Optimization Concepts—I: Introductory Level

Miguel Casquilho and João Luís de Miranda

Abstract Optimization subjects are addressed from both an introductory and an advanced standpoint. Theoretical subjects and practical issues are focused, conjugating Optimization basics with the implementation of useful tools and SC (supply chain) models. By now, in the *Introductory* section, Linear Programming, Integer Programming, related models, and others of interest are treated. In another chapter about these subjects in this handbook, more advanced related topics are addressed.

Keywords Optimization · Supply chain · Linear Programming

1 Introduction

Optimization is the act or methodology of driving something, such as a system, as near as possible to full functionality or effectiveness, the word having appeared one and a half century ago. Nowadays, the concept is essential, in a global context, because even a small improvement in a mathematical, economical or industrial solution can be decisive in comparison with other, nonoptimal solutions.

A crucial contribution to optimization was the discovery of a computationally simple method to solve "linear programming" problems, in the 1940s, by George Dantzig, the so-called *simplex method*. This method, which is important in the

M. Casquilho (✉)
Department of Chemical Engineering, Instituto Superior Técnico,
Ave. Rovisco Pais IST, 1049-001 Lisbon, Portugal
e-mail: mcasquilho@tecnico.ulisboa.pt

J.L. de Miranda
Departamento de Tecnologias e Design, Escola Superior de Tecnologia e Gestão,
Instituto Politécnico de Portalegre, 7300-110 Portalegre, Portugal
e-mail: jlmiranda@estgp.pt

M. Casquilho · J.L. de Miranda
CERENA—Centro de Recursos Naturais e Ambiente, Instituto Superior Técnico,
University of Lisbon, Ave. Rovisco Pais, IST, 1049-001 Lisbon, Portugal

A.P. Barbosa-Póvoa et al. (eds.), *Optimization and Decision Support Systems for Supply Chains*, Lecture Notes in Logistics,
DOI 10.1007/978-3-319-42421-7_5

Supply Chain problems, has since been very much used and studied, and important extensions have been obtained, some maintaining the efficiency of the original algorithm (as the Transportation Problem, the Transshipment Problem, the Assignment Problem), some addressing other, more complex situations, namely Integer Programming, which uses Linear Programming as a tool included in other techniques, of which the branch-and-bound is explained in this text. (Source computer code of some of these subjects can be obtained on request.)

2 Linear Programming (LP)

"**The Best of the 20th Century: Editors Name Top 10 Algorithms**" (Cipra 2000)

1947: George Dantzig, at the RAND Corporation, has created the simplex method for linear programming. In terms of widespread application, Dantzig's algorithm is one of the most successful of all times: linear programming dominates the world of industry, where economic survival depends on the ability to optimize within budgetary and many other constraints. (Of course, the "real" problems of industry are often nonlinear, and the use of linear programming is sometimes dictated by the computational budget.) The simplex method appears an elegant way of arriving at optimal answers. Although theoretically susceptible to exponential delays, the algorithm in practice is highly efficient—which in itself says something interesting about the nature of computation.

In the following, some examples are given, the theory of which can be found in several classical books, such as Zionts (1974), Bronson (2010), Hillier and Lieberman (2009). An intuitive algebraic approach is given below to solve linear programming problems. It is meant to show that the method is iterative, as is readily revealed.

$$\begin{array}{lrcl} [\max]z = & 0.56x_1 + 0.42x_2 & & \\ \text{s. to} & x_1 + 2x_2 & \leq & 240 \\ & 1.5x_1 + x_2 & \leq & 180 \\ & x_1 & \leq & 110 \end{array} \tag{1}$$

$$\mathbf{A} \quad \begin{array}{lrcl} [\max]z = & 0.56x_1 + 0.42x_2 & & \\ & x_1 + 2x_2 + \{x_3\} & = & 240 \\ & 1.5x_1 + x_2 + \{x_4\} & = & 180 \\ & x_1 + \{x_5\} & = & 110 \end{array} \tag{2}$$

This has (always) an obvious, sure solution. Let

$$x_1, x_2 = 0 \tag{3}$$

Then

$$\begin{bmatrix} x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 240 \\ 180 \\ 110 \end{bmatrix} \tag{4}$$

$$z = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 240 \\ 180 \\ 110 \end{bmatrix} = 0 \tag{5}$$

Is this optimal? How to improve it?

There does not appear (Hillier and Lieberman 2005) to exist a systematic way of setting *all* the nonbasic variables, i.e., those that were made zero, simultaneously to optimal values—hence, an *iterative* method should be followed.

Choose the variable that increases the objective function *most* per unit (this choice is somehow arbitrary), in the example, x_1, because its coefficient (0.56) is the largest.

According to the constraints, x_1 can be increased till:

$$\mathbf{B} \quad \begin{matrix} x_1 & = 240 \\ 1.5x_1 & = 180 \\ x_1 & = 110 \end{matrix} \quad \rightarrow \quad \begin{matrix} x_1 & = 240 \\ x_1 & = 120 \\ x_1 & = 110 \end{matrix} \tag{6}$$

The *third* equation in (2) leads to $x_1 = 110$ and $x_5 = 0$. This choice comes from the fact that it determines the smallest, most stringent limit. If this limit were exceeded, at least one other variable would become negative. The variable x_1 will be the *entering* variable and x_5 the *leaving* variable:

$$\mathbf{C} \quad x_1 = 110 - x_5 \tag{7}$$

Substituting for x_1 everywhere (except in "its own" constraint, i.e., the one that led to its choice), we have

$$\begin{matrix} [\max]z = & 0.56(110 - x_5) & +0.42x_2 & & & & & & \\ & (110 - x_5) & +2x_2 & +x_3 & & & = & 240 \\ & 1.5(110 - x_5) & +x_2 & & +x_4 & & = & 180 \\ & x_1 & & & & +x_5 & = & 110 \end{matrix} \tag{8}$$

$$\mathbf{A} \quad \begin{matrix} [\max]z = & 0.42x_2 & & & -0.56x_5 & & +61.6 \\ & +2x_2 & +\{x_3\} & & -x_5 & = & 130 \\ & x_2 & & +\{x_4\} & -1.5x_5 & = & 15 \\ \{x_1\} & & & & +x_5 & = & 110 \end{matrix} \tag{9}$$

which is equivalent to Eq. (2).

We now have a **new** (equivalent) LP problem, **to be treated as the original was:** the essence of the simplex method has just been found! The process can continue *iteratively*, until there is no variable leading to improvement in the objective function. In sum:

A. In the system of equations, find the identity matrix (immediate solution).
B. Search for an *entering* variable (or finish).
C. Consequently, find a *leaving* variable (if wrongly chosen, negative values will appear).

3 Transportation Problem and special cases

In the supply chain environment, several problems related to transportation and others apparently unrelated can be formulated and solved by the technique used for the typical *transportation problem*, frequently simply denoted by the initials TP. Besides the TP, we shall address: the (simple) *production scheduling*; the *transshipment problem*; and the *assignment problem* (AP). These problems can be solved by their own algorithms: the TP, the production scheduling and the transshipment, by the "stepping-stone" method; and the AP by the Hungarian method. As all these problems are particular cases of Linear Programming (LP), the problems will be presented and then formulated as LP problems. Indeed, with the current availability of high quality LP software, namely, the suggested IBM ILOG CPLEX (2015a, b, c), it looks unnecessary to go into the details of those other methods. Otherwise, a negative remark about the Hungarian method to solve the AP is that, although it is easy and elegant to be done by hand, it is quite hard to program, with no readily available source code in the literature.

The general goal is to "transport" (whatever that may be) goods to the customers at minimum global cost of transportation, according to the unit costs of transportation (certainly dependent on distance, etc.) from the sources to the destinations. The problems mentioned are dealt with in the following sections, mainly based on examples.

4 The Transportation Problem

The Transportation Problem (TP) arises from the need of programming the optimal distribution of a *single* product from given sources (supply) to given destinations (demand).

The product is available in m sources, with known quantities (also said *capacities*), a_i, $i = 1 \,..\, m$ (the two dots denoting a range, as used in various computer languages, avoiding the vague 1, …, m), and is needed in n destinations, with known quantities (or capacities), b_j, $j = 1 \,..\, n$, and will be sent directly from the sources to the destinations at unit costs, c_{ij}, all these values being the known data.

The objective is to find the quantities to be transported, x_{ij}, at minimum global cost, usually in a given time period, such as a week. The problem can thus be formulated according to the model of Eq. (10), where obviously the sum of the a's must equal that of the b's, without which the problem would be infeasible.

$$[\min] z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

Subject to

$$\sum_{j=1}^{n} x_{ij} = a_i \quad i = 1 \ldots m \tag{10}$$

$$\sum_{i=1}^{m} x_{ij} = b_j \quad j = 1 \ldots n$$

$$x_{ij} \geq 0 \quad \forall i, j$$

The physical (typically integer) units of x, a and b can be, say, kg (or m^3, bags, etc.), and c in \$/kg (with \$ a generic money unit, such as dollar, euro). The scheme in Fig. 1 makes the problem clear.

The model in Eq. (10) is, of course, in all its components (including the last one, of non-negativity of the variables), an instance of Linear Programming (LP). Our notation "[max]" ([min], [opt]) means that the maximum of *both* sides is required, and not that the maximum of z, the objective function, is equal to the right-hand side (otherwise even not yet known).

It is remarkable that the TP can be envisaged as an integer programming problem. The x's will always be, namely in the optimum, multiples of the greatest common divisor of the set of a's and b's. So, if these are integers (as is usual), the

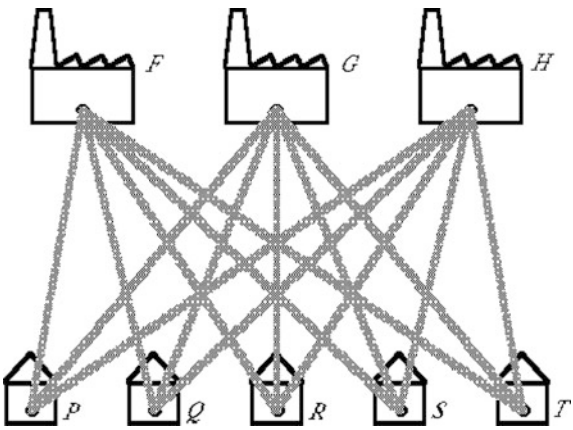

Fig. 1 Transportation problem: from 3 factories to 5 warehouses

x's will be integers; if, e.g., these are multiples of 7, so will they be, etc. If the problem is stated with "decimals", as 4.7, the x's will be multiples of 0.1, so, with appropriate multiplication by a suitable constant, the results will be integer.

Any problem having the above structure can be considered a TP, whatever may be the subject under analysis.

Example

A company, as in Fig. 1, has 3 production centres, factories F, G and H, in given locations (different or even coincident) with production capacities of 100, 120, and 120 ton (per day), respectively, of a certain (single) product with which it must supply 5 warehouses, P, Q, R, S, and T, needing 40, 50, 70, 90, and 90 ton (per day), respectively. The unit costs of transportation, the matrix **C**, are those in Table 1. Determine the most economical (cheapest) transportation plan, matrix **X**.

Resolution

Introduce the transportation matrix, **X**, in Eq. (11), the values of whose elements must be found. (Notice that z, in Eq. (10), does not result from a typical product of matrices!)

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} \end{bmatrix} \tag{11}$$

The problem could be solved by the adequate "stepping-stone" method (which is simple and very efficient), but its formulation leads directly to the LP in Eq. (12).

$$\begin{aligned} [\min] z = {} & c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{14}x_{14} + c_{15}x_{15} \\ & + c_{21}x_{21} + \cdots + c_{25}x_{25} \\ & + c_{31}x_{31} + \cdots + c_{35}x_{35} \end{aligned}$$

Table 1 Costs of transportation ($/ton) from the factories to the warehouses

	P	Q	R	S	T	
F	4	1	2	6	9	100
G	6	4	3	5	7	120
H	5	2	6	4	8	120
	40	50	70	90	90	

Subject to

$$\begin{aligned} x_{11}+x_{12}+x_{13}+x_{14}+x_{15} &= a_1 \\ x_{21}+x_{22}+x_{23}+x_{24}+x_{25} &= a_2 \\ x_{31}+x_{32}+x_{33}+x_{34}+x_{35} &= a_3 \end{aligned} \qquad (12)$$

$$\begin{aligned} x_{11}+x_{21}+x_{31} &= b_1 \\ x_{12}+x_{22}+x_{32} &= b_2 \\ &\ldots \\ x_{15}+x_{25}+x_{35} &= b_5 \end{aligned}$$

A TP has really $m + n - 1$ *independent* constraints (not $m + n$), as (any) one of the constraints shown above is superfluous (dependent). This is due to the sheer nature of the TP, in which total supply must equal total demand. If, as happens in several circumstances, supply and demand are not equal, one (no need for more) fictitious entity (source or destination) is introduced, as detailed below.

The fact that the current solvers can be embedded in commercial worksheets, means they natively accept the TP in tabular form and the constraints in Eq. (12) are readily available for solution. The solution is given in Table 2, with a minimum global cost of $z^* = 1400$ \$ (per day, the period considered). In this particular problem, it happens that there are two (i.e., multiple) solutions, the other differing in $x_{11} = 10$, $x_{12} = 50$, $x_{31} = 30$ and $x_{32} = 0$.

The TP is, naturally, "balanced", i.e., the total supply is equal to the total demand. As mentioned, in the cases where there is excess *supply*, the problem can be readily converted to a TP by creating (at cost 0) one *fictitious destination*; or if there is excess *demand* (insufficient supply), one *fictitious source*. So, product could, respectively, be left "at home" or, possibly, bought from some competitor to guarantee the supply to the customer.

Table 2 Quantities to be transported (ton) from the factories to the warehouses

	P	Q	R	S	T	
F	40	20	40			100
G			30		90	120
H		30		90		120
	40	50	70	90	90	

5 The Production Scheduling

The (simple) "production scheduling" problem will be presented through an example akin to many in the literature, such as Hillier and Lieberman (2006, pp. 330–331).

Example
See Table 3.

Resolution
The "transportation" in the production scheduling is not in space, but in time, between months in this example. Assuming the basic TP problem where backorders are not considered production of a certain month is not used to supply the previous month, thus the corresponding unit costs should be prohibited, making them "very large", say, M (the classical "big M"), infinity, or, for computing purposes, sufficiently large (in this problem, e.g., 100 will be enough). Using data in Tables 3 and 4 is obtained, by adding the storage costs and introducing a fictitious fifth month for balancing.

In order to define a sufficiently large M, try some "reasonably" large value, i.e., at least large compared to the other cost values in the problem. (The naïve choice of the *greatest* number in the computer is not valid, because of probable overflow.) If this value is effective in the solution (prohibiting the related x's), then it is a good choice, but, if it is not effective (too small), try a greater new value. If the value is "never" sufficiently large, then, the problem has no physical solution (is *impossible*), although it always has a mathematical one.

The solution has a (minimum) global cost of 82.7 $ with the production schedule given in Table 5.

Table 3 Production scheduling data for Manufacture Co.

Month	Scheduled installations	Max. production	Unit production cost	Unit storage cost
1	15	25	1.08	0.015
2	15	35	1.11	0.015
3	25	30	1.10	0.015
4	20	10	1.13	0.015

Table 4 TP-like data for the Northern Airplane Co. problem

Month	1	2	3	4	(5)	Supply
1	1080	1095	1110	1125	0	25
2	M	1110	1125	1140	0	35
3	M	M	1100	1115	0	30
4	M	M	M	1130	0	10
Demand	15	15	25	20	25	**(100)**

Table 5 Production schedule for the Northern Airplane Co. problem

Month	1	2	3	4	(5)	Supply
1	**15**	**10**	**5**	**0**	0	25
2	–	**5**	**0**	**0**	30	35
3	–	–	**20**	**10**	0	30
4	–	–	–	**10**	0	10
Demand	15	15	25	20	25	**(100)**

6 The Transshipment Problem and the Assignment Problem

The transshipment problem and the assignment problem (AP) can be considered problems reducible to TP's. The transshipment is typically treated like a TP, whereas the AP has the Hungarian algorithm, which is very efficient. Notwithstanding, this algorithm will not be presented, as the AP is a particular LP and is appropriately and easily solved by the usual LP software.

The method to reduce a transshipment problem to a TP is simply to consider that the transshipment points are supply points or demand points or both, by inserting them on the supply side or the demand side (or both). Upon inserting these transshipments as referred, each individual capacity must be "corrected" by adding to it the original global capacity.

The AP is a particular case of the TP, having a square cost matrix and being soluble by considering all the values of supply as 1 and all the values of demand also as 1.

Example, Transshipment

The example is represented in Fig. 2 and is akin to problems from the literature.

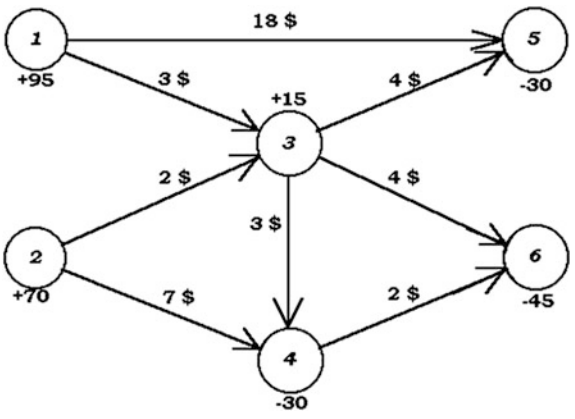

Fig. 2 Transshipment problem: sources (1 and 2), destinations (5 and 6), and transshipment points (3 and 4)

Resolution

In order to convert the transshipment to a TP, identify: (a) every pure supply point (producing only), usually labelled with a positive quantity, such as Point 1 with +95 units; (b) every pure demand point (receiving only), usually labelled with a negative quantity, such as Point 5 with −30 units; and (c) every mixed point (producing or receiving), labelled with a positive (if net producer) or negative (if net receiver) quantity, such as Point 3 with +15 units. Make the original TP balanced, which results here in a dummy destination (Point 7), and register the original capacity of the TP, Q (here $Q = 180$).

The cost matrix becomes the one in Table 6. (The number of times Q is inserted on the supply side and on the demand side is, of course, the same, thus maintaining the equilibrium necessary for a TP.) The solution is in Table 7.

So: from Point 1, 20 units go to Point 3, and 75 stay home; from Point 2, 70 units go to Point 3; from Point 3, 30 go to Point 4, etc.; and Point 4 just receives 30 (from Point 3), with its quantity (equal to Q) meaning it was not used as a transshipment point.

Example, Assignment

Suppose that n tasks are to be accomplished by n workers and the workers have the abilities for every task as given in Table 8.

Table 6 Cost matrix for the transshipment problem

	3	4	5	6	(7)	Supply
1	3	M	18	M	0	95
2	2	7	M	M	0	70
3	0	3	4	4	0	$15 + Q$
4	M	0	M	2	0	Q
Demand	Q	$30 + Q$	30	45	75	$(180 + 2Q)$

Table 7 Solution to the transshipment problem

	3	4	5	6	(7)	Supply
1	20	–	0	–	75	95
2	70	0	–	–	0	70
3	90	30	30	45	0	$15 + Q$
4	–	180	–	0	0	Q
Demand	Q	$30 + Q$	30	45	75	$(180 + 2Q)$

Table 8 Ability of each worker for each task

	1	*2*	*3*	*4*
1	15	16	14	14
2	14	14	13	15
3	13	15	13	14
4	15	16	14	14

Table 9 Assignments (solution)

	1	*2*	*3*	*4*
1	1	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	1	0	0

These "positive" abilities are preferably converted to "costs", replacing each element by, e.g., its difference to their maximum (here, 16). After that, solve the AP as a TP with each supply equal to 1 and each demand also equal to 1. The solution to this example is in Table 9.

So, Worker 1 does Task 1, Worker 2 does Task 4, etc. (i.e., 1-1, 2-4, 3-3, 4-2), at a minimum global cost of 5 cost units. This particular problem has multiple solutions, another being 1-2, 2-4, 3-3, 4-1.

As this problem has, obviously, always n elements of value 1 (the assignments) and the remaining $n^2 - n$ of value zero, its optimum solution is very *degenerate* if, as was done here, it is considered a TP (degenerate in relation to the $m + n - 1$ possible positive cells in a TP). This is an argument in favour of the Hungarian method, but the strength of that method is not significant when common software is used. (The algorithm of Jonker and Volgenant would be preferable, if the AP itself is under study.)

In the supply chain, several problems related to transportation can be formulated and solved by the technique used for the typical transportation problem (TP), with its own very efficient algorithm (stepping-stone): the (simple) production scheduling; the transshipment problem; and the assignment problem (AP). The AP can be solved by its own algorithm, but the availability of software for Linear Programming makes it practical to solve them as TP's, after convenient simple conversions.

7 Integer Programming by Branch-and-Bound

An example from Ecker and Kupferschmid (1988, p. 217 ff) is used, with modifications and a different, more systematic path to the solution. Using, for example, the simple software Lindo (2013) semi-manually, or CPLEX, the successive solutions can be obtained, to observe the Branch-and-Bound methodology. Of course, these software tools or any other adequate to the problem (possibly) use this technique without necessarily showing the path to the solution.

The problem, shown in (MILP.1), will be to maximize the objective function, z, subject to: three constraints, with the typical constraint of nonnegative independent variables; and being integers. It is this last constraint that makes this a problem of *integer programming* or *mixed integer programming* in the cases where some variables are continuous. Usually, this subject, which is related to Linear Programming, is denoted by MILP, "mixed integer linear programming".

$$\begin{aligned} [\max] z = & -3 \quad x_1 \quad +7 \quad x_2 \quad +12 \quad x_3 \\ & -3 \quad x_1 \quad +6 \quad x_2 \quad +8 \quad x_3 \quad \leq \quad 12 \\ \text{subject to} \quad & +6 \quad x_1 \quad -3 \quad x_2 \quad +7 \quad x_3 \quad \leq \quad 8 \\ & -6 \quad x_1 \quad +3 \quad x_2 \quad +3 \quad x_3 \quad \leq \quad 5 \\ & x_1, x_2, x_3 \text{ nonnegative integers} \end{aligned} \tag{MILP.1}$$

The resolution of a MILP inherits ***no ease*** from the Linear Programming proper. The typical way to solve a MILP is through the technique of *branch-and-bound*, as explained based on this example.

Begin by solving the MILP as a simple LP. If the solution happens to be integer—which is, of course, rare—, it is the optimum sought. The solution to the original problem, say, P0, is

$$X = [0 \quad 0.303 \quad 1.272] \tag{MILP.2}$$

$$z = 17.39$$

The solution, i.e., X, in Eq. (MILP.2) is not integer, so the problem has not been solved. Now, to solve it, let us introduce the B&B technique.

(a) Select any non-integer variable, x_k (prefer the "least integer", i.e., values closer to halves, and ignore draws).
(b) Replace the current problem by two new problems:
    (i) Current problem augmented with constraint $x_k \leq \lfloor x_k \rfloor$, and
    (ii) Current problem augmented with constraint $x_k \geq \lceil x_k \rceil$.
(c) Solve the two problems and repeat, if necessary.

The notation $\lfloor \rfloor$ indicates the 'floor' and $\lceil \rceil$ the 'ceiling' functions (the Iverson notation). So, the original problem will be successively branched: this will occur in an unpredictable way, giving rise to possibly many problems; and also, each time there is branching, the new level will be more difficult than the previous one (one more constraint). Remember that the difficulty (computational complexity) of an LP problem depends essentially on its number of constraints (not its number of variables), which leads to about 1.5 m iterations (some authors preferring 2–3 m), with m the number of constraints. (There are ways, beyond the context of this text, to attenuate this growing difficulty, such as solving dual problems. Also, if there are binary variables, branching leads to just fixing in the simple values of 0 or 1.)

In the example, branching will thus be around the variable x_2 = 0.303. The two new problems will be the original problem augmented with one of the two constraints, having now 4 constraints, giving, respectively,

P1	P2
Original problem	Original problem
$x_2 \leq 0$	$x_2 \geq 1$

The solutions to the problems are

$$\text{P1} \quad X = [0 \quad 0 \quad 1.143] \tag{MILP.3}$$

$$z = 13.7$$

and

$$\text{P2} \quad X = [0.667 \quad 1 \quad 1] \tag{MILP.4}$$

$$z = 17$$

In the process of choosing the sub-problem to branch on, it is impossible to predict the best choice. So, select the sub-problem with the best (i.e., most promising) value of the objective function, z. In the example, maybe the process will end up getting an integer solution with $z > z_1$—for instance, $z^* = 14$—, permitting to avoid to explore its branches, so P1 will not be chosen. This avoidance is the advantageous feature of the B&B.

Neither P1 or P2 problems have an integer solution. The more promising is P2, because it presents a greater z, so it is chosen, giving P3 and P4. Branching will be around $x_1 = 0.667$. The two new problems will be the current problem augmented with one of the constraints, having now 5 constraints, giving, respectively,

P3 (from P2)	P4 (from P2)
Original problem	Original problem
$x_2 \geq 1$	$x_2 \geq 1$
$x_1 \leq 0$	$x_1 \geq 1$

The solutions to the problems are

$$\text{P3} \quad X = 0 \quad 1 \quad 0.667 \tag{MILP.5}$$

with

$$z = 15$$

and

$$\text{P4} \quad X = 1 \quad 1.348485 \quad 0.863636 \qquad \text{(MILP.6)}$$

$$z = 16.8$$

The active problems are now P1 (z = 13.7), P3 (z = 15) and P4 (z = 16.8). Problem P2 (replaced by P3 and P4 has been *fathomed*, i.e., discarded after evaluation. So, P4 will be chosen to branch from, giving P5 $(x_2 \leq 1)$ and P6 $(x_2 \geq 2)$.

The B&B technique proceeds in this way, generating a sequence of solutions The path to the solution (apart from the change of 'min' to 'max of symmetrical') is not coincident with the one given in the example cited in Ecker and Kupferschmid (1988). (In the 3rd row of solutions, the rightmost, 4th, solution, "Infeasible", is possibly wrong, although without further influence.) The integer solution is z^* = 15, with X^* = [2; 3; 0], where of course the term "integer" relates to the values of X^*.

Acknowledgments The Authors thank the College of Technology and Management at the Portalegre Polytechnics Institute (ESTG/IPP) and Instituto Superior Técnico (IST). These works are partially developed at the Centre for Chemical Processes (CPQ/IST) with the support of FCT project PEst-OE/EQB/UI0088, and other developments at CERENA/IST with the support of FCT project UID/ECI/04028/2013.

References

Bronson, R., Naadimuthu, G. (2010). Schaum's outline of theory and problems of Operations Research. New York, NY, USA: McGraw-Hill.

Cipra, B. A. (2000). The best of the 20th century: Editors name Top 10 Algorithms. In J. G. Ecker & M. Kupferschmid (Eds.) (1988). *Introduction to operations research*. New York, NY, USA: Wiley. ISBN 0-471-63362-3. *SIAM News*, *33*(4).

CPLEX. (2012). IBM ILOG CPLEX optimizer.

Ecker, J. G., & Kupferschmid, M. (1988). *Introduction to operations research*. Chichester, UK: Wiley.

Hillier, F. S., & Lieberman, G. J. (2005). *Introduction to operations research*. New York, NY, USA: McGraw-Hill.

Hillier, F. S., & Lieberman, G. J. (2006). *Introduction to operations research*. New York, NY, USA: McGraw-Hill.

Hillier, F. S., & Lieberman, G. J. (2009). *Introduction to operations research*. New York, NY, USA: McGraw-Hill.

IBM ILOG CPLEX Optimization Studio OPL Language User's Manual. (2015a). © Copyright IBM Corp., 1987. http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.studio.help/pdf/opl_languser.pdf?lang=en

IBM ILOG CPLEX Optimization Studio CPLEX User's Manual. (2015b). © Copyright IBM Corporation 1987. http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.studio.help/pdf/usrcplex.pdf?lang=en

IBM ILOG CPLEX Optimization Studio CP Optimizer User's Manual. (2015c). © Copyright IBM Corporation 1987. http://www-01.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.studio.help/pdf/usrcpoptimizer.pdf?lang=en

Lindo. (2013). Lindo Systems, Inc., http://www.lindo.com/
Taha, H. (1992). *Operations research: An introduction*. New York, NY, USA: MacMillan Publishing Company.
Zionts, S. (1974). *Linear and integer programming*. Englewood Cliffs, NJ, USA: Prentice-Hall. ISBN 0-13-536763-8.