ILP basics

February 22, 2022

1 ILP basics

Combinatorial Optimization course, FEE CTU in Prague. Created by Industrial Informatics Department.

During the second lab of the course, we concentrate on the (I)LP modeling basics. The goal is to go through several simple examples and try to implement them using the ILP formalism. In this document, we show the basic syntax on a simple LP problem. Afterward, several NP-hard problems are formally presented, for which the students should fill the ILP formulations.

1.1 Introduction: Linear programming

In the linear programming, we try to solve the following problem:

$$\min_x \{c^{\mathrm{T}} x \mid Ax \leq b\}, \text{ where } c \in \mathbb{R}^n, x \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$$

i.e., we want to minimize a linear objective function, subject to linear inequality constraints defininy the feasible region (here, convex polytope).

1.1.1 Example

Take as an example the following problem:

$$\min_{x,y} -x + 2y \quad \text{s.t.} \tag{1}$$

$$-4x - 9y \leq -18 \tag{2}$$

$$\frac{3}{2}x - y \leq \frac{27}{4} \tag{3}$$

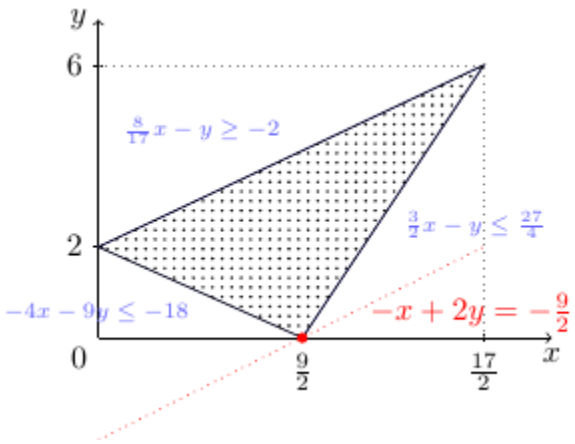
$$\frac{8}{17}x - y \geq -2 \tag{4}$$

Rewritten to the matrix form, the same problem would look as follows:

$$\min_{x,y} \begin{bmatrix} -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \text{ s.t.} \tag{5}$$

$$\begin{bmatrix} -4 & -9 \\ \frac{3}{2} & -1 \\ -\frac{8}{17} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} \leq \begin{bmatrix} -18 \\ \frac{27}{4} \\ 2 \end{bmatrix} \tag{6}$$

For this simple problem, we can visualize the individual constraints, the feasible region and even the optima, which is $x = \frac{9}{2}, y = 0$.



Now, let us try to rewrite this problem using the Gurobi syntax. Then, we can call the solver and see if the solution that we have found 'graphically' is, indeed, correct.

The model itself will consist of variables (including definition of their domains) and constraints. Here, the implementation is pretty straightforward:

[1]:
```
# Gurobi model for the example problem
import gurobipy as g  # import Gurobi

# Input data ----------------------
# The program above

# Model ---------------------------
model = g.Model()  # create an empty model
# - variables
x = model.addVar(vtype=g.GRB.CONTINUOUS, name="x")  # method addVar is used to␣
 ↪add variable
y = model.addVar(vtype=g.GRB.CONTINUOUS, name="y")
# - constraints
model.addConstr(-4*x - 9*y <= -18)  # method addConstr is used to add constraint
model.addConstr(3/2*x - y <= 27/4)
model.addConstr(8/17*x - y >= -2)
# - objective
model.setObjective(-x + 2*y, g.GRB.MINIMIZE)
```

```
model.optimize()

# Solution ------------------------
print("x:", x.X)  # use [variable].X to get the optimal value
print("y:", y.X)
```

```
Using license file /home/benedond/Apps/gurobi/gurobi910/gurobi.lic
Academic license - for non-commercial use only - expires 2021-04-16
Gurobi Optimizer version 9.1.0 build v9.1.0rc0 (linux64)
Thread count: 2 physical cores, 4 logical processors, using up to 4 threads
Optimize a model with 3 rows, 2 columns and 6 nonzeros
Model fingerprint: 0xb2d66890
Coefficient statistics:
  Matrix range     [5e-01, 9e+00]
  Objective range  [1e+00, 2e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [2e+00, 2e+01]
Presolve removed 3 rows and 2 columns
Presolve time: 0.01s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0   -4.5000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.01 seconds
Optimal objective -4.500000000e+00
x: 4.5
y: 0.0
```

While executing the model, Gurobi solver prints some information:

- `Thread count: 2 physical cores, 4 logical processors, using up to 4 threads`

Some parts of the solver can work in parallel - which might be beneficial when solving complex models.

- `Optimize a model with 3 rows, 2 columns and 6 nonzeros`

Here, we see that the matrix A contains 3 rows (constraints), 2 columns (variables) and 6 non-zero coefficients at total.

```
Presolve removed 3 rows and 2 columns
Presolve time: 0.01s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0   -4.5000000e+00   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.01 seconds
```

This parts just says that Gurobi was able to find the solution during the "Presolve" phase.

- `Optimal objective -4.500000000e+00`

The value of the objective function $c^{T} x $ is -4.5.

1.2 Integer linear programming

Now we solve similar problem, but the domains of (some) variables are restricted to be integers.

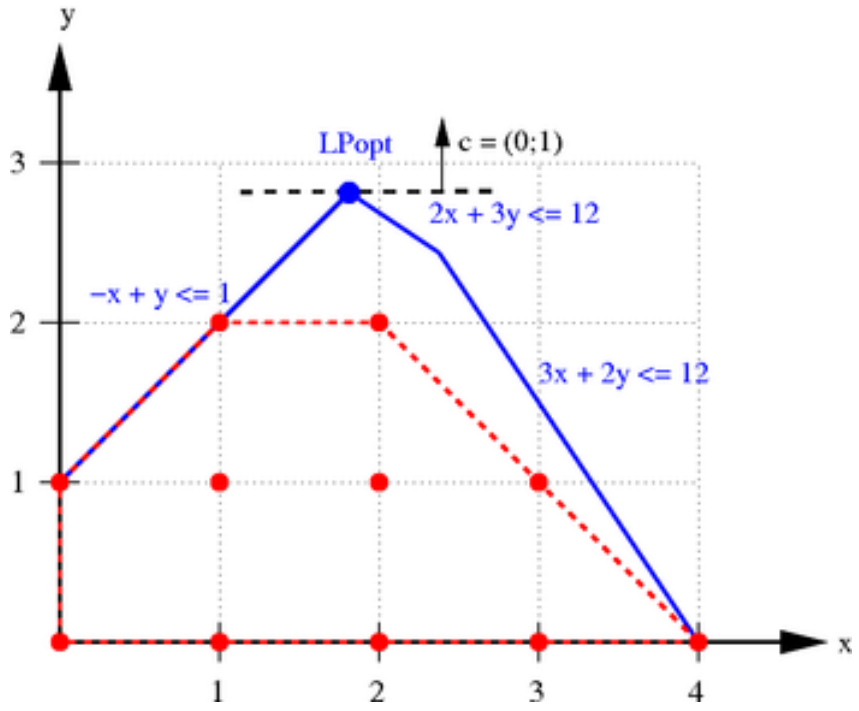
$$\min_{x}\{c^T x \mid Ax \leq b\}, c \in \mathbb{R}^n, x \in \mathbb{Z}^n, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$$

- Note that solving the problem as LP and rounding the solution **may not lead to optimal solution** (it might even lead to infeasible solution - see the previous example).

- Also note that optimizing over a subset will not give better solution (it may only worsen). Therefore, the optimal objective of LP provides a bound for the optimal objective of ILP (lower bound for the minimization).

Examine the following situation and compare the optimal objectives of LP and ILP variants of the same problem:

$$\max_{x,y} y \tag{7}$$
$$\text{subject to} \tag{8}$$
$$-x + y \leq 1 \tag{9}$$
$$2x + 3y \leq 12 \tag{10}$$
$$3x + 2y \leq 12 \tag{11}$$
$$x \geq 0 \tag{12}$$
$$y \geq 0 \tag{13}$$



The area inside of the blue polygon represents the feasible region of the LP formulation, while the red dots represent the feasible solutions of the ILP formulation.

Now, lets enjoy some fun while implementing the models for some well-known NP-hard problems. The goal is to understand the problems and formalize them in Gurobi language.

1.2.1 Problem 1: Two partition

Given multiset S of positive integers, we ask if it can be partitioned to two subsets S_1, S_2, such that sum of the numbers in S_1 equals to the sum of the numbers in S_2, and each number is assigned (either to S_1 or to S_2).

Example 1 Multiset $S = \{1, 1, 1, 2, 2, 3\}$ can be partitioned, e.g., to $S_1 = \{2, 3\}$, $S_2 = \{1, 1, 1, 2\}$. Note that it is not a unique solution.

Example 2 Multiset $S = \{2, 4\}$ cannot be partitioned.

[2]:
```
import gurobipy as g

# Example input
S = [100, 50, 50, 50, 20, 20, 10]

# TODO: implement the model and find the solution
```

1.2.2 Problem 2: Maximum independent set

Given graph $G = (V, E)$, we are trying to find subset $V^\star \subseteq V$, such that $V^\star$ is as large as possible and for each pair of vertices $u, v \in V^\star$, $u \neq v$, it holds that there is no edge $e = \{u, v\}$ in the

original graph.

[3]:
```
import gurobipy as g

# Example input
n = 5  # number of vertices
edges = [(0, 1), (2, 3), (0, 4), (3, 1), (3, 4)]  # list of edges

# TODO: implement the model and find the solution
```

1.2.3 Problem 3: Minimum vertex cover

Given graph $G = (V, E)$, we are looking for subset $V^\star \subseteq V$, such that $V^\star$ is as small as possible and each edge $e = \{u, v\}$ is 'covered' by at least one vertex $x \in V^\star$, i.e. $\forall e = \{u, v\} \in E \; \exists x \in V^\star$ such that $x = u$ or $x = v$.

[4]:
```
import gurobipy as g

# Example input
n = 5  # number of vertices
edges = [(0, 1), (2, 3), (0, 4), (3, 1), (3, 4)]  # list of edges

# TODO: implement the model and find the solution
```

1.2.4 Problem 4: SAT (boolean satisfiability problem)

Given formula φ (in CNF), we are looking for truth assignment (True / False) to each variable, such that φ is satisfied (evaluates to True).

Example 1 Formula $\varphi = (x_1 \vee x_2) \wedge (\neg x_2 \vee x_3)$ can be satisfied by $x_1 :=$ True, $x_3 :=$ True, $x_2 :=$ False.

Example 2 Formula $\varphi = (x_1) \wedge (\neg x_1)$ cannot be satisfied.

[ ]:
```
import gurobipy as g

# Example input
# SAT formula: (x0 or ~x1) and (~x0 or x1 or x2) <- encode directly into the␣
 ↪model

# TODO: implement the model and find the solution
```