J Sched (2014) 17:145–160
DOI 10.1007/s10951-013-0344-y

The total adjustment cost problem: Applications, models, and solution algorithms

Stefan Kreter · Julia Rieck · Jürgen Zimmermann

Received: 3 August 2012 / Accepted: 1 August 2013 / Published online: 18 August 2013
© Springer Science+Business Media New York 2013

Abstract Resource leveling problems arise whenever it is expedient to reduce the fluctuations in resource utilization over time, while maintaining a prescribed project completion deadline. Several resource leveling objective functions may be defined, consideration of which results in well-balanced resource profiles. In this paper, we concentrate on a special objective function that determines the costs arising from increasing or decreasing the resource utilizations. The resulting total adjustment cost problem occurs, for example, in the construction industry and can be formulated using mixed-integer linear programming models. Apart from a discrete time-based formulation, two polynomial formulations, namely an event-based model and a start-based model, which exploit structural properties of the problem are presented. In addition, a heuristic solution algorithm is proposed to generate start solutions for the problem. We use CPLEX 12.4 to solve medium-scale instances known from the literature. A computational performance analysis shows that the discrete time-based model and the start-based model are suitable for practical applications.

Keywords Project scheduling · Resource adjustment · Minimum and maximum time lags · Mixed-integer linear programming formulations

1 Introduction

Industrial as well as agricultural production activities require a number of resources, like tools, components, facilities, and people, for execution. In order to avoid the extremes of resource overload and resource underload, associated with an increase in costs, companies are forced to utilize expensive renewable resources such as special machines, equipment, or highly qualified manpower evenly over time. Scheduling problems in which interdependent activities are to be scheduled such that the resource utilization will be as smooth as possible over a medium-term planning horizon are called *resource leveling problems* (Demeulemeester and Herroelen 2002, Sect. 5).

Resource leveling problems (RLPs) are interesting from both practical and theoretical points of view. On the one hand, the resource utilization is generally linked to quality of products, reject rates, and balanced material flows; therefore, pursuing a resource leveling approach makes good economic sense. On the other hand, RLPs are $\mathcal{NP}$-hard in the strong sense and difficult to solve to optimality (Neumann et al. 2003b, Sect. 3.4). Thus, many researchers have developed heuristic algorithms for RLPs to generate good solutions for practical relevant problem sizes (Neumann et al. 2003a; Ballestin et al. 2007). However, RLPs also have nice structural properties, consideration of which can be used to find optimal solutions for medium-scale instances in moderate computation times.

A multitude of resource leveling functions may be defined, where the minimization will provide well-balanced resource profiles. In this paper, we consider a special objective function that coincides with the cumulative costs arising from increasing or decreasing the utilizations of resources. Despite its practical importance, the resulting “total adjustment cost function” has achieved little attention in the literature. In comparison with changeover costs that depend on the activity sequence (Schwindt 2005, Sect. 5.2), adjustment costs arise at every jump discontinuity in the resource profiles. Hence, adjustment costs serve as surrogate or auxiliary costs

S. Kreter (✉)· J. Rieck · J. Zimmermann
Operations Research Group, Institute of Management and Economics, Clausthal University of Technology, 38678 Clausthal-Zellerfeld, Germany
e-mail: stefan.kreter@tu-clausthal.de

on a quantity basis that help to reach a stability and continuity in resource utilizations, which is, e.g., important during coating, casting, or heat-treatment processes (Harris and Ioannou 1998; Zhuchkov 2004; Al-Joma and Mangin 2004; Steinboeck et al. 2013).

The remainder of this paper is organized as follows: In Sect. 2, we formally describe basic concepts on project scheduling and present a formulation of the general problem with temporal constraints. In Sect. 3, we discuss different resource leveling objective functions and give a review on exact and heuristic solution methods. Section 4 contains structural properties of the "total adjustment cost problem" and Sect. 5 is devoted to practical applications. In Sect. 6, mixed-integer linear model formulations, among which, two *polynomial* formulations are presented. Based on those models, we proceed to describe preprocessing methods for improving the quality of the resulting formulations in terms of computation time and solution gap (cf. Sect. 7). The results of a comprehensive performance analysis are given in Sect. 8, and finally, conclusions are presented in Sect. 9.

2 Problem description

In what follows, we assume that the project under consideration consists of $n \geq 1$ activities, as well as two fictitious activities, 0 and $n+1$, representing the beginning and completion of the project, respectively. Each activity i has a given processing time $p_i \in \mathbb{N}_0$ and must be carried out without interruption. For the activities 0 and $n+1$, as well as for milestones, which specify significant events, we set the processing time to zero.

$S_i \geq 0$ indicates the start time and $C_i := S_i + p_i$ the completion time of activity i. We assume that the underlying project begins at time 0, i.e., $S_0 := 0$. Then, the project duration equals S_{n+1}. Furthermore, we consider start-to-start relationships between activities. A minimum time lag $d_{ij}^{\min} \in \mathbb{N}_0$ between activities i and j makes sure that activity j cannot be started earlier than $d_{ij}^{\min}$ time units after the start of activity i. If $d_{ij}^{\min} = p_i$, i.e., activity j can be begun as soon as activity i is finished, the minimum time lag is referred to as "precedence constraint." A maximum time lag $d_{ij}^{\max} \in \mathbb{N}_0$ exists, if activity j must be started no later than $d_{ij}^{\max}$ time units after activity i. More formal, a given minimum time lag between the start of activities i and j says that $S_j - S_i \geq d_{ij}^{\min}$, and a maximum time lag between the start of activities i and j implies that $S_j - S_i \leq d_{ij}^{\max}$ has to be satisfied.

We assume that a project is represented by an activity-on-node network $N = (V, A; \delta)$, where activities are identified with nodes. Thus, the node set may be specified by $V := \{0, 1, \ldots, n+1\}$. Set A contains an arc for every minimum and maximum time lag, respectively. For a minimum time lag, an arc $\langle i, j \rangle$ having weight $\delta_{ij} := d_{ij}^{\min}$, and for a maximum time lag, a backward arc $\langle j, i \rangle$ with weight $\delta_{ji} := -d_{ij}^{\max}$ is introduced. At a medium-term planning level, a deadline $\overline{d}$ for the termination of the project is usually given. To ensure the adherence to the prescribed deadline, the constraint $S_{n+1} \leq \overline{d}$ must be introduced. Hence, the maximum project duration $\overline{d}$ may be considered by adding an arc $\langle n+1, 0 \rangle$ with weight $\delta_{n+1,0} := -\overline{d}$ to the network. In order to guarantee that activities 0 and $n+1$ really represent the beginning and completion of the project in question, the conditions $S_i - S_0 \geq 0$ and $S_{n+1} - S_i \geq p_i, i \in V$, must be fulfilled.

Owing to the maximum project duration $\overline{d}$, the set of feasible start times of activity $i \in V$ forms a proper time window $[ES_i, LS_i]$, where ES_i is the earliest and LS_i the latest start time of activity i with respect to the given temporal constraints. By definition, $ES_0 = LS_0 := 0$. For activity $i \in V \setminus \{0\}$, both the earliest start time, which equals the length of a longest path from node 0 to node i, and the latest start time, which equals the negative of the longest path length from node i to node 0, can be determined by applying some label-correcting algorithm (see, e.g., Ahuja et al. 1993, Sect. 5.4).

A sequence of start times $S = (S_0, S_1, \ldots, S_{n+1})$, where $S_i \geq 0, i \in V$, and $S_0 = 0$, is termed a "schedule." A schedule is said to be feasible if it satisfies all temporal constraints of the project given by minimum and maximum time lags. The set of all feasible schedules is denoted by $\mathcal{S}_T$. Then, the problem of finding an optimal schedule for some objective function $f : \mathbb{R}_{\geq 0}^{n+2} \rightarrow \mathbb{R}$ can be formulated as follows:

$$\left.\begin{array}{ll} \text{Minimize} & f(S) \\ \text{subject to} & S_j - S_i \geq \delta_{ij} \quad \langle i, j \rangle \in A \\ & S_0 = 0. \end{array}\right\} \qquad (1)$$

The feasible region $\mathcal{S}_T$ of the described problem is nonempty, iff network N contains no cycle of positive length (Bartusch et al. 1988).

In project scheduling, we distinguish between time-based, financial, and resource-based objectives. A resource leveling approach always considers some nonregular resource-based objective function that evaluates the resource utilization over time. We assume that renewable resources, such as machines, equipment, manpower, or space, are required for carrying out the project activities. Let $\mathcal{R}$ be the set of renewable resources available at each point in time, independently of their utilization formerly. The amount of resource $k \in \mathcal{R}$ used constantly during execution of activity $i \in V$ (resource requirement) is represented by $r_{ik} \in \mathbb{N}_0$. The project in question consists of n activities that can be separated into m milestones (with $p_i = 0$) and $n' := n - m$ real activities (with $p_i > 0$). The set of real activities is denoted by V^r and the set of milestones by V^m. Without loss of generality, we suppose that the resource requirements of milestones, as well as activities 0 and $n+1$, are zero. In order to avoid that two activities i and j with $C_i = S_j$ compete for a renewable resource, we assume that each real activity is executed during the half-open time interval $[S_i, S_i + p_i[$.

Given some schedule S, the set of points in time from interval $[0, \overline{d}]$ at which at least one real activity $i \in V^r$ is started or completed, is denoted by DT (decision times concerning schedule S). Furthermore, we make use of the so called "active set," $\mathcal{A}(S,t)$, which contains all real activities in progress at time t, i.e., $\mathcal{A}(S,t) := \{i \in V^r | S_i \leq t < S_i + p_i\}$. Thus, $r_k(S,t) := \sum_{i \in \mathcal{A}(S,t)} r_{ik}$ represents the total amount of resource $k \in \mathcal{R}$ required for those activities in progress at time t. The resulting resource profiles $r_k(S, \cdot) : [0, \overline{d}] \rightarrow \mathbb{R}_{\geq 0}$ are step functions continuous from the right at their jump points.

3 Objective functions and literature review

Resource leveling functions usually aim at leveling the available renewable resources over the planning horizon. Consequently, the resources necessary to carry out the activities involved should be distributed evenly within the project duration. In this section, we briefly discuss three different objective functions for RLPs. In particular, we consider the "total adjustment cost function," as well as two other functions that are usually proposed in the literature. Each function may be used to model specific applications appearing in resource leveling (i.e., scheduling of maintenance as well as consulting teams, scheduling of machines, or scheduling of internal staff). In Sect. 4, we will depict the common and different structural properties of the three objective functions.

The first resource leveling objective function coincides with the cumulative costs arising from increasing or decreasing the resource utilizations. The resulting "total adjustment cost function" is used, for example, if the resources represent different kinds of manpower, where changing the size of work force from period to period is associated with high costs.

For a formal description of the total adjustment costs, some parameters must be introduced. Let $\sigma_t := \max\{\tau \in DT | \tau < t\}$ be the decision time preceding some $t \in DT \backslash \{0\}$, then

$$\Delta r_{kt} := \begin{cases} r_k(S,t) - r_k(S,\sigma_t), & \text{if } 0 < t \leq \overline{d} \\ r_k(S,0), & \text{otherwise} \end{cases}$$

represents the jump difference in the resource profile of resource $k \in \mathcal{R}$ at some time $t \in DT$. Moreover, let $\Delta^+ r_{kt} := (\Delta r_{kt})^+$ and $\Delta^- r_{kt} := (-\Delta r_{kt})^+$ be the increase and decrease, respectively, in utilization of resource k at time t, where $(z)^+ := \max(z, 0)$. Consequently, the total adjustment cost function may be given by

$$f(S) := \sum_{k \in \mathcal{R}} \sum_{t \in DT} (c_k^+ \, \Delta^+ r_{kt} + c_k^- \, \Delta^- r_{kt}).$$

The parameters $c_k^+ \geq 0$ and $c_k^- \geq 0$ denote the costs for positive and negative jumps in resource utilization of resource k by one unit. Since the condition $\sum_{t \in DT} \Delta^+ r_{kt} = \sum_{t \in DT} \Delta^- r_{kt}$ holds for all k, the total adjustment cost function may be formulated by

$$f(S) := \sum_{k \in \mathcal{R}} c_k \sum_{t \in DT \backslash \{\overline{d}\}} \Delta^+ r_{kt}, \tag{A-RL}$$

where $c_k := c_k^+ + c_k^-, k \in \mathcal{R}$. Hence, it is sufficient to consider only the positive jump differences in the resource profiles.

Several authors have dealt with the "classical resource leveling objective function," where smooth resource profiles have to be realized, and high resource utilizations are more penalized than low resource utilizations. The corresponding function can be specified by

$$f(S) := \sum_{k \in \mathcal{R}} c_k \int_{t \in [0, \overline{d}]} r_k^2(S,t) \, dt, \tag{C-RL}$$

where $c_k \geq 0$ is the cost incurred per unit of resource $k \in \mathcal{R}$ and per time unit.

The "overload problem" considers costs if either given supplies of renewable resources or thresholds $Y_k, k \in \mathcal{R}$, for the resource utilization are exceeded. Let $c_k \geq 0$ be, for example, the cost for overtime premiums, then the "overload cost function" may be given as follows

$$f(S) := \sum_{k \in \mathcal{R}} c_k \int_{t \in [0, \overline{d}]} (r_k(S,t) - Y_k)^+ dt. \tag{O-RL}$$

Note that we could find an individual application for each objective function. In Sect. 5, an application for the total adjustment cost problem is presented. Other resource leveling objective functions are not suitable in this particular case as they may lead to undesirable results (e.g., resource profiles with breaks). Moreover, some applications may exist for which a simultaneous consideration of two or three objectives in a multi-criteria approach could be of interest. We will investigate this further in Sect. 9.

One of the first exact solution approaches for RLPs was introduced in Petrovic (1969). The method is based on dynamic programming and aims at minimizing objective function (C-RL). Later on, Ahuja (1976, Sect. 11) presented a procedure that enumerates every combination of feasible start times to minimize a quadratic variant of objective function (A-RL). A mixed-integer linear model and an extended dynamic programming approach for overload problems are specified in papers by Easa (1989) and Bandelloni et al. (1994). All aforementioned procedures only treated instances with a maximum of 16 real activities, one renewable resource, and just precedence constraints between activities.

Younis and Saad (1996) proposed a method based on integer programming to solve problems with more than one renewable resource. Thereby, the authors considered objective functions that are similar to (A-RL) and (O-RL).

For all described RLPs with general temporal constraints and several renewable resources, Neumann and Zimmermann (2000) presented a time-window based branch-and-bound procedure, and Neumann et al. (2006) as well as Gather et al. (2011) developed a tree-based enumeration scheme that enumerates extreme points of order polytopes (quasistable schedules). Mixed-integer programming (MIP) formulations for problems with objective functions (C-RL) and (O-RL) are introduced by Rieck et al. (2012). The authors solved instances with a maximum of 50 real activities and varying project completion deadlines to optimality using CPLEX 12. The corresponding algorithm outperforms the procedures presented by Neumann and Zimmermann (2000) as well as Gather et al. (2011) in terms of computation time. We therefore use this observation as an opportunity to develop smart MIP-formulations for the total adjustment cost problem. Moreover, the underlying structural properties (cf. Sect. 4) allow a completely new programming model that can be formulated as a MIP, where the number of variables and the number of constraints is polynomial in $|V|$ and $|\mathcal{R}|$.

In order to solve RLPs approximately, many authors developed heuristic approaches, where most of them may be used for all objective functions mentioned above. Shifting and priority-rule methods for problems with precedence constraints can be found in Burgess and Killebrew (1962), Galbreath (1965), Morder and Phillips (1970, Sect. 8), Ahuja (1976), as well as Harris (1978, 1990). In addition, several metaheuristics either based on local search or on ant colony optimization are introduced by Takamoto et al. (1995), Savin et al. (1997), Raja and Kumanan (2007), and Geng et al. (2011). In the latter (relatively new) article, the authors do not execute a comprehensive performance study, they rather consider only one small problem instance with nine real activities.

Neumann and Zimmermann (1999, 2000) proposed heuristic methods being suitable for most RLPs as well as projects with general temporal constraints. In Neumann et al. (2003a) neighborhoods, which allow schedule-improvement procedures to reach optimal solutions independently of the initial schedule chosen, are presented. Moreover, Ballestin et al. (2007) published a population-based approach of the integrated greedy-type that generates the best-known results for large instances with a maximum of 1,000 activities so far. Due to the success of the procedure, we adapted the Ballestin et al. (2007) method to generate upper bounds as well as start solutions for the total adjustment cost problem (cf. Sect. 7).

4 Structural properties for the total adjustment cost problem

The objective function (A-RL) is lower semicontinuous and quasiconcave on each set of schedules implying the same precedence constraints [i.e., (A-RL) is locally quasiconcave], just as functions (C-RL) and (O-RL). For that reason, there invariably exists a quasistable schedule (cf. Rem. 1) that will be optimal for the total adjustment cost problem (Neumann et al. 2003b, Sect. 3.3).

Remark 1 A feasible schedule S is termed *quasistable* if there is no pair of opposite order-preserving shifts.

Let S be a quasistable schedule. For each activity $i \in V$, there is either an activity $j \in V$ such that $S_i + p_i = S_j$ or $S_i + \delta_{ij} = S_j$, or an activity $h \in V$ such that $S_i = S_h + p_h$ or $S_i = S_h + \delta_{hi}$. As we assume that the input data, p_i and δ_{ij}, $i, j \in V$, are integers and $S_0 := 0$, every quasistable schedule will be integer valued and no optimal solution will be lost due to an adequate discretization of the time horizon.

For a given schedule S, the values of (C-RL) and (O-RL) may be determined by considering the total amount of resource $k \in \mathcal{R}$ required for all activities in progress at each point in time $t \in \{0, \ldots, \overline{d} - 1\}$. In contrast, evaluation of objective function (A-RL) requires the knowledge of jump differences in the resource profiles at each point in time. If there is an increase or decrease in resource utilizations from one period to the next, the change must be absorbed with costs, and if the resource utilization remains constant, no additional costs arise. Therefore, the value of (A-RL) may be calculated by considering the start and end times (events) of activities, i.e., points in time $\tau \in \{S_i, C_i\}_{i \in V^r}$. Notice, if more than one event occurs at the same time, for example $C_i = S_j$ holds for two consecutive activities i and j, only one point must be incorporated.

Furthermore, functions (C-RL) and (O-RL) are r-monotone (cf. Rem. 2), whereas (A-RL) is not r-monotone.

Remark 2 A function is called r-monotone if for partial schedules S and S' with $r_k(S, t) \leq r_k(S', t)$, $k \in \mathcal{R}, t \in [0, \overline{d}]$, the condition $f(S) \leq f(S')$ is implicitly specified (Zimmermann 2001).

For problems with r-monotone objective functions, the generation of lower bounds within a branch-and-bound or branch-and-cut approach is performed using the information of already fixed activity start and completion times. Let $\mathcal{C}$ be the completed set, i.e., the set of all activities which have already been scheduled, and let $S^{\mathcal{C}}$ be a partial schedule, where $S_i^{\mathcal{C}} \geq 0$ for every activity $i \in \mathcal{C} \subseteq V$ and $S_0^{\mathcal{C}} = 0$. A partial schedule is feasible if all temporal constraints between activities $i \in \mathcal{C}$ are satisfied. Then, $f(S^{\mathcal{C}})$ forms a lower bound on objective functions (C-RL) and (O-RL). Since

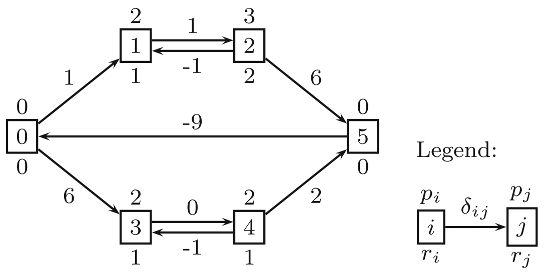

Fig. 1 Activity-on-node network $N = (V, A; \delta)$

function (A-RL) does not have the r-monotone characteristic, the determination of lower bounds is more difficult and requires the estimation of jump differences in resource profiles with respect to scheduled and unscheduled activities. Consequently, the values of lower bounds, e.g., the values of linear programming relaxations used in solvers like CPLEX, are relatively weak for the total adjustment cost problem in comparison to lower bounds for the classical resource leveling or the overload problem.

The following example will illustrate the structural properties of the total adjustment cost problem. Thereby, we consider a project network with four real activities and one renewable resource (index k can be eliminated, cf. Fig. 1).

Let $c := 1$ be the respective costs per resource unit. Figure 2 depicts the resource profiles of an optimal schedule for problem (1) with objective function (A-RL), i.e., $S^* = (0, 2, 3, 6, 6, 9)$, and with objective function (C-RL), i.e., $S^\diamond = (0, 1, 2, 6, 7, 9)$. It is obvious that successive activities should be scheduled without breaks in the total adjustment cost problem to obtain as much end-start relationships as possible. Furthermore, assume that the resource profile of a partial schedule $S^{\mathcal{C}} := S^\diamond$ is illustrated in the lower part of Fig. 2 and an additional activity i with $r_i := 2$, $p_i := 1$, $ES_i := 0$, and $LS_i := 8$ has to be scheduled. Then, due to the r-monotone characteristic, $f(S^{\mathcal{C}})$ represents a lower bound on the classical RLP. For the total adjustment cost problem, a better objective value is achieved by setting $S_i := 5$, i.e., $f(S^{\mathcal{C}})$ indicates no lower bound.

5 Practical applications

The total adjustment cost problem is discussed by Ahuja (1976, Sect. 11) as well as Younis and Saad (1996), but no explicit and common application area is given in both papers. That is why this section is devoted to interesting practical applications that can be found in the construction industry.

Large construction sites typically exist in the fields of structural engineering, underground mining, traffic route engineering, bridge-building, or power plant construction. At all those sites, a lot of international partners are involved to carry out a variety of work packages and activities. For example, facilities and machines must be put into operation, mechanical control software must be tested, or maintenance activities must be performed. The individual activities require time (usually some weeks or months) and renewable resources (usually employees with different skills). Moreover, there are given temporal and/or precedence relationships among activities, which result from technological or organizational constraints in practice, and that may be modeled by minimum and maximum time lags. A precedence constraint specifies, for example, the sequence in which two consecutive operations have to be performed, e.g., building a foundation must precede establishing a building structure.

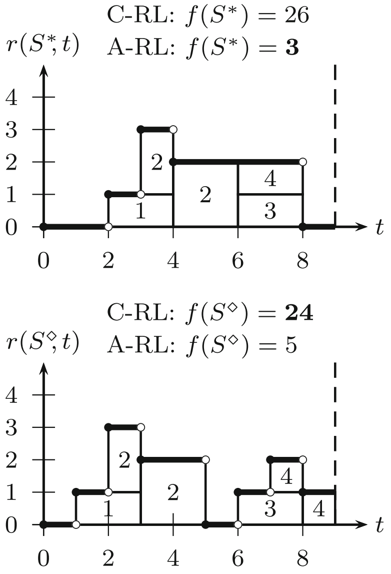

Fig. 2 Resource profiles of schedules S^* and $S^\diamond$

Most activities identified by the site management could be covered by internal staff of the construction company. However, there are some activities that must be arranged by specialized, international professionals (e.g., electrical engineers, installers, plant mechanics, or scientists). The commitment of those professionals (modeled as renewable resources) must be planned in detail. Since the employees usually want to work continuously at the same construction site, the constitution of large periods of employment is of particular importance. If the completion time of a work package is equal to the start time of a successive work package, both packages can be consolidated and offered to one and the same person. In addition, the consideration of such (meta) work packages is also advantageous for the construction company, as an employment-interruption results in high administrative costs (e.g., for entry visa, residence permit, and work permit), or rather transport and relocation costs. In what follows, those costs are considered as "adjustment costs" and by minimizing these costs, we obtain smooth resource profiles. Note that a solution S with a few large resource jumps and another solution S' with a lot of small resource jumps are equiva-

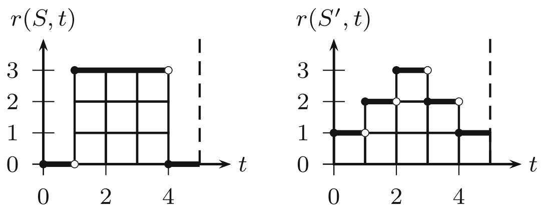

Fig. 3 Resource profiles of schedules S and S'

lent (cf. Fig. 3), if $f(S) = f(S')$ and no quantity discounts are considered. Since specialized professionals usually come from different regions and countries, the costs for entry visa, transport or relocation have to be considered individually, i.e., no quantity discounts for groups can be achieved.

In contrast to traditionally qualified staff, high-qualified professionals know the extent of their possibilities and skills, have high requirements from themselves and their surrounds, and often possess a very high inner motivation (Blaskova and Grazulis 2009, Sect. 16), therefore they want to perform "knowledge work" without paid breaks on-site (in marked contrast to machines). For example, if machines are needed for the execution of activities and must be rented, fixed and variable costs occur. Then, at some points in time it is optimal to rent more resource units than used to reduce the fixed costs (Nübel 2001).

As a special example, we consider the construction of offshore wind farms in the North Sea. In order to cover more than 15 % of the German energy requirements in the future, more than 5,500 wind turbines will be installed within the next 20 years (Haubrich et al. 2008). In this process, certain sub-projects, and within the sub-projects a large number of work packages, must be defined and executed. Sub-projects are the construction of grid connection and cable, the erection and assembly of upper and lower parts of the substation, pile driving, laying of foundations for turbines, and erection of turbines. For performing sub-projects and work packages, several external professionals are required (e.g., offshore and sub-sea engineers, geologists, or turbine operators). Figure 4 depicts a simplified activity-on-node network for constructing offshore wind farms, where the work packages are weighted with their durations and resource requirements (only one kind of professional, i.e., the offshore engineers that supervise the whole construction process, is modeled as a renewable resource). Additionally, the resource profile of an optimal schedule S^* for the total adjustment cost problem is given.

The solution S^* shows that four offshore engineers are necessary to carry out the project activities and if each professional can be employed for successive work packages, two persons must be engaged for 23 weeks and two persons for 12 weeks, respectively. The corresponding staff planning schedule is demonstrated in Fig. 5.

6 Model formulations

In this section, we present three different MIP-formulations for the total adjustment cost problem. The basic idea of the first formulation comes from Pritsker et al. (1969) who used a discretization of the time horizon (cf. Sect. 6.1). The associated *discrete time-based formulation* is often very efficient in solving project scheduling problems with tight project completion deadlines (Rieck et al. 2012). In Sect. 6.2, we adapted a *start/end event-based formulation* that was presented by Koné et al. (2011) within the field of resource-constrained project scheduling. The third formulation is entirely new and

Fig. 4 Activity-on-node network for constructing offshore wind farms and resource profile of schedule S^*

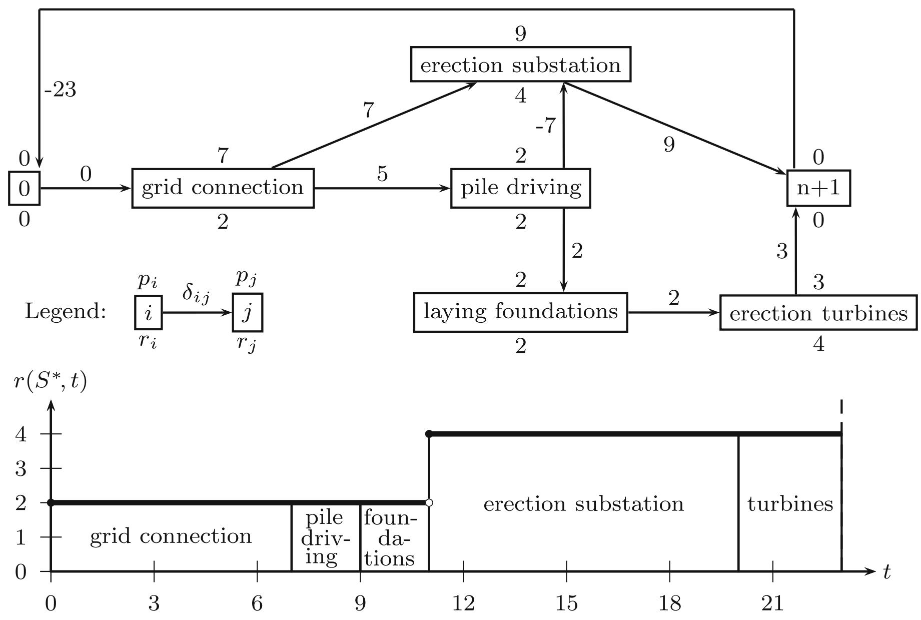

Fig. 5 Staff planning schedule

week/name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Brown	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Martin	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Smith												✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Walker												✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

uses a polynomial number of variables and constraints due to the fact that a positive jump in resource profiles only occurs at a start time of a real activity (cf. Sect. 6.3).

6.1 Discrete time-based formulation

Most common model formulations for project scheduling problems are based on a time discretization, where the time axis is divided into equidistant subintervals. The corresponding discrete time-based model applies binary variables x_{it} that allocate a feasible start time $t \in W_i := \{ES_i, \ldots, LS_i\}$ to each activity $i \in V$, i.e.,

$$x_{it} := \begin{cases} 1, & \text{if activity } i \text{ starts at time } t \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

In order to specify the total adjustment cost objective function, auxiliary variables $\Delta^+_{kt} \geq 0$, which indicate the positive adjustment of resource utilizations for resource $k \in \mathcal{R}$ and time $t \in T := \{0, \ldots, \overline{d} - 1\}$, are introduced. If the start time of real activity i is an element of set $\{t - p_i + 1, \ldots, t\}$, activity i is in progress at time t and requires renewable resources. Consequently, the total resource requirement for some resource k and some schedule S at time t may be given by

$$r_k(S,t) = \sum_{i \in V^r} r_{ik} \sum_{\tau = \max\{ES_i, t-p_i+1\}}^{\min\{t, LS_i\}} x_{i\tau}.$$

The discrete time-based formulation for the total adjustment cost problem then takes the form:

$$\text{Minimize} \quad \sum_{k \in \mathcal{R}} c_k \sum_{t \in T} \Delta^+_{kt} \tag{A-RL-I}$$

subject to:

$$\sum_{t \in W_i} x_{it} = 1 \qquad i \in V \tag{3}$$

$$\sum_{t \in W_j} t\, x_{jt} - \sum_{t \in W_i} t\, x_{it} \geq \delta_{ij} \qquad \langle i, j \rangle \in A \tag{4}$$

$$\Delta^+_{kt} \geq \sum_{i \in V^r} r_{ik} \sum_{\tau = \max\{ES_i, t-p_i+1\}}^{\min\{t, LS_i\}} x_{i\tau} - \sum_{i \in V^r} r_{ik} \sum_{\tau = \max\{ES_i, t-p_i\}}^{\min\{t-1, LS_i\}} x_{i\tau} \qquad k \in \mathcal{R}, t \in T \tag{5}$$

$$x_{00} = 1 \tag{6}$$

$$\Delta^+_{kt} \geq 0 \qquad k \in \mathcal{R}, t \in T \tag{7}$$

$$x_{it} \in \{0, 1\} \qquad i \in V, t \in W_i. \tag{8}$$

Constraints (3) and (8) guarantee that each activity receives exactly one start time within the project duration. Since $\sum_{t \in W_i} t\, x_{it}$ equals the start time of activity $i \in V$, inequalities (4) ensure that the temporal constraints will be satisfied. Constraints (5) and (7) estimate the positive jump difference in the resource profile of resource k at time t. Finally, condition (6) sets the project beginning to zero.

The formulation incorporates $\mathcal{O}(|V|^2 + |\mathcal{R}||T|)$ constraints, as well as $|\mathcal{R}||T|$ real-valued auxiliary variables and $\sum_{i \in V} |W_i|$ binary variables. Since all numbers depend on the time discretization, the model cannot be designated as a polynomial model. Easa (1989) as well as Younis and Saad (1996) presented float-formulations for RLPs that have some similarities to the discrete time-based formulation. However, such models turned out to be less efficient in numerical tests (Rieck et al. 2012).

6.2 Start/end event-based formulation

In order to reduce the number of decision variables and the number of constraints for problems with large time horizons, Koné et al. (2011) presented a MIP-formulation that considers only the start and end times (henceforth called events) of the activities. In Sect. 4 we have pointed out that at least one integer valued, quasistable schedule is optimal for the total adjustment cost problem. Consequently, the number E of possible events is restricted to the number of start and end times of the activities or the project completion deadline $\overline{d}$. Since the condition $S_i = C_i$ is satisfied for fictitious activities $i \in \{0, n+1\}$, as well as for milestones $i \in V^m$, the number of start and end times may be determined by $2\,|V^r| + 2 + |V^m|$, i.e., $E := \min\{2\,|V^r| + 2 + |V^m|, \overline{d}\}$.

The start/end event-based formulation involves two types of binary decision variables indicating whether activity $i \in V$ starts or ends at event $e \in \mathcal{E} := \{1, \ldots, E\}$, i.e.,

$$x^s_{ie}\ (x^c_{ie}) := \begin{cases} 1, & \text{if activity } i \text{ starts (ends) at event } e \\ 0, & \text{otherwise.} \end{cases}$$

In order to specify the corresponding points in time of events $e \in \mathcal{E}$, continuous variables $t_e \in [0, \overline{d}]$ are introduced. Additionally, continuous variables $\boldsymbol{\Delta}^+_{ke} \geq 0$ state the positive jump in the resource profile of resource $k \in \mathcal{R}$ at event e. The start/end event-based formulation can then be given as follows:

$$\text{Minimize} \quad \sum_{k\in\mathcal{R}} c_k \sum_{e\in\mathcal{E}} \boldsymbol{\Delta}^+_{ke} \qquad \text{(A-RL-II)}$$

subject to

$$\sum_{e\in\mathcal{E}} x^s_{ie} = 1 \qquad i \in V \tag{9}$$

$$\sum_{e\in\mathcal{E}} x^c_{ie} = 1 \qquad i \in V \tag{10}$$

$$x^s_{01} = 1,\; x^c_{01} = 1,\; t_1 = 0 \tag{11}$$

$$t_{e+1} - t_e \geq 1 \qquad e \in \mathcal{E} \setminus \{E\} \tag{12}$$

$$t_e + p_i\, x^s_{ie} - p_i \left(1 - x^c_{if}\right) \leq t_f \qquad i \in V,\; e,\, f \in \mathcal{E} : f > e \tag{13}$$

$$t_e + p_i\, x^s_{ie} + 2\overline{d}\left(1 - x^c_{if}\right) - \overline{d}\left(x^s_{ie} - x^c_{if}\right) \geq t_f \qquad i \in V,\; e,\, f \in \mathcal{E} : f > e \tag{14}$$

$$\sum_{f=1}^{e} x^c_{if} + x^s_{ie} \leq 1 \qquad i \in V^r, e \in \mathcal{E} \tag{15}$$

$$ES_i\, x^s_{ie} \leq t_e \qquad i \in V, e \in \mathcal{E} \tag{16}$$

$$LS_i\, x^s_{ie} + \overline{d}\left(1 - x^s_{ie}\right) \geq t_e \qquad i \in V, e \in \mathcal{E} \tag{17}$$

$$x^c_{ie}\left(ES_i + p_i\right) \leq t_e \qquad i \in V, e \in \mathcal{E} \tag{18}$$

$$x^c_{ie}\left(LS_i + p_i\right) + \overline{d}\left(1 - x^c_{ie}\right) \geq t_e \qquad i \in V, e \in \mathcal{E} \tag{19}$$

$$t_f - t_e - \delta_{ij} \geq 2\overline{d}\left(x^s_{ie} + x^s_{jf} - 2\right) \qquad \langle i, j\rangle \in A,\; e,\, f \in \mathcal{E} \tag{20}$$

$$\boldsymbol{\Delta}^+_{ke} \geq \sum_{i\in V^r} r_{ik}\left(x^s_{ie} - x^c_{ie}\right) \qquad e \in \mathcal{E}, k \in \mathcal{R} \tag{21}$$

$$\boldsymbol{\Delta}^+_{ke} \geq 0 \qquad e \in \mathcal{E}, k \in \mathcal{R} \tag{22}$$

$$t_e \in [0, \overline{d}] \qquad e \in \mathcal{E} \tag{23}$$

$$x^s_{ie} \in \{0, 1\},\; x^c_{ie} \in \{0, 1\} \qquad i \in V, e \in \mathcal{E}. \tag{24}$$

Constraints (9), (10), and (24) ensure that each start and each completion time of an activity coincides with an event. Equations (11) set the start time of the project to zero. Constraints (12) guarantee that two successive events are one time period apart. Supposing that activity i starts at event e and ends at event f, the condition $t_f = t_e + p_i$ must be satisfied (cf. inequalities (13) and (14)). Constraints (15) ensure that the completion-event of real activity i has a greater number than the start-event of i. By means of inequalities (16) to (20), the given temporal constraints are fulfilled, and finally the positive adjustment of resource k at event e is estimated with constraints (21) and (22).

The start/end event-based model contains $\mathcal{O}(|V|^4 + |V|\,|\mathcal{R}|)$ constraints along with $|\mathcal{E}|(1+|\mathcal{R}|)$ real-valued auxiliary and $2|V|\,|\mathcal{E}|$ binary variables. All numbers are independent of the scaling of the time axis, therefore the model can be classified as a *polynomial* model.

Koné et al. (2011) used the start/end event-based methodology in order to solve resource-constrained project scheduling problems with precedence constraints. For those problems, there always exists an optimal solution in which the start time of an activity is either 0 or coincides with the end time of some other activity. Therefore, the number of possible events can be defined as $E' := \min\{n + 1, \overline{d}\}$ which is significantly lower than the number E of events needed for problems with general temporal constraints. In addition, RLPs are characterized by many feasible schedules with the same objective function value. Therefore, the events may be positioned at various analogous points in time, which reduces the performance of a solver.

Note that polynomial flow-based and on/off event-based formulations are also available for resource-constrained project scheduling problems (Koné et al. 2011). However, both formulations cannot be extended to a RLP as they are not able to describe total resource requirements at specific points in time, which are needed for determining the objective function values.

6.3 Start-based formulation

The start-based formulation focuses on the fact that a positive jump in the resource profiles occurs, only if one or more real activities are started. Thus, the corresponding model considers characteristics of the total adjustment cost function, as well as the idea that a time-index must not be explicitly used while specifying the decision variables (cf. the start/end event-based model in Sect. 6.2).

For all real activities $i, j \in V^r, i < j$, we make use of binary decision variables g_{ij} indicating whether activities i and j start at the same time, i.e.,

$$g_{ij} := \begin{cases} 1 \text{ or } 0, & \text{if the start of activity } i \text{ is equal to} \\ & \text{the start of activity } j \\ 0, & \text{otherwise.} \end{cases}$$

Additionally, we consider binary decision variables e_{ij} specifying if the start of real activity i and the completion of real activity j, $i \neq j$, occurs at the same point in time, i.e.,

$$e_{ij} := \begin{cases} 1 \text{ or } 0, & \text{if the start of activity } i \text{ is equal to} \\ & \text{the completion of activity } j \\ 0, & \text{otherwise.} \end{cases}$$

Both types of variables have some "degree of freedom" when more than two activities start or end at the same time. For three activities with $S_i = S_j = S_h$, $i < j < h$, the variables g_{ij}, g_{ih}, and g_{jh} could be equal to one by definition. However, the resource requirements r_{ik}, r_{jk}, and r_{hk} must be considered once to determine the correct jump difference in the resource profiles. That is always guaranteed in our model if $g_{ij} = g_{ih} := 1$ and $g_{jh} := 0$ is satisfied.

Let $S_i \in [ES_i, LS_i]$ be the start time of activity $i \in V$. If only one real activity i is started at a specific point in time, then continuous variables $\Delta^+_{ki} \geq 0$ indicate the positive adjustment of resource $k \in \mathcal{R}$ at the start time of activity i. In case that a set of real activities, e.g., activities i, j, and h, is started at the same time, $(\Delta^+_{ki} + \Delta^+_{kj} + \Delta^+_{kh}) \geq 0$ represents the positive jump in the resource profile of resource k. Using two "big-M-values" for each activity pair $\{i, j\}, i, j \in V^r$, and one "big-M-value" for each renewable resource k, the start-based formulation for the total adjustment cost problem takes the form:

$$\text{Minimize} \quad \sum_{k \in R} c_k \sum_{i \in V^r} \Delta^+_{ki} \qquad \text{(A-RL-III)}$$

subject to

$$S_j - S_i \geq \delta_{ij} \qquad \langle i, j \rangle \in A \qquad (25)$$

$$S_i - (S_j + p_j) \leq \mathrm{M}_{ij}(1 - e_{ij}) \qquad i, j \in V^r, i \neq j \qquad (26)$$

$$S_j + p_j - S_i \leq \mathrm{M}_{ij}(1 - e_{ij}) \qquad i, j \in V^r, i \neq j \qquad (27)$$

$$\sum_{\substack{i \in V^r \\ i \neq j}} e_{ij} \leq 1 \qquad j \in V^r \qquad (28)$$

$$S_i - S_j \leq \mathrm{M}'_{ij}(1 - g_{ij}) \qquad i, j \in V^r, i < j \qquad (29)$$

$$S_j - S_i \leq \mathrm{M}'_{ij}(1 - g_{ij}) \qquad i, j \in V^r, i < j \qquad (30)$$

$$\sum_{\substack{i \in V^r \\ i < j}} g_{ij} \leq 1 \qquad j \in V^r \qquad (31)$$

$$\sum_{\substack{k \in V^r \\ j < k}} g_{jk} \leq (1 - g_{ij}) \qquad i, j \in V^r, i < j \qquad (32)$$

$$\Delta^+_{ki} \geq r_{ik} + \sum_{\substack{j \in V^r \\ i < j}} r_{jk} g_{ij} - \sum_{\substack{j \in V^r \\ j \neq i}} r_{jk} e_{ij} - \sum_{\substack{j \in V^r \\ j < i}} g_{ji} \mathrm{M}_k$$

$$i \in V^r, k \in \mathcal{R} \qquad (33)$$

$$\Delta^+_{ki} \geq 0 \qquad i \in V^r, k \in \mathcal{R} \qquad (34)$$

$$S_i \in [ES_i, LS_i] \qquad i \in V \qquad (35)$$

$$g_{ij} \in \{0, 1\} \qquad i, j \in V^r, i < j \qquad (36)$$

$$e_{ij} \in \{0, 1\} \qquad i, j \in V^r, i \neq j. \qquad (37)$$

Inequalities (25) correspond to the given minimum and maximum time lags between activities. If the start time of activity i is not equal to the completion time of activity j, then decision variable e_{ij} receives the value zero—otherwise, the value might be equal to one (cf. constraints (26) and (27)). With constraints (28), it is guaranteed that every completing activity j is assigned to at most one starting activity, even if $C_j = S_i$ holds for more than one activity $i \in V^r$. Since the strongest LP-relaxation will result from choosing the smallest possible "big-M-value," we set $\mathrm{M}_{ij} := \max\{LS_i - (ES_j + p_j), LS_j + p_j - ES_i\}$. Inequalities (29) and (30) specify the values of decision variables g_{ij} in an analogous manner, where $\mathrm{M}'_{ij} := \max\{LS_i - ES_j, LS_j - ES_i\}$. If $S_i \neq S_j$ is satisfied for activities i and j, then variables g_{ij} will be set to zero—otherwise, the values might be equal to one. Constraints (31) ensure that starting activity j is assigned to at most one starting activity i. If $g_{ij} = 1$, constraints (32) guarantee that the start of any other activity $k > j$ cannot be assigned to starting activity j, i.e., only nontransitive pairs of decision variables can receive the value one, and thus each resource requirement is considered once within the calculation of Δ^+_{ki}. Inequalities (33), together with objective function (A-RL-III), make sure that the significant decision variables are set to one and the positive adjustment of resource k at the start time of real activity i is estimated correctly. In order to ensure that no resource jump is counted more than once, the big-M_k-value reduces the irrelevant decision variables $\Delta^+_{ki} \geq 0$ to zero. Constant M_k represents the maximum possible resource adjustment that may occur for resource k throughout the planning horizon (cf. Sect. 7).

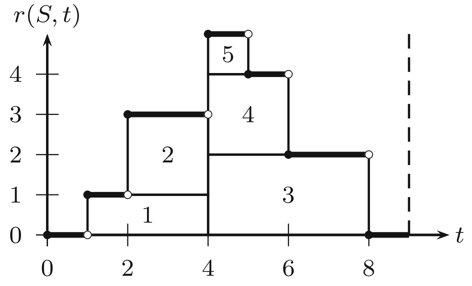

Fig. 6 Resource profile of schedule $S = (0, 1, 2, 4, 4, 4, 9)$

In order to investigate the allocation of decision variables in more detail, we consider some special cases where either several opportunities for choosing positive decision variables exist, or only an unique allocation is possible. In what follows, instances with one renewable resource are analyzed, thus, the index k can be omitted. Figure 6 depicts a resource profile with five real activities. We concentrate on the resource jump at time $t = 4$, where activities 1 and 2 are completed and activities 3, 4, 5 are started, i.e., $C_1 = C_2 = S_3 = S_4 = S_5 := 4$.

Due to constraints (26) to (32), at most two variables e_{ij} and at most two variables g_{ij} will be set to one. A possible valid assignment is $e_{31} = e_{32} := 1$ and $e_{41} = e_{51} = e_{42} = e_{52} := 0$, as well as $g_{34} = g_{35} := 1$ and $g_{45} := 0$. The allocation leads to the following system of inequalities

$$\begin{aligned}
\Delta^+_3 &\geq r_3 + r_4 g_{34} + r_5 g_{35} - r_1 e_{31} - r_2 e_{32} = 2 \\
\Delta^+_4 &\geq r_4 + r_5 g_{45} - r_1 e_{41} - r_2 e_{42} - g_{34}\,\mathrm{M} = 2 - \mathrm{M} \\
\Delta^+_5 &\geq r_5 - r_1 e_{51} - r_2 e_{52} - g_{35}\,\mathrm{M} - g_{45}\,\mathrm{M} = 1 - \mathrm{M}
\end{aligned}$$

which ensures that the correct jump difference of two resource units is considered. Another alternative for setting decision variables is, for example, $g_{34} = g_{35} = g_{45} := 0$, as well as $e_{51} = e_{42} := 1$ and $e_{31} = e_{41} = e_{32} = e_{52} = 0$. Then, the resource requirements of activities 1 and 5 as well as those of activities 2 and 4 are compensated with each other, and only r_3 has to be taken into account. Conse-

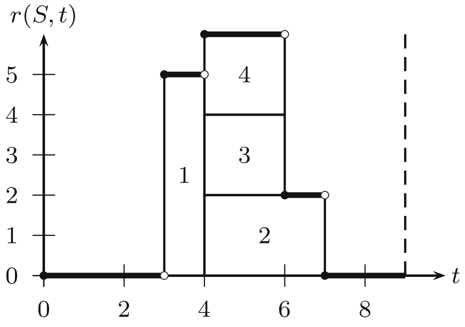

Fig. 7 Resource profile of schedule $S = (0, 3, 4, 4, 4, 9)$

quently, the continuous decision variables receive the values $\Delta_3^+ = 2,\ \Delta_4^+ = \Delta_5^+ = 0$.

Figure 7 shows a resource profile with four real activities, where activity 1 is completed at time $t = 4$, and activities 2, 3, and 4 are started at the same time, i.e., $C_1 = S_2 = S_3 = S_4 := 4$. In addition, the resource requirement of activity 1 is larger than the resource requirement of any starting activity.

A valid allocation is, for example, $g_{23} = g_{24} := 1,\ g_{34} := 0$, and $e_{21} := 1,\ e_{31} = e_{41} := 0$, which leads to

$$\begin{aligned}
&\Delta_2^+ \geq r_2 + r_3 g_{23} + r_4 g_{24} - r_1 e_{21} = 1\\
&\Delta_3^+ \geq r_3 + r_4 g_{34} - r_1 e_{31} - g_{23}\,\mathrm{M} = 2 - \mathrm{M}\\
&\Delta_4^+ \geq r_4 - r_1 e_{41} - g_{24}\,\mathrm{M} - g_{34}\,\mathrm{M} = 2 - \mathrm{M}.
\end{aligned}$$

The resulting jump difference at $t = 4$ is equal to one resource unit. Another feasible alternative for specifying positive decision variables is $g_{23} = g_{24} := 0,\ g_{34} := 1$ and $e_{31} := 1,\ e_{21} = e_{41} := 0$. For the continuous variables, we then obtain the values $\Delta_2^+ = 2, \Delta_3^+ = 0 \geq -1$, and $\Delta_4^+ = 0 \geq 2 - \mathrm{M}$. However, an adjustment of two resource units will not be considered in an optimal solution, since objective function (A-RL-III) is to be minimized, and therefore the minimal sum of resource requirements finds its way into the calculation, i.e., $\Delta_2^+ + \Delta_3^+ + \Delta_4^+ = 1$.

In order to restrict the solution space and to synchronize the binary decision variables, the following inequalities may be added:

$$e_{ij} + e_{ji} + g_{ij} \leq 1 \qquad i, j \in V^r, i < j \tag{38}$$

$$\sum_{i \in V^r} \sum_{\substack{j \in V^r \\ j > i}} g_{ij} \leq |V^r| - 1 \tag{39}$$

$$\sum_{i \in V^r} \sum_{\substack{j \in V^r \\ j \neq i}} e_{ij} \leq |V^r| - 1. \tag{40}$$

With constraints (38), the number of positive decision variables for two real activities i and j, $i < j$, is restricted to no more than one. Inequalities (39) and (40) set upper bounds on the total number of positive decision variables. Additionally, we are able to fix some binary variables in advance. If the length l_{ij} of a longest path from activity i to j or the length l_{ji} from j to i, $i, j \in V^r, i < j$, in the underlying network N is greater than zero, we can fix $g_{ij} := 0$. Furthermore, condition $e_{ij} := 0$ holds if $l_{ij} > -p_j$, and condition $e_{ji} := 0$ is satisfied if $l_{ij} > p_i,\ i, j \in V^r, i \neq j$.

The resulting model contains $\mathcal{O}(|V|^2 + |V||\mathcal{R}|)$ constraints, as well as $|V| + |V^r||\mathcal{R}|$ real-valued auxiliary and $\frac{3}{2}|V^r|^2 - \frac{3}{2}|V^r|$ binary variables. Since all numbers are independent of the scaling of the time axis, the model can be classified as a *polynomial* model.

7 Preprocessing techniques

The three MIP-formulations introduced in Sect. 6 may be improved by reducing the domains of auxiliary variables (cf. Sect. 7.1). Additionally, the solution process of a branch-and-bound or branch-and-cut approach is usually faster if a start solution is provided (cf. Sect. 7.2).

7.1 Domain reduction techniques

In all model formulations, we used continuous auxiliary variables that represent the positive adjustment of resource $k \in \mathcal{R}$ at a specific time or event. The domain of those variables may be restricted by considering lower and upper resource requirement bounds (Rieck et al. 2012). Let $P_{kt} \geq 0$ be the minimum and $H_{kt} \geq 0$ the maximum requirement for resource $k \in \mathcal{R}$ that can occur at time $t \in \{0, \ldots, \overline{d} - 1\}$ in a feasible schedule. Rieck et al. (2012) determine the values P_{kt} by analyzing the unavoidable time interval, $[LS_i, EC_i[$, for each activity $i \in V$, and the values H_{kt} by making use of the concept of antichains. With these bounds, we are able to estimate the maximum positive jump

$$\mathrm{M}_{kt} := H_{kt} - P_{k,t-1}$$

in the resource profile of resource $k \in \mathcal{R}$ at time t, where $P_{k,-1} := 0$ for all k. Moreover,

$$\mathrm{M}_k := \max_{t \in T}\{H_{kt} - P_{k,t-1}\}$$

represents the maximum positive jump that may occur for resource k throughout the planning horizon. The domains of continuous variables may then be defined as follows:

discrete time-based formulation

$$\Delta_{kt}^+ \in [0, \mathrm{M}_{kt}] \qquad k \in \mathcal{R}, t \in T \tag{41}$$

start/end event-based formulation

$$\boldsymbol{\Delta}_{ke}^+ \in [0, \mathrm{M}_k] \qquad k \in \mathcal{R}, e \in \mathcal{E} \tag{42}$$

start-based formulation

$$\Delta_{ki}^+ \in [0, \mathrm{M}_k] \qquad k \in \mathcal{R}, i \in V^r. \tag{43}$$

7.2 Generation of start solutions

For RLPs with locally concave objective functions, there invariably exists a quasistable schedule which will be optimal (cf. Sect. 4). Therefore, in order to generate good feasible solutions $S^\star$ for RLPs, quasistable schedules should be determined. This structural property is utilized by Ballestin et al. (2007) within a population-based method. The algorithm starts with a set of feasible solutions that are generated by means of a priority-based method with a serial generation scheme (Zimmermann 1997). Then, some solution is chosen with respect to its objective function value. A set of activities is removed (destruction step) and locally optimal reinserted (reconstruction step) to improve the current schedule. In a subsequent local improvement step, the early and late free float of each activity is consecutively used to optimize its position. Afterward, the quasistable characteristic of the constructed solution S is recovered by a function *quasistable*. If the resulting solution S' is better than the worst S^w in the population, then it replaces S^w. The algorithm terminates once a stop criterion is fulfilled and returns the best solution found.

We improved the Ballestin et al. (2007) algorithm to obtain a quasistable schedule S' with $f(S') \leq f(S)$ after executing function *quasistable*. Hence, in our algorithm non-quasistable schedule S is transferred into a quasistable schedule S' with lower or equal objective function value.

Let $N(O(S))$ be the order network of schedule S that results from the underlying network N by introducing arcs $\langle i, j\rangle, i, j \in V, i \neq j$ with $\delta_{ij} := p_i$ if $S_j - S_i \geq p_i$ is satisfied. Thus, arc set $A(S)$ of the order network may be specified as follows

$$A(S) := A \cup \{(i, j) \in V \times V \mid i \neq j \text{ and } S_j \geq S_i + p_i\}.$$

Neumann et al. (2003b, Sect. 3.2) have shown that every quasistable schedule may be represented by a spanning tree (i.e., a weakly connected directed graph with $|V|$ nodes and $|V|-1$ arcs) of the corresponding order network, where every arc represents a binding temporal or precedence constraint.

Since a non-quasistable schedule S does not contain $|V| - 1$ binding constraints, schedule S can be associated with a spanning forest. Each spanning forest is comprised of different components (spanning subtrees). The activities of a component $V' \subset V\backslash\{0\}$ may be shifted upward or downward until a temporal or precedence constraint becomes binding. For each component, a left- and a right-shift are executed, the first binding constraints are determined, and the constraint that leads to the lowest adjustment costs is realized. Thereby, it is ensured that a quasistable schedule S' with lower or equal objective function value is generated.

For component V', arcs $\langle i, j\rangle \in A(S)$ with $i \in V'$, $j \notin V'$, are candidates for binding constraints. In addition to and in contrast to Ballestin et al. (2007), we also consider activities $i \in V'$, $j \notin V'$, which are simultaneously in execution. For those activities, a precedence constraint can be added when activities in V' are moved. Figure 8 depicts the order network of non-quasistable schedule $S = (0, 0, 3, 5)$, where activities 1 and 2 are carried out simultaneously, and the corresponding spanning forest with component V'.

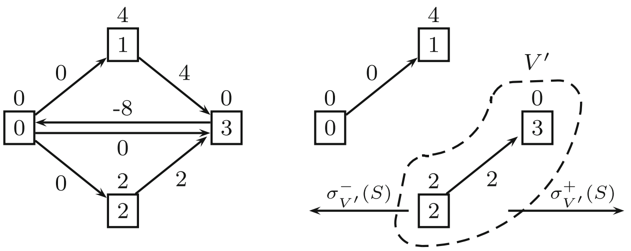

Fig. 8 Order network $N = (O(S))$ and spanning forest with component V'

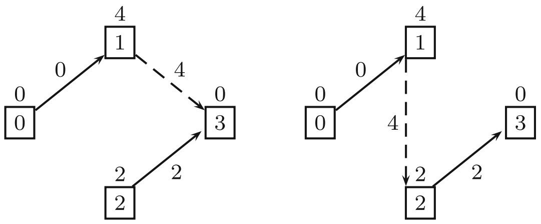

Fig. 9 Spanning trees of $S'^1 = (0, 0, 2, 4)$ and $S'^2 = (0, 0, 4, 6)$

If component $V' = \{2, 3\}$ is left-shifted, then the first temporal constraint that becomes binding is $S_3 - S_1 = \delta_{13} = 4$, and if V' is right-shifted, then the first binding precedence constraint is $S_2 - S_1 = p_1 = 4$. The spanning trees of the resulting quasistable schedules $S'^1 = (0, 0, 2, 4)$ and $S'^2 = (0, 0, 4, 6)$ are given in Fig. 9. With $c := 2$ and $r_1 = r_2 := 1$, we obtain $f(S'^1) = 4$ and $f(S'^2) = 2$, which means that schedule S'^2 is realized. Without the consideration of activities that are simultaneously in execution as it is done in Ballestin et al. (2007), component V' would be right-shifted until the backward arc $\langle 3, 0\rangle$ becomes binding. Then, a quasistable schedule S' is generated, but it is not ensured that S' has a lower or equal objective function value than initial schedule S.

In order to integrate the described capabilities, we assume that component V'_i is the one that contains activity i. Arc set $A_{V'}(S)$ of schedule S and component V', which contains all arcs between activities i and j that are candidates for binding temporal and precedence constraints, can be defined as

$$A_{V'}(S) := A(S) \cup \big\{(i, j) \mid V'_i = V' \text{or} V'_j = V',\, V'_i \neq V'_j, \\ S_i \geq S_j \text{ and } S_i < S_j + p_j\big\}.$$

Thereby, arcs $\langle i, j\rangle \in A_{V'}(S)\backslash A(S)$ obtain weights $\delta_{ij} := p_i$. The times $\sigma^-_{V'}(S)$ and $\sigma^+_{V'}(S)$ by which activities $i \in V'$ may be left- or right-shifted to generate a quasistable schedule S' with $f(S') \leq f(S)$ are then given by

$$\sigma^{-}_{V'}(S) := \min\{S_i - S_j - \delta_{ji} \mid (j,i) \in A_{V'}(S),\, i \in V',\, j \notin V'\}$$
$$\sigma^{+}_{V'}(S) := \min\{S_j - S_i - \delta_{ij} \mid (i,j) \in A_{V'}(S),\, i \in V',\, j \notin V'\}.$$

8 Performance analysis

In this section, the results of a comprehensive performance analysis are presented. In order to investigate the performance of the three MIP-formulations

- discrete time-based (TB) formulation:
  (A-RL-I), (3)–(8), (41),
- start/end event-based (EB) formulation:
  (A-RL-II), (9)–(24), (42),
- start-based (SB) formulation:
  (A-RL-III), (25)–(40), (43),

we solved small- and medium-scale problem instances to optimality using the branch-and-cut procedure provided by CPLEX 12.4 and ILOG's Concert interface for communication purposes. Results obtained by the three models are compared to the solutions of an adapted version of the tree-based branch-and-bound method developed by Gather et al. (2011) for the classical RLP. In the adapted version of the approach, lower bounds are determined in a problem-specific way. Moreover, all structural properties used within the branch-and-bound method are based on the locally concave characteristic of function (C-RL). Since function (A-RL) is locally concave as well, the algorithm is also suitable for the total adjustment cost problem. The MIP-formulations as well as the branch-and-bound method received the same initial solution or upper bound, respectively, at the beginning of the procedure. The tests were performed on an Intel Core i7 CPU 990X with 3.47 GHz and 24GB RAM under Windows 7.

The problem instances used for testing the models are introduced by Rieck et al. (2012) and contain 10, 15, 20, 30, and 50 real activities as well as 1, 3, and 5 renewable resources. The test sets are generated using control parameters that influence the behavior of solution procedures (e.g., number of maximum time lags, resource factor, and restrictiveness of Thesen; Schwindt 1998). In particular, the restrictiveness of Thesen (RT) affects the run time of MIP-formulations; therefore, it should be investigated in more detail. RT $\in [0, 1]$ measures the degree to which temporal constraints restrict the total number of feasible activity sequences, i.e., RT $= 0$ indicates a parallel network and RT $= 1$ a series network. For each combination (number $|V^r|$ of real activities/number $|\mathcal{R}|$ of renewable resources), 40 instances are considered, where the restrictiveness of Thesen is set to 0.3 and to 0.6 in equal parts. In order to determine the impact of expanding time-windows $[ES_i, LS_i]$ for activities $i \in V$ (i.e., the project slack), we tested each instance using different maximum project completion deadlines $\overline{d} := \alpha ES_{n+1}$, where $\alpha \in \{1.0, 1.5, 2.0\}$. In the following computational study, the different test sets are given by the names $|V^r| - |\mathcal{R}| - \text{RT} - \alpha$.

For every instance, an initial solution is generated (cf. Sect. 7.2) and posted to the solver. Moreover, under preliminary tests, we have investigated the effectiveness of general CPLEX-cuts during optimization. The solution process without default CPLEX-cuts performs well for the TB-model and the SB-model. For the EB-model, clique and cover cuts are added. Since the total adjustment cost problem is $\mathcal{NP}$-hard, we cannot expect that a branch-and-cut approach will terminate within a reasonable time limitation, which is why we allow maximum computation times between 3 and 6 h.

Table 1 lists the results of all MIP-formulations and the tree-based branch-and-bound method for instances with 10 real activities. Column "t_{cpu}" designates the average computation times (in seconds) and column "Inst$_{<\beta\text{h}}$" displays the number of instances solved to proven optimality within a time limit of β hours. If the optimality of an instance is not proven, then a computation time of β hours is taken into account while calculating t_{cpu}. Line "# Opt" indicates the total numbers of instances solved to proven optimality within β hours.

Regarding the average computation times, the TB-model, the SB-model, as well as the tree-based branch-and-bound method are efficient. All average run times are lower than 15 seconds. In contrast, the results of the EB-model are sobering. Only 46 out of 120 problem instances with $\alpha = 1.5$ are solved to optimality. We therefore skip pursuing further details for larger instances with more than 10 activities. The difficulties of the model result from placing the events over the time axis. For each event, uncountable possible points in time may have to be determined. Notice that the EB-formulation can be improved by setting the number of events to the project completion deadline, i.e., $E = \overline{d}$. Then, the solver develops precise combinations and assigns one integer-valued start time to each event. In that case, the formulation is converging to the TB-model, but it needs more decision variables.

In order to consider larger instances with 15 and 20 real activities, Table 2 summarizes the results for the TB-formulation, the SB-formulation, and the tree-based branch-and-bound method. We concentrate on problems with three renewable resources and varying project completion deadlines. To get an impression on the quality of the linear programming relaxation, we investigate the average gap (column "gap$_{\varnothing}$") and maximal gap (column "gap$_{\max}$") in percentage for instances for which the optimality was not proven.

For instances with RT $= 0.3$ (i.e., rather parallel networks) and tight project completion deadlines ($\alpha = 1.0$), the TB-model performs well. However, if the value α is increased, the average run times are significantly longer. In contrast, the SB-model turns out to be much more robust in

Table 1 Computation times and numbers of instances solved ($n = 10$)

Instances	TB-model		EB-model		SB-model		Tree-based B&B	
	t_{cpu}	$\text{Inst}_{<3\text{h}}$	t_{cpu}	$\text{Inst}_{<3\text{h}}$	t_{cpu}	$\text{Inst}_{<3\text{h}}$	t_{cpu}	$\text{Inst}_{<3\text{h}}$
10-1-0.3-1.0	0.02	20	152.14	20	0.10	20	0.06	20
10-3-0.3-1.0	0.03	20	531.36	20	0.05	20	0.12	20
10-5-0.3-1.0	0.04	20	463.98	20	0.12	20	0.52	20
10-1-0.3-1.5	0.43	20	5,696.74	11	0.20	20	1.15	20
10-3-0.3-1.5	7.89	20	7,864.63	8	0.24	20	3.52	20
10-5-0.3-1.5	5.44	20	10,152.37	5	0.65	20	14.21	20
10-1-0.6-1.0	0.02	20	119.50	20	0.03	20	0.01	20
10-3-0.6-1.0	0.01	20	214.32	20	0.03	20	0.02	20
10-5-0.6-1.0	0.02	20	631.61	20	0.06	20	0.02	20
10-1-0.6-1.5	0.53	20	5,951.83	15	0.09	20	0.34	20
10-3-0.6-1.5	2.96	20	9,846.03	4	0.05	20	0.57	20
10-5-0.6-1.5	12.01	20	9,405.33	3	0.13	20	1.80	20
# Opt		240		166		240		240

Table 2 Computation times and the numbers of instances solved ($n = 15, 20$)

Instances	TB-model				SB-model				Tree-based B&B	
	t_{cpu}	$\text{Inst}_{<3\text{h}}$	$\text{gap}_{\varnothing}$	$\text{gap}_{\max}$	t_{cpu}	$\text{Inst}_{<3\text{h}}$	$\text{gap}_{\varnothing}$	$\text{gap}_{\max}$	t_{cpu}	$\text{Inst}_{<3\text{h}}$
15-3-0.3-1.0	0.21	20	0	0	1.63	20	0	0	109.29	20
15-3-0.3-1.5	794.18	19	11	11	37.36	20	0	0	7,982.33	9
20-3-0.3-1.0	8.13	20	0	0	80.81	20	0	0	6,413.25	9
20-3-0.3-1.5	5,061.66	13	44	66	2,311.72	19	45	45	10,800.00	0
15-3-0.6-1.0	0.05	20	0	0	0.17	20	0	0	8.85	20
15-3-0.6-1.5	1,358.29	19	53	53	1.09	20	0	0	727.47	20
20-3-0.6-1.0	0.08	20	0	0	0.62	20	0	0	644.59	20
20-3-0.6-1.5	5,494.30	11	37	75	8.67	20	0	0	9,878.78	2
# Opt		142				159				100

relation to the deadlines for project completion. All instances with 15, and nearly all instances with 20 real activities are solved to optimality within 3 h. Furthermore, the tree-based branch-and-bound method is not able to stick with the branch-and-cut procedures provided by CPLEX. Less than 50 % of the problem instances are optimally solved. Considering a rather series network with RT $= 0.6$, the SB-formulation is able to strengthen its success, and all instances are solved in less than 30 seconds. In addition, the performance of the tree-based branch-and-bound method is better than that for instances with RT $= 0.3$. However, the average run times are much higher than those of the two models. In order to arrive at fair comparisons, the tree-based method might have run longer than 3 h (the program uses just one processor core). Nevertheless, if we consider the instances solved to proven optimality, the run times of the SB-model and the tree-based method differ by a factor of more than 60, which is above a computation time reduction that a multicore processor running CPLEX can induce. For the TB-model, the restrictiveness of Thesen has no obvious impact on the run times and the number of instances solved. The procedure obtains similar results for both RT-values. As expected, the restrictiveness of Thesen influences the run times of procedures that take advantage of problem structures during the solution process (i.e., the SB-model as well as the tree-based method). As a rule, a lower value for the restrictiveness of Thesen complicates the problem of finding and proving an optimal schedule. Rather parallel networks involving RT $= 0.3$ induce a large number of feasible activity sequences, whereas in rather series networks, many (indirect) precedence constraints lead to a reduction of feasible activity sequences. For instances not optimally solved, the average and maximal gaps are worse, i.e., $\text{gap}_{\varnothing} \geq 11$ % and $\text{gap}_{\max} \geq 11$ %.

We investigate the robustness of the TB-model and the SB-model in relation to varying RT-values and expanded

Table 3 Computation times and the numbers of instances solved ($n = 15, 20$ and $\alpha = 2.0$)

Instances	TB-model				SB-model			
	t_{cpu}	$\mathrm{Inst}_{<3\mathrm{h}}$	$\mathrm{gap}_{\varnothing}$	$\mathrm{gap}_{\max}$	t_{cpu}	$\mathrm{Inst}_{<3\mathrm{h}}$	$\mathrm{gap}_{\varnothing}$	$\mathrm{gap}_{\max}$
15-3-0.3-2.0	4,413.34	13	30	48	85.71	20	0	0
20-3-0.3-2.0	9,296.03	5	58	89	4,294.86	16	49	52
15-3-0.6-2.0	5,396.44	14	41	77	1.32	20	0	0
20-3-0.6-2.0	8,998.53	4	64	90	11.65	20	0	0
# Opt		36				76		

Table 4 Computation times and the numbers of instances solved ($n = 30, 50$)

Instances	TB-model				SB-model			
	t_{cpu}	$\mathrm{inst}_{<6\mathrm{h}}$	$\mathrm{gap}_{\varnothing}$	$\mathrm{gap}_{\max}$	t_{cpu}	$\mathrm{inst}_{<6\mathrm{h}}$	$\mathrm{gap}_{\varnothing}$	$\mathrm{gap}_{\max}$
30-3-0.3-1.0	120.04	20	0	0	12,470.24	12	44	65
30-3-0.3-1.5	21,600.00	0	58	82	17,040.97	1	76	98
50-3-0.3-1.0	7,304.88	7	16	37	21,600.00	0	78	99
30-3-0.6-1.0	2.04	20	0	0	419.92	20	0	0
30-3-0.6-1.5	21,600.00	0	52	88	6,901.77	16	29	40
50-3-0.6-1.0	2,931.97	16	12	26	14,900.54	7	40	81
# Opt		63				56		

project completion deadlines by setting $\alpha := 2.0$ for instances involving 15 or 20 real activities. Table 3 depicts the results.

The SB-formulation indeed works very well. Only four instances with RT $= 0.3$ are not proven optimally solved; for these instances, the average and maximum gap are larger than 49 %. Particularly, the run times for instances with RT $=$ 0.6 are shorter than expected. As opposed to this, the TB-model is only able to solve less than 50 % of all problem instances to optimality within 3 h. Thus, the model, due to its nonpolynomial characteristic (cf. Sect. 6.1), is not suitable for instances involving large project completion deadlines.

Table 4 shows the results for medium-scale instances with a maximum of 50 real activities and project deadlines $\overline{d} = \alpha ES_{n+1}$, $\alpha = \{1.0, 1.5\}$. Here, the boundaries of an exact solution methodology become apparent.

Within a time limit of 6 h, about half of all instances are optimally solved. Moreover, the strength of the TB-formulation appears by regarding the results for instances with tight project completion deadlines. In contrast, for projects with $\alpha = 1.5$, the SB-model produces significantly better results than the TB-model.

In what follows, we perform some computational studies on instances with 15 and 20 real activities to generate additional insights into the behavior of MIP-formulations. First, to evaluate the quality of heuristic solutions that are used as start solutions, we measure the objective function value differences of start and optimal solutions or best solutions found. The results show that the optimal solutions (or best solutions found) are on average less than 1 % better than the start solutions. Therefore, the formulations usually spend the whole time on the verification of the optimality of a solution. For the TB-model, an average run time improvement of 20 % is obtained if a start solution is posted to the solver. Thus, a "warm" start helps the formulation to find and prove more optimal solutions within the time limitation. For the SB-model and a start solution, no significant run time improvement can be detected.

In order to examine the impact of large activity resource requirements, we solved instances with 15 and 20 real activities, three renewable resources, $\alpha \in \{1.0, 1.5\}$, and $r_{ik} := \gamma\, r_{ik}$ with $\gamma \in \{5, 10\}$. As the resource requirements only affect the M_k or M_{kt} values, as well as the domains of auxiliary variables in the models, no explicit differences concerning performance may be determined by analyzing the results.

The previous tables demonstrate that the average run times increase and the numbers of optimal solutions found decrease for the SB-model, when varying the RT-factor from 0.6 to 0.3. However, RT $= 0.3$ does not correspond to a "significant" parallel network. To determine the impact of smaller RT-values, we generated instances with 15 and 20 real activities, three renewable resources, $\alpha \in \{1.0, 1.5, 2.0\}$, and RT $\in \{0.1, 0.2\}$. Table 5 shows the computation times and the numbers of instances solved for the TB-model and the SB-model.

As could be supposed from Table 2, the TB-model does not depend on the RT-factor. For the SB-model and RT $= 0.2$,

Table 5 Computation times and the numbers of instances solved ($n = 15$, 20 and RT = 0.1, 0.2)

Instances	TB-model				SB-model			
	t_{cpu}	Inst$_{<3\text{h}}$	gap$_{\emptyset}$	gap$_{\max}$	t_{cpu}	Inst$_{<3\text{h}}$	gap$_{\emptyset}$	gap$_{\max}$
15-3-0.1-1.0	1.26	20	0	0	173.98	20	0	0
15-3-0.1-1.5	90.19	20	0	0	3,213.22	19	89	89
15-3-0.1-2.0	1,507.30	20	0	0	6,581.81	13	93	97
20-3-0.1-1.0	43.23	20	0	0	8,523.92	6	36	75
20-3-0.1-1.5	4,796.01	14	19	31	10,800.00	0	83	98
20-3-0.1-2.0	9,583.19	3	28	55	10,800.00	0	94	99
15-3-0.2-1.0	0.31	20	0	0	16.78	20	0	0
15-3-0.2-1.5	168.73	20	0	0	524.14	20	0	0
15-3-0.2-2.0	3,358.14	18	19	32	1,141.57	20	0	0
20-3-0.2-1.0	9.96	20	0	0	2,468.39	18	16	18
20-3-0.2-1.5	5,539.11	12	39	64	10,096.54	2	50	79
20-3-0.2-2.0	9,411.47	4	45	79	10,599.25	1	63	82
# Opt		191				139		

all instances with 15 activities and some instances with 20 activities are solved to optimality. In the event of RT $= 0.1$, the SB-model is not able to terminate the enumeration for all instances with 15 activities. Moreover, it collapses for instances with 20 activities. Considering both models, the average (maximum) gap for instances for which the optimality was not proven is always larger than 16 % (18 %). Furthermore, the gaps increase significantly for large project completion deadlines. Thus, the linear programming relaxation is the main drawback in the solution process. If better lower bounds could be generated during the optimization, then the computation times would certainly be improved.

9 Conclusions

The paper presented above considers the total adjustment cost problem which has received little attention in the literature pertaining to resource leveling. The problem and its objective function have nice structural properties that could be exploited successfully to obtain a feasible or optimal solution. Particularly, the total adjustment cost function is locally quasiconcave and may be determined by considering the start and end times of activities. Furthermore, we presented some interesting practical applications that can be usefully employed in the construction industry. In addition, a discrete time-based model and two polynomial model formulations, an event-based model and a start-based model, are introduced. The main advantage of the polynomial models is that they are independent of a scaling of the time axis. Small-scale and medium-scale instances are solved to optimality using CPLEX. In order to facilitate the solution process, preprocessing techniques are used, and a start solution is transferred to the solver.

The results of a comprehensive performance analysis show that the discrete time-based formulation is efficient for instances with tight project completion deadlines. The corresponding branch-and-cut procedure is able to solve instances with up to 50 real activities. Even in cases, where the project deadlines are increased, the start-based model performs well. There are even many instances with 30 real activities and $\alpha = 1.5$ that are solved to optimality. The two other exact methods are found to be of little value as alternative solution approaches.

Future research will include the consideration of the total adjustment cost problem in combination with further real-life conditions. In that context, it should be included that activity durations may be stochastic or that activities may be carried out in many alternative execution modes which differ in processing time, time lags, and resource requirements (see, e.g., Hartmann and Briskorn 2010). Furthermore, a simultaneously consideration of (A-RL), (C-RL), and (O-RL) in a multi-criteria approach could be interesting. Since all objective functions are locally quasiconcave, there always exists a quasistable schedule that will be optimal for a resulting weighted objective function (cf. Sect. 4). Therefore, the heuristic algorithm (cf. Sect. 7.2), the tree-based branch-and-bound method, and the TB-model can be extended to a multi-criteria approach. In addition, the study of a quadratic variant of the total adjustment cost objective function, which is no longer locally concave, is an interesting topic.

Acknowledgments The benchmarks for the total adjustment cost problem presented herein and the results obtained (i.e., upper and lower bounds) may be downloaded from http://www.wiwi.tu-clausthal.de/abteilungen/unternehmensforschung/forschung.

References

Ahuja, H. N. (1976). *Construction performance control by networks*. New York: Wiley.

Ahuja, R., Magnanti, T., & Orlin, J. (1993). *Network flows*. Englewood Cliffs: Prentice Hall.

Al-Joma, M. & Mangin, J.-C. (2004). A scheduling model for repetitive activities. In F. Khosrowshahi, (Ed.), *Proceedings 20th Annual Association of Researchers in Construction Management (ARCOM) Conference* (pp. 1197–1206), Edinburgh.

Ballestin, F., Schwindt, C., & Zimmermann, J. (2007). Resource leveling in make-to-order production: Modeling and heuristic solution method. *International Journal of Operations Research*, *13*(2), 76–83.

Bandelloni, M., Tucci, M., & Rinaldi, R. (1994). Optimal resource leveling using non-serial dynamic programming. *European Journal of Operational Research*, *78*, 162–177.

Bartusch, M., Möhring, R., & Radermacher, F. (1988). Scheduling project networks with resource constraints and time windows. *Annals of Operations Research*, *16*, 201–240.

Blaskova, M., & Grazulis, V. (2009). *Motivation of human potential: Theory and practice*. Vilnius, Lithuania: Publishing Centre of Mykolas Romeris University.

Burgess, A., & Killebrew, J. (1962). Variation in activity level on a cyclical arrow diagram. *Journal of Industrial Engineering*, *13*, 76–83.

Demeulemeester, E., & Herroelen, W. (2002). *Project scheduling: A research handbook*. Boston: Kluwer.

Easa, S. M. (1989). Resource leveling in construction by optimization. *Journal of Construction Engineering and Management*, *115*, 302–316.

Galbreath, R. V. (1965). Computer program for leveling resource usage. *Journal of the Construction Division, Proceedings of the American Society of Civil Engineers*, *91*, 107–124.

Gather, T., Zimmermann, J., & Bartels, J.-H. (2011). Exact methods for the resource levelling problem. *Journal of Scheduling*, *14*(6), 557–569.

Geng, J., Weng, L., & Liu, S. (2011). An improved ant colony optimization algorithm for nonlinear resource-leveling problems. *Computers and Mathematics with Applications*, *61*, 2300–2305.

Harris, R. B. (1978). *Precedence and arrow networking techniques for construction*. New York: Wiley.

Harris, R. B. (1990). Packing method for resource leveling (Pack). *Journal of Construction Engineering and Management*, *116*, 39–43.

Harris, R. B., & Ioannou, P. G. (1998). Scheduling projects with repeating activities. *Journal of Construction Engineering and Management*, *124*, 269–279.

Hartmann, S., & Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of Operational Research*, *207*, 1–14.

Haubrich, H.-J., Siemes, P., & Ohrem, S. (2008). Möglichkeiten der Netzintegration von Offshore-Großwindanlagen. In P. Dietz (Ed.), *Netzintegration von Offshore-Großwindanlagen: Grundlast von der Nordsee* (pp. 167–177). Clausthal-Zellerfeld: Papierflieger.

Koné, O., Artigues, C., Lopez, P., & Mongeau, M. (2011). Event-based MILP models for resource-constrained project scheduling problems. *Computers and Operations Research*, *38*, 3–13.

Moder, J. J., & Phillips, C. R. (1970). *Project Management with CPM and PERT* (2nd ed.). New York: Van Nostrand Reinhold Company.

Neumann, K., Schwindt, C., & Zimmermann, J. (2003a). Order-based neighborhoods for project scheduling with nonregular objective functions. *European Journal of Operational Research*, *149*, 325–343.

Neumann, K., Schwindt, C., & Zimmermann, J. (2003b). *Project scheduling with time windows and scarce resources* (2nd ed.). Berlin: Springer.

Neumann, K., Schwindt, C., & Zimmermann, J. (2006). Resource-constrained project scheduling with time windows. In J. Józefowska & J. Weglarz (Eds.), *Perspectives in modern project scheduling* (pp. 375–407). New York: Springer.

Neumann, K., & Zimmermann, J. (1999). Resource levelling for projects with schedule-dependent time windows. *European Journal of Operational Research*, *117*, 591–605.

Neumann, K., & Zimmermann, J. (2000). Procedures for resource levelling and net present value problems in project scheduling with general temporal and resource constraints. *European Journal of Operational Research*, *127*, 425–443.

Nübel, H. (2001). The resource renting problem subject to temporal constraints. *OR Spektrum*, *23*, 359–381.

Petrovic, R. (1969). On optimization of resource leveling in project plans. In H. J. Lombaers (Ed.), *Project Planning by Network Analysis: Proceedings of the Second International Congress* (pp. 268–273). Amsterdam, North-Holland.

Pritsker, A. A. B., Watters, L. J., & Wolfe, P. M. (1969). Multi-project scheduling with limited resources: A zero-one programming approach. *Management Science*, *16*, 93–108.

Raja, K., & Kumanan, S. (2007). Resource leveling using petrinet and memetic approach. *American Journal of Applied Science*, *4*(5), 317–322.

Rieck, J., Zimmermann, J., & Gather, T. (2012). Mixed-integer linear programming for resource leveling problems. *European Journal of Operational Research*, *221*, 27–37.

Savin, D., Alkass, S., & Fazio, P. (1997). A procedure for calculating the weight-matrix of a neural network for resource leveling. *Advances in Engineering Software*, *28*, 277–283.

Schwindt, C. (1998). Generation of resource-constrained project scheduling problems subject to temporal constraints. WIOR-Report 543, Institute for Economic Theory and Operations Research, University Karlsruhe, Germany.

Schwindt, C. (2005). *Resource allocation in project management*. Berlin: Springer.

Steinboeck, A., Wild, D., & Kugi, A. (2013). Nonlinear model predictive control of a continuous slab reheating furnace. *Control Engineering Practice*, *21*, 495–508.

Takamoto, M., Yamada, N., Kobayashi, Y., & Nonaka, H. (1995). Zero-one quadratic programming algorithm for resource leveling of manufacturing process schedules. *Systems and Computers in Japan*, *26*(10), 68–76.

Younis, M. A., & Saad, B. (1996). Optimal resource leveling of multi-resource projects. *Computers and Industrial Engineering*, *31*, 1–4.

Zhuchkov, S. M. (2004). Conserving energy and resources in the production of rolled sections and wire rod. *Metallurgist*, *48*, 528–535.

Zimmermann, J. (1997). Heuristics for resource levelling problems in project scheduling with minimum and maximum time lags. WIOR-Report 491, Institute for Economic Theory and Operations Research, University Karlsruhe, Germany.

Zimmermann, J. (2001). *Ablauforientiertes Projektmanagement: Modelle Verfahren und Anwendungen*. Wiesbaden: Gabler.