^{n} Y_{ik}^j + \sum_{i=1}^{n} Y_{ij}^k = 1 \quad \text{for each pair } j, k \ (j \neq k)$$

(either commodity $j$ flows through node $k$, or commodity $k$ flows through node $j$, but not both!)

Author: Dantzig, G.B., Fulkerson, D.R., and Johnson, S.M.

Title: Solutions of a large scale traveling salesman problem


Notes:

Key: subtour elimination constraints
<table>
<thead>
<tr>
<th><strong>Author</strong></th>
<th>Miller, C.E., Tucker, A.W., and Zemlin, R.A.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
<td>Integer programming formulation of traveling salesman problems</td>
</tr>
<tr>
<td><strong>Pub.</strong></td>
<td>J. ACM, Volume 7 (1960), pp. 326-329</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>$n(n-1)/2$ subtour elimination constraints</td>
</tr>
<tr>
<td><strong>Key</strong></td>
<td>subtour elimination constraints</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Author</strong></th>
<th>Kusiak, Andrew and Finke, Gerd</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
<td>Modeling and solving the flexible forging module scheduling problem</td>
</tr>
<tr>
<td><strong>Pub.</strong></td>
<td>Engineering Optimization, Volume 12 (1987), pp. 1-12</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Key</strong></td>
<td>2-commodity flow model of TSP</td>
</tr>
</tbody>
</table>
Wong, Richard T.

**Title**: Integer programming formulations of the traveling salesman problem


**Key**: multi-commodity flow model

---

Claus, A.

**Title**: A new formulation for the travelling salesman problem


**Key**: multi-commodity flow model