1

Formulations

1.1 Introduction

A wide variety of practical problems can be formulated and solved using integer programming. We start by describing briefly a few such problems.

1. *Train Scheduling*. Certain train schedules repeat every hour. For each line, the travel times between stations are known, and the time spent in a station must lie within a given time interval. Two trains traveling on the same line must for obvious reasons be separated by at least a given number of minutes. To make a connection between trains A and B at a particular station, the difference between the arrival time of A and the departure time of B must be sufficiently long to allow passengers to change, but sufficiently short so that the waiting time is not excessive. The problem is to find a feasible schedule.
2. *Airline Crew Scheduling*. Given the schedule of flights for a particular aircraft type, one problem is to design weekly schedules for the crews. Each day a crew must be assigned a duty period consisting of a set of one or more linking flights satisfying numerous constraints such as limited total flying time, minimum rests between flights, and so on. Then putting together the duty periods, weekly schedules or pairings are constructed which must satisfy further constraints on overnight rests, flying time, returning the crew to its starting point, and so on. The objective is to minimize the amount paid to the crews, which is a function of flying time, length of the duty periods and pairings, a guaranteed minimum number of flying hours, and so forth.
3. *Production Planning*. A multinational company holds a monthly planning meeting in which a three-month production and shipping plan is drawn up based on their latest estimates of potential sales. The plan covers 200–400 products produced in five different factories with shipments to 50 sales areas.

Integer Programming, Second Edition. Laurence A. Wolsey.

Companion website: www.wiley.com/go/wolsey/integerprogramming2e

Solutions must be generated on the spot, so only about 15 minutes' computation time is available. For each product, there is a minimum production quantity, and production is in batches – multiples of some fixed amount. The goal is to maximize contribution.

4. *Electricity Generation Planning*. A universal problem is the unit commitment problem of developing an hourly schedule spanning a day or a week so as to decide which generators will be producing and at what levels. Constraints to be satisfied include satisfaction of estimated hourly or half-hourly demand, reserve constraints to ensure that the capacity of the active generators is sufficient should there be a sudden peak in demand, and ramping constraints to ensure that the rate of change of the output of a generator is not excessive. Generators have minimum on- and off-times, and their start-up costs are a nonlinear function of the time they have been idle.
5. *Telecommunications*. A typical problem given the explosion of demand in this area concerns the installation of new capacity so as to satisfy a predicted demand for data/voice/video transmission. Given estimates of the requirements between different centers, the existing capacity and the costs of installing new capacity which is only available in discrete amounts, the problem is to minimize cost taking into account the possibility of failure of a line or a center due to a breakdown or accident.
6. *Radiation Therapy Treatment*. In intensity modulated radiation therapy, both the shape of the beam and its intensity need to be controlled by opening and closing multileaf collimators. The goal is to provide the tumor coverage required while maintaining low radiation on critical and normal tissues.
7. *Kidney Exchange Programs*. Patients suffering from kidney failures require a transplant. Though a patient may have a donor willing to donate a kidney, an exchange between the donor–patient pair may be impossible for reasons of blood or tissue incompatibility. The problem is to organize a "best possible" set of exchanges between a number of such pairs in which a patient from one pair receives a compatible kidney from a donor in another pair.
8. *Cutting Problems*. Whether cutting lengths of paper from rolls, plastic from large rectangular sheets, or patterns to make clothes, the problem is in each case to follow precisely determined cutting rules, satisfy demand, and minimize waste.

Other recent application areas include problems in molecular biology, statistics, VLSI, and machine learning.

This book tries to provide some of the understanding and tools necessary for tackling such problems.

1.2 What Is an Integer Program?

Suppose that we have a *linear program*

$$\text{(LP)} \qquad \max\{cx : Ax \leq b, x \geq 0\}$$

where A is an m by n matrix, c an n-dimensional row vector, b an m-dimensional column vector, and x an n-dimensional column vector of variables or unknowns. Now, we add in the restriction that certain variables must take integer values.

If some but not all variables are integer, we have a

(Linear) Mixed Integer Program, written as

$$\text{(MIP)} \qquad \begin{array}{rcl} \max cx & + & hy \\ Ax & + & Gy \leq b \\ x & \geq & 0 \text{ and integer}, y \geq 0 \end{array}$$

where A is again m by n, G is m by p, c is an n row-vector, h is a p row-vector, x is an n column-vector of integer variables, and y is a p column-vector of real variables.

If all variables are integer, we have a

(Linear) Integer Program, written as

$$\text{(IP)} \qquad \begin{array}{rcl} \max cx & & \\ Ax & \leq & b \\ x & \geq & 0 \text{ and integer}, \end{array}$$

and if all variables are restricted to 0–1 values, we have a

0–1 or Binary Integer Program

$$\text{(BIP)} \qquad \begin{array}{rcl} \max cx & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n. \end{array}$$

Throughout the text, we will normally use the convention that if there is only one set of variables, they are denoted by x. If there are both real and integer variables, x and possibly z will denote integer variables and y and w real variables. Note that this differs from the choice of x and y in the 1st edition.

Another type of problem that we wish to study is a "combinatorial optimization problem." Here typically we are given a finite set $N = \{1, \dots, n\}$, weights c_j for each $j \in N$, and a set $\mathcal{F}$ of feasible subsets of N. The problem of finding a minimum weight feasible subset can be expressed as follows:

Combinatorial Optimization Problem

$$\text{(COP)} \qquad \min_{S \subseteq N} \left\{ \sum_{j \in S} c_j \; : \; S \in \mathcal{F} \right\}.$$

In Section 1.3, we will see various examples of integer programs (IPs) and combinatorial optimization problems (COPs), and also see that often a COP can be formulated as an IP or a 0–1 IP.

Given that integer programs look very much like linear programs, it is not surprising that linear programming theory and practice is fundamental in understanding and solving integer programs. However, the first idea that springs to mind, namely "rounding," is often insufficient, as the following example shows:

Example 1.1 Consider the integer program:

$$\begin{array}{lllcl} \max & 1.00x_1 & +0.64x_2 & & \\ & 50x_1 & +31x_2 & \leq & 250 \\ & 3x_1 & -2x_2 & \geq & -4 \\ & x_1, & x_2 & \geq & 0 \text{ and integer.} \end{array}$$

As we see from Figure 1.1, the linear programming solution (376/193, 950/193) is a long way from the optimal integer solution (5, 0). □

For 0–1 IPs, the situation is often even worse. The linear programming solution may well be (0.5, … , 0.5), giving no information whatsoever. What is more, it is typically very difficult just to answer the question whether there exists a feasible 0–1 solution.

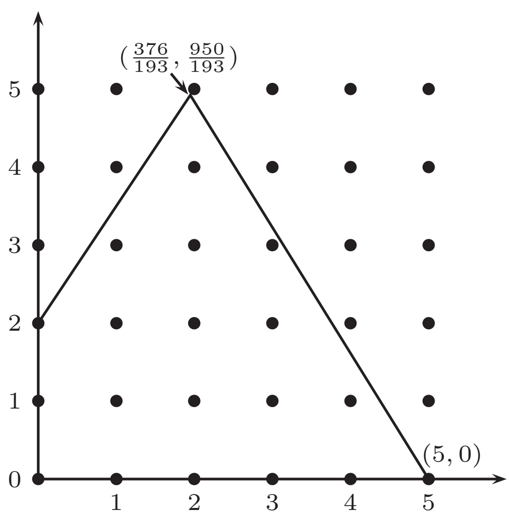


Figure 1.1 LP and IP solutions.

1.3 Formulating IPs and BIPs

As in linear programming, translating a problem description into a formulation should be done systematically. A clear distinction should be made between the data of the problem instance and the variables (or unknowns) used in the model.

(i) Define what appear to be the necessary variables.
(ii) Use these variables to define a set of constraints so that the feasible points correspond to the feasible solutions of the problem.
(iii) Use these variables to define the objective function.

If difficulties arise, define an additional or alternative set of variables and iterate.

Defining variables and constraints may not always be as easy as in linear programming. Especially for COPs, we are often interested in choosing a subset $S \subseteq N$. For this, we typically make use of the *incidence vector of* S, which is the n-dimensional 0–1 vector x^S such that $x^S_j = 1$ if $j \in S$, and $x^S_j = 0$ otherwise.

Below we formulate four well-known integer programming problems.

The Assignment Problem

There are n people available to carry out n jobs. Each person is assigned to carry out exactly one job. Some individuals are better suited to particular jobs than others, so there is an estimated cost c_{ij} if person i is assigned to job j. The problem is to find a minimum cost assignment.

Definition of the variables.
$x_{ij} = 1$ if person i does job j, and $x_{ij} = 0$ otherwise.
Definition of the constraints.
Each person i does one job:

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \text{for } i = 1, \dots, n.$$

Each job j is done by one person:

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \text{for } j = 1, \dots, n.$$

The variables are 0–1:

$$x_{ij} \in \{0, 1\} \quad \text{for } i = 1, \dots, n, j = 1, \dots, n.$$

Definition of the objective function.
The cost of the assignment is minimized:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}.$$

The 0–1 Knapsack Problem

There is a budget b available for investment in projects during the coming year and n projects are under consideration, where a_j is the outlay for project j and c_j is its expected return. The goal is to choose a set of projects so that the budget is not exceeded and the expected return is maximized.

Definition of the variables.
$x_j = 1$ if project j is selected, and $x_j = 0$ otherwise.
Definition of the constraints.
The budget cannot be exceeded:

$$\sum_{j=1}^{n} a_j x_j \leq b.$$

The variables are 0–1:

$$x_j \in \{0, 1\} \quad \text{for } j = 1, \dots, n.$$

Definition of the objective function.
The expected return is maximized:

$$\max \sum_{j=1}^{n} c_j x_j.$$

The Set Covering Problem

Given a certain number of regions, the problem is to decide where to install a set of emergency service centers. For each possible center, the cost of installing a service center and which regions it can service are known. For instance, if the centers are fire stations, a station can service those regions for which a fire engine is guaranteed to arrive on the scene of a fire within eight minutes. The goal is to choose a minimum cost set of service centers so that each region is covered.

First, we can formulate it as a more abstract COP. Let $M = \{1, \dots, m\}$ be the set of regions, and $N = \{1, \dots, n\}$ the set of potential centers. Let $S_j \subseteq M$ be the regions that can be serviced by a center at $j \in N$ and c_j its installation cost. We obtain the problem:

$$\min_{T \subseteq N} \left\{ \sum_{j \in T} c_j \,:\, \cup_{j \in T} S_j = M \right\}.$$

Now, we formulate it as a 0–1 IP. To facilitate the description, we first construct a 0–1 *incidence matrix* A such that $a_{ij} = 1$ if $i \in S_j$ and $a_{ij} = 0$, otherwise. Note that this is nothing but processing of the data.

Definition of the variables.

$x_j = 1$ if center j is selected, and $x_j = 0$ otherwise.

Definition of the constraints.

At least one center must service region i:

$$\sum_{j=1}^{n} a_{ij}x_j \geq 1 \quad \text{for } i = 1, \dots, m.$$

The variables are 0–1:

$$x_j \in \{0, 1\} \quad \text{for } j = 1, \dots, n.$$

Definition of the objective function.

The total cost is minimized:

$$\min \sum_{j=1}^{n} c_j x_j.$$

The Traveling Salesman Problem (TSP)

This is perhaps the most notorious problem in Operations Research because it is so easy to explain, and so tempting to try and solve. A salesman must visit each of n cities exactly once and then return to his starting point. The time taken to travel from city i to city j is c_{ij}. Find the order in which he should make his tour so as to finish as quickly as possible.

This problem arises in a multitude of forms: a truck driver has a list of clients he must visit on a given day, or a machine must place modules on printed circuit boards, or a stacker crane must pick up and depose crates. Now, we formulate it as a 0–1 IP.

Definition of the variables.

$x_{ij} = 1$ if the salesman goes directly from town i to town j, and $x_{ij} = 0$, otherwise. (x_{ii} is not defined for $i = 1, \dots, n$.)

Definition of the constraints.

He leaves town i exactly once:

$$\sum_{j:j \neq i} x_{ij} = 1 \quad \text{for } i = 1, \dots, n.$$

He arrives at town j exactly once:

$$\sum_{i:i \neq j} x_{ij} = 1 \quad \text{for } j = 1, \dots, n.$$

So far these are precisely the constraints of the assignment problem. A solution to the assignment problem might give a solution of the form shown in Figure 1.2 (i.e. a set of disconnected subtours). To eliminate these solutions, we need more

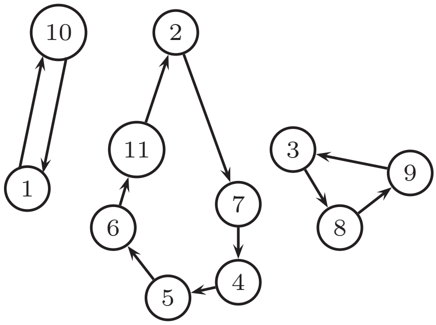


Figure 1.2 Subtours.

constraints that guarantee connectivity by imposing that the salesman must pass from one set of cities to another, so-called *cut-set* constraints:

$$\sum_{i\in S}\sum_{j\notin S} x_{ij} \geq 1 \quad \text{for } S \subset N, S \neq \phi.$$

An alternative is to replace these constraints by *subtour elimination* constraints:

$$\sum_{i\in S}\sum_{j\in S} x_{ij} \leq\mid S \mid -1 \quad \text{for } S \subset N, 2 \leq |S| \leq n-1.$$

The variables are 0–1:

$$x_{ij} \in \{0,1\} \quad \text{for } i = 1,\dots,n, j = 1,\dots,n, i \neq j.$$

Definition of the objective function.
The total travel time is minimized:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}.$$

1.4 The Combinatorial Explosion

The four problems we have looked at so far are all combinatorial in the sense that the optimal solution is some subset of a finite set. Thus, in principle, these problems can be solved by enumeration. To see for what size of problem instances this is a feasible approach, we need to count the number of possible solutions.

The Assignment Problem. There is a one-to-one correspondence between assignments and permutations of $\{1,\dots,n\}$. Thus, there are $n!$ solutions to compare.
The Knapsack and Covering Problems. In both cases, the number of subsets is 2^n. For the knapsack problem with $b = \sum_{j=1}^{n} a_j/2$, at least half of the subsets are feasible and thus there are at least 2^{n-1} feasible subsets.

Table 1.1 Some typical functions.

n	$\log n$	$n^{0.5}$	n^2	2^n	$n!$
10	3.32	3.16	10^2	1.02×10^3	3.6×10^6
100	6.64	10.00	10^4	1.27×10^{30}	9.33×10^{157}
1000	9.97	31.62	10^6	1.07×10^{301}	4.02×10^{2567}

The Traveling Salesman Problem. Starting at city 1, the salesman has $n-1$ choices. For the next choice $n-2$ cities are possible, and so on. Thus, there are $(n-1)!$ feasible tours.

In Table 1.1, we show how rapidly certain functions grow. Thus, a traveling salesman problem (TSP) with $n = 101$ has approximately 9.33×10^{157} tours.

The conclusion to be drawn is that using complete enumeration we can only hope to solve such problems for very small values of n. Therefore, we have to devise some more intelligent algorithms, otherwise, the reader can throw this book out of the window.

1.5 Mixed Integer Formulations

Modeling Fixed Costs

Suppose we wish to model a typical nonlinear fixed charge cost function:

$$h(y) = f + py \text{ if } 0 < y \le C \text{ and } h(y) = 0 \text{ if } y = 0$$

with $f > 0$ and $p > 0$ (see Figure 1.3).

Definition of an additional variable.

$$x = 1 \text{ if } y > 0 \text{ and } x = 0 \text{ otherwise.}$$

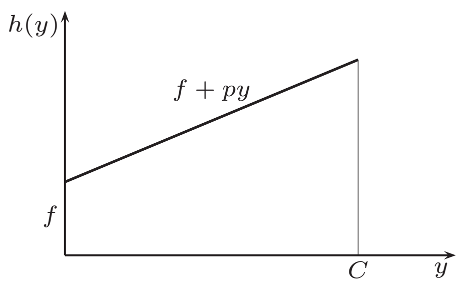


Figure 1.3 Fixed cost function.

Definition of the constraints and objective function.
We replace $h(y)$ by $fx + py$, and add the constraints $y \leq Cx, x \in \{0, 1\}$.

Note that this is not a completely satisfactory formulation, because although the costs are correct when $y > 0$, it is possible to have the solution $y = 0, x = 1$. However, as the objective is minimization, this will typically not arise in an optimal solution.

Uncapacitated Facility Location (UFL)

Given a set of potential depots $N = \{1, \dots, n\}$ and a set $M = \{1, \dots, m\}$ of clients, suppose there is a fixed cost f_j associated with the use of depot j, and a transportation cost c_{ij} if all of client i's order is delivered from depot j. The problem is to decide which depots to open and which depot serves each client so as to minimize the sum of the fixed and transportation costs. Note that this problem is similar to the covering problem, except for the addition of the variable transportation costs.

Definition of the variables.
We introduce a fixed cost or depot opening variable $x_j = 1$ if depot j is used, and $x_j = 0$ otherwise.
y_{ij} is the fraction of the demand of client i satisfied from depot j.
Definition of the constraints.
Satisfaction of the demand of client i:

$$\sum_{j=1}^{n} y_{ij} = 1 \quad \text{for } i = 1, \dots, m.$$

To represent the link between the y_{ij} and the x_j variables, we note that $\sum_{i\in M} y_{ij} \leq m$, and use the fixed cost formulation above to obtain:

$$\sum_{i\in M} y_{ij} \leq m x_j \text{ for } j \in N,\ y_{ij} \geq 0 \text{ for } i \in M, j \in N,\ x_j \in \{0, 1\} \text{ for } j \in N.$$

Definition of the objective function.
The objective is $\sum_{j\in N} h_j(y_{1j}, \dots, y_{mj})$, where $h_j(y_{1j}, \dots, y_{mj}) = f_j + \sum_{i\in M} c_{ij} y_{ij}$ if $\sum_{i\in M} y_{ij} > 0$, so we obtain

$$\min \sum_{i\in M} \sum_{j\in N} c_{ij} y_{ij} + \sum_{j\in N} f_j x_j.$$

Uncapacitated Lot-Sizing (ULS)

The problem is to decide on a production plan for a single product over an n-period horizon. The basic model can be viewed as having data:

f_t is the fixed cost of producing in period t.
p_t is the unit production cost in period t.
h_t is the unit storage cost in period t.
d_t is the demand in period t.

We use the natural (or obvious) *variables*:

y_t is the amount produced in period t.
s_t is the stock at the end of period t.
$x_t = 1$ if production occurs in t, and $x_t = 0$ otherwise.

To handle the fixed costs, we observe that a priori no upper bound is given on y_t. Thus, we either must use a very large value $C = M$, or calculate an upper bound based on the problem data.

For *constraints* and *objective*, we obtain:

$$\begin{array}{rcl} \min \sum_{t=1}^{n} p_t y_t + \sum_{t=1}^{n} h_t s_t & + & \sum_{t=1}^{n} f_t x_t \\ s_{t-1} + y_t & = & d_t + s_t \quad \text{for } t = 1, \ldots, n \\ y_t & \leq & M x_t \quad \text{for } t = 1, \ldots, n \\ s_0 = 0, s_t, y_t & \geq & 0, x_t \in \{0,1\} \quad \text{for } t = 1, \ldots, n. \end{array}$$

If we impose that $s_n = 0$, then we can tighten the variable upper bound constraints to $y_t \leq \left(\sum_{i=t}^{n} d_i\right) x_t$. Note also that by substituting $s_t = \sum_{i=1}^{t} y_i - \sum_{i=1}^{t} d_i$, the objective function can be rewritten as $\sum_{t=1}^{n} c_t y_t + \sum_{t=1}^{n} f_t x_t - K$ where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i\right)$.

Discrete Alternatives or Disjunctions

Suppose $y \in \mathbb{R}^n$ satisfies $0 \leq y \leq u$, and either $a^1 y \leq b_1$ or $a^2 y \leq b_2$ (see Figure 1.4 in which the feasible region is shaded). We introduce binary variables x_i for $i = 1, 2$. Then if $M \geq \max\{a^i y - b_i : 0 \leq y \leq u\}$ for $i = 1, 2$, we take as constraints:

$$\begin{aligned} a^i y - b_i \leq M(1 - x_i) \quad & \text{for } i = 1, 2 \\ x_1 + x_2 = 1, x_i \in \{0,1\} \quad & \text{for } i = 1, 2 \\ 0 \leq y \leq u. & \end{aligned}$$

Now if $x_1 = 1, y$ satisfies $a^1 y \leq b_1$, whereas $a^2 x \leq b_2$ is inactive, and conversely if $x_2 = 1$.

Such disjunctions arise naturally in scheduling problems. Suppose that two jobs must be processed on the same machine and cannot be processed simultaneously. If p_i are the processing times and the variables t_i the start times for $i = 1, 2$, then either job 1 precedes job 2 and so $t_2 \geq t_1 + p_1$, or job 2 comes first and $t_1 \geq t_2 + p_2$.

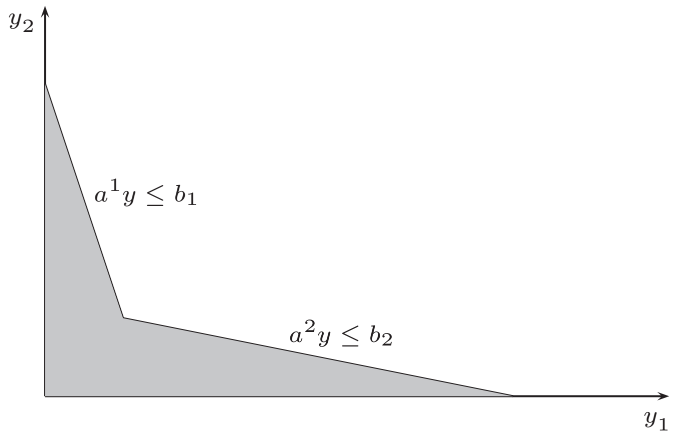


Figure 1.4 Either/or constraints.

1.6 Alternative Formulations

In Sections 1.3 and 1.5, we have formulated a small number of integer programs for which it is not too difficult to verify that the formulations are correct. Here and in Section 1.7, we examine alternative formulations and try to understand why some might be better than others. First, we make precise what we mean by a formulation.

Definition 1.1 A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ is a *polyhedron*.

Definition 1.2 A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a *formulation* for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X = P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

Note that this definition is restrictive. For example we saw in modeling fixed costs in Section 1.5 that we did not model the set $X = \{(0, 0), (y, 1) \text{ for } 0 < y \leq C\}$, but the set $X \cup \{(0, 1)\}$.

Example 1.2 In Figure 1.5, we show two different formulations for the set:

$$X = \{(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (2, 3)\}. \qquad \square$$

Note that it would not be easy to write a formulation for the set $X \setminus \{(2, 2)\}$ without introducing many more variables, see Exercise 2.

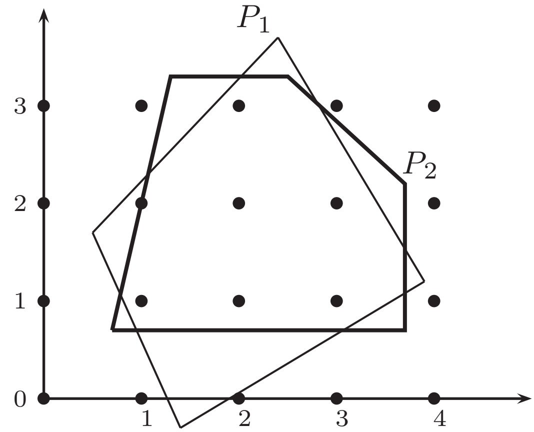


Figure 1.5 Two different formulations of an IP.

Equivalent Formulations for a 0–1 Knapsack Set

Consider the set of points $X =$

$$\{(0,0,0,0),(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(0,1,0,1),(0,0,1,1)\}.$$

Check that the three polyhedra below are formulations for X.

$$P_1 = \{x \in \mathbb{R}^4 \, : \, 0 \leq x \leq \mathbf{1}, 83x_1 + 61x_2 + 49x_3 + 20x_4 \leq 100\},$$
$$P_2 = \{x \in \mathbb{R}^4 \, : \, 0 \leq x \leq \mathbf{1}, 4x_1 + 3x_2 + 2x_3 + 1x_4 \leq 4\},$$

$$P_3 = \{x \in \mathbb{R}^4 \, : \begin{array}{lllll} 1x_1 & +1x_2 & +1x_3 & & \leq 1 \\ 1x_1 & & & +1x_4 & \leq 1 \\ & & x & \geq 0 & \end{array} \}.$$

An Equivalent Formulation for UFL

For fixed j, consider the constraints:

$$\sum_{i \in M} y_{ij} \leq m x_j, \quad x_j \in \{0,1\}, 0 \leq y_{ij} \leq 1 \text{ for } i \in M.$$

Logically, these constraints express the condition: if any $y_{ij} > 0$ for one or more $i \in M$, then $x_j = 1$, or stated a little differently: for each i, if $y_{ij} > 0$, then $x_j = 1$. This immediately suggests a different set of constraints:

$$0 \leq y_{ij} \leq x_j \quad \text{for } i \in M, x_j \in \{0,1\}.$$

This leads to the alternative formulation:

$$
\begin{aligned}
\min \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}y_{ij} + \sum_{j=1}^{n} f_j x_j & & & \\
\sum_{j=1}^{n} y_{ij} &= 1 & & \text{for } i \in M \\
y_{ij} - x_j &\le 0 & & \text{for } i \in M, j \in N \\
y_{ij} \ge 0 \text{ for } i \in M, j \in N, & & & x_j \in \{0,1\} \text{for } j \in N.
\end{aligned}
$$

In the two examples, we have just examined, the variables have been the same in each formulation, and we have essentially modified or added constraints. Another possible approach is to add or choose different variables, in which case we talk of *extended formulations*.

An Extended Formulation for ULS

What variables, other than production and stock levels, might be useful in describing an optimal solution of the lot-sizing problem? Suppose we would like to know when the items being produced now will actually be used to satisfy demand. Following the same steps as before, this leads to:

Definition of the variables.

w_{it} is the amount produced in period i to satisfy demand in period t

$x_t = 1$ if production occurs in period t (as above), and $x_t = 0$ otherwise.

Definition of the constraints.

Satisfaction of demand in period t:

$$\sum_{i=1}^{t} w_{it} = d_t \quad \text{for all } t.$$

Variable upper-bound constraints:

$$w_{it} \le d_t x_i \quad \text{for all } i, t, i \le t.$$

The variables are mixed:

$$w_{it} \ge 0 \text{ for all } i, t, i \le t, x_t \in \{0,1\} \text{for all } t.$$

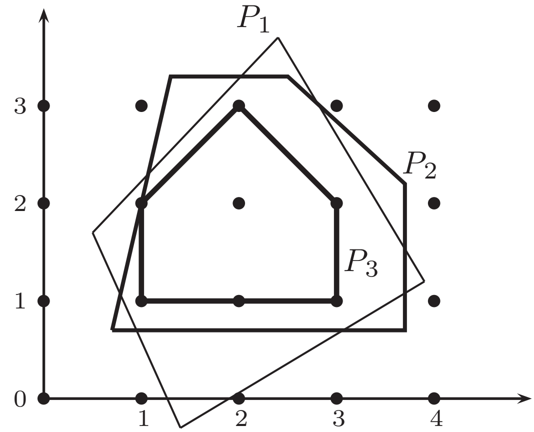
Definition of the objective function.

$$\min \sum_{i=1}^{n}\sum_{t=i}^{n} c_i w_{it} + \sum_{t=1}^{n} f_t x_t.$$

Note that here we are again using modified costs with c_t the production cost in period t and zero storage costs. It is optional whether we choose to define the old variables in terms of the new by adding defining equalities:

$$y_i = \sum_{t=i}^{n} w_{it}.$$

Figure 1.6 The ideal formulation.



1.7 Good and Ideal Formulations

In Figure 1.5, we show two different formulations of the same problem. We have also seen two possible formulations for the uncapacitated facility location problem. Geometrically, we can see that there must be an infinite number of formulations, so how can we choose between them?

The geometry again helps us to find an answer. Look at Figure 1.6 in which we have repeated the two formulations shown in Figure 1.5 and added a third one P_3. Formulation P_3 is *ideal*, because now if we solve a linear program over P_3, the optimal solution is at a vertex (extreme point). In this ideal case, each vertex is integer and so the IP is solved.

We can now formalize this idea.

Definition 1.3 Given a set $X \subseteq \mathbb{R}^n$, the *convex hull of* X, denoted conv(X), is defined as follows: $\text{conv}(X) = \{x : x = \sum_{i=1}^{t} \lambda_i x^i, \sum_{i=1}^{t} \lambda_i = 1, \lambda_i \geq 0$ for $i = 1, \dots, t$ over all finite subsets $\{x^1, \dots, x^t\}$ of $X\}$.

Proposition 1.1 *If* $X = \{x \in \mathbb{Z}^n : Ax \leq b\}$, *conv*($X$) *is a polyhedron.*

Definition 1.4 Given a polyhedron P, x is an *extreme point* of P if $x^1, x^2 \in P$ with $x = \lambda x^1 + (1 - \lambda)x^2$, $0 < \lambda < 1$ implies that $x = x^1 = x^2$. Note that extreme points correspond to the geometric idea of vertices.

Proposition 1.2 *When a linear program:* $\max\{cx : x \in PX\}$ *where* P *is a polyhedron has finite optimal value, there exists an extreme point of* P *that is optimal.*

Because of these two results, we can replace the IP: $\{\max cx : x \in X\}$ by the equivalent linear program: $\max\{cx : x \in \text{conv}(X)\}$. This ideal reduction to a linear program holds for unbounded integer sets $X = \{x : Ax \leq b \text{ and integer}\}$, and also mixed integer sets $X = \{(x, y) : Ax + Gy \leq b, x \geq 0 \text{ and integer}, y \geq 0\}$ with A, G, b rational. However, whether X is bounded or not, this is in general only a theoretical solution, because in most cases, there is an enormous (exponential) number of inequalities needed to describe conv(X), and there is no simple characterization of all these inequalities.

So we might rather ask the question: Given two formulations P_1 and P_2 for X, when can we say that one is *better* than the other? Because the ideal solution conv(X) has the property that $X \subseteq \text{conv}(X) \subseteq P$ for all formulations P, this suggests the following definition.

Definition 1.5 Given a set $X \subseteq \mathbb{Z}^n$, and two formulations P_1 and P_2 for X, P_1 is a *better formulation* than P_2 if $P_1 \subset P_2$.

In Chapter 2, we will see that this turns out to be a useful definition and a very important idea for the development of effective formulations. For the moment, we examine some of our formulations in the light of these definitions.

Equivalent Formulations for a Knapsack Set

Looking again at the set $X =$

$$\{(0,0,0,0), (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1), (0,1,0,1), (0,0,1,1)\}$$

and the three formulations

$$P_1 = \{x \in \mathbb{R}^4 : 0 \leq x \leq 1, 83x_1 + 61x_2 + 49x_3 + 20x_4 \leq 100\},$$
$$P_2 = \{x \in R^4 : 0 \leq x \leq 1, 4x_1 + 3x_2 + 2x_3 + 1x_4 \leq 4\},$$
$$\begin{array}{lllllll} P_3 = \{x \in \mathbb{R}^4 : & 1x_1 & +1x_2 & +1x_3 & & \leq 1 \\ & 1x_1 & & & +1x_4 & \leq 1 \\ & & & & x \geq 0 & \}, \end{array}$$

it is easily seen that $P_3 \subset P_2 \subset P_1$. It can be checked that $P_3 = \text{conv}(X)$ and P_3 is thus an ideal formulation.

Formulations for UFL

Let P_1 be the formulation given in Section 1.5 with a single constraint

$$\sum_{i \in M} y_{ij} \leq m x_j$$

for each j, and P_2 be the formulation given in Section 1.5 with the m constraints

$$y_{ij} \leq x_j \quad \text{for } i \in M$$

for each j. Note that if a point (x, y) satisfies the constraints $y_{ij} \leq x_j$ for $i \in M$, then summing over $i \in M$ shows that it also satisfies the constraints $\sum_i y_{ij} \leq m x_j$. Thus $P_2 \subseteq P_1$. To show that $P_2 \subset P_1$, we need to find a point in P_1 that is not in P_2. Suppose for simplicity that n divides m, that is, $m = kn$ with $k \geq 2$ and integer. Then a point in which each depot serves k clients, $y_{ij} = 1$ for $i = k(j-1)+1, \dots, k(j-1)+k, j = 1, \dots, n$, $y_{ij} = 0$ otherwise, and $x_j = k/m$ for $j = 1, \dots, n$ lies in $P_1 \backslash P_2$.

The two formulations that we have seen for the lot-sizing problem (ULS) have different variables, so it is not immediately obvious how they can be compared. We now examine this question briefly.

Formally, assuming for simplicity that all the variables are integer, we have a first formulation

$$\min\{cx : x \in P \cap \mathbb{Z}^n\}$$

with $P \subseteq \mathbb{R}^n$, and a second extended formulation

$$\min\{cx : (x, w) \in Q \cap (\mathbb{Z}^n \times \mathbb{R}^p)\}$$

with $Q \subseteq \mathbb{R}^n \times \mathbb{R}^p$.

Definition 1.6 Given a polyhedron $Q \subseteq \mathbb{R}^n \times \mathbb{R}^p$, the *projection* of Q onto the subspace $\mathbb{R}^n$, denoted $\text{proj}_x Q$, is defined as follows:

$$\text{proj}_x Q = \{x \in \mathbb{R}^n : (x, w) \in Q \text{ for some } w \in \mathbb{R}^p\}.$$

Thus, $\text{proj}_x Q \subseteq \mathbb{R}^n$ is the formulation obtained from Q in the space $\mathbb{R}^n$ of the original x variables, and it allows us to compare Q with other formulations $P \subseteq \mathbb{R}^n$.

Comparing Formulations for ULS

We can now compare the formulation P_1:

$$\begin{aligned} s_{t-1} + y_t &= d_t + s_t \quad \text{for } t = 1, \dots, n \\ y_t &\leq M x_t \quad \text{for } t = 1, \dots, n \\ s_0 = s_n = 0, s_t, y_t &\geq 0, 0 \leq x_t \leq 1 \quad \text{for } t = 1, \dots, n \end{aligned}$$

and $P_2 = \text{proj}_{y,s,x} Q_2$, where Q_2 takes the form:

$$\begin{aligned}
s_{t-1} + y_t &= d_t + s_t \quad \text{for } t = 1, \dots, n \\
\sum_{i=1}^{t} w_{it} &= d_t \quad \text{for } t = 1, \dots, n \\
w_{it} &\leq d_t x_i \quad \text{for all } i, t, i \leq t \\
y_i &= \sum_{t=i}^{n} w_{it} \quad \text{for } i = 1, \dots, n \\
w_{it} &\geq 0 \quad \text{for all } i, t, \ i \leq t \\
0 &\leq x_t \leq 1 \quad \text{for all } t.
\end{aligned}$$

It can be shown that P_2 describes the convex hull of solutions to the problem and is thus an ideal formulation. It is easily seen that $P_2 \subset P_1$. For instance the point $y_t = d_t, x_t = d_t/M$ for all t is an extreme point of P_1 that is not in P_2.

1.8 Notes

Introduction

Much of the material in this book except for the later chapters on decomposition algorithms and heuristics are treated in greater detail and at a more advanced level in Nemhauser and Wolsey [233] and the excellent recent book of Conforti et al. [74]. An even more advanced theoretical treatment of integer programming is given in Schrijver [266]. Other books treating integer programming from a somewhat different angle include Martin [226] and Bertsimas and Weismantel [51]. The book of Jünger et al. [190] to celebrate 50 years of integer programming includes chapters of reminiscences from the founders of the field and up-to-date surveys of many of the major topics. For the related area of combinatorial optimization, there is the undergraduate text of Papadimitriou and Stieglitz [243] and the graduate level texts of Cook et al. [79] and of Korte and Vygen [198] and the ultimate 3-volume bible of Schrijver [267].

The journals publishing articles on integer programming cover a wide spectrum from theory to applications. The more theoretical include *Mathematical Programming, Mathematics of Operations Research, the SIAM Journal on Discrete Mathematics, the SIAM Journal on Optimization* and *Discrete Applied Mathematics*, while *Operations Research, the European Journal of Operations Research, Networks, Management Science, Transportation Science, OR Letters, Optimization Letters*, etc. contain more integer programming applications.

For those unfamiliar with linear programming, Dantzig [91] is the ultimate classic, and the book of Chvatal [71] is exceptional. Vanderbei [283] treats interior point as well as simplex methods, see also Wright [300]. For graph theory, the books of Berge [45] and Bondy and Murty [57] are classics. For network flows Ford and Fulkerson [125] and Lawler [204] remain remarkably stimulating, while Ahuja et al. [9] is clear and comprehensive and Williamson [292] is hot off the press.

In the notes at the end of each chapter, we will cite only selected references, a few of the important original sources, and also some, where possible, recent surveys and articles that can be used for further reading.

Notes for the Chapter

The importance of formulations in integer programming is well established. Certain special classes of problems that are often tackled as integer programs have had books dedicated to them: knapsack problems by Martello and Toth [224] and Kellerer et al. [195], traveling salesman problems by Applegate et al. [12], location problems by Mirchandani and Francis [230] and Laporte et al. [203], network models and network routing by Ball et al. [30, 31], vehicle routing by Golden et al. [152] and Toth and Vigo [276], and production planning by Pochet and Wolsey [250]. For machine scheduling problems, Queyranne and Schultz [254] contains a variety of integer programming formulations.

The documentation provided by some of the different mathematical programming systems, such as CPLEX [84], GUROBI [171], LINDO [210], SCIP [144, 268], and XPRESS [301], includes a wide range of formulations.

1.9 Exercises

1. Suppose that you are interested in choosing a set of investments $\{1, \ldots, 7\}$ using 0–1 variables. Model the following constraints:
   (i) You cannot invest in all of them.
   (ii) You must choose at least one of them.
   (iii) Investment 1 cannot be chosen if investment 3 is chosen.
   (iv) Investment 4 can be chosen only if investment 2 is also chosen.
   (v) You must choose either both investments 1 and 5 or neither.
   (vi) You must choose either at least one of the investments 1,2,3, or at least two investments from 2,4,5,6.

2. Formulate the following as mixed integer programs:
   (i) $u = \min\{y_1, y_2\}$, assuming that $0 \leq y_j \leq C$ for $j = 1, 2$
   (ii) $v = |y_1 - y_2|$ with $0 \leq y_j \leq C$, for $j = 1, 2$
   (iii) the set $X \setminus \{x^*\}$ where $X = \{x \in \{0, 1\}^n : Ax \leq b\}$ and $x^* \in X$.

3. Modeling disjunctions.

(i) Extend the formulation of discrete alternatives of Section 1.5 to the union of two bounded polyhedra (*polytopes*) $P_k = \{y \in \mathbb{R}^n : A^k y \leq b^k, 0 \leq y \leq u\}$ for $k = 1, 2$ where $\max_k \max_i \{a_i^k y - b_i^k : 0 \leq y \leq u\} \leq M$.

(ii) Show that an extended formulation for $P_1 \cup P_2$ is the set Q:

$$\begin{aligned} y &= w^1 + w^2 \\ A^k w^k &\leq b^k x^k \quad \text{for } k = 1, 2 \\ 0 \leq w^k &\leq u x^k \quad \text{for } k = 1, 2 \\ x^1 + x^2 &= 1 \\ y \in \mathbb{R}^n, w^k \in \mathbb{R}^n, x^k &\in \{0, 1\} \quad \text{for } k = 1, 2. \end{aligned}$$

4. Show that

$$\begin{aligned} X &= \{x \in \{0,1\}^4 : 97x_1 + 32x_2 + 25x_3 + 20x_4 \leq 139\} \\ &= \{x \in \{0,1\}^4 : 2x_1 + x_2 + x_3 + x_4 \leq 3\} \\ &= \{x \in \{0,1\}^4 : x_1 + x_2 + x_3 \leq 2 \\ &\quad x_1 + x_2 + x_4 \leq 2 \\ &\quad x_1 + x_3 + x_4 \leq 2\}. \end{aligned}$$

5. John Dupont is attending a summer school where he must take four courses per day. Each course lasts an hour, but because of the large number of students, each course is repeated several times per day by different teachers. Section i of course k denoted (i, k) meets at the hour t_{ik}, where courses start on the hour between 10 a.m. and 7 p.m. John's preferences for when he takes courses are influenced by the reputation of the teacher, and also the time of day. Let p_{ik} be his preference for section (i, k). Unfortunately, due to conflicts, John cannot always choose the sections he prefers.

(i) Formulate an integer program to choose a feasible course schedule that maximizes the sum of John's preferences.

(ii) Modify the formulation so that John never has more than two consecutive hours of classes without a break.

(iii) Modify the formulation so that John chooses a schedule in which he starts his day as late as possible.

6. Prove that the set of feasible solutions to the formulation of the traveling salesman problem in Section 1.3 is precisely the set of incidence vectors of tours.

7. The QED Company must draw up a production program for the next nine weeks. Jobs last several weeks and once started must be carried out without interruption. During each week, a certain number of skilled workers are required to work full-time on the job. Thus, if job i lasts p_i weeks, $l_{i,u}$ workers are required in week u for $u = 1, \dots, p_i$. The total number of workers available in week t is L_t. Typical job data $(i, p_i, l_{i1}, \dots, l_{ip_i})$ is shown below.

Job	Length	Week1	Week2	Week3	Week4
1	3	2	3	1	—
2	2	4	5	—	—
3	4	2	4	1	5
4	4	3	4	2	2
5	3	9	2	3	—

(i) Formulate the problem of finding a feasible schedule as an IP.
(ii) Formulate when the objective is to minimize the maximum number of workers used during any of the nine weeks.
(iii) Job 1 must start at least two weeks before job 3. Formulate.
(iv) Job 4 must start not later than one week after job 5. Formulate.
(v) Jobs 1 and 2 both use the same machine and cannot be carried out simultaneously. Formulate.

8. Show that the uncapacitated facility location problem of Section 1.5 with a set $N = \{1, \dots, n\}$ of depots can be written as the COP

$$\min_{S \subseteq N} \left\{ c(S) + \sum_{j \in S} f_j \right\}$$

where $c(S) = \sum_{i=1}^{m} \min_{j \in S} c_{ij}$.

9. Show that the set covering problem of Section 1.3 can be written as the COP

$$\min_{S \subseteq N} \left\{ \sum_{j \in S} f_j \; : \; v(S) = v(N) \right\},$$

where $v(S) = \sum_{i=1}^{m} \min\{\sum_{j \in S} a_{ij}, 1\}$.

10. A set of n jobs must be carried out on a single machine that can do only one job at a time. Each job j takes p_j hours to complete. Given job weights w_j for $j = 1, \dots, n$, in what order should the jobs be carried out so as to minimize the weighted sum of their start times? Formulate this scheduling problem as a mixed integer program.

11. Given a graph $G = (V, E)$ with edge weights $c \in \mathbb{R}^{|E|}$ and a subset $X \subseteq V$, the set of edges $E(X) = \{e = (i,j) \in E : |X \cap \{i,j\}| = 1\}$ is a *cut*. Formulate the problem of finding a maximum weight cut as
(i) a quadratic 0–1 integer program,
(ii) a linear integer program.

12. Using a mixed integer programming system, solve an instance of the uncapacitated facility location problem, where f_j is the cost of opening depot j, and c_{ij} is the cost of satisfying all client i's demand from depot j, with $f = (4, 3, 4, 4, 7)$ and

$$(c_{ij}) = \begin{pmatrix} 12 & 13 & 6 & 0 & 1 \\ 8 & 4 & 9 & 1 & 2 \\ 2 & 6 & 6 & 0 & 1 \\ 3 & 5 & 2 & 1 & 8 \\ 8 & 0 & 5 & 10 & 8 \\ 2 & 0 & 3 & 4 & 1 \end{pmatrix}.$$

13. The symmetric traveling salesman problem is a TSP in which $c_{ij} = c_{ji}$ for all $(i,j) \in A$. Consider an instance on the graph shown in Figure 1.7, where the missing edges have a very high cost. Solve with a mixed integer programming system.

14. Formulate and solve an instance of the lot-sizing problem over six periods, with demands $(6, 7, 4, 6, 3, 8)$, unit production costs $(3, 4, 3, 4, 4, 5)$, unit storage costs $(1, 1, 1, 1, 1, 1)$, set-up costs $(12, 15, 30, 23, 19, 45)$, and a maximum production capacity of 10 items per period.

15. Formulate the problem of placing N queens on an N by N chessboard such that no two queens share any row, column, or diagonal. Solve for $N = 10$.

16. Frequency Assignment. Frequencies from the range $\{1, \dots, 6\}$ must be assigned to 10 stations. For each pair of stations, there is an interference

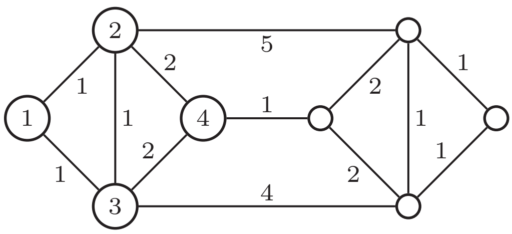


Figure 1.7 TSP Instance.

parameter which is the minimum amount by which the frequencies of the two stations must differ. The pairs with nonzero parameter are given below:

$$\begin{array}{lcccccccc} e = & (5,7) & (2,3) & (3,4) & (4,5) & (5,6) & (6,7) & (7,8) & (8,9) \\ & 3 & 2 & 4 & 2 & 3 & 1 & 1 & 2 \\ e = & (9,10) & (1,8) & (2,10) & (3,10) & (5,10) & (7,10) & (2,5) & (5,9) \\ & 3 & 2 & 1 & 1 & 4 & 2 & 2 & 2 \end{array}$$

The goal is to minimize the difference between the smallest and largest frequency assigned. Formulate and solve.