Invited Review

Recent Results on Resource-Constrained Project Scheduling with Time Windows: Models, Solution Methods, and Applications*

Klaus Neumann[1], **Christoph Schwindt**[1], **Jürgen Zimmermann**[2]

[1] Institute for Economic Theory and Operations Research, University of Karlsruhe, D-76128 Karlsruhe, Germany,
(e-mail: {neumann,schwindt}@wior.uni-karlsruhe.de)

[2] Institute of Economic Theory, Technical University of Clausthal, D-38678 Clausthal-Zellerfeld, Germany, (e-mail: juergen.zimmermann@tu-clausthal.de)

Abstract This paper surveys recent results on deterministic project scheduling with general temporal constraints, limited resources, and different kinds of regular or nonregular objective functions. First, we formulate a "basic project scheduling problem" and sketch an order-based structural analysis of the feasible region of that basic problem. Second, we classify regular and nonregular objective functions important in practice. Third, we briefly discuss exact and heuristic solution methods, where we exploit the structural results presented. Also, we show how to adapt those solution procedures to modifications of our basic project scheduling problem: break calendars for renewable resources, scheduling problems with different kinds of resources (renewable and cumulative resources), renewable resources with sequence-dependent changeover times, and so-called multi-mode project scheduling problems, where in addition nonrenewable resources are taken into account. Finally, we study applications of resource-constrained project scheduling to make-to-order production in manufacturing, batch scheduling in process industries, and investment projects.

Key words Deterministic project scheduling – regular and nonregular objective functions – exact and heuristic solution methods – make-to-order production – batch production – investment projects

* This work has been partially supported by the Deutsche Forschungsgemeinschaft under Grant Ne 137/4.

1 Introduction

In literature on project management, a project is generally viewed as a one-time undertaking with a specific objective. A project can be divided into subtasks (activities), which require time, personnel, and equipment for their execution. The life cycle of a project consists of the five phases conception, definition, planning, execution, and termination (cf. Klein 2000, Sect. 1.2). In the conception phase, a feasibility study and an economic and risk analysis are performed to decide on the execution of the project. The definition phase specifies the project objective, the organizational form, major tasks and corresponding milestones, and the resources assigned to the project. The first step of the planning phase is the structural analysis, where the project activities and temporal relationships among them are identified. The subsequent time, resource, and cost analyses provide the processing time, the resource requirements, and the payment associated with each activity. Moreover, the time analysis quantifies the temporal constraints between activities. Next, the temporal scheduling step, where the resource constraints are disregarded, yields the earliest and latest start and completion times and the slack times of the activities as well as the critical activities and the shortest project duration. In the last step of the planning phase, the activities are scheduled subject to the temporal and resource constraints and the objective specified. The resulting schedule is the basis for the project implementation. In the following execution phase, the progress of the project is monitored against the schedule, which generally results in an repeated updating of the project schedule. The final termination phase evaluates and reviews the project in support of future decisions.

For managerial aspects of the five phases of the project life cycle, we refer to Meredith and Mantel (1999), Sect. 1.3, and Kerzner (2000), Sect. 2.7. This paper is concerned with models and methods for the planning phase. More specifically, we deal with resource-constrained project scheduling problems, where scarce resources have to be allocated over time to the execution of the project activities such that all temporal constraints are observed and some objective function is minimized. We only treat the case of deterministic project scheduling, where all input data are assumed to be deterministically known in advance. An overview of different types of stochastic project scheduling problems can be found in Neumann (1999) and Möhring (2000). In recent years, several survey papers have reviewed new results in deterministic project scheduling (see, e.g., Elmaghraby 1995, Özdamar and Ulusoy 1995, Herroelen et al. 1998, Brucker et al. 1999, or Kolisch and Padman 2001). The present paper differs from those survey papers by

(*i*) emphasizing on the case of general temporal constraints given by minimum and maximum time lags between project activities,

(*ii*) discussing and classifying a great variety of regular and nonregular objective functions,

(*iii*) presenting a structural analysis of the feasible region that can be exploited for developing efficient solution procedures, and

(*iv*) studying applications of project scheduling to production and operations management and investment projects.

The paper is organized as follows. In Section 2, a basic project scheduling problem with limited renewable resources is formulated. In Section 3, we study structural properties of the feasible region of our basic project scheduling problem. We will see that the feasible region is generally disconnected and represents the union of finitely many so-called order polytopes. Several different classes of objective functions important from a theoretical or practical point of view are discussed in Section 4. Also, we state which specific points of the feasible region represent possible optimal solutions to the corresponding project scheduling problems. Section 5 is concerned with exact solution methods of the branch-and-bound type, where we exploit the structural results from Section 4. Heuristic solution procedures are dealt with in Section 6. In particular, we briefly sketch truncated branch-and-bound procedures, priority-rule methods, and schedule-improvement techniques and summarize the results of an experimental performance analysis. Section 7 focuses on several generalizations of the basic project scheduling problem. We study the case of calendarization, where renewable resources are not available during certain time intervals and interruptible and non-interruptible activities occur. Also, we discuss so-called cumulative resources, which represent storage facilities needed between different production stages in process industries. Moreover, we deal with the case where there are sequence-dependent changeover times between the processing of successive activities on certain resources. Finally, we are concerned with multi-mode scheduling problems with renewable and nonrenewable resources, where each activity may be carried out in one out of several alternative execution modes. In Section 8, we present applications of resource-constrained project scheduling to make-to-order production in manufacturing, batch scheduling in process industries, and investment projects. Section 9 is devoted to concluding remarks.

2 Basic project scheduling problem

We consider a project that consists of n *activities* $i = 1, \ldots, n$ with *duration* or *processing time* $p_i \in \mathbb{N}$ each of which is carried out without interruption. In addition, we introduce the dummy activities 0 and $n+1$ where $p_0 = p_{n+1} = 0$, which represent the beginning and completion, respectively, of the project. Then $V = \{0, 1, \ldots, n, n+1\}$ is the set of all activities.

Let $S_i \geq 0$ be the *start time* of activity $i \in V$, where $S_0 = 0$. Thus, the project always begins at time 0, and S_{n+1} represents the *project duration*. We assume that $S_{n+1} \leq \overline{d}$, where $\overline{d}$ is a prescribed maximum project duration. A sequence of start times $S = (S_0, S_1, \ldots, S_{n+1})$ with $S_0 = 0$ is called a *schedule*.

In project scheduling it is expedient to assign a *network* to the project under consideration. To do so, we identify the activities $0, 1, \ldots, n+1$ of the

project with the nodes $0, 1, \ldots, n+1$ of a network. If there is a prescribed *minimum time lag* $d_{ij}^{min} \in \mathbb{Z}_{\geq 0}$ between the start of two different activities i and j, i.e.,

$$S_j - S_i \geq d_{ij}^{min} \tag{1}$$

we introduce an arc $\langle i, j \rangle$ with weight $\delta_{ij} = d_{ij}^{min}$ (see Figure 1). In the special case $d_{ij}^{min} = p_i$, (1) is called a *precedence constraint*. If there is a given *maximum time lag* $d_{ij}^{max} \in \mathbb{Z}_{\geq 0}$ between the start of activities i and j, i.e.,

$$S_j - S_i \leq d_{ij}^{max} \tag{2}$$

we introduce a backward arc $\langle j, i \rangle$ with weight $\delta_{ji} = -d_{ij}^{max}$ (cf. Figure 1). In particular, we assume that there is a maximum time lag $d_{0,n+1}^{max} = \overline{d}$ between the project beginning and project completion, which guarantees that the project is terminated by the prescribed maximum project duration $\overline{d}$.

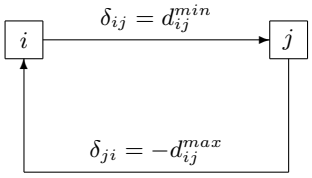

Fig. 1 Minimum and maximum time lags

In this way, a unique *activity-on-node project network* N with node set $V = \{0, 1, \ldots, n+1\}$, arc set E, and arc weights δ_{ij} ($\langle i, j \rangle \in E$) can be assigned to the project in question (cf. Neumann and Schwindt 1997, Neumann et al. 2001, Sect. 1.2). Due to the presence of maximum time lags in addition to minimum ones, the project network N contains cycles. The two types of inequalities (1) and (2) can be combined in the *temporal constraints*

$$S_j - S_i \geq \delta_{ij} \quad (\langle i, j \rangle \in E) \tag{3}$$

A schedule S which satisfies (3) is called *time-feasible*. The set of time-feasible schedules is denoted by $\mathcal{S}_T$. It holds that $\mathcal{S}_T \neq \emptyset$ exactly if network N does not contain any cycle of positive length (cf. Roy 1964, Neumann 1975, Sect. 6.4).

Now assume that a set $\mathcal{R}$ of *renewable resources* are required for carrying out the activities of the project. Renewable resources are available at each point in time during the execution of the project independently of their utilization formerly (for example, machines, manpower, or equipment). Let $R_k \in \mathbb{N}$ be the capacity of renewable resource k available and $r_{ik} \in \mathbb{Z}_{\geq 0}$ be the amount of resource k used by activity i, where $r_{ik} \leq R_k$ ($i \in V,\ k \in \mathcal{R}$).

Given a schedule $S = (S_i)_{i \in V}$,

$$\mathcal{A}(S, t) := \{i \in V \mid S_i \leq t < S_i + p_i\}$$

is the set of activities in progress, also called the *active set*, at time $t \in [0, \overline{d}]$.

$$r_k(S,t) := \sum_{i \in \mathcal{A}(S,t)} r_{ik}$$

is then the amount of resource $k \in \mathcal{R}$ used at time $t \in [0, \overline{d}]$ given schedule S. The *resource constraints* say that

$$r_k(S,t) \leq R_k \quad (k \in \mathcal{R},\ 0 \leq t \leq \overline{d}) \tag{4}$$

A schedule S satisfying (4) is called *resource-feasible*. A schedule which is both resource- and time-feasible is called *feasible*. The set of feasible schedules is denoted by $\mathcal{S}$.

The *basic project scheduling problem* to be studied in what follows consists of minimizing some objective function $f : \mathbb{R}_{\geq 0}^{n+2} \to \mathbb{R}$ on the set $\mathcal{S}$ of feasible schedules or in more detail,

$$\left.\begin{array}{lll} \text{Min. } f(S) & & \\ \text{s.t.} & S_j - S_i \geq \delta_{ij} & (\langle i,j \rangle \in E) \\ & S_i \geq 0 & (i \in V) \\ & S_0 = 0 & \\ & r_k(S,t) \leq R_k & (k \in \mathcal{R},\ 0 \leq t \leq \overline{d}) \end{array}\right\} \quad \text{(P)}$$

A feasible schedule S which minimizes objective function f on $\mathcal{S}$ is called *optimal*. We assume that f is lower semicontinuous, i.e., $f(S) \leq \liminf_{S' \to S} f(S')$ for all schedules S. Since the feasible region $\mathcal{S}$ of (P) is compact, f thus attains its minimum on $\mathcal{S}$ provided that $\mathcal{S} \neq \emptyset$. All objective functions f discussed subsequently are lower semicontinuous but not necessarily continuous.

3 Structural questions

Obviously, the set of time-feasible schedules $\mathcal{S}_T$ represents a (convex) polytope. The set of feasible schedules $\mathcal{S}$, however, is generally disconnected and represents the union of finitely many polytopes, and the decision problem whether or not $\mathcal{S} \neq \emptyset$ is NP-complete (cf. Bartusch et al. 1988). Next, we review an order-based structural analysis of $\mathcal{S}$ (cf. Neumann et al. 2000, 2001, Sect. 2.3, and Zimmermann 2001, Sect. 2.1), which will turn out to be useful for solving the basic project scheduling problem (P) with different objective functions f.

Let $O \subset V \times V$ be a *strict order* (that is, an asymmetric and transitive relation) in activity set V. Then

$$\mathcal{S}_T(O) := \{S \in \mathcal{S}_T \mid S_j \geq S_i + p_i \text{ for all } (i,j) \in O\}$$

is called the corresponding *order polytope*. Strict order O is said to be *time-feasible* if $\mathcal{S}_T(O) \neq \emptyset$, and *feasible* if $\emptyset \neq \mathcal{S}_T(O) \subseteq \mathcal{S}$. Recall that a feasible

strict order O in V is termed *inclusion-minimal* if there is no feasible strict order O' in V with $O' \subset O$. Then the *basic structural theorem*, which was first proved by Bartusch et al. (1988), is as follows:

Theorem 1 *Let $\mathcal{O}$ be the (finite) set of all inclusion-minimal feasible strict orders in activity set V. Then $\mathcal{S} = \bigcup_{O \in \mathcal{O}} \mathcal{S}_T(O)$.*

For the project network shown in Figure 2 and a given resource capacity of $R = 3$, Figure 3 illustrates a covering of the feasible region $\mathcal{S}$ by seven inclusion-maximal order polytopes $\mathcal{S}_T(O)$, cf. Neumann et al. (2000).

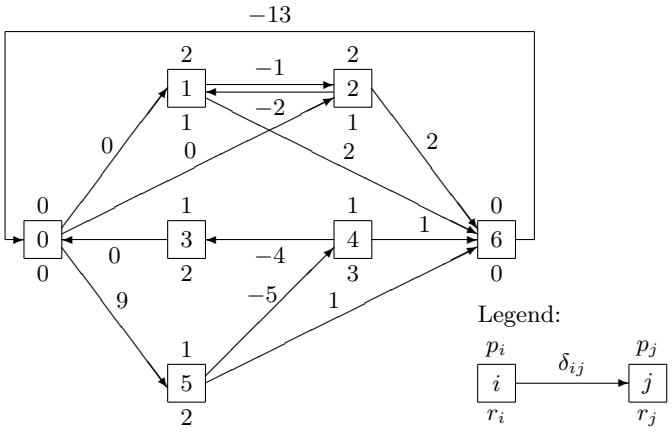

Fig. 2 Project network N with a single resource

Given a schedule $S \in \mathcal{S}_T$,

$$O(S) := \{(i,j) \in V \times V \mid i \neq j,\ S_j \geq S_i + p_i\}$$

is the strict order induced by schedule S. The corresponding order polytope $\mathcal{S}_T(O(S))$ is called the *schedule polytope* of S. $\mathcal{S}_T(O(S))$ is the set of all time-feasible schedules that belong to the *schedule network* $N(O(S))$, which results from the underlying project network N by adding an arc $\langle i,j \rangle$ with weight p_i for each $(i,j) \in O(S)$. If the latter arc $\langle i,j \rangle$ already belongs to N, then its arc weight δ_{ij} is replaced by $\max(p_i, \delta_{ij})$.

Sometimes we have to consider distinct schedules S that give rise to the same precedence constraints $S_j \geq S_i + p_i$ and thus induce the same strict order $O(S)$. We then define

$$\mathcal{S}_T^{=}(O(S)) := \{S' \in \mathcal{S}_T(O(S)) \mid O(S') = O(S)\}$$

Set $\mathcal{S}_T^{=}(O(S))$ is called the *equal-order set* for schedule S and represents a polytope generally without a part of its boundary. Since $S \in \mathcal{S}_T^{=}(O(S))$ for each $S \in \mathcal{S}$ and there are only finitely many distinct strict orders $O(S)$, we obtain $\mathcal{S} = \bigcup_{S \in \mathcal{S}} \mathcal{S}_T^{=}(O(S))$ and thus a finite partition of feasible region $\mathcal{S}$.

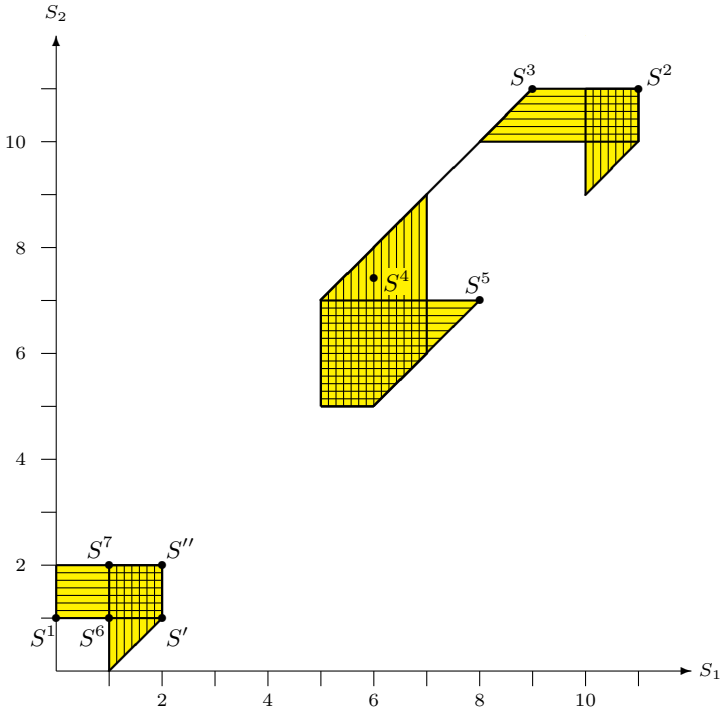

Fig. 3 Covering of the feasible region $\mathcal{S}$ by seven order polytopes

Special points of $\mathcal{S}$ are of particular interest and will turn out to be possible optimal solutions to basic project scheduling problem (P) for specific objective functions f. Let $\mathcal{M}$ be a compact subset of $\mathbb{R}^{n+2}$. Recall that $S \in \mathcal{M}$ is a *minimal point* (or a *maximal point*) of $\mathcal{M}$ exactly if there is no $S' \in \mathcal{M}$, $S' \neq S$, with $S \leq S'$ (or $S' \geq S$, respectively) where "$\leq$" is meant componentwise. $S \in \mathcal{M}$ is an *extreme point* of $\mathcal{M}$ precisely if S does not lie on a line segment that joins two other points of $\mathcal{M}$. $S \in \mathcal{M}$ is said to be a *local extreme point* of $\mathcal{M}$ if S does not lie on a line segment that joins two other points of $\mathcal{M}$ and totally belongs to $\mathcal{M}$. Finally, a schedule S is called *locally order-optimal* for objective functions f if S is a local minimizer of f on the order polytope $\mathcal{S}_T(O)$ of some $O \in \mathcal{O}$. In Figure 3, S^1 is a minimal point of $\mathcal{S}$, S^2 is a maximal point of $\mathcal{S}$, S^3 is an extreme point of $\mathcal{S}$ (and a local extreme point as well), S^4 is a locally order-optimal schedule for some objective function f, S^5 is a local extreme point of $\mathcal{S}$ but no extreme point of $\mathcal{S}$, S^6 is a minimal point of $\mathcal{S}_T(O(S^6))$, i.e. the square with corners S^6, S^7, S', and S'', and S^7 is an extreme point of $\mathcal{S}_T(O(S^6))$.

4 Different types of objective functions

In this section, we discuss some special (mostly nonregular) objective functions f of scheduling problem (P) that are important in practice (cf. Neu-

mann and Zimmermann 1999a,b, Klein 2000, Sect. 3.4, Kimms 2001, Neumann et al. 2001, Sect. 3.1 and 3.3, and Nübel 2001). Moreover, we state which specific points of feasible region $\mathcal{S}$ represent possible optimal solutions to the respective scheduling problems, where we always assume that $\mathcal{S} \neq \emptyset$.

Recall that objective function f is said to be *regular* if f is nondecreasing, i.e., $S \leq S'$ implies $f(S) \leq f(S')$. Obviously, there is always a *minimal point* of $\mathcal{S}$ that represents an optimal solution to (P) with regular f. In literature, minimal points of $\mathcal{S}$ are called *active schedules* (see Sprecher et al. 1995 and Neumann et al. 2000). Examples of regular functions f are

- (*i*) the *projection duration* S_{n+1}, which equals the *makespan* $C_{\max}$ known from machine scheduling (cf. e.g., Pinedo 2001, Sect. 2.1) and represents the objective function most frequently used in practice,
- (*ii*) the *maximum lateness* $L_{\max} = \max_{i \in V} L_i$, where $L_i := S_i + p_i - d_i$ is the *lateness* and $d_i \in \mathbb{Z}_{\geq 0}$ is a given *due date* for activity i, and
- (*iii*) the *weighted tardiness* $\sum w_i^T T_i$, where $w_i^T \in \mathbb{Z}_{\geq 0}$ and $T_i := \max(L_i, 0)$ is the *tardiness* of activity i.

Now we turn to nonregular objective functions. The simplest nonregular functions are the so-called *antiregular* or nonincreasing functions f, i.e., $S \leq S'$ implies $f(S) \geq f(S')$. Obviously, there is always a *maximal point* of $\mathcal{S}$ that represents an optimal solution to (P) with antiregular f. An example of an antiregular function is the *weighted earliness* $\sum w_i^E E_i$, where $w_i^E \in \mathbb{Z}_{\geq 0}$ and $E_i := \max(-L_i, 0)$ is the *earliness* of activity i.

If the underlying objective function f is *linear*, there is always an *extreme point* of $\mathcal{S}$ that represents an optimal solution to (P). Function f given by $f(S) := \sum_{i \in V} w_i S_i$ with $w_i \in \mathbb{Z}$ is linear. If $w_i > 0$, activity i should be executed as early as possible, and for $w_i < 0$, activity i should be carried out as late as possible.

For a *convex* function f, local minimizers on polytopes $\mathcal{S}_T(O)$, $O \in \mathcal{O}$, are global minimizers as well and can efficiently be determined by so-called *steepest descent procedures* (cf. Simmons 1975, Sect. 8.1, and Schwindt 2000). Thus for convex objective functions f, it is sufficient to determine one local minimizer of f on each order polytope $\mathcal{S}_T(O)$, which is also called a *locally order-optimal schedule*. An example of a convex function is the *weighted earliness-tardiness* $\sum(w_i^E E_i + w_i^T T_i)$. The latter objective function is important for just-in-time production and rescheduling. By *rescheduling* we mean the following. In practice, input data such as processing times of activities or the amount of resources available are often subject to change, for example, if workers are absent or machines break down. As a consequence, a previously feasible schedule generally becomes infeasible. Then we want to find a new feasible schedule "as close as possible" to the old one because larger deviations cause additional difficulties. This can be done by solving a project scheduling problem with weighted earliness-tardiness objective function, where the due date d_i equals the old (now "infeasible") completion time of activity i.

A half-line $\ell(S,z)$ in $\mathbb{R}^{n+2}_{\geq 0}$ passing through point S is said to have *binary direction* if $\ell(S,z) = S + \mathbb{R}z$ for some $z \in \{0,1\}^{n+2}$. Function f is called *binary-monotone* if f is monotone (that is, either nondecreasing or nonincreasing) on each half-line in $\mathbb{R}^{n+2}_{\geq 0}$ with binary direction. It can be shown that there is always a *local extreme point* of $\mathcal{S}$ that represents an optimal solution to (P) with binary-monotone f (cf. Neumann et al. 2000). An example of a binary-monotone function is the (negative) *net present value* of a project. Let β be the discount factor per unit time where $0 < \beta \leq 1$. Moreover, let $c_i^F \in \mathbb{Z}$ be the *cash flow* associated with activity i, which is supposed to occur at the completion time $S_i + p_i$ of activity i and may be positive (i.e. a payment received) or negative (i.e. a disbursement incurred). Then $\sum_{i \in V} c_i^F \beta^{S_i + p_i}$ is the net present value of the project in question, which is to be maximized.

Finally, we consider the objective function of the *resource-investment problem*, where the cost of purchasing resources is to be minimized, and of *resource-levelling problems*, where some measure of the variation of resource utilization over time is to be minimized.

Function f is termed *locally regular* if f is regular on each equal-order set $\mathcal{S}_T^=(O(S))$, $S \in \mathcal{S}$. From this it follows that there is always a minimal point S of some schedule polytope $\mathcal{S}_T(O(S))$ which is a minimizer of a locally regular function f on $\mathcal{S}$ (cf. Neumann et al. 2001, Sect. 3.3). For example, the resource-investment objective function f given by $f(S) := \sum_{k \in \mathcal{R}} c_k \max_{0 \leq t \leq \overline{d}} r_k(S,t)$ is locally regular where $c_k \in \mathbb{Z}_{\geq 0}$ is the procurement cost per unit of resource k. Note that f is lower semicontinuous but not continuous.

Recall that function f is said to be *quasiconcave* if $f(\lambda S + [1-\lambda]S') \geq \min[f(S), f(S')]$ for all $\lambda \in [0,1]$ and all $S, S' \in \mathbb{R}^{n+2}_{\geq 0}$. This means that a quasiconcave function attains its minimum on a line segment at one of the two endpoints of that segment. Function f is called *locally quasiconcave* if f is quasiconcave on each equal-order set $\mathcal{S}_T^=(O(S))$, $S \in \mathcal{S}$. There is always an extreme point S of some schedule polytope $\mathcal{S}_T(O(S))$ that represents an optimal solution to scheduling problem (P) with locally quasiconcave f (cf. Neumann et al. 2001, Sect. 3.3). Examples of locally quasiconcave functions are the resource-levelling objective functions f given by the total squared utilization cost $f(S) := \sum_{k \in \mathcal{R}} c_k \int_0^{\overline{d}} r_k^2(S,t)\,dt$ or the total overload cost $f(S) := \sum_{k \in \mathcal{R}} c_k \int_0^{\overline{d}} \max[r_k(S,t) - Y_k, 0]\,dt$ where $c_k \in \mathbb{Z}_{\geq 0}$ is a cost incurred per unit of resource k and per unit time and $Y_k \in \mathbb{Z}_{\geq 0}$ is some threshold for the usage of resource k. Further resource-levelling functions, where resources that represent different kinds of manpower are considered and changing the size of work force over time should be smoothed, can be found in Neumann and Zimmermann (1999a) and Kimms (2001), Sect. 11.1. Recently, renting resources instead of buying them has become more and more important in practice. In Nübel (2001) and Neumann et al. (2001), Sect. 3.6, a *resource-renting problem* has been discussed, whose objective

function is the sum of fixed and variable renting costs and represents a locally quasiconcave function, too.

5 Exact solution methods

5.1 Relaxation-based approach

For solving project scheduling problem (P) with regular, antiregular, linear, convex, or binary-monotone objective functions f, a so-called **relaxation-based approach** turns out to be expedient (cf. Brucker et al. 1999, De Reyck et al. 1999, Neumann et al. 2000). If we omit the resource constraints (4) from problem (P), the resulting *resource relaxation* has feasible region $\mathcal{S}_T$ instead of $\mathcal{S}$. For regular, antiregular, linear, or binary-monotone functions f, there is always a vertex of polytope $\mathcal{S}_T$ which represents a minimizer of f on $\mathcal{S}_T$. For convex functions f, each local minimizer of f on $\mathcal{S}_T$ is a global one as well. For regular or antiregular functions f, the *earliest schedule* ES (i.e. the vector of earliest start times ES_i of activities $i \in V$) or *latest schedule* LS (i.e. the vector of latest start times LS_i of activities $i \in V$), respectively, is optimal. Note that $ES_i = d_{0i}$ and $LS_i = -d_{i0}$, where d_{ij} is the longest path length from node i to node j in project network N. For linear functions f, the resource relaxation of problem (P) is a linear program whose dual represents a network flow problem (cf. Russell 1970). For the net present value function, Kamburowski (1990), De Reyck (1998), Zimmermann (2000), and Schwindt and Zimmermann (2001) have devised a network simplex-like method, a recursive search procedure, and steepest descent approaches, respectively. Schwindt (2000) has proposed primal and dual steepest descent algorithms for the weighted earliness-tardiness function.

For regular objective function $C_{\max}$, exact methods of the branch-and-bound type for the resource-constrained problem (P) have been offered by Bartusch et al. (1988), De Reyck and Herroelen (1998a), Schwindt (1998), Fest et al. (1999), and Dorndorf et al. (2000). Those methods can be adapted to general regular and antiregular objective functions without any difficulty (cf. De Reyck et al. 1999, Neumann et al. 2001, Sect. 2.10). Branch-and-bound procedures for the net present value function have been developed by De Reyck and Herroelen (1998b), Neumann and Zimmermann (2000b) and for the weighted earliness-tardiness function by Schwindt (2000). For the case of precedence constraints instead of general temporal constraints, similar methods have been devised by Vanhoucke et al. (2001a,b) for the net present value and weighted earliness-tardiness problems.

We briefly sketch the **enumeration scheme** of such a branch-and-bound method for objective function $C_{\max}$, which has been devised by De Reyck and Herroelen (1998a) and combines the relaxation-based enumeration scheme of Bell and Park (1990) with the concept of minimal delaying alternatives used by Demeulemeester and Herroelen (1992). A similar ap-

proach has been proposed by Icmeli and Erengüç (1996) for the resource-constrained net present value problem with precedence constraints.

Let S be an optimal solution to the resource relaxation. A so-called resource conflict at a point in time t, that is

$$\sum_{i \in F} r_{ik} > R_k \text{ for some } k \in R \text{ and } F \subseteq \mathcal{A}(S,t)$$

can be resolved successively by introducing additional temporal constraints, which delay certain activities from *forbidden set* F. If F is minimal with respect to set inclusion, it is called a *minimal forbidden set* (cf. Bartusch et al. 1988, Stork and Uetz 2001). A resource conflict at time t caused by the execution of activities from $\mathcal{A}(S,t)$ can be resolved by introducing a precedence constraint $S_j \geq S_i + p_i$ between two activities i and j, $i \neq j$, from each minimal forbidden set $F \subseteq \mathcal{A}(S,t)$. An inclusion-minimal set B which contains at least one element from each minimal forbidden set $F \subseteq \mathcal{A}(S,t)$ is called a *minimal delaying alternative* (cf. Christofides et al. 1987, Demeulemeester and Herroelen 1992). A pair (i,B) where $i \in F \setminus B$ and B is a minimal delaying alternative is called a *minimal delaying mode* for F. It can be shown that it is sufficient to enumerate all minimal delaying modes to resolve a resource conflict (see De Reyck and Herroelen 1998a).

The basic idea of the enumeration scheme is as follows. We identify the enumeration nodes with time-feasible strict orders O in V and the corresponding search spaces $\mathcal{P}$ with order polytopes $\mathcal{S}_T(O)$. Let Γ be the set of feasible schedules to be enumerated and Ω be a set of strict orders O in activity set V. We start with $\Gamma = \emptyset$ and $\Omega = \{\emptyset\}$, and $O = \emptyset$ represents the root node of the enumeration tree. At first, we remove $O = \emptyset$ from Ω and determine the earliest schedule $S = \min \mathcal{S}_T(O) = \min \mathcal{S}_T$, i.e. the unique minimal point of search space $\mathcal{S}_T(O)$. If S is feasible, we set $\Gamma := \Gamma \cup \{S\}$. Otherwise, there is a time $t \geq 0$ such that active set $\mathcal{A}(S,t)$ is forbidden, and we compute all minimal delaying modes (i,B) for $\mathcal{A}(S,t)$. Then, we add the transitive hull O' of each relation $O \cup \{(i,j) \mid j \in B\}$ to Ω (and O' becomes a child node of O in the enumeration tree) provided that $\mathcal{S}_T(O') \neq \emptyset$. After that, we again delete a strict order O from Ω, consider the search space $\mathcal{P} = \mathcal{S}_T(O)$, determine the earliest schedule $S = \min \mathcal{P}$, and reiterate the previous steps until all strict orders from Ω have been investigated, i.e. $\Omega = \emptyset$. If $\mathcal{S} \neq \emptyset$, the resulting set Γ contains an optimal schedule.

The approach by Bartusch et al. (1988) differs from the above scheme in the forbidden sets considered in the course of the algorithm. Instead of generating one forbidden set $F = \mathcal{A}(S,t)$ in each iteration, all minimal forbidden sets for which all activities can overlap in time are determined in advance. At each level of the enumeration tree, one of those minimal forbidden sets is considered. For given minimal forbidden set F, the offsprings O' of node O arise from branching over all pairs (i,j) of activities $i,j \in F$ such that for the transitive hull O' of relation $O \cup \{(i,j)\}$, search space $\mathcal{S}_T(O')$ is nonempty.

Next, we briefly discuss three alternative enumeration schemes. To resolve resource conflicts, Schwindt (1998) has used *disjunctive precedence constraints* $S_j \geq \min_{i \in F \setminus \{j\}}(S_i + p_i)$, where $j \in F$ and F is a minimal forbidden set. Instead of delaying a single activity j, again several activities from a minimal delaying alternative B can be shifted at the same time. Then we obtain disjunctive precedence constraints between activity sets $A := \mathcal{A}(S,t) \setminus B$ and B

$$\min_{j \in B} S_j \geq \min_{i \in A}(S_i + p_i)$$

In literature, disjunctive precedence constraints are also referred to as *AND/OR precedence constraints* or *waiting conditions* (cf. Möhring et al. 2000). They have been introduced by Igelmund and Radermacher (1983) in the form of *preselective strategies* for resource-constrained project scheduling with stochastic activity durations. The use of disjunctive precedence constraints markedly reduces the number of enumeration nodes which have to be considered. On the other hand, the introduction of disjunctive precedence constraints leads to disconnected search spaces. Each search space $\mathcal{P}$, however, possesses a unique minimal point, and the problem of minimizing the project duration on set $\mathcal{P}$ can be solved by a straightforward fixed-point algorithm providing the minimal point of $\mathcal{P}$ in pseudo-polynomial time (cf. Schwindt 1998).

For resolving resource conflicts, Fest et al. (1999) have replaced the disjunctive precedence constraints between sets A and B by *dynamic release dates*

$$d_{0j}^{min} = \min_{i \in A}(S_i + p_i) \text{ for all } j \in B$$

being equal to the earliest completion time of some activity $i \in A$ with respect to current schedule S. Dynamic release dates only temporarily establish a precedence relationship between sets A and B, and hence the resource conflict may reappear later on. In comparison with disjunctive precedence constraints, the main advantage of the latter approach is that the minimal point of search space $\mathcal{P}$ can be calculated very efficiently in $\mathcal{O}(n)$ time. Furthermore, the enumeration scheme admits a simple and effective dominance criterion, which allows to weed out enumeration nodes by comparing corresponding release date vectors.

Whereas the relaxation-based enumeration schemes mentioned thus far are based on resolving resource conflicts, the approach by Dorndorf et al. (2000) relies on binary decisions of scheduling a given activity $j \in V$ at its (current) earliest possible start time d_{0j} or delaying j by introducing some release date $d_{0j}^{min} \geq d_{0j} + 1$. The size of the corresponding enumeration tree can considerably be reduced by applying so-called constraint propagation techniques, which are used to check whether certain start times can be excluded from further consideration because they cannot lead to feasible, active, or optimal schedules.

Apart from the enumeration scheme, a **branch-and-bound procedure** is characterized by the search strategy, preprocessing techniques, lower bounds on the minimum objective function value, and dominance rules.

The *search strategy* determines that enumeration node from which branching is performed next. Due to favorable memory requirements and ease of implementation, the enumeration tree is generally traversed according to a depth-first search. Alternatively, a scattered-search approach can be used, where the enumeration tree is partitioned into a given number of subtrees, which are generated simultaneously according to a depth-first search each (cf. Klein and Scholl 2000).

Preprocessing refers to a phase between formulation and solution of a problem, which may consist of adding inequalities, tightening bounds on variables, or fixing variables. In project scheduling, sometimes additional temporal constraints can be added to the underlying project network based on the examination of resource conflicts induced by two or more activities. The constraints added must be met by each optimal schedule and do not represent alternative manners of resolving resource conflicts. For example, a forbidden set $F = \{i, j\}$ where $d_{ij} > -p_j$ can be broken up in advance by precedence constraint $S_j \geq S_i + p_i$ because the temporal constraints prevent activity j from being completed before the start of activity i. Preprocessing can be used in the root of the enumeration tree as well as in each enumeration node to dynamically reduce its search space $\mathcal{P}$ (in the latter case, preprocessing is referred to as *immediate selection* or *constraint propagation*). Preprocessing and constraint propagation techniques for project scheduling have been discussed by Brucker et al. (1998), Baptiste et al. (1999), and Dorndorf et al. (1999).

Obviously, the solution of the resource relaxation provides a *lower bound* on the minimum objective function value. Tighter lower bounds based on interval capacities or disjunctive activities (i.e. activities which cannot be performed simultaneously) can be found in Heilmann and Schwindt (1997), De Reyck and Herroelen (1998a), Klein and Scholl (1999), and Neumann et al. (2001), Sect. 2.5. Lower bounds based on interval capacities result from comparing the minimum resource requirement and the resource supply in certain time intervals. Disjunctive activities can be used to refine the resource relaxation with precedence constraints in the course of constraint propagation. The corresponding lower bound equals the minimum objective function value on the reduced search space. For objective function $C_{\max}$, lower bounds LB can be computed in a destructive manner by showing that there is no feasible schedule S with $S_{n+1} < LB$. Destructive lower bounds again apply constraint propagation techniques to generate constraints which contradict hypothetical project deadlines. Further lower bounds on the minimum project duration based on Lagrange relaxation and column generation techniques have been proposed by Möhring et al. (1999) and Brucker and Knust (2000), respectively. Due to their considerable computational requirements, these bounds are primarily dedicated to the performance analysis of heuristic solution procedures.

Dominance rules allow to prune certain nodes of the enumeration tree by means of dominance considerations. In the course of the relaxation-based branch-and-bound procedures, different versions of a subset-dominance rule are used. The subset dominance rule compares the current search space $\mathcal{P}$ with search spaces $\mathcal{P}'$ which have been generated on a different traversal path in the enumeration tree. Clearly, the current enumeration node is redundant and thus can be deleted if $\mathcal{P} \subseteq \mathcal{P}'$. Further dominance rules, which depend on the specific enumeration scheme used, can be found in De Reyck and Herroelen (1998a), Fest et al. (1999), Dorndorf et al. (2000), and Franck et al. (2001a).

5.2 Tree-based approach

For locally regular or locally quasiconcave objective functions f, the resource relaxation of (P) is generally already NP-hard (cf. Neumann et al. 2001, Sect. 3.4). Thus, the relaxation-based approach is not recommended. Instead, Neumann and Zimmermann (1999a, 2000a) and Neumann et al. (2000) have proposed a so-called **tree-based approach**, whose basic idea is as follows. As mentioned in Section 4, minimal or extreme points (i.e. vertices) of schedule polytopes represent candidates for optimal schedules. A vertex S of schedule polytope $\mathcal{S}_T(O(S))$ can be represented by a spanning tree G of schedule network $N(O(S))$ with arc set E^G and arc weights δ_{ij}^G. S is the unique solution to the system of $n+2$ linear equations $S_0 = 0$, $S_j - S_i = \delta_{ij}^G$ ($\langle i,j \rangle \in E^G$).

To enumerate the vertices of all schedule polytopes $\mathcal{S}_T(O(S))$, we enumerate the corresponding spanning trees. These spanning trees are constructed by consecutively fixing start times of activities such that in each step, a temporal constraint $S_j - S_i \geq \delta_{ij}$ or a precedence constraint $S_j \geq S_i + p_i$ becomes binding. In this way, we construct a sequence of subtrees of some schedule network $N(O(S))$.

More precisely, let Γ be the set of time-feasible schedules to be enumerated. Moreover, let set Ω contain the pair $(\mathcal{C}, S^{\mathcal{C}})$ for each partial schedule $S^{\mathcal{C}} := (S_i)_{i \in \mathcal{C}}$ (or equivalently, for each subtree with node set $\mathcal{C}$) already constructed. We start with $\Gamma = \emptyset$ and $\Omega = \{\mathcal{C}, S^{\mathcal{C}}\}$ where $\mathcal{C} = \{0\}$ and $S_0 = 0$. In each iteration, we remove an arbitrary pair $(\mathcal{C}, S^{\mathcal{C}})$ from Ω. If $\mathcal{C} = V$, we add $S = S^{\mathcal{C}}$ to Γ. Otherwise, we extend the current partial schedule $S^{\mathcal{C}}$ as follows. For each $j^* \in V \setminus \mathcal{C}$, we determine the set $\mathcal{D}_{j^*}$ of tentative start times $t \in [ES_{j^*}, LS_{j^*}]$ for which there is an activity $i \in \mathcal{C}$ such that

- (*i*) $t = S_i + \delta_{ij^*}$, i.e., temporal constraint $S_{j^*} - S_i \geq \delta_{ij^*}$ is binding, or
- (*ii*) $t = S_i - \delta_{j^*i}$, i.e., temporal constraint $S_i - S_{j^*} \geq \delta_{j^*i}$ is binding, or
- (*iii*) $t = S_i + p_i$, i.e., precedence constraint $S_{j^*} \geq S_i + p_i$ is binding, or
- (*iv*) $t = S_i - p_{j^*}$, i.e., precedence constraint $S_i \geq S_{j^*} + p_{j^*}$ is binding.

For each $t \in \mathcal{D}_{j^*}$, we then add the corresponding extended partial schedule $S^{\mathcal{C}'}$ with $\mathcal{C}' = \mathcal{C} \cup \{j\}$ and $S_j = t$ to Ω. Next, we take a new pair $(\mathcal{C}, S^{\mathcal{C}})$

from Ω and proceed in the same way until all partial schedules from Ω have been investigated. For locally regular objective functions, there is always an optimal schedule corresponding to a minimal point of some schedule polytope. Such a schedule S can be represented by a spanning outtree of schedule network $N(O(S))$ with root 0. Thus, we merely consider cases (i) and (iii) when computing sets $\mathcal{D}_{j^*}$, which means that we only add arcs $\langle i, j^* \rangle$ with $i \in \mathcal{C}$ and $j^* \in V \setminus \mathcal{C}$ to the current subtree.

Specific implementations of the tree-based approach for the resource investment problem, different resource levelling problems, and the resource renting problem can be found in Nübel (2001), Neumann et al. (2001), Sect. 3.6, and Zimmermann (2001), Sect. 5.3. Further enumeration schemes for those problems have been presented by Möhring (1984), Nübel (1999), and Neumann and Zimmermann (2000a). The procedure by Möhring (1984) for the resource investment problem is based on algorithms for the recognition of so-called interval graphs and the orientation of their complements. Nübel (1999) enumerates the possible alternatives of how to resolve fictitious resource conflicts which arise from upper bounds on the maximum resource demands. Lower and upper bounds for the case of precedence constraints instead of general temporal constraints can be found in Kimms (2001), Ch. 12 and 13. Neumann and Zimmermann (2000a) have devised a time-window based branch-and-bound procedure for the resource investment and different resource levelling problems enumerating integral start times of activities. Lower bounds for the latter problems have been proposed by Neumann and Zimmermann (2000a) and Selle (2001).

Interestingly, for locally regular and locally quasiconcave objective functions f, a branch-and-bound method for solving problem (P) generally runs much faster than for the corresponding resource relaxation because due to the resource constraints in the former problem, a much smaller number of feasible schedules need to be enumerated. The schedules enumerated in the course of the relaxation-based approach represent vertices of (large) order polytopes $\mathcal{S}_T(O)$ covering feasible region $\mathcal{S}$. In contrast, the schedules enumerated by the tree-based approach represent the vertices of all schedule polytopes $\mathcal{S}_T(O(S))$, $S \in \mathcal{S}$. Hence, the number of schedules enumerated by the tree-based approach is generally much larger than for the relaxation-based approach.

6 Heuristic solution procedures

If the search for an optimal schedule in a branch-and-bound method is terminated prematurely, we obtain heuristic methods, so-called **truncated branch-and-bound procedures**. For example, we speak of a truncated branch-and-bound algorithm A with *performance guarantee* $\varepsilon > 0$, if the relative error of the objective function value for a feasible solution found by A is at most ε. The branch-and-bound procedure for objective function $C_{\max}$ presented in Section 5 can easily be truncated to such an ε-approximate

heuristic. Let $LB(O)$ be the lower bound belonging to enumeration node O and let UB be the objective function value of the best feasible solution found so far or a corresponding initialization value. Then the set of strict orders or nodes O in the enumeration tree we branch from is restricted to strict orders O with $LB(O) < UB/(1+\varepsilon)$.

For large problem instances with hundreds of activities, the computation of schedules with a relative error of at most ε may be too time-consuming. In this case, we recommend the truncation of the branch-and-bound procedure to a *filtered beam search procedure*, where the number of offsprings of an enumeration node is limited by the beam width (cf. e.g., Morton and Pentico 1993, Sect. 6.3).

A variant of the enumeration scheme sketched in Section 5 has been used by Cesta et al. (2002) for a multi-pass heuristic. Instead of minimal delaying modes (i, B), single pairs (i, j) are considered such that the addition of (i, j) to strict order O prevents the simultaneous execution of all activities from some selected minimal forbidden set $F \subseteq \mathcal{A}(S, t)$. The addition of pairs (i, j) to O is repeated until either $\mathcal{S}_T(O) = \emptyset$ or the earliest schedule $S = \min \mathcal{S}_T(O)$ is feasible.

Heuristic methods for problem (P) that schedule the activities successively according to certain priority rules provide feasible solutions generally much faster than truncated branch-and-bound procedures. Such **priority-rule methods** have been proposed by Neumann and Zhan (1995), Brinkmann and Neumann (1996), Neumann and Zimmermann (1999a,b, 2000a), Franck et al. (2001a), Selle and Zimmermann (2001), and Zimmermann (2001), Ch. 3. One iteration of such a priority-rule method is in principle as follows:

(i) Among the activities not yet scheduled, one activity, say i^*, is selected by applying some priority rule. Of course, for different objective functions, different priority rules have turned out to be expedient (cf. Neumann et al. 2001, Sect. 3.7).

(ii) For activity i^* selected, a start time S_{i^*} is computed such that all temporal and resource constraints are satisfied. S_{i^*} is generally chosen in a way that the so-called *additional-cost function* (representing the increase in the objective function value arising when activity i^* is scheduled at time S_{i^*}) is minimized on an appropriate decision set $\mathcal{D}_{i^*}$ of tentative start times for i^*. Set $\mathcal{D}_{i^*}$ depends on the type of objective function and the activities already scheduled, and its cardinality is linear in n (for details we refer to Neumann and Zimmermann 1999a).

In step (ii) it may happen that for a selected activity i^*, there does not exist any start time S_{i^*} complying with the temporal and resource constraints. In this case, some of the activities already scheduled have to be shifted such that activity i^* can be scheduled observing all temporal and resource constraints. Such *unscheduling procedures* have been used by Neumann and Zimmermann (2000a), Selle and Zimmermann (2000), and Franck et al. (2001a).

A priority-rule method determines at most one feasible schedule S. If the deviation of the corresponding objective function value from a lower bound on the minimum objective function value is too large, **schedule-improvement procedures** can be used to find "better" feasible schedules. Schedule-improvement procedures belong to the class of local search algorithms, which start with one or several initial solutions and iteratively generate and evaluate neighboring solutions. For an introduction to local search algorithms, we refer to Aarts and Lenstra (1997) and Michalewicz and Fogel (1999). Neighbors S' of a schedule S are obtained by applying some neighborhood operators to appropriate representations of S. We briefly sketch two neighborhood operators, which have been proposed by Neumann et al. (2002) for (approximately) solving problem (P).

At first, we consider the case of regular, antiregular, linear, binary-monotone, or convex objective functions f. Recall that for the latter objective functions, a minimizer of f on a nonempty order polytope $\mathcal{S}_T(O)$ can be determined efficiently and thus the relaxation-based approach can be used (see Section 5).

Let $tr(\rho)$ be the transitive hull of relation ρ and $cr(O)$ be the covering relation of strict order O, i.e. the inclusion-minimal relation ρ with $tr(\rho) = O$. The set of solutions Σ consists of all time-feasible strict orders O in activity set V for which removing some pair (i, j) from $cr(O)$ leads to a resource-infeasible minimizer of f on the new search space. The schedule $S(O)$ belonging to time-feasible strict order $O \in \Sigma$ is given by a minimizer of f on $\mathcal{S}_T(O)$. The neighborhood $\mathcal{N}(O)$ of strict order O is defined on the basis of deleting or adding appropriate pairs of activities from or to covering relation $cr(O)$. We distinguish between two cases:

(i) If minimizer $S(O)$ is feasible, then for each pair $(i, j) \in cr(O)$, reduced strict order $O' := tr(cr(O) \setminus \{(i, j)\})$ is a neighbor of O.

(ii) If schedule $S(O)$ is not resource-feasible, we determine the earliest point in time t with $r_k(S, t) > R_k$ for some $k \in \mathcal{R}$. For $g, h \in \mathcal{A}(S, t)$, expanded strict order $O' := tr(O \cup \{(g, h)\})$ is a neighbor of O provided that $\mathcal{S}_T(O') \neq \emptyset$.

Next, we consider problem (P) with locally regular or locally quasiconcave objective functions f. As stated in Section 4, for such a function f there is always a minimizer of f on some schedule polytope $\mathcal{S}_T(O(S))$ representing a minimal point or vertex of $\mathcal{S}_T(O(S))$, respectively. Moreover, each minimal point or vertex of $\mathcal{S}_T(O(S))$ can be represented by a spanning outtree rooted at node 0 or a spanning tree, respectively, of schedule network $N(O(S))$.

The rationale for a neighborhood function $\mathcal{N}$ on the solution set Σ of spanning trees G of schedule networks $N(O(S))$ is as follows. For two adjacent vertices of a nonempty schedule polytope, there are two corresponding spanning trees which differ in exactly one arc. Now let $G = \langle V, E^G \rangle$ be a spanning tree of $N(O(S))$ representing some vertex of schedule polytope $\mathcal{S}_T(O(S))$. A neighbor G' in the neighborhood $\mathcal{N}(G)$ of G is determined in

two steps. First, we delete an arc $\langle i, j\rangle$ from G and shift the resulting subtree T which does not contain node 0 until a temporal or precedence constraint corresponding to some arc $\langle g, h\rangle$ becomes binding. Second, we add arc $\langle g, h\rangle$ and obtain spanning tree G'. For the case of outtrees, we only consider arcs $\langle g, h\rangle = \langle g, j\rangle$ with terminal node j. This ensures that spanning outtrees G are transformed into spanning outtrees G'.

A tabu search procedure based on the above neighborhood functions has been devised in Neumann et al. (2001), Sect. 3.8. A different tabu search procedure for problem (P) using a start time vector representation of schedules has been proposed by Neumann and Zimmermann (2000a). For problem (P) with regular objective function $C_{\max}$, a genetic algorithm and a tabu search procedure based on a priority-list representation of schedules can be found in Franck et al. (2001a). For problem (P) with net present value objective function, a tabu search method where the set of solutions contains all integral start time vectors between the earliest and latest schedule has been offered by Icmeli and Erengüç (1994). A comprehensive overview on heuristic solution procedures for the case of precedence constraints and specific objective function $C_{\max}$ can be found in Hartmann and Kolisch (2000).

Finally, we summarize the results of an **experimental performance analysis** for the solution methods discussed in Sections 5 and 6 (for details we refer to Dorndorf et al. 2000, Franck et al. 2001a, and Zimmermann 2001, Sect. 4.4 and 5.4).

Different objective functions behave quite differently as far as computational effort and accuracy of the solutions obtained by the individual methods are concerned. In a sense, objective function $C_{\max}$ is the easiest to handle, and locally quasiconcave functions are the hardest. The branch-and-bound methods using the relaxation-based approach generally work well for projects with up to about 100 activities. Of course, there are also hard "pathological" instances with 50 activities that cannot be solved to optimality within a few minutes of computation time. Among the exact solution methods for objective function $C_{\max}$, the algorithm by Dorndorf et al. (2000) has shown the best performance. Branch-and-bound procedures derived from the tree-based approach can efficiently be used for small instances with about 30 activities. Tree-based local search methods, however, have been applied to projects with up to 500 activities (see Zimmermann 2001, Sect. 5.4). Truncated branch-and-bound procedures are recommended when dealing with medium-sized projects with at most 200 activities. Priority-rule methods are the fastest algorithms and have been used for large projects containing up to 1000 activities (see Franck et al. 2001a). Notice, however, that priority-rule techniques sometimes do not provide a feasible schedule although there is one. Schedule-improvement procedures like tabu search or genetic algorithms lead to more accurate feasible solutions but are more time-consuming.

7 Expansions of the basic project scheduling problem

In this section we consider several expansions of the basic project scheduling model (P) introduced in Section 2, which are needed when coping with the applications of project scheduling to be discussed in Section 8. At first we deal with break calendars, which specify time intervals during which some renewable resources cannot be used. In that case, it is often necessary to relax the requirement that activities must not be interrupted when being in progress. Instead, we assume that the execution of certain activities can be suspended during breaks, whereas other activities must not be interrupted. After that, we are concerned with cumulative resources, which are depleted and replenished over time. A cumulative resource can be regarded as the inventory level in some storage facility of finite capacity. The inventory level is bounded from below by some safety stock and from above by the capacity of the storage facility. Next, we treat the case of sequence-dependent changeover times, where resource units have to be reconfigured between the processing of two consecutive activities and the time needed for changeover may depend on both the preceding and following activities. Finally, we consider multi-mode project scheduling problems, where activities may be performed in alternative execution modes, which differ in durations and resource requirements. The execution modes of an activity reflect tradeoffs between the time and resource demands.

7.1 Calendarization

When scheduling real-life projects, one often has to take into account *breaks* during which renewable resources are not available. Breaks may arise from weekends, holidays, or scheduled maintenance times for machines. They can be represented by *break calendars*, i.e. right-continuous step functions $b : [0, \overline{d}] \rightarrow \{0, 1\}$, where $b(t) = 0$ precisely if time t falls into a break, and $b(t) = 1$, otherwise. $\int_t^{t'} b(\tau)d\tau$ with $t \leq t'$ is the *total working time* in interval $[t, t'[$. Scheduling the activities of a project subject to break calendars is termed *calendarization*.

In practice, different resources generally have different calendars. For what follows, we assign an *activity calendar* b_i to each activity $i \in V$ by putting $b_i(t)$ to be equal to zero if there is some resource used by activity i that is not available at time t, and equal to one, otherwise. The execution of an activity may be suspended at the beginning of a break. In that case, it has to be resumed at the end of the break. Let V^{bi} denote the set of those (break-)interruptible activities. Given start time S_i, the completion time of activity $i \in V^{bi}$ is

$$C_i(S_i) = \min\{t \geq S_i + p_i \mid \int_{S_i}^{t} b_i(\tau)d\tau = p_i\} \tag{5}$$

$V^{ni} := V \setminus V^{bi}$ is the set of all activities which must not be interrupted. For activities $i \in V^{ni}$ we have constraints

$$b_i(t) = 1 \quad (S_i \leq t < S_i + p_i) \tag{6}$$

Minimum and maximum time lags may depend on calendars, too. For example, due to (5) a precedence constraint $S_j \geq C_i(S_i)$ between an interruptible activity i and some activity j depends on the activity calendar of activity i. That is why we associate a *time lag calendar* b_{ij} with each arc $\langle i, j \rangle \in E$ of project network N. The temporal constraints then read

$$\int_{S_i}^{S_j} b_{ij}(\tau) d\tau \geq \delta_{ij} \quad (\langle i, j \rangle \in E) \tag{7}$$

If $\delta_{ij} \geq 0$, the total working time $\int_{S_i}^{S_j} b_{ij}(\tau) d\tau$ in interval $[S_i, S_j[$ must be greater than or equal to $d_{ij}^{min} = \delta_{ij}$. In case of $\delta_{ij} < 0$, the total working time $\int_{S_j}^{S_i} b_{ij}(\tau) d\tau = -\int_{S_i}^{S_j} b_{ij}(\tau) d\tau$ in interval $[S_j, S_i[$ must not be greater than $d_{ji}^{max} = -\delta_{ij}$.

A schedule satisfying the calendar constraints (6) and (7) is called *calendar-feasible*. Franck et al. (2001b) have devised polynomial-time label-correcting algorithms for finding the earliest and latest calendar-feasible schedules ES and LS. A preliminary version of those algorithms can be found in Zhan (1992). The principal idea of the procedure for computing schedule ES is to start with some schedule $S \leq ES$ and to iteratively delay the activities until all calendar constraints are satisfied. Each time the earliest start time of some activity i is increased, the calendar constraints are reexamined for all direct successors j of node i in project network N. Latest schedule LS can be determined similarly by starting from some schedule $S \geq LS$ and successively advancing the activities. It can be shown that schedules ES and LS coincide with the unique minimal and maximal points, respectively, of the set of all calendar-feasible schedules (see Franck 1999, Sect. 3.2).

7.2 Cumulative resources

The availability of renewable resources like manpower or machinery is independent of their previous utilization. Resources whose availability at a given time t results from all (positive and negative) requirements that have occurred by time t are called *cumulative resources* or *storage resources*. Such a cumulative resource k can be regarded as the inventory level in some storage facility which is depleted and replenished over time. The inventory level in resource k is bounded from below by some safety stock $\underline{R}_k \in \mathbb{Z}_{\geq 0}$ and from above by the capacity $\overline{R}_k \in \mathbb{Z}_{\geq 0}$ of the storage facility.

Let $\mathcal{R}^{\gamma}$ be the set of all cumulative resources under consideration and let $r_{ik} \in \mathbb{Z}$ denote the (storage) demand of activity i for resource k. If $r_{ik} > 0$, activity i replenishes resource k by r_{ik} units, and if $r_{ik} < 0$, resource k is

depleted by $-r_{ik}$ units. We assume that resources $k \in \mathcal{R}^\gamma$ are depleted at start times and replenished at completion times of activities. To simplify writing, we also suppose that an activity cannot deplete and replenish one and the same cumulative resource. r_{0k} corresponds to the initial stock of resource k.

Now let $V_k^- := \{i \in V \mid r_{ik} < 0\}$ and $V_k^+ := \{i \in V \mid r_{ik} > 0\}$ denote the sets of all activities $i \in V$ depleting and replenishing, respectively, resource k. Then $\mathcal{A}_k(S,t) := \{i \in V_k^- \mid S_i \leq t\} \cup \{i \in V_k^+ \mid S_i + p_i \leq t\}$ is the *active set* of all activities that determine the inventory level

$$r_k(S,t) := \sum_{i \in \mathcal{A}_k(S,t)} r_{ik}$$

in resource k at time t.

The *inventory constraints*, which say that at any point in time $t \in [0, \overline{d}]$ the inventory level in each resource k must be between the safety stock and the storage capacity, can be written as

$$\underline{R}_k \leq r_k(S,t) \leq \overline{R}_k \quad (k \in \mathcal{R}^\gamma,\ 0 \leq t \leq \overline{d}) \tag{8}$$

A schedule S which satisfies the inventory constraints is termed *inventory-feasible*.

A renewable resource k can be interpreted as a cumulative resource with zero safety stock. At the project beginning, the inventory is replenished by $r_{0k} = R_k$ units. The start of an activity $i \in V$ depletes the inventory by r_{ik} units, which are returned to k at the completion of i. Clearly, with the renewable resources modelled in that way, a schedule is resource-feasible precisely if it is inventory-feasible. That is the reason why project scheduling problems with cumulative resources represent a generalization of basic project scheduling problem (P) with renewable resources.

The concept of cumulative resources has been introduced by Schwindt (1998). Neumann and Schwindt (1999) have devised a branch-and-bound algorithm for minimizing the duration of a project subject to temporal constraints (3) and inventory constraints (8). An alternative branch-and-bound procedure for this problem has been developed by Laborie (2001). Carlier and Rinnooy Kan (1982) have considered polynomial-time algorithms for the special case where all replenishments occur at predetermined points in time.

For given schedule S, we distinguish between two types of resource conflicts and forbidden sets. We speak of an *inventory shortage* at point in time t if

$$\sum_{i \in F} r_{ik} < \underline{R}_k \text{ for some } k \in \mathcal{R}^\gamma \text{ and } \mathcal{A}_k(S,t) \setminus F \subseteq V_k^-,\ F \setminus \mathcal{A}_k(S,t) \subseteq V_k^+$$

and set F is termed a *shortage set*. An inventory shortage can be resolved by introducing precedence constraints $S_j \geq S_i + p_i$ between replenishing activities i outside set F and depleting activities j from set F. In other

words, we add a minimum time lag $d_{ij}^{min} = p_i$ between the start of activities i and j. Symmetrically, the conflict

$$\sum_{i \in F} r_{ik} > \overline{R}_k \text{ for some } k \in \mathcal{R}^\gamma \text{ and } \mathcal{A}_k(S,t) \setminus F \subseteq V_k^+,\ F \setminus \mathcal{A}_k(S,t) \subseteq V_k^-$$

is called an *inventory excess* and F is referred to as a *surplus set*. For resolving an inventory excess, we delay the completion of some replenishing activities j from set F up to the start of some depleting activities i outside set F, i.e., $S_j + p_j \geq S_i$. This means that we introduce maximum time lags $d_{ji}^{max} = p_j$ between the start of activities i and j. Analogously to the case of renewable resources, an inclusion-minimal set $B \subseteq F$ with $\underline{R}_k \leq \sum_{i \in F \setminus B} r_{ik} \leq \overline{R}_k$ is called a *minimal delaying alternative*.

In the relaxation-based algorithm by Neumann and Schwindt (1999), at each iteration a disjunctive precedence constraint of type $\min_{j \in B} S_j \geq \min_{i \in A}(S_i + p_i)$ or $\min_{j \in B}(S_j + p_j) \geq \min_{i \in A} S_i$ is introduced between set $A = V_k^+ \setminus \mathcal{A}_k(S,t)$ or $A = V_k^- \setminus \mathcal{A}_k(S,t)$ and a minimal delaying alternative $B \subseteq \mathcal{A}_k(S,t)$. The branching is performed over the set of all minimal delaying alternatives B for a given forbidden active set $F = \mathcal{A}_k(S,t)$. The enumeration scheme by Laborie (2001) is based on investigating, for appropriate pairs (i,j) of replenishing and depleting activities i and j, the two alternatives whether or not j is started before the completion of i.

7.3 Sequence-dependent changeover times

When coping with projects whose activities are distributed over different locations sharing common renewable resources, *changeover times* for the transportation of resource units have to be taken into account. Changeover times also arise when resource units have to be cleaned or teared down and reinstalled between the execution of two consecutive activities. During the changeover, those resource units are not available for processing activities. The changeover times are generally sequence-dependent, which means that the time needed for changing over a resource unit may depend on both the activity preceding and the activity following the changeover.

Let $\vartheta_{ij}^k \in \mathbb{Z}_{\geq 0}$ denote the changeover time between activities i and j on resource k, where we assume that the *weak triangle inequality*

$$\vartheta_{hi}^k + p_i + \vartheta_{ij}^k \geq \vartheta_{hj}^k$$

is satisfied for all activities $h, i, j \in V$ and all resources $k \in \mathcal{R}$. Given a schedule S and a resource k, we have to find an assignment of r_{ik} resource units to each activity $i \in V$ of the project such that no two activities share common resource units if, inclusive the changeover time, they overlap in time. We say that schedule S is *changeover-feasible* if such an assignment exists. Clearly, any changeover-feasible schedule is resource-feasible as well.

In what follows, we outline the basic principle of a branch-and-bound procedure by Neumann et al. (2001), Sect. 2.13, for finding a time- and

changeover-feasible schedule with minimum project duration. An alternative procedure for the case of single-unit resource requirements $r_{ik} \in \{0,1\}$ for all $i \in V$ and all $k \in \mathcal{R}$ has been offered by Trautmann (2001), Sect. 3.3. Let S be some time-feasible schedule and let

$$O_k(S) := \{(i,j) \in V \times V \mid i \neq j,\ S_j \geq S_i + p_i + \vartheta_{ij}^k\}$$

be the schedule-induced strict order in set V containing all pairs (i,j) such that the time lag between the completion of activity i and the start of activity j is sufficiently large for a changeover from i to j on resource k. $\mathcal{A} \subseteq V$ is called an *antichain* in strict order $O_k(S)$ if $(i,j) \notin O_k(S)$ and $(j,i) \notin O_k(S)$ for all $i,j \in \mathcal{A}$. It follows from the definition of strict order $O_k(S)$ that activities from an antichain $\mathcal{A}$ in $O_k(S)$ overlap in time and thus must be executed on different resource units each.

The changeover-feasibility for a given schedule S can be tested by computing for each resource $k \in \mathcal{R}$ a maximum-weight antichain $\mathcal{A}_k(S)$ in strict order $O_k(S)$, where the weights of activities $i \in V$ correspond to the resource requirements r_{ik}. It is well-known that such a maximum-weight antichain can be computed efficiently via network flow techniques by finding a maximum-weight stable set in the transitively orientable comparability graph of $O_k(S)$ (see, e.g., Kaerkes and Leipholz 1977). S is changeover-feasible exactly if

$$\sum_{i \in \mathcal{A}_k(S)} r_{ik} \leq R_k \quad (k \in \mathcal{R}) \tag{9}$$

At each iteration of the branch-and-bound algorithm, a maximum-weight antichain $\mathcal{A}_k(S)$ in the schedule-induced strict order $O_k(S)$ belonging to current schedule S is determined for each resource k. If for some $k \in \mathcal{R}$ antichain $\mathcal{A}_k(S)$ is a forbidden set, the resource conflict is resolved by introducing a disjunctive precedence constraint $\min_{j \in B} S_j \geq \min_{i \in A}(S_i + p_i + \vartheta_{ij}^k)$ between a set $A \subset \mathcal{A}_k(S)$ and a corresponding minimal delaying alternative B for $\mathcal{A}_k(S)$, where $A = \mathcal{A}_k(S) \setminus B$.

7.4 Multi-mode project scheduling

In many applications of project scheduling, certain activities may be carried out in *alternative execution modes*, which differ in durations and resource requirements. For example, the duration of an activity may be shortened by increasing the number of resource units allotted (time-resource tradeoff) or some resources used may be replaced by other ones (resource-resource tradeoff). Multi-mode project scheduling problems often include *nonrenewable resources* like a budget, which correspond to cumulative resources that are depleted but never replenished.

Let $\mathcal{R}^\nu$ be the set of all nonrenewable resources, $\mathcal{R}$ be again the set of renewable resources, and $R_k \in \mathbb{N}$ be the availability of $k \in \mathcal{R} \cup \mathcal{R}^\nu$. For each activity $i \in V$, a finite set $\mathcal{M}_i$ of execution modes m_i with associated requirements $r_{ikm_i} \in \mathbb{Z}_{\geq 0}$ for renewable or nonrenewable resources

$k \in \mathcal{R} \cup \mathcal{R}^\nu$ and durations p_{im_i} is given. We assume that minimum and maximum time lags d_{ij}^{min} and d_{ij}^{max} depend on the modes m_i and m_j in which activities i and j are performed. This means that arcs $\langle i, j \rangle \in E$ are now weighted by matrices $\delta_{ij} = (\delta_{im_ijm_j})_{m_i \in \mathcal{M}_i, m_j \in \mathcal{M}_j}$.

A binary vector $x = (x_{im_i})_{i \in V, m_i \in \mathcal{M}_i}$ satisfying the *mode assignment constraints*

$$\sum_{m_i \in \mathcal{M}_i} x_{im_i} = 1 \quad (i \in V)$$

is called a (full) *mode assignment*. Each mode assignment corresponds to one single-mode project scheduling problem with the respective resource requirements $r_{ik}(x) := \sum_{m_i \in \mathcal{M}_i} r_{ikm_i} x_{im_i}$, durations $p_i(x) := \sum_{m_i \in \mathcal{M}_i} p_{im_i} x_{im_i}$, and arc weights $\delta_{ij}(x) := \sum_{m_i \in \mathcal{M}_i} \sum_{m_j \in \mathcal{M}_j} \delta_{im_ijm_j} x_{im_i} x_{jm_j}$. We say that mode assignment x is *feasible* if first,

$$\sum_{i \in V} r_{ik}(x) \leq R_k \quad (k \in \mathcal{R}^\nu)$$

and second, the project network $N(x)$ with arc weights $\delta_{ij}(x)$ $(\langle i, j \rangle \in E)$ does not contain any cycle of positive length. A schedule S is termed *feasible with respect to mode assignment* x if the temporal and renewable-resource constraints are satisfied, i.e.,

$$S_j - S_i \geq \delta_{ij}(x) \quad (\langle i, j \rangle \in E)$$

and

$$r_k(S, x, t) \leq R_k \quad (k \in \mathcal{R})$$

where $r_k(S, x, t) := \sum_{i \in \mathcal{A}(S,x,t)} r_{ik}(x)$ is the requirement for resource $k \in \mathcal{R}$ and $\mathcal{A}(S, x, t) := \{i \in V \mid S_i \leq t < S_i + p_i(x)\}$ is the active set at time t given schedule S and mode assignment x. A multi-mode project scheduling problem consists of finding a *schedule-assignment pair* (S, x) such that x is a feasible mode assignment, S is a schedule feasible with respect to x, and some objective function f is minimized. The first subproblem of finding a feasible mode assignment x is called the *mode assignment problem*, and the second subproblem coincides with the single-mode project scheduling problem belonging to mode assignment x.

Three different approaches to solving the multi-mode project duration problem have been proposed in literature. The tabu search procedure by De Reyck and Herroelen (1999) performs a local search in the set of possible mode assignments. For given mode assignment x, the resulting single-mode problem is then solved by the branch-and-bound algorithm of De Reyck and Herroelen (1998a). Franck (1999), Sect. 7.2, has adapted the priority-rule method for the single-mode case from Section 6 to the case of multiple execution modes. At each iteration, the activity to be scheduled is chosen on the basis of a first priority rule. A second priority rule provides the execution mode for the selected activity. A streamlined multi-pass version of this procedure has been presented by Heilmann (2001). The basic principle of the branch-and-bound procedure by Heilmann (2002) is to consider

single-mode problems arising from *mode relaxations*, which belong to *partial mode assignments* $\underline{x}$ where only the unavoidable resource requirements, durations, and time lags occurring in all selectable execution modes are taken into account. The mode relaxations are stepwise refined by iteratively assigning execution modes to activities and thus transforming the partial mode assignments $\underline{x}$ into a full mode assignment x.

Brucker and Knust (2000) have described a destructive lower bound, which is based on falsifying hypothetical project deadlines $\overline{d}' \leq \overline{d}$. For a given deadline $\overline{d}'$, they construct a linear-programming relaxation of the problem. Using column-generation techniques, it is then determined whether or not the linear program is solvable. In the latter case, deadline $\overline{d}'$ has been disproved and $\overline{d}' + 1$ thus represents be a lower bound on the minimum project duration.

A special case of the multi-mode project duration problem has been studied by Demeulemeester et al. (2000), where a branch-and-bound algorithm for the discrete time-resource tradeoff problem has been offered. For each real activity, a work content for a single renewable resource is specified. The alternative execution modes arise from all undominated integral duration-requirement combinations for which the product of duration and requirement is greater than or equal to the given work content.

8 Applications

In this section, we briefly discuss applications of resource-constrained project scheduling to make-to-order production in manufacturing industry, batch scheduling in process industries, and investment projects.

8.1 Make-to-order production in manufacturing

If all products are manufactured in response to firm customer orders, that is, no inventories are built up for future sale, we speak of *make-to-order production*. In addition to given customers orders, prescribed delivery dates for these orders have to be met. The execution of a customer order, which may contain several final products, can be viewed as a *project* to be performed. Manufacturing the gross requirement for one product is considered a *job*, and the processing of a job on a machine represents an *operation* or *activity*. We then want to find a production schedule which

- (*i*) minimizes the makespan or project duration,
- (*ii*) satisfies the primary requirement for each final product,
- (*iii*) complies with the limited capacity of machines,
- (*iv*) observes the delivery dates of customer orders, and
- (*v*) allows for overlapping of operations, without interrupting any operation.

Condition (v) means that the transportation lot size of some product from a machine to a different machine may be smaller than the production lot size (i.e. the gross requirement) for that part, that is, some units of the part may be transferred from the first to the second machine before the processing of the whole production lot on the first machine is completed. Such an overlapping of operations can reduce the makespan substantially.

The production scheduling problem sketched represents a problem of type (P), where condition (i) says that the objection function is the makespan $C_{\max}$. In general, each job has to be processed on several machine types (which correspond to renewable resources) in a prescribed order given by a so-called process plan. In Neumann and Schwindt (1997) and Neumann et al. (2001), Sect. 2.9, it is shown that conditions (ii) and (iii) lead to resource constraints (4) and conditions (iv) and (v) result in minimum and maximum time lags between operations, that is, temporal constraints.

8.2 Batch scheduling in process industries

In this subsection we are concerned with the scheduling of batch plants in process industries, where similarly to the case of manufacturing dealt with in Subsection 8.1, final products arise from several successive transformations of intermediate products. In process industries, the operations correspond to chemical reactions in processing units such as reactors, heaters, or filters. Each operation can be carried out on one out of several alternative processing units, which may differ in speed. The processing units are operated by workers. Each operation may consume several input products and may produce several output products. Perishable intermediate products must be consumed within a prescribed shelf life time, which may be equal to zero. In the latter case, the product cannot be stocked. In addition, the storable intermediate products must be buffered in dedicated storage facilities like tanks or silos. Further peculiarities encountered in process industries are sequence-dependent cleaning times on processing units and large processing times, which may necessitate the explicit consideration of breaks like weekends.

The *batch scheduling problem* consists of allocating processing units, intermediate products, and storage space over time to the execution of the operations such that

- (i) the makespan is minimized and is less than or equal to the planning horizon,
- (ii) each operation is carried out on some processing unit,
- (iii) each processing unit performs no more than one operation at a time,
- (iv) the time lag between two consecutive operations on a processing unit is sufficiently large for cleaning,
- (v) there are sufficiently many workers available to operate the processing units,

(*vi*) the inventories of intermediate products do neither fall below the safety stocks nor exceed the storage space available,

(*vii*) perishable products are consumed within the prescribed shelf life times, and

(*viii*) operations which require permanent supervision and control are not processed during a weekend, and no operation is executed on a processing unit during a maintenance time.

In what follows we sketch a resource-constrained project scheduling model for the batch scheduling problem, which has been discussed in Schwindt and Trautmann (2000). Analogously to the case of make-to-order production, the execution of all operations can be viewed as a project, where the *makespan* from condition (*i*) coincides with the project duration and the project deadline $\overline{d}$ is chosen to be equal to the planning horizon. For each operation we have one real activity $i \in V$.

We combine identical *processing units* to form a pool, which is modelled as a renewable resource k. Processing units are identical if they can execute the same operations with the same processing and cleaning times. If operation $i \in V$ can be executed on a processing unit of resource k, we introduce a corresponding execution mode $m_i \in \mathcal{M}_i$ for activity i with $r_{ikm_i} = 1$. The duration p_{im_i} equals the processing time of operation i on a processing unit of resource k. Condition (*ii*) then corresponds to the mode assignment constraints, and condition (*iii*) means that the capacity R_k of resource k coincides with the number of processing units in the pool. The *cleanings of processing units* from condition (*iv*) represent sequence-dependent changeovers between operations i and j on the respective unit of renewable resource k. The changeover time ϑ_{ij}^k is set to be equal to the cleaning time of the processing unit between the execution of operations i and j.

Analogously to identical processing units, the *workers* form a pool corresponding to a renewable resource k. The requirements r_{ikm_i} for resource k by modes $m_i \in \mathcal{M}_i$ where $i \in V$ equal the number of workers needed for operating the processing unit belonging to mode m_i. From condition (*v*) it follows that the capacity R_k of resource k equals the number of workers available.

The inventories in *intermediate storage facilities* can be modelled as cumulative resources. We identify each intermediate product with one cumulative resource $k \in \mathcal{R}^\gamma$. If operation i produces the intermediate product, r_{ik} equals the number of units produced, and if operation i consumes the intermediate product, r_{ik} equals the negative number of units consumed. According to condition (*vi*), $\underline{R}_k$ is equal to the safety stock of the product and $\overline{R}_k$ coincides with the number of units that can be stocked. For perishable products with zero shelf life time, we have $\underline{R}_k = \overline{R}_k = 0$. The case of general shelf life times in condition (*vii*) can be modelled by introducing auxiliary activities and auxiliary cumulative resources (for details see Schwindt and Trautmann 2002).

Condition (*viii*) means that certain operations cannot be in progress during *breaks* like weekends and maintenance times. We model breaks by introducing an activity calendar b_i for each such operation i, where $b_i(t) = 0$ exactly if time t falls into an interval during which resources $k \in \mathcal{R}$ required for processing i are not available. For the remaining activities $i \in V$, we have $b_i(t) = 1$ for all $0 \leq t \leq \overline{d}$.

Figure 4, which is taken from Neumann et al. (2001), Sect. 2.15, shows the schedule-generation scheme of a relaxation-based branch-and-bound procedure for the batch scheduling problem, which has been proposed by Schwindt and Trautmann (2000).

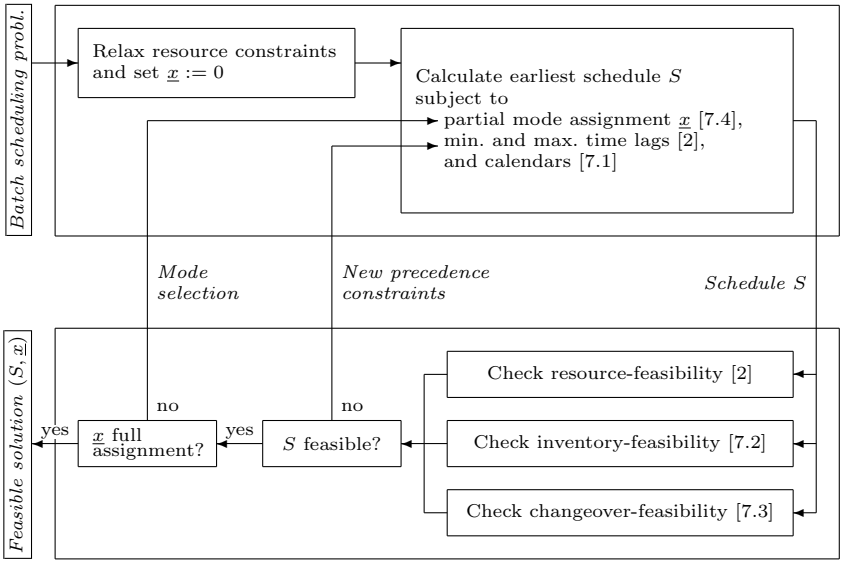

Fig. 4 Enumeration scheme for solving the batch scheduling problem

The algorithm is based on the above representation of the problem as a multi-mode resource-constrained project scheduling problem with renewable and cumulative resources, sequence-dependent changeover times, and activity calendars. The numbers in brackets refer to the sections covering the respective issues. Starting with the partial mode assignment $\underline{x} = 0$ where no execution mode has been selected for any activity, the procedure calculates at each iteration the earliest calendar-feasible schedule S for given partial assignment $\underline{x}$. It is then checked whether schedule S is resource-feasible, inventory-feasible, and changeover-feasible. If this is not the case, new precedence constraints are introduced for resolving the respective resource conflict and a new earliest schedule is computed for the refined relaxation. Otherwise, schedule S is feasible with respect to partial mode assignment $\underline{x}$. If in the latter case there are still activities for which no execution mode has been selected, we proceed by fixing an execution

mode and return to the calculation of the next earliest schedule. Otherwise, $\underline{x}$ is a full assignment, and thus schedule-assignment pair $(S, \underline{x})$ is a feasible solution to the batch scheduling problem.

Based on the procedure from Figure 4, for the first time, Schwindt and Trautmann (2000) have provided a feasible solution to a benchmark problem from industry submitted by Westenberger and Kallrath (1995). The latter case study covers most of the features occurring in production scheduling of batch plants.

8.3 Investment projects

Project managers are frequently confronted with the problem to decide whether some given project should be performed or to select one out of several mutually exclusive projects from a given portfolio. For the assessment of investments, the net present value criterion is well-established in research and practice. In classical preinvestment analysis, investments are specified by a stream of payments, i.e. a series of payments with associated payment times. Given a stream of payments and a proper discount rate, the net present value of the project is obtained by summing up all payments discounted to the project beginning (cf. Figure 5a, where exogenous parameters are written in italics).

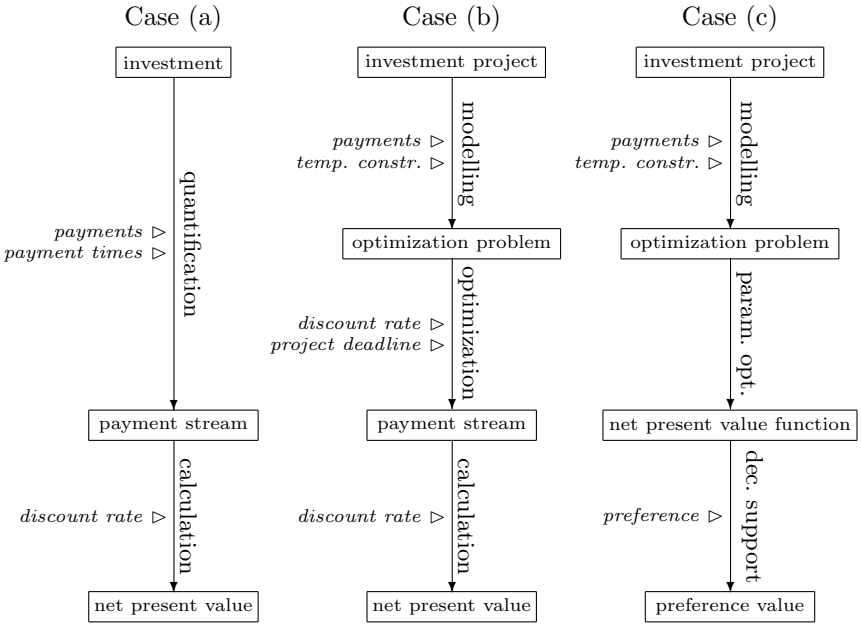

Fig. 5 Evaluation of investments and investment projects

In case of *investment projects*, the payment times are no longer given in advance but are subject to optimization. An investment project consists of a set of activities each of which is associated with a payment. Moreover, there are prescribed minimum and maximum time lags between the activities of the project. Thus, the stream of payments results from maximizing the net present value of the project subject to the temporal constraints that are given by the minimum and maximum time lags (cf. Figure 5b). The latter problem can be solved by the steepest ascent algorithm by Schwindt and Zimmermann (2001) for the time-constrained net present value problem.

The formulation of this optimization problem presupposes the knowledge of the *required rate of return* (i.e. the discount rate) for discounting the payments and the specification of a *maximum project duration* (i.e. the project deadline). When dealing with real investments in material goods like in building industry, however, often neither the proper discount rate to be applied is known with sufficient accuracy nor is the project deadline fixed when the investment project must be evaluated. The required rate of return is a theoretical quantity and can only be estimated. The project deadline generally arises from negotiations between the investor performing the project and his or her customers. The parametric optimization approach by Schwindt and Zimmermann (2002) provides the maximum project net present value $C^*(\alpha, \overline{d})$ of the project as a function of the discount rate α and project deadline $\overline{d}$ chosen, where $\alpha = -\ln \beta$ (see Section 4). The resulting *net present value curve* can then serve as a basis for the decision of the investor, which depends on his individual risk preference (cf. Figure 5c). The basic idea for computing the net present value curve is to cover its domain by a finite number of sets $\mathcal{M}$ such that on each of those sets, function C^* can be specified in closed form. Clearly, C^* is a closed-form function on connected subsets $\mathcal{M}$ of its domain where the binding temporal constraints for optimal schedules S are the same for all $(\alpha, \overline{d}) \in \mathcal{M}$.

9 Conclusions

This paper has summarized recent research on resource-constrained project scheduling with time windows. First, a basic project scheduling problem (P) has been formulated, whose feasible region represents the union of finitely many polytopes and where several different types of (mostly nonregular) objective functions are of great importance in practice. Minimization of the makespan, the maximum lateness, or the weighted earliness plus tardiness, maximization of the net present value as well as resource-investment, resource-levelling, and resource-renting problems are special cases of problem (P). Second, several different exact solution methods of the branch-and-bound type and heuristic solution techniques such as truncated branch-and-bound, priority-rule, and schedule-improvement procedures have been sketched. Third, the following expansions of the basic problem (P) have been briefly discussed: calendarization, so-called cumulative or storage resources,

sequence-dependent changeover times between the execution of consecutive activities, and multi-mode project scheduling. Finally, applications to make-to-order production in manufacturing, batch scheduling in process industries (e.g. chemical or pharmaceutical industries), and investment projects have been considered. It has been shown how to model those practical problems as project scheduling problems and how to solve them.

An important field of future research is new applications of resource-constrained project scheduling, for example to short- and medium-term planning in the area of supply chain management for different sectors of industry. Such applications require the handling of a very large number of activities or operations, say 5000 or more. Since the most efficient heuristic solution procedures proposed recently can only deal with up to 1000 activities (cf. Section 6), new decomposition methods for larger problem instances have to be developed which exploit the special problem structure. Another area of future research important to practice is rescheduling (cf. Section 4). Different realistic measures for the distance between (given infeasible and desired feasible) schedules have to be found in addition to the weighted earliness-tardiness criterion, and fast rescheduling algorithms have to be developed.

References

1. Aarts E, Lenstra JK, eds. (1997) Local Search in Combinatorial Optimization. John Wiley, Chichester
2. Baptiste P, Le Pape C, Nuijten W (1999) Satisfiability tests and time-bound adjustments for cumulative scheduling problems. Annals of Operations Research 92:305–333
3. Bartusch M, Möhring RH, Radermacher FJ (1988) Scheduling project networks with resource constraints and time windows. Annals of Operations Research 16:201–240
4. Bell CE, Park K (1990) Solving resource-constrained project scheduling problems by A* search. Naval Research Logistics 37:61–84
5. Brinkmann K, Neumann K (1996) Heuristic procedures for resource-constrained project scheduling with minimal and maximal time-lags: The resource-levelling and minimum project duration problems. Journal of Decision Systems 5:129–155
6. Brucker P, Knust S (2000) A linear programming and constraint propagation-based lower bound for the RCPSP. European Journal of Operational Research 127:355–362
7. Brucker P, Knust S, Schoo A, Thiele O (1998) A branch & bound algorithm for the resource-constrained project scheduling problem. European Journal of Operational Research 107:272–288
8. Brucker P, Drexl A, Möhring RH, Neumann K, Pesch E (1999) Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research 112:3–41
9. Carlier J, Rinnooy Kan AHG (1982) Scheduling subject to nonrenewable resource constraints. Operations Research Letters 1:52–55

10. Cesta A, Oddi A, Smith SF (2002) A constrained-based method for project scheduling with time windows. Journal of Heuristics, to appear
11. Christofides N, Alvarez-Valdes R, Tamarit JM (1987) Project scheduling with resource constraints: A branch and bound approach. European Journal of Operational Research 29:262–273
12. Demeulemeester EL, Herroelen WS (1992) A branch and bound procedure for the multiple resource-constrained project scheduling problem. Management Science 38:1803–1818
13. Demeulemeester EL, De Reyck B, Herroelen WS (2000) The discrete time/ resource tradeoff problem in project networks: A branch-and-bound approach. IIE Transactions 32:1059–1069
14. De Reyck B (1998) Scheduling projects with generalized precedence relations: Exact and heuristic procedures. PhD Thesis, University of Leuven
15. De Reyck B, Herroelen WS (1998a) A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 111:152–174
16. De Reyck B, Herroelen WS (1998b) An optimal procedure for the resource-constrained project scheduling problem with discounted cash flows and generalized precedence relations. Computers & Operations Research 25:1–17
17. De Reyck B, Herroelen WS (1999) The multi-mode resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 119:538–556
18. De Reyck B, Demeulemeester EL, Herroelen WS (1999) Algorithms for scheduling projects with generalized precedence constraints, In: Węglarz J (ed.) Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Boston, 77–105
19. Dorndorf U, Phan Huy T, Pesch E (1999) A survey of interval capacity consistency tests for time- and resource-constrained scheduling. In: Węglarz J (ed.) Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Boston, 213–238
20. Dorndorf U, Pesch E, Phan Huy T (2000) A time-oriented branch-and-bound algorithm for resource-constrained project scheduling with generalised precedence constraints. Management Science 46:1365–1384
21. Elmaghraby SE (1995) Activity nets: A guided tour through some recent developments. European Journal of Operational Research 82:383–408
22. Fest A, Möhring RH, Stork F, Uetz M (1999) Resource-constrained project scheduling with time windows: A branching scheme based on dynamic release dates. Technical Report 596, Technical University of Berlin
23. Franck B (1999) Prioritätsregelverfahren für die ressourcenbeschränkte Projektplanung mit und ohne Kalender. Shaker, Aachen
24. Franck B, Neumann K, Schwindt C (2001a) Truncated branch-and-bound, schedule-construction, and schedule-improvement procedures for resource-constrained project scheduling. OR Spektrum 23:297–324
25. Franck B, Neumann K, Schwindt C (2001b) Project scheduling with calendars. OR Spektrum 23:325–334
26. Hartmann S, Kolisch R (2000) Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. European Journal of Operational Research 127:394–407
27. Heilmann R (2001) Resource-constrained project scheduling: A heuristic for the multi-mode case. OR Spektrum 23:335–357

28. Heilmann R (2002) A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags. European Journal of Operational Research, to appear
29. Heilmann R, Schwindt C (1997) Lower bounds for RCPSP/max. Report WIOR-511, Institute for Economic Theory and Operations Research, University of Karlsruhe
30. Herroelen WS, De Reyck B, Demeulemeester EL (1998) Resource-constrained project scheduling: A survey of recent developments. Computers and Operations Research 25:279–302
31. Icmeli O, Erengüç SS (1994) A tabu search procedure for the resource constrained project scheduling problem with discounted cash flows. Computers and Operations Research 21:841–853
32. Icmeli O, Erengüç SS (1996) A branch and bound procedure for the resource-constrained project scheduling problem with discounted cash flows. Management Science 42:1395–1408
33. Igelmund G, Radermacher FJ (1983) Preselective strategies for the optimization of stochastic project networks under resource constraints. Networks 13:1–28
34. Kaerkes R, Leipholz B (1977) Generalized network functions in flow networks. Operations Research Verfahren 27:225–273
35. Kamburowski J (1990) Maximizing the project net present value in activity networks under generalized precedence relations. Proceedings of 21st DSI Annual Meeting, San Diego, 748–750
36. Kerzner H (2000) Project Management: A Systems Approach to Planning, Scheduling, and Controlling. Van Nostrand Reinhold, New York
37. Kimms A (2001) Mathematical Programming and Financial Objectives for Scheduling Projects. Kluwer, Boston
38. Klein R (2000) Scheduling of Resource-Constrained Projects. Kluwer, Boston
39. Klein R, Scholl A (1999) Computing lower bounds by destructive improvement: An application to resource-constrained project scheduling. European Journal of Operational Research 112:322–346
40. Klein R, Scholl A (2000) Scattered branch and bound. Central European Journal of Operations Research 7:177–201
41. Kolisch R, Padman R (2001) An integrated survey of deterministic project scheduling. Omega 29:249–272
42. Laborie P (2001) Algorithms for propagating resource constraints in AI planning and scheduling: Existing approaches and new results. In: Proceedings of the IJCAI-01 Workshop on Planning with Resources, Seattle
43. Meridith JR, Mantel Jr SJ (1999) Project Management: A Managerial Approach. John Wiley, New York
44. Michalewicz Z, Fogel DB (1999) How to Solve It: Modern Heuristics. Springer, Berlin
45. Möhring RH (1984) Minimizing costs of resource requirements in project networks subject to a fixed completion time. Operations Research 32:89–120
46. Möhring RH (2000) Scheduling under uncertainty: Optimizing against a randomizing adversary. In: Jansen K, Khuller S (eds.) Approximation Algorithms for Combinatorial Optimization: Third International Workshop (APPROX 2000), Lecture Notes in Computer Science Vol. 1913. Springer, Berlin, 15–26
47. Möhring RH, Schulz AS, Stork F, Uetz M (1999) Resource-constrained project scheduling: Computing lower bounds by solving minimum cut problems. In:

Nešetřil J (ed.) Proceedings of the 7th Annual European Symposium on Algorithms, Lecture Notes in Computer Science Vol. 1643. Springer, Berlin, 139–150
48. Möhring RH, Skutella M, Stork F (2000) Scheduling with AND/OR precedence constraints. Technical Report 689, Technical University of Berlin
49. Morton TE, Pentico DW (1993) Heuristic Scheduling Systems. John Wiley, New York
50. Neumann K (1975) Operations Research Verfahren, Band III. Carl Hanser, München
51. Neumann K (1999) Scheduling of projects with stochastic evolution structure. In: Węglarz J (ed.) Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Boston, 309–332
52. Neumann K, Schwindt C (1997) Activity-on-node networks with minimal and maximal time lags and their application to make-to-order production. OR Spektrum 19:205–217
53. Neumann K, Schwindt C (1999) Project scheduling with inventory constraints. Report WIOR-572, University of Karlsruhe
54. Neumann K, Zhan J (1995) Heuristics for the minimum project-duration problem with minimal and maximal time-lags under fixed resource constraints. Journal of Intelligent Manufacturing 6:145–154
55. Neumann K, Zimmermann J (1999a) Resource levelling for projects with schedule-dependent time windows. European Journal of Operational Research 117:591–605
56. Neumann K, Zimmermann J (1999b) Methods for resource-constrained project scheduling with regular and nonregular objective functions and schedule-dependent time windows. In: Węglarz J (ed.) Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Boston, 261–287
57. Neumann K, Zimmermann, J (2000a) 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
58. Neumann K, Zimmermann J (2000b) Branch-and-bound and truncated branch-and-bound procedures for resource-constrained project scheduling with discounted cash flows and general temporal constraints. Report WIOR-581, University of Karlsruhe
59. Neumann K, Nübel H, Schwindt C (2000) Active and stable project scheduling. Mathematical Methods of Operations Research 52:441–465
60. Neumann K, Schwindt C, Zimmermann J (2001) Project Scheduling with Time Windows and Scarce Resources. Lecture Notes in Economics and Mathematical Systems Vol. 508. Springer, Berlin
61. Neumann K, Schwindt C, Zimmermann J (2002) Order-based neighborhoods for project scheduling with nonregular objective functions. European Journal of Operational Research, to appear
62. Nübel H (1999) A branch-and-bound procedure for the resource investment problem subject to temporal constraints. Report WIOR-574, University of Karlsruhe
63. Nübel H (2001) The resource renting problem subject to temporal constraints. OR Spektrum 23:359–382
64. Özdamar L, Ulusoy G (1995) A survey on the resource-constrained project scheduling problem. IEE Transactions 27:574–586
65. Pinedo M (2001) Scheduling Theory, Algorithms, and Systems. Prentice Hall, Englewood Cliffs

66. Roy B (1964) Les problèmes d'ordonnancement. Dunod, Paris
67. Russell AH (1970) Cash flows in networks. Management Science 16:357–373
68. Schwindt C (1998) Verfahren zur Lösung des ressourcenbeschränkten Projektdauerminimierungsproblems mit planungsabhängigen Zeitfenstern. Shaker, Aachen
69. Schwindt C (2000) Minimizing earliness-tardiness costs of resource-constrained projects. In: Inderfurth K, Schwödiauer G, Domschke W, Juhnke F, Kleinschmidt P, Wäscher G (eds.) Operations Research Proceedings 1999. Springer, Berlin, 402–407
70. Schwindt C, Trautmann N (2000) Batch scheduling in process industries: An application of resource-constrained project scheduling. OR Spektrum 22:501–524
71. Schwindt C, Trautmann N (2002) Storage problems in batch scheduling. In: Chamoni P, Leisten R, Martin A, Minnemann J, Stadtler H (eds.) Operations Research Proceedings 2001. Springer, Berlin, 213–217
72. Schwindt C, Zimmermann J (2001) A steepest ascent approach to maximizing the net present value of projects. Mathematical Methods of Operations Research 23:435–450
73. Schwindt C, Zimmermann J (2002) Parametrische Optimierung als Instrument zur Bewertung von Investitionsprojekten. Zeitschrift für Betriebswirtschaft 72, to appear
74. Selle T (2001) Lower bounds for resource levelling problems with renewable resources. Report WIOR-607, University of Karlsruhe
75. Selle T, Zimmermann J (2000) A bidirectional heuristic for maximizing the net present value of large-scale projects subject to limited resources. Report WIOR-599, University of Karlsruhe
76. Simmons DM (1975) Nonlinear Programming for Operations Research. Prentice Hall, Englewood Cliffs
77. Sprecher A, Kolisch R, Drexl A (1995) Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem, European Journal of Operational Research 80:94–102
78. Stork F, Uetz M (2001) On the representation of resource constraints in project scheduling. Technical Report 693, Technical University of Berlin
79. Trautmann N (2001) Anlagenbelegungsplanung in der Prozessindustrie. Gabler, Wiesbaden
80. Vanhoucke M, Demeulemeester EL, Herroelen WS (2001a) An exact procedure for the resource-constrained weighted earliness-tardiness project scheduling problem. Annals of Operations Research 102:179–196
81. Vanhoucke M, Demeulemeester EL, Herroelen WS (2001b) On maximizing the net present value of a project under renewable resource constraints. Management Science 47:1113–1121
82. Westenberger H, Kallrath J (1995) Formulation of a job shop problem in process industry. Working paper, Bayer AG, Leverkusen and BASF AG, Ludwigshafen
83. Zhan J (1992) Calendarization of time-planning in MPM networks. ZOR — Methods and Models of Operations Research 36:423–438
84. Zimmermann J (2000) Project scheduling with discounted cash flows and general temporal constraints. In: Inderfurth K, Schwödiauer G, Domschke W, Juhnke F, Kleinschmidt P, Wäscher G (eds.) Operations Research Proceedings 1999. Springer, Berlin, 419–424

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