1 Introduction to integer programming

Many real-world decision-making problems require decisions over a **discrete** set of available choices. **Integer programming** provides a general and comprehensive framework to model such decision-making settings. With the success of modern integer programming software such as Gurobi, CPLEX, and Xpress, integer programming is arguably the most prevalent optimization framework in operations research and business analytics.

Let us consider a toy example. First, the following is a simple linear program (LP) with two variables.

$$\begin{aligned} \max \quad & 4x + 5y \\ \text{s.t.} \quad & x + 3y \leq 10, \\ & 3x + y \leq 10, \\ & x, y \geq 0. \end{aligned}$$

We can draw the feasible region of this LP as in (1.1). We know that linear programming is "easy"

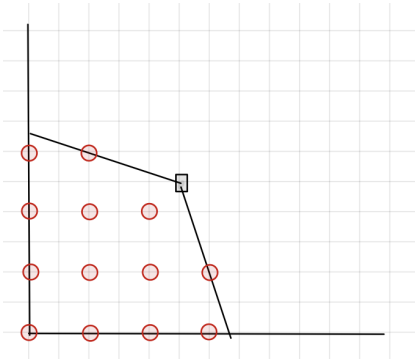

Figure 1.1: Depicting the feasible region of the linear and integer programs

in that there is a polynomial time algorithm for solving linear programs, such as the ellipsoid method and the interior point method. By solving the linear program, we know that the optimal solution is $(x, y) = (5/2, 5/2)$ which is at the intersection of two lines $x + 3y = 10$ and $3x + y = 10$. What if we want a solution whose componets are all **integers**?

$$\begin{aligned} \max \quad & 4x + 5y \\ \text{s.t.} \quad & x + 3y \leq 10, \\ & 3x + y \leq 10, \\ & x, y \geq 0, \\ & (x, y) \in \mathbb{Z}^2. \end{aligned}$$

Basically, we add an additional condition that $(x, y) \in \mathbb{Z}^2$, where $\mathbb{Z}$ denotes the set of integers, to the above linear program. In Figure 1.1, the red dots depict the set of solutions that satisfy the linear inequalities and the **integrality condition**. In fact, the second optimization problem is called an **integer program**. As the objective function and the other constraints are given by linear functions, the optimization problem is also referred to as an **integer linear program**.

What is an optimal solution to the integer program? The first attempt is to look at the optimal solution to the linear program, which is $(5/2, 5/2)$, and round the components to the nearest integers. However, there are a couple of issues with the rounding procedure.

1. There can be many integer solutions obtained from rounding the optimal solution to the linear program. We may round $5/2$ to 2 or 3. Hence, $(2, 2)$, $(2, 3)$, $(3, 2)$, and $(3, 3)$ may be obtained from rounding $(5/2, 5/2)$. When the number of variables is d, the number of solutions from rounding is up to 2^d.

2. We cannot guarantee the feasibility of solutions obtained from rounding. $(2, 2)$ is feasible, but $(2, 3)$, $(3, 2)$, and $(3, 3)$ are all infeasible.

3. More importantly, it is not always the case that there is an optimal solution from the list of solutions obtained from rounding. In fact, the optimal solution to the our integer program is given by $(1, 3)$.

Therefore, linear programming combined with rounding does not necessary solve integer programming. It turns out that integer programming is NP**-hard**, which implies that there would be no polynomial time algorithm unless $P = NP$. Integer programming includes difficult problems in computer science such as Satisfiability and the Traveling Salesman Problem (TSP). Furthermore, linear programming is a class of convex optimization, while the set of solutions to a integer program is discrete and thus non-convex. These computational challenges require methodologies that deal with the discrete nature of integer programming, which has motivated an extensive research on integer programming both in theory and practice.

2 Brief history of integer programming

In fact, integer programming has a long histroy shortly after George B. Dantzig developed the simplex method for **linear programming**. Since then, integer programming models for **combinatorial optimization** were extensively studied both in theory and practice. For example, Dantzig, Fulkerson, and Johnson [6, 7] in the 1950's developed an integer programming formulation for the Traveling Salesman Problem (TSP), which was the precursor of the now-called **branch-and-cut algorithm**. In 1965, Jack Edmonds [8] developed the blossom algorithm for the matching problem based on integer programming. There were massive activities in the 1970's and 1980's on developing integer programming formulations for discrete optimization problems that arise in business operations.

In the 1990's and 2000's, the research community advanced the technology of **general purpose cutting-planes** that can be implemented in branch-and-bound algorithms. Here, "general" means that the cutting planes work for arbitrary integer programs not just for some specific formulations. This development led to the modern success of integer programming software. In fact, the history of general purpose cuts dates back to 1958 when Ralph Gomory developed **Gomory's fractional cuts** and the first convergent **cutting plane algorithm** for pure integer programs [9]. Later in 1963, he developed **Gomory's mixed integer cuts** for mixed integer programs [10]. However,

until the mid 1990's, it was a common belief that Gomory's cuts are useless in practice and that one has to exploit the underlying combinatorial structure of a given problem [5]. In the summer of 1993, Sebastian Ceria and Gérard Cornuéjols obtained the first successful implementation of Gomory's mixed integer cuts and demonstrated that Gomory's cuts are indeed powerful [2, 5]. Later on, various general purpose cutting planes have been proposed and studied, such as split cuts, disjunctive cuts, and mixed-integer rounding cuts.

In modern days, integer programming is being used in machine learning and data-driven decision making problems as well. Let us mention a few that are important in the literature.

- Sparse regression [3].
- Neural network verification [1].
- Learning bayesian networks [13].
- Sample average approximation for chance-constrained programming [15, 14].
- Distributionally robust chance-constrained programming [4, 16, 11, 12].

3 Some integer programming formulations

Before we formally state and define what it is, let us discuss a few exciting integer programming models from combinatorial optimization, machine learning, and data-driven decision making.

3.1 Bipartite matching

A **bipartite graph** is a graph $G = (V, E)$ where

- the vertex set V is partitioned into two sets V_1 and V_2,
- each edge $e \in E$ crosses the partition, i.e. e has one end in V_1 and the other end in V_2.

For example, Figure 1.2 shows a bipartite graph on 7 vertices where one set contains 3 and the other has 4. A **matching** is a set of edges without common vertices. In Figure 1.2, the set of green

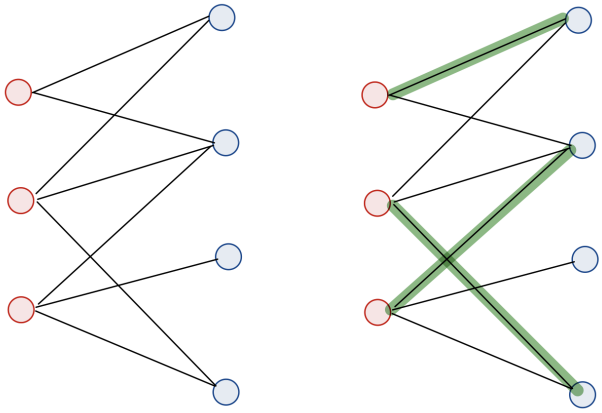

Figure 1.2: Bipartite graph and a matching

edges gives rise to a matching.

Suppose that each edge $e \in E$ has a weight w_e. Given a set of edges F, the weight of F is defined as the sum of weights of the edges in F, given by, $\sum_{e \in F} w_e$. The **matching problem** is to find a matching that has the maximum weight.

The matching problem has numerous applications such as auctions and scheduling. The matching problem can be formulated as the following integer program.

$$\begin{aligned} \min \quad & \sum_{e \in E} w_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(v)} x_e \leq 1, \quad \forall v \in V, \\ & x_e \in \{0,1\}, \quad \forall e \in E \end{aligned}$$

where for $v \in V$,

$$\delta(v) = \{e \in E : \text{ one end of } e \text{ is } v\}.$$

In fact, the integer program is a valid formulation for the matching problem even if G is not bipartite. When G is bipartite, we will see that we may replace the integrality constraint $x_e \in \{0,1\}$ for $e \in E$ by

$$0 \leq x_e \leq 1 \quad \text{or} \quad x_e \in [0,1], \quad \forall e \in E.$$

In other words, the following linear program

$$\begin{aligned} \min \quad & \sum_{e \in E} w_e x_e \\ \text{s.t.} \quad & \sum_{e \in \delta(v)} x_e \leq 1, \quad \forall v \in V, \\ & 0 \leq x_e \leq 1, \quad \forall e \in E \end{aligned}$$

solves the bipartite matching problem. The linear program is called the **LP relaxation** of the integer program.

One might wonder when it is the case that the LP relaxation computes an integer optimal solution. This is not always true, and we will study some sufficient conditions under which the LP relaxation is equivalent to the given integer program.

3.2 Sparse regression

Let $(x^1, y^1), \ldots, (x^n, y^n)$ be n data points where x is the vector of features and y is the response variable. We want to infer a linear model

$$y = \beta^\top x$$

from the data set. The mean squared error (MSE) is given by

$$\frac{1}{n} \sum_{i=1}^{n} \left(y^i - \beta^\top x^i \right)^2.$$

To find a valid β, one would minimize

$$\text{MSE} + \text{Regularization}.$$

Here, **sparse regression** corresponds to applying the ℓ_0 regularization term where

$$\|\beta\|_0 = |\{\beta_i : \beta_i \neq 0\}|,$$

that is, the number of nonzero components in β. We solve

$$\min_{\beta} \quad \frac{1}{n}\sum_{i=1}^{n}\left(y^i - \beta^\top x^i\right)^2 + \lambda\|\beta\|_0$$

where λ is some penalty parameter. The ℓ_0 regularization term is non-convex and in fact combinatorial. The following is an equivalent reformulation of the problem. For some sufficiently large $M > 0$, we consider

$$\begin{aligned} \min \quad & \frac{1}{n}\sum_{i=1}^{n}\left(y^i - \beta^\top x^i\right)^2 + \lambda\sum_{j\in[d]} z_j \\ \text{s.t.} \quad & -Mz_j \leq \beta_j \leq Mz_j, \quad \forall j \in [d], \\ & z_j \in \{0,1\}, \qquad\qquad \forall j \in [d]. \end{aligned}$$

Here, the coefficient M is often referred to as the **big-M**. The computational tractability of this integer programming formulation heavily depends on the value of M. When M is large, the constraint $-Mz_j \leq \beta_j \leq Mz_j$ is too loose. On the other hand, we cannot set M arbitrarily small, for otherwise, the formulation will be invalid. There have been various techniques developed for improving the formulation.

References

[1] Anderson, R., Huchette, J., Ma, W., Tjandraatmadja, C., Vielma J.P.: Strong mixed-integer programming formulations for trained neural networks. Mathematical Programming **183**, 3–39 (2020) 2

[2] Balas, E., Ceria, S., Cornuéjols, G., Natraj, N.: Gomory cuts revisited. Operations Research Letters **19**, 1–9 (1996) 2

[3] Bertsimas, D., King, A., Mazumder, R.: Best subset selection via a modern optimization lens. The Annals of Statistics **44**, 813–852 (2016) 2

[4] Chen, Z., Kuhn, D., Wiesemann, W.: Data-driven chance constrained programs over Wasserstein balls. Operations Research, Articles in advance. 2

[5] Cornuéjols, G.: Revival of the Gomory cuts in the 1990's. Annals of Operations Research **149**, 63–66 (2007) 2

[6] Dantzig, G.B., Fulkerson, D.R., Johnson, S.: Solution of a large scale traveling salesman problem. Operations Research **2**, 393–410 (1954) 2

[7] Dantzig, G.B., Fulkerson, D.R., Johnson, S.: On a linear programming combinatorial approach to the traveling salesman problem. Operations Research **2**, 58–66 (1959) 2

[8] Edmonds, J.: Paths, trees, and flowers. Canadian Journal of Mathematics **17**, 449–467 (1965). 2

[9] Gomory, R.E.: Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society **64**, 275–278 (1958) 2

[10] Gomory, R.E.: An algorithm for the mixed integer problem. Technical Report RM-2597, The Rand Corporation (1960) 2

[11] Ho-Nguyen, N., Kilinc-Karzan, F., Kücükyavuz, S., Lee, D.: Distributionally robust chance-constrained programs with right-hand side uncertainty under Wasserstein ambiguity. Mathematical Programming **196** 641–672 (2022). 2

[12] Ho-Nguyen, N., Kilinc-Karzan, F., Kücükyavuz, S., Lee, D.: Strong formulations for distributionally robust chance-constrained programs with left-hand side uncertainty under Wasserstein ambiguity. INFORMS Journal on Optimization, Articles in advance 2

[13] Chen, R., Dash, S., Gao, T.: Integer programming for causal structure learning in the presence of latent variables. ICML 2021. 2

[14] Luedtke, J.: A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Mathematical Programming **146** 219–244 (2014) 2

[15] Luedtke, J., Ahmed, S., Nemhauser, G.L.: An integer programming approach for linear programs with probabilistic constraints. Mathematical Programming **122** 247–272 (2010) 2

[16] Xie, W.: On distributionally robust chance constrained programs with Wasserstein distance. Mathematical Programming **186** 115–155 (2021) 2