Chapter 1

Exponential Size Integer Programming Formulations

Although typically models need to be a reasonable size in order for us to code them and send them to a solver, there are some ways that we can allow having models of exponential size. The first example here is the cutting stock problem, where we will model with exponentially many variables. The second example is the traveling salesman problem, where we will model with exponentially many constraints. We will also look at some other models for the traveling salesman problem.

1.1 Cutting Stock

This is a classic problem that works excellent for a technique called *column generation*. We will discuss two versions of the model and then show how we can use column generation to solve the second version more efficiently. First, let's describe the problem.

Cutting Stock:

You run a company that sells of pipes of different lengths. These lengths are $L_1, \ldots, L_k$. To produce these pipes, you have one machine that produces pipes of length L, and then cuts them into a collection of shorter pipes as needed.

You have an order come in for d_i pipes of length i for $i = 1, \ldots, k$. How can you fill the order while cutting up the fewest number of pipes?

Example 1: A plumber stocks standard lengths of pipe, all of length 19 m. An order arrives for:

- 12 lengths of 4m
- 15 lengths of 5m
- 22 lengths of 6m

How should these lengths be cut from standard stock pipes so as to minimize the number of standard pipes used?

An initial model for this problem could be constructed as follows:

- Let N be an upper bound on the number of pipes that we may need.
- Let $z_j = 1$ if we use pipe i and $z_j = 0$ if we do not use pipe j, for $j = 1, \dots, N$.
- Let x_{ij} be the number of cuts of length L_i in pipe j that we use.

Then we have the following model

$$\begin{aligned} \min \quad & \sum_{j=1}^{N} z_j \\ \text{s.t.} \quad & \sum_{i=1}^{k} L_i x_{ij} \le L z_j \ \text{ for } j = 1, \dots, N \\ & \sum_{j=1}^{N} x_{ij} \ge d_i \ \text{ for } i = 1, \dots, k \\ & z_j \in \{0,1\} \text{ for } j = 1, \dots, N \\ & x_{ij} \in \mathbb{Z}_+ \text{ for } i = 1, \dots, k,\ j = 1, \dots, N \end{aligned} \tag{1.1.1}$$

Exercise 1. *In the example above, show that we can choose* $N = 16$.

For our example above, using $N = 16$, we have

$$\begin{aligned} \min \quad & \sum_{j=1}^{16} z_j \\ \text{s.t.} \quad & 4x_{1j} + 5x_{2j} + 6x_{3j} \le 19 z_j \\ & \sum_{j=1}^{16} x_{1j} \ge 12 \\ & \sum_{j=1}^{16} x_{2j} \ge 15 \\ & \sum_{j=1}^{16} x_{3j} \ge 22 \\ & z_j \in \{0,1\} \text{ for } j = 1, \dots, 16 \\ & x_{ij} \in \mathbb{Z}_+ \text{ for } i = 1, \dots, 3,\ j = 1, \dots, 16 \end{aligned} \tag{1.1.2}$$

Additionally, we could break the symmetry in the problem. That is, suppose the solution uses 10 of the 16 pipes. The current formulation does not restrict which 10 pipes are used. Thus, there are many possible solutions. To reduce this complexity, we can state that we only use the first 10 pipes. We can write a constraint that says *if we don't use pipe j, then we also will not use any subsequent pipes*. Hence, by not using pipe 11, we enforce that pipes $11, 12, 13, 14, 15, 16$ are not used. This can be done by adding the constraints

$$z_1 \geq z_2 \geq z_3 \geq \cdots \geq z_N. \tag{1.1.3}$$

See **??** for code for this formulation.

Unfortunately, this formulation is slow and does not scale well with demand. In particular, the number of variables is $N + kN$ and the number of constraints is N (plus integrality and non-negativity constraints on the variables). The solution times for this model are summarized in the following table:

INPUT TABLE OF COMPUTATIONS

1.1.1 Pattern formulation

We could instead list all patterns that are possible to cut each pipe. A pattern is an vector $a \in \mathbb{Z}^k_+$ such that for each i, a_i lengths of L_i can be cut from a pipe of length L. That is

$$\begin{aligned} &\sum_{i=1}^{k} L_i a_i \leq L \\ &a_i \in \mathbb{Z}_+ \text{ for all } i = 1, \dots, k \end{aligned} \tag{1.1.4}$$

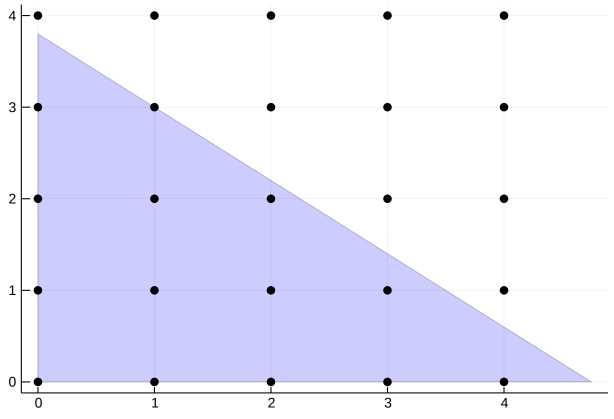
In our running example, we have

$$\begin{aligned} &4a_1 + 5a_2 + 6a_3 \leq 19 \\ &a_i \in \mathbb{Z}_+ \text{ for all } i = 1, \dots, 3 \end{aligned} \tag{1.1.5}$$

For visualization purposes, consider the patterns where $a_3 = 0$. That is, only patterns with cuts of length 4m or 5m. All patterns of this type are represented by an integer point in the polytope

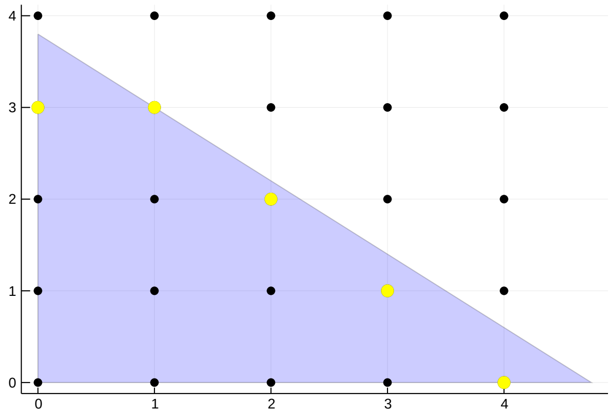
$$P = \{(a_1, a_2) : 4a_1 + 5a_2 \leq 19, a_1 \geq 0, a_2 \geq 0\} \tag{1.1.6}$$

which we can see here:



[0]This table was taken from notes by Laurent Lessard.

where P is the blue triangle and each integer point represents a pattern. Feasible patterns lie inside the polytope P. Note that we only need patterns that are maximal with respect to number of each type we cut. Pictorially, we only need the patterns that are integer points represented as yellow dots in the picture below.



For example, the pattern $[3,0,0]$ is not needed (only cut 3 of length 4m) since we could also use the patten $[4,0,0]$ (cut 4 of length 4m) or we could even use the pattern $[3,1,0]$ (cut 3 of length 4m and 1 of length 5m).

1.1.2 Column Generation

Consider the (**??**), but in this case we are instead minimizing. Thus we can write it as

$$\begin{aligned} \min \quad & (c_N - c_B B^{-1} N) x_N + c_B B^{-1} b \\ \text{s.t.} \quad & x_B + B^{-1} N) x_N = B^{-1} b \\ & x \geq 0 \end{aligned} \tag{1.1.7}$$

In our LP we have $c = \mathbb{1}$, that is, $c_i = 1$ for all $i = 1, \dots, k$. Hence, we can write it as

$$\begin{aligned} \min \quad & (\mathbb{1}_N - \mathbb{1}_B B^{-1} N) x_N + \mathbb{1}_B B^{-1} b \\ \text{s.t.} \quad & x_B + B^{-1} N) x_N = B^{-1} b \\ & x \geq 0 \end{aligned} \tag{1.1.8}$$

Now, if there exists a non-basic variable that could enter the basis and improve the objective, then there is one with a reduced cost that is negative. For a particular non-basic variable, the coefficient on it is

$$\left(1 - \mathbb{1}_B B^{-1} N^i\right) x_i \tag{1.1.9}$$

where N^i is the i-th column of the matrix N. Thus, we want to look for a column a of N such that

$$1 - \mathbb{1}_B B^{-1} a < 0 \quad \Rightarrow \quad 1 < \mathbb{1}_B B^{-1} a \tag{1.1.10}$$

Pricing Problem:
(knapsack problem!)
Given a current basis B of the *master* linear program, there exists a new column to add to the basis that improves the LP objective if and only if the following problem has an objective value strictly larger than 1

$$\begin{aligned} \max \quad & \mathbb{1}_B B^{-1} a \\ \text{s.t.} \quad & \sum_{i=1}^{k} L_i a_i \leq L \\ & a_i \in \mathbb{Z}_+ \text{ for } i = 1, \dots, k \end{aligned} \tag{1.1.11}$$

Example 2: Pricing Problem(knapsack problem!) After choosing the initial columns ....
we can find the objective function....

$$\begin{aligned} \max \quad & ....a \\ \text{s.t.} \quad & 4a_1 + 5a_2 + 6a_3 \leq 19 \\ & a_i \in \mathbb{Z}_+ \text{ for } i = 1, \dots, k \end{aligned} \tag{1.1.12}$$

1.1.3 Cutting Stock - Multiple widths

Gurobi has as excellent demonstration application to look at: `https://www.gurobi.com/cutting-stock-problem-with-multiple-master-rolls/`
`https://demos.gurobi.com/cutstock/`.
Here are some solutions:

- `https://github.com/fzsun/cutstock-gurobi`.
- `http://www.dcc.fc.up.pt/~jpp/mpa/cutstock.py`

Here is an AIMMS description of the problem:
`https://download.aimms.com/aimms/download/manuals/AIMMS3OM_CuttingStock.pdf`

1.2 Spanning Trees

See [?] for a list of 11 models for the minimum spanning tree and a comparison using CPLEX.

1.3 Traveling Salesman Problem

See http://www.math.uwaterloo.ca/tsp/index.html for excellent material on the TSP.
See also this chapter https://www.math.uwaterloo.ca/~bico/papers/comp_chapter1.pdf.

Also, watch this excellent talk by Bill Cook "Postcards from the Edge of Possibility": https://m.youtube.com/watch?v=5VjphFYQKj8

We consider a directed graph, graph $G = (N, A)$ of nodes N and arcs A. Arcs are directed edges. Hence the arc (i, j) is the directed path $i \to j$.

A *tour* is a cycle that visits all the nodes in N exactly once and returns back to the starting node.

Given costs c_{ij} for each arc $(i, j) \in A$, the goal is to find a minimum cost tour.

Traveling Salesman Problem:
NP-Hard
Given a directed graph $G = (N, A)$ and costs c_{ij} for all $(i, j) \in A$, find a tour of minimum cost.

ADD TSP FIGURE

In the figure, the nodes N are the cities and the arcs A are the directed paths city$i \to$ city$j$.

Models When constructing an integer programming model for TSP, we define variables x_{ij} for all $(i, j) \in A$ as

$$x_{ij} = 1 \text{ if the arc } (i, j) \text{ is used and } x_{ij} = 0 \text{ otherwise.}$$

We want the model to satisfy the fact that each node should have exactly one incoming arc and one leaving arc. Furthermore, we want to prevent self loops. Thus, we need the constraints:

$$\sum_{j \in N} x_{ij} = 1 \qquad \text{for all } i \in N \quad \text{[outgoing arc]} \tag{1.3.1}$$

$$\sum_{i \in N} x_{ij} = 1 \qquad \text{for all } j \in N \quad \text{[incoming arc]} \tag{1.3.2}$$

$$x_{ii} = 0 \qquad \text{for all } i \in N \quad \text{[no self loops]} \tag{1.3.3}$$

Unfortunately, these constraints are not enough to completely describe the problem. The issue is that *subtours* may arise. For instance

ADD SUBTOURS FIGURE

[0]This figure was taken from notes by Laurent Lessard.

MTZ Model

- This model adds variables $u_i \in \mathbb{Z}$ with $1 \leq u_i \leq n$ that decide the order in which nodes are visited.
- We set $u_1 = 1$ to set a starting place.
- Crucially, this model relies on the following fact

Let x be a solution to (1.3.1)-(1.3.3) with $x_{ij} \in \{0,1\}$. If there exists a subtour in this solution that contains the node 1, then there also exists a subtour that does not contain the node 1.

The following model adds constraints

$$\text{If } x_{ij} = 1, \quad \text{then} \quad u_i + 1 \leq u_j. \tag{1.3.4}$$

This if-then statement can be modeled with a big-M, choosing $M = n$ is a sufficient upper bound. Thus, it can be written as

$$u_i + 1 \leq u_j + n(1 - x_{ij}) \tag{1.3.5}$$

Setting these constraints to be active enforces the order $u_i < u_j$.

Consider a subtour now $2 \to 5 \to 3 \to 2$. Thus, $x_{25} = x_{53} = x_{32} = 1$. Then using the constraints from (1.3.5), we have that

$$u_2 < u_5 < u_3 < u_2, \tag{1.3.6}$$

but this is infeasible since we cannot have $u_2 < u_2$.

As stated above, if there is a subtour containing the node 1, then there is also a subtour not containing the node 1. Thus, we can enforce these constraints to only prevent subtours that don't contain the node 1. Thus, the full tour that contains the node 1 will still be feasible.

This is summarized in the following model:

Traveling Salesman Problem - MTZ Model:

$$
\begin{aligned}
\min \sum_{i,j \in N} c_{ij} x_{ij} & & & (1.3.7)\\
\sum_{j \in N} x_{ij} = 1 & \quad \text{for all } i \in N & \text{[outgoing arc]} & \quad (1.3.8)\\
\sum_{i \in N} x_{ij} = 1 & \quad \text{for all } j \in N & \text{[incoming arc]} & \quad (1.3.9)\\
x_{ii} = 0 & \quad \text{for all } i \in N & \text{[no self loops]} & \quad (1.3.10)\\
u_i + 1 \leq u_j + n(1 - x_{ij}) & \quad \text{for all } i,j \in N, i,j \neq 1 & \text{[prevents subtours]} & \quad (1.3.11)\\
u_1 = 1 & & & (1.3.12)\\
2 \leq u_i \leq n & \quad \text{for all } i \in N, i \neq 1 & & (1.3.13)\\
u_i \in \mathbb{Z} & \quad \text{for all } i \in N & & (1.3.14)\\
x_{ij} \in \{0,1\} & \quad \text{for all } i,j \in N & & (1.3.15)
\end{aligned}
$$

Example 3:
Distance Matrix:

```
A \ B | 1  2  3  4
----------------------
1     | 0  1  2  3
2     | 1  0  1  2
3     | 2  1  0  4
4     | 3  2  4  0
```

$$
\begin{aligned}
\min\quad & x_{1,2} + 2x_{1,3} + 3x_{1,4} + x_{2,1} + x_{2,3} + 2x_{2,4} + \\
& 2x_{3,1} + x_{3,2} + 4x_{3,4} + 3x_{4,1} + 2x_{4,2} + 4x_{4,3} \\
\text{Subject to} \\
& x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} = 1 && \text{outgoing from node 1} \\
& x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} = 1 && \text{outgoing from node 2} \\
& x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} = 1 && \text{outgoing from node 3} \\
& x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} = 1 && \text{outgoing from node 4} \\
\\
& x_{1,1} + x_{2,1} + x_{3,1} + x_{4,1} = 1 && \text{incoming to node 1} \\
& x_{1,2} + x_{2,2} + x_{3,2} + x_{4,2} = 1 && \text{incoming to node 2} \\
& x_{1,3} + x_{2,3} + x_{3,3} + x_{4,3} = 1 && \text{incoming to node 3} \\
& x_{1,4} + x_{2,4} + x_{3,4} + x_{4,4} = 1 && \text{incoming to node 4} \\
\\
& x_{1,1} = 0 && \text{No self loop with node 1} \\
& x_{2,2} = 0 && \text{No self loop with node 2} \\
& x_{3,3} = 0 && \text{No self loop with node 3} \\
& x_{4,4} = 0 && \text{No self loop with node 4} \\
\\
& u_1 = 1 && \text{Start at node 1} \\
& 2 \le u_i \le 4, \quad \forall i \in \{2,3,4\} \\
& u_2 + 1 \le u_3 + 4(1 - x_{2,3}) \\
& u_2 + 1 \le u_4 + 4(1 - x_{2,4}) \le 3 \\
& u_3 + 1 \le u_2 + 4(1 - x_{3,2}) \le 3 \\
& u_3 + 1 \le u_4 + 4(1 - x_{3,4}) \le 3 \\
& u_4 + 1 \le u_2 + 4(1 - x_{4,2}) \le 3 \\
& u_4 + 1 \le u_3 + 4(1 - x_{4,3}) \le 3 \\
& x_{i,j} \in \{0,1\} \quad \forall i \in \{1,2,3,4\}, j \in \{1,2,3,4\} \\
& u_i \in \mathbb{Z}, \quad \forall i \in \{1,2,3,4\}
\end{aligned}
$$

Example 4: 5 nodes

$$\begin{aligned}
\text{min} \quad & x_{1,2} + 2x_{1,3} + 3x_{1,4} + 4x_{1,5} + x_{2,1} + x_{2,3} + 2x_{2,4} + 2x_{2,5} + 2x_{3,1} + \\
& x_{3,2} + 4x_{3,4} + x_{3,5} + 3x_{4,1} + 2x_{4,2} + 4x_{4,3} + 2x_{4,5} + \\
& 4x_{5,1} + 2x_{5,2} + x_{5,3} + 2x_{5,4} \\
\text{Subject to} \quad & x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} + x_{1,5} = 1 \\
& x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} + x_{2,5} = 1 \\
& x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} + x_{3,5} = 1 \\
& x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} + x_{4,5} = 1 \\
& x_{5,1} + x_{5,2} + x_{5,3} + x_{5,4} + x_{5,5} = 1 \\
& \\
& x_{1,1} + x_{2,1} + x_{3,1} + x_{4,1} + x_{5,1} = 1 \\
& x_{1,2} + x_{2,2} + x_{3,2} + x_{4,2} + x_{5,2} = 1 \\
& x_{1,3} + x_{2,3} + x_{3,3} + x_{4,3} + x_{5,3} = 1 \\
& x_{1,4} + x_{2,4} + x_{3,4} + x_{4,4} + x_{5,4} = 1 \\
& x_{1,5} + x_{2,5} + x_{3,5} + x_{4,5} + x_{5,5} = 1 \\
& \\
& x_{1,1} = 0 \\
& x_{2,2} = 0 \\
& x_{3,3} = 0 \\
& x_{4,4} = 0 \\
& x_{5,5} = 0 \\
& \\
& u_1 = 1 \\
& 2 \le u_i \le 5 \quad \forall i \in \{1,2,3,4,5\} \\
& u_2 + 1 \le u_3 + 5(1 - x_{2,3}) \\
& u_2 + 1 \le u_4 + 5(1 - x_{2,4}) \\
& u_2 + 1 \le u_5 + 5(1 - x_{2,5}) \\
& u_3 + 1 \le u_2 + 5(1 - x_{3,2}) \\
& u_3 + 1 \le u_4 + 5(1 - x_{3,4}) \\
& u_4 + 1 \le u_2 + 5(1 - x_{4,2}) \\
& u_4 + 1 \le u_3 + 5(1 - x_{4,3}) \\
& u_3 + 1 \le u_5 + 5(1 - x_{3,5}) \\
& u_4 + 1 \le u_5 + 5(1 - x_{4,5}) \\
& u_5 + 1 \le u_2 + 5(1 - x_{5,2}) \\
& u_5 + 1 \le u_3 + 5(1 - x_{5,3}) \\
& u_5 + 1 \le u_4 + 5(1 - x_{5,4}) \\
& x_{i,j} \in \{0,1\} \quad \forall i \in \{1,2,3,4,5\}, j \in \{1,2,3,4,5\} \\
& u_i \in \mathbb{Z}, \quad \forall i \in \{1,2,3,4,5\}
\end{aligned}$$

Pros of this model

- Small description
- Easy to implement

Cons of this model

- Linear relaxation is not very tight. Thus, the solver may be slow when given this model.

1.3.1 Dantzig-Fulkerson-Johnson Model

This model does not add new variables. Instead, it adds constraints that conflict with the subtours. For instance, consider a subtour

$$2 \to 5 \to 3 \to 2. \tag{1.3.16}$$

We can prevent this subtour by adding the constraint

$$x_{25} + x_{53} + x_{32} \leq 2 \tag{1.3.17}$$

meaning that at most 2 of those arcs are allowed to happen at the same time. In general, for any subtour S, we can have the *subtour elimination constraint*

$$\sum_{(i,j)\in S} x_{ij} \leq |S| - 1 \qquad \text{Subtour Elimination Constraint.} \tag{1.3.18}$$

In the previous example with $S = \{(2,5),(5,3),(3,2)\}$ we have $|S| = 3$, where $|S|$ denotes the size of the set S.

This model suggests that we just add all of these subtour elimination constraints.

Traveling Salesman Problem - DFJ Model:

$$\min \sum_{i,j\in N} c_{ij} x_{ij} \tag{1.3.19}$$

$$\sum_{j\in N} x_{ij} = 1 \qquad \text{for all } i \in N \quad \text{[outgoing arc]} \tag{1.3.20}$$

$$\sum_{i\in N} x_{ij} = 1 \qquad \text{for all } j \in N \quad \text{[incoming arc]} \tag{1.3.21}$$

$$x_{ii} = 0 \qquad \text{for all } i \in N \quad \text{[no self loops]} \tag{1.3.22}$$

$$\sum_{(i,j)\in S} x_{ij} \leq |S| - 1 \qquad \text{for all subtours } S \quad \text{[prevents subtours]} \tag{1.3.23}$$

$$x_{ij} \in \{0,1\} \qquad \text{for all } i,j \in N \tag{1.3.24}$$

Example 5: DFJ Model for $n = 4$ nodes

Distance Matrix:

A \ B	1	2	3	4
1	0	1	2	3
2	1	0	1	2
3	2	1	0	4
4	3	2	4	0

$$\begin{aligned} 1\,\text{min} \quad & x_{1,2} + 2x_{1,3} + 3x_{1,4} + x_{2,1} + x_{2,3} + 2x_{2,4} \\ & + 2x_{3,1} + x_{3,2} + 4x_{3,4} + 3x_{4,1} + 2x_{4,2} + 4x_{4,3} \end{aligned}$$

Subject to

$$\begin{array}{lr}
x_{1,1} + x_{1,2} + x_{1,3} + x_{1,4} = 1 & \text{outgoing from node 1} \\
x_{2,1} + x_{2,2} + x_{2,3} + x_{2,4} = 1 & \text{outgoing from node 2} \\
x_{3,1} + x_{3,2} + x_{3,3} + x_{3,4} = 1 & \text{outgoing from node 3} \\
x_{4,1} + x_{4,2} + x_{4,3} + x_{4,4} = 1 & \text{outgoing from node 4}
\end{array}$$

$$\begin{array}{lr}
x_{1,1} + x_{2,1} + x_{3,1} + x_{4,1} = 1 & \text{incoming to node 1} \\
x_{1,2} + x_{2,2} + x_{3,2} + x_{4,2} = 1 & \text{incoming to node 2} \\
x_{1,3} + x_{2,3} + x_{3,3} + x_{4,3} = 1 & \text{incoming to node 3} \\
x_{1,4} + x_{2,4} + x_{3,4} + x_{4,4} = 1 & \text{incoming to node 4}
\end{array}$$

$$\begin{array}{lr}
x_{1,1} = 0 & \text{No self loop with node 1} \\
x_{2,2} = 0 & \text{No self loop with node 2} \\
x_{3,3} = 0 & \text{No self loop with node 3} \\
x_{4,4} = 0 & \text{No self loop with node 4}
\end{array}$$

$$\begin{array}{lr}
x_{1,2} + x_{2,1} \leq 1 & S = [(1,2),(2,1)] \\
x_{1,3} + x_{3,1} \leq 1 & S = [(1,3),(3,1)] \\
x_{1,4} + x_{4,1} \leq 1 & S = [(1,4),(4,1)] \\
x_{2,3} + x_{3,2} \leq 1 & S = [(2,3),(3,2)] \\
x_{2,4} + x_{4,2} \leq 1 & S = [(2,4),(4,2)] \\
x_{3,4} + x_{4,3} \leq 1 & S = [(3,4),(4,3)] \\
x_{2,1} + x_{1,3} + x_{3,2} \leq 2 & S = [(2,1),(1,3),(3,2)] \\
x_{1,2} + x_{2,3} + x_{3,1} \leq 2 & S = [(1,2),(2,3),(3,1)] \\
x_{3,1} + x_{1,4} + x_{4,3} \leq 2 & S = [(3,1),(1,4),(4,3))] \\
x_{1,3} + x_{3,4} + x_{4,1} \leq 2 & S = [(1,3),(3,4),(4,1)] \\
x_{2,1} + x_{1,4} + x_{4,2} \leq 2 & S = [(2,1),(1,4),(4,2)] \\
x_{1,2} + x_{2,4} + x_{4,1} \leq 2 & S = [(1,2),(2,4),(4,1)] \\
x_{3,2} + x_{2,4} + x_{4,3} \leq 2 & S = [(3,2),(2,4),(4,3)] \\
x_{2,3} + x_{3,4} + x_{4,2} \leq 2 & S = [(2,3),(3,4),(4,2)]
\end{array}$$

$$x_{i,j} \in \{0,1\} \quad \forall i \in \{1,2,3,4\}, j \in \{1,2,3,4\}$$

Example 6:
Consider a graph on 5 nodes.
Here are all the subtours of length at least 3 and also including the full length tours.
Hence, there are many subtours to consider.

Pros of this model

- Very tight linear relaxation

Cons of this model

- Exponentially many subtours S possible, hence this model is too large to write down.

Solution: Add subtour elimination constraints as needed. We will discuss this in a future section on *cutting planes* .

1.3.2 Traveling Salesman Problem - Branching Solution

We will see in the next section

1. That the constraint (1.3.1)-(1.3.3) always produce integer solutions as solutions to the linear relaxation.
2. A way to use branch and bound (the topic of the next section) in order to avoid subtours.

1.4 Google maps data

https://www.geeksforgeeks.org/python-calculate-distance-duration-two-places-using-google-distan

1.5 Literature

Gilmore-Gomory Cutting Stock [**?**]
http://www.optimization-online.org/DB_HTML/2018/06/6648.html
http://www.optimization-online.org/DB_HTML/2018/06/6670.html
http://www.optimization-online.org/DB_FILE/2017/11/6331.pdf