Annals of Operations Research 140, 163–188, 2005

On a Fix-and-Relax Framework for a Class of Project Scheduling Problems

LAUREANO F. ESCUDERO escudero@umh.es
Centro de Investigación-Operativa, Universidad Miguel Hernández, Ave. de la Universidad, 03202 Elche, Spain

JAVIER SALMERON* jsalmero@nps.edu
Operations Research Department, Naval Postgraduate School, Monterey, CA 93943, USA

Abstract. We consider the following problem: Given a set of projects to be executed along a multi-year time horizon, find a sequencing and scheduling feasible solution that optimizes a merit function. A feasible solution should satisfy availability of storable and non-storable resources with budget carrier and non-carrier periods, and precedence, exclusivity and implication relationships among the projects. This is an *NP-hard* problem. We present several *Fix-and-Relax* strategies which partition the set of binary variable into clusters in order to selectively explore the branch-and-bound tree. Computational performance is favorably compared with a state-of-the-art optimization engine over a set of real-life cases.

Keywords: multi-period project management, 0-1 programming, branch-and-bound selective search

Introduction

Scheduling problems arise in many practical circumstances when planning the utilization of resources for a better project management. Often, these problems are stated as optimization problems having the following form: given a set of projects to be executed along a time horizon subject to various constraints, find a feasible sequencing and scheduling solution that minimizes the value of a given objective function. Typical elements are: budget restrictions for carrier and non-carrier periods, limited resource availability, multi-period projects, subsets of projects with exclusivity and implication constraints, and precedence relationships in the execution of the projects, among others.

The class of project scheduling problems studied in this work does not belong to the type of problems known as Resource-Constrained Project Scheduling, see Pinedo (1995), Klein (2000) and Baptiste, Le Pape, and Nuijten (2001), among others, although they have some common structures. The problem of concern can be formulated as a 0-1 model and is NP-hard. There is a vast literature on its polyhedral analysis and on tight 0-1 formulations and facet defining inequalities identification, see e.g., Wolsey (1990, 1997, 2002), Sousa and Wolsey (1992), Constantino (1996), Miller, Nemhauser and Savelsbergh (2003) and Waterer et al. (2002), among others. On the other hand, efficient

*Corresponding author.

heuristic algorithms for general 0-1 models have been introduced; see Balas and Martin (1980) and Balas et al. (2000), among others.

However, it is unlikely that an algorithm is developed to solve large-scale instances of the problem up to optimality in affordable computing effort. We present different strategies for the Fix-and-Relax (F&R) algorithmic framework (Dillenberger et al., 1994) for obtaining a satisfactory solution to our 0-1 model, such that the nodes of the branch-and-bound tree are selectively explored. The variables are grouped into clusters for branching selection, according to different criteria, and the clusters are ordered based on given priority rules. See also Kularajan et al. (2000) for the crew scheduling problem.

Our particular application is drawn from the electric power sector in Spain. The electric distribution grid (medium- and low-voltage assets) is responsible for carrying electric power from transmission substations to loads (i.e., to the final customers). This is an evolving system that requires continuous upgrade to satisfy customers demand while complying with reliability standards. Investments (that we call "projects") include network expansion (e.g., building a new corridor or a new distribution substation), upgrades (e.g., adapting a substation to remote control systems or improving protective equipment), maintenance of all the existing assets, and other regulatory compliances. The problem can be succinctly stated as finding the combination of projects (among thousands) and their scheduling (over a pre-specified time horizon), with limited resources, in order to maximize a given merit function. This function is basically based on two aspects: revenue and improvement on quality of service. It is typical that these two objectives conflict with each other, which will be discussed in detail in the section devoted to report our computational results. The required investment to undertake any of these projects ranges from a few thousand dollars (for minor equipment maintenance) to the millions of dollars (for the construction of new substations). The decision of which projects are undertaken is centralized by the distribution utility, but it is worth noting that geographical considerations (such as "invest more where it is more needed") and other restrictions by type of project may apply as well.

Our motivation to explore F&R originates in the difficulty to solve realistic large-scale instances of this problem, whose formulation is described later in the paper. Ruled-based heuristics hardly produce satisfactory results for this problem. One reason is the multiple types of constraints in the problem, which makes difficult to guarantee feasibility in the "neighborhood" selection. An additional difficulty is determining valid criteria to account for projects with negative (or zero) return on investment, yet these projects might be needed to ensure compliance with regulation and/or to contribute to improve the quality of service. The problem, however, lends itself to apply F&R which is implemented under a variety of strategies. The approach cannot guarantee the solution's optimality but the results that we have obtained while testing it on a set of real-life cases produce quasi-optimal solutions in reasonable computing time, where a state-of-the-art optimization engine takes much more computing time and, frequently, fails to give a solution.

The rest of the paper is organized as follows. Section 1 states the project scheduling problem to address. Section 2 presents the problem's elements and introduces the 0-1 model. Section 3 is devoted to present the F&R algorithm as well as the strategies to use for variables' partitioning. Section 4 reports the computational experience. Section 5 concludes. Finally, an Appendix presents some information about the data that have been used for the test cases.

1. Problem statement

Let a set of *projects* to be executed along a given time horizon. The projects have time windows for their execution, if any. An execution time interval is given by a number of (consecutive) time periods, so-called *legs*, without preemption. Let us term *macro project* to a subset of projects such that if one project from the subset is executed, then, all projects in the same subset must been executed (probably, in a different time interval). Some macro projects are distributed in the so-called *missions*, such that it can be required that a given number of macro-projects must be selected to accomplish a given mission. The projects that belong to a macro-project can be called *conditional mandatory* projects. A very frequent case is a mission with alternative macro projects in the sense that one and only one of these subsets of projects can be executed. It is worth noting that the projects can only belong to one macro-project, if any, and the macro-projects can only belong to one mission, if any. Figure 1 shows a case where the macro-project {P1, P2, P3} does not belong to any mission, and the macro projects {P4, P5, P6}, {P7} and {P8, P9} belong to the same mission. For example, if project P4 is to be executed then the projects P5 and P6 must also be executed, and none of the projects P7, P8 or P9 can be executed in case of alternative macro-projects. On the other hand, if project P1 is to be executed then the projects P2 and P3 must also be executed. Additionally, there are so-called *non-mandatory* projects. This class of projects will only be executed if their schedule satisfies given constraints and contributes to the objective function value improvement, or if they are essential to satisfy any other requirement. Non-mandatory projects do not belong to any macro project. Another class of projects is the so-called *unconditional mandatory* projects. This class of projects does not belong to any macro-project or mission, but they have to be executed.

Some types of *precedence* relationships in the execution of the projects are also considered. We represent them by an acyclic directed graph where the nodes are associated

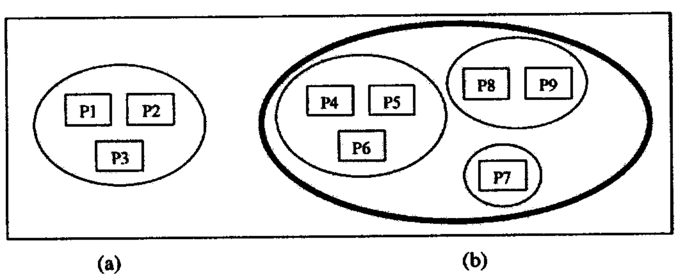

Figure 1. Macro-project (figure 1(a)) and Mission (figure 1(b)).

with the projects and the arcs refer to the existence of precedences between the projects represented by the from-nodes and the to-nodes. Two types of precedences are considered. The first type forces a *minimum time lag* (i.e., number of time periods) between the execution's starting of any two projects. The second type of precedence forces a *maximum time lag* between the starting of both executions, in case that the second project in the pair starts after the first one.

As stated above, a set of resources is required to undertake the selected projects. Two types of resources are allowed, namely resources that can be stored from one time period to the next one, and resources that cannot be stored (i.e., the unused labor hours at the end of a period cannot be used in subsequent periods). There is a maximum availability for the *non-storable resources.* On the other hand, a given amount can be made available for the *storable resources* at (the beginning of) any time period. Some resources can require the consumption of a given minimum amount per period.

It is assumed that each project has assigned a *dedicated resource* (e.g., management team, working station) for its execution. Let us say that the projects with the same dedicated resource belong to the same *type*, such that the simultaneous execution of the projects is not allowed. A *setup* in a dedicated resource is required between the consecutive executions of two projects.

The projects can be assigned to *budget categories*. A category is a set of projects that are classified under a specific *attribute*. An attribute can be associated with a physical circumstance, such as the geographical zone or environmental location where the project can impact. It is also frequent to consider attributes related to the origin of the project proposal. In this case, the categories can be the different technical, administrative and management levels that can be influenced by the accomplishment of the projects. In addition, there is a *total budget* for each period that applies to all the projects under study.

There is a maximum availability for the total budget and, independently, for the budget categories for the so-called non-*carrier periods* (i.e., time periods whose budget surplus at the end of the period cannot be used in the next periods). Additionally, a given amount can be made available in both types of budgets for the *carrier periods*. In any case, a minimum budget can be required to be used for certain categories of projects (e.g., maintenance works, geographical zones, etc.) as well as for the total budget.

A set of *executable categories* can also be defined, such that lower and upper bounds on the number of projects to execute must be satisfied for each category. This requirement can be easily accommodated by using the resource object model.

The goal consists of determining the projects' selection and scheduling, such that the given constraints are satisfied and the chosen objective function is optimized. Let the following two types of functions: *minsum* functions (e.g., the total project execution cost to minimize, the total benefit to maximize, etc.) and *minmax* functions (e.g., the makespan to minimize, etc.). In case of several functions, one of them is optimized and the others ones (so-called *goals*) are appended to the constraint system for forcing to reach certain levels.

Application cases of this class of project scheduling problems can be found in energy generators maintenance planning (see Escudero, 1981), other production units maintenance scheduling (see Escudero, 1982) and production sequencing and scheduling (see Escudero, 1988; Wolsey, 1990, 1997, and Sousa and Wolsey, 1992, among others). Escudero, Gómez, and Salmerón (1997) study an investment selection and scheduling problem in an electric distribution network. In any case, all of these works only consider the deterministic version of the problem. See in Alonso-Ayuso et al. (2003) a modeling framework for this problem under uncertainty in some parameters, such as resource availability and consumption and the objective function coefficients.

2. Mathematical formulation

The 0-1 model requires the following sets, parameters and variables.

2.1. Sets

T,	set of time periods.
J,	set of candidate projects.
$T_j \subseteq T$,	subset of feasible time periods to begin the execution of project j, for $j \in J$.
R,	set of resources, for $r \in R$.
$R^1 \subseteq R$,	subset of non-storable resources.
$R^2 \subseteq R$,	subset of storable resources. Note: $R = R^1 \cup R^2$, $R^1 \cap R^2 = \emptyset$.
A,	set of attributes.
C^a,	set of categories of attribute $a \in A$.
$J^{ac} \subseteq J$,	subset of projects in category c of attribute a, for $c \in C^a$, $a \in A$.
I,	set of project types.
$J^i \subseteq J$,	subset of projects that belong to type i, for $i \in I$, such that $J = \bigcup_{i \in I} J^i$ and $J^i \cap J^{i'} = \emptyset, \forall i, i' \in I, i \neq i'$.
L,	set of macro-projects.
$J_0 \subseteq J$,	subset of non-mandatory projects.
$J_l \subseteq J$,	subset of projects in macro-project l, for $l \in L$ such that $J_l \cap J_{l'} = \emptyset, \forall l, l' \in L, l \neq l'$. *Note* 1: $J_0 \cup \bigcup_{l \in L} J_l \subseteq J$. *Note* 2: The set of unconditional mandatory projects is $J - J_0 - \bigcup_{l \in L} J_l$
M,	set of missions.
$L_m \subseteq L$,	subset of macro-projects that belong to mission m, for $m \in M$, such that $L_m \cap L_{m'} = \emptyset, \forall m, m' \in M, m \neq m'$.
$\underline{P} \subseteq J \times J$,	subset of project pairs with a minimum time lag precedence relationship (type 1), for $(j, j') \in \underline{P}$.
$\overline{P} \subseteq J \times J$,	subset of project pairs with a maximum time lag precedence relationship (type 2), for $(j, j') \in \overline{P}$.
G,	set of goals.

2.2. Parameters

d_j, duration (i.e., number of consecutive legs) of project j, for $j \in J$. Note: d_j and T_j are assumed to be non-contradictory, by considering that the projects are completed within the time horizon.

$d(t, t')$, index of the leg to be performed in time period t for any project that started in time period t', for $t' \leq t$; It is calculated as $d(t, t') = t - t' + 1$

p_{jd}, investment (cost) required to undertake the d th leg of project j, for $1 \leq d \leq d_j$, $j \in J$.

q^r_{jd}, amount of resource r that is needed to undertake the d th leg of project j, for $1 \leq d \leq d_j$, $j \in J$, $r \in R$.

f_t, a 0-1 budget carrier indicator such that its value 1 means that period t is a carrier of the total budget surplus at (the end of) period t and, then, it can be used at time period $t + 1$; otherwise, its value is 0, for $t \in T - \{|T|\}$. For example, if the periods are quarters and the budget can be used within the same fiscal year, then the settings for two consecutive fiscal years (eight-quarter periods) would be as follows: $f_i = 1$, for $i = 1, 2, 3, 5, 6, 7$, and $f_4 = 0$. Note: In a broader context, this indicator can account for the carryover fraction of the budget, $0 \leq f_t \leq 1$.

f^{ac}_t, a 0-1 budget carrier indicator for category c of attribute a in time period t, with the same meaning as above, for $c \in C^a$, $a \in A$, $t \in T - \{|T|\}$.

$\underline{p}^0_t$, minimum global budget to be used in time period t by all the projects, for $t \in T$.

$\overline{p}^0_t$, maximum available global budget (for $f_t = 0$) and input available amount (for $f_t > 0$) to be used in time period t by all the projects, for $t \in T$.

$\underline{p}^{ac}_t$, minimum budget to be used in time period t by all the projects in category c of attribute a, for $c \in C^a$, $a \in A$, $t \in T$.

$\overline{p}^{ac}_t$, maximum available budget (for $f^{ac}_t = 0$) and input available amount (for $f^{ac}_t > 0$) to be used in time period t by all the projects in category c of attribute a, for $c \in C^a$, $a \in A$, $t \in T$.

$\underline{q}^r_t$, minimum amount of resource r to be used in time period t, for $r \in R$, $t \in T$.

$\overline{q}^r_t$, maximum available amount of resource r and input available amount in time period t, for $r \in R^1$ and $r \in R^2$, respectively, for $t \in T$.

$\underline{n}_m$, $\overline{n_m}$, minimum and maximum number of macro-projects that can be selected for mission m, respectively, for $m \in M$. Note $0 \leq \underline{n}_m \leq \overline{n_m}$. Typical cases: $\underline{n}_m = \overline{n_m} = 1$ (i.e., only one option can and must be selected for the mission) and $\underline{n}_m = 0$, $\overline{n_m} = 1$ (i.e., select one option or leave the mission undone).

$\underline{t}(j, j')$, minimum time-lag between the execution starting of projects j and j' (i.e., project j comes first and project j' cannot start until, at least, $\underline{t}(j, j')$ periods later), for $(j, j') \in \underline{P}$.

$\overline{t}(j, j')$, maximum time-lag between the execution starting of projects j and j', in case project j' does not start before project j, for $(j, j') \in \overline{P}$.

s^i, setup time (i.e. number of time periods) between the ending and the starting of the execution of two consecutive projects of type i, for $i \in I$.

b_{jt}, benefit for starting the execution of project j in time period t, for $t \in T_j, j \in J$.

$\underline{b}^g_t$, floor to be achieved for goal g in time period t, for $t \in T$, $g \in G$.

$\underline{b}^g$, total floor to be achieved for goal g in the time horizon, for $g \in G$.

w^g_{jt}, contribution of project j to reaching the floor in goal g if it starts its execution in time period t, for $t \in T_j, j \in J, g \in G$.

2.3. *Variables*

x_{jt}, 0-1 variable such that its value is 1 if project j starts its execution in time period t and, otherwise, its value is 0, for $t \in T_j$, $j \in J$.

y_l, 0-1 variable such that its value is 1 if macro-project l is performed (i.e., all projects $j \in J_l$ are executed) and, otherwise, its value is 0 (i.e., none of the projects in the set are executed), for $l \in L$.

S^0_t, total budget surplus at (the end of) time period t, for $t \in T$.

S^{ac}_t, budget surplus for category c of attribute a at (the end of) time period t, for $c \in C^a$, $a \in A$, $t \in T$.

H^r_t, surplus of storable resource r at (the end of) time period t, for $r \in R^2$, $t \in T$.

Note: A computationally oriented version of the model does not require the explicit representation of the S- and H-variables. See below the constraint system (2.21)–(2.23).

2.4. *The 0-1 model*

The mixed 0-1 model, where the S- and H- variables appear explicitly in the formulation, has the following objective and constraint system:

Objective: To determine the projects' selection and scheduling to maximize the global benefit along the time horizon:

$$\max_{\mathbf{x},\mathbf{y},\mathbf{S},\mathbf{H}} \sum_{j \in J} \sum_{t \in T_j} b_{jt} x_{jt} \tag{1}$$

subject to the constraints (3)–(20) below.

Remark. We could use other objectives such as minimizing the makespan:

$$\min\{\max\{(t + d_j - 1)x_{jt} \mid t \in T_j, j \in J\}\} \tag{2}$$

Constraint system:

$$\sum_{t\in T_j} x_{jt} \leq 1, \quad \forall j \in J_0 \tag{3}$$

$$\sum_{t\in T_j} x_{jt} = 1, \quad \forall j \in J - J_0 - \bigcup_{l\in L} J_l \tag{4}$$

$$\underline{n}_m \leq \sum_{l\in L_m} y_l \leq \overline{n_m}, \quad \forall m \in M \tag{5}$$

$$\sum_{t\in T_j} x_{jt} = y_l, \quad \forall j \in J_l,\ l \in L \tag{6}$$

$$\sum_{j\in J^i} \sum_{t'\in T^1(j,i,t)} x_{jt'} \leq 1, \quad \forall i \in I, t \in T \tag{7}$$

where $T^1(j,i,t) = \{t' \in T_j \mid t - d_j - s^i < t' \leq t\}$

$$\sum_{t'\in T^2(j,j',t)} x_{jt'} \geq \sum_{t'\in T^3(j,j',t)} x_{j't'}, \quad \forall t \in T_{j'},\ (j,j') \in \underline{P} \tag{8}$$

where $T^2(j,j',t) = \{t' \in T_j \mid t' \leq t - \underline{t}(j,j')\}$ and $T^3(j,j',t) = \{t' \in T_{j'} \mid \underline{t}(j,j') < t' \leq t\}$

$$\sum_{t'\in T^4(j,t)} x_{jt'} \leq \sum_{t'\in T^5(j,j',t)} x_{j't'}, \quad \forall t \in T_j : t < \mid T \mid - \overline{t}(j,j'),\ (j,j') \in \overline{P} \tag{9}$$

where $T^4(j,t) = \{t' \in T_j \mid t' \leq t\}$ and $T^5(j,j',t) = \{t' \in T_{j'} \mid t' \leq t + \overline{t}(j,j')\}$

$$\underline{p}_t^0 \leq \sum_{j\in J} \sum_{t'\in T^6(j,t)} p_{j,d(t,t')} x_{jt'}, \quad \forall t \in T \tag{10}$$

where $T^6(j,t) = \{t' \in T_j \mid t - d_j < t' \leq t\}$

$$\sum_{j\in J} \sum_{t'\in T^6(j,t)} p_{j,d(t,t')} x_{jt'} + S_t^0 = f_{t-1} S_{t-1}^0 + \overline{p}_t^0, \quad \forall t \in T \tag{11}$$

where $S_0^0 = 0$

$$\underline{p}_t^{ac} \leq \sum_{j\in J^{ac}} \sum_{t'\in T^6(j,t)} p_{j,d(t,t')} x_{jt'}, \quad \forall c \in C^a, a \in A, t \in T \tag{12}$$

$$\sum_{j\in J^{ac}} \sum_{t'\in T^6(j,t)} p_{j,d(t,t')} x_{jt'} + S_t^{ac} = f_{t-1} S_{t-1}^{ac} + \overline{p}_t^{ac}, \quad \forall c \in C^a, a \in A, t \in T \tag{13}$$

where $S_0^{ac} = 0$

$$\underline{q}_t^r \leq \sum_{j\in J} \sum_{t'\in T^6(j,t)} q_{j,d(t,t')}^r x_{jt'}, \quad \forall r \in R, t \in T \tag{14}$$

$$\sum_{j \in J} \sum_{t' \in T^6(j,t)} q^r_{j,d(t,t')} x_{jt'} \leq \overline{q}^r_t, \quad \forall r \in R^1, t \in T \tag{15}$$

$$\sum_{j \in J} \sum_{t' \in T^6(j,t)} q^r_{j,d(t,t')} x_{jt'} + H^r_t = H^r_{t-1} + \overline{q}^r_t, \quad \forall r \in R^2, t \in T \tag{16}$$

where $H^r_0 = 0$

$$\sum_{j \in J | t \in T_j} w^g_{jt} x_{jt} \geq \underline{b}^g_t, \quad \forall g \in G, t \in T \tag{17}$$

$$\sum_{j \in J} \sum_{t \in T_j} w^g_{jt} x_{jt} \geq \underline{b}^g, \quad \forall g \in G \tag{18}$$

$$x_{jt} \in \{0,1\}, \quad \forall t \in T_j, j \in J; y_l \in \{0,1\}, \forall l \in L \tag{19}$$

$$S^0_t \geq 0, \quad \forall t \in T; S^{ac}_t \geq 0, \forall c \in C^a, a \in A, t \in T; H^r_t \geq 0, \forall r \in R^2, t \in T \tag{20}$$

Constraints (3) allow the execution of the non-mandatory projects. Constraints (4) force the execution of the unconditional mandatory projects. In any case, only one starting time period is allowed for each project.

Constraints (5) force that the number of macro-projects to be selected for undertaking each mission satisfies the given lower and upper bounds. See that the mission is non-mandatory for a zero lower bound in the number of macro-projects to undertake.

Constraints (6) force the execution of the projects that belong to selected macro-projects, and prevents the execution of the projects that belong to the macro-projects that have not been selected. Notice that a macro-project could not belong to any mission. Escudero (1988) studies the constraints (5) and (6) for the selection of one and only one macro-project per mission, in order to tightening the model in conjunction with the knapsack constraints presented below.

Constraints (7) avoid the simultaneous execution of more than one project per type i, so that a given setup time s^i is required. This is accomplished by limiting (for a given period t) the number of projects $j \in J^i$ during the periods $t' \in T^1(j,i,t) = \{t' \in T_j \mid t - d_j - s^i < t' \leq t\}$, where the duration d_j is included to avoid project overlapping. Sousa and Wolsey (1992), Constantino (1996) and Wolsey (1997), among others study the polyhedron given by the constraints (3) and (7) for tightening the LP formulation. Akker, van den, Hurkens and Savelsbergh (2000) study the same time-indexed formulation by using the Dantzig-Wolfe Decomposition for column generation in large-scale instances. For conciseness, we have used similar structures to T^1 in sets $T^2, \ldots, T^6$ below.

Constraints (8) and (9) ensure that the precedence relationships types 1 and 2 are not violated, respectively. For example, the sets $T^2(j,j',t) = \{t' \in T_j \mid t' \leq t - \underline{t}(j,j')\}$ and $T^3(j,j',t) = \{t' \in T_{j'} \mid \underline{t}(j,j') < t' \leq t\}$ for any two projects $(j,j') \in \underline{P}$ with a minimum time lag precedence relationship, serve in (8) to bound the set of time periods in which project j can be carried out should project j' begin its execution in period t. It can be shown that the constraint systems (8) and (9) are stronger than the more "natural" systems where the sets $T^3(j, j', t)$ and $T^4(j, t)$ are replaced by the variables $x_{j't}$ and x_{jt}, respectively, see Wolsey (1997).

As an illustration of the constraint system (8), let two projects, say, j and j' with the following time relationship: $(j, j') \in \underline{P}$, $T_j = \{1, 2, 3, 4\}$, $T_{j'} = \{4, 5, 6\}$ and $\underline{t}(j, j') = 2$. The associated constraint system is as follows:

$$x_{j1} + x_{j2} \geq x_{j'4}$$
$$x_{j1} + x_{j2} + x_{j3} \geq x_{j'4} + x_{j'5}$$
$$x_{j1} + x_{j2} + x_{j3} + x_{j4} \geq x_{j'4} + x_{j'5} + x_{j'6}$$

Constraints (10) and (12) force that the lower bounds on the total budget and the category budgets to be used in each time period are satisfied, respectively. Constraints (14) force the related lower bounds for the resource volume to use. These three sets of constraints use the auxiliary set $T^6(j, t) = \{t' \in T_j \mid t - d_j < t' \leq t\}$ which is used for accounting purposes: In period t, we have budget and resource consumption not only by those projects starting in t, but also by those that started before t and are still in progress, i.e., those projects $\{j\}$ whose starting time period is $t - d_j + 1, t - d_j + 2, \ldots, t$.

Constraints (11) and (13) represent the balance equations for the total budget and the category budgets along the time horizon, respectively. Constraints (16) represent the related balance equations for the storable resources. It is worth noting that the use of the carryover 0-1 f-indicator allows the transfer of budget surplus between consecutive time periods.

As an illustration of the constraint system (11), let two projects $j, j' \in J$, $T_j = \{1, 2, 3, 4\}$, $T_{j'} = \{4, 5, 6\}$, $d_j = 2$, $d_{j'} = 3$, $f_t = 1, \forall t \in T\backslash\{4\}$, and $f_4 = 0$. The constraint system for $t = 4, 5, 6$ is as follows:

$$p_{j2}x_{j3} + P_{j1}x_{j4} + p_{j'1}x_{j'4} - S_3^0 + S_4^0 = \overline{p_4}^0$$
$$p_{j2}x_{j4} + p_{j'2}x_{j'4} + p_{j'1}x_{j'5} + S_5^0 = \overline{p_5}^0$$
$$p_{j'3}x_{j'4} + p_{j'2}x_{j'5} + p_{j'1}x_{j'6} - S_5^0 + S_6^0 = \overline{p_6}^0$$

The knapsack constraints (15) ensure that the availability of the non-storable resources is not violated at any time period.

Constraints (17) and (18) ensure that the given levels for each single goal and the global goal are satisfied, respectively.

Note that, since the non-negative S -and H-variables only appear in (11), (13) and (16), it is worthwhile substituting them via simple recursion. As a result, it is not difficult to see that these constraints together with the non-negativity bounds (20) can be replaced by the knapsack constraints (21), (22) and (23), respectively.

$$\sum_{\substack{t' \in T| \\ t' \leq t}} \tilde{f}_{t'}(t) \sum_{j \in J} \sum_{t'' \in T^6(j,t')} p_{jd(t',t'')} x_{jt''} \leq \sum_{\substack{t' \in T| \\ t' \leq t}} \tilde{f}_{t'}(t) \overline{p}_{t'}^0, \quad \forall t \in T \tag{21}$$

where

$$\tilde{f}_{t'}(t) = \begin{cases} \prod_{t' \leq t'' < t} f_{t''}, & \text{if } t' = 1, 2, \ldots, t-1 \\ 1, & \text{if } t' = t \end{cases}$$

$$\sum_{\substack{t' \in T| \\ t' \leq t}} \tilde{f}_{t'}(t) \sum_{j \in J^{ac}} \sum_{t'' \in T^6(t,t')} p_{jd(t',t'')} x_{jt''} \leq \sum_{\substack{t' \in T| \\ t' \leq t}} \tilde{f}_{t'}(t) \overline{p}_{t'}^{ac}, \quad \forall c \in C^a, a \in A, t \in T \tag{22}$$

$$\sum_{\substack{t' \in T| \\ t' \leq t}} \sum_{j \in J} \sum_{t'' \in T^6(j,t')} q^r_{jd(t',t'')} x_{jt''} \leq \sum_{\substack{t' \in T| \\ t' \leq t}} \overline{q}^r_{t'}, \quad \forall r \in R^2, t \in T \tag{23}$$

For example, if budget carryover is not allowed ($f_t = 0, \forall t \in T$) then, we have: $\tilde{f}_1(t) = \cdots = \tilde{f}_{t-1}(t) = 0$; $\tilde{f}_t(t) = 1$. Thus, (21) simply states that $\sum_{j \in J} \sum_{t'' \in T^6_{jt}} p_{jd(t,t'')} x_{jt''} \leq \overline{p}_t^0, \forall t \in T$, as expected, since $\overline{p}_t^0$ is the only budget available in time period t. However, if we allow to carryover any surplus between the periods $t = 1$ and $t = 2$, i.e., $f_1 = 1$, then we would have $\tilde{f}_1(2) = 1$ and $\tilde{f}_2(2) = 1$ for period $t = 2$. In this case, constraint (21) becomes $\sum_{j \in J} \sum_{t'' \in T^6(j,1)} p_{jd(1,t'')} x_{jt''} + \sum_{j \in J} \sum_{t'' \in T^6(j,2)} p_{jd(2,t'')} x_{jt''} \leq \overline{p}_1^0 + \overline{p}_2^0$, which indicates that the total expenses in the first two periods cannot exceed the total budget in those periods.

3. Fix-and-relax methodology

3.1. *Fix and relax: Basic algorithm*

We first review the Fix-and-Relax (F&R) methodology introduced by Dillenberger et al. (1994). Let us consider the following pure 0-1 model (**IP**):

$$\mathbf{IP} : \max_z \{f(z) : z \in Z \cap \{0, 1\}^n\}, \tag{24}$$

where z is an n-dimensional vector, Z is a polytope in $\Re^n$ and f is a real-valued convex function over Z. The components of z are denoted $z_1, \ldots, z_n$. Let $V = \{1, 2, \ldots, n\}$ be the set of decision variable indices for the model **IP**. As it is well known, a Branch and Bound (B&B) scheme to solve **IP** eventually becomes inefficient (as n increases) because of the exponential growth in the number of nodes to explore. From a practical point of view, sometimes it is also difficult to find a feasible solution. F&R is a general-purpose methodology that alleviates this difficulty by solving **IP** in a number of steps, each of which involves a subproblem of smaller complexity than the original **IP**.

Let $V_1, \ldots, V_k$ be a direct partition of the set V, that is, $V_i \subseteq V$, $\forall i = 1, \ldots, k$, $V = \bigcup_{i=1}^{k} V_i$, and $V_i \cap V_{i'} = \emptyset$, $\forall i, i' = 1, \ldots, k \mid i \neq i'$. The cardinality of each V_i is denoted $|V_i| = n_i$, therefore $n = \sum_{i=1,\ldots,k} n_i$. Problem **IP** can be rewritten as:

$$\begin{aligned}\mathbf{IP}: &\max_{z \in Z} f(z) \\ &\text{s.t. } z_j \in \{0,1\}, \quad \forall j \in V_i, i = 1, \ldots, k\end{aligned} \tag{25}$$

The F&R framework requires to solve a sequence of k mixed-0-1 subproblems (hereafter *stages*) denoted $\mathbf{IP}^r$, for $r = 1, \ldots, k$, which are defined as follows:

$$\begin{aligned}\mathbf{IP}^r: &\max_{z \in Z} f(z) \\ \text{s.t.} &\begin{cases} z_j = \hat{z}_j, \forall j \in V_i, i = 1, \ldots, r-1 \text{ (if } r > 1) \\ z_j \in \{0,1\}, \forall j \in V_r \\ z_j \in [0,1], \forall j \in V_i, i = r+1, \ldots, k \text{ (if } r < k) \end{cases}\end{aligned} \tag{26}$$

where the values $\hat{z}_j$ for $j \in V_i$, $i = 1, \ldots, r-1$ in stage $r > 1$ are retrieved from the solution to problems $\mathbf{IP}^1, \ldots, \mathbf{IP}^{r-1}$, respectively.

Since only a reduced subset of (non-fixed) 0-1 variables are kept integer at each stage r, $\mathbf{IP}^r$ can be solved with relative efficiency. We next develop several F&R-based schemes.

Let $V^*(P)$ denote the optimal objective function value for a generic problem **P** in the argument, and let $\underline{V}$(P) and $\bar{V}$(P) denote a lower bound and an upper bound on that solution, respectively. A basic implementation of F&R is as follows:

BFRA: Basic Fix-and-Relax algorithm

Input: Partition $V_1, \ldots, V_k$ for a given number of stages $k \geq 1$
Step 1: Set $r = 1$ and solve $\mathbf{IP}^1$.
If $\mathbf{IP}^1$ is infeasible, STOP: "Problem **IP** is infeasible".
Otherwise, set $\bar{V}(\mathbf{IP}) = V^*(\mathbf{IP}^1)$.
Step 2: If $r = k$, set $\underline{V}(\mathbf{IP}) = V^*(\mathbf{IP}^k)$ and STOP: "Problem **IP** is feasible".
Otherwise, increase r by 1.
Step 3: Solve $\mathbf{IP}^r$.
If $\mathbf{IP}^r$ is infeasible, STOP: "Problem **IP** status is unknown".
Otherwise, go back to Step 2
Output: **IP** status ("Infeasible", "Feasible" or "Unknown"). If status is "Feasible", the best lower and upper bounds that have been found for the optimal solution are $\underline{V}(\mathbf{IP})$ and $\bar{V}(\mathbf{IP})$, respectively.

As indicated in Step 3, BFRA has the potential to fail. This may occur if $\mathbf{IP}^1$ is feasible but, at some stage $r > 1$, the associated problem $\mathbf{IP}^r$ becomes infeasible. In this situation, BFRA is unable to recognize which of the following cases provokes the infeasibility:

(a) Problem **IP** (and consequently $\mathbf{IP}^r$) is integer-infeasible; see that this can only happen if the LP relaxation of **IP** is feasible but **IP** is itself infeasible, since $\mathbf{IP}^1$is feasible by hypothesis.

(b) Problem **IP** is integer-feasible, but having fixed z_j to $\hat{z}_j$ (which is an estimate of the true optimal value) for $j \in V_i$, $i = 1, \ldots, r-1$, makes $\mathbf{IP}^r$ infeasible.

We might also be interested in obtaining a solution that guarantees a given optimality gap. Unfortunately, BFRA yields a gap, $(\bar{V}(\mathbf{IP}) - \underline{V}(\mathbf{IP}))/\underline{V}(\mathbf{IP})$, without any guarantee to satisfy the desired optimality tolerance.

In the following section we present an algorithm that avoids these inconveniences.

3.2. *Enhanced fix and relax scheme*

Based on the same ideas as BFRA, we present an enhanced F&R algorithm that detects problem infeasibility and guarantees a predefined optimality gap. The improved scheme has the potential to "step back" if any infeasibility occurs, or when the gap exceeds the allowed tolerance. A step back groups two or more of the original stages into a single one, which in turn provides a closer approach to the original **IP**. An enhanced version of F&R is as follows.

EFRA: Enhanced Fix-and-Relax algorithm

Input: Partition $V_1, \ldots, V_k$, for a given number of stages $k \geq 1$. Gap tolerance, $\varepsilon > 0$.
Set $r = 1$.
Step 1: Solve $\mathbf{IP}^1$.
If $\mathbf{IP}^1$ is infeasible, STOP: "Problem **IP** is infeasible".
Otherwise, set $\bar{V}(IP) = V^*(\mathbf{IP}^1)$.
Step 2: If $r = k$, set $\underline{V}(\mathbf{IP}) = V^*(\mathbf{IP}^k)$ and STOP: "Problem **IP** is feasible".
Otherwise, increase r by 1.
Step 3: Solve $\mathbf{IP}^r$.
If $\mathbf{IP}^r$ is feasible and $(\bar{V}(\mathbf{IP}) - \underline{V}(\mathbf{IP}))/\underline{V}(\mathbf{IP}) \leq \varepsilon$, go to Step 2.
Step 4: (Backwards grouping step): Redefine the partition structure:

$$\begin{cases} V_{r-1} \leftarrow V_{r-1} \cup V_r \\ V_i \leftarrow V_{i+1},\ \forall i = r, \ldots, k-1 \\ k \leftarrow k-1 \end{cases}$$

Decrease r by 1. If $r = 1$ go back to Step 1.
Otherwise, go back to Step 3

Output: **IP** status ("Infeasible" or "Feasible"). If status is "Feasible", the best found lower and upper bounds are $\underline{V}$(**IP**) and $\bar{V}$(**IP**), respectively, where ($\bar{V}$(**IP**) − $\underline{V}$(**IP**)) /$\underline{V}$(**IP**) $\leq \varepsilon$

Notice that, in the worst case, EFRA ends up solving the original **IP**; in real problems, this situation is very unlikely, though.

3.3. *Fix and relax schemes for the project scheduling problem*

In this section we define the partitions that have been used in the computational experimentation reported in Section 4.

3.3.1. *Variable-worth strategies*

We base the selection of a partition upon the concept of *variable worth*: Consistent worths, say c_{jt} are assigned to our decision variables x_{jt} for $t \in T_j, j \in J$. For exposition's clarity, we present strategies as if there were no y- variables in our problem. Should these variables exist, they are assigned to the first partition.

Let us assume a rearrangement of the indices j and t in non-increasing *worth* order:

$$c_{(jt)^1} \geq \cdots \geq c_{(jt)^n}, \tag{27}$$

where $n = \sum_{j \in J} |T_j|$ is the number of x-decision variables. Let also n' be the preferred number of integer variables per stage, so $k = \lceil n/n' \rceil$ is the number of stages (where $\lceil . \rceil$ represents the smallest integer greater than or equal to the argument). We define the first stage with the indices of the n' most valuable pairs (j, t), the second stage with the next n' pairs, and so forth. Formally,

$$V_r = \{(j,t)^i \mid i = (r-1)n' + 1, \ldots, \min\{rn', n\}\}, \quad \forall r = 1, \ldots, k \tag{28}$$

We next propose five different strategies for assigning *worth* to variables:

1. A strategy is based upon the implicit time structure of our model. It is so-called *Fix-and-Relax Time Partitioning* (F&R-TP). The parameters are as follows:

$$\text{F \& R - TP} \equiv \begin{cases} k = |T| \\ c_{jt} = |T| - t, \quad \forall t \in T_j, j \in J \end{cases} \tag{29}$$

   This strategy proposes to assign promptness as variable worth and to set one stage per time-period in the problem. Thus, F&R-TP will determine first the 0-1 values of those variables associated with decisions that occur early in time. F&R-TP accounts for promptness but disregards other quantitative measures of worth such as benefit and resource consumption, among others.

2. A strategy so-called *Fix-and-Relax Objective-Partitioning* (F&R-OP) assigns each variable a worth equals to its objective function value. The parameters are as follows:

$$\text{F\&R-OP} \equiv \begin{cases} k = \lceil n/n' \rceil \\ c_{jt} = b_{jt}, \quad \forall t \in T_j, j \in J \end{cases} \tag{30}$$

In this partition, the number of stages depends on how many variables, n', we want to include in each stage. F&R-OP makes decisions about the most profitable pairs (j, t) first.

3. Another strategy is the so-called *Fix-and-Relax Cost-Partitioning* (F&R-CP). It assigns each variable a worth equals to the budget required by the associated project. The parameters are as follows:

$$\text{F\&R-CP} \equiv \begin{cases} k = \lceil n/n' \rceil \\ c_{jt} = \sum_{d=1,\dots,d_j} p_{jd}, \quad \forall t \in T_j, j \in J \end{cases} \tag{31}$$

Note that the time period t is not relevant here. In fact, c_{jt} depends only on the budget required by project j. Therefore, all the x_{jt} variables for a given project j are included in the same stage. F&R-CP makes decisions about the most expensive projects first.

4. An intermediate strategy is the so-called *Fix-and-Relax Ratio-Partitioning* (F&R-RP). It assigns each variable a worth consistent with its relative benefit. The worth is calculated as the ratio between the objective function value and the budget required to accomplishing the project. The parameters are as follows:

$$\text{F\&R-RP} \equiv \begin{cases} k = \lceil n/n' \rceil \\ c_{jt} = \dfrac{b_{jt}}{\sum_{d=1,\dots,d_j} p_{jd}}, \quad \forall t \in T_j, j \in J \end{cases} \tag{32}$$

F&R-RP makes decisions about the pairs (j, t) with a most favorable "bang-for-the-buck" first.

5. The last worth-based partition strategy that we propose is the so-called *Fix-and-Relax Random-Partitioning* (F&R-rP). It randomly assigns variables to partitions. For instance, variables can be arranged in the order they are entered in the problem, such that the parameters are as follows:

$$\text{F\&R-rP} \equiv \begin{cases} k = \lceil n/n' \rceil \\ c_{jt} = \dfrac{1}{j}, \quad \forall t \in T_j, j \in J \end{cases} \tag{33}$$

3.3.2. Dynamic schemes

As opposed to the variable-worth strategies presented in the previous section, we now depict a simple *dynamic* setup for the variable partition. We employ the term dynamic to refer to any strategy that neither prefixes the number of stages k nor their composition $V_1, \ldots, V_k$. Instead, they are determined during the course of the (conveniently modified) BFRA and EFRA. The new strategy is the so-called *Fix-and-Relax Dynamic-Partition* (F&R-DP).

The dynamic algorithm starts solving the LP Relaxation of **IP** (24); let us name it **RIP**. Let also the following **x**- and **y**-variable partition: V_1 is the set of indices for those variables with fractional values in the **RIP** optimal solution; V_2 is the set of indices for 0-1 valued variables.

We continue by setting $k = 2$ and solving $\mathbf{IP}^1$ which (by the definition of BFRA and EFRA) yields 0-1 values for all variables in V_1. Additionally, V_2 is updated to contain only fractional variables from the optimal solution to $\mathbf{IP}^1$, whereas the remaining non-forced 0-1-valued variables are transferred to a new set, say V_3. Then, we update the number of stages to $k = 3$ and solve $\mathbf{IP}^2$. Again, by looking at the fractional and 0-1-valued variables, we split V_3 into an updated V_3 and a new V_4, respectively. This process is repeated until either V_k is empty or all the 0-1 variables take integer values in the solution for problem $\mathbf{IP}^{k-1}$.

Let us denote by $\hat{\mathbf{x}}^{\mathbf{P}}_{jt}$ the solution for a given variable $\mathbf{x}_{jt}$ in a generic problem **P**. The adaptation of the BFRA scheme to a dynamic version so-called BFRA-Dyn is as follows:

BFRA-Dyn: Basic Fix-and-Relax algorithm with Dynamic Partition Setup

Input: None

Step 0: Solve **RIP**.
Let $V_1 = \{(j,t) \mid \hat{x}^{\mathbf{RIP}}_{jt} \notin \{0,1\}\}$ and $V_2 = \{(j,t) | \hat{x}^{\mathbf{RIP}}_{jt} \in \{0,1\}\}$. Set $k = 2$.

Step 1: Solve $\mathbf{IP}^1$ where $k = 2$ is the current number of stages.
If $\mathbf{IP}^1$ is infeasible, STOP: "Problem **IP** is infeasible".
Otherwise, set $\bar{V}(\mathbf{IP}) = V^*(\mathbf{IP}^1)$.

Step 2: If $V_k = \emptyset$, set $\underline{V}(\mathbf{IP}) = V^*(\mathbf{IP}^{k-1})$ and STOP: "Problem **IP** is feasible".

Step 3: Define a copy of V_k: $\tilde{V} = V_k$, and

- update $V_k = \{(j,t) | (j,t) \in \tilde{V}, \hat{x}^{\mathbf{IP}^{k-1}}_{jt} \notin \{0,1\}\}$
- create $V_{k+1} = \{(j,t) | (j,t) \in \tilde{V}, \hat{x}^{\mathbf{IP}^{k-1}}_{jt} \in \{0,1\}\}$
- increase k by 1

Step 4: Solve $\mathbf{IP}^{k-1}$ where k is the current number of stages.
If $\mathbf{IP}^{k-1}$ is infeasible, STOP: "Problem **IP** status is unknown".
Otherwise, go back to Step 2.

Output: **IP** status: "Infeasible", "Feasible" or "Unknown". If status is "Feasible", the best found lower and upper bounds are $\underline{V}$(**IP**) and $\bar{V}$(**IP**), respectively.

Remark. The definition of $V_1 = \{(j,t) \mid \hat{x}_{jt}^{RIP} \notin \{0,1\}\}$ in Step 0 and the update of $V_k = \{(j,t) \mid (j,t) \in \bar{V}, \hat{x}_{jt}^{\mathbf{IP}^{k-1}} \notin \{0,1\}\}$ in Step 3 can be limited to a maximum number of variables with non-integer values, leaving the remaining variables in the sets V_2 and V_{k+1}, respectively.

Likewise, we might devise a dynamic version of the EFRA scheme. It is worth noting that, in the worst case, the F&R-DP strategy would require as many stages as variables.

3.4. *Algorithm refinements*

In this section, we present some refinements to our model **IP** that strengthen its formulation and (potentially) guide the algorithm to a more efficient search for the optimal solution. First, data preprocessing: Our model presents a large amount of information that can be preprocessed, mainly in conjunction with the structures (3), (4), and (7)–(9), and specially exploiting the sets given by (5) and (6) in conjunction with the 0-1 knapsack constraints (15) and (21)–(23), see Escudero (1988). However, our aim was directed to investigate the performance of the F&R strategies and, then, we did not take benefit from the special structures.

Second, *Explicit-Constraint Branching* (ECB): Appleget and Wood (2000) propose the use of redundant artificial constraints and variables to speed up computations. Note that, by construction, any linear combination of integer variables with integer coefficients results in an integer-valued variable. Sometimes, the appending of this type of constraints to the original model results in a more efficient B&B search for integer solutions by assigning high branching priority to the related artificial integer variables. There is an intuitive reason for using ECB: Once the sum of integer variables is fixed to a given integer value, it appears reasonable that the variables in the summation take also integer values in order to equate that target. Several issues arise here, though: How many (and which) variables are included in each new constraint? What coefficients are to be used in the linear combinations? What branching priorities should be used for the new variables? We do not intend to perform an analysis of these questions in our computation. Rather, we use the underlying idea to build our own ECB scheme as follows:

$$\sum_{\substack{j \in J \mid \\ t \in T_j}} x_{jt} = \tilde{x}_t, \; \forall t \in T \tag{34}$$

$$\sum_{\substack{j \in J^{ac} \mid \\ t \in T_j}} x_{jt} = \tilde{x}_t^{ac}, \; \forall c \in C^a, \; a \in A, \; t \in T \tag{35}$$

$$\tilde{x}_t; \tilde{x}_t^{ac} \text{ integer}, \ \forall c \in C^a, \ a \in A, \ t \in T \tag{36}$$

Our new variable type $\tilde{x}_t$ in the identity (34) accounts for the total number of projects whose execution starts at time period t. Similarly, $\tilde{x}_t^{ac}$ in (35) represents the number of projects in category c of attribute a to start at time period t.

We append the constraints (34)–(36) to the original 0-1 model. A higher priority was assigned to all the $\tilde{x}$-variables, which in turn yielded computing time savings of 20% on average.

4. Computational experience

In this section, we report the results of the computational experience for optimizing the model (1)–(23) with the refinements (34)–(36). All the tests have been performed on a Pentium II, 500 MHz personal computer with 128 Mb of RAM. The code has been programmed in Digital Fortran, version 6.0, Digital (1998), using the Xpress-MP XOSL solver, release 12.11, Dash (1999–2000) as the optimization motor.

Our test cases consider ten budget categories but only a limited number of resource types (no more than two). In any case, we can also considerer the budget categories as other types of resources but with special characteristics. On the other hand, our test cases exhibit a particular project-time structure: A large number of projects must be allocated to a reduced number of time periods. Our computational results concentrate on the ability of the proposed F&R scheme to better exploit the project-time structure inherent to the class of project scheduling problems we are dealing with, which in turn makes F&R outperform direct integer programming techniques such as Branch-and-Bound (B&B). We show that the computing time can be dramatically reduced while still obtaining good-quality solutions.

4.1. Test case description

We present results for three real cases drawn from the electric power sector in Spain. Additionally, seven realistic cases are created as excursions from the three baseline cases. Data correspond to a 5-year horizon (from 1998 through 2002), where more than 13,000 candidate projects are analyzed.

The projects in our test cases involve a variety of activities ranging from the construction of electric distribution lines and substations to the installation of remotely controlled equipment at the facilities. Other important subsets of projects are specifically intended to improve electric service's quality and to comply with existing regulation.

Two main types of benefit functions are considered in the objective function: *Profit* (*PR*, in US dollars), and *Quality of Service* (*QS*, in MWh). The former represents an expected income (or cost, if negative). It is important noting that not all the projects are profitable, that is, PR_{jt} could be negative for a specific project j undertaken at period t. On the other hand, QS_{jt} accounts for improvement on the reliability of the electric

Table 1
Test cases description: Problem dimension.

Cs	Re	$\|T\|$	$\|J\|$	$\|C\|, \|R\|$	$\|L\|, \|M\|$	$\|P\|, \|\bar{P}\|$	$\|G\|$	m	n	nel	d	f_{LP}	fv	t_{LP}
1	no	5	2,500	10, 1	6, 2	0, 0	2	2,529	12,523	50,101	0.15	154.599	0.36	3.1
2	no	3	4,000	10, 1	6, 2	0, 0	2	4,061	12,055	94,163	0.57	75.903	0.12	8.8
3	no	5	6,500	10, 1	50, 10	0, 0	2	6,709	32,629	254,240	0.11	191.261	0.46	70.5
4	no	3	8,000	10, 2	50, 10	10, 10	2	8,242	24,102	212,362	0.10	162.919	0.31	33.1
5	no	5	9,500	10, 1	50, 10	10, 10	2	9,759	47,579	276,125	0.05	218.741	0.09	197.1
6	no	3	11,000	10, 2	50, 10	10, 10	2	11,242	33,102	266,362	0.07	162.919	0.22	39.6
7	no	5	12,500	10, 1	50, 10	10, 10	2	12,809	62,629	488,540	0.06	233.898	0.13	244.5
8	yes	1	13,271	10, 0	0, 0	0, 0	2	13,289	13,289	89,862	0.05	78.331	0.04	43.0
9	yes	5	13,271	10, 0	0, 0	0, 0	2	13,283	66,367	199,091	0.02	232.472	0.04	15.8
10	yes	5	13,271	10, 0	0, 0	0, 0	2	13,345	66,429	449,290	0.05	15.017	0.11	106.7

distribution system, which in turn reduces the so-called "non-supplied energy" (and, indirectly, its cost to the utility). Usually, both benefits are conflicting with each other since the most profitable projects have little impact on the QS, and vice versa.

The most common objective pursued in the context of this application entails a compromise between both goals: maximize the composite function given by a linear combination of both benefits. For doing this, we define $b_{jt} = PR_{jt} + \rho_j QS_{jt}$, where ρ_j (in \$/MWh) reflects the revenue obtained by avoiding penalties for non-supplied energy. The index j indicates a dependency of this saving on the project. In our case, ρ_j roughly ranges from \$300/MWh to \$1,800/MWh. Alternatively, the maximization of either PR or QS can also be obtained but, in this case, a minimum floor is enforced for the other goal.

Table 1 presents the characteristics of our 10 test cases. The additional notation for the headings is as follows: *Cs*, Case index, *Re*, case type, where *yes* and *no* stand for real and realistic, respectively; $|C|$, number of categories among all attributes, i.e., $|C| = \sum_{a \in A} C^a$; m, number of constraints; n, number of 0-1 variables; *nel*, number of nonzero elements in the constraint matrix; d, constraint matrix density (%); f_{LP}, optimal solution value of the LP relaxation; fv, fraction of the variables with non-integer values in the LP solution; t_{LP}, computing time (secs.) to solve the LP problem. Table 2 presents selected statistics on our problem data.

4.2. *Strategies*

For each test case, we have implemented the F&R strategies presented in Section 3.3: (29)–(33) plus the dynamic strategy. We allow $n' = 1,000 \times |T|$ integer variables at each stage for non-dynamic strategies (except for the F&R-TP strategy in which $k = |T|$ and $n' = |J|$ by definition). This choice results in a number of stages k ranging from 1 to 14 in our test cases. The number of stages is not determined in advance for the F&R-DP strategy. It depends on how many variables take integer values during the different stages.

Table 2
Test cases description: Value or range variation of more characteristics and parameters.

Brackets of project cost: p_{j1} ($), and # of projects		Project goal coefficients and objective function coefficients [b], b_{jt}: NPV_{jt} ($ × 1000)	QS_{jt} (MWh)	$NPV_{jt} + \rho_j \cdot QS_{jt}$ ($ × 1000)	Other data[c]
[0 – 10,000], 5,734 projects	Min.	−5,829	0	−4,682	# Missions: 10
(10,000 – 50,000], 5,539 pr.	Avg.	−10	12	12	# of macro-projects per mission: 5
(50,000–100,000], 1,083 pr.	Max.	5,218	3,641	7,271	
(100,000–1 Mill.], 823 pr.					# projects per macro-projects: between 2 and 5
(1 Mill.–8.3 Mill], 92 pr.	Max.	91,732	161,967	366,970	
Total: ≈646 Mill, 13,271 pr. achievable					
Total Budget ($ Mill)[a]:					# min. and max. lag precedences:
$[\underline{p}_t^0, \bar{p}_t^0] = [100, 166.7]$					20 Lag periods: 1 and 2

[a]In addition to total budget constraints, there are category budget constraints for a total of ten categories in two attributes, which are not displayed for conciseness.
[b]E.g., in case #9, we maximize NPV + ρQS where $\rho_j = \$1,800/\text{MWh}\ \forall j$. The optimal solution value is $232 Mill. (out of a maximum of 366.9 Mill.). In case #10, NPV is maximized alone, subject to a minimum target for QS in period 1: $\underline{b}_1^{QS} = 10,000\,\text{MWh}$. In this case, $15 Mill. are achieved (out of a maximum of 91.7 Mill.)
[c]Only for cases 1–7, created as excursions of cases 8–10; see also Table 1.

For the test cases, we find that the number of stages is ranging from 64 for case #8 to 2,317 for case #9 (taking into account that it was halted due to time limitations).

In addition to the F&R strategies, we still attempt to solve the problem by Direct B&B (DB&B). Specifically, we use the default strategy (without any special insight) as well as two other strategies trying to mimic the strategies F&R-TP and F&R-OP. In all the strategies the branching node is chosen according to the best objective function value.

1. *Direct Branch-and-Bound with No-Priorities* (DB&B-NP): The branching priorities are set to the default value $BP(x_{jt}) = 500,\ \forall t \in T_j,\ j \in J$. Therefore, the non-integer valued variables are branched in the order they appear in the problem. This strategy can be compared to F&R-rP in the sense that the order in which variables are entered in the problem determines its branching ordering.
2. *Direct Branch-and-Bound with Time-Priorities* (DB&B-TP): The branching priorities are set as follows: $BP(x_{jt}) = t,\ \forall t \in T_j,\ j \in J$, such that a smaller value means a higher priority (i.e., the variable is branched earlier). This strategy is analogous to F&R-TP where the decisions related to earlier periods are determined first.
3. *Direct Branch-and-Bound with Objective-Priorities* (DB&B-OP): The branching priorities are defined as to ensure that the variables are branched in the order of their

benefits (whether these are positive or negative). Specifically, we assign priority values between 100 and 200 for variables with positive benefit, and from 300 to 400 for variables with negative benefit. Clearly, this strategy is parallel to F&R-OP.

Remark. The $\bar{x}$- and y-variables are given the priority value 0 (i.e., the highest priority value in the optimization engine that we use).

4.3. Numerical results

The Tables 3 to 7 present the results for our 10 test cases. Table 8 summarizes the results from Tables 3 to 7. The additional notation of the headings on Tables 3 to 7 is as follows:

Table 3
Results for Cases 1 and 2.

	Case 1					Case 2				
Strategy	k	f_{IP}	Gap	t_{IP}	nn	k	f_{IP}	Gap	t_{IP}	nn
DB&B-NP	1	154.584	9.E–05	10,313	4,808	1	75.901	2.E–05	3,046	3,732
DB&B-TP	1	154.596	1.E–05	931	5,099	1	75.903	5.E–06	488	3,450
DB&B-OP	1	154.536	4.E–04	2,548	4,071	1	75.647	3.E–03	923	3,410
F&R-TP	5	154.596	1.E–05	998	4,908	3	75.903	1.E–06	1,412	3,688
F&R-OP	3	154.586	8.E–05	133	939	4	75.894	1.E–04	169	927
F&R-CP	3	154.467	9.E–04	166	988	4	75.902	2.E–05	141	963
F&R-RP	3	154.591	5.E–05	165	1,002	4	75.883	3.E–04	161	939
F&R-rP	3	154.577	1.E–04	146	1,219	4	75.903	3.E–06	126	1,084
F&R-DP	214	154.587	8.E–05	852	650	611	75.903	3.E–06	3,673	780

Table 4
Results for Cases 3 and 4.

	Case 3					Case 4				
Strategy	k	f_{IP}	Gap	t_{IP}	Nn	k	f_{IP}	Gap	t_{IP}	nn
DB&B-NP	1	–	–	86,400	8,297	1	162.896	1.E-04	44,007	11,227
DB&B-TP	1	191.170	5.E–04	17,891	15,550	1	162.850	4.E–04	20,802	13,801
DB&B-OP	1	–	–	86,400	72,302	1	–	–	86,400	73,288
F&R-TP	5	190.971	2.E–03	12,183	13,314	3	162.908	7.E–05	8,933	7,356
F&R-OP	7	191.179	4.E–04	449	339	8	162.628	2.E–03	572	2,017
F&R-CP	7	191.196	3.E–04	399	308	8	162.718	1.E–03	554	1,628
F&R-RP	7	191.011	1.E–03	447	558	8	162.327	4.E–03	678	1,722
F&R-rP	7	190.528	4.E–03	617	1,051	8	162.846	5.E–04	1,073	5,475
F&R-DP	518	191.199	3.E–04	11,037	1,356	554	162.885	2.E–04	10,997	1,202

Table 5
Results for Cases 5 and 6.

	Case 5					Case 6				
Strategy	k	f_{IP}	Gap	t_{IP}	nn	k	f_{IP}	Gap	t_{IP}	nn
DB&B-NP	1	–	–	86,400	5,867	1	162.862	3.E–04	63,444	12,108
DB&B-TP	1	–	–	86,400	29,700	1	162.919	4.E–09	1,950	4,134
DB&B-OP	1	–	–	86,400	14,590	1	162.855	4.E–04	3,443	6,387
F&R-TP	5	218.741	3.E–06	19,683	16,480	3	162.901	1.E–04	23,937	8,208
F&R-OP	10	218.666	3.E–04	764	1,040	11	162.721	1.E–03	464	450
F&R-CP	10	218.741	1.E–06	407	433	11	162.336	4.E–03	594	1,253
F&R-RP	10	218.663	4.E–04	608	877	11	162.754	1.E–03	607	1,602
F&R-rP	10	218.707	2.E–04	1,088	2,794	11	162.521	2.E–03	853	2,087
F&R-DP	952	218.740	8.E–06	27,540	1,786	691	162.887	2.E–04	29,802	6,405

Table 6
Results for Cases 7 and 8.

	Case 7					Case 8				
Strategy	k	f_{IP}	Gap	t_{IP}	nn	k	f_{IP}	Gap	t_{IP}	nn
DB&B-NP	1	–	–	86,400	5,875	1	78.330	1.E-05	98	1,464
DB&B-TP	1	–	–	86,400	14,220	1	78.330	1.E-05	98	1,464
DB&B-OP	1	–	–	86,400	18,570	1	78.329	2.E-05	252	496
F&R-TP	5	223.890	3.E–05	9,629	16,004	1	78.330	1.E–05	98	1,464
F&R-OP	13	223.896	8.E–06	1,169	1,139	14	78.328	4.E–05	250	132
F&R-CP	13	223.859	2.E–04	1,690	1,225	14	78.284	6.E–04	243	181
F&R-RP	13	223.826	3.E–04	1,682	1,694	14	78.313	2.E–04	248	163
F&R-rP	13	223.886	5.E–05	1,581	1,500	14	78.330	1.E–05	439	131
F&R-DP	783	–	–	86,400	1,708	64	78.330	1.E–05	1,328	213

Table 7
Results for Cases 9 and 10.

	Case 9					Case 10				
Strategy	k	f_{IP}	Gap	t_{IP}	nn	k	f_{IP}	Gap	t_{IP}	nn
DB&B-NP	1	–	–	86,400	4,821	1	–	–	86,400	3,027
DB&B-TP	1	232.471	2.E-06	47,506	35,408	1	15.015	2.E-04	5,587	12,821
DB&B-OP	1	–	–	86,400	5,660	1	–	–	86,400	2,090
F&R-TP	5	232.469	1.E–05	5,034	11,678	5	15.005	8.E–04	18,397	22,012
F&R-OP	14	232.471	2.E–06	434	391	14	15.006	8.E–04	1,208	435
F&R-CP	14	232.469	1.E–05	422	388	14	14.961	4.E–03	1,173	584
F&R-RP	14	232.471	4.E–06	395	327	14	14.962	4.E–03	1,291	636
F&R-rP	14	232.471	3.E–06	765	552	14	14.965	3.E–03	1,529	534
F&R-DP	2,317	–	–	86,400	3,210	437	–	–	86,400	745

Table 8
Summary of results.

Strategy	# of unsolved	time[a]: # of best	min. gap	max. gap	gap[a]: # of best	Total time**	Total # of nodes[b]
DB&B-NP	5	1	1×10^{-5}	∞	1	552,908	61,226
DB&B-TP	2	1	2×10^{-8}	∞	5	268,053	135,647
DB&B-OP	6	0	2×10^{-5}	∞	0	525,566	200,864
F&R-TP	0	1	1×10^{-6}	1×10^{-3}	4	100,304	105,112
F&R-OP	0	3	1×10^{-6}	1×10^{-3}	2	5,612	7,809
F&R-CP	0	4	1×10^{-6}	3×10^{-3}	2	5,789	7,951
F&R-RP	0	1	4×10^{-6}	3×10^{-3}	0	6,282	9,520
F&R-rP	0	1	3×10^{-6}	3×10^{-3}	0	8,217	16,427
F&R-DP	3	0	2×10^{-6}	∞	2	344,429	18,055

[a] Two or more strategies for the same case can be tied for best time and/or best gap.
[b] For methods with unsolved problems, the reported total time and total # of nodes are after one day of computations for unsolved problems.

f_{IP}, objective function value of the incumbent solution obtained so far for the (original) integer problem; *Gap*, optimality gap defined as $(f_{LP} - f_{IP})/f_{IP}\%$; t_{IP}, computing time (secs.) for the algorithm; *nn*, number of explored B&B nodes in the total of F&R stages. The computing time is very small for the F&R strategies when comparing it with the computing time that is required by the optimization engine for similar quality in the results (i.e., similar *Gap*).

We also notice that the LP solution value (see Table 1) is extremely close to the integer solution value, and only a small fraction of the variables take non-integer values at the LP solution. So, it gives an indication of how tight is the constraint system (3)–(23) and (34)–(36). Yet, this does not make the problem much easier for obtaining the optimal integer solution, probably because those variables have proven decisive in the problem (e.g., due to their cost, profit or relationship with other variables). Therefore, devising a framework to round these variables that preserves feasibility without compromising the solution quality is a complex task.

Some conclusions can be drawn by inspecting the solution quality versus the computing time in Tables 3 to 7: (1) Four of the F&R strategies, namely, F&R-OP, F&R-CP, F&R-RP and F&R-rP, clearly outperform all the other strategies. (2) Amongst the direct B&B strategies, DB&B-TP provides the best results, although its performance is still far from those F&R strategies. (3) The default B&B strategy (DB&B-NP) gives the worst results for the test cases, but the other B&B strategies are too-inefficient as well. (4) F&R-TP (which is intuitively the most natural partition of the problem) happens to be substantially inferior to the other non-dynamic F&R strategies. (This happens to be especially true as the number of projects increases and the number of periods remains the same). (5) The dynamic scheme (F&R-DP) requires too many stages, which in turn makes it less attractive than the non-dynamic F&R strategies. However, the implementation of the dynamic strategy can be improved by reducing the time for model generation (which is negligible in the other strategies since the number of stages is small), which may offer savings up to 40% of the time shown in the tables.

The F&R-rP strategy requires more computing time to obtain the incumbent solution than the strategies F&R-OP, F&R-CP and F&R-RP. On the other hand, F&R-OP and F&R-CP are the strategies that require the smallest computing time. However, the strategy F&R-RP has also a very good performance given the high dimensions of the problem. Notice that the three strategies provide quasi-optimal solutions in less than 30 minutes of a small-size personal computer.

5. Conclusions

Project scheduling is a very broadly studied application field in 0-1 programming. In this work we have presented a 0-1 model for a class of project scheduling problems that can be used in a very wide set of application areas. Important sequencing and scheduling structures have been considered, such as macro-projects (i.e., implicative relationships among the projects), alternative macro-projects for accomplishing missions, exclusivity constraints, budget carrier and non-carrier time periods, budget and other storable resources where the stock variables are not explicitly considered in the model, and multi-period execution projects, among others. A Fix-and-Relax algorithmic framework has been described. It allows a variety of strategies for partitioning the set of variables for branching selection purposes. By exploring only a limited set of nodes from the branch-and-bound tree, (hopefully) good feasible solutions can be obtained with affordable computing time. A computational experience is reported for a set of real-life and realistic cases to compare the performance of the different partitioning strategies that have been presented. Practically optimal solutions have been obtained in all test cases by using the non-dynamic F&R strategies. The partitions F&R-OP (based on project objective function value), F&R-CP (based on project cost) and F&R-RP (based on project objective function value versus project cost) require smaller computing time than the others, with a similar optimality gap. Overall, based on the test cases that we have used for our experimentation, we recommend employing either the strategy F&R-OP or the strategy F&R-CP. The results for these two strategies have been favorably compared with the results for the other F&R strategies. In addition, the performance of any F&R strategy (but the dynamic partition) is clearly superior to the strategies of the state-of-the-art optimization engine that has been used in this research.

Acknowledgments

This research has been partially supported by grant TIC 2000-1750-C06-04(05), CYCIT, Spain and by the National Research Council, USA. We also thank two anonymous referees for their insightful comments on this paper.

References

Akker, J.M. van den, C.A.J. Hurkens, and M.W.P. Savelsbergh. (2000). "Time-indexed Formulations for Machine Scheduling Problems: Column Generation." *INFORMS Journal on Computing* 12, 111–124.

Alonso-Ayuso, A., M.F. Clement, L.F. Escudero, M.L. Gil, and M.T. Ortuño. (2003). "FRC-S3, on Dealing with the Uncertainty for Stochastic Sequencing and Scheduling Problem Solving." Report I-2003-10, Centro de Investigación-Operativa, Universidad Miguel Hernández, Elche, Spain.

Appleget, J.A. and R.K. Wood. (2000). "Explicit-Branching Constraints for Solving Mixed-Integer Programs." In M. Laguna and J.L. Gonzalez Velarde (eds.), *Computing tools for Optimization and Simulation* Kluwer Academic Publishers pp. 245–261.

Balas, E. and R. Martin. (1980). "Pivot and Complement—A Heuristic for 0-1 Programming." *Management Sciences* 26, 86–96.

Balas, E., S. Ceria, M. Dawende, F. Margot, and G. Pataki. (2000). "OCTANE: A New Heuristic for 0-1 Programs." *Operations Research* 49, 207–225.

Baptiste, Ph,. C. Le Pape, and W. Nuijten. (2001). *Constraint-Based Scheduling* Kluwer Academic Publishers.

Constantino, M. (1996). "A Cutting Plane Approach to Capacitated lot Sizing with Set-up Costs." *Mathematical Programming* 75, 353–376.

Dash Optimization, Inc. (1999). Xpress-MP User Manual, http://www.dash.co.uk/.

Dash Optimization, Inc. (2000). Xpress-MP Optimizer Subroutine Library, http://www.dash.co.uk/.

Digital Equipment Corporation (1998). Digital Visula Fortran, Professional Edition 6.0.0, http://www.compaq.com.fortran/.

Dillenberger, Ch., L.F. Escudero, A. Wollensak, and W. Zhang. (1994). "On Practical Resource Allocation for Production Planning and Scheduling with Period Overlapping Setups." *European Journal of Operational Research* 75, 275–286.

Escudero, L.F. (1981). "On Energy Generators Maintenance Scheduling Constrained by the Hourly Distribution of the Weekly Energy Demand." Report G3203420 IBM Scientific Center, Palo Alto, California.

Escudero, L.F. (1982). "On Maintenance Scheduling of Production Units." *European Journal of Operational Research* 9, 264–274.

Escudero, L.F. (1988). "S3 sets. An Extension of the Beale-Tomlin Special Ordered Sets." *Mathematical Programming* 42, 113–123.

Escudero, L.F., E. Gómez, and J. Salmerón. (1997). "SISPIR: An integrated system for investment selection and scheduling in a distribution network," In *Proceedings of the European Power Delivery 97 Conference* Programming models pp. 122–134.

Kularajan, K., G. Mitra, F. Ellison, and B. Nygreen. (2000). "Constraint Classification, Preprocessing and a Branch and Relax Approach to Solving Mixed Integer Programming Models." *International Journal of Mathematical Algorithms* 2, 1–45.

Klein, R. (2000). *Scheduling of Resource Constrained Projects*. Kluwer Academic Publishers, pp. 73–109.

Linderoth, J. and M.W.P. Savelsbergh. (1999). "Search Strategies for Mixed Integer Programming." *INFORMS Journal on Computing* 11, 173–187.

Miller, A.J., G.L. Nemhauser, and M.W.P. Savelsbergh. (2003). "A Multi-Item Production Planning Model with Setup Times: Algorithms, Reformulations, and Polyhedral Characterizations for a Special Case." *Mathematical Programming* Series B 95, 71–90.

Sousa, J. and L.A. Wolsey. (1992). "A Time Indexed Formulation of Non-Preemtive Single Machine Scheduling Problems." *European Journal of Operational Research* 54, 353–357.

Pinedo, M. (1995). *Scheduling Theory, Algorithms and Systems* Prentice-Hall.

Waterer, H., E.L. Johnson, P. Nobili, and M.W.P. Savelsbergh. (2002), "The Relation of Time Indexed Formulations of Single Machine Scheduling Problems to the Node Packing Problem." *Mathematical Programming* Series A 93, 477–494.

Wolsey, L.A. (1990). "Valid Inequalities for Mixed Integer Programs with Generalized and Variable Upper Bounds." *Discrete Applied Mathematics* 25, 251–261.

Wolsey, L.A. (1997). "MIP Modeling of Changeovers in Production Planning and Scheduling Problems." *European Journal of Operational Research* 99, 154–165.

Wolsey, L.A. (2002). "Solving Multi-Item Lot-Sizing Problems with an MIP Solver Using Classification and Reformulation." *Management Science* 48, 1587–1602.