Chapter 1

Formulations

Contents

Discrete optimization problems arise in a great variety of contexts in science, engineering and management. In such problems, we seek to find a solution $\mathbf{x}^*$ in a discrete set $\mathcal{F}$ that optimizes (minimizes or maximizes) an objective function $c(\mathbf{x})$ defined for all $\mathbf{x} \in \mathcal{F}$. A natural and systematic way to study discrete optimization problems is to express them as integer optimization problems, which is exactly the emphasis of this book.

In general, given matrices $\mathbf{A} \in \mathbb{Z}^{m\times n}$, $\mathbf{B} \in \mathbb{Z}^{m\times k}$, and vectors $\mathbf{b} \in \mathbb{Z}^m$, $\mathbf{c} \in \mathbb{Z}^n$, $\mathbf{d} \in \mathbb{Z}^k$, the problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{d}'\mathbf{y} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{b}, \\ & \mathbf{x} \in \mathbb{Z}^n_+, \ \mathbf{y} \in \mathbb{R}^k_+, \end{aligned}$$

is the **mixed integer optimization** problem. Notice that even if there are inequality constraints, we can still write the problem in the above form by adding slack or surplus variables. If there are no continuous variables $\mathbf{y}$, the problem is called the **integer optimization** problem. If furthermore, there are no continuous variables and the components of the vector $\mathbf{x}$ are restricted to be either zero or one, the problem is called the **binary** (or **zero-one**) **integer optimization** problem. Finally, it is customary to assume that the entries of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$ are integers.

Integer optimization is a rather powerful modeling framework that provides great flexibility for expressing discrete optimization problems. On the other hand, the price for this flexibility is that integer optimization is computationally more demanding than linear optimization. In this chapter, we introduce general guidelines for obtaining strong integer optimization formulations for discrete optimization problems. We introduce modeling techniques, discuss what constitutes a strong formulation, and compare alternative formulations of the same problem.

1.1 Modeling techniques

In this section, we outline some modeling techniques that facilitate the formulation of discrete optimization problems as integer and mixed integer optimization problems. In comparison to linear optimization, integer optimization is significantly richer in modeling power. Unfortunately, there is no systematic way to formulate discrete optimization problems, and devising a good model is often an art, which we plan to explore through examples.

Binary choice

An important use of a binary variable x is to encode a choice between two alternatives: we may set x to zero or one, depending on the chosen alternative.

Example 1.1 (The knapsack problem) We are given n items. The jth item has weight w_j and its value is c_j. Given a bound K on the weight that can be carried in a knapsack, we would like to select items to maximize the total value. In order to model this problem, we define a binary variable x_j which is one, if item j is chosen, and zero, otherwise. The problem can then be formulated as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} w_j x_j \le K, \\ & x_j \in \{0,1\}, \qquad \forall\, j. \end{aligned}$$

Forcing constraints

A very common feature in discrete optimization problems is that certain decisions are dependent. In particular, suppose decision A can be made only if decision B has also been made. In order to model such a situation, we can introduce binary variables x (respectively, y) equal to one, if decision A (respectively, B) is chosen, and zero, otherwise. The dependence of the two decisions can be modeled using the constraint

$$x \le y,$$

i.e., if $y = 0$ (decision B is not made), then $x = 0$ (decision A cannot be made). Next, we present an example where forcing constraints are used.

Example 1.2 (Facility location problems) Suppose we are given n potential facility locations and a list of m clients who need to be serviced from these locations. There is a fixed cost c_j of opening a facility at location j, while there is a cost d_{ij} of serving client i from facility j. The goal is to select a set of facility locations and assign each client to one facility, while minimizing the total cost.

We define a binary decision variable y_j for each location j, which is equal to one, if facility j is opened, and zero, otherwise. In addition, we define a variable x_{ij}, which corresponds to the fraction of the demand of client i that is served by facility j. The facility location problem is then formulated as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} c_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} x_{ij} & & \\ \text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1, & \forall\, i, & \qquad (1.1) \\ & x_{ij} \le y_j, & \forall\, i,j, & \\ & 0 \le x_{ij} \le 1,\ y_j \in \{0,1\}, & \forall\, i,j. & \end{aligned}$$

Here, the forcing constraint $x_{ij} \le y_j$ captures the fact that if there is no facility at location j ($y_j = 0$), then client i cannot be served there, and we must have $x_{ij} = 0$.

Relations between variables

A constraint of the form $\sum_{j=1}^{n} x_j \leq 1$, where all variables are binary, implies that at most one of the variables x_j can be one. Similarly, if the constraint is of the form $\sum_{j=1}^{n} x_j = 1$, then exactly one of the variables x_j should be one.

Disjunctive constraints

Let $\mathbf{x}$ be a nonnegative vector representing a collection of decision variables. Suppose that we are given two constraints $\mathbf{a}'\mathbf{x} \geq b$ and $\mathbf{c}'\mathbf{x} \geq d$, in which all of the components of $\mathbf{a}$ and $\mathbf{c}$ are nonnegative. We would like to model a requirement that at least one of the two constraints is satisfied. In order to achieve this, we define a binary variable y and impose the constraints:

$$\mathbf{a}'\mathbf{x} \geq yb, \quad \mathbf{c}'\mathbf{x} \geq (1-y)d, \quad y \in \{0,1\}.$$

More generally, suppose we are given m constraints $\mathbf{a}_i'\mathbf{x} \geq b_i$, $i = 1,\ldots,m$, where $\mathbf{a}_i \geq \mathbf{0}$ for each i, and require that at least k of them are satisfied. We can achieve this by introducing m binary variables y_i, $i = 1,\ldots,m$, and the constraints:

$$\sum_{i=1}^{m} y_i \geq k, \quad \mathbf{a}_i'\mathbf{x} \geq b_i y_i, \quad y_i \in \{0,1\}, \quad i = 1,\ldots,m.$$

Restricted range of values

Suppose we want to restrict a variable x to take values in a set $\{a_1,\ldots,a_m\}$. We can achieve this by introducing m binary variables y_j, $j = 1,\ldots,m$, and the constraints

$$x = \sum_{j=1}^{m} a_j y_j, \quad \sum_{j=1}^{m} y_j = 1, \quad y_j \in \{0,1\}.$$

Arbitrary piecewise linear cost functions

Suppose we are interested to minimize an arbitrary piecewise linear, not necessarily convex, cost function. We will show that we can accomplish this using binary variables. Suppose that $a_1 < a_2 < \cdots < a_k$, and that we have a continuous piecewise linear function $f(x)$ specified by the points $(a_i, f(a_i))$ for $i = 1,\ldots,k$, defined on the interval $[a_1, a_k]$ (see Figure 1.1). Then, any $x \in [a_1, a_k]$ and the corresponding value $f(x)$ can be expressed in the form

$$x = \sum_{i=1}^{k} \lambda_i a_i, \qquad f(x) = \sum_{i=1}^{k} \lambda_i f(a_i), \qquad \sum_{i=1}^{k} \lambda_i = 1, \qquad \lambda_1,\ldots,\lambda_k \geq 0.$$

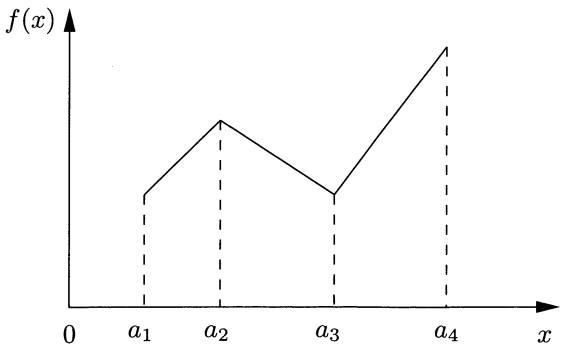


Figure 1.1: A continuous piecewise linear cost function.

The critical observation is that the choice of coefficients $\lambda_1,\ldots,\lambda_k$ used to represent a particular x is not unique. However, it becomes unique if we require that at most two consecutive coefficients λ_i can be nonzero. In this case, any $x \in [a_i, a_{i+1}]$, is represented uniquely as $x = \lambda_i a_i + \lambda_{i+1} a_{i+1}$, with $\lambda_i + \lambda_{i+1} = 1$. To this effect, we define the binary variable y_i, $i = 1,\ldots,k-1$, which is equal to one, if $a_i \leq x < a_{i+1}$, and zero, otherwise. The problem is then formulated as follows:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{k} \lambda_i f(a_i) \\
\text{s.t.} \quad & \lambda_1 \leq y_1, \quad \lambda_k \leq y_{k-1}, \\
& \lambda_i \leq y_{i-1} + y_i, \qquad i = 2,\ldots,k-1, \\
& \sum_{i=1}^{k} \lambda_i = 1, \quad \sum_{i=1}^{k-1} y_i = 1, \\
& \lambda_i \geq 0, \quad y_i \in \{0,1\}.
\end{aligned}$$

Notice that if $y_j = 1$, then $\lambda_i = 0$ for i different than j or $j+1$.

The previous collection of examples is by no means an exhaustive list of possible modeling devices. They only serve to illustrate the power of modeling with binary variables. In order to acquire more confidence, we introduce some more examples.

Example 1.3 (The set covering, set packing, and set partitioning problems) Let $M = \{1,\ldots,m\}$ and $N = \{1,\ldots,n\}$. Let $M_1, M_2, \ldots, M_n$ be a given collection of subsets of M. For example, the collection might consist of all subsets of size at least k. We are also given a weight c_j for each set M_j in the collection. We say that a subset F of N is a **cover** of M if $\cup_{j\in F} M_j = M$. We say that F is a **packing** of M if $M_j \cap M_k = \emptyset$, for all $j,k \in F$, $j \neq k$. We say that F is a **partition** of M if it is both a cover and a packing of M (see Figure 1.2). The **weight** of a subset F of N is defined as $\sum_{j\in F} c_j$.

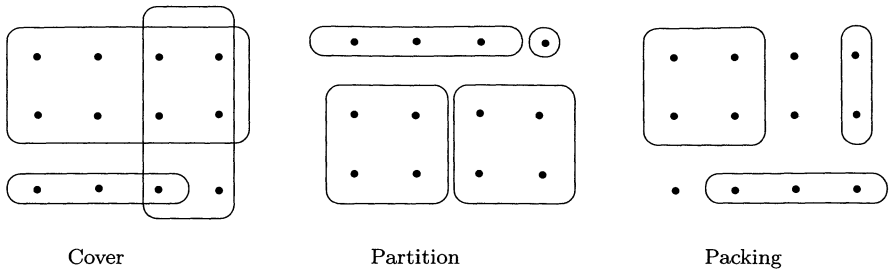


Figure 1.2: (a) A cover, (b) a partition, and (c) a packing.

In the **set covering** problem we would like to find a cover F of minimum weight, in the **set packing** problem we would like to find a packing F of maximum weight, while in the **set partitioning** problem both minimization and maximization versions are possible. In order to formulate these problems as integer optimization problems, we introduce the $m \times n$ incidence matrix $\mathbf{A}$ of the family $\{M_j \mid j \in N\}$, whose entries are given by

$$a_{ij} = \begin{cases} 1, & \text{if } i \in M_j, \\ 0, & \text{otherwise.} \end{cases}$$

We also define a decision variable x_j, $j = 1, \ldots, n$, which is equal to one, if $j \in F$, and zero, otherwise. Let $\mathbf{x} = (x_1, \ldots, x_n)$. Then, F is a cover, packing, partition if and only if

$$\mathbf{Ax} \geq \mathbf{e}, \qquad \mathbf{Ax} \leq \mathbf{e}, \qquad \mathbf{Ax} = \mathbf{e},$$

respectively, where $\mathbf{e}$ is an m-dimensional vector with all components equal to 1. More generally, given a nonnegative vector $\mathbf{b}$, and a nonnegative matrix $\mathbf{A}$, packing, covering and partitioning type of formulations are of the form:

$$\mathbf{Ax} \geq \mathbf{b}, \qquad \mathbf{Ax} \leq \mathbf{b}, \qquad \mathbf{Ax} = \mathbf{b},$$

$$\mathbf{x} \in \{0,1\}^n.$$

Note that by defining variables $\mathbf{y} = \mathbf{e} - \mathbf{x}$, we can transform a covering formulation to a packing formulation and vice versa. For example, if $\mathbf{x} \in \{0,1\}^n$ satisfies $\mathbf{Ax} \geq \mathbf{b}$, then $\mathbf{Ay} \leq \mathbf{Ae} - \mathbf{b}$, $\mathbf{y} \in \{0,1\}^n$.

The previous formulation types encompass a variety of important problems such as crew scheduling problems, vehicle routing problems, etc. (see Section 1.4).

A sequencing problem with setup times

A flexible machine can perform m operations, indexed from 1 to m. Each operation j requires a unique tool j. The machine can simultaneously hold B tools in its tool magazine, where $B < m$. Loading or unloading tool j into the machine magazine requires s_j units of setup time. Only one tool at a time can be loaded or unloaded. At the start of the day,

n jobs are waiting to be processed by the machine. Each job i requires multiple operations. Let J_i denote the set of operations required by job i, and assume for simplicity that for all i, $|J_i|$ is no larger than the magazine capacity B of the machine. Before the machine can start processing job i, all the required tools belonging to the set J_i must be setup on the machine. If a tool $j \in J_i$, is already loaded on the machine, we avoid the setup time for tool j. If tool $j \in J_i$ is not already loaded, we must set it up, possibly (if the tool magazine is currently full) after unloading an existing tool that job i does not require. Once the tools are setup, all $|J_i|$ operations of job i are processed. Notice that, because of commonality in tool requirements for different jobs and the limited magazine capacity, the setup time required prior to each job is sequence dependent. We want to formulate an integer optimization problem to determine the optimal job sequence that minimizes the total setup time to complete all the jobs. We assume that at the start of the day, the tool magazine is completely empty. We define decision variables that capture the job sequence:

$$x_{ir} = \begin{cases} 1, & \text{if job } i \text{ is the } r\text{th job processed,} \\ 0, & \text{otherwise.} \end{cases}$$

In addition, we define decision variables that describe the magazine setups:

$$y_{jr} = \begin{cases} 1, & \text{if tool } j \text{ is on the magazine, while the } r\text{th job is processed,} \\ 0, & \text{otherwise.} \end{cases}$$

We also let $y_{j0} = 0$, for all j, which represents the fact that we are starting with an empty magazine. Since every job needs to be processed,

$$\sum_{r=1}^{n} x_{ir} = 1, \qquad \forall\, i.$$

Since exactly one job will be processed at a time,

$$\sum_{i=1}^{n} x_{ir} = 1, \qquad \forall\, r.$$

In order to process job i, all tools in the set J_i need to be in the magazine. Therefore, we have

$$x_{ir} \leq y_{jr}, \qquad \forall\, j \in J_i,\ \forall\, r, i.$$

Since the total capacity of the magazine is B, we have

$$\sum_{j=1}^{m} y_{jr} \leq B, \qquad \forall\, r.$$

we incur a setup delay only if we load or unload a tool, which is the case if $y_{jr} \neq y_{j,r-1}$ for some j. Therefore, the objective function is

$$\text{minimize} \quad \sum_{j=1}^{m} \sum_{r=1}^{n} s_j |y_{jr} - y_{j,r-1}|.$$

We obtain a binary optimization problem by defining a new decision variable z_{jr}, writing the objective function as

$$\text{minimize} \quad \sum_{j=1}^{m} \sum_{r=1}^{n} s_j z_{jr},$$

and by introducing the constraints

$$\begin{aligned} z_{jr} &\geq y_{jr} - y_{j,r-1}, && \forall\, j, r, \\ z_{jr} &\geq y_{j,r-1} - y_{jr}, && \forall\, j, r. \end{aligned}$$

The overall formulation becomes

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{m} \sum_{r=1}^{n} s_j z_{jr} \\ \text{subject to} \quad & z_{jr} \geq y_{jr} - y_{j,r-1}, && \forall\, j, r, \\ & z_{jr} \geq y_{j,r-1} - y_{jr}, && \forall\, j, r, \\ & \sum_{r=1}^{n} x_{ir} = 1, && \forall\, i, \\ & \sum_{i=1}^{n} x_{ir} = 1, && \forall\, r, \\ & x_{ir} \leq y_{jr}, && \forall\, j \in J_i,\ \forall\, r, i, \\ & \sum_{j=1}^{m} y_{jr} \leq B, && \forall\, r, \\ & x_{ir}, y_{jr}, z_{jr} \in \{0,1\}. \end{aligned}$$

1.2 Guidelines for strong formulations

In linear optimization, a good formulation is one that has a small number n, m of variables and constraints, respectively, because the computational complexity of the problem grows polynomially in n and m. In addition, given the availability of several efficient algorithms for linear optimization, the choice of a formulation, although important, does not critically affect our ability to solve the problem. The situation in integer optimization is drastically different. Extensive computational experience suggests that the choice of a formulation is crucial. In this section, we provide guidelines for arriving at strong formulations. The key concept we introduce is that of a linear relaxation.

Definition 1.1 *Given a mixed integer optimization problem*

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{d}'\mathbf{y} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{b}, \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0}, \\ & \mathbf{x} \in \mathbb{Z}^n, \end{aligned}$$

its **linear relaxation** *is defined as*

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{d}'\mathbf{y} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} = \mathbf{b}, \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0}, \end{aligned}$$

where the requirement that $\mathbf{x}$ *is a vector of integers is relaxed. If the integer variables* x_j *are further restricted to be either* $0, 1, \ldots, U$, *then in the linear relaxation,* x_j *takes values between* 0 *and* U.

Note that if an optimal solution to the relaxation is feasible for the mixed integer optimization problem, it is also an optimal solution to the latter.

Example 1.4 (The facility location problem revisited) In Example 1.2, we presented an integer optimization formulation of the facility location problem. Let us consider the following alternative formulation [**aggregate facility location** formulation (AFL)] :

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} c_j y_j + \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1, && \forall\, i, \\ & \sum_{i=1}^{m} x_{ij} \leq m y_j, && \forall\, j, \\ & 0 \leq x_{ij} \leq 1,\ y_j \in \{0,1\}. \end{aligned} \tag{1.2}$$

Notice that the constraint $\sum_{i=1}^{m} x_{ij} \leq m y_j$ forces x_{ij} to be zero whenever $y_j = 0$, but allows x_{ij} to be different from zero if $y_j = 1$. Therefore, this constraint is equivalent to the constraints $x_{ij} \leq y_j$, $i = 1, \ldots, m$, in the original formulation. For this reason, the set of feasible solutions and the optimal cost is the same for both formulations. Notice, however, that the aggregate formulation (1.2) has $m + n$ constraints, while the original formulation (1.1) had $m + mn$ constraints.

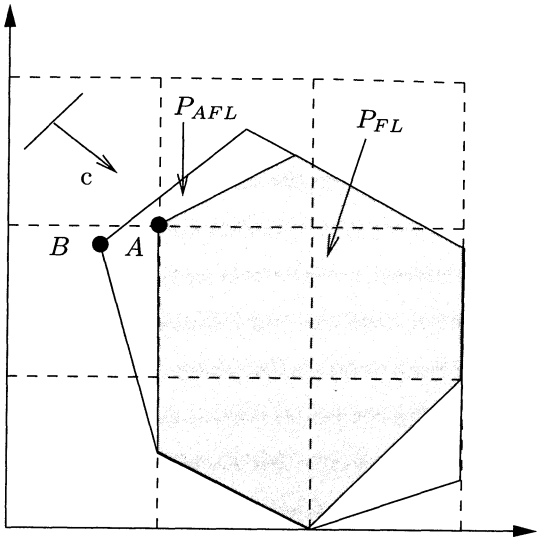


Figure 1.3: The two polyhedra P_{FL} and P_{AFL} contain exactly the same set of integer solutions.

In order to compare the two formulations, let us consider their corresponding linear relaxations, in which we replace the integrality restrictions $y_j \in \{0,1\}$ by $0 \leq y_j \leq 1$. We then define the following two polyhedra, which are the feasible sets of the two relaxations:

$$P_{\text{FL}} = \left\{ (\mathbf{x},\mathbf{y})' \;\middle|\; \begin{array}{l} \displaystyle\sum_{j=1}^{n} x_{ij} = 1, \quad \forall\, i, \\ x_{ij} \leq y_j, \quad \forall\, i,j, \\ 0 \leq x_{ij} \leq 1,\ 0 \leq y_j \leq 1 \end{array} \right\},$$

$$P_{\text{AFL}} = \left\{ (\mathbf{x},\mathbf{y})' \;\middle|\; \begin{array}{l} \displaystyle\sum_{j=1}^{n} x_{ij} = 1, \quad \forall\, i, \\ \displaystyle\sum_{i=1}^{m} x_{ij} \leq m y_j, \quad \forall\, j, \\ 0 \leq x_{ij} \leq 1,\ 0 \leq y_j \leq 1 \end{array} \right\}.$$

Clearly, $P_{\text{FL}} \subset P_{\text{AFL}}$ and the inclusion can be strict (Exercise 1.16). In other words, the feasible set of the linear relaxation of formulation (1.1) is closer to the set of integer solutions than the linear relaxation of formulation (1.2) (see Figure 1.3).

Let Z_{IP} be the optimal cost of the integer optimization problem, and let Z_{FL} and Z_{AFL} be the optimal costs of the two linear relaxations we have introduced. Since $P_{\text{FL}} \subset P_{\text{AFL}}$, it follows that $Z_{\text{AFL}} \leq Z_{\text{FL}}$. Moreover, $Z_{\text{FL}} \leq Z_{\text{IP}}$, since an optimal solution to the integer optimization problem belongs to P_{FL}. To summarize,

$$Z_{\text{AFL}} \leq Z_{\text{FL}} \leq Z_{\text{IP}}.$$

We next discuss the implications of this ordering. Enumerative methods such as branch and bound methods discussed in Section 11.1 for solving integer minimization problems depend on the availability of lower bounds such as Z_{FL}. The sharper the bound, i.e., the closer it is to Z_{IP}, the better these methods behave.

Consider, for example, an objective function such as the one depicted in Figure 1.3. For this objective function, the optimal solution over the polyhedron P_{FL} corresponds to point A in the figure, and is integer. By the previous inequalities, solution A is indeed the optimal solution to the facility location problem. So, in this case, we have solved the integer optimization problem by just solving a linear optimization problem. On the other hand, for the same objective function, the optimal solution over the polyhedron P_{AFL} is point B, which is fractional.

Thus, formulation (1.1) is preferable to formulation (1.2), despite the fact that (1.2) has a significantly smaller number of constraints.

What is then an ideal formulation of an integer optimization problem? Let $\mathcal{F} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ be the set of feasible integer solutions to a particular integer optimization problem. We assume that the feasible set is bounded and, therefore, $\mathcal{F}$ is finite. We consider the convex hull of $\mathcal{F}$:

$$\text{conv}(\mathcal{F}) = \left\{ \sum_{i=1}^{k} \lambda_i \mathbf{x}_i \;\middle|\; \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geq 0 \right\}.$$

The set $\text{conv}(\mathcal{F})$ is a polyhedron that has integer extreme points. Furthermore, the feasible set P of any linear relaxation satisfies $\text{conv}(\mathcal{F}) \subset P$. If we knew $\text{conv}(\mathcal{F})$ explicitly, i.e., if we could represent $\text{conv}(\mathcal{F})$ in the form $\text{conv}(\mathcal{F}) = \{\mathbf{x} \mid \mathbf{D}\mathbf{x} \leq \mathbf{d}\}$, we could solve the integer optimization problem

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{x} \in \mathcal{F}, \end{array}$$

by finding an extreme point solution to the linear optimization problem

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{x} \in \text{conv}(\mathcal{F}). \end{array}$$

Given our ability to solve linear optimization problems efficiently, it is then desirable to have a formulation, whose linear relaxation is indeed the convex hull $\text{conv}(\mathcal{F})$ of the integer feasible solutions (see Figure 1.4). Unfortunately, this is often difficult. In light of this, we strive for a compromise whereby we attempt to come up with a polyhedron that closely approximates $\text{conv}(\mathcal{F})$. This leads to the central message of this chapter.

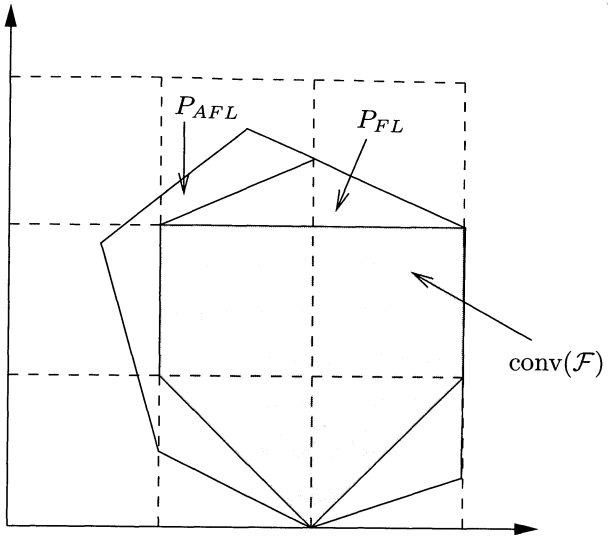


Figure 1.4: The convex hull of the integer feasible solutions to the facility location problem.

The **quality of a formulation** of an integer optimization problem with feasible solution set $\mathcal{F}$, can be judged by the closeness of the feasible set of its linear relaxation to the convex hull of $\mathcal{F}$. In particular, consider two formulations A and B of the same integer optimization problem. If we denote by P_A and P_B the feasible sets of the corresponding linear relaxations, we consider formulation A to be at least as **strong** as formulation B if

$$P_A \subseteq P_B.$$

Let us illustrate this message in an example.

Example 1.5 (The pigeonhole principle)
The pigeonhole principle is a central proof method in combinatorics and states that we cannot place $n+1$ pigeons into n holes in such a way that no two pigeons share the same hole. Let us first write a formulation of this problem. Let x_{ij} be one, if pigeon i occupies hole j, $i = 1, \ldots, n+1$ and $j = 1, \ldots, n$, and, zero, otherwise. We consider the following two formulations of the problem. The first is

$$\begin{aligned} &\sum_{j=1}^{n} x_{ij} = 1, && i = 1, \ldots, n+1, \\ &x_{ij} + x_{kj} \leq 1, && j = 1, \ldots, n,\ i \neq k,\ i, k = 1, \ldots, n+1, \\ &x_{ij} \in \{0, 1\}, && i = 1, \ldots, n+1,\ j = 1, \ldots, n, \end{aligned} \tag{1.3}$$

and the second is given by

$$\begin{aligned} &\sum_{j=1}^{n} x_{ij} = 1, && i = 1, \ldots, n+1, \\ &\sum_{i=1}^{n+1} x_{ij} \leq 1, && j = 1, \ldots, n, \\ &x_{ij} \in \{0, 1\}, && i = 1, \ldots, n+1,\ j = 1, \ldots, n. \end{aligned} \tag{1.4}$$

Clearly the pigeonhole problem is infeasible. Note, however, that the linear relaxation of problem (1.3) is feasible as the solution $x_{ij} = 1/n$ satisfies all the constraints, while the linear relaxation of problem (1.4) is infeasible. It turns out that enumerative approaches based on solving the linear relaxation of problem (1.3) require an almost complete enumeration of all the integer solutions, while problem (1.4) detects infeasibility immediately by solving the linear relaxation.

1.3 Modeling with exponentially many constraints

In this section, we demonstrate through examples that strong formulations and, in particular, the convex hull of integer feasible solutions, may involve an exponential number of linear inequality constraints. However, this does not necessarily prevent the efficient solution of the linear relaxation of such problems, as they can often be solved by cutting plane methods (see Chapter 5).

The minimum spanning tree problem

Let $G = (V, E)$ be an undirected graph with node set V ($|V| = n$) and edge set E ($|E| = m$). Every edge $e \in E$ has an associated cost c_e. The cost of a tree is simply the sum of the costs of the edges in the tree. The minimum spanning tree problem asks for a spanning tree (an acyclic, connected subgraph of G) of minimum cost. Tree optimization problems arise in the design of transportation, communication, and computer networks, since at the very least such networks should be connected. Our goal in this example is to illustrate the effectiveness of alternative formulations and to learn new principles for deriving strong formulations.

In order to formulate the problem, we define for each $e \in E$, a variable x_e which is equal to one, if edge e is included in the tree, and zero, otherwise. Since a spanning tree should have $n-1$ edges, we introduce the constraint

$$\sum_{e \in E} x_e = n - 1.$$

Moreover, the chosen edges should not contain a cycle. It can be shown (Exercise 1.17) that this is guaranteed if for any nonempty set $S \subset V$, the

number of edges with both endpoints in S is less than or equal to $|S| - 1$. For any $S \subset V$, we define

$$E(S) = \{\{i,j\} \in E \mid i,j \in S\},$$

and we can express this set of constraints as

$$\sum_{e \in E(S)} x_e \leq |S| - 1, \qquad S \subset V,\ S \neq \emptyset, V.$$

This leads to a formulation of the minimum spanning tree problem:

$$\begin{aligned}
\text{minimize} \quad & \sum_{e \in E} c_e x_e \\
\text{subject to} \quad & \sum_{e \in E} x_e = n - 1, \\
& \sum_{e \in E(S)} x_e \leq |S| - 1, \qquad S \subset V,\ S \neq \emptyset, V, \\
& x_e \in \{0,1\}.
\end{aligned}$$

This formulation is called the **subtour elimination formulation**, since it contains constraints that eliminate all subtours (cycles). We denote the feasible set of the linear relaxation of this formulation by P_{sub}, where we replace the constraint $x_e \in \{0,1\}$ with $0 \leq x_e \leq 1$. Notice that the subtour elimination formulation has an exponential number of constraints, namely $2^n - 1$.

The subtour elimination formulation uses the definition of a tree as a subgraph containing $n - 1$ edges and no cycles. Using an alternative, but equivalent definition, a tree is a connected graph containing $n - 1$ edges. Given a subset S of V, we define the **cutset** $\delta(S)$ (see also Figure 1.5) by

$$\delta(S) = \{\{i,j\} \in E \mid i \in S,\ j \notin S\}.$$

Note that $\delta(\{i\})$ is the set of edges incident to i. We can then express the connectivity requirement in terms of the constraints

$$\sum_{e \in \delta(S)} x_e \geq 1, \qquad S \subset V,\ S \neq \emptyset, V.$$

We call the resulting formulation the **cutset formulation**, and we denote the feasible set of its linear relaxation by P_{cut}. Both formulations have an exponential number of constraints. Are these formulations equally strong? We show that the subtour elimination formulation is stronger than the cut-set formulation. The proof demonstrates how we can compare alternative formulations of discrete optimization problems.

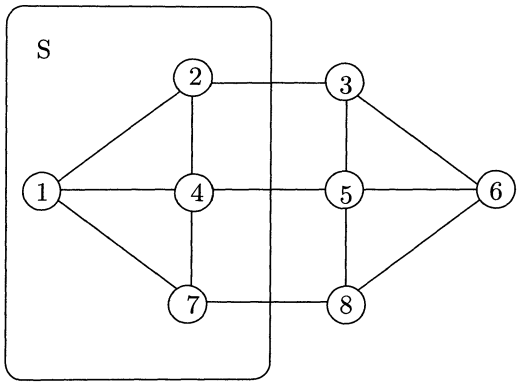


Figure 1.5: Let $S = \{1,2,4,7\}$. Then, $\delta(S) = \big\{\{2,3\},\ \{4,5\},\ \{7,8\}\big\}$, and $E(S) = \big\{\{1,2\},\ \{1,4\},\ \{2,4\},\ \{4,7\},\ \{1,7\}\big\}$.

Theorem 1.1 *The following properties hold.*

(a) *We have $P_{\text{sub}} \subset P_{\text{cut}}$, and there exist examples for which the inclusion is strict.*

(b) *The polyhedron P_{cut} can have fractional extreme points.*

Proof.

(a) For any set S of nodes, we have

$$E = E(S) \cup \delta(S) \cup E(V \setminus S).$$

Therefore,

$$\sum_{e \in E(S)} x_e + \sum_{e \in E(V \setminus S)} x_e + \sum_{e \in \delta(S)} x_e = \sum_{e \in E} x_e.$$

For $\mathbf{x} \in P_{\text{sub}}$, and for $S \neq \emptyset$, V, we have

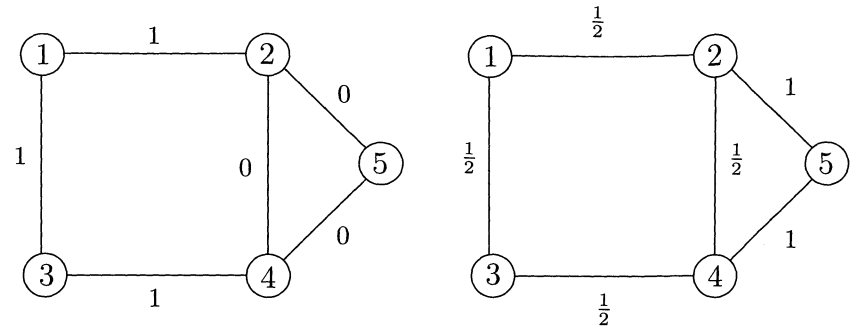
$$\sum_{e \in E(S)} x_e \leq |S| - 1,$$

and

$$\sum_{e \in E(V \setminus S)} x_e \leq |V \setminus S| - 1.$$

Since

$$\sum_{e \in E} x_e = n - 1,$$



(a) (b)

Figure 1.6: An example of a graph, in which a minimum spanning tree has cost 2, while the cost of an optimal solution over P_{cut} is 3/2. (a) The cost coefficients. (b) An optimal solution $\mathbf{x}^*$ over P_{cut}.

we obtain that

$$\sum_{e \in \delta(S)} x_e \geq 1,$$

and therefore $\mathbf{x} \in P_{\text{cut}}$.

Consider the example in Figure 1.6(a). The solution $\mathbf{x}^*$ shown in Figure 1.6(b) belongs to P_{cut}, but it does not belong to P_{sub}, since the edges in $E(S)$ for $S = \{2, 4, 5\}$ have total weight 5/2, while the constraints defining P_{sub} dictate that $\sum_{e \in E(S)} x_e \leq 2$. The example shows that the inclusion may be strict.

(b) In order to show that the polyhedron P_{cut} may have fractional extreme points, we use the objective function shown in Figure 1.6(a), under which the unique optimal solution over P_{cut} is $\mathbf{x}^*$ shown in Figure 1.6(b). This establishes that this unique fractional solution with a cost of 3/2 is an extreme point of P_{cut}. □

In Theorem 1.1, we have shown that the cutset formulation is weaker than the subtour elimination formulation. In addition, in Section 3.2 we will show that $P_{\text{sub}} = \text{conv}(\mathcal{F})$, i.e., the polyhedron P_{sub} is a representation of the convex hull of the set $\mathcal{F}$ of vectors corresponding to spanning trees.

According to the principle regarding strong formulations, the subtour elimination formulation is a strong one. This seems somewhat counterintuitive, as the formulation involves an exponential number of constraints. Does this prevent us from optimizing over the feasible set P_{sub} of the linear relaxation efficiently? In Section 5.4, we will see that we can optimize over P_{sub} efficiently both theoretically and practically.

The traveling salesman problem

Given an undirected graph $G = (V, E)$ and costs c_e for every edge $e \in E$, the objective is to find a **tour** (a cycle that visits all nodes) of minimum cost. In order to model the problem, we define for every edge $e \in E$ a variable x_e equal to one, if edge e is included in the tour, and zero, otherwise. Since each node must participate in two edges of the tour, we have

$$\sum_{e \in \delta(\{i\})} x_e = 2, \qquad i \in V.$$

Also, if S is a nonempty proper subset of V, there must be at least two edges joining S to $V \setminus S$, and we have

$$\sum_{e \in \delta(S)} x_e \geq 2, \qquad S \subset V,\ S \neq \emptyset, V.$$

A cutset formulation of the traveling salesman problem is as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(\{i\})} x_e = 2, && i \in V, \\ & \sum_{e \in \delta(S)} x_e \geq 2, && S \subset V,\ S \neq \emptyset, V, \\ & x_e \in \{0, 1\}. \end{aligned} \tag{1.5}$$

Using ideas similar to the subtour elimination formulation of the minimum spanning tree problem, we can also formulate the traveling salesman problem in terms of the following constraints:

$$\begin{aligned} & \sum_{e \in \delta(\{i\})} x_e = 2, && i \in V, \\ & \sum_{e \in E(S)} x_e \leq |S| - 1, && S \subset V,\ S \neq \emptyset, V, \\ & x_e \in \{0, 1\}. \end{aligned}$$

Let P_{tspcut} and P_{tspsub} be the polyhedra corresponding to the linear relaxations of these two formulations. It turns out that the two formulations are equally strong, i.e., $P_{\text{tspcut}} = P_{\text{tspsub}}$ (Exercise 1.20).

Exercise 1.21 deals with a different formulation of the variant of the traveling salesman problem that involves a directed graph. This formulation has a polynomial number of constraints, but it is not as strong as the natural extension of the cutset formulation to directed graphs, which has an exponential number of constraints.

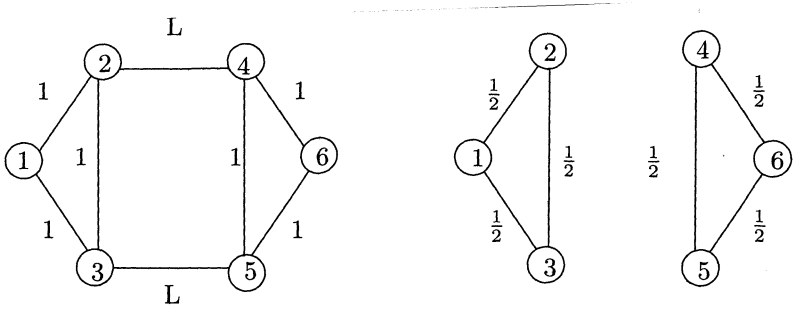


Figure 1.7: An example of a graph, in which an optimal matching has cost $L+2$, while the cost of the optimal solution over P_{degree} has cost 3. (a) The cost coefficients. (b) An optimal solution over P_{degree}.

The perfect matching problem

We have an even number n of persons that need to be matched into pairs in order to perform a certain job. If person i is matched with person j, there is a cost of c_{ij}. A perfect matching is a pairing of persons, so that each individual is matched with exactly one other individual. The goal is to find a perfect matching that minimizes the total cost. We represent the set of people by an undirected graph $G = (V, E)$ where V is the set of individuals, and the cost of edge $e = \{i, j\}$ is c_e. If $\{i, j\} \notin E$, this indicates that i and j cannot be matched. We let x_e be one, if edge $e = \{i, j\}$ is selected, i.e., persons i and j are matched, and zero, otherwise. Then, the **perfect matching** problem can be formulated as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(\{i\})} x_e = 1, \qquad i \in V, \\ & x_e \in \{0, 1\}. \end{aligned}$$

We denote by P_{degree} the polyhedron corresponding to the linear relaxation of the above formulation.

Let $\mathcal{F}$ be the set of all vectors $\mathbf{x}$ corresponding to matchings. Figure 1.7 shows that P_{degree} is not equal to $\text{conv}(\mathcal{F})$.

The example in Figure 1.7 shows that the above formulation of the problem is not particularly strong, as its linear relaxation is not equal to the convex hull of vectors corresponding to matchings. A strengthening of the formulation is to consider the class of inequalities

$$\sum_{e \in \delta(S)} x_e \geq 1, \qquad S \subset V,\ S \neq V,\ |S| \text{ odd}.$$

Notice that all vectors corresponding to matchings satisfy this inequality, as in every set with an odd number of nodes, there should be at least one edge leaving this set of nodes. Note that the example in Figure 1.7(b) violates this set of inequalities, since for the set $S = \{1, 2, 3\}$ we have $\sum_{e \in \delta(S)} x_e = 0$. We then consider the polyhedron defined by the constraints we have introduced:

$$\begin{aligned} P_{\text{matching}} = \Big\{ \mathbf{x} \;\Big|\; & \sum_{e \in \delta(\{i\})} x_e = 1, \quad i \in V, \\ & \sum_{e \in \delta(S)} x_e \geq 1, \quad S \subset V,\ S \neq V,\ |S| \text{ odd}, \\ & 0 \leq x_e \leq 1,\ e \in E \Big\}. \end{aligned}$$

It turns out (see Section 3.4) that $P_{\text{matching}} = \text{conv}(\mathcal{F})$.

Cut covering problems

Our next example encompasses a large collection of integer optimization problems including all problems we considered in this section. Its purpose is to show the power and flexibility of modeling with an exponential number of constraints. For further results see Section 12.2. Let $G = (V, E)$, $|V| = n$ be an undirected graph. Let $f : 2^V \to \mathbb{Z}_+$ be a given set function and $D \subseteq V$. Given costs $c_e \geq 0$ for every $e \in E$, we consider the following family of discrete optimization problems known as cut covering problems:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(\{i\})} x_e = f(\{i\}), \quad i \in D \subseteq V, \\ & \sum_{e \in \delta(S)} x_e \geq f(S), \qquad S \subseteq V, \\ & x_e \in \{0, 1\}. \end{aligned} \tag{1.6}$$

Note that because of $c_e \geq 0$, there exists always an optimal solution $\mathbf{x}$ to problem (1.6) such that

$$F = \{e \in E \mid x_e = 1\}$$

is minimal with respect to inclusion, i.e., $\mathbf{x} - \mathbf{e}_k$ is infeasible, for all $k \in F$ ($\mathbf{e}_k$ is a vector of zeros except that it has an one in the kth coordinate).

cut covering problems include (see also Table 12.1):

(a) The minimum spanning tree by taking $f(S) = 1$, for all $S \neq \emptyset, V$ and $D = \emptyset$.

(b) The traveling salesman problem by taking $f(S) = 2$, for all $S \neq \emptyset, V$ and $D = V$.

(c) The perfect matching problem by taking $f(S) = 1$, for every $S \neq \emptyset, V$ with $|S|$ odd and $D = V$.

(d) **The Steiner tree problem** is defined as the problem in which a set $T \subset V$ of nodes needs to be connected by a tree possibly using nodes in $V \setminus T$. Assuming that $c_e \geq 0$, for all $e \in E$, the choice $f(S)$ equal one, if $S \cap T \neq \emptyset, T$, and $f(S)$ equal zero, otherwise, and $D = \emptyset$ models the Steiner tree problem.

(e) **The survivable network design problem** is an important problem arising in the design of communication, utility, and transportation networks. Given costs c_e, for all $e \in E$ and requirements r_{ij} for every pair of nodes $i, j \in V$, the objective is to select a set of edges from E at minimum cost, so that between every pair of nodes i and j there are at least r_{ij} paths that do not share any edges. By requiring that there are at least r_{ij} edge disjoint paths, the network has enough connectivity, so that even if some edges in the network become unavailable, nodes i and j could still be connected. The problem belongs to the class of cut covering problems (Exercise 1.12) by letting $D = \emptyset$ and

$$f(S) = \max_{i \in S,\ j \in V \setminus S} r_{ij}, \qquad S \neq \emptyset, V. \tag{1.7}$$

(f) The vehicle routing problem can be formulated as a cut covering problem (see Exercise 1.13).

We next show that we can sometimes strengthen the formulation of a discrete optimization problem defined on an undirected graph by formulating the problem in a corresponding directed graph.

Directed versus undirected formulations

The Steiner tree problem on an undirected graph $G = (V, E)$ with nonnegative edge weights $c_e \geq 0$, for all $e \in E$ and with a set T of terminal nodes can be formulated as a cut covering problem (1.6) with $f(S) = 1$, if $S \cap T \neq \emptyset, T$ and $D = \emptyset$. Specifically, the formulation is

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(S)} x_e \geq 1, \quad \forall\, S \subset V, \quad S \cap T \neq \emptyset, T, \\ & x_e \in \{0, 1\}. \end{aligned} \tag{1.8}$$

We can enhance this formulation by considering a collection of subsets $V_1, \ldots, V_p$ of V satisfying:

(a) $V_i \cap T \neq \emptyset$, $i = 1, \ldots, p$.

(b) $V_i \cap V_j = \emptyset$, $i, j = 1, \ldots, p$, $i \neq j$.

(c) $\cup_{i=1}^{p} V_i = V$.

Let $\delta(V_1, \ldots, V_p)$ be the set of edges, whose endpoints lie in different sets V_i. We consider the following formulation of the Steiner tree problem.

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e x_e \\ \text{subject to} \quad & \sum_{e \in \delta(V_1, \ldots, V_p)} x_e \geq p - 1, \qquad (V_1, \ldots, V_p) \\ & \qquad\qquad\qquad\qquad\qquad \text{satisfying (a)-(c)}, \\ & x_e \in \{0, 1\}. \end{aligned} \tag{1.9}$$

Given the undirected graph $G = (V, E)$, we create a directed graph (V, A) by directing each edge $\{i, j\} \in E$, thus creating two arcs (i, j), $(j, i) \in A$ with cost $c_{ij} = c_{ji} \geq 0$. In the directed Steiner tree problem, the goal is to find a minimum cost directed subtree that contains a directed path between some given root vertex 1 ($1 \in T$), and every other terminal in T. Of course, a directed subtree solution can be transformed back to an undirected solution. The directed formulation of the Steiner tree problem is as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j) \in A} c_{ij} y_{ij} \\ \text{subject to} \quad & \sum_{(i,j) \in \delta^+(S)} y_{ij} \geq 1, \quad S \subset V,\ 1 \in S,\ T \setminus S \neq \emptyset, \\ & y_{ij} + y_{ji} \leq 1, \quad e = \{i, j\} \in E, \\ & y_{ij} \in \{0, 1\}, \end{aligned} \tag{1.10}$$

where $\delta^+(S)$ represents the set of arcs (i, j) with $i \in S$, and $j \notin S$.

Note that if $\mathbf{y}$ is feasible for problem (1.10), then by setting

$$x_e = y_{ij} + y_{ji}, \text{ for all } e = \{i, j\} \in E,$$

we obtain a feasible solution for problems (1.8) and (1.9), respectively. Although all three formulations (1.8), (1.9) and (1.10) are equivalent as integer optimization problems, this is not the case for their linear relaxations. Let $Z_{\text{steiner}}(T)$, $Z_{\text{partition}}(T)$, and $ZD_{\text{steiner}}(T)$ be the value of the linear relaxations of formulations (1.8), (1.9) and (1.10), respectively.

We next show that

$$Z_{\text{steiner}}(T) \leq Z_{\text{partition}}(T) \leq ZD_{\text{steiner}}(T),$$

that is, the directed formulation is at least as strong than both the undirected formulation and the formulation based on Steiner partitions.

Since formulation (1.8) is a special case of formulation (1.9) corresponding to $p = 2$, $Z_{\text{steiner}}(T) \leq Z_{\text{partition}}(T)$. Without loss of generality we assume that the root vertex $1 \in V_1$, and consider the inequalities

$$\sum_{(i,j)\in\delta^+(V\setminus V_k)} y_{ij} \geq 1, \qquad k = 2,\ldots,p.$$

Adding these inequalities together with the inequalities $y_{ji} \geq 0$ for $j \in V_k$, $k = 2,\ldots,p$ and $i \in V_1$, $(i,j) \in E$, we obtain

$$\sum_{e\in\delta(V_1,\ldots,V_p)} (y_{ij} + y_{ji}) \geq p - 1.$$

Setting $x_e = y_{ij} + y_{ji}$ where $e = \{i,j\}$, we obtain that the solution $\mathbf{x}$ is feasible for the linear relaxation of formulation (1.9), proving that $Z_{\text{partition}}(T) \leq ZD_{\text{steiner}}(T)$.

There are examples such that $Z_{\text{steiner}}(T) < ZD_{\text{steiner}}(T)$, that is, the directed formulation is strictly stronger in terms of the bound it produces. It does, however, have twice as many variables.

It is not true, however, that directed formulations are always stronger. Consider for example the following directed formulation of the traveling salesman problem.

$$\begin{aligned}
\text{minimize} \quad & \sum_{i,j\in V} c_{ij}y_{ij} & & \\
\text{subject to} \quad & \sum_{\{i|(i,j)\in A\}} y_{ij} = 1, & & j \in V, \\
& \sum_{\{j|(i,j)\in A\}} y_{ij} = 1, & & i \in V, \\
& \sum_{\{(i,j)\in A|i\in S,\, j\notin S\}} y_{ij} \geq 1, & & S \subset V,\ S \neq \emptyset, V, \\
& y_{ij} \in \{0,1\}, & & i,j \in V.
\end{aligned} \tag{1.11}$$

Starting with an undirected problem, we direct the problem, introducing two arcs (i,j) and (j,i) for every edge $e = \{i,j\}$, such that $c_{ij} = c_{ji}$. We denote by ZD_{TSP}, Z_{TSP} the values of the linear relaxation of the directed formulation (1.11) and the undirected formulation (1.5). We have (Exercise 1.22) that $ZD_{\text{TSP}} = Z_{\text{TSP}}$, that is, there is no benefit to passing to the directed formulation.

1.4 Modeling with exponentially many variables

In this section, we outline the column generation method of formulating discrete optimization problems as integer optimization problems involving an exponential number of variables. The general principle of formulating problems with an exponential number of variables is to enumerate all partially feasible solutions, and represent any additional constraints in a set packing, set covering or set partitioning type of formulation. The key idea is that we do not need to include all the variables a priori, but rather to generate new variables on demand. This method allows great flexibility in modeling very complicated restrictions and interdependencies that would otherwise be very difficult to model. For this reason, this methodology, which is called column generation, has had an important impact in practice. In Chapter 5, we discuss how such models can be solved. We illustrate the method with several examples.

Example 1.6 The cutting stock problem
Consider a paper company that has a supply of large rolls of paper, of width W. (We assume that W is a positive integer.) However, customer demand is for smaller widths of paper; in particular b_i rolls of width w_i, $i = 1,2,\ldots,m$, need to be produced. We assume that $w_i \leq W$ for each i, and that each w_i is an integer. Smaller rolls are obtained by slicing a large roll in a certain way, called a pattern. For example, a large roll of width 70 can be cut into three rolls of width $w_1 = 17$ and one roll of width $w_2 = 15$, with a waste of 4.

In general, a pattern, say the jth pattern, can be represented by a column vector $\mathbf{A}_j$, whose ith entry a_{ij} indicates how many rolls of width w_i are produced by that pattern. For example, the pattern described earlier is represented by the vector $(3,1,0,\ldots,0)$. For a vector $(a_{1j},\ldots,a_{mj})$ to be a representation of a feasible pattern, its components must be nonnegative integers and satisfy

$$\sum_{i=1}^{m} a_{ij}w_i \leq W.$$

Let n be the number of all feasible patterns and consider the $m \times n$ matrix $\mathbf{A}$ with columns $\mathbf{A}_j$, $j = 1,\ldots,n$. Note that n scales exponentially with m.

The goal of the company is to minimize the number of large rolls used while satisfying customer demand. Let x_j be the number of large rolls cut according to pattern j. Then, the problem under consideration is

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{n} x_j & & \\
\text{subject to} \quad & \sum_{j=1}^{n} a_{ij}x_j = b_i, & & i = 1,\ldots,m, \\
& x_j \in \mathbb{Z}_+, & & j = 1,\ldots,n.
\end{aligned}$$

Note that the above formulation has an exponential (in m) number of variables.

Example 1.7 Combinatorial auctions
With the explosion of the internet, auctions have increased in popularity in recent years. Examples include auctions for airport time slots, wireless bandwidth, railroad segments, delivery routes and rare stamps and coins, among others.

Let N be the set of bidders and M the set of m distinct items that are being auctioned. For every subset of items S of M let $b_j(S)$ be the bid that bidder $j \in N$ is willing to pay for the set S. It is natural to assume that if $S \cap T = \emptyset$, then $b_j(S) + b_j(T) \leq b_j(S \cup T)$, i.e., bidder j is willing to pay more for $S \cup T$ than for sets S and T individually. For example, a collector of rare stamps is willing to pay more for a complete collection than two incomplete collections. For this reason, it is natural to allow bidders to bid on combinations of different items. Such auctions are called combinatorial auctions. The key question in combinatorial auctions is to determine the winner of each item. We outline a method based on integer optimization that makes a fair determination possible.

Let $b(S) = \max_{j \in N} b_j(S)$. For every set $S \subseteq M$ we define the decision variable x_S to be one, if the highest bid on set S is accepted, and zero, otherwise. The problem of determining the winners in a combinatorial auctions can be formulated as follows:

$$\begin{aligned} \text{maximize} \quad & \sum_{S \subset M} b(S) x_S \\ \text{subject to} \quad & \sum_{S: i \in S} x_S \leq 1, \quad i \in M, \\ & x_S \in \{0,1\}, \quad S \subset M. \end{aligned}$$

The constraints in the above formulation ensure that no object in M is assigned to more than one bidder, while the objective function maximizes the total revenue from the auction. Note that the problem is a set packing problem involving $2^{|M|}$ variables.

Example 1.8 The vehicle routing problem
An undirected graph $G = (V, E)$ represents a transportation network. Node $i \in V$, for $i \neq 1$, represents customers with demand of d_i units. An additional node 0 represents the central depot. The travel costs are c_e for every arc $e \in E$. A company has m vehicles with capacities q_k, $k = 1, \ldots, m$ that need to visit all customers in order to satisfy demand. Each vehicle is to follow a route that starts at a central depot (node 0), visits some customers, and returns to the depot. We assume that the demand of each customer (a) can be carried by at least one single vehicle, and (b) cannot be divided into several vehicles. The basic vehicle routing problem is the problem of constructing routes for the vehicles starting and ending at the depot that minimize the total transportation cost.

Vehicle routing problems encountered in practice often involve additional requirements: (a) Each customer can only be visited within certain time windows; (b) In addition to delivery of items, the problem may also involve collection of items from the customer; (c) There may be multiple depots, multiple type of items, and loading and unloading times.

In order to model such problems, we start by enumerating feasible partial tours that satisfy all the restrictions. Note that because we use enumeration, we can model very complicated restrictions. Let x_j, $j = 1, \ldots, N$ be one, if partial tour j is used, and zero, otherwise. Clearly N is exponential in $|V|$. Let a_{ij} be one, if node i is visited in partial solution j. Let c_j be the cost of partial tour j. Since every node needs to be visited exactly once, the vehicle routing problem can be formulated as a set partitioning problem as follows:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{e}, \\ & \mathbf{x} \in \{0,1\}^N. \end{aligned}$$

1.5 Summary

The main message of this chapter is that strong formulations are central to being able to solve integer optimization problems efficiently. The quality of a formulation is judged by the closeness of its linear relaxation to the convex hull of integer feasible solutions. However, strong formulations occasionally require an exponential number of constraints. Finally, formulations give another view of the complexity of a discrete optimization problem, in the sense that for problems that are efficiently solvable, the strongest possible formulations (convex hull of integer feasible solutions) are often known.

For example, the description of the convex hull of the set of all matchings and all spanning trees is explicitly known. In contrast, the convex hull of the set of integer feasible solutions to the traveling salesman problem is not known. It is known that both the minimum spanning tree and matching problems are efficiently solvable. However, there is no known polynomial time algorithm for the traveling salesman problem. This suggests that our ability to find the strongest possible formulations of a discrete optimization problem (the convex hull of all integer feasible solutions) is directly related to our ability to solve it efficiently. In a sense, the complexity of a problem is characterized by our ability to construct a formulation, with a polynomial number of variables, whose linear relaxation is the convex hull of all integer feasible solutions.

1.6 Exercises

Exercise 1.1 (Disjunctive constraints) Suppose that we are given m constraints $\mathbf{a}_i'\mathbf{x} \geq b_i$, $i = 1, \ldots, m$, but without the restriction $\mathbf{a}_i \geq \mathbf{0}$. Model the requirement that at least k of them are satisfied. Assume that there exists a number γ such that $\mathbf{a}_i'\mathbf{x} \geq \gamma$ for $i = 1, \ldots, m$, and for all feasible $\mathbf{x}$.

Exercise 1.2 (Selection of the dream team) The coach of the national basketball team is faced with the decision of selecting 12 players for the upcoming international tournament. He has limited his final selection to 20 players, $p_1, \ldots, p_{20}$. For each player, the coach has collected several statistics that can be summarized as follows. His rebounding average r_i, his assists average a_i, his height h_i, his scoring average s_i, and his overall defense ability d_i. The players have been divided into four broad categories: play makers (PM) ($p_1, \ldots, p_5$),

shooting guards (SG) ($p_4, \ldots, p_{11}$), forwards (F) ($p_9, \ldots, p_{16}$), and centers (C) ($p_{16}, \ldots, p_{20}$). Notice that there are players that can be used in multiple roles (for example player p_4 can be used both as a play maker and a shooting guard). Players p_4, p_8, p_{15}, p_{20} play in the NCAA (college level), while all of the rest play in the NBA (professional level). For balance purposes, the team should consist of at least 3 play makers, 4 shooting guards, 4 forwards, and 3 centers, which implies that some players with dual roles should be selected. In addition, at least 2 players from the NCAA should be selected, while the mean rebounding, assists, scoring average, height, and defense ability should be at least r, a, s, h, d, respectively. The problem is further complicated by the fact that there are compatibility problems among some of the players. Player p_5 has declared that if player p_9 is selected, then he does not want to be in the team. Also, players p_2 and p_{19} can only be selected together as they play in the same team for years and feel that they are much more effective together. Finally, at most 3 players from the same team should be selected, so that the coach is not accused of favoritism (players p_1, p_7, p_{12}, p_{16} play for the same team). Faced with these difficulties, the coach has decided that he would like to maximize the scoring average, while satisfying the various constraints. Formulate the problem that the coach is facing as an integer optimization problem.

Exercise 1.3 (Playing times for the players in the dream team) This is a continuation of Exercise 1.2. After some careful thought, the coach would also like to decide how much play time to give to each player as some of these players in the initial list of 20, although extremely talented, were returning from long injuries and some were aging. For various reasons (injury, age) each player has an upper bound u_i on the average number of minutes he can play. In international tournaments, the duration of a game is 40 minutes. The coach has decided that there were two team compositions that he will use in the tournament depending on the type of opponent and circumstances in a game: (PM, SG, SG, F, C) or (PM, SG, F, F, C). Looking at the schedule, he predicts that these two schemes will be used equally in the tournament. Therefore, he realizes that the average play time of play makers would be 40 minutes, shooting guards 60 minutes, forwards 60 minutes, and centers 40 minutes. Formulate the combined problem of selection and allocation of average play time in order to maximize the scoring average.

Exercise 1.4 An airline operates a fleet of 15 jet aircraft, all equipped with the JET32 engine. The airline performs its own engine related repairs and maintenance at its repair facility. The maintenance director is reviewing the spare parts ordering and stocking policy for the next three years. The JET32 engine consists of 4 main modules, A, B, C, and D. When planes come in for repairs, sometimes the entire engine must be replaced because of extensive damage and wear. More often, however, only certain modules need replacement. The following table contains the forecasted requirements for individual engine modules and complete engines for the next 3 years. The airline places orders for complete engines and modules at the beginning of the year with JET Inc., the manufacturer of the JET32 engine. The following table shows the projected prices for engines and modules that JET Inc. might charge in the next three years.

Note that complete engines cost less than the total cost of buying one module of each type. Assume that the cost of "cannibalizing," i.e., breaking a complete engine into four individual modules, is negligible compared to the cost

Year	Module A	Module B	Module C	Module D	Complete Engine
1	5	4	4	2	1
2	2	1	1	7	0
3	3	4	3	0	2

Table 1.1: Forecasted engine/module requirements.

Year	Module A	Module B	Module C	Module D	Complete Engine
1	0.5	2.0	5.0	1.0	7.8
2	0.6	2.2	5.5	1.1	7.5
3	0.7	2.5	6.0	1.3	7.0

Table 1.2: Forecasted engine/module prices.

of these modules. The mix of engines and modules that the airline orders from JET Inc. must, therefore, account for the economies in ordering complete engines. Assuming that the airline does not have any inventory of modules or engines in hand, formulate an integer optimization problem to determine the order quantities for the next 3 years, while minimizing the total cost of purchases. Assume that there are no inventory carrying costs.

Exercise 1.5 (Plan for a move) Suppose you are planning to move to your new house. You have n items of size a_j, $j = 1, \ldots, n$, that need to be moved. You have rented a truck that has size Q and you have bought m boxes. Box i has size b_i, $i = 1, \ldots, m$. Formulate an integer optimization problem in order to decide whether the move is possible. Note that you can put multiple items in the same box and size is the only criterion determining if an item can be put into a box.

Exercise 1.6 (Separable piecewise nonlinear optimization) Consider the separable nonlinear optimization problem

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{n} f_{0,j}(x_j) \\ \text{subject to} \quad & \sum_{j=1}^{n} f_{i,j}(x_j) \leq b_i, \quad i = 1, \ldots, m, \\ & 0 \leq x_j \leq u_j, \qquad\quad j = 1, \ldots, n, \end{aligned}$$

where the functions $f_{i,j}(x_j)$, $i = 0, 1, \ldots, m$, $j = 1, \ldots, n$ are continuous, piecewise linear with at most μ pieces. Propose an integer optimization formulation to the separable nonlinear optimization problem.

Exercise 1.7 (Data mining)

(a) We are given points $(\boldsymbol{x}_i, a_i)$, $i = 1, \ldots, m$, where $\boldsymbol{x}_i \in \Re^n$ and $a_i \in \{0, 1\}$, which indicates the category that point $\boldsymbol{x}_i$ belongs to. We would like to

decide whether it is possible to separate the points $\boldsymbol{x}_i$ by a hyperplane $\boldsymbol{c}'\boldsymbol{x} = 1$ such that all points of category 0 satisfy $\boldsymbol{c}'\boldsymbol{x} \leq 1$ and all points of category 1 satisfy $\boldsymbol{c}'\boldsymbol{x} > 1$. Propose a linear optimization problem to find the vector $\boldsymbol{c}$.

(b) We are given points $(\boldsymbol{x}_i, y_i)$, $i = 1, \ldots, m$, where $\boldsymbol{x}_i \in \Re^n$ and $y_i \in \Re$. We would like to find a hyperplane $\boldsymbol{c}'\boldsymbol{x} = 1$ and a partition of the sets of points into two sets with the following properties:

(i) for all points $(\boldsymbol{x}_i, y_i)$ such that $\boldsymbol{c}'\boldsymbol{x}_i \leq 1$ we want to detect a vector $\boldsymbol{\beta}_1$ so as to minimize

$$\sum_{\{i:\, \mathbf{c}'\mathbf{x}_i \leq 1\}} |y_i - \boldsymbol{\beta}_1'\mathbf{x}_i|.$$

(ii) for all points $(\boldsymbol{x}_i, y_i)$ such that $\boldsymbol{c}'\boldsymbol{x}_i > 1$ we want to detect a vector $\boldsymbol{\beta}_2$, so as to minimize

$$\sum_{\{i:\, \mathbf{c}'\mathbf{x}_i > 1\}} |y_i - \boldsymbol{\beta}_2'\mathbf{x}_i|.$$

Propose an integer optimization problem to detect the vectors $\mathbf{c}$, $\boldsymbol{\beta}_1$, $\boldsymbol{\beta}_2$.

Exercise 1.8 (Constructing piecewise quadratic approximations) Given n points with coordinates (x_i, y_i), $i = 1, 2, \ldots, n$, such that $x_1 < x_2 < \ldots < x_n$, we would like to construct a piecewise quadratic approximating function $f(x)$ to the points y_i, $i = 1, 2, \ldots, n$ consisting of $n - 1$ pieces. The function $f(x)$ is as follows:

$$f(x) = \begin{cases} a_1\, x^2 + b_1\, x + c_1, & x_1 \leq x < x_2, \\ a_2\, x^2 + b_2\, x + c_2, & x_2 \leq x < x_3, \\ \quad\vdots & \quad\vdots \\ a_{n-1}\, x^2 + b_{n-1}\, x + c_{n-1}, & x_{n-1} \leq x \leq x_n. \end{cases}$$

Our objective is to select the coefficients (a_i, b_i, c_i), $i = 1, 2, \ldots, n-1$ so that the function $f(x)$ is continuous and convex. Recall that a function $f(x)$ is convex if its derivative $f'(x)$ is nondecreasing.

(a) Suppose we are interested in minimizing the error

$$\sum_{i=1}^{n} |y_i - f(x_i)|.$$

Propose a linear optimization model to accomplish this.

(b) Instead of constructing an approximating function $f(x)$ consisting of $n - 1$ pieces, we can construct a piecewise quadratic approximating function $g(x)$ consisting of $k - 1$ pieces, $k < n$. The function $g(x)$ is a follows:

$$g(x) = \begin{cases} a_1\, x^2 + b_1\, x + c_1, & z_1 \leq x < z_2, \\ a_2\, x^2 + b_2\, x + c_2, & z_2 \leq x < z_3, \\ \quad\vdots & \quad\vdots \\ a_{k-1}\, x^2 + b_{k-1}\, x + c_{k-1}, & z_{k-1} \leq x \leq z_k, \end{cases}$$

where $z_1 = x_1$ and $z_k = x_n$. The other points $z_2, z_3, \ldots, z_{k-1}$ can be chosen among the set $\{x_2, \ldots, x_{n-1}\}$. We would still like to impose the condition that $g(x)$ is continuous and convex. Suppose we are interested in minimizing the error $\sum_{i=1}^{n} |y_i - g(x_i)|$. Propose a linear integer optimization model to accomplish this.

Exercise 1.9 (Linear regression and extensions) In the linear regression model, we assume that there are d factors $X_1, \ldots, X_d$ that affect the output variable Y in a linear way, i.e., there are coefficients β_0 and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)'$ such that

$$Y = \beta_0 + \boldsymbol{\beta}'\mathbf{X} + \epsilon,$$

where ϵ is a normally distributed random variable with mean zero and standard deviation σ. We are given data $(y_i, \mathbf{x}_i)$, $i = 1, \ldots, n$, where $y_i \in \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}^d$. The values y_i, $\mathbf{x}_i$, $i = 1, \ldots, n$ are realizations of the output Y and the factors $\mathbf{X}$. We propose several ways to find the coefficients β_0 and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)'$.

(a) The classical least squares problem is:

$$\text{minimize} \sum_{i=1}^{n} (y_i - \beta_0 - \boldsymbol{\beta}'\mathbf{x}_i)^2.$$

Find explicit expressions of the optimal solutions for β_0^* and $\boldsymbol{\beta}^*$.

(b) Consider the problem

$$\text{minimize} \sum_{i=1}^{n} |y_i - \beta_0 - \boldsymbol{\beta}'\mathbf{x}_i|,$$

and propose an efficient way to solve it.

(c) In this part we consider the factor selection problem, in which we stipulate that only $k < d$ factors $X_1, \ldots, X_d$ affect Y. Extend your formulation in part (b) to model the additional constraint that up to k out of the d coefficients $\beta_1, \ldots, \beta_d$ have nonzero values.

(d) Given $n = 2k + 1$ numbers a_i, $i = 1, \ldots, n$ median $(a_1, \ldots, a_n)$ is that number a among a_i, $i = 1, \ldots, n$ with the property that k of the a_i's are larger than or equal to a, and k of the a_i's are smaller than or equal to a. For example, median $(3, 5, -2, 1, 7) = 3$. Compared to the mean, the median is much less sensitive to outliers, and thus more "robust". Formulate the **robust linear regression problem** for $n = 2k + 1$:

$$\text{minimize} \quad \text{median}\, (|y_1 - \beta_0 - \boldsymbol{\beta}'\mathbf{x}_1|, \ldots, |y_n - \beta_0 - \boldsymbol{\beta}'\mathbf{x}_n|)$$

as an integer optimization problem.

Exercise 1.10 (A production and distribution problem) A company produces a set of K products at I plants. It then ships these products to J market zones. For $k = 1, \ldots, K$, $i = 1, \ldots, I$, and $j = 1, \ldots, J$, the following data are

given:

v_{ik} = variable cost of producing one unit of product k at plant i,
c_{ijk} = cost of shipping one unit of product k from plant i to zone j,
f_{ik} = fixed cost associated with producing product k at plant i,
M_{ik} = maximal quantity of product k produced at plant i,
m_{ik} = minimal quantity of product k that can be produced at plant i, if plant i produces a nonzero quantity,
q_{ik} = capacity of plant i used to produce one unit of product k,
Q_i = capacity of plant i,
d_{jk} = demand for product k at market zone j.

(a) Formulate the problem of minimizing the total cost of production and transportation that the company is facing, as an integer optimization problem. Indicate how your model can incorporate the following additional constraints.

(b) No plant may produce more than K_1 products.

(c) Every product can be produced in at most I_1 plants.

(d) For a particular product k_0, plant 3 must produce it if neither plant 1 nor plant 2 produce it.

(e) Each market zone must be sourced by exactly one plant for all products.

Exercise 1.11 (A dynamic single item lot sizing problem) We consider the production of a single product over T periods. If we decide to produce at period t, a setup cost c_t is incurred. For $t = 1, \ldots, T$, let d_t be the demand for this product in period t, and let p_t, h_t be the unit production cost and unit storage cost (per period), respectively.

(a) Formulate an integer optimization problem in order to minimize the total cost of production, storage, and setup.

(b) Suppose we allow demand to be lost in every period except for period T, at a cost of b_t per unit of lost demand. Show how to modify the model to handle this constraint.

(c) Suppose that production can occur in at most five periods, but no two such periods can be consecutive. Show how to modify the model to handle this option.

Exercise 1.12 (The survivable network design problem) Show that Eqs. (1.7) correctly model the survivable network design problem.

Exercise 1.13 (The vehicle routing problem) An undirected graph $G = (V, E)$ represents a transportation network. Node $i \in V$, for $i \neq 1$, represents customers with demand of b_i units. The travel costs are d_e for every arc $e \in E$. A company has m vehicles, each of capacity Q, that need to visit all customers in order to satisfy demand. Each vehicle is to follow a route that starts at a central depot (node 1), visits some customers, and returns to the depot. Suppose that the demand of each customer can be carried by a single vehicle, i.e., $b_i \leq Q$, for all i. Assuming that the demand of any customer cannot be divided into several vehicles, formulate the problem of constructing routes for the vehicles that minimize the total transportation cost.

Exercise 1.14 (The fixed charge network design problem) We are given a directed graph $G = (V, A)$ and a demand or supply b_i for each $i \in V$, such that $\sum_{i \in V} b_i = 0$. There are two types of costs: transportation costs c_{ij} of shipping one unit from node i to node j, and building costs d_{ij} of establishing a link (i, j) between nodes i and j of capacity u_{ij}. We would like to build such a network in order to minimize the total building and transportation costs, so that all demand is met. Formulate the problem as an integer optimization problem.

Exercise 1.15 (Job shop scheduling) A factory consists of m machines M_1, $\ldots$, M_m, and needs to process n jobs every day. Job j needs to be processed once by each machine in the order $(M_{j(1)}, \ldots, M_{j(m)})$. Machine M_i takes time p_{ij} to process job j. A machine can only process one job at a time, and once a job is started on any machine, it must be processed to completion. The objective is to minimize the sum of the completion times of all the jobs. Provide an integer optimization formulation for this problem.

Exercise 1.16 (Facility location) For the facility location problem, prove that the inclusion $P_{\text{FL}} \subset P_{\text{AFL}}$ can be strict.

Exercise 1.17 Let $G = (V, E)$ be an undirected graph with n nodes. Show that G is a tree if and only if the total number of edges is $n - 1$, and for any nonempty set $S \subset V$, the number of edges with both endpoints in S is less than or equal to $|S| - 1$.

Exercise 1.18 (A multicut formulation of the MST problem) Given an undirected graph $G = (V, E)$, with $|V| = n$ and $|E| = m$, consider a partition of V into disjoint nonempty sets $C_0, C_1, \ldots, C_k$ of nodes, whose union is V. Let $\delta(C_0, C_1, \ldots, C_k)$ be the set of edges, whose endpoints lie in different sets C_i. Let

$$P_{\text{mcut}} = \left\{ \mathbf{x} \in \mathbb{R}^m \;\middle|\; 0 \leq x_e \leq 1, \; \sum_{e \in E} x_e = n - 1, \; \sum_{e \in \delta(C_0, C_1, \ldots, C_k)} x_e \geq k, \text{ for all } k \text{ and for all partitions } C_0, C_1, \ldots, C_k \text{ of } V \right\}.$$

Prove that $P_{\text{mcut}} = P_{\text{sub}}$, where

$$P_{\text{sub}} = \left\{ \mathbf{x} \in \mathbb{R}^m \;\middle|\; 0 \leq x_e \leq 1, \; \sum_{e \in E} x_e = n - 1, \; \sum_{e \in E(S)} x_e \leq |S| - 1, \; S \subset V, S \neq \emptyset, V \right\}.$$

Exercise 1.19 (A directed cut formulation of the MST) Given an undirected graph $G = (V, E)$, with $|V| = n$ and $|E| = m$, form a directed graph $D = (V, A)$ by replacing each edge $\{i, j\}$ in E by arcs (i, j), (j, i) in A. We select a node r in V as the root node. Let $y_{ij} = 1$ if the tree contains arc (i, j) when we root the tree at node r (in other words the solution will be a tree with directed edges away from the root). Let $\delta^+(S)$ be the set of arcs going out of S. Let

$$P_{dcut} = \Big\{ \mathbf{x} \in \mathbb{R}^m | \ \mathbf{0} \leq \mathbf{x} \leq \mathbf{e}, \ x_e = y_{ij} + y_{ji}, \ \forall \ e \in E,$$

$$\sum_{e \in A} y_e = n - 1, \ \sum_{e \in \delta^+(S)} y_e \geq 1, \ r \in S, \ \forall S \subset V, \ y_e \geq 0 \Big\}.$$

Prove that $P_{dcut} = P_{\text{sub}}$.

Exercise 1.20 (The undirected traveling salesman problem) For the undirected traveling salesman problem, prove that

$$P_{\text{tspcut}} = P_{\text{tspsub}}.$$

Exercise 1.21* (The directed traveling salesman problem) Given a directed graph $G = (V, A)$, with $|V| = n$ and $|A| = m$, a natural extension of the traveling salesman formulation in Eq. (1.5), involves the constraints:

$$\begin{aligned}
\sum_{\{i|(i,j)\in A\}} y_{ij} &= 1, && j \in V, \\
\sum_{\{j|(i,j)\in A\}} y_{ij} &= 1, && i \in V, \\
\sum_{\{(i,j)\in A | i\in S, \, j\notin S\}} y_{ij} &\geq 1, && S \subset V, \ S \neq \emptyset, V, \\
y_{ij} &\in \{0, 1\}, && i, j \in V.
\end{aligned}$$

Let $\mathcal{F}$ be the set of feasible solutions.

(a) Consider now the following set of (polynomially many) constraints:

$$\begin{aligned}
u_i - u_j + n y_{ij} &\leq n - 1, && (i, j) \in A, \ i, j \neq 1, \\
\sum_{\{i|(i,j)\in A\}} y_{ij} &= 1, && j \in V, \\
\sum_{\{j|(i,j)\in A\}} y_{ij} &= 1, && i \in V, \\
y_{ij} &\in \{0, 1\}, && i, j \in V.
\end{aligned}$$

Let $\mathcal{F}'$ be the set of feasible solutions. Prove that $\mathcal{F} = \mathcal{F}'$.

(b) Let $P_{\text{tsp}-\text{dcut}}$ and $P_{\text{tsp}-\text{polynomial}}$ be the polyhedra associated with the linear relaxations of the formulations corresponding to $\mathcal{F}$ and $\mathcal{F}'$, respectively. Prove that

$$P_{\text{tsp}-\text{dcut}} \subset P_{\text{tsp}-\text{polynomial}},$$

and that the inclusion can be strict, i.e., the first formulation is stronger.

(c) Prove that $P_{\text{tsp}-\text{dcut}} \neq \text{conv}(\mathcal{F})$.

Exercise 1.22 (Directed versus undirected formulations for the traveling salesman problem) Let ZD_{TSP}, Z_{TSP} be the values of the linear relaxations of the directed formulation (1.11) and the undirected formulation (1.5), respectively. Show that $ZD_{\text{TSP}} = Z_{\text{TSP}}$.

Exercise 1.23 (Computational Exercise: Facility Location) This exercise refers to Examples 1.2 and 1.4. We are given 20 possible facility locations that can serve 200 customers. The fixed costs and the costs of servicing a customer from a specific location are given in the file FL.dat. The file FL.prj describes the formulation introduced in Example 1.2, while AFL.prj is the aggregate formulation introduced in Example 1.4. We are solving the linear relaxations of both formulations.

(a) Solve the facility location problem under the two formulations. Which formulation gives a better objective function value?

(b) Record the number of constraints and the number of variables under both formulations. Which formulation has a smaller dimension?

(c) What can you observe about the optimal decision variables $\boldsymbol{y}$?

(d) Relate your answers in parts (a)-(c) to what has been discussed in this chapter. Which formulation is better?

1.7 Notes and sources

This chapter uses extensively material from Bertsimas and Tsitsiklis (1997), Chapter 10.

1.1. The derivation of integer optimization models is more an art than a formal methodology. The journal *Interfaces* often publishes interesting large scale discrete optimization models. Examples of integer optimization modeling techniques can also be found in nearly all textbooks about integer optimization. We refer, in particular to the books Papadimitriou and Steiglitz (1982), Nemhauser and Wolsey (1988), Williams (1978) and Wolsey (1998).

1.3. Edmonds (1971) has shown that the convex hull of the integer feasible solutions to the minimum spanning tree problem is given by P_{sub}. The survey paper by Magnanti and Wolsey (1995) discusses many different formulations for tree related problems, and their applications to discrete optimization. The formulation of the travelling salesman problem with exponentially many constraints has been introduced by Dantzig et al. (1954, 1959). More information on the traveling salesman problem can be found in Lawler et al. (1985). Edmonds (1965) provided a polynomial time algorithm for the matching problem and showed that the convex hull of the integer feasible solutions to the matching problem is given by P_{matching}. For a textbook exposition of matching algorithms we refer to Papadimitriou and Steiglitz (1982), Nemhauser and Wolsey (1988), Cook et al. (1998) and Korte and Vygen (2000). A thorough treatment of the theory of matchings

can be found in Lovász and Plummer (1986). For an analysis and a computational investigation of the directed versus the undirected formulation for Steiner tree problems we refer to the papers by Chopra and Rao (1994a,b). Cut covering problems have been considered in Goemans and Williamson (1995a).

1.4. The cutting stock problem was introduced in Gilmore and Gomory (1961). For a survey on combinatorial auctions see de Vries and Vohra (2003).

Chapter 2

Methods to enhance formulations

Contents