Chapter 5

Mixed Linear–Integer Programming Models for Combinatorial Optimization Problems

5.1. Introduction

A combinatorial optimization problem can be typically formulated as a mixed integer linear programming (MILP) model by first translating each elementary decision into a variable and by then specifying the objective function and the related constraints. To obtain a *linear* programming model, the objective is also required to be a *linear* function of the variables and the constraints are required to be *linear (in)equalities* involving the variables. While modeling can sometimes be considered to be an "art" which cannot be reduced to a standard set of procedures, it is of prime interest to devise at least some general rules to help with the formulation process. Note that, even though nearly all books on operations research and combinatorial optimization deal broadfly with MILP models, only a few references (see for instance [PLA02, WIL90]) discuss some sort of systematic approach in the MILP modeling of combinatorial optimization problems. The purpose of this chapter is to provide an insight into MILP modeling of combinatorial optimization problems. First, introductory MILP models are recalled together with general modeling techniques; then more or less standard MILP formulations of several combinatorial optimization problems are discussed.

5.1.1. *Preliminaries*

A general MILP model is composed of three fundamental elements:

Chapter written by Federico DELLA CROCE.

– Decision variables ($x_1, x_2, ..., x_j, ..., x_n$)

the unknowns of the problem, namely those entities such that, *given any (feasible) solution of the problem, if we know their value, then we can immediately verify if the solution is feasible and compute the cost function value.* Identification of the decision variables is the primary issue in ILP modeling and the determination of a solution for an ILP model consists of assigning a value to the decision variables.

– Constraints

the mathematical relations that must express the requirements of the considered problem. When the variables set is properly defined, then all the requirements can be naturally formulated by means of equalities or inequalities on the given variables.

– Objective function

the function f of the decision variables x_1, x_2, ..., x_j, ..., x_n to be minimized (maximized).

As we are dealing with MILP models, both the objective function and the constraints are restricted to *linear* expressions of the variables.

The general formulation of a MILP model is:

$$\begin{gathered}
\min\,(\max)\ f(x_1,...,x_n) = c_1x_1 + c_2x_2 + ... + c_jx_j + ... + c_nx_n \\
a_{11}x_1 + a_{12}x_2 + ... + a_{1j}x_j + ... + a_{1n}x_n \gtreqless b_1 \\
..... \\
a_{i1}x_1 + a_{i2}x_2 + ... + a_{ij}x_j + ... + a_{in}x_n \gtreqless b_i \\
..... \\
a_{m1}x_1 + a_{m2}x_2 + ... + a_{mj}x_j + ... + a_{mn}x_n \gtreqless b_m \\
x_j \geqslant 0, \quad j = 1, ..., n
\end{gathered} \tag{5.1}$$

with $x_1, ..., x_j, ..., x_n$, decision variables, $c_1, ..., c_j, ..., c_n$, cost (profit) coefficients in the objective function for a minimization (maximization) problem, $b_1, ..., b_i, ..., b_m$, so-called right-end sides in the constraints set, $a_{11}, ..., a_{ij}, ..., a_{mn}$, variables coefficients in the constraints set.

The decision variables are typically split into three main categories:

1) positive real variables ($x_j \geqslant 0$);

2) positive integer variables($x_j \geqslant 0$, integer);

3) binary variables ($x_j \in \{0, 1\}$).

Note that we can immediately handle problems where a given variable x_j is free ($-\infty \leqslant x_j \leqslant +\infty$) or non-positive ($x_j \leqslant 0$). To do this, if x_j is free, it is sufficient to set $x_j = u_j - v_j$ with $u_j \geqslant 0$, $v_j \geqslant 0$ and substitute x_j with $u_j - v_j$ correspondingly. Analogously, if $x_j \leqslant 0$, it is sufficient to set $x_j = -y_j$ with $y_j \geqslant 0$ and substitute x_j with $-y_j$ correspondingly.

In the following we provide a miscellany of basic MILP formulations of well-known combinatorial optimization problems.

5.1.2. *The knapsack problem*

In the knapsack problem, a hiker must decide which items to include in his/her knapsack on a forthcoming trip. He/she must choose from among a set N of n items, where item i has weight w_i (say in kilograms) and utility u_i. The objective is to maximize the hiker's trip utility subject to the limitation that he/she can carry at most W kg. Let x_i be a $0-1$ variable such that $x_i = 1$ if item i is included in the knapsack, $x_i = 0$ if it is not.

The standard MILP formulation of this model is as follows:

$$\max \sum_{i \in N} u_i x_i \tag{5.2}$$

subject to

$$\sum_{i \in N} w_i x_i \leqslant W \tag{5.3}$$

$$x_i \in \{0, 1\} \quad \forall i \in N \tag{5.4}$$

where the objective function [5.2] maximizes the hiker's trip utility, constraint [5.3] forbids exceeding the weight capacity W, and constraints [5.4] impose the restriction that the x_i variables must be binary.

5.1.3. *The bin-packing problem*

In the bin-packing problem, there is a given set N of n items with sizes $\{s_1, ..., s_n\}$ to be packed into bins such that the sum of the sizes of all items assigned to each bin does not exceed the bin capacity, W (where $s_j \leqslant W \ \ \forall j \in N$, or else the problem is not feasible), and the objective is to use as few bins as possible. Let u_i ($i = 1, ..., n$) be a $0 - 1$ variable such that $u_i = 1$ if bin i is used, $u_i = 0$ if it is not. Let x_{ij} ($i = 1, ..., n$, $j \in N$) be a $0 - 1$ variable such that $x_{ij} = 1$ if item j is assigned to bin i, $x_{ij} = 0$ if it is not.

The standard MILP formulation of this model is as follows:

$$\min \sum_{i=1}^{n} u_i \tag{5.5}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \in N \tag{5.6}$$

$$\sum_{j \in N} s_j x_{ij} \leqslant W u_i \quad \forall i = 1, ..., n \tag{5.7}$$

$$u_i, x_{ij} \in \{0, 1\} \quad \forall i = 1, ..., n, \ \ \forall j \in N \tag{5.8}$$

where the objective function [5.5] minimizes the number of bins used, constraints [5.6] impose the restriction that each item is assigned to exactly one bin, constraints [5.7] impose the restriction that the weight capacity W for each bin is not excedded and constraints [5.8] impose the restriction that the variables u_i and x_{ij} must be binary.

5.1.4. *The set covering/set partitioning problem*

Consider a $0/1$ $m \times n$ matrix $A = (a_{ij})$. Let $M = (1, ..., m)$ be the set of rows of A and correspondingly let $N = (1, ..., n)$ be the set of columns of A. Let c_j ($j = 1, ..., n$) denote the cost of column j, where we assume $c_j > 0, \forall j$. We say that a column $j \in N$ covers a row $i \in M$ if $a_{ij} = 1$. In the set covering problem we search for a minimum cost subset $S \subseteq N$ of columns such that each row $i \in M$ is covered by at least one column $j \in N$. We are dealing with a set partitioning problem if the

requirement is that each row $i \in M$ is covered by exactly one column $j \in N$. Let x_j be a $0-1$ variable such that $x_j = 1$ if column j is selected ($j \in S$), $x_j = 0$ if it is not.

The standard MILP formulation of the set covering problem is as follows:

$$\min \sum_{j \in N} c_j x_j \tag{5.9}$$

subject to

$$\sum_{j \in N} a_{ij} x_j \geqslant 1 \quad \forall i \in M \tag{5.10}$$

$$x_j \in \{0, 1\} \quad \forall j \in N \tag{5.11}$$

where the objective function [5.9] minimizes the weighted sum of selected columns, constraints [5.10] impose the restriction that each row is covered by at least one column and constraints [5.11] impose the restriction that the x_j variables must be binary.

To deal with the set partitioning problem, it is sufficient to substitute inequalities (5.10) with equalities as follows:

$$\sum_{j \in N} a_{ij} x_j = 1 \quad \forall i \in M. \tag{5.12}$$

5.1.5. *The minimum cost flow problem*

In the minimum cost flow problem we want to determine a least cost shipment of a commodity through a network in order to satisfy demands at certain nodes from available supplies at other nodes.

Consider a directed network $G(N, A)$ defined by a set N of n nodes and a set A of m directed arcs. Each arc (i, j) is associated with a cost c_{ij} denoting the cost per unit flow on that arc where the flow cost is assumed to vary linearly with the amount of flow. Each node $i \in N$ is associated with an integer value b_i representing its supply/demand. If $b_i > 0$, i is a supply node; if $b_i < 0$, i is a demand node; finally, if $b_i = 0$, i is a transhipment node. We assume that the network is balanced, that is $\sum_{i \in N} b_i = 0$.

Let $x_{ij} \geqslant 0$ denote the flow on arc $(i,j) \in A$. A standard MILP formulation of the minimum cost flow problem is as follows:

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{5.13}$$

subject to

$$\sum_{k:(j,k)\in A} x_{jk} - \sum_{i:(i,j)\in A} x_{ij} = b_j \quad \forall j \in N \tag{5.14}$$

$$x_{ij} \geqslant 0 \quad \forall (i,j) \in A \tag{5.15}$$

where the objective function [5.13] minimizes the total flow cost, constraints [5.14], the so-called *flow balance constraints*, impose the restriction that, for each node, the flow emanating from the node minus the flow entering the node must be equal to the supply/demand of that node, and constraints [5.15] indicate that the variables x_{ij} are positive.

Note that, as the constraints coefficients matrix is totally unimodular, if b_j are integers then an optimal solution of the problem in which all x_{ij}s are integers exists.

5.1.6. *The maximum flow problem*

In the maximum flow problem we still consider a directed network $G(N,A)$ defined by a set N of n nodes and a set A of m directed arcs. We want to determine the maximum amount of flow from a specified source node s to another specified sink node t. No cost is associated with the flow, but each arc $(i,j) \in A$ is associated with a capacity u_{ij} that denotes the maximum amount that can flow on the arc. Let $x_{ij} \geqslant 0$ denote, as before, the flow on arc $(i,j) \in A$ and let v denote the flow sent from s to t. A standard MILP formulation of the maximum flow problem is as follows:

$$\max v \tag{5.16}$$

subject to

$$\sum_{k:(j,k)\in A} x_{jk} - \sum_{i:(i,j)\in A} x_{ij} = \begin{cases} v, & j = s \\ 0, & j \neq s,t \\ -v, & j = t \end{cases} \tag{5.17}$$

$$0 \leqslant x_{ij} \leqslant u_{ij} \quad \forall (i,j) \in A \tag{5.18}$$

where the objective function [5.16] maximizes the flow v sent from s to t (with $v = \sum_{k:(s,k)\in A} x_{sk} = \sum_{i:(i,t)\in A} x_{it}$), constraints [5.17] represent the *flow balance constraints* and constraints [5.18] indicate that the flow on arc $(i,j) \in A$ is positive and not greater than the arc capacity u_{ij}.
Note that, as for the minimum cost flow problem, the constraints coefficients matrix is totally unimodular, so, if u_{ij} are integers then an optimal solution of the problem in which all x_{ij}s and v are integers exists.

5.1.7. *The transportation problem*

The transportation problem is a special case of the minimum cost flow problem where the node set N is partitioned into two subsets N_1 and N_2; each node $i \in N_1$ is a supply node providing a positive amount $a_i > 0$ of goods to be shipped and each node $j \in N_2$ is a demand node requiring a positive amount $b_j > 0$ of the same goods. Correspondingly, for each arc $(i,j) \in A$, we have $i \in N_1$ and $j \in N_2$. Also, we still assume that the network is balanced, that is $\sum_{i\in N_1} a_i = \sum_{j\in N_2} b_j$.

By denoting with $x_{ij} \geqslant 0$ the amount of goods sent on arc $(i,j) \in A$ from node $i \in N_1$ to node $j \in N_2$, we get the following standard MILP formulation of the transportation problem:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{5.19}$$

subject to

$$\sum_{j\in N_2} x_{ij} = a_i \quad \forall i \in N_1 \tag{5.20}$$

$$\sum_{i\in N_1} x_{ij} = b_j \quad \forall j \in N_2 \tag{5.21}$$

$$x_{ij} \geqslant 0 \quad \forall (i,j) \in A \tag{5.22}$$

where the objective function [5.19] minimizes the total transportation cost, constraints [5.20] impose the restriction that for each node $i \in N_1$ the total amount of shipped

goods is equal to a_i, constraints [5.21] impose the restriction that for each node $j \in N_2$ the total amount of received goods is equal to b_j, and constraints [5.22] indicate that the variables x_{ij} are positive.

Note that, as the transportation problem is a special case of the minimum cost flow problem, if all a_is and b_js are integers, then an optimal solution of the problem in which all x_{ij}s are integers exists.

5.1.8. *The assignment problem*

In the assignment problem, two equally sized sets N_1 and N_2 (with $|N_1| = |N_2|$) and a collection of pairs $A \subseteq N_1 \times N_2$ representing possible assignments are given. Each pair $(i, j) \in A$ with $i \in N_1$ and $j \in N_2$ is associated with a cost c_{ij}. In the assignment problem we search for a minimum cost assignment where each element $i \in N_1$ is assigned to exactly one element $j \in N_2$. Let x_{ij} be a $0 - 1$ variable such that $x_{ij} = 1$ if element $i \in N_1$ is assigned to element $j \in N_2$, $x_{ij} = 0$ if it is not.

The standard formulation of the assignment problem is as follows:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{5.23}$$

subject to

$$\sum_{j\in N_2} x_{ij} = 1 \quad \forall i \in N_1 \tag{5.24}$$

$$\sum_{i\in N_1} x_{ij} = 1 \quad \forall j \in N_2 \tag{5.25}$$

$$x_{ij} \in \{0, 1\} \quad \forall (i, j) \in A \tag{5.26}$$

where the objective function [5.23] minimizes the total assignment cost, constraints [5.24] impose the restriction that each element $i \in N_1$ is assigned to exactly one element $j \in N_2$, constraints [5.25] impose the restriction that each element $j \in N_2$ is assigned to exactly one element $i \in N_1$, and constraints [5.26] impose the restriction that the variables x_{ij} must be binary.

Note that the assignment problem can be viewed as a special case of the transportation problem with $|N_1| = |N_2|$ and $a_i = b_j = 1,\ \forall i \in N_1,\ \forall j \in N_2$. Hence,

the integrality constraints of the variables can be dropped. Furthermore, as constraints [5.24] imply that all x_{ij} variables are $\leqslant 1$, (5.26) can be substituted by constraints:

$$x_{ij} \geqslant 0 \quad \forall (i,j) \in A \tag{5.27}$$

5.1.9. *The shortest path problem*

In the shortest path problem, we search for the minimum distance path between two nodes s and t of a given network. This problem can be modeled as a special case of the minimum cost flow problem where node s is a supply node with unitary supply ($b_s = 1$), node t is a demand node with unitary demand ($b_t = -1$), and all other nodes are transhipment nodes ($b_k = 0,\ \forall k \in N,\ k \neq s,t$). The model is then:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{5.28}$$

subject to

$$\sum_{k:(j,k)\in A} x_{jk} - \sum_{i:(i,j)\in A} x_{ij} = \begin{cases} 1, & j = s \\ 0, & j \neq s,t \\ -1, & j = t \end{cases} \tag{5.29}$$

$$x_{ij} \geqslant 0 \quad \forall (i,j) \in A \tag{5.30}$$

Indeed, as all b_is are integers, the solution to this special minimum cost flow problem is an integer and will send exactly one unit of flow from node s to node t along the shortest path.

5.2. General modeling techniques

In the following we introduce several modeling tricks that enable generation of linear models in presence of special types of non-linear expressions or in the presence of logic conditions between real/integer variables and/or between binary variables.

5.2.1. *Min/Max, Max/Min, Min/Abs models*

Consider a mathematical programming model where the constraints are expressed by linear expressions of the decision variables, but the objective function is as follows:

$$\min \max\{e_1, e_2, ..., e_n\}$$

where $e_1, e_2, ..., e_n$ are linear expressions of the variables. We have a so-called *min-max model* that is non-linear but can be transformed into an equivalent linear model. We need to introduce a fictitious variable y in the objective function to be minimized. We then have as the objective function:

$$\min y.$$

Furthermore, we need to add to the constraints set the following inequalities:

$$y \geqslant e_1,\ y \geqslant e_2,\ ...,\ y \geqslant e_n.$$

The generated model is linear and is equivalent to the corresponding original model: indeed, the value of y corresponds exactly to the value $\max\{e_1, e_2, ..., e_n\}$, as it is constrained by the introduced inequalities not to be less than the largest of the e_i terms and is minimized by the objective function. Hence, this implies that in the optimal solution at least one of the constraints $y \geqslant e_i$ is saturated, namely $\exists i : y = e_i$.

Analogously, for a problem of the type:

$$\max \min\{e_1, e_2, ..., e_n\}$$

where $e_1, e_2, ..., e_n$ are linear expressions of the variables, we can generate an equivalent linear model. In this case, the objective function (having introduced a fictitious variable y) becomes:

$$\max y$$

and the original constraints set must be integrated with the following inequalities:

$$y \leqslant e_1,\ y \leqslant e_2,\ ...,\ y \leqslant e_n.$$

Finally, consider a so-called *min-abs model* with an objective function of the type:

$$\min |e|$$

where e is a linear expression of the variables. Here we want to minimize the absolute value (which is non-linear) of a linear expression. As $|e| = \max\{e, -e\}$, the *min-abs model* can be written as a *min-max model* and correspondingly transformed into a linear model. Notice that the same can also be done for expressions of the type $\min |e_1| + |e_2| + ... + |e_n|$ provided that $e_1, ..., e_n$ are linear expressions of the decision variables. However, it is not possible to derive equivalent linear formulations in the presence of objective functions such as $\min\{\min\{e_1, e_2, ..., e_n\}\}$, $\max\{\max\{e_1, e_2, ..., e_n\}\}$, or $\max\{|e_1| + |e_2| + ... + |e_n|\}$.

5.2.2. *Handling logic conditions*

In many cases the real/integer decision variables of the problem are subject to further logic conditions. These relations can be either specific requirements on the value of a single variable or may provide logic conditions on two or more variables. It is possible to model these relations by introducing fictitious binary variables, linking these variables to the original decision variables, and finally adding linear constraints on the binary variables. Some examples of this technique are given below.

5.2.2.1. *One-to-many conditions*

Assume that a given positive variable x_j is restricted to having only k values $p_1,\ p_2, ...,\ p_k$. This condition cannot be expressed linearly if only variable x_j is considered. Besides, by introducing k binary variables $y_1,\ y_2, ...,\ y_k$, the condition can be expressed by means of the equations:

$$\sum_{i=1}^{k} y_i = 1 \tag{5.31}$$

$$x_j = \sum_{i=1}^{k} p_i y_i \tag{5.32}$$

where equation [5.31] imposes the restriction that only one of the variables y_i is equal to 1 and, correspondingly, equation [5.32] imposes the restriction that variable x_j can have only one of the values $p_1,\ p_2, ...,\ p_k$.

5.2.2.2. *Fixed charge constraints*

Consider the so-called ***fixed charge*** case. Here we have a given variable x_j which cannot exceed a given threshold value M. This variable (typically corresponding to an activity) is associated in the objective function not only with a unitary cost coefficient c_j but also with a ***fixed charge*** f_j that must be considered only if x_j has the value > 0. To model this requirement, it is sufficient to introduce a fictitious binary variable y_j so that the global cost of the activity becomes:

$$c_j x_j + f_j y_j \tag{5.33}$$

and add the constraint:

$$x_j \leqslant M y_j. \tag{5.34}$$

Note that, as variable y_j can only have values $\{0, 1\}$, then either we have $x_j = y_j = 0$ or $y_j = 1$, which only implies that x_j is limited so as not to exceed M.

5.2.2.3. *Big-M constraints formulation*

The linear formulation of the ***fixed charge*** constraint seen above is a special case of the so-called ***big-M*** constraints formulation. Assume that a given (real or integer) positive variable x_j, which cannot exceed a given threshold value M, is restricted to having either value 0 or a value not less than a constant K. To model this requirement, it is sufficient to introduce a fictitious binary variable y_j and add the following constraints:

$$x_j \leqslant M y_j \tag{5.35}$$

$$x_j \geqslant K y_j. \tag{5.36}$$

Note that, as the variable y_j can only have values $\{0, 1\}$, then either we have $x_j = 0$ when $y_j = 0$ from (5.35), or we have $K \leqslant x_j \leqslant M$ when $y_j = 1$ from both (5.35,5.36).

The ***big-M*** constraints formulation can also be used to model logic conditions on real/integer variables as the following example on two variables $x_i \geqslant 0$ and $x_j \geqslant 0$

shows. Suppose that x_i and x_j, where x_i (x_j) cannot exceed a given threshold value M_i (M_j), represent two activities that are incompatible. This implies the following logic condition:

$$\text{IF } x_i > 0 \text{ THEN } x_j = 0 \text{ AND (VICE VERSA) IF } x_j > 0 \text{ THEN } x_i = 0.$$

To model this condition, it is sufficient to introduce two binary variables $y_i, y_j \in \{0, 1\}$ and add the following constraints:

$$x_i \leqslant M y_i \tag{5.37}$$

$$x_j \leqslant M y_j \tag{5.38}$$

$$y_i + y_j \leqslant 1. \tag{5.39}$$

Constraints [5.37,5.38] are introduced to link the real variables to the corresponding binary ones, while constraint [5.39] induces the logic condition on the real variables, namely it forbids the x_i, x_j variables to be contemporaneously > 0.

5.2.2.4. *Either–or constraints*

Consider the case where a choice must be made between two constraints so that one must hold, but not both. Namely, consider an *either–or* constraint of the following type:

EITHER

$$f(x_1, ..., x_n) \geqslant k \tag{5.40}$$

OR

$$f(x_1, ..., x_n) \leqslant \gamma \tag{5.41}$$

where we know that $f(x_1, ..., x_n)$ is bounded above by some large coefficient M. To model this constraint, it is sufficient to introduce a $0/1$ variable y_j, such that $y_j = 1$ ($y_j = 0$) if $f(x_1, ..., x_n) \geqslant k$ ($f(x_1, ..., x_n) \leqslant \gamma$) and add the constraints:

LOGIC CONDITION	LINEAR CONSTRAINT
$y_1 \rightarrow y_2$	$y_1 \leqslant y_2$
$y_1 \rightarrow \overline{y_2}$	$y_1 \leqslant 1 - y_2$
$(y_1 \bigcap y_2) \rightarrow y_3$	$y_1 + y_2 - 1 \leqslant y_3$
$(y_1 \bigcup y_2) \rightarrow y_3$	$y_1 + y_2 \leqslant 2y_3$
$\bigcap_{(i \in I)} y_i \rightarrow \omega$	$\sum_{i \in I} y_i - \lvert I \rvert + 1 \leqslant \omega$
$\bigcup_{(i \in I)} y_i \rightarrow \omega$	$\sum_{i \in I} y_i \leqslant \lvert I \rvert \cdot \omega$

Table 5.1. *Correspondence between logic conditions and linear constraints on* $0/1$ *variables*

$$f(x_1, ..., x_n) \leqslant \gamma + My_j \tag{5.42}$$

$$f(x_1, ..., x_n) \geqslant k - M(1 - y_j) \tag{5.43}$$

where, either $y_j = 0$, constraint [5.42] holds and reduces to constraint [5.41], while constraint [5.43] becomes redundant, or $y_j = 1$, constraint [5.43] holds and reduces to constraint [5.40], while constraint [5.42] becomes redundant.

5.2.2.5. *Linearization of quadratic* $0/1$ *variables*

Consider a mathematical programming model involving quadratic terms of the form $x_i x_j$ with $x_i, x_j \in \{0, 1\}$. It can be transformed into an equivalent linear model by introducing a new variable $y_{ij} \in \{0, 1\}$ such that $y_{ij} = x_i x_j$. As x_i and x_j are $0/1$ variables, this corresponds to $y_{ij} = \min\{x_i, x_j\}$ and can be made explicit by means of the following three linear constraints:

$$x_i \geqslant y_{ij}$$

$$x_j \geqslant y_{ij}$$

$$x_i + x_j \leqslant 1 + y_{ij}$$

To conclude this section, Table 5.1 depicts several examples of logic conditions and the corresponding linear constraints on binary variables to be added in a MILP

model (the left column indicates the logic condition and the right column the corresponding linear constraint with $y_i \in \{0,1\}, \forall i$). For an insightful work on how to give tight representations (with respect to the convex hull of the considered formulation) of logical conditions by means of linear constraints, see [YAN99].

5.3. More advanced MILP models

In this section we discuss fairly standard MILP formulations of several known combinatorial optimization problems spanning locations, graphs, networks and scheduling problems.

5.3.1. *Location models*

5.3.1.1. *The p-median problem*

In the p-median problem we consider a set N of n potential facilities and a set M of m clients. We want to open at most p facilities (medians) and assign each client to each nearest open facility so as to minimize the sum of the distances between the clients and their nearest open facility.

Let y_j be a $0-1$ variable such that $y_j = 1$ if node j is a median (it is open as a facility), $y_j = 0$ if it is not. Let x_{ij} be a $0-1$ variable such that $x_{ij} = 1$ if client i is allocated to facility j (and correspondingly facility j is median), $x_{ij} = 0$ if it is not. Let d_{ij} be the cost of allocating client i to facility j. A standard MILP formulation of the p-median problem (see [DAS95]) is as follows:

$$\min \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} x_{ij} \tag{5.44}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, ..., m \tag{5.45}$$

$$\sum_{j=1}^{n} y_j \leqslant p \tag{5.46}$$

$$x_{ij} \leqslant y_j \quad i = 1, ..., m,\ j = 1, ..., n \tag{5.47}$$

$$x_{ij}, y_j \in \{0,1\} \quad i = 1, ..., m,\ j = 1, ..., n \tag{5.48}$$

where the objective function [5.44] minimizes the total distance, constraints [5.45] impose the restriction that each client is allocated to a facility, constraint [5.46] requires there to be at most p medians, constraints [5.47] forbid the assignment of a client to a closed facility, and constraints [5.48] impose the restriction that the variables x_{ij} and y_j are binary.

5.3.1.2. *The p-center problem*

In the p-center problem, we still consider (as for the p-median problem) a set N of n potential facilities and a set M of m clients, where we want to open at most p facilities and assign each client to each nearest open facility. However, the objective function in this case is to minimize the radius, namely the maximum distance between a client and the nearest facility.

Let y_j be a $0-1$ variable such that $y_j = 1$ if node j is open as a facility, $y_j = 0$ if it is not. Let x_{ij} be a $0-1$ variable such that $x_{ij} = 1$ if client i is allocated to facility j, $x_{ij} = 0$ if he is not. Let d_{ij} be the cost of allocating client i to facility j. Let z be the radius that we want to minimize. A standard MILP formulation of the p-center problem (see [DAS95]) is as follows:

$$\min z \tag{5.49}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1, ..., m \tag{5.50}$$

$$\sum_{j=1}^{n} y_j \leqslant p \tag{5.51}$$

$$x_{ij} \leqslant y_j \quad i = 1, ..., m,\ j = 1, ..., n \tag{5.52}$$

$$\sum_{j=1}^{n} d_{ij} x_{ij} \leqslant z \quad i = 1, ..., m \tag{5.53}$$

$$x_{ij}, y_j \in \{0,1\} \quad i = 1, ..., m,\ j = 1, ..., n \tag{5.54}$$

where the objective function [5.49] minimizes the value of the radius, while constraints [5.50] impose the restriction that each client is allocated to a facility, constraint [5.51] requires that there are at most p open facilities, constraints [5.52] forbid the assignment of a client to a closed facility, constraints [5.53] force z to be not less than the distance from any client to the assigned facility, and constraints [5.54] impose the restriction that the variables x_{ij} and y_j are binary.

A different formulation of the p-center problem is provided in [ELL04]. Let us arrange the distance matrix (that is the matrix of the d_{ij} values) in such a way that $D^{\min} = D^0 < D^1 < ... < D^K = D^{\max}$ are the sorted different values in that matrix. Let y_j always be a $0-1$ variable such that $y_j = 1$ if node j is open as a facility, $y_j = 0$ if it is not. Let z_k ($k = 1, ..., K$) be a $0-1$ variable such that $z_k = 0$ only if it is possible to choose p facilities and cover all the clients within the radius D^{k-1}. Note that in an optimal solution $z_k = 0$ implies $z_{k+1} = ... = z_K = 0$ and the objective function value is strictly less than D^k. Analogously, $z_k = 1$ implies $z_{k-1} = ... = z_1 = 1$. The formulation is as follows.

$$\min D^0 + \sum_{k=1}^{K} (D^k - D^{k-1}) z_k \tag{5.55}$$

subject to

$$\sum_{j=1}^{n} y_j \geqslant 1 \tag{5.56}$$

$$\sum_{j=1}^{n} y_j \leqslant p \tag{5.57}$$

$$z_k + \sum_{j: d_{ij} < D^k} y_j \geqslant 1 \quad i = 1, ..., n,\ k = 1, ..., K \tag{5.58}$$

$$y_j, z_k \in \{0,1\} \quad j = 1, ..., n,\ k = 1, ..., K. \tag{5.59}$$

Here the objective function [5.55] still minimizes the value of the radius (note that all the cost coefficients are positive). Constraint [5.56] (which is redundant in the previous formulation) discards solutions with no open facilities. Constraint [5.57] requires there to be at most p open facilities. Constraints [5.58] indicate that for a given k, $z_k = 0$ if and only if all clients can be served at a distance strictly less than D^k. Hence, when $z_k = 1$, the value $D^k - D^{k-1}$ is added to the radius in the objective function [5.55]. Finally, constraints [5.59] impose the restriction that the variables y_j and z_k are binary.

5.3.2. *Graphs and networks models*

5.3.2.1. *The maximum clique problem*

Consider a graph $G = (V, E)$ composed of a set V of n vertices and a set E of m edges. When two vertices are connected by an edge, they are denoted as adjacent. A clique in G is a subgraph where the vertices are all pairwise adjacent. The maximum clique problem consists of finding a clique of maximum cardinality.

Let x_j ($j \in V$) be a variable $0 - 1$ such that $x_j = 1$ if vertex j belongs to the clique, $x_j = 0$ if it does not.

The standard MILP formulation of this model is as follows:

$$\max \sum_{j \in V} x_j \tag{5.60}$$

subject to

$$x_i + x_j \leqslant 1 \quad \forall i, j : (i, j) \notin E \tag{5.61}$$

$$x_j \in \{0, 1\} \quad \forall j \in V \tag{5.62}$$

where the objective function [5.60] maximizes the cardinality of the clique, constraints [5.61] impose the restriction that for all pairs of non-adjacent vertices at most one of them can belong to the clique, and constraints (5.62) impose the restriction that the variables x_j must be binary.
With the above formulation, the set of constraints [5.61] has cardinality $|\overline{E}|$. Let $\overline{E_j}$ indicate the set of vertices non-adjacent to vertex j. A more compact reformulation (involving just $|V|$ constraints, though widening the solutions space of the corresponding linear programming relaxation, see [CRO94]) of this constraints set is as follows.

$$\sum_{i \in \overline{E_j}} x_i + |\overline{E_j}| x_j \leqslant |\overline{E_j}| \quad \forall j \in V \tag{5.63}$$

where constraints [5.63] indicate that for each vertex j, if j is in the clique ($x_j = 1$), then all vertices non-adjacent to j cannot belong to the clique.

5.3.2.2. *The graph coloring problem*

Consider a graph $G = (V, E)$ with $|V| = n$. A coloring of G is an assignment of colors to the vertices such that two adjacent vertices have different colors. A k-coloring of G is a coloring that uses k colors. The chromatic number of G is the smallest number of colors needed to color G and it is denoted by $\chi(G)$. The graph coloring problem consists of finding $\chi(G)$. Let u_i ($i = 1, ..., n$) be a $0 - 1$ variable such that $u_i = 1$ if color i is used (as there are n vertices, at most n colors will be used), $u_i = 0$ if it is not. Let x_{ij} ($i = 1, ..., n$, $j \in N$) be a $0 - 1$ variable such that $x_{ij} = 1$ if vertex j is colored with color i, $x_{ij} = 0$ if it is not.

The standard MILP formulation (see, for example, [COL02]) of this model is as follows:

$$\min \sum_{i=1}^{n} u_i \tag{5.64}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j \in N \tag{5.65}$$

$$x_{ij} + x_{kj} \leqslant u_j \quad \forall i \in V,\ k \in V,\ (i,k) \in E, \quad j = 1, ..., n \tag{5.66}$$

$$u_i, x_{ij} \in \{0, 1\} \quad \forall i = 1, ..., n,\ \forall j \in N \tag{5.67}$$

where the objective function [5.64] minimizes the number of colors, constraints [5.65] impose the restriction that each vertex is colored, constraints [5.66] impose the restriction that two adjacent vertices cannot have the same color and that $u_j = 1$ when some vertex has color j, and constraints [5.67] impose the restriction that the variables u_i and x_{ij} must be binary.

5.3.2.3. *The shortest spanning tree problem*

The shortest spanning tree problem can be expressed in the following way. A graph $G = (V, E)$ has a set N of n vertices and a set E of m edges, where each edge $(i, j) \in E$ has a cost (distance) c_{ij}. We search for a spanning tree (a connected acyclic graph) $T = (V, E')$ in the graph G such that the cost of the edges of T, that is $\sum_{(i,j)\in E'}$, is minimum. Note that $|E'| = n - 1$, that is a spanning tree in a graph with n vertices contains exactly $n - 1$ edges. In a tree each of the vertices can be considered, without loss of generality, as being the root vertex 1: then, given vertex 1, the edges of a tree can always be split into levels (at most $n - 1$), where the edges containing vertex 1 are assigned to the first level, the edges not containing vertex 1 but containing vertices adjacent to vertex 1 are assigned to the second level and so on. Let x_{ij}^k be a $0-1$ variable such that $x_{ij}^k = 1$ if edge (i, j) belongs to the spanning tree and is assigned to level k, $x_{ij}^k = 0$ if it does not. The following MILP formulation holds for the minimum spanning tree problem.

$$\min \sum_{k=1}^{n-1} \sum_{(i,j)\in E} c_{ij} x_{ij}^k \tag{5.68}$$

subject to

$$\sum_{k=1}^{n-1} x_{ij}^k \leqslant 1 \quad \forall (i,j) \in E \tag{5.69}$$

$$\sum_{j>1, j\in V} x_{1j}^1 \geqslant 1 \tag{5.70}$$

$$\sum_{k=1}^{n-1} \Big(\sum_{i<j} x_{ij}^k + \sum_{l>j} x_{jl}^k\Big) \geqslant 1 \quad \forall j \neq 1 \in V \tag{5.71}$$

$$\sum_{k=1}^{n-1} \sum_{(i,j)\in E} x_{ij}^k = n - 1 \tag{5.72}$$

$$\sum_{h<i} x_{hi}^{k-1} + \sum_{l>i,\ l\neq j} x_{il}^{k-1} + \sum_{h<j\ h\neq i} x_{hj}^{k-1} + \sum_{l>j} x_{jl}^{k-1} \geqslant x_{ij}^k \quad \forall k = 2, ..., n \quad \forall (i,j) \in E \tag{5.73}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k = 1, ..., n-1, \;\; \forall (i,j) \in E \tag{5.74}$$

where the objective function [5.68] minimizes the total cost (distance) of the tree; constraints [5.69] impose the restriction that each edge can be assigned to at most one level of the tree; constraint [5.70] imposes the restriction that vertex 1 is the root vertex, that is at least one edge containing vertex 1 must be assigned to the first level; constraints [5.71] impose the restriction that for each vertex j at least one edge containing vertex j belongs to the tree and is assigned to some level $1 \leqslant k \leqslant n-1$; constraint [5.72] imposes the restriction that the tree is composed of exactly $n-1$ edges; constraints [5.73] impose the restriction that for each edge (i,j) and for each level $k \geqslant 2$, if edge (i,j) is assigned to level k, then at least one edge containing either vertex i or vertex j must be assigned to level $k-1$; finally, constraints [5.74] impose the restriction that the variables x_{ij}^k must be binary.

5.3.2.4. *The traveling salesman problem*

The traveling salesman problem can be expressed in the following way. Given a directed graph $G = (N, A)$ with a set N of n nodes and a set A of m arcs, where each arc $(i,j) \in A$ has a cost (distance) c_{ij}, we search for the minimum cost (distance) Hamiltonian circuit on graph G. Note that a circuit is Hamiltonian if and only if all nodes are visited exactly once by the circuit. Let A_S denote the subset of arcs linking the nodes belonging to subset $S \subset N$. Let x_{ij} be a $0-1$ variable such that $x_{ij} = 1$ if arc (i,j) belongs to the circuit, $x_{ij} = 0$ if it does not. The standard MILP formulation (see [DAN54]) of the traveling salesman problem is as follows.

$$\min \sum_{i,j \in N, \;\; i \neq j} c_{ij} x_{ij} \tag{5.75}$$

subject to

$$\sum_{i \in N, \; i \neq j} x_{ij} = 1 \quad \forall j \in N \tag{5.76}$$

$$\sum_{j \in N, \; j \neq i} x_{ij} = 1 \quad \forall i \in N \tag{5.77}$$

$$\sum_{(i,j) \in A_S} x_{ij} \leqslant |S| - 1, \;\; \forall S : S \subseteq N, \, 2 \leqslant |S| \leqslant n-2 \tag{5.78}$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in N, \quad i \neq j \tag{5.79}$$

where the objective function [5.75] minimizes the total cost (distance) of the circuit, constraints [5.76] impose the restriction that exactly one arc enters each node, constraints [5.77] impose the restriction that exactly one arc exits from each node, constraints [5.78] are the so-called subtour elimination constraints, namely they impose the restriction that no subtour can exist, and constraints [5.79] impose the restriction that the variables x_{ij}^k must be binary.

The above formulation presents a limited number of variables, just $O(m)$, but an exponential number of subtour elimination constraints [5.78]. A formulation with a polynomial number of variables and constraints is proposed in [MAC03]. Here, another formulation with $O(mn)$ variables and $O(n^2)$ constraints based on positional variables is given. Let x_{ij}^k be a $0-1$ variable such that $x_{ij}^k = 1$ if arc (i,j) is the $k-th$ arc of the circuit, $x_{ij}^k = 0$ if it is not. Note that any node can be arbitrarily selected to be the first node of the circuit (and correspondingly one of its outgoing arcs will be the first arc of the circuit). A MILP formulation of the traveling salesman problem is as follows.

$$\min \sum_{k=1}^{n} \sum_{i,j \in N,\ i \neq j} c_{ij} x_{ij}^k \tag{5.80}$$

subject to

$$\sum_{k=1}^{n} \sum_{i \in N,\ i \neq j} x_{ij}^k = 1 \quad \forall j \in N \tag{5.81}$$

$$\sum_{k=1}^{n} \sum_{j \in N,\ j \neq i} x_{ij}^k = 1 \quad \forall i \in N \tag{5.82}$$

$$\sum_{i,j \in N,\ i \neq j} x_{ij}^k = 1 \quad \forall k = 1, ..., n \tag{5.83}$$

$$\sum_{i \in N,\ i \neq j} x_{ij}^k = \sum_{l \in N,\ l \neq j} x_{jl}^{k+1} \quad \forall j \in N, \quad \forall k = 1, ..., n-1 \tag{5.84}$$

$$\sum_{i \in N,\ i \neq j} x_{ij}^n = \sum_{l \in N,\ l \neq j} x_{jl}^1 \quad \forall j \in N \tag{5.85}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k = 1,...,n, \;\; \forall i,j \in N, \;\; i \neq j \tag{5.86}$$

where the objective function [5.80] minimizes the total cost (distance) of the circuit; constraints [5.81] impose the restriction that exactly one arc enters each node at exactly one position; constraints [5.82]impose the restriction that exactly one arc exits from each node in exactly one position; constraints [5.83] impose the restriction that exactly one arc is assigned to each position in the circuit; constraints [5.84] impose the restriction that for each node j, if it has an ingoing arc at position $k < n$ in the circuit, then it must also have an outgoing arc at position $k + 1$ in the circuit (*vice versa*, if it does not have an ingoing arc at position k in the circuit, then it cannot have an outgoing arc at position $k + 1$ in the circuit); constraints [5.85] impose the restriction that for each node j, if it has an ingoing arc at the last position in the circuit, then it must also have an outgoing arc at the first position in the circuit (*vice versa*, if it does not have an ingoing arc at the last position in the circuit, then it cannot have an outgoing arc at the first position in the circuit); finally, constraints [5.86] impose the restriction that the variables x_{ij}^k must be binary.

5.3.2.5. *The multicommodity network flow problem*

The minimum cost multicommodity network flow problem generalizes the minimum cost flow problem to multiple commodities. Here we want to determine a least cost shipment of several commodities through a network in order to satisfy demands at certain nodes from available supplies at other nodes, where the commodities share the capacity of the same arcs and all the flows are integer. Consider a network $G = (N, A)$ composed of a set N of n nodes, and a set A of m arcs, where the direct arc (i, j) connects node i to node j. Let K be a set of k commodities where each commodity must generally be shipped from multiple sources to multiple destinations, and where sources and destinations are nodes of the considered network. A parameter b_i^k is associated with each node i and each commodity k. If $b_i^k > 0$, node i is the source node for commodity k with a supply of b_i^k units of flow. If $b_i^k < 0$, node i is the destination node for commodity k with a demand of $-b_i^k$ units of flow. If $b_i^k = 0$, node i is a transhipment node for commodity k. Let c_{ij}^k denote the unit flow cost of commodity k on arc (i, j). We assume $c_{ij}^k \geqslant 0 \; \forall (i,j) \in A, \forall k \in K$. Each arc (i, j) has a capacity $u_{ij} \geqslant 0$ that restricts the total flow of all commodities on that arc. Let $x_{ij}^k \geqslant 0$ denote the flow of commodity k on arc (i, j). The so-called node–arc formulation (see, for instance, [AHU93]) of the minimum cost multicommodity network flow problem is as follows.

$$\min \sum_{k \in K, \; (i,j) \in A} c_{ij}^k x_{ij}^k \tag{5.87}$$

subject to

$$\sum_{l \in N} x_{jl}^k - \sum_{i \in N} x_{ij}^k = b_j^k \quad \forall j \in N, \quad \forall k \in K \tag{5.88}$$

$$\sum_{k \in K} x_{ij}^k \leqslant u_{ij} \quad \forall (i,j) \in A \tag{5.89}$$

$$x_{ij}^k \;\; integer \geqslant 0 \quad \forall (i,j) \in A, \;\; \forall k \in K \tag{5.90}$$

where the objective function [5.87] minimizes the total multicommodity flow cost; constraints [5.88] represent the ordinary mass balance constraints, that is the requirement that, for each commodity, the difference between the flow that enters a node and the flow that leaves that node is equal to the supply/demand of that node; constraints [5.89] are the so-called bundle constraints, that is they state that the total flow on each arc must be less than or equal to its capacity; finally, constraints [5.90] impose the restriction that the variables x_{ij}^k must be positive integers.

A different formulation of the same problem (the so-called arc–path formulation, see [AHU93]), but with an exponential number of variables, can be obtained in the following way. Let P_k denote the collection of all directed paths p between all source–destination pairs of commodity k. Also, let the set of all sources and destinations of commodity k be represented by Q_k. Let δ_{ij}^{pk} be a $0-1$ arc–path constant where $\delta_{ij}^{pk} = 1$ if arc (i,j) belongs to path p of commodity k, $\delta_{ij}^{pk} = 0$ if it does not. Let γ_i^{pk} be a node–path constant where $\gamma_i^{pk} = 1$ if node i is the source of path p of commodity k, $\gamma_i^{pk} = -1$ if node i is the destination of path p of commodity k, $\gamma_i^{pk} = 0$ if node i is neither the source nor the destination of path p of commodity k. Let $c_p^k = \sum_{(i,j) \in A} c_{ij}^k \delta_{ij}^{pk}$ denote the unit flow cost of path p of commodity k. Let y_p^k denote the integer flow on path p of commodity k. The arc–path formulation of the minimum cost multicommodity network flow problem is as follows.

$$\min \sum_{k \in K, \; p \in P_k} c_p^k y_p^k \tag{5.91}$$

subject to

$$\sum_{p \in P_k} \gamma_{ij}^{pk} y_p^k = b_i^k \quad \forall i \in Q^k, \quad \forall k \in K \tag{5.92}$$

$$\sum_{k \in K,\ p \in P_k} \delta_{ij}^{pk} y_p^k \leqslant u_{ij} \quad \forall (i,j) \in A \tag{5.93}$$

$$y_p^k \ \ integer \geqslant 0 \quad \forall k \in K \quad \forall p \in P_k \tag{5.94}$$

where the objective function [5.91] minimizes the total multicommodity flow cost, constraints [5.92] represent the ordinary mass balance constraints, constraints [5.93] are the bundle constraints, and constraints [5.94] impose the restriction that the variables y_p^k must be positive integers.

5.3.3. *Machine scheduling models*

5.3.3.1. *The job shop problem*

In the job shop problem a set of m machines and a set of n jobs are given. Each job $j \in J$ consists of a set of m operations to be executed on distinct machines, ordered according to a given permutation $\sigma_j = (\sigma_j^1, \sigma_j^2, ..., \sigma_j^m)$ of the machines: job j must be executed first on machine σ_j^1, then on σ_j^2, and so on. Let $p_j(i)$ be the processing time of job j on machine i. Each job cannot start before its release date r_j and pre-emption is not allowed, that is once an operation has started execution, it must be executed for its entire duration without interruption. Each resource can execute at most one operation at a time. For each job, the first operation starts after the job release date and the operations are processed according to the specified order. Let $C_j(i)$ denote the completion time of job j on machine i. Note that the final completion time for job j corresponds to $C_j(\sigma_j^m)$, namely the completion time for job j on its last visited machine σ_j^m. Let $C_{\max}$ denote the maximum of the jobs' final completion times. Let $x_{kl}(i)$ be a $0-1$ variable such that $x_{kl}(i) = 1$ if job k is processed before job l on machine i, $x_{kl}(i) = 0$ if it is not, i.e. when job l is processed before job k on machine i. Let M be some large constant such that $C_j(i) \leqslant M \ \forall j \in J, \forall i = 1, ..., m$ is always verified. The standard MILP model (see [BAK74]) based on a big-M formulation (see subsection 5.2.2.3) of the job shop problem is as follows.

$$\min C_{\max} \tag{5.95}$$

subject to

$$C_j(\sigma_j^m) \leqslant C_{\max} \quad \forall j \in J \tag{5.96}$$

$$C_j(\sigma_j^1) \geqslant r_j + p_j(\sigma_j^1) \quad \forall j \in J \tag{5.97}$$

$$C_j(\sigma_j^{t+1}) \geqslant C_j(\sigma_j^t) + p_j(\sigma_j^{t+1}) \quad \forall j \in J, \quad \forall t = 1, ..., m-1 \tag{5.98}$$

$$\begin{cases} C_k(i) \geqslant C_l(i) + p_k(i) - Mx_{kl}(i) \quad \forall k, l \in J : k < l, \quad \forall i = 1, ..., m \\ C_l(i) \geqslant C_k(i) + p_l(i) - M[1 - x_{kl}(i)] \quad \forall k, l \in J : k < l, \quad \forall i = 1, ..., m \end{cases} \tag{5.99}$$

$$C_{\max}, C_j(i) \geqslant 0 \quad \forall j \in J, \ \ \forall i = 1, ..., m \tag{5.100}$$

$$x_{kl}(i) \in \{0, 1\} \quad \forall k, l \in J : k < l, \ \ \forall i = 1, ..., m \tag{5.101}$$

where the objective function [5.95] minimizes the maximum of the jobs' completion times, constraints [5.96] ensure that the maximum of the jobs' completion times is not less than the final completion time of each job j, constraints [5.97] impose the restriction that each job cannot start processing before its release date, constraints [5.98] impose the precedence conditions between the operations for each job, constraints [5.99] impose the restriction that for each machine i either job k precedes job l or job l precedes job k, constraints [5.100] impose the restriction that $C_{\max}$ and the variables $C_j(i)$ must be positive, and constraints [5.101] impose the restriction that the variables $x_{kl}(i)$ must be binary.

The continuous relaxation of this big-M formulation does not allow tightening of the lower bounds because the links in constraints [5.99] between the decision variables $x_{kl}(i)$ and the secondary variables $C_j(i)$ are quite weak. To get a tighter, though more complex, formulation (requiring $O(m^2n^2)$ variables and $O(m^3n^3)$ constraints), we use positional completion times variables (see [LAS92] where this type of formulation was introduced for single machine models) related to each job operation. Given n jobs and m machines, we then have $\tau = mn$ operations. We know that for any solution of the job shop problem, any given operation i is processed in some position k if there are $k-1$ operations completing not later than operation i and the remaining $\tau - k$ operations do not complete before operation i. Let $y_{i,k}$ be a $0-1$ variable such that $y_{i,k} = 1$ if operation i is processed in position k, $y_{i,k} = 0$ if it is not. Let p_i denote the processing time of operation i. Let M_i denote the set of operations to be executed

on the same machine on which operation i is processed. Let s_i denote the operation following i for the same job. Let C_k be the completion time of the k-th operation. The MILP model based on positional completion times of the job shop problem is as follows.

$$\min C_\tau \tag{5.102}$$

subject to

$$\sum_{k=1}^{\tau} y_{i,k} = 1 \quad \forall i = 1, ..., \tau \tag{5.103}$$

$$\sum_{i=1}^{\tau} y_{i,k} = 1 \quad \forall k = 1, ..., \tau \tag{5.104}$$

$$C_k \geqslant C_{k-1} \quad \forall k = 2, ..., \tau \tag{5.105}$$

$$C_k \geqslant C_l + p_i[y_{i,k} + \sum_{j \in \{M_i \cup s_i\}} y_{j,l} - 1] \quad \forall k = 2, ..., \tau, \ \ \forall l = 1, ..., k-1, \ \ \forall i = 1, ..., \tau \tag{5.106}$$

$$\sum_{j=1}^{k-1} y_{i,j} \geqslant y_{s_i,k} \quad \forall k = 2, ..., \tau, \quad \forall i : s_i \neq \{\} \tag{5.107}$$

$$C_k \geqslant 0 \quad \forall j \in J, \ \ \forall k = 1, ..., \tau \tag{5.108}$$

$$y_{i,k} \in \{0, 1\} \quad \forall i = 1, ..., \tau \quad \forall k = 1, ..., \tau \tag{5.109}$$

where the objective function [5.102] minimizes the maximum of the jobs' completion times, that is the completion time of the last operation; constraints [5.103] impose the restriction that each operation occupies exactly one position; constraints [5.104] impose the restriction that each position is occupied by exactly one operation; constraints [5.105] impose the restriction that, given an operation in some position k, it

cannot be completed before the operation in position $k-1$; constraints [5.106] impose the restriction that, given an operation i in some position k, the previous operation of the same job, and all operations processed on the same machine as operation i, must be completed before the start of operation i if they are placed in a position $l < k$; constraints [5.107] impose the restriction that, given two consecutive operations i, s_i of the same job with s_i placed in position k, operation i must be placed in a position $j < k$; finally, constraints [5.108] impose the restriction that the variables C_k must be positive and constraints [5.109] impose the restriction that the variables $y_{i,k}$ must be binary.

5.3.3.2. *The single machine problem: time-indexed formulation*

In the non-pre-emptive single-machine problem, a set J of n jobs has to be scheduled, where each job j has a processing time p_j, a release time r_j, and the machine can handle no more than one job at a time. The objective function is the minimization of the weighted sum of the start times. Assuming that all p_j's are integers, a time-indexed formulation [AKK00] based on time discretization, i.e. time is divided into periods, where period t starts at time $t-1$ and ends at time t, can be devised. The planning horizon is denoted by T, which means that all jobs have to be completed by time T. It is sufficient, for instance, to assume that $T \geqslant \max_j r_j + \sum_j p_j$. Let c_{jt} be the cost of job j if it is started in period t and let x_{jt} be a $0-1$ variable such that $x_{jt} = 1$ if job j is started at the beginning of period t, $x_{jt} = 0$ if it is not. The time-indexed formulation is as follows:

$$\min \sum_{j=1}^{n} \sum_{t=1}^{T-p_j+1} c_{jt} x_{jt} \tag{5.110}$$

subject to

$$\sum_{t=1}^{T-p_j+1} x_{jt} = 1 \quad \forall j = 1, ..., n \tag{5.111}$$

$$\sum_{j=1}^{n} \sum_{s=t-p_j+1}^{t} x_{js} \leqslant 1 \quad \forall t = 1, ..., T \tag{5.112}$$

$$x_{jt} = 0 \quad \forall j = 1, ..., n \quad \forall t = 1, ..., r_j \tag{5.113}$$

$$x_{jt} \in \{0, 1\} \quad \forall j = 1, ..., n \quad \forall t = 1, ..., T - p_j + 1 \tag{5.114}$$

where the objective function [5.110] minimizes the weighted sum of the start times, constraints [5.111] impose the restriction that each job is processed between time 0 and time T, constraints [5.112] impose the restriction that the machine can handle no more than one job at a time, constraints [5.113] impose the restriction that each job cannot start processing before its release time, and constraints [5.114] impose the restriction that the variables x_{jt} must be binary. This formulation can be used to model several single-machine scheduling problems using an appropriate choice of the objective coefficients and possibly restriction of the set of variables. It is, however, pseudo-polynomial in the number of variables.

5.3.3.3. *The single machine problem with tardy jobs cost function*

In this case we always have a non-pre-emptive single-machine problem, where a set J of n jobs has to be scheduled: each job j has a processing time p_j, release time 0, a due date d_j, a deadline $\tilde{d}_j$ and a weight w_j, and the machine can handle no more than one job at a time. Let C_j denote the completion time of job j. We say that a job j is tardy (early) if its completion time C_j is such that $C_j > d_j$ ($C_j \leqslant d_j$). Let x_j be a $0-1$ variable such that $x_j = 1$ if job j is early, $x_{jt} = 0$ if it is not, namely if job j is tardy. Consider as the objective function the minimization of the weighted number of tardy jobs. Notice that the presence of deadlines imposes the restriction for each job j that $C_j \leqslant \tilde{d}_j$. Assume that the jobs are indexed in non-decreasing order of due dates, that is $d_1 \leqslant d_2 \leqslant ... \leqslant d_n$. A peculiarity of this problem is that for any feasible solution, it is sufficient to know for each job j whether it is early or tardy in order to get the corresponding schedule. Indeed, if we know the values $x_j \ \forall j$, then the related schedule can be obtained by sequencing the jobs in non-decreasing order of $d_j x_j + \tilde{d}_j(1-x_j)$. Based on the above remark, the following formulation [BAP02] with n variables and $2n$ constraints holds for the considered problem.

$$\max \sum_{j=1}^{n} w_j x_j \tag{5.115}$$

subject to

$$\sum_{i:\ \tilde{d}_i \leqslant d_j} p_i + \sum_{k:\ d_k \leqslant d_j\ \cap\ \tilde{d}_k > d_j} p_k x_k \leqslant d_j \quad \forall j = 1, ..., n \tag{5.116}$$

$$\sum_{i:\ \tilde{d}_i \leqslant \tilde{d}_j} p_i + \sum_{k:\ d_k \leqslant \tilde{d}_j\ \cap\ \tilde{d}_k > \tilde{d}_j} p_k x_k \leqslant \tilde{d}_j \quad \forall j = 1, ..., n \tag{5.117}$$

$$x_j \in \{0, 1\} \quad \forall j = 1, ..., n \tag{5.118}$$

where the objective function [5.115] maximizes the weighted number of early jobs, which is equivalent, apart from a constant factor, to minimizing the weighted number of tardy jobs; constraints [5.116] impose the restriction that for each due date d_j all jobs i with deadlines not greater than d_j ($\tilde{d}_i \leqslant d_j$) plus all early jobs k with deadlines greater than d_j ($\tilde{d}_k > d_j$) but due date not greater than d_j ($d_k \leqslant d_j$) must be completed by d_j; constraints [5.117] impose the restriction that for each deadline $\tilde{d}_j$, all jobs i with deadlines not greater than $\tilde{d}_j$ ($\tilde{d}_i \leqslant \tilde{d}_j$) plus all early jobs k with deadlines greater than $\tilde{d}_j$ ($\tilde{d}_k > \tilde{d}_j$) but due date not greater than $\tilde{d}_j$ ($d_k \leqslant \tilde{d}_j$) must be completed by $\tilde{d}_j$; finally, constraints [5.118] impose the restriction that the variables x_j must be binary.

5.4. Conclusions

In this chapter we have discussed basic and advanced MILP models for known combinatorial optimization problems and we have outlined some general techniques for MILP modeling.

Typically, a MILP model is considered "good" if its continuous linear programming relaxation is sufficiently tight, that is the optimal solution value of the continuous linear programming relaxation is sufficiently close to the optimal solution value of the considered problem.

We wish to point out that, to derive a "good" MILP model, it is strongly recommended that all the structural properties of the considered problem be taken into account (and this is why modeling is sometimes considered to be an "art"). Indeed, the more we know about a problem, the better will be the corresponding MILP model, where the general modeling techniques presented here, together with a reasonable background on known MILP models, are often a sufficient tool to derive such a model for problems having strong structural properties.

5.5. Bibliography

[AHU93] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, "*Network Flows: Theory, Algorithms, and Applications*", Prentice Hall, New Jersey, 1993.

[BAK74] K.R. Baker, "*Introduction to Sequencing and Scheduling*", Wiley, New York, 1974.

[BAP02] Ph. Baptiste, personal communication, 2002.

[COL02] P. Coll, J. Marenco, I. Mendez Diaz, P. Zabala, "Facets of the graph coloring polytope", *Annals of Operations Research*, 116, 2002, 79–90.

[DAN54] G.B. Dantzig, R. Fulkerson, S.M. Johnson, "Solution of a large-scale traveling salesman problem", *Operations Research*, 2, 1954, 393–410.

[DAS95] M.S. Daskin, "*Network and Discrete Location*", Wiley, New York, 1995.

[CRO94] F. Della Croce, R. Tadei, "A multi-KP modeling for the maximum clique problem", *European Journal of Operational Research*, 73, 1994, 555–561.

[ELL04] S. Elloumi, M. Labbé, Y. Pochet, "A New Formulation and Resolution Method for the p-Center Problem", *INFORMS Journal on Computing*, 16, 2004, 84–94.

[LAS92] J.B. Lasserre, M. Queyranne, "Generic scheduling polyhedra and a new mixed-integer formulation for single-machine scheduling", in E. Balas, G. Cornuéjols, R. Kannan eds., "Integer Programming and Combinatorial Optimization, Proceedings of the Second IPCO-Conference", University Printing and Publications, Carnegie Mellon University, Pittsburgh, 1992, 136–149.

[MAC03] N. Maculan, G. Plateau, A. Lisser, "Integer linear models with a polynomial number of variables and constraints for some classical combinatorial problems", *Pesquisa Operacional*, 23, 1, 2003, 161–168.

[PLA02] F. Plastria, "Formulating logical implications in combinatorial optimisation", *European Journal of Operational Research*, 140, 2002, 338–353.

[AKK00] J.M. Van Den Akker, C.A.J. Hurkens, M.W.P. Savelsbergh, "Time-Indexed Formulations for Machine Scheduling Problems: Column Generation", *INFORMS Journal on Computing*, 12, 2000, 111–124.

[WIL90] H. P. Williams, "*Model Building in Mathematical Programming*", Wiley, Chichester, 1990.

[YAN99] H. Yan, J. N. Hooker, "Tight representation of logical constraints as cardinality rules", *Mathematical Programming*, 85, 1999, 363–378.