European Journal of Operational Research 208 (2011) 177–205

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Invited Review

# Project scheduling with finite or infinite number of activity processing modes – A survey

Jan Węglarz *, Joanna Józefowska, Marek Mika, Grzegorz Waligóra

*Institute of Computing Science, Poznań University of Technology, ul. Piotrowo 2, 60-965 Poznań, Poland*

ARTICLE INFO

*Article history:*
Received 14 May 2009
Accepted 14 March 2010
Available online 23 March 2010

*Keywords:*
Project scheduling
Multiple processing modes
Deterministic scheduling theory

ABSTRACT

This paper surveys single-project, single-objective, deterministic project scheduling problems in which activities can be processed using a finite or infinite (and uncountable) number of modes concerning resources of various categories and types. The survey is based on a unified framework of a project scheduling model including resources, activities, objectives, and schedules. Most important models and solution approaches across the class of problems are characterized, and directions for future research are pointed out.



## 1. Introduction

Quantitative approaches to project management date back to the 1950s. However, the early models, like CPM and PERT, assumed that activities were characterized by (deterministic or probabilistic) durations only, i.e. that resources were practically unlimited. Since this assumption is unrealistic in majority of practical situations, the impact of limited resources on project characteristics started to be taken into account in such models as CPM/MCX or CPM/resources (notice that at that time resources meant renewable resources, whereas nonrenewable ones were aggregated as cost).

During the last five decades, project management and scheduling became one of the most important directions in both research and practice of operations management, or, more generally, in operational research. This follows from an extremely large-scale of practical situations in which some structured sets of activities have to be processed using various scarce resources.

During these years, the methodology of project scheduling has been developing constantly, trying, from one side to model adequately new practical problems, and, from the other side, to efficiently solve the resulting optimization problems.

Of course, the methodology benefited from the development of both: optimization (especially combinatorial one) and computational possibilities. It is quite evident that a lot of books and papers surveying the state-of-the-art across the field have been published up to date. Some of them, most important for the scope of this survey, will be mentioned in the following sections.

In order to make this survey complementary to the existing ones we concentrated our attention on the single-project problems which are deterministic, single-objective, and multi-mode. Moreover, we do not deal with project scheduling under uncertainty, robust and reactive project scheduling, or flexibility, stability and sensitivity analysis. Extensive surveys on these subjects can be found in (Herroelen and Leus, 2004, 2005; Billaut et al., 2008). Still, we widely characterize resources of all categories, possible trade-off problems, and thoroughly analyze preemptibility issues. The focus of this survey is to consider problems in which activities can be processed using a finite or infinite number of modes, following from their requirements concerning resources of various categories and types. Let us stress that the term "infinite" means, in fact, "uncountable", since countable (and infinite) number of modes has no practical meaning. We carry out our analysis on the basis of a unified framework of the project scheduling model including their most important components, such as resources, activities, objectives, and schedules. An important added value of this survey is pointing out some new problems which are worth being investigated in the future.

Let us also mention that although we mainly focus on problem formulations in this survey, and only give a brief information on the methods applied, we do break this pattern in Section 5. This section is devoted to the multi-mode resource-constrained project scheduling problem which is the main problem tackled in this paper, and therefore for that case we also provide some descriptions of the presented approaches.

The content of the paper is the following.

In Section 2 we remind the formulation of the standard resource-constrained project scheduling problem (RCPSP) which will be the basis for further generalizations. Section 3 describes

* Corresponding author. Tel.: +48 61 8790790; fax: +48 61 8771525.
*E-mail address:* jan.weglarz@cs.put.poznan.pl (J. Węglarz).

0377-2217/$ - see front matter 
doi:10.1016/j.ejor.2010.03.037

components of the project scheduling model, i.e. resources, activities, objectives, and schedules. In Section 4 we survey the results concerning project scheduling problems for various time/cost and time/resource trade-offs. In Section 5 the same is done for the multi-mode RCPSP (MRCPSP) which includes also resource-resource trade-offs. Section 6 deals with multi-mode problems with objectives other than the makespan, from among which the maximization of the net present value (NPV) is discussed most extensively. In Section 7 a brief survey on discrete–continuous project scheduling is given. Various assumptions concerning preemptibility of activities are discussed in Section 8. The paper ends with suggestions for future work (Section 9) and with an updated bibliography.

## 2. Resource-constrained project scheduling problem

Practice shows that resources constitute an essential feature of any project. In this section we present a formulation of the basic *resource-constrained project scheduling problem*, referred to as the RCPSP. The RCPSP is a classical discrete problem, i.e. the planning horizon is divided into a discrete number of time periods, activity durations are discretely-divisible, and resources are discrete (see Section 3.1.1). The following sections of the paper illustrate the extensions of the basic model, introduced in order to find more adequate descriptions of practical situations.

Let us consider a set of $n$ nonpreemptable activities of durations $d_i,\ i = 1,\ldots,n$. Precedence constraints between activities mean that no activity may start before all its predecessors are completed. Activities are labeled from $A_0$ to $A_{n+1}$, with activity $A_0$ being the unique initial activity without predecessors (source), and $A_{n+1}$ being the unique terminal activity without successors (sink). If such an activity $A_0$ (or $A_{n+1}$) does not naturally exist, then a dummy activity of zero duration and zero resource requirements is added appropriately.

Moreover, each activity requires some discrete renewable resources, i.e. such that only their temporary availability at every moment is constrained (see Section 3.1.1). We assume that there are $R$ scarce resources and the number of available units of resource $k$, $k = 1,\ldots,R$, is $R_k$. Moreover, all activities and resources are available at the start of the project. The objective of the RCPSP is to find precedence- and resource-feasible completion (or start) times for all activities such that the duration of the project is minimized.

The RCPSP may be formulated as an integer programming problem. The 0–1 decision variable $x_{jt} = 1$ if activity $A_j$ is assigned a completion time at the end of period $t$; otherwise, $x_{jt} = 0$. Associated with each activity $A_j$ are its earliest finish time $EF_j$, and latest finish time $LF_j$, calculated as in (Kelley and Walker, 1959). The value of $LF_{n+1}$ is set equal to the scheduling horizon $H$, which never exceeds the sum of all activity durations. The following formulas (1)–(5) define the problem (Pritsker et al., 1969):

$$\text{Minimize} \quad \sum_{t=EF_{n+1}}^{LF_{n+1}} t x_{n+1,t} \tag{1}$$

$$\text{Subject to} \quad \sum_{t=EF_j}^{LF_j} x_{jt=1} \quad \text{for } j = 0,\ldots,n+1 \tag{2}$$

$$\sum_{t=EF_i}^{LF_i} t x_{it} \leqslant \sum_{t=EF_j}^{LF_j} t x_{jt} - d_j \quad \text{for all } (A_i, A_j) \in P \tag{3}$$

$$\sum_{j=1}^{n} \sum_{q=max\{t,EF_j\}}^{min\{t+d_j-1,LF_j\}} r_{jk} x_{jq} \leqslant R_k \quad \text{for } k = 1,\ldots,R; \quad t = 1,\ldots,H \tag{4}$$

$$x_{jt} \in \{0,1\} \quad \text{for } i = 0,\ldots,n+1; \quad t = EF_j,\ldots,LF_i \tag{5}$$

Constraints (2) ensure that each activity is completed exactly once. The set of all pairs of activities $(A_i,\ A_j)$ such that $A_i$ directly precedes $A_j$ is denoted by $P$. Hence, precedence constraints are represented by inequalities (3). Constraints (4) guarantee that no more than the available number of units of each resource are required in any time period, and constraints (5) state that we consider binary decision variables. The solution of the problem (1)–(5) defines an optimal schedule as a list of activity completion times.

The RCPSP is strongly NP-hard (Błażewicz et al., 1983). Analyses of the problem resulted in the development of exact as well as heuristic algorithms. Surveys of approaches and results for the RCPSP may be found in (Icmeli et al., 1993; Elmaghraby, 1995; Özdamar and Ulusoy, 1995; Herroelen et al., 1998; Brucker et al., 1999; Kolisch and Padman, 2001; Demeulemeester and Herroelen, 2002). Hartmann and Kolisch (2000) and Kolisch and Hartmann (2006) provide an extensive comparison of computational results obtained by the best heuristic algorithms known to date.

## 3. Project scheduling components

### 3.1. Resources

Each activity (except dummy ones) requires for its processing some resources. Examples of resources are: machines and tools, human and their skills, raw materials and semi-finished products, natural resources (energy, water, land, etc.), information, money, etc.

In general, resources can be considered at three levels of abstraction: categories, types, and units. A resource unit is a basic amount of a given resource. A resource type is a set of resource units identical in the sense of the ability to fulfill the same functions. And finally, resource category is a set of resource types having the same characteristic features like, for example, the same kind of divisibility, the same kind of availability limitations or the same type of functionality constraints.

Resources are divided into categories taking into account three points of view: resource constraints, resource divisibility, and resource preemptibility. In this section both basic and new categories of resources within the above three points of view are shortly described according to these points of view.

#### 3.1.1. Resource categories by resource divisibility

Two resource categories can be distinguished from the viewpoint of resource divisibility: discrete (i.e. discretely-divisible) and continuous (i.e. continuously-divisible) ones. A *discrete* resource can be allocated to activities in discrete amounts from a given finite set of possible allocations. A *continuous* resource can be allocated in arbitrary, a priori unknown, amounts from a given interval. Typical examples of discrete resources are: machines, tools, workers, whereas examples of continuous resources include: energy, liquids and money.

#### 3.1.2. Resource categories by resource preemptibility

From the viewpoint of the resource preemptibility, resources can be either preemptable or nonpreemptable (Błażewicz et al., 1986). A resource is *preemptable* if each of its units may be preempted, i.e. withdrawn from the processing of a current activity, allotted to another activity, and then returned to the previously interrupted activity whose processing may be resumed, as though resource preemption had not occurred. Resources without this property are called *nonpreemptable*.

Notice that the occurrence of nonpreemptable resources may result in deadlocks, as it is considered in operating systems theory (see, e.g. Coffman and Denning, 1973). Consequently, resources of this type are of particular importance in computer systems, and will not be taken into account in this survey. Thus, preemptibility becomes a feature of the set of activities which will be defined in Section 3.2.8 and discussed in Section 8.

*3.1.3. Resource categories by resource constraints*

Most of the resource categories are considered from the viewpoint of resource constraints. There are three basic and several new categories of resources. Basic resource categories are renewable, nonrenewable, and doubly constrained resources. They are taken into account in many project scheduling papers which were published over last few decades. All the new categories have been introduced in recent years.

*3.1.3.1. Basic categories.* The earliest papers considered two categories of resources from the viewpoint of their availability namely renewable resources represented by labor, machinery, equipment, etc., and nonrenewable ones usually represented by cost (money).

The definitions of these two basic resource categories were proposed by Słowiński (1980) and Węglarz (1980a, 1981). They also introduced one new category called doubly constrained resources.

As we have mentioned, a resource is *renewable* if only its total usage, i.e. temporary availability at every moment, is constrained. In other words, the available amount of such a resource is constant at every moment of the planning horizon. Usually, units of such resources used during the execution of an activity are released immediately after the completion of this activity, and they can be used by other activities in the following periods of the planning horizon.

A resource is *nonrenewable* if only its total consumption, i.e. integral availability up to the project deadline or a given moment, is limited. In other words, the capacity of a nonrenewable resource is limited for the entire project (or a given time interval). Units of a nonrenewable resource are consumed by an activity during its processing, and cannot be assigned to any other activity.

The availability of a *doubly constrained* resource is limited both for the entire project (time interval) and at every moment. These resources can be either consumed (e.g. money, raw materials, etc.) or used (e.g. blades, cartridges, etc.) by an activity during its execution. According to the transformation proposed by Talbot (1982), each doubly constrained discrete resource can be replaced by a pair of renewable and nonrenewable discrete resources, assuming that the number of units consumed during the execution of an activity is constant over the whole activity duration.

*3.1.3.2. Other categories.* Several new categories of resources have been introduced in recent years. We present these categories as they were introduced, i.e. as discretely-divisible and limited per period or set of periods (time is a discrete parameter). However, in many cases it is also possible to treat these resources as continuous ones, and consider their availabilities as limited at every time instant or in time intervals (time as a continuous parameter).

Let us start the review of the new categories with *partially* (non)renewable *resources*. This resource category was introduced by Böttcher et al. (1999), in order to define resources with availabilities defined for subsets of periods. For each partially (non)renewable resource type, there are a number of subsets of periods (time intervals), each characterized by a given availability of the resource units. Notice, that these subsets need not be disjoint, moreover, they may cover time periods which are not consecutive. The concept of partially (non)renewable resources can be viewed as a generalization of the three abovementioned basic categories of resources.

Some authors (cf. Baptiste et al., 1999; Artigues, 2008) distinguish a special case of renewable resources called *unary* (or *disjunctive*) resources with availabilities limited to one unit per period.

*Dedicated resources*, considered by Bianco et al. (1998), can be assigned to only one activity at a time. This feature causes that a dedicated resource can be, in fact, expressed by a unary resource.

A *spatial resource*, introduced by de Boer (1998), is defined as a resource used by a group of activities. It is occupied by these activities from the start of the first activity till the completion of the last activity of the group. If it is possible, i.e. if other resource and precedence constraints permit, all activities from the same group can be scheduled simultaneously. However, it is forbidden to execute activities from different groups using the same spatial resource unit at a time. Examples of spatial resources include dry docks of a shipyard, shop floor space, rooms, pallets, containers, etc.

For some resources a physical location of a particular resource unit among other units of the same type of resource is given, and it is important for activity processing. Such resources are called *adjacent resources* (Duin and Van Der Sluis, 2006; Paulus and Hurink, 2006). Examples of such resources include adjacent parts of dry docks in a shipyard and check-in desks at airports, which can be treated as special cases of spatial resources, as well as other types of resources like processors in multi-processor or grid environments.

*Cumulative resources* (cf. Neumann et al., 2002) are considered when the availability of a resource in a given time period cumulatively depends on the utilization (depletions and replenishments) of this resource during the previous time periods. Examples of such resources are investment capital, storage facilities, intermediate products. The availability of a cumulative resource in a given time period can be bounded from below (e.g. safety stock) and from above (e.g. the capacity of storage facility). There are some similarities between nonrenewable and cumulative resources. Indeed, a nonrenewable resource can be treated as a cumulative resource without replenishment ability and with an availability lower bound equal to zero. In other words, the concept of cumulative resources is a generalization of the concept of nonrenewable resources. Notice that the term "cumulative resource" is used by some authors (cf. Baptiste et al., 1999; Artigues, 2008) for discrete renewable resources with the capacity greater than one unit.

*Reusable resources* (Shewchuk and Chang, 1995) represent resources which act like renewable ones, but are consumed by little in each period they are used. Examples of such resources are tools with single or multiple cutting edges. After some number of periods in which such a resource is used, it becomes too dull, breaks or something else happens that makes this resource unable to be used for the execution of other activities. Such a "consumed" resource may be returned to the resource pool of the project, but first it must be recycled (e.g. cutting edges have to be reground). Such recycling takes time and, in consequence, each reusable resource becomes unavailable for some time periods.

*Synchronizing resources* are a special category of renewable resources introduced by Schwindt and Trautmann (2003). Units of a synchronizing resource have the ability to ensure a simultaneous start of a set of activities to which these units are allocated. For example, if one has to exam a large group of students (e.g. all students from the same year of the same field of study), which is greater than the size of the classroom, then the whole student group may be divided into subgroups (e.g. laboratory groups) of known sizes less than the capacity of the classrooms. In consequence, an exam can be represented by a set of single activities, each of them being a part of this exam for a given subgroup of students. Since the exam has to start at the same time for all subgroups of students, the classrooms should be modeled as a synchronizing resource. Notice that if the division into subgroups is not arbitrary and is known a priori, than we cannot model the exam as one activity with multiple execution modes differing in allocated classrooms, but we need to model the exam as a set of activities (one for each subgroup).

The main feature of *multi-skill resources* (Néron, 2002) is the so-called resource flexibility, which means that such a multi-skill resource is able to be allocated for different kinds of resource requirements. In other words, a multi-skill resource has the functionality of several renewable resources. Typical examples of multi-skill resources commonly used in the literature are staff members (each of them having his own set of skills) but other examples like multi-skill machines, tools, robots, etc. can also be considered. Although each unit of a multi-skill resource usually has more than one skill, it can still be assigned to only one activity at a time. Moreover, in some situations it is necessary to take into account that there exist *hierarchical levels* for each skill (Toroslu, 2003). It is assumed that a given multi-skill resource unit has a skill level equal to or less than a larger level it masters. In problems where such hierarchical levels of skills are considered, it is necessary to specify the minimal required levels of the skills for each activity.

Considered by Tiwari et al. (2009) *heterogeneous resources* may be viewed as something between the concepts of renewable resources and multi-skill resources. A typical example of such a resource is a set of workers. Each individual treated as a unit of a heterogeneous resource has the same skill (like renewable resources), but some individuals may be less or more skilled than the others (like multi-skill resources with hierarchical levels). The time needed to perform an activity and the quality of the performed work depend on the skill level of the individuals assigned to the activity. As the number of highly skilled workers is usually smaller than the number of less skilled workers, it is possible to begin the work by a less skilled worker and finish it by a higher skilled one. Such an approach usually guarantees to finish the work earlier, and with a proper quality level.

Another category of resources is considered in (Neumann et al., 2006), when units of renewable resources need a setup (or must be changed over) when passing from one activity $A_i$ to another activity $A_j$. Such resources are called *changeover resources*. It is assumed that setup is sequence-dependent, i.e. the setup time depends on both the consecutive activities $A_i$ and $A_j$ for which the units of the changeover resource are allocated. Changeover resources are, for example, used when a project is simultaneously executed in several geographically distributed locations, and some renewable resources can be used in each location, but in order to be put into service for processing activities in a given location, it is necessary to transport units of such resources from another location. Thus, the time needed for transport is treated as a setup (changeover) time.

*Allocatable resources* are special renewable resources introduced by Schwindt and Trautmann (2003). If an activity $A_i$ requires a number of units of such a resource, they have to be allocated to this activity by another activity $A_k$ (so-called allocating activity) starting no later than activity $A_i$. In other words, activity $A_k$ prepares a given number of units of an allocatable resource for processing activity $A_i$. Thus, these allocatable resource units remain occupied from the start of the allocating activity $A_k$ until the completion of $A_i$. Allocatable resources can be viewed as renewable resources which require some setup operations before they are ready to process a given activity. An example of such a resource is equipment that has to be installed each time before it is used for executing some activity $A_i$. If this setup operation also requires some scarce renewable resources (like staff or tools), it can be modeled as an activity $A_k$.

Presented by Mika et al. (2006), *auxiliary resources* are considered in the context of setup operations. Resources of this kind are used only for setting up units of other resources, and are not used during the processing of any project activity. If the number of auxiliary resources is greater than one, then it is possible in some cases to perform a setup in one of several ways (so-called setup modes), differing among themselves in the duration and the auxiliary resources necessary to perform the setup. An example of an auxiliary resource is a set of workers who are capable of and responsible for proper setting up other resources like: specialized machines, computers, robots, etc. Usually such a group of workers can be divided into several subgroups according to their skills, resulting in several types of auxiliary resources (e.g. ordinary workers, skilled workers, and experts), which may be used in several combinations to perform setups in different setup modes. It is worth mentioning that a similar concept of resources, so-called *complementary resources*, was previously presented by Artigues and Roubellat (2001) in the context of generalized job shop scheduling problems.

One may notice that both allocatable and auxiliary resources are used in the context of setup operations. However, they have different functionalities. Combining both concepts in one model results in following conclusions. Auxiliary resources are used during setup operation (denoted as $A_k$), to set up allocatable resources which are next used to perform an activity $A_i$. The preparation of allocatable resources for executing activity $A_i$ starts at the beginning of the setup (i.e. activity $A_k$), and next they are used during the execution of activity $A_i$. Therefore they cannot be released until the end of activity $A_i$, whereas auxiliary resources, which are used for setup operations only, are released after the completion of activity $A_k$, and can be used by another setup operation at the same time when activity $A_i$ is executed.

### 3.2. Activities

Usually each activity is characterized by: resource requirements, processing model, and precedence constraints with other activities, but other parameters can occur in the problem formulation as well. Main parameters concerning activities will be discussed in this section.

#### 3.2.1. Resource requirements

Each activity requires for its processing some scarce resources. The amount of a resource needed to execute a given activity is called *resource requirement* (resource demand, resource request). While resource requirements for discrete resources can be strictly discrete only and are usually expressed in integer values, requirements for continuous resources can be either discrete or continuous. It is also possible that both types of resource requirements for the same continuous resource may occur in a project. Usually, the resource requirement is constant during the processing of an activity, but it may also change in time if variable resource requirements are allowed. If multiple resources are considered, it is common to represent the resource requirements of an activity as a vector. Each element of the vector corresponds to a different resource and its value represents the amount of the resource required by the activity. In practice, it often happens that activity processing time depends on the resource requirements. A pair whose first element is the vector of resource requirements and the second element is activity processing time is called *a mode*. Thus, in such a case for each activity a finite set of modes is defined which represent the alternative ways of processing the activity.

#### 3.2.2. Activity processing model

*Activity processing model* describes a processing characteristic of an activity as a function of resource amounts allotted to this activity. Two types of the activity processing characteristic have been considered up to now, namely activity processing time and activity processing rate.

In the *processing time vs. resource amount* activity model, the activity processing time is a nonincreasing function of the amounts of the resources (in general, renewable, nonrenewable, and doubly constrained ones) allotted to this activity. In the simplest case, as it is for the RCPSP, the activity processing time is a fixed integer denoting the duration of an activity under fixed resource requirements. If an activity can be executed in a finite number of modes, its processing time is a vector of possible durations for different modes. In the case of infinite (uncountable) number of modes, the processing time is a continuous function of the allotted resource amounts.

For the case of discrete resources, the resource-duration interaction model was introduced by Elmaghraby (1977) for nonpreemptable activities, extended by Słowiński (1980) for preemptable activities, and further analyzed by Talbot (1982). A straightforward generalization of this model for continuous resources was first presented by Błażewicz et al. (1986).

Notice that such functions (continuous or discrete) relating the activity processing time with resource amounts allotted to the activity are considered in the project scheduling literature as time/resource and time/cost trade-offs. In fact, they both are special cases of the processing time vs. resource amount activity model. In the case of the *time/resource trade-off*, the processing time of an activity is a nonincreasing function of the usage of renewable resources allotted to this activity. In the case of the *time/cost trade-off*, the processing time of an activity is a nondecreasing function of the activity execution cost representing the total consumption of all nonrenewable resources allotted to this activity. Particular project scheduling problems based on these two trade-offs, where the functions are either discrete or continuous, are extensively discussed in Section 4.

Moreover, notice that the *resource/resource trade-off*, considered in the MRCPSP (see Section 5) is another special case of the processing time vs. resource amount model, where the activity processing time is a constant function of the resource amounts allotted to the activity. This trade-off has never been considered in the context of continuous resources, although such a generalization is possible.

The other activity processing model considered in the literature is the *processing rate vs. resource amount* model in which the activity processing rate is an increasing function of the amounts of the resources allotted to this activity at a time. This model is discussed in Section 7 for the case of a single continuous (renewable or doubly constrained) resource.

### 3.2.3. Precedence constraints

The set of activities of a project is partially ordered by a *precedence relation* $\prec$ , due to technical or other reasons. $A_i \prec A_j$ means that the processing of activity $A_i$ must be completed before activity $A_j$ is started. This relation usually is represented by an acyclic, connected digraph. It is assumed that the nodes of the graph are labeled topologically, and there is exactly one starting node (source) and exactly one finishing node (sink). There are two possible representations of the precedence relation: so-called AoA (*Activity-on-Arc*) and AoN (*Activity-on-Node*) networks. In the AoA network (Kelley and Walker, 1959; Malcolm et al., 1959) the set of arcs represents activities, while the set of nodes represents events. In the AoN network (Fondahl, 1962) the set of nodes represents activities, and the set of arcs represents precedence constraints between activities.

It is well-known that every set of precedence constraints has an AoN representation, but this is not true for the AoA representation if dummy activities are not allowed. Moreover, it is straightforward to convert any AoA network into an equivalent AoN network, but the reverse operation is not that simple, because in many examples it may be necessary to introduce some dummy activities and some dummy events. Dummy activities are inserted into an AoA network in order to: (i) preserve uniqueness of activities, (ii) satisfy the condition of a unique initial and terminal event, (iii) correctly represent all required precedence constraints. Moreover, Krishnamoorthy and Deo (1979) showed that finding an AoA network with the minimum number of dummy activities required to satisfy the given precedence relation is NP-hard.

The standard precedence constraints indicate that no activity may be started before all its predecessors are completed. This idea is extended in the *precedence diagramming* by introducing *minimal time lags* (see, e.g. Moder et al., 1983). The idea of precedence diagramming was proposed by Fondahl (1962), but time lags were probably first developed by B. Roy of the Metra Group in France, in the Metra Potential Method (MPM) (see Roy, 1962). A minimal time lag $\underline{\delta}_{ij}$ is defined as the minimum time interval (number of time units) that must be completed between start (finish) of activity $A_i$ and start (finish) of another activity $A_j$. There are four possible types of minimal time lags: start-to-start, start-to-finish, finish-to-start, and finish-to-finish. Any type of minimal time lags can be easily converted to another type, except for the case of more than one activity processing mode. In some cases negative minimal time lags are considered, as well. Note, that the standard type of precedence constraints is a typical strict finish-to-start relationship with zero minimal time lag.

Another more general kind of precedence constraints, named the *generalized precedence constraints* (GPRs), was proposed by Elmaghraby and Kamburowski (1989, 1992). In this type of precedence constraints both minimal and maximal time lags occur. A maximal time lag $\bar{\delta}_{ij}$ between activities $A_i$ and $A_j$ denotes that $A_j$ may not be started (finished) later than $\bar{\delta}_{ij}$ time periods after the start (finish) of activity $A_i$. Similar to the minimal time lags, four possible types of maximal time lags are considered in the literature: start-to-start, start-to-finish, finish-to-start, and finish-to-finish. A maximal time lag can be represented by a minimal time lag in the opposite direction with a negative value. As a result, there arise cycles in the activity network, and therefore special approaches are usually needed to solve project scheduling problems with maximal time lags.

### 3.2.4. Time parameters

The life cycle of an activity of a project can be determined by the following time parameters: a ready time, a due date or a deadline. *Ready time* (also known as *release date*) is a time at which an activity is ready for processing. In many project scheduling problems ready times for all activities are identical, then it is assumed that they are all equal to 0. Moreover, ready time for activity $A_j$ may be represented by the minimal start-to-start time lag $\underline{\delta}_{0j}$ between the source activity $A_0$ and activity $A_j$.

*Due date* specifies a time limit by which an activity should be completed. It can be defined either for particular activities or for the entire project, as well as for both. Usually penalty costs per time unit are considered for activities completed either before or after their due dates.

*Deadline* can be defined similarly to due date. The main difference is that a deadline cannot be violated. Deadline for activity $A_j$ can be represented by the maximal start-to-finish time lag $\bar{\delta}_{0j}$ between the source activity $A_0$ and activity $A_j$.

As mentioned before, both ready times and deadlines can be modeled by generalized precedence relations, but the reverse operation (i.e. modeling GPRs by ready times and deadlines) is not possible. However, in some applications (see, e.g. Baptiste et al., 1999) ready times and deadlines can be used instead of precedence constraints.

### 3.2.5. Weights

Sometimes some additional parameters are defined for activities. These parameters are commonly named *weights*, and often occur in the objective function of the problem. Practical interpretation of weights depends on the problem. Once weight may denote a priority of an activity, otherwise it may denote some additional cost (or reward) for executing an activity, or something else.

#### 3.2.6. Cash flows

In problems where financial objectives are considered, a series of *cash flows* occur over the course of the project. From the viewpoint of the contractor (i.e. the executor of the project), negative cash flows (cash outflows) are usually his expenditures induced by the use of various resources during the execution of activities. Positive cash flows (cash inflows) usually represent payments for the realized parts of the project, and occur at the completion times of some activities or at the prescribed moments over the project planning horizon. In most cases time value of money is taken into account by discounting the cash flows.

#### 3.2.7. Setups

In many practical situations resources should be specially prepared for executing some activities. Operations performed in order to prepare such resource units are called either *setups* (changeovers), if they occur before an activity is started, or *removals*, if they occur after the completion of an activity. Setups must be performed directly before (or directly after, in the case of removals) the execution of an activity, and therefore cannot be considered as separate activities. A setup is measured by the time needed to perform it, and additionally by the cost of such an operation. There are two types of setup times directly taken from the machine scheduling theory: *sequence-independent* and *sequence-dependent* ones (see, e.g. Allahverdi et al., 1999, 2008). While a sequence-dependent setup time varies depending on the sequence of activities performed on a given resource unit, a sequence-independent setup time is the same for all possible sequences of activities. Recently, Mika et al. (2003, 2008) introduced the concept of a *schedule-dependent* setup time which depends not only on a sequence of activities on a given resource unit but also on the resources allocated to preceding activities. Some other types of setups which can occur in project scheduling problems are studied in (Mika et al., 2006).

#### 3.2.8. Preemptibility

In many models of project scheduling problems it is assumed that activities are nonpreemptable, but in some projects this assumption is relaxed and it is allowed to preempt activities. In general, each of the project activities may be either preemptable or nonpreemptable. However, usually it is assumed that (non)preemptibility concerns all activities at once. Under this assumption, we talk about a set of nonpreepmtable activities if none of them may be preempted, whereas we talk about a set of preemptable activities if each activity can be preempted at any time and restarted later with no cost. Preemption may be either discrete, if activity preemption is allowed at the end of time periods only, or continuous, if preemption may occur at an arbitrary time instant. Activity preemption is also called activity splitting. For a brief discussion on preemptibility in project scheduling see Section 8.

### 3.3. Objectives

Motivated by real-world situations, a wide variety of objectives for project scheduling have been studied in the literature. Let us just mention that from among all of them, some objectives may be related to time, as they concern temporary usage of renewable and doubly constrained resources, whereas others – to cost, as they deal with consumption of nonrenewable and doubly constrained resources. These two kinds usually represent conflicting objectives, since shortening the processing time results in increasing the resource consumption, and vice versa – decreasing the execution cost (in terms of the resources consumed) lengthens the project duration. Thus, involving resources of various categories (renewable, nonrenewable, doubly constrained) in one model requires considering both kinds of objectives simultaneously, which basically leads to multi-objective problems. Usually, nonrenewable resource availabilities are incorporated to the problem constraints and a single (mostly time-based) objective is considered, but – it must be stressed – this is a simplifying assumption. As mentioned in Section 1, we do not deal with multi-objective problems in this survey. As far as only discrete resources are concerned, we refer in this subject to the review by Ballestín and Blanco (2010). In fact, the first analysis considering multi-objective problems with discrete resources of various categories is presented in (Słowiński, 1981). The only work on multi-objectiveness for continuous resources is presented in (Węglarz, 1991), where the most general case of doubly constrained continuous resources is considered.

Further, we consider only problems with a single objective. Classifications of various optimization objectives may be found in (Brucker et al., 1999; Demeulemeester and Herroelen, 2002; Błażewicz et al., 2007). The objectives may be related to time, resource usage, project cost, or other performance measures. The objective functions used to evaluate projects can be categorized into *regular* and *nonregular* measures of performance. A regular measure of performance is a nondecreasing (in the minimization case) function of the activity completion times. Regular measures are also referred to as *early completion measures*. A measure of performance not possessing the above property is considered *nonregular*. Intuitively speaking, delaying activities may improve the value of a nonregular measure of performance. Nonregular performance measures are sometimes referred to as *free completion measures*.

#### 3.3.1. Time-based objectives

A time-based objective is a function of activity completion times ($C_j$) and is minimized. The most natural performance measure in this group is the *makespan*, i.e. the function $\max\{C_j\}$, which represents the project duration. Minimizing the project duration is one of the most common objectives used in project scheduling, with strong practical motivation. Majority of problems discussed in this survey are makespan minimization problems, however, other objectives are also considered in the literature. For example, Słowiński (1989) considers minimization of the *mean weighted flow time*. For some technical reasons Ballestín and Blanco (2010) introduce a similar time-based objective which is minimization of the *total weighted sum of start times*. Other time-based measures are calculated using the difference $L_j$ between activity completion time and its due date, called *lateness*. Maximum lateness $L_{max} = \max\{L_j\}$ is the most commonly known objective. Another example is the *mean weighted lateness* considered by Słowiński (1989). Since an activity may finish either before or after its due date, two nonnegative values are defined for each activity: tardiness $T_j = \max\{0, L_j\}$ and earliness $E_j = \max\{0, -L_j\}$. An objective function may be then defined as a weighted sum of total (or maximum) earliness and tardiness (Błażewicz et al., 2007). In this way we may define the *mean* (*weighted*) *tardiness*, the *mean (weighted) earliness* (Błażewicz et al., 2007), and other objectives. Moreover, minimization of the (*weighted*) *number of tardy activities* may also be desired (Słowiński, 1989). Majority of time-based objectives are regular measures of performance. Only simultaneous minimization of activity earliness and tardiness leads to nonregular measures.

#### 3.3.2. Resource-based objectives

A resource-based objective is a function of resource usage, representing the cost of resources required for the completion of the project. An important example of an objective of this type is the so-called *resource availability cost problem*. The goal is to minimize the availability cost (*procurement cost*) of the renewable resources while the project should be completed by a predefined deadline. This problem was introduced by Möhring (1984) as the *resource investment problem*, together with an exact procedure based on graph theoretical algorithms.

Another example is the *resource levelling problem* which consists in constructing a time-feasible schedule for which the usages for various resources over time are as balanced as possible. The quality of solutions may be measured by the sum of absolute or squared deviations of the resource usage around some predefined level, for example around the average resource usage (Burgess and Killebrew, 1962). Another suitable objective may be to minimize the *total adjustment cost*, i.e. the cost of jumps in the resource usage for each resource type (Brinkman and Neumann, 1996). This objective reflects, e.g. the costs of hiring and laying off workers in consecutive periods. We may easily imagine a situation where only the resource usage exceeding some predefined level is penalized. Then, it is suitable to minimize the total cost of the resource usage exceeding the given limit, so-called *total overload*.

Finally, in the multi-mode problems a natural objective is to minimize the total consumption of nonrenewable resources without violating the project deadline, i.e. minimization of the *mode-dependent project cost* or *weighted resource consumption* (Słowiński, 1989).

Another issue occurs when resources may be either purchased or hired. The objective is then to minimize the *total renting cost* (Neumann et al., 2002).

Resource-based objectives are, in general, nonregular measures of performance.

#### 3.3.3. Financial objectives

During the execution of the project incoming and outgoing cash flows may take place. Cash outflows are usually induced by the execution of project activities and the use of resources. Cash incomes are usually the result of payments upon completion of certain project parts. In some models, cash flows occur in some predefined time periods. Maximization of the *net present value* (NPV) of the project is an objective reflecting the demand of effective usage of money spent on the project. Time value of money is taken into account by discounting the cash flows. It is worth noticing that if positive and negative cash flows are considered simultaneously, maximization of the net present value becomes a nonregular measure of performance.

The objective of maximization the net present value was introduced to project scheduling by Russell (1970). Extensive surveys on the unconstrained project scheduling problem with discounted cash flows (so-called max-NPV problem), as well as the resource-constrained project scheduling problem with discounted cash flows (RCPSPDCF) to maximize the net present value can be found in (Icmeli et al., 1993; Özdamar and Ulusoy, 1995; Herroelen et al., 1997; Kimms, 2001; Kolisch and Padman, 2001; Demeulemeester and Herroelen, 2002; Drezet, 2008). In this survey, problems with multiple processing modes and discounted cash flows to maximize the NPV will be discussed in Sections 4.1.2 and 6.

#### 3.3.4. Other objectives

Obviously, practical applications lead to many other optimization objectives that may be considered in project scheduling, not all of them being easy to classify. It is worth mentioning that quite a lot of research is devoted to the analysis of project robustness or stability. The motivation is to develop schedules for which unexpected events occurring during the realization of the project may be easily handled without additional cost (Demeulemeester and Herroelen, 2002). Icmeli-Tukel and Rom (1997) propose to minimize the *estimated rework time and cost*, in case some project activities need to be reworked. In some projects some setup time is required before an activity may start using a certain resource. Then, the *minimization of setup time* or *cost* may be desired.

### 3.4. Schedules

In the classical RCPSP a *schedule* is defined by a sequence of activity start (completion) times, but it is insufficient for problems where activities can be executed in multiple modes. Thus, in these cases additional information about processing modes is necessary.

A schedule is *time-feasible* if it satisfies all precedence and time constraints defined for the project, and *resource-feasible* if all resource constraints are met. A schedule is said to be *feasible* if it is both time- and resource-feasible. An *optimal schedule* is a feasible schedule for which a given performance measure is optimized.

Sprecher et al. (1995) define three other kinds of schedules based on classification of schedules for the job shop problem. A feasible schedule is *semi-active* if none of the activities can be locally left shifted (no activity can be started earlier without changing the processing order or violating the constraints). A schedule is called *active* if it is feasible and none of the activities can be locally or globally left shifted (no activity can be started earlier without violating the constraints). Finally, a *non-delay* schedule is defined as a feasible schedule for which the corresponding unit-time-duration (each activity is divided into $d_j$ activities with duration equal to one) schedule is active. The following relation holds: $NDS \subseteq AS \subseteq SAS \subseteq FS \subseteq S$, where $S$, $FS$, $SAS$, $AS$ and $NDS$ denote sets of schedules, feasible schedules, semi-active schedules, active schedules and non-delay schedules, respectively. It is worth mentioning that for regular performance measures an optimal schedule is always an active schedule, but for some problem instances it may not belong to the set of non-delay schedules.

## 4. Trade-off problems

### 4.1. Time/cost trade-off problem

The *time/cost trade-off problem* (TCTP) is one of the problems where activities can be executed in multiple (infinite, in general) modes. As mentioned in Section 3.2.1, in the case of the time/cost trade-off, the processing time of an activity is a nondecreasing function of the activity execution cost. The cost of an activity represents an aggregated consumption of all nonrenewable resources allotted to this activity. As a result, in the TCTP a single nonrenewable resource is considered whose consumption by an activity represents the cost corresponding to this activity. In the simplest situation the nonrenewable resource can be money, which represents cost in a direct way, but also other nonrenewable resources can be considered. In each case, the duration of an activity is a function of its cost, represented as the nonrenewable resource consumption. Normally, the acceleration in the activity execution comes at some cost, and vice versa – accomplishing an activity in a longer duration reduces the cost corresponding to this activity. The duration of each activity is bounded from below by the crash duration, which corresponds to the maximum allocation of the resource, and from above by the normal duration, which corresponds to the minimum (normal) nonrenewable resource allocation.

Two objectives are commonly considered in the TCTP. Firstly, under a given project deadline, the objective is to find activity durations and a schedule that minimize the total project cost (so-called *deadline problem*). The second objective reverses the problem formulation. Now, a limited availability of the resource is given, and the goal is to find activity durations and a schedule that minimize the project duration (so-called *budget problem*).

#### 4.1.1. Continuous time/cost trade-off problem

In the *continuous time/cost trade-off problem* (CTCTP) the duration of an activity is described by a continuous nonincreasing function of the activity cost. This function is called direct activity cost function. The literature has provided extensive analyses of the CTCTP over the years. For a comprehensive survey we refer to (Demeulemeester and Herroelen, 2002).

#### 4.1.2. Discrete time/cost trade-off problem

In the *discrete time/cost trade-off problem* (DTCTP) the duration of an activity is a discrete, nonincreasing function of the amount of the single nonrenewable resource allotted to this activity. As a result, each activity can be executed in several processing modes, following from all possible resource allocations. In the DTCTP, also a third objective is considered, apart from the two mentioned earlier, namely the construction of the efficient time/resource profile over the feasible project durations. Surveys on the DTCTP can be found in (De et al., 1995; Kolisch and Padman, 2001; Demeulemeester and Herroelen, 2002).

Since then the DTCTP has been considered in several papers.

Vanhoucke et al. (2002) describe a solution procedure for the DTCTP in which three special cases of time-switch constraints are involved. These constraints impose a specified starting time on the project activities and force them to be inactive during specified time periods. The authors propose a branch-and-bound (B&B) algorithm and a heuristic procedure which both make use of a lower bound calculation for the DTCTP (without time-switch constraints). The procedures were validated on a randomly generated problem set. The authors also discuss an illustrative example based on a real-life situation. Then, in (Vanhoucke, 2005) a new B&B algorithm for the same DTCTP with time-switch constraints is proposed, which outperforms the previous one.

Akkan et al. (2005) give a comprehensive literature review on the DTCTP, including exact algorithms, lower bounds, approximation algorithms, and heuristic algorithms. The paper provides lower and upper bounds using column generation techniques based on network decomposition. Furthermore, a computational study is provided to demonstrate that the presented bounds are tight, and that large and hard instances can be solved in short run-time.

Peng and Wang (2008) extend the general DTCTP to a new multi-mode resource-constrained DTCTP model (MRC-DTCTP) in which renewable resource constraints are added to the problem. By predefining the resource price, the renewable resources are related to the project costs, including direct cost and indirect cost. Every activity can be executed in the crashing way in which the project direct costs are used to shorten the activity duration. According to the characteristics of the MRC-DTCTP, a genetic algorithm for solving it is developed, and its effectiveness is verified by a comparison with an exact algorithm.

Hadjiconstantinou and Klerides (2009) study a path-based approach to the mathematical formulations of the DTCTP with all the three objectives mentioned earlier in Section 4.1. The formulations are subsequently solved using a new exact cutting plane algorithm. The fundamental idea behind the proposed approach is the derivation of global optimality and feasibility cuts via the critical path of activities, defined as the path of longest duration in the project. This approach can also be seen as a delayed constraint generation methodology, where a selection of path constraints are added gradually, hoping that only a small fraction of such constraints is needed to prove optimality. This classical approach is enhanced by several speed-up techniques, proposed in the paper. Extensive computational results reported for almost 5000 benchmark test problems demonstrate the effectiveness of the proposed algorithms in solving to optimality for the first time some of the hardest and largest instances in the literature within reasonable computational time.

Hazir et al. (2010) investigate the budget variant of the DTCTP. This multi-mode project scheduling problem requires assigning modes to the activities of a project so that the total completion time is minimized and the budget and the precedence constraints are satisfied. This problem is often encountered in practice, as timely completion of the projects without exceeding the budget is crucial. The main contribution of the paper is describing an effective Benders decomposition-based exact branch-and-cut (B&C) algorithm to solve the DTCTP instances of realistic sizes. Although Benders decomposition often exhibits a very slow convergence, the authors include several algorithmic features to enhance the performance of the proposed tailored approach. Computational results confirm the efficiency of the proposed algorithm, which can solve large-scale instances to optimality.

It is worth mentioning that in the deadline version of the DTCTP the objective of the cost minimization can be replaced by the maximization of the NPV. In such a case, positive cash flows are associated with project events or time instants, while costs are associated with activities by means of their discrete time/cost profiles. The DTCTP with the objective to maximize the NPV has only been considered in very few papers.

Erenguc et al. (1993) were the first to consider the DTCTP with discounted cash flows throughout the life of the project, and where shorter activity durations can be attained by incurring greater direct costs. The objective of this problem is to determine the activity durations and a schedule of activity start times so that the NPV of all cash flows is maximized. The problem is formulated as a mixed integer nonlinear program which is amenable to solution using the generalized Benders decomposition technique. The algorithm is tested on 140 project scheduling problems, the largest of which contains 30 nodes and 64 activities. The computational results are quite encouraging since 123 of the 140 problems require less than 1 CPU second of solution time.

Icmeli and Erenguc (1996) introduced a new model of the resource-constrained DTCTP with discounted cash flows, as a combination of the RCPSPDCF and the DTCTP problems. In the presented problem the activity durations can be reduced from their normal durations by allocating more resources, and crashing costs are incurred if the duration of an activity is shorter than normal. The considered problem involves determining the timing and durations of activities such that the NPV of all cash flows is maximized in the presence of precedence and resource constraints. They propose a heuristic procedure with embedded three priority rules, and compare the obtained results with tight upper bounds obtained using the Lagrangian relaxation.

Vanhoucke and Debels (2007) elaborate on three extensions of the DTCTP in order to cope with more realistic settings: time-switch constraints, work continuity constraints, and the net present value maximization. They give an extensive literature overview of existing procedures for these problem types, and discuss a new metaheuristic approach in order to provide near-optimal heuristic solutions for the considered problems. They present computational results for the problems under study by comparing the results for both exact and heuristic procedures. The authors demonstrate that the heuristic algorithms produce consistently good results for two versions of the DTCTP.

More recently, the discrete time, cost, and quality trade-off problem has also been considered. Tareghian and Taheri (2006) develop a solution procedure to study the trade-offs among time, cost, and quality in the management of a project. This problem assumes the duration and quality of project activities to be discrete, nonincreasing functions of a single nonrenewable resource. The objective is to minimize the total cost of the project while maximizing the quality of the project and also meeting a given deadline. Three inter-related integer programming models are developed such that each model optimizes one of the given entities by assigning desired bounds on the other two. Various forms of quality aggregations and effect of activity mode reductions are also investigated. The computational performance of the models is presented using a numerical example.

The same authors in (Tareghian and Taheri, 2007) propose a metaheuristic solution procedure for the discrete time, cost, and quality trade-off problem. They apply an electromagnetic scatter search to solve this problem. In the implementation an initial population of feasible solutions is generated using frequency memory to well sample the feasible region. A number of these solutions are then selected and improved locally. The improved solutions are then combined to generate new set of solutions. The combination process utilizes attraction–repulsion mechanisms borrowed from the electromagnetism theory. The whole process is stopped when no significant improvement in the set of solutions are observed. The validity of the proposed solution procedure is demonstrated, and its applicability is tested on a randomly generated large and complex problem having 19900 activities.

### 4.2. Time/resource trade-off problem

Another well-known problem in which activities can be executed in multiple (possibly infinite) modes is the *time/resource trade-off problem* (TRTP). As mentioned in Section 3.2.1, in the case of the time/resource trade-off, the processing time of an activity is a nonincreasing function of the usage of renewable resources allotted to this activity. In the TRTP a single renewable resource is considered, and the duration of an activity is a function of the resource usage. Normally, the shorter duration of an activity, the larger use of the renewable resource and vice versa – performing an activity in a longer duration reduces the resource requirement. The basic problem is to find activity durations and a schedule that minimize the project duration. However, also in this case it is possible to reverse the problem formulation. The dual problem is the minimization of the resource availability subject to a given project deadline.

#### 4.2.1. Discrete time/resource trade-off problem

Based on the discrete processing time vs. resource amount model, the *discrete time/resource trade-off problem* (DTRTP) was introduced in (De Reyck et al., 1998). In this problem the duration of an activity is a discrete nonincreasing function of the amount of a single renewable resource allotted to this activity. Each activity has a specified work content and can be performed in different modes, as long as the required work content is met. A set of allowable processing modes can then be specified for each activity, each characterized by a fixed duration and an associated constant resource requirement such that the product of them should be at least equal to the activity specified work content. Note that only one resource is considered, and hence, no resource/resource trade-offs between multiple resources are included.

Up to now, only a few papers have dealt with the DTRTP. In (De Reyck et al., 1998) a tabu search approach to the problem is proposed. Next, the same authors in (Demeulemeester et al., 2000) develop a B&B procedure for the problem. Both the solution approaches, as well as an extensive description of the DTRTP itself, can be found in (Demeulemeester and Herroelen, 2002). Since then the DTRTP has been considered in three papers.

Ranjbar and Kianfar (2007) develop a genetic algorithm for solving the DTRTP. In the proposed metaheuristic a new method based on the resource utilization ratio is developed for generation of crossover points, and also a local search method is incorporated with the algorithm. Comparative computational results reveal that this procedure outperforms the tabu search approach by De Reyck et al. (1998).

Vanhoucke and Debels (2008) investigate the effect of three activity assumptions on the total lead time and the total resource utilization of a project in the RCPSP. They analyze the influence of variable activity durations under a fixed work content (the DTRTP), the possibility of allowing activity preemption (the resulting problem is called the preemptive DTRTP denoted as PDTRTP), and the use of fast tracking to decrease a project duration (the problem denoted as PDTRTP-FT). The authors give an overview of the procedures developed in the literature, and present some modifications to an existing solution approach to cope with the activity assumptions under study. They present computational results on a generated dataset and evaluate the impact of all assumptions on the quality of the schedule.

Ranjbar et al. (2009) introduce the DTRTP with multiple resources. The resulting problem is denoted as MDTRTP. In the MDTRTP there are multiple renewable resources, each with time/resource trade-offs. In fact, the MDTRTP becomes a special case of the MRCPSP (see Section 5) with the absence of nonrenewable resources. The authors present a new hybrid metaheuristic algorithm based on scatter search and path relinking methods. In the SS algorithm for the MDTRTP, they use path relinking concepts to generate children from parent solutions, in the form of a new combination method. They also incorporate new strategies for diversification and intensification to enhance the search, in the form of local search and forward–backward scheduling, based on so-called reverse schedules, with the activity dependencies reversed. The proposed algorithm is also modified to tackle the RCPSP and MRCPSP. The performance of the algorithm is tested on four datasets, which show that in most cases it outperforms the other heuristic approaches presented in the literature.

It can be also mentioned that in the context of constrained-based scheduling, the discrete time/resource trade-off model is considered as *elastic activity* model. Fully and partially elastic activity models were introduced in (Baptiste et al., 1999), and extensively discussed in (Baptiste et al., 2001). Elastic activities (jobs) were then considered in (Pedersen et al., 2007) in the context of a practical large-scale scheduling problem.

#### 4.2.2. Continuous time/resource trade-off problem

The *continuous time/resource trade-off problem* (CTRTP), as a straightforward generalization of the DTRTP, has never been considered explicitly. In the CTRTP the duration of an activity is described by a continuous nonincreasing function of the amount of the single renewable, continuous resource allotted to this activity.

However, a variety of machine scheduling problems using the continuous processing time vs. resource amount model were broadly studied in a number of papers. A comprehensive survey on these papers is given in (Błażewicz et al., 2007).

## 5. Multi-mode resource-constrained project scheduling problem

The *multi-mode resource-constrained project scheduling problem* (MRCPSP) is a generalization of the RCPSP where each activity can be performed in one of several modes. Three basic categories of resources: renewable, nonrenewable, and doubly constrained ones are considered in this problem. All resources are discrete. As for the RCPSP, the planning horizon is divided into time periods which means that time is a discrete variable as well.

As mentioned in Section 3, doubly constrained resources need not be considered explicitly in the MRCPSP, because such a resource can be replaced (according to the transformation proposed by Talbot (1982)) by a pair of one renewable and one nonrenewable resources.

A finite set of modes $M_j$ is defined for each nondummy activity $A_j$. Let us recall that a mode is a pair whose first element is a vector of resource requirements and the second element is the activity processing time. Since renewable as well as nonrenewable resources are considered, time/cost, time/resource, and resource/resource trade-offs may occur. In order to explain the meaning of these trade-offs in the context of the MRCPSP, let us assume that activity $A_j$ is executed in mode $m \in M_j$ with the longest duration. Then time/cost and time/resource trade-offs mean that there exists another mode in set $M_j$ with a shorter duration and greater nonrenewable or renewable resource requirements, respectively. Of course, a combination of these two trade-offs is also possible. Finally, the resource/resource trade-off means that at least two modes exist in set $M_j$ with the same durations and different renewable and/or nonrenewable resource requirements. Thus, in this case a so-called resource substitution occurs.

The mathematical model of the MRCPSP introduced by Talbot (1982) and based on the Pritsker's model of the RCPSP (presented in Section 2) is formulated as follows:

$$\text{Minimize} \quad \sum_{t=EF_{n+1}}^{LF_{n+1}} t x_{n+1,m,t} \tag{6}$$

$$\text{subject to} \quad \sum_{m=1}^{|M_j|} \sum_{t=EF_j}^{LF_j} x_{jmt} = 1 \quad \text{for } j = 0, \ldots, n+1 \tag{7}$$

$$\sum_{m=1}^{|M_i|} \sum_{t=EF_i}^{LF_i} t x_{imt} \le \sum_{m=1}^{|M_j|} \sum_{t=EF_j}^{LF_j} t x_{jmt} - d_{jm} \quad \text{for all } (A_i, A_j) \in P \tag{8}$$

$$\sum_{j=1}^{n} \sum_{m=1}^{|M_j|} \sum_{q=max\{t,EF_j\}}^{min\{t+d_{jm}-1,LF_j\}} r^{\rho}_{jmk} x_{jmq} \leqslant R^{\rho}_k \quad \text{for } k = 1, \ldots, R; \; t = 1, \ldots, H \tag{9}$$

$$\sum_{j=1}^{n} \sum_{m=1}^{|M_j|} \sum_{t=EF_j}^{LF_j} r^{\nu}_{jml} x_{jmt} \leqslant R^{\nu}_l \quad \text{for } l = 1, \ldots, N \tag{10}$$

$$x_{jmt} \in \{0,1\} \quad \text{for } i = 0, \ldots, n+1; \; m \in M_j; \; t = EF_j, \ldots, LF_j \tag{11}$$

where:

[–] binary variable $x_{jmt} = 1$ if activity $A_j$ executed in mode $m \in M_j$ is completed at the end of time period $t$
[–] $R^{\rho}_k (R^{\nu}_k)$ is the number of available units of the $k$th ($l$th) renewable (nonrenewable) resource
[–] $r^{\rho}_{jmk}$ is the number of units of the $k$th renewable resource ($k = 1, \ldots, R$) required by activity $A_j$ executed in mode $m \in M_j$
[–] $r^{\nu}_{jml}$ is the number units of the $l$th nonrenewable resource ($l = 1, \ldots, N$) required by activity $A_j$ executed in mode $m \in M_j$
[–] $d_{jm}$ is the duration of activity $A_j$ executed in mode $m \in M_j$
[–] $EF_j, LF_j$ are calculated assuming that the shortest duration mode is assigned to each activity, and the planning horizon $H$ is calculated for the modes with the longest durations.

Constraints (7) ensure that each nonpreemptable activity is performed exactly once in exactly one mode. Precedence constraints are guaranteed by (8). Constraints (9) and (10) ensure that the renewable and nonrenewable resource limits are not exceeded, respectively. Finally, constraints (11) define the binary status of the decision variables.

The MRCPSP is NP-hard in the strong sense being a generalization of the RCPSP. Moreover, for more than one nonrenewable resource the problem of finding a feasible solution is already NP-complete (Kolisch, 1995).

Alternative modes of executing activities of a project were first considered in time/cost trade-off problems (see Section 4.1). The discrete version of the problem (DTCTP) is, in fact, a subproblem of the MRCPSP. As mentioned in Section 4.2.1, activity modes with different renewable resource requirements and different durations were considered for the first time by Elmaghraby (1977) for a project scheduling problem where the total consumption of a single nonrenewable resource (i.e. cost) was minimized. The first attempts to model and solve project scheduling problems where activity modes are possible combinations of renewable, nonrenewable, and doubly constrained resource requirements are presented by Węglarz (1980a) and Słowiński (1980) for the preemptive case (see Section 8). Multi-objective versions of these problems are considered by Słowiński (1981). For nonpreemptable activities executed in several modes using renewable, nonrenewable, and doubly constrained resources, the first approach was presented by Talbot (1982).

A new mathematical formulation of the MRCPSP, based on the *feasible subsets* concept, is presented by Maniezzo and Mingozzi (1999). A feasible subset of the set $F_\lambda$ of all activity-mode combinations is a set of pairs $(A_j, m)$, where $m \in M_j$ denotes the processing mode assigned to activity $A_j$ that satisfy the following conditions:

(i) each activity $A_j, j = 1, 2, \ldots, n$, occurs in at most one pair $(A_j, m)$
(ii) $\sum_{(A_j,m) \in F_\lambda} r^{\rho}_{jmk} \leqslant R^{\rho}_k$ for $k = 1, 2, \ldots, R$
(iii) $\sum_{(A_j,m) \in F_\lambda} r^{\nu}_{jml} \leqslant R^{\nu}_l$ for $l = 1, 2, \ldots, N$
(iv) there are no precedence constraints between any activities from subset $F_\lambda$.

Three types of binary variables are used in this formulation: $x_{jt} = 1$ if and only if activity $A_j$ is completed at time $t$; $y_{\lambda t} = 1$ if and only if all activities of feasible subset $F_\lambda$ are executed in time period $t$; $\zeta_{jm} = 1$ if and only if activity $A_j$ is performed in mode $m \in M_j$. The corresponding mathematical model of the MRCPSP is formulated as follows:

$$\text{Minimize} \quad \sum_{t=EF_{n+1}}^{LF_{n+1}} t x_{n+1,t} \tag{12}$$

$$\text{subject to} \quad \sum_{\lambda \in \mathscr{F}_{jm}} \sum_{t=ES_j}^{LF_j} y_{\lambda t} = d_{jm} \zeta_{jm} \quad \text{for } j = 0, \ldots, n+1; \; m \in M_j \tag{13}$$

$$\sum_{\lambda=1}^{\Lambda} y_{\lambda t} \leqslant 1 \quad \text{for } t = 1, \ldots, H \tag{14}$$

$$\sum_{m=1}^{|M_j|} \zeta_{jm} = 1 \quad \text{for } j = 0, \ldots, n+1 \tag{15}$$

$$x_{jt} \geqslant \sum_{\lambda \in \mathscr{F}_j} y_{\lambda t} - \sum_{\lambda \in \mathscr{F}_j} y_{\lambda, t+1} \quad \text{for } j = 1, \ldots, n; \; t = EF_j, \ldots, LF_j \tag{16}$$

$$\sum_{t=EF_j}^{LF_j} t x_{jt} - \sum_{t=ES_i}^{LF_i} t x_{it} \geqslant \sum_{m=1}^{|M_j|} d_{jm} \zeta_{jm} \quad \text{for all } (A_i, A_j) \in P \tag{17}$$

$$\sum_{j=0}^{n+1} \sum_{m=1}^{|M_j|} r^{\nu}_{jml} \zeta_{jm} \leqslant R^{\nu}_l \quad \text{for } l = 1, \ldots, N \tag{18}$$

$$x_{jt} \in \{0,1\} \quad \text{for } j = 0, \ldots, n+1; \; t = 1, \ldots, H \tag{19}$$

$$y_{\lambda t} \in \{0,1\} \quad \text{for } \lambda = 1, \ldots, \Lambda; \; t = 1, \ldots, H \tag{20}$$

$$\zeta_{im} \in \{0,1\} \quad \text{for } j = 0, \ldots, n+1; \; m \in M_j \tag{21}$$

where: $\Lambda$ is the number of feasible subsets, $\mathscr{F}_{jm} = \{\lambda : (A_j, m) \in F_\lambda\}$, and $\mathscr{F}_j = \bigcup_{m=1}^{|M_j|} \mathscr{F}_{jm}$.

Constraints (13) guarantee that feasible subsets containing activity-resource combination for activity $A_j$ executed in mode $m \in M_j$ are in progress for exactly $d_{jm}$ time periods. Constraints (14) ensure that in each time period $t$ at most one feasible subset is in progress, whereas constraints (15) ensure that each activity is performed in exactly one mode. Constraints (16) guarantee that the processing of activity $A_j$ terminates at time period $t$ if this activity belongs to a feasible subset being in progress at time period $t$ and does not belong to a feasible subset being in progress at time period $t+1$. Precedence constraints are guaranteed by (17). Constraints (18) ensure that the nonrenewable resource limitations are not exceeded. Finally, constraints (19)–(21) define the binary status of the decision variables. Notice, that the renewable resource limitations do not occur explicitly in this formulation, because feasibility of a solution with respect to renewable resources is guaranteed by definition by the feasible subsets.

Recently, two papers concerning new mathematical formulations of the extensions of the MRCPSP have been published. Sabzehparvar and Seyed-Hosseini (2008) propose a mathematical formulation for the MRCPSP with generalized precedence constraints and mode-dependent time lags. Zapata et al. (2008) propose three alternative formulations of the multi-project version of the MRCPSP, in two of them time is a continuous variable. Obviously, under some assumptions these models may be used to formulate mathematically the MRCPSP.

The MRCPSP has been broadly studied in the recent three decades. Several exact and heuristic approaches, as well as some extensions and special cases, have been developed.

### 5.1. Exact approaches

Exact approaches are proposed by Talbot (1982), Patterson et al. (1989), Speranza and Vercellis (1993), Sprecher (1994), Sprecher et al. (1997), Hartmann and Drexl (1998), Sprecher and Drexl (1998), and Zhu et al. (2006). All approaches, except the one proposed by Zhu et al. (2006), are based on the B&B method and on the idea to enumerate partial schedules. A detailed comparison of these methods is provided by Hartmann and Drexl (1998).

Talbot (1982) proposes a two-stage algorithm. In stage one, activities, resources, and modes are sorted using some selected rules. In stage two, a priority rule heuristic is used to calculate an upper bound and then a B&B with backtracking is used. At each level $j$ of the search tree the first unscheduled activity $A_j$ from the sorted list is assigned its first mode, and it is added to the partial schedule with the earliest precedence- and resource-feasible completion time $C_j$. If the assignment of this mode to activity $A_j$ is prevented because of either type of infeasibility, the next mode is considered for assignment. If none of the modes of activity $A_j$ is resource-feasible, then the algorithm backtracks to the previous level $j-1$, and an attempt is made to reassign activity $A_{j-1}$ previously scheduled at this level to the earliest feasible completion time greater than the previously assigned completion time $C_{j-1}$. If it is impossible, the next mode is selected for activity $A_{j-1}$. This process continues until an attempt is made to backtrack below activity $A_0$, or until activity $A_{n+1}$ is scheduled. In the former case the best solution found so far is the optimal one. In the latter case a new improved solution with makespan $T^*$ is found. If $T^*$ is equal to the known solution lower bound then the obtained schedule is optimal. Otherwise, a new upper bound is set to $T^*$ and the process starts anew with activity $A_0$ executed in its first mode.

Patterson et al. (1989) also use a depth-first B&B with backtracking. The solution is represented by two resource- and precedence-feasible lists: a list of selected modes and a list of activity completion times. The solution method consists of two phases: a problem initialization phase, and an enumeration phase. The initialization phase is similar to the stage one of the Talbot's algorithm (1982). Activities and their modes are sorted using some rules in order to heuristically generate a good initial schedule. Based on this schedule, an upper bound is calculated on completion time of each activity. In the enumeration phase the *precedence tree* is used to guide the search in the set of all precedence-feasible sequences of activities. At each level of the search tree only one activity from the set of eligible activities is chosen together with an assigned mode. An eligible activity is defined as an activity, the predecessors of which are already scheduled. Next, the precedence- and resource-feasible start time of this activity is calculated that is not less than the start time assigned at the previous level of the search tree. Then another activity is chosen at the next level of the search tree. If we obtain the sink dummy activity it means that a complete schedule is found, and the upper bound on completion time of each activity can be computed, and the backtracking to the previous level occurs. At the previous level the next untested mode for this activity is chosen. If all modes are already tested than the next eligible activity is chosen. If all activities from the set of eligible activities at the given level of the tree are already tested, the backtracking to previous level occurs once more. Optimality is guaranteed when a solution equal to a known bound is found, or when backtracking proceeds to the dummy activity $A_0$. The bounding rules include the consideration of continuously updated upper bounds on the completion time of each activity, as well as procedures ensuring precedence and resource feasibility. The performance of this approach was examined for 91 problems with the number of activities equal to 10, 20, 30, 50, 100, and 500 (Patterson et al., 1990).

Speranza and Vercellis (1993) propose a depth-first B&B procedure which enumerates the set of tight schedules. This algorithm is used for a single project being a part of the multi-project model. It is showed that the set of tight schedules dominates the set of schedules, and therefore an enumeration procedure can be reduced to the set of tight schedules only. The presented algorithm uses so-called maximal extensions of partial schedules in order to generate tight schedules. However, Hartmann and Sprecher (1996) showed that the method might fail to find optimal solution if at least two renewable resources are used during the execution of the project. Moreover, for some problems with limited availability of nonrenewable resources the algorithm does not even find an existing feasible solution.

Sprecher (1994) also uses the precedence tree as an enumeration scheme. Moreover, the author proposes the following bounding rules: (i) Basic Time Window Rule – if an activity should be scheduled but the assigned completion time is greater than the current latest finish time, then the current partial schedule does not lead to a solution better than the best currently known; (ii) Nondelayability Rule – if an eligible activity cannot be feasibly scheduled in any mode in the current partial schedule without exceeding its latest completion time, then no other eligible activity needs to be examined at this level; (iii) Nonrenewable Resource Rule – if scheduling each currently unscheduled activity in the mode with the lowest requirement for a nonrenewable resource would exceed the capacity of this nonrenewable resource, then the current partial schedule cannot be feasibly completed; (iv) Local Left Shift Rule – if an activity that has been started at the current level of the B&B tree can be locally left shifted without changing its mode, then the current partial schedule needs not be completed; (v) Single Enumeration Rule which is further improved by Hartmann and Drexl (1998) and called Precedence Tree Rule. Consider two activities $A_i$ and $A_j$ scheduled at the previous and at the current level of the B&B tree, respectively. If $S_i = S_j$ and $i > j$, than the current partial schedule needs not be completed.

Sprecher et al. (1997) propose a B&B algorithm in which enumeration scheme named *mode and delay alternatives* is used, which is a multi-mode extension of the delay alternative concept proposed by Christofides et al. (1987) and used by Demeulemeester and Herroelen (1992) for the RCPSP. In the mode and delay alternatives each node $g$ of the search tree is associated with a fixed time $t_g$ at which activities may start processing. An activity $A_j$ becomes eligible at time $t_g$ when all its predecessors have completion times smaller than $t_g$, and it is in progress at time $t_g$ if $C_j - d_{jm} \leqslant t_g < C_j$, assuming that activity $A_j$ is processed in mode $m \in M_j$. At each level $g$ of the search tree firstly a new decision point $t_g$ is calculated as the earliest completion time of activities currently in progress. Then a set of eligible activities is computed. All eligible activities with modes assigned at the previous level are temporally started at time $t_g$. For the set of all new eligible activities (i.e. those becoming eligible at time $t_g$) a set of mode alternatives is calculated, where a mode alternative represents one of the possible choices of modes for eligible activities. Selecting one mode alternative the new eligible activities are temporally started at time $t_g$ as well. Now, when all eligible activities are added to the set of activities in progress, some resource conflicts may occur. In order to resolve these conflicts, a set of minimal delay alternatives is computed, where a delay alternative is the set of activities the removal of which from the set of activities in progress at time $t_g$ makes all resource constraints satisfied. A delay alternative is minimal if no proper subset of the delay alternative is also a delay alternative. One minimal delay alternative is chosen, and the activities from this set are delayed in the partial schedule considered at this level of the search tree. Next, another decision point is calculated for the next level of the search tree. If the complete schedule is obtained, a backtracking to the previous level occurs and the next minimal delay alternative or the next mode alternative is tested. The main differences between this approach and the precedence tree are that at each level more than one activity can be scheduled, and that at the current level one may withdraw decisions made at the previous level.

Another enumeration scheme, named *mode and extension alternatives*, is proposed in the same paper (Sprecher et al., 1997). It is very similar to the mode and delay alternatives, and it is almost the same except that extension alternatives are employed instead of the delay alternatives. An extension alternative is a subset of activities in progress for which all resource constraints are satisfied. An extension alternative is a subset of the activities in progress which are delayed but are scheduled at time $t_g$. The bounding rules introduced or redefined in this implementation of B&B include: (i) Non-delayability Rule modified for the presented enumerating schemes is defined in the following way – if an eligible activity, the mode of which has not yet been fixed, cannot be started in the mode with the shortest duration at the current level of the search tree without exceeding its latest completion time, then no mode alternative needs to be examined at the current level; (ii) Data Reduction (so-called *preprocessing*) which looks as follows – the project data can be adapted by executing the following steps: (a) removing all nonexecutable modes (modes which always violate the resource constraints), (b) deleting the redundant nonrenewable resources (resources with availabilities greater than the maximal total resource requirements), and (c) eliminating all inefficient modes (modes with duration not shorter and resource requirements not less than those of another mode for the same activity). Steps b and c are executed iteratively until further elimination is no more possible. This preprocessing rule is also used in some metaheuristic approaches described in Section 5.3.2; (iii) Multi mode Rule defined in the following way – assume that no currently unscheduled activity will be started before the completion time of a scheduled activity $A_j$ when the current partial schedule is completed, if a multi mode left shift or a mode reduction of activity $A_j$ with resulting mode $m' \in M_j,\ 1 \leqslant m' \leqslant |M_j|$, can be performed on the current partial schedule and, moreover, $r'_{jm}l^{v} \leqslant r^{v}_{jml}$ if holds for each nonrenewable resource $l = 1, \dots, N$, then the current partial schedule needs not be completed; (iv) and finally Immediate Selection that looks as follows – the following situation is assumed: all activities that start before the current decision point $t_g$ complete at or before $t_g$. After selecting a mode alternative, there is an eligible activity $A_j$ with assigned mode $m \in M_j$ which cannot be simultaneously processed with any other activity $A_i$ in its assigned mode $m' \in M_i$. Moreover, activity $A_j$ cannot be simultaneously processed with any unscheduled activity $A_h$ in any mode $m'' \in M_h$. Then the delay alternative obtained from the set of activities in progress at time $t_g$ after removing activity $A_j$ from this set is the only minimal delay alternative that has to be examined, and the extension set containing only activity $A_j$ is the only extension alternative that has to be examined.

Sprecher and Drexl (1998) use in their B&B algorithm the precedence tree enumeration scheme and many bounding rules proposed by Sprecher (1994). The newly introduced bounding rule is the Cutset Rule which is defined as follows. Let $\overline{PS}$ denote a previously evaluated partial schedule with cutset $CS(\overline{PS})$, maximal completion time $C_{\max}(\overline{PS})$, and leftover capacities $R^{v}_{l}(\overline{PS})$ of the nonrenewable resources $l = 1, \dots, N$. Let $PS$ be the current partial schedule considered to be extended by scheduling some activity $A_j$ with start time $S_j$. If we have $CS(PS) = CS(\overline{PS}),\ S_j \geqslant C_{\max}(\overline{PS})$, and $R^{v}_{l}(PS) \leqslant R^{v}_{l}(\overline{PS})$ for all $l = 1, \dots, N$, then $PS$ needs not be completed, where a cutset of a partial schedule $PS$ is defined as the set of activities scheduled in $PS$.

Hartmann and Drexl (1998) extend the approach proposed by Sprecher and Drexl (1998) by two new bounding rules, namely the Order Swap Rule and the Immediate Selection Rule for Precedence Tree. The Order Swap Rule is defined as follows. Consider a scheduled activity the finish time of which is less than or equal to any start time that may be assigned when completing the current partial schedule. If an order swap on this activity together with any of those activities that finish at its start time can be performed, then the current partial schedule needs not be completed. The Immediate Selection Rule for Precedence Tree looks as follows. Consider an eligible activity $A_j$ no mode of which is simultaneously performable with any currently unscheduled activity in any mode. If the earliest feasible start time of each other eligible activity in any mode is equal to the maximal completion time of the currently scheduled activities, then $A_j$ is the only eligible activity that needs to be selected for being scheduled at the current level of the B&B tree.

Another exact approach was proposed by Zhu et al. (2006). Although it is primarily developed for the multi-mode version of the RCPSP with partially renewable resources, it can be successfully used for the classical MRCPSP because both renewable and nonrenewable resources can be easily modeled using partially renewable resources. Instead of using the B&B method, which explicitly enumerates all (partial) schedules, they use B&C approach. In this approach the linear programming relaxation of the integer linear programming model is used to obtain a lower bound of the project duration at each node of the search tree. If the node of the search tree cannot be fathomed and has fractional value of the objective function, then the algorithm tries to find cuts, i.e. valid inequalities that are violated by the fractional solution but are satisfied by all feasible integer solutions represented by this node in the search tree. If no cut is found for a given node, then the branching is performed to create new nodes in the search tree. The rules used in the considered B&C algorithm can be divided into four categories: reduction of the number of variables, branching and bound tightening, cuts, and high-level search strategy. The number of variables is reduced using: (i) time windows derived from the improved distance matrix, (ii) lower bound based on the usage of renewable resources, (iii) lower bound obtained by a truncated version of B&C, (iv) upper bound obtained by the authors' implementation of a genetic algorithm. The considered B&C algorithm uses the cut-generating features built into the mixed integer programming (MIP) solver that is a part of CPLEX (ILOG, 2002), as well as two other problem-specific cuts: cuts from resource conflicts and cuts from precedence relations. In order to make the branching more effective, a procedure operating on the so-called special ordered set (SOS) is used for bound tightening. Another bound tightening procedure, which takes into account the current value of the makespan, is proposed for the problem with makespan minimization criterion. Finally, local branching (Fischetti and Lodi, 2003) procedure is used as high-level search strategy. The performance of the proposed B&C algorithm is examined on the basis of computational experiments where datasets containing instances of the MRCPSP with 20 and 30 activities from PSPLIB – a library of data for project scheduling (Kolisch and Sprecher, 1997) – are used. Optimal solutions are found for all instances with 20 activities and for 506 out of 552 instances with 30 activities. Moreover, for 5 instances with 30 activities the considered B&C algorithm found solutions with the makespan better than the best one known, and for 23 instances it found solutions worse than those reported in PSPLIB. Unfortunately, these results were not recorded in the PSPLIB files containing the best known solutions. The computational times of the B&C reported by the authors are smaller than those reported by Sprecher and Drexl (1998) for their B&B, however, taking into account the processor clocks, the B&B performs faster than the B&C.

A comparison of the presented exact approaches, except the last one by Zhu et al. (2006), is reported by Hartmann and Drexl (1998). The obtained theoretical and experimental results show that for both precedence tree and mode and delay alternatives enumerating schemes there exist instances for which the sets of schedules generated by these two procedures may differ between themselves, but the mode and extension alternatives enumerating scheme is able to generate the same schedules as the two other approaches. Moreover, the algorithm by Sprecher and Drexl (1998) outperforms the other approaches. And finally, finding optimal solutions for a number of activities greater than 20 is computationally intractable by any of the analyzed algorithms. The last conclusion is still true. The newest exact approach – B&C by Zhu et al. (2006) – was tested on instances from PSPLIB. The algorithm found optimal solutions for all instances with 20 activities, but only 506 out of 552 instances with 30 activities were solved optimally (notice that 307 of those instances can be solved optimally by CPM assuming the shortest duration mode for each activity). Thus, there still exist instances with 30 activities for which optimal solutions have not been found.

### 5.2. Lower bounds

The first lower bound for the MRCPSP was proposed by Talbot (1982). It was calculated as the length of the critical path for the project where shortest duration mode is assigned to each activity, and resource constraints are neglected.

Maniezzo and Mingozzi (1999) propose four other rules for computing the lower bounds. The first one is computed as a solution of a weighted node packing problem on graph $\widetilde{G}$, which consists in finding an independent set of $\widetilde{G}$ of maximum weight. $\widetilde{G} = (V', E)$ is an intersection graph, where $V'$ is a set of nondummy activities of the project, and $(A_i, A_j) \in E$ if and only if the precedence network does not contain any path from vertex $A_i$ to vertex $A_j$ (or from $A_j$ to $A_i$), and for each renewable resource the sum of minimal renewable resource requirements for both activities $A_i$ and $A_j$ is smaller than the availability of this resource. The second lower bound, giving results not worse than the previous one, is based on the mathematical formulation (12)–(21) of the MRCPSP, and is calculated as a solution of the LP-relaxation of the problem where precedence and nonpreemptibility constraints (14), (16), (17) are removed. The resulting LP problem solved to find the lower bound looks as follows: (i) the objective is to minimize $\sum_{\lambda=1}^{\Lambda} h_\lambda$, where $h_\lambda = \sum_{t=1}^{H} y_{\lambda t}$ is the total processing time of feasible subset $F_\lambda$; (ii) constraints (15), (18), and (21) are the same as in formulation (12)–(21); (iii) constraints (13) are changed to $\sum_{\lambda \in \mathscr{F}_{jm}} \sum_{t=ES_j}^{LF_j} y_{\lambda t} = d_{jm}\zeta_{jm}$ for $j = 0, \ldots, n+1; m = 1, \ldots, |M_j|$, and constraints $h_\lambda \geqslant 0$ for $\lambda = 1, \ldots, \Lambda$ are added; (iv) all other constraints, i.e. (14), (16), (17), (19), and (20), are removed from the model.

The third and the fourth lower bounds, which give results not worse than the first one and no better than the second one, are based on the same mathematical formulation of the LP-relaxed problem where for the third lower bound equations are replaced by inequalities, and for the fourth lower bound additionally nonrenewable resource constraints are removed. The formulation of the third lower bound looks as follows: (i) the objective is to minimize $\sum_{\lambda=1}^{|MF|} h_\lambda$, where $MF = \{\lambda : \lambda \in \{1, \ldots, \Lambda\}, F_{\lambda'} \not\subseteq F_\lambda, \lambda' \in \{1, \ldots, \Lambda\} \setminus \{\lambda\}\}$ is the set of maximal feasible subsets; (ii) constraints (15), (18), and (21) are the same as in formulation (12)–(21); (iii) constraints (13) are changed to $\sum_{\lambda \in \mathscr{M}_{jm}} h_\lambda \geqslant d_{jm}\zeta_{jm}$ for $j = 0, \ldots, n+1; m = 1, \ldots, |M_j|$, where $\mathscr{M}_{jm}$ is the set of indices of all maximal feasible subsets containing activity $A_j$ performed in mode $m$, and constraints $h_\lambda \geqslant 0$ for $\lambda = 1, \ldots, |MF|$ are added; (iv) all other constraints, i.e. (14), (16), (17), (19), and (20), are removed from the model. The formulation of the fourth lower bound is the same except for constraints (18) which are also removed.

Pesch (1999) uses the Talbot's B&B (1982) as a base algorithm to test several various strategies in order to analyze the influence of different settings of the problem parameters on the quality of the enumeration approaches by defining different classes of problem instances. The main conclusion from this paper is that probably no procedure exists that is able to efficiently solve problem instances of different classes. Therefore different implementations are necessary, and they are to be applied to different classes of problem instances.

A destructive lower bound for the MRCPSP with minimal and maximal time lags is presented in (Brucker and Knust, 2003). The lower bound calculations are based on two methods for proving infeasibility of a given threshold value $T$ for the makespan. The first one uses constraint propagation techniques, while the second one is based on the linear programming formulation by Maniezzo and Mingozzi (1999) which is solved by a column generation procedure. Computational results are reported for several test instances of the multi-mode problem with and without time lags and the single-mode version with time lags.

### 5.3. Heuristics

In this section we describe heuristic approaches developed to find semi-optimal solutions to the MRCPSP. We divide this section into three parts, similarly to the classification made by Kolisch and Hartmann (1999). First, we present some priority rule-based algorithms. The next part is devoted to metaheuristics and other heuristics based on local search procedures. Finally, some other approaches are discussed.

#### 5.3.1. Priority rule-based heuristics

Priority rules are to determine the order of activities which is used by a so-called schedule generation mechanisms to generate a schedule (e.g. serial or parallel schedule generation schemes – see, e.g. Kolisch, 1996; Kelley, 1963; Bedworth, 1973). A schedule is generated either once for an activity order determined by one priority rule (so-called single-pass methods), or several times for different combinations of priority rules and schedule generation mechanisms (so-called multi-pass methods). In the literature, there are several papers concerning such an approach.

In stage one of the Talbot's B&B (Talbot, 1982), a priority rule heuristic is used to reorder the list of activities and to provide a good initial solution that is next used to obtain initial upper bounds for completion times of activities. The eight rules proposed by Talbot are the following: maximum average activity duration (MAX ADUR), maximum activity duration (MAX DUR), minimum late finish time (MIN LFT), maximum average resource demand (MAX RD), minimum early finish time (MIN EFT), minimum late finish time reduced by smallest duration (MIN (L-D)), random (RAND), and minimum late finish time reduced by average duration (MIN (L-ADUR)). A small computational experiment (100 problem instances with 10 activities, 3 renewable resources, and 1–3 modes for each activity) shows that random rule is outperformed by all other heuristics, and moreover the best results are obtained using priority rules based on the minimum late finish time.

Drexl and Grünewald (1993) present a stochastic scheduling method named STOCOM which finds suboptimal solutions of the MRCPSP. The activities from the set of eligible activities, as well as modes assigned to them, are selected randomly with probability either proportional to the weights that are calculated for each activity-mode combination regarding the longest duration mode from all modes of all eligible activities, or proportional to the weights calculated regarding the latest completion times of eligible activities. Computational results show that this method is superior to other deterministic scheduling rules existing when the experiment was performed. Extensions to problems in which resource requirements of activities vary with time, in addition to time-varying supply resource profiles, are discussed as well. Boctor (1996a,b) uses this approach in his computational experiment, but unfortunately it completely failed being unable to find a feasible solution to any of the 240 test instances of the problem. The reason for such a result is that this heuristic may generate feasible or infeasible schedules, scheduling all activities at their earliest start times which are calculated taking into account precedence constraints only. Feasibility with respect to resource constraints is checked solely for the final schedule.

Özdamar and Ulusoy (1994) propose heuristic approach named local constraint based analysis (LCBA) for the MRCPSP with one nonrenewable resource only. In this algorithm the selection of activities and their assigned modes is made locally at every decision point. LCBA imposes precedence relationships on activities which guarantee the feasibility of the current makespan constrained by the available resource levels at that decision point as long as a complete sequence of schedulable activities is found. Parallel schedule generation scheme (SGS), also called Brooks algorithm (see, e.g. Kolisch, 1996; Bedworth, 1973), is used as a decoding rule. The procedure is also adapted to various model extensions such as flexible resource requirement levels. The procedure is tested on a set of 95 instances with 20–57 activities, 1–6 renewable and one nonrenewable resource. The computational experiment results obtained for LCBA and three dispatching rules show that the proposed procedure yields an average increase over the precedence-based lower bound of 59% which is a better result than this obtained by priority rules which were applied for comparative purposes and yielded an average deviation of 65%. Unfortunately, the constraint-based approach has two disadvantages. First, the worst-case time complexity of the procedure is exponential. Secondly, it is not suited for solving the MRCPSP with multiple scarce nonrenewable resources.

### 5.3.2. Metaheuristics and local search heuristics

Kolisch and Drexl (1997) apply a local search strategy where a solution is represented by both a mode assignment list and a list of activity completion times. The proposed method consists of three phases. Firstly, in the construction phase, an initial mode assignment is generated and then, if it satisfies nonrenewable resource constraints, a fast heuristic for the resulting RCPSP is used. In the second phase, a prescribed number of iterations of a local search, that performs a single neighborhood search on the set of feasible mode assignments, is performed. Finally, an intensification phase is performed. In this phase, a schedule with an improved objective function is searched on the basis of the best mode assignment obtained during the previous phase. The performance of this algorithm is checked on the basis of computational experiments where two sets of benchmark instances of the problem with 10 and 30 activities were generated using ProGen (Kolisch et al., 1995). Each set contains 640 instances, where in each instance two renewable and two renewable resources, as well as three modes per activity, are considered. The results obtained by the proposed algorithm are compared with the results obtained by a truncated B&B (Talbot, 1982) and STOCOM (Drexl and Grünewald, 1993). The proposed heuristic is the only one from the three tested approaches that finds feasible solutions for all 10-activity instances and for most 30-activities instances.

Özdamar (1999) proposes two versions of genetic algorithms: pure (PGA) and hybrid (HGA). A solution in the HGA is represented by two lists: a mode assignment list and a list of priority rules. The $i$th position on the second list denotes a priority rule that is used for the $i$th scheduling decision. The following priority rules are used: minimum total slack (MIN SLK), MIN LFT, shortest processing time (SPT), RAND, weighted resource utilization and precedence (WRUP), minimum late start time (MIN LST), minimum early start time (MIN EST), MIN EFT, and most total successors (MTS). As a decoding rule, a forward–backward scheduling (see, e.g. Li and Willis, 1992) employing parallel SGS is used. Both two-point crossover and uniform crossover operators are proposed. A mutation operator randomly changes one position in mode assignment list and one position in the list of priority rules. In the PGA a solution is represented by both a mode assignment list and an activity list. Serial SGS (see, e.g. Kelley, 1963; Kolisch, 1996) is employed as a decoding rule. Two-point linear crossover is used to generate offspring solutions. If an activity list in the obtained offspring is not precedence-feasible, a repair procedure is run. A mutation operator for the PGA randomly changes one position in the mode assignment list and pairwise swaps two activities from the activity list, but precedence infeasible swaps are not permitted. In both the algorithms, if the offspring is not feasible with respect to any nonrenewable resource, then it is not evaluated. In other words, only solutions with nonrenewable resource-feasible assignments of modes are permitted. The performance of the proposed approaches is evaluated using 536 problem instances with 10 activities, 3 modes per activity, two renewable and two nonrenewable resources as well as 32 instances with 90 activities, two modes per activity, two renewable and two nonrenewable resources. All instances are generated by ProGen and available via Internet at PSPLIB. The obtained results are compared with those reported by Kolisch and Drexl (1997) and show that the HGA outperforms all other algorithms tested in that experiment.

Hartmann (2001) proposes a genetic algorithm. Before executing it, preprocessing is applied in order to reduce the search space by adapting the project data. A solution is represented by a precedence-feasible list of activities and a mode assignment list. Such a representation allows to generate schedules infeasible with respect to nonrenewable resource constraints. In such a case, a penalty function is used to calculate fitness of an infeasible solution. The next population is generated using one point crossover (two different cut points are used: one for the activity list and another one for the mode assignment list), and a mutation that swaps two adjacent activities on the activity list (if it is precedence-feasible) and changes randomly a mode on the mode assignment list. The ranking method is used as a selection operator. A schedule is constructed applying the serial SGS, and a single-pass or multi-pass local search based on the multi-mode left shift operation is performed on this schedule. Next, if the schedule is improved, an encoding rule is applied to reversely transform the schedule into activity and mode assignment lists. This operation, called inheritance, unfortunately does not significantly improve the obtained results. The author also shows that repetition approach and island model of GA, as well as other selection operators, do not improve the performance of the proposed approach. Data sets with 10, 12, 14, 16, 18, 20, and 30 activities from PSPLIB are used in a computational experiment in which the best settings of the developed genetic algorithm are experimentally chosen, and the performance of the final fine tuned version of GA is compared with performance of other approaches, namely local search by Kolisch and Drexl (1997), genetic algorithm by Özdamar (1999), simulated annealing by Bouleimen and Lecocq (2003), and the truncated B&B by Hartmann and Drexl (1998). The obtained results show that the new genetic algorithm clearly performs better than other heuristics. The only exception is the truncated B&B which is slightly better for instances with small numbers of activities.

Schultmann and Rentz (2001) present a case study for the environment-friendly dismantling and recycling of buildings, and show how the dismantling on the construction site can be modeled as the MRCPSP. This problem is solved by truncated version of the B&B by Sprecher (1994). The obtained results are better than those obtained previously by other approaches.

Józefowska et al. (2001) propose two versions of simulated annealing: with and without a penalty function where a solution is represented by two lists: a mode assignment list and an activity list. Serial SGS is used as decoding rule. A penalty for the violation of nonrenewable resource constraints is added to the upper bound of the makespan in the version with penalty function. Neighbour solutions are generated by a random shift of a chosen activity and/or by changing randomly the chosen mode. The search process is controlled by the adaptive cooling scheme. Preprocessing is applied at the start of both the algorithms. Performance of both the proposed approaches is examined during a computational experiment where data sets from PSPLIB with 10, 12, 14, 16, 18, 20, and 30 activities, 3 modes per activity, two renewable and nonrenewable resources are used. The obtained results indicate that the version with penalty function is the better one and comparable with the Hartmann's GA (Hartmann, 2001).

Another implementation of simulated annealing is proposed by Bouleimen and Lecocq (2003) who use the same decoding rule and solution representation (except that the mode assignment should be resource-feasible). Neighbour solutions are generated using a two-phase procedure. In phase one, a resource-feasible change of a randomly chosen activity is made. Then for a fixed resource-feasible mode assignment, nonrenewable resource constraints are removed, and an improved schedule is searched by generating neighbour activity list using a random shift operation. The search process is controlled by a cooling scheme where the control parameter is changed according to the geometrical progression, and "reheating" is allowed for instances where the search process prematurely traps in local optimum. Computational experiment is performed for the same data sets as in Józefowska et al. (2001), but no comparison with any other approach is made.

The first tabu search approach to the MRCPSP is presented in (Nonobe and Ibaraki, 2002). In this implementation a solution is represented by an activity list and a mode assignment list. Neighbour solutions are generated either by a change on the mode assignment list, or by shifting some activities on the activity list. The schedule is generated by applying a scheme proposed by the authors, named CONSTRUCT, but further observations show that in some cases an optimal solution cannot be obtained by this procedure. Computational results are only reported for data set with 30 activities from PSPLIB, but no comparison with any other approach is presented.

Alcaraz et al. (2003) propose another genetic algorithm. As in the other approaches, preprocessing precedes the main genetic algorithm. The solution is encoded using an activity list, a mode assignment list, and an additional bit (forward/backward gene) denoting the scheduling generation scheme used to build the schedule: serial forward or backward. The fitness function is calculated either simply as the makespan, if the obtained schedule is feasible, or otherwise as $C_{max} + MFC_{max} - CP_{min} + PF$, where $C_{\max}$ is the makespan of the current solution, $MFC_{\max}$ is the maximum feasible makespan from the current population, $CP_{\min}$ is the critical path length calculated for the shortest duration modes, and $PF = \sum_{l=1}^{N}\left(\min\left\{0, R_l^{\nu} - \sum_{j=1}^{n} r_{jml}^{\nu}\right\}\right)$ for $l = 1, 2, \ldots, N$ is a penalty for violating nonrenewable resource constraints. It is showed that this fitness function is better than the one proposed by Hartmann (2001). A so-called two-point forward–backward crossover is proposed in which an offspring is build either from the head or from the tail depending on the value of the parent's forward/backward gene. The mutation operator consists of two-phases: in the first one activities are reordered in the activity list with a given probability in the way that the activity list after this operation is still precedence-feasible, and in the second phase, modes from the mode assignment list are changed with a given probability. The forward/backward gene may be changed in the first phase. Finally, a random replacement procedure is applied that replaces with a given probability solutions from the current population with other solutions generated randomly. The computational experiment is carried out using PSPLIB data sets containing instances with 10, 12, 14, 16, 18, 20, and 30 activities. The comparison with the local search by Kolisch and Drexl (1997), the genetic algorithm by Hartmann (2001), and the genetic algorithm by Özdamar (1999) is made only for the data set with 10 activities after generating 6000 solutions, and show that the proposed GA outperforms the local search and GA by Özdamar, but performs slightly worse than the Hartmann's GA. More results are presented for the comparison with the simulated annealing by Józefowska et al. (2001). In this case all data sets except the one with 30 activities are considered, and a stop criterion for both the analyzed algorithms is fixed at 5000 generated solutions. The presented results show that the proposed GA outperforms SA.

Chyu et al. (2005) propose a hybrid ant colony optimization (H-ACO) in which both the solution construction mechanism of B&B, and ant colony optimization (ACO) algorithms are hybridized. First, B&B is used to find a set of feasible (i.e. satisfying nonrenewable resource constraints) mode assignments. For each feasible mode assignment the following procedure is applied. Firstly, CPM is used to calculate the schedule length when the resource constraints are neglected. Secondly, a disjunctive rule is applied to add some arcs to the AoN network in order to remove resource conflicts. Thirdly, a new project makespan is calculated for the new AoN network representing the structure of the project. Next, a given number of mode assignments with the best potential values (the makespan obtained during the third step of abovementioned procedure) are chosen for the second phase of H-ACO where the ACO algorithm with the forward–backward improvement is applied to each chosen mode assignment. Finally, the best schedules obtained by ACO for each chosen mode assignment are compared, and the best one is chosen as a solution of the considered instance of the problem. H-ACO is compared with SA by Józefowska et al. (2001) using the standard data sets from PSPLIB. The obtained results show that H-ACO outperforms SA.

A population learning algorithm (PLA) is proposed by Jędrzejowicz and Ratajczak (2006). In this approach a solution is represented by a list of objects, where each object refers to one activity and includes some important information about it. At the beginning of PLA a preprocessing procedure runs. The main PLA consists of three learning stages. At the first learning stage three procedures are used. They are: the crossover operator, simple local search algorithm (LSA), and the RHIA procedure that is based on the idea of improving the resource utilization in a homogeneous intervals proposed in (Valls et al., 2004). At the second learning stage two evolutionary operators: crossover and mutation, as well as two heuristics procedures LSA and EPTA (exact precedence tree algorithm – based on precedence tree branching rule), are used. And finally, at the third learning stage two heuristics LSA and EPTA are applied. If the new generated solution is infeasible regarding nonrenewable resource constraints then a procedure trying to improve the usage of nonrenewable resources is run. The performance of PLA is validated for standard data sets from PSPLIB. The obtained results are compared with those presented in (Józefowska et al., 2001), and show that these two approaches are very similar in terms of performance.

Elloumi et al. (2006) develop an evolutionary algorithm which after preprocessing starts with generating an initial population of individuals represented by an activity list and a mode assignment list. For each individual the fitness function is calculated using rank-based assignment method, clustering heuristics for density computation, and penalty for violation of the nonrenewable resource constraints. Neighbourhood is generated using one point crossover, mutation operating on both lists independently, ranking and the roulette wheel selection methods, as well as the left shift procedure. The computational experiment, where instances from PSPLIB with 10, 12, 14, 16, 18, 20, and 30 activities are used, show that the proposed approach is comparable to the genetic algorithm by Alcaraz et al. (2003), and performs better than the genetic algorithms by Hartmann (2001) and by Özdamar (1999), the truncated B&B by Sprecher and Drexl (1998), simulated annealing algorithms by Józefowska et al. (2001) and by Bouleimen and Lecocq (2003), and local search by Kolisch and Drexl (1997).

Another metaheuristic approach, named particle swarm optimization (PSO), is applied to the MRCPSP in (Zhang et al., 2006). In this approach a solution is represented by two particles: the first one in the form of a random key representation of activity priorities, and the second one in the form of a mode assignment list. A new solution is generated using for each particle two formulas calculating both a new position, and a new velocity of a particle taking into account the position and the velocity of the best particle. A repairing procedure is applied for solutions with mode assignments infeasible according to nonrenewable resource constraints. Serial SGS is used as a decoding rule. A comparison with other heuristics including: the truncated B&B (Sprecher and Drexl, 1998), simulated annealing (Bouleimen and Lecocq, 2003) and genetic algorithm (Hartmann, 2001) is based on a computational experiment where instances of projects with 10, 12, 14, 16, 18 and 20 activities available in PSPSLIB are used. The results show that the proposed PSO performs a little bit worse than GA, and better than SA. Moreover, it is outperformed by truncated B&B for small number of activities (10) and (12).

A combinatorial particle swarm optimization (CPSO) is used by Jarboui et al. (2008) to generate solutions of the mode assignment subproblem of the MRCPSP. Next, for a fixed mode assignment, a local search algorithm is used to find suboptimal solutions of the resulting RCPSP. A computational experiment is carried out for seven data sets from PSPLIB containing instances with 10–30 activities. The results obtained by the proposed approach are compared with results generated by two other approaches only, namely the simulated annealing by Bouleimen and Lecocq (2003) and the particle swarm optimization by Zhang et al. (2006), and show that the proposed approach performs better than the two other approaches.

Chiang et al. (2008) propose another ant colony optimization algorithm called ACO-MRCPSP. A solution is represented by a mode assignment list and a random key list. The proposed algorithm consists of four phases: (1) preprocessing; (2) initialization where some control parameters are calculated for a given instance of the problem; (3) construction where artificial ants are guided to construct feasible solutions; and (4) feedback where the best schedule found so far is taken as the guide for properly reinforcing the pheromone concentration on the links of specific paths in the construction graphs. The performance of the ACO-MRCPSP is checked on the basis of a computational experiment where instances from PSPLIB with 10, 20, and 30 nondummy activities are used. The obtained results are compared with the results provided for four other state-of-the-art algorithms: simulated annealing by Józefowska et al. (2001), genetic algorithms by Özdamar (1999) and by Alcaraz et al. (2003), and evolutionary algorithm by Elloumi et al. (2006). The presented results show that the ACO-MRCPSP outperforms the other approaches.

An evolutionary algorithm, known as differential evolution, is proposed by Damak et al. (2009). In this approach a solution is represented by a mode assignment list and an activity list which needs not be precedence-feasible. Neighbour solutions are generated using two operators: mutation and crossover. Selection operator uses the values of the objective function which is penalized for solutions infeasible with respect to the nonrenewable resources. The performance of this algorithm is evaluated on the basis of a computational experiment where instances from PSPLIB with 10, 12, 14, 16, 18, and 20 activities are used. The obtained results are compared with the results obtained by two other approaches only: simulated annealing by Bouleimen and Lecocq (2003) and particle swarm optimization by Jarboui et al. (2008). Unfortunately, they are not compared with the results obtained by other more efficient algorithms.

Tseng and Chen (2009) develop a two-phase genetic local search algorithm where the same genetic local search algorithm runs with different initial populations for both phases for different search purposes. In the first phase, the initial population is generated randomly, and the set of good solutions (so-called elite set) is searched. In the second phase, the initial population contains mainly solutions from the elite set and the purpose of this phase is to search more thoroughly within the regions located by the solutions from the elite set. Similarly to other approaches, preprocessing is executed before the start of the main procedure. A single solution is represented by an activity list and a mode assignment list. Fitness function is calculated in the same way as in Alcaraz et al. (2003), and the proposed forward–backward local search method is used to transform a given solution to the standard representation. After this transformation each schedule has exactly one solution representation. Neighbour solutions are generated using two-point crossover proposed by Alcaraz et al. (2003) and its slightly modified version, as well as two mutation operators which allow to diversify population lightly or heavily, respectively. The first mutation operator is taken from Alcaraz et al. (2003), whereas the second one is a new concept developed by the authors. Selection is made using ranking and 2-tournament methods. Computational experiment carried out using instances from PSPLIB with 10, 12, 14, 16, 18, 20, and 30 nondummy activities was executed to compare the performance of the proposed approach with five other algorithms: local search by Kolisch and Drexl (1997), simulated annealing by Józefowska et al. (2001), and three versions of genetic algorithms – by Özdamar (1999), by Hartmann (2001), and by Alcaraz et al. (2003). The presented results show that the proposed two-phase genetic local search algorithm outperforms the other approaches.

Lova et al. (2009) propose a hybrid genetic algorithm (MM-HGA). As in many other algorithms, before the genetic algorithm itself is started, a preprocessing procedure runs in order to reduce the search space. A solution is represented by an activity list and a mode assignment list, and two additional genes namely: forward/backward gene adapted from (Alcaraz et al., 2003) and serial/parallel gene denoting the variant of SGS used to build the schedule. A new fitness function is proposed. It is showed that applying this function in the proposed approach improves its performance compared with two other fitness functions proposed by Hartmann (2001) and by Alcaraz et al. (2003), respectively. Two-point crossover as well as mutation operators are used to obtain the next generation. Mutation is applied to both the activity list and the mode assignment list, as well as to the two additional genes. On the activity list a random shift is performed. Mutation applied on the mode assignment list depends on the feasibility of the solution with respect to nonrenewable resources. For a feasible solution modes are changed randomly with a given probability that is the same for all activities. If a solution is infeasible than a so-called massive mutation is applied: a mode is randomly chosen for a randomly chosen activity until either a feasible mode assignment is found or modes are already chosen for all activities. The forward/backward and serial/parallel genes are randomly changed with a given mutation probability. A 2-tournament selection operator with elitism is applied to generate the next population. Moreover, two additional mechanisms are used: a random replacement of some solutions from the current population, and a multi-mode forward–backward improvement. The data sets from PSPLIB with instances containing 10, 12, 14, 16, 18, 20, and 30 activities are used in the computational experiment. The performance of MM-HGA is compared with local search (Kolisch and Drexl, 1997), genetic algorithms (Özdamar, 1999; Hartmann, 2001; Alcaraz et al., 2003), and simulated annealing (Bouleimen and Lecocq, 2003) for the instances with 10 activities. Moreover, for all data sets, except the one with 30 activities, MM-HGA is compared with simulated annealing by Józefowska et al. (2001) and genetic algorithm by Alcaraz et al. (2003). The obtained results show that MM-HGA outperforms the other heuristics.

Yet another implementation of a genetic algorithm is presented by Van Peteghem and Vanhoucke (2010). They use a so-called bi-population version of GA in which two different populations of the same size are utilized: a population $POP_R$ that contains right-justified schedules only, and a population $POP_L$ that contains left-justified schedules only. The idea of justifying the schedule to the right or to the left is adapted from (Valls et al., 2005). A solution is encoded in the form of a topological ordering random key representation and a mode assignment list. A random key representation is a vector of priority values, and when topological ordering (see, e.g. Valls et al., 2003) is employed, priorities preserve precedence constraints, i.e. if $A_i$ precedes $A_j$ then the priority value for $A_i$ is smaller than the one for $A_j$. Serial SGS is used to construct a schedule during forward–backward scheduling. The forward procedure is applied to the left-justified population, and is used to build a right-justified schedule. Next, the completion times of activities are used as the priority values for the random key representation, and each activity is scheduled as late as possible. Genetic operators are then applied to the right-justified schedules, and the backward procedure runs for the obtained population of right-justified schedules. A mode improvement procedure runs together with serial SGS. This procedure is applied with a given probability to activities of the project, and checks for a chosen activity if the change of the assigned mode leads to an earlier completion time of this activity without increasing the penalty function value. The penalty function is calculated for the nonrenewable resources consumption that exceeds resource availability limitations. Similarly to other approaches, preprocessing runs before the start of the main algorithm. Two fitness functions are investigated: the one proposed by Hartmann (2001) and the other one proposed by Alcaraz et al. (2003). The offspring solutions are generated using 2-tournament selection, one-point crossover, and two mutation operators (one acting on the mode assignment list, and the other acting on a random key vector). In the computational experiment the data sets from PSPLIB containing instances with 10, 12, 14, 16, 18, 20, and 30 activities are used. The performance of the proposed algorithm is compared with the performance of other approaches including: the genetic algorithms by Özdamar (1999), by Hartmann (2001), and by Alcaraz et al. (2003), as well as the local search by Kolisch and Drexl (1997), and the simulated annealing by Józefowska et al. (2001). The obtained results show that the considered genetic algorithm approach is the most powerful heuristic developed up to now. The average relative deviation from optimum is less than 0.5% for all data sets, and the percentage of optimal solutions found is from about 88% for the data set with 20 activities up to almost 100% for the data set with 10 activities.

#### 5.3.3. Other approaches

Talbot (1982) proposes to use his B&B algorithm as a heuristic setting up a limit on computational time. Also Patterson et al. (1989) as well as Sprecher and Drexl (1998) suggest to use their B&B algorithms as heuristic procedures by imposing a time limit.

Maniezzo and Mingozzi (1999) use a new mathematical formulation (12)–(21) for the MRCPSP to develop new heuristic algorithm based on the Benders decomposition and named HBEND. The LP-relaxed problem in the form used to calculate the third lower bound proposed by them is decomposed into a master problem and a subproblem, which run iteratively. At each iteration the master problem constructs an RCPSP instance assigning modes to activities in such a way that nonrenewable resource constraints hold. Next, the subproblem heuristically finds a valid lower bound for this instance of the RCPSP. This lower bound is then used to obtain a valid Benders cut that is added to the master problem in the next iteration. A number of the best RCPSP instances obtained from the master problem are memorized and solved optimally using the B&B algorithm for the RCPSP developed by Mingozzi et al. (1998). It is done at the end of HBEND to improve the quality of the MRCPSP upper bound. This approach is compared with the local search approach by Kolisch and Drexl (1997), and the truncated B&B by Sprecher and Drexl (1998). The test instances with 10, 20, and 30 activities from PSPLIB are used in this experiment, and the obtained results show that HBEND requires a number of cuts before achieving good results. Moreover, it outperforms both the considered algorithms (except results for the set with 10 activities where HBEND is outperformed by the B&B).

This approach is once more presented in (Boschetti and Maniezzo, 2009), where it is compared with other well performing algorithms, namely the truncated B&B by Sprecher and Drexl (1998), the genetic algorithm by Hartmann (2001), two versions of simulated annealing proposed respectively by Józefowska et al. (2001) and by Bouleimen and Lecocq (2003), and the particle swarm optimization (PSO) by Zhang et al. (2006). The results of the computational experiment obtained for instances from data sets with 10 and 20 activities available in PSPLIB show that HBEND is outperformed by other heuristics except the truncated version of B&B.

### 5.4. Special cases and extensions

Some special cases as well as extensions of the basic MRCPSP model have been presented in several papers.

#### 5.4.1. Special cases

The most commonly considered special case of the MRCPSP is its version without nonrenewable and doubly constrained resources. Such problem is studied by Elmaghraby (1977), Boctor

(1993, 1996a,b), Mori and Tseng (1997), Knotts et al. (2000), Artigues and Roubellat (2000), Gagnon et al. (2005), and Lova et al. (2006). Other special cases are the DTCTP and the DTRTP described in Sections 4.1.2 and 4.2.1, respectively. Obviously, the RCPSP (see Section 2) is also a special case of the MRCPSP, but the review of the literature concerning the RCPSP is out of the scope of this paper.

Elmaghraby (1977) was the first one who considered multiple operating modes of activities in the project scheduling problem. The objective of this problem is to minimize both costs and the makespan. The set of constraints is formulated, and an example of such a problem is presented.

Boctor (1993) compares 21 priority rule heuristics on the basis of a computational experiment consisting of 240 problems with 50 and 100 activities, and 1, 2, and 4 renewable resources. Nonrenewable and doubly constrained resources are not considered. The following priority rules are used for ordering activities: MIN SLK, MIN LFT, SPT, maximum number of immediate successors (MAX NIS), maximum remaining work (MAX RWK), longest processing time (LPT), and maximum number of subsequent candidates (MAX CAN). Ties, if occur, are broken by choosing the activity with the smallest number. Modes are selected using the shortest feasible mode (SFM), the least criticality ratio (LCR), and the least resource proportion (LRP) heuristics. The resulting heuristics that are examined in a computational experiment appear as possible combinations of any priority rule for activities and any mode assignment rule. The results of the computational experiment show that the best combination of heuristic rules are the following: MIN SLK-SFM, MAX RWK-SFM, MAX CAN-SFM, and MIN LFT-SFM. Another heuristic approach for the same problem is also proposed by Boctor (1996a). This heuristic uses forward and backward parallel scheduling of schedulable and nondominated activity-mode combinations. An activity-mode combination is a subset of the set of eligible activities where for each activity a mode is assigned. Such a combination is called schedulable at a given time period if the resources available (nonallocated) during this period and in succeeding periods allow to execute all activities from the corresponding subset in the assigned modes. An activity-mode combination is dominated by another combination if at least one of the following conditions holds: (i) the first combination is a subset of the second one; (ii) both combinations are identical except that the mode assigned to one activity in the second combination is shorter than the mode for the same activity in the first combination. The results of the computational experiment for the same set of 240 problems show that the proposed approach is better than the previously examined priority rules. The same problem is considered once more by Boctor (1996b) who uses a simulated annealing algorithm to find suboptimal schedules. A solution is represented by an activity list. Modes are assigned to activities during the execution of the serial SGS procedure that is used as a decoding rule. For a given activity, a mode that guarantees the earliest precedence- and resource-feasible completion time of this activity in the generated schedule is selected. Neighbour solutions are generated by shifting a randomly chosen activity to a new precedence-feasible position in the activity list, chosen in a random way. The search process is controlled by a cooling scheme where the control parameter is changed according to the geometrical progression. The computational experiment is once more carried out for all 240 problems from the Boctor's benchmark set. The obtained results show that the proposed approach outperforms all heuristics proposed previously by Boctor (1993, 1996a).

Mori and Tseng (1997) propose a genetic algorithm for the MRCPSP without nonrenewable or doubly constrained resources. A direct representation of a schedule is used. This representation contains information about the activity, the assigned mode, the priority, and the calculated start and completion times of this activity. The initial population is build by setting activities in an ascending order, randomly choosing a mode for each activity. The priority is determined randomly for activity order interval, and start and completion times of each activity are calculated. Two parent chromosomes are used in the crossover operation. One of the parents is chosen randomly from the current population while the second parent is always the solution with the smallest project duration from among all solutions in the current population. The crossover point is chosen randomly, and the offspring solution inherits the head from the second parent and the tail is constructed using the remaining activities from the first parent. Two mutation operators are used. In the first one, a set of activities is chosen and then a mode is randomly chosen for each selected activity. In the second one, a new solution is constructed using the same method as in the initialization phase. A new generation is build of: the 20 best solutions from the previous generation, 10 solutions generated using a crossover operator, 7 solutions generated using a mutation operator, and 3 solutions generated randomly. A Random Activity Network Generator proposed by Demeulemeester et al. (1993) is used to construct 700 problems for a computational experiment. The generated test problems have 10, 20, 30, 40, 50, 60, or 70 activities, 2 to 4 modes for each activity, and 4 renewable resources. The proposed GA is compared with STOCOM (Drexl and Grünewald, 1993) that generates 100 feasible solutions for each test problem. The obtained results show that the proposed GA is significantly better than STOCOM, especially for larger problems.

The next approach to the MRCPSP without nonrenewable resources is presented by Knotts et al. (2000) who propose to use the agent technology that is known from artificial intelligence. For each activity of the project one agent is created that is responsible for acquiring the resources required by this activity. Agents try to find the best solution during the simulation process in which they act according to some rules which determine their behavior. The performance of the approach is validated on the basis of a computational experiment in which an extended data set from (Maroto and Tormos, 1994) is used, and the obtained results are compared with the results obtained by commercial software. These results show that the considered approach performs better than some commercial applications but is outperformed by others.

Artigues and Roubellat (2000) develop a polynomial activity insertion algorithm, which is used to insert an activity into an existing schedule for problems with minimization of the maximum lateness. This algorithm can also be used to generate neighbour solutions in local search methods applied to the MRCPSP with renewable resources only and minimization of the makespan. In such an application this insertion algorithm must be used jointly with an activity removal algorithm. Unfortunately, no computational results are reported for the application of this method to the MRCPSP.

Another approach proposed by Gagnon et al. (2005) for the MRCPSP without nonrenewable resources is based on tabu search. A solution is represented by an activity list and a mode assignment list, and seven operators are proposed to generate neighbor solutions. A computational experiment is performed using data sets by Boctor (1993). The proposed approach is compared with simulated annealing (Boctor, 1996b) only.

Lova et al. (2006) consider several single-pass and multi-pass heuristics based on priority rules for the MRCPSP with renewable resources only. They analyze three components of such heuristics: schedule generation scheme, priority rule, and mode selection rule on the basis of a computational experiment in which Boctor's (1993) data sets are used. The analyzed schedule generation schemes are serial and parallel; priority rules include: MIN EFT, MIN EST, MAX DUR MIN LFT, MIN LST, MIN SLK, MAX NIS, MAX RWK, SPT, minimum activity number (MIN AN), greatest rank positional weight (GRPW), greatest resource demand (GRD), minimum latest start and finish time (MIN LSTLFT), minimum free slack (MIN FREE); and earliest feasible finish time (EFFT) is used as a mode selection rule. The results show that serial SGS greatly outperforms parallel SGS, and the multi-pass heuristic combining eight priority rules (LSTLFT, LFT, LST, and RWK with both versions of SGS) gives the best results.

*5.4.2. Extensions*

Apart from the above special cases, also some extensions of the MRCPSP have been considered in several papers.

An extension of the MRCPSP to its version with generalized precedence relations (also called time windows), denoted as the MRCPSP-GPR (or MRCPSP/max), is studied in several publications. Exact approaches based on the B&B method are proposed in (De Reyck and Herroelen, 1998; Dornorf, 2002; Heilmann, 2003). Some heuristic algorithms are proposed by De Reyck and Herroelen (1999) who use a hybrid of tabu search and truncated version of their B&B, by Heilmann (2001) who proposes a multi-pass priority rule approach with backplanning which is based on an integration approach and embedded in random sampling, and by Calhoun et al. (2002) who implement tabu search. Moreover, Van Hove and Deckro (1998) propose a B&B approach for the MRCPSP with minimal time lags only. An exhaustive review of some project scheduling problems with time windows may be found in (Neumann et al., 2002). Recently, Barrios et al. (2009) propose for the MRCPSP-GRP a so-called double genetic algorithm which outperforms other state-of-the-art approaches in medium and large instances. This version of a genetic algorithm consists of two-phases. In the first phase algorithm searches for the best modes of the activities, and in the second phase the makespan is minimized. For each phase different parameters and mechanisms are defined including representation, fitness, operators, etc.

An extended version of the MRCPSP with so-called *mode identity constraints* is considered by Salewski et al. (1997). The resulting problem is called the *mode identity resource-constrained project scheduling problem* (MIRCPSP), and is motivated by real-world situations where several activities should be performed in the same way, i.e. by allocating them the same resources. Practical examples of such a problem occur in an audit staff scheduling, timetabling, course scheduling, etc. Formally, in this problem the set of all project activities is partitioned into several disjoint subsets, and all activities belonging to the same subset have to be performed by the same resources. The time and cost of executing activities from such a subset depend on the resources assigned. Moreover, for each activity a deadline, a ready time, and a set of mode-dependent finish-to-start time lags with direct predecessors are defined. A mathematical model of the problem is formulated, and the NP-hardness in the strong sense is proved. A two-phase heuristic is used to find a good feasible schedule. In phase one, for each subset of activities a mode is selected randomly. In phase two, a solution is build by scheduling randomly chosen activities from the eligible set.

Ahn and Erenguc (1998) and Erenguc et al. (2001) propose heuristic and exact procedures, respectively, for the RCPSP with *multiple crashable modes* (RCPSPMCM). In this problem each activity can be executed in one of several modes, duration of which may be shortened (crashed) by additional cost. Thus, this problem can be viewed as a combination of the MRCPSP with the TCTP. Indeed, in the absence of the resource constraints the RCPSPMCM reduces to the TCTP, and in the absence of crashing within a mode it becomes the MRCPSP.

The *multi-skill project scheduling problem* (MSPSP) (see e.g. Bellenguez and Néron, 2005; Bellenguez-Morineau and Néron, 2008; Bellenguez-Morineau, 2008) is another project scheduling problem where activities may be executed in one of several modes. In this problem multi-skill resources are used during activity processing. Each unit of a multi-skill resource treated as unary resource is able to perform activities requiring different skills. Thus, if a given resource unit is allotted to an activity requiring one skill, it cannot be allotted at the same time to any other activity requiring another skill managed by this resource unit. In other words, each resource unit can be assigned to at most one skill at a time. In consequence, the MSPSP can be treated as multi-mode version of the project scheduling problem with resource/resource trade-off. Li and Womer (2009) extend this model by employing general precedence relations, and replacing the project makespan minimization by a cost-related objective function. Another extension of the MSPSP is presented in (Valls et al., 2009) where there are general precedence relations considered, activity processing times depending on the skill level of the resource unit assigned to this activity, and an objective function which takes into account: criticality of activities, assignment of the best skilled resources to each activity, and well-balanced resource workload levels.

The MRCPSP with renewable and nonrenewable resources replaced by partially renewable ones is considered by Zhu et al. (2006) who propose a B&C approach to solve this problem optimally. The authors show on the basis of a computational experiment that the proposed approach appears very promising, and can be successfully applied to the MRCPSP as well.

Mika et al. (2008) present an extension of the MRCPSP where schedule-dependent setup times are considered. Three algorithms are proposed for this problem and compared on the basis of a computational experiment. The obtained results show that the proposed tabu search algorithm outperforms the other two presented approaches, namely the multi-start iterative improvement and random sampling. In all the considered approaches, a solution is represented in the same way, and the number of generated solutions is identical.

Li and Womer (2008) model the supply chain configuration problem with resource constraints as the MRCPSP with due dates and additional quality level constraints, and show that this model can be easily extended by applying minimal and maximal time lags, variable resource availability, and various objectives.

A problem of scheduling tests in automotive research and development projects is considered by Bartels and Zimmermann (2009), who model this problem as the MRCPSP with renewable and cumulative resources, as well as minimal and maximal time lags. Some resources used during the course of the project have to be created by certain activities of the project in order to enable the execution of further activities, but it is unknown in advance how many units of these resources are required. Moreover, partially ordered destructive relation between pairs of activities is introduced, because some tests (activities) may destroy a resource unit being used, and therefore it cannot be used in other tests anymore. A mixed integer linear programming (MILP) model of this problem is formulated and used in the computational experiment for small instances solved optimally by CPLEX. For large instances both single-pass and multi-pass priority rule-based heuristics are proposed.

A sports league scheduling problem which occurs in planning non-professional table-tennis leagues is considered by Knust (2010). This problem consists in finding a schedule for a time-relaxed double round robin tournament with many different hard and soft constraints. One of the two presented approaches is to model this problem as the MRCPSP with partially renewable resources and time-dependent resource profiles. All hard constraints are covered by introducing appropriate additional renewable, nonrenewable, or partially renewable resources, whereas soft constraints are incorporated into the objective function in the form of penalties. All activities have unit processing times. Although a large number of resources have to be introduced to cover all hard constraints, each activity requires only a few of them and, in consequence, the solution algorithm is quite fast. A two-stage local search algorithm is proposed, where in the first stage some theoretical results on so-called balanced home-away assignments (Knust and von Thaden, 2006) are implemented, and in the second stage algorithms known from the RCPSP are used (in this implementation – the genetic algorithm by Hartmann (1998)).

## 6. Multi-mode problems with other objectives

### 6.1. Financial objectives

#### 6.1.1. Multi-mode resource-constrained project scheduling problem with discounted cash flows

The *multi-mode resource-constrained project scheduling problem with discounted cash flows* (MRCPSPDCF) is an extension of the MRCPSP where cash flows (positive and/or negative) occur over the course of the project. If cash flows are associated with activities, they can, in the most general case, depend on the processing modes of the activities. The objective of the MRCPSPDCF is to maximize the NPV of all cash flows of the project.

The MRCPSPDCF has only been considered in few papers, where heuristic approaches have been developed. To the best of our knowledge, no exact procedure for the MRCPSPDCF has been studied.

Sung and Lim (1994) study a problem with positive and negative cash flows, and availability constraints imposed on capital and renewable resources. The objective is to maximize the NPV. Resource-duration interactions are considered in analyzing the problem. A two-phase heuristic solution algorithm is exploited and tested with various numerical problems for its effectiveness and efficiency.

Ulusoy et al. (2001) present the general MRCPSPDCF with renewable, nonrenewable, and doubly constrained resources. Positive and negative cash flows are associated with events and/or activities, depending on the considered model. Four payment models are considered: lump-sum payment at the completion of the project, payments at fixed event nodes, payments at equal time intervals, and progress payments. A genetic algorithm with a special crossover operator able to exploit the multi-component nature of the problem is proposed. 93 problems from the set of instances from the literature (Ulusoy and Özdamar, 1995) are solved, under the four payment models and different resource combinations. The efficiency of the presented GA algorithm is analyzed.

Mika et al. (2005) consider the general MRCPSPDCF, where a project is represented by an AoN network. A positive cash flow is associated with each activity. Four payment models are considered: lump-sum payment at the completion of the project, payments at activity completion times, payments at equal time intervals, and progress payments. Local search metaheuristics: simulated annealing and tabu search are proposed to solve the problem. In both the implementations, a solution is classically represented by an activity list and a mode assignment list. A comprehensive computational experiment is described, performed on a set of instances based on standard test problems constructed by the ProGen project generator, where, additionally, activity cash flows are generated randomly from the uniform distribution. The experiment is carried out for the adopted PSPLIB instances with 10, 12, 14, 16, 18, 20, and 30 instances, four payment models, five values of the discount rate, and various combinations of the interval lengths in payment models, resulting in a huge number of 192100 MRCPSPDCF instances solved by each of the metaheuristics tested.

Seifi and Tavakkoli-Moghaddam (2008) consider a problem similar to the one presented by Ulusoy et al. (2001). The main differences are: the AoN representation of the project network, and the objective function which is a sum of the NPV and the proposed by Liu and Wang (2006) activity cost measure. The measure is minimized and takes into account the costs of executing some activities by subcontractors. The same four payment models as in Ulusoy et al. (2001) are considered but for the "payments at fixed event nodes" model events occur at the completion times of activities, and therefore this payment model becomes the "payments at activity completion times" model considered by Mika et al. (2005). A simple implementation of simulated annealing is proposed for the problem under consideration, and its performance is verified on the basis of a computational experiment in which instances from PSPLIB are used.

Kavlak et al. (2009) consider a so-called client-contractor bargaining problem in the context of the MRCPSPDCF with renewable resources only. Two payment models are analyzed: progress payments and payments at activity completion times. The project is represented by an AoN network. The project duration is bounded from above by a deadline imposed by the client, which constitutes a hard constraint. The bargaining objective is to maximize the bargaining objective function comprised of the objectives of both the client and the contractor. The bargaining objective function is expected to reflect the two-party nature of the problem environment, and seeks a compromise between the client and the contractor. The bargaining power concept is introduced into the problem by the bargaining power weights used in the bargaining objective function. Simulated annealing algorithm and genetic algorithm approaches are proposed as solution procedures. A solution is represented by a combination of three serial lists: activity list, mode assignment list, and idle time list. The idle time value represents the exact idle time to be inserted before the start of the corresponding activity in the activity list. The proposed solution methods are experimentally compared on the PSPLIB instances with 14, 20, and 30 activities, adopted by eliminating the tardiness costs, relaxing the deadlines, and excluding the nonrenewable resources. Also sensitivity analysis is conducted for different parameters used in the model, namely the profit margin, the discount rate, and the bargaining power weights.

Chen et al. (2010) consider the general MRCPSPDCF with cash inflows and outflows. An ant colony system (ACS) algorithm is proposed to solve this intractable problem. In the presented algorithm, the AoA network of the problem is first converted into a mode-on-node (MoN) graph, which next becomes the construction graph for the ACS algorithm. Based on the construction graph, the authors apply the serial SGS for artificial ants to explore the solutions to the problem. In the process of this algorithm, each ant maintains a schedule generator and builds its solution by selecting arcs on the graph using pheromone and heuristic information. Eight different domain-based heuristics are developed to enhance the search skill of ants by considering the factors of time, cost, resources, and precedence constraints. The proposed ACS approach is compared with the authors' implementations of genetic algorithm by Ulusoy et al. (2001), as well as simulated annealing and tabu search by Mika et al. (2005), on 55 randomly generated instances with from 13 up to 98 activities. On the basis of experimental results the authors state that their algorithm outperforms the other three metaheuristics.

#### 6.1.2. Problems with probabilistic cash flows

Although we do not deal with nondeterministic models in this survey, two papers considering probabilistic cash flows are worth mentioning, as they belong to the small group of papers on the NPV maximization under multiple processing modes of activities.

Özdamar and Dündar (1997) introduce a new model concerning housing projects, where an initial capital covers activity expenditures in the starting phase of the project, and then customers who arrive randomly over the project span, provide the necessary funds for continuation. Capital is considered as a nonrenewable resource which is the only limited resource in the model. It is reduced by activity expenditures and augmented by the sales of flats. Activities can be carried out in different operating modes, and the total cost of an activity is fixed irrespective of its operating mode. The rate of activity expenditures differs from mode to mode. The objective is to maximize the NPV. The authors propose a flexible heuristic algorithm for solving the capital-constrained mode selection problem, where there exist general precedence relationships among activities, and the magnitude of precedence lags depend on the specific activity mode selected. The algorithm is tested using a typical housing project with real data and also by using hypothetical test problems.

Similar situations in the housing industry are further analyzed in (Özdamar, 1998), where the same stochastic model, involving probabilistic cash inflows, is used. The contractor, who is the owner of the project, starts with an initial capital to cover the activity expenditures, and then capital is augmented by the sale of flats, which take place randomly over the progress of the project. In this risky environment, the contractor has to decide on the rate of expenditure at each decision time in order to maintain a positive cash balance. Hence, activities are performed in multiple processing modes with different durations and the same total cost. A heuristic to construct and reconstruct schedules during the progress of the project is proposed with the aim of maximizing the NPV while completing the project on time. The heuristic incorporates dynamic mode selection objectives which change adaptively according to the current status of the project. Computational experiments demonstrate that the heuristic provides satisfactory results regarding the feasibility of the schedules with respect to the project due date and the nonrenewable resource constraints.

#### 6.1.3. Payment scheduling problem

In the models presented above it is assumed that the amounts and timing of cash flows are known. In (Dayanand and Padman, 1997) the authors argue that the expenses associated with activities are usually known but the amounts and timing of payment can be important variables to negotiate by the contractor in order to improve financial returns. Consequently, the *payment scheduling problem* (PSP) is formulated in which both the amounts and the timing of the payments have to be determined to maximize the NPV subject to a project deadline.

The multi-mode version of the PSP has been considered in a few papers only.

Ulusoy and Cebelli (2000) report a new approach to the payment scheduling problem. The authors set up a special multi-mode problem, where the goals of the contractor and the client are joined in one model. They look for an equitable solution, which is defined as such in which both the contractor and the client deviate from their respective ideal solutions by an equal percentage. The ideal solutions for the contractor and the client result from having a lump-sum payment at the start and at the end of the project, respectively. A double-loop genetic algorithm is proposed to solve the problem, where the outer loop represents the client and the inner loop – the contractor. Ninety-three problems from the set of instances from the literature (Ulusoy and Özdamar, 1995) are solved, and some computational results are reported.

He and Xu (2008) consider the so-called *multi-mode project payment scheduling problem* (MPPSP) with bonus-penalty structure where activities can be performed with several modes, and a bonus-penalty structure exists at the deadline of the project. In the problem the decisions on when to schedule events and payments, the magnitude of each payment, and the performing mode of each activity need to be optimized. A two-module simulated annealing heuristic is proposed to solve the mixed integer nonlinear programming (MINLP) models for the contractor and the client, and a satisfactory solution, which consists of payment event set, event schedule, and payment amount set, may be found through iterations between the heuristic's two-modules. The profits of the two parties of the contract are changed significantly by the bonus-penalty structure, and the structure may be considered as a coordination mechanism essentially, which may enhance the flexibility of payment scheduling and be helpful for the two parties to get more profits from the project. Through solving and analyzing an instance the insight that the bonus-penalty structure may advance the project completion effectively, and improve the profits of the two parties in the meantime, can be obtained.

He et al. (2009) consider the MPPSP with no resource constraints, where the activities can be performed with one of several discrete modes, and the objective is to assign activity modes and progress payments so as to maximize the NPV of the contractor under the constraint of project deadline. Using the event-based method, the basic model of the problem is constructed, and in terms of different payment rules it is further extended through changing the constraints on payment time as the progress-based, expense-based, and time-based models. The strong NP-hardness of the problem is proved by simplifying it to the deadline subproblem of the discrete time/cost trade-off problem. Simulated annealing and tabu search metaheuristics are proposed to solve the problem. The two algorithms are computationally compared on a data set constructed by ProGen project generator.

### 6.2. Resource-based objectives

Hsu and Kim (2005) consider the *multi-mode resource investment problem* (MRIP) which is an extension of the resource investment problem (RIP), where at least one activity may be executed in one of several modes. In this formulation activities are executed using renewable resources only, and a project duration is upper-bounded by a given due date. A priority rule-based heuristic, which simultaneously takes into account the resource usage level and the project due date in one decision rule, is proposed to find suboptimal solutions of the problem.

Sabzehparvar et al. (2008) propose a mathematical model for the *multi-mode resource investment problem with general precedence relations* (MRIP-GPR). In this model activities are executed using renewable resources only, project has to be finished before its deadline, and for some pairs of activities there are minimal and maximal time lags which depend on the modes of both activities.

Mika and Węglarz (2004) consider the *multi-mode resource leveling problem* (MRLP) which is formulated similarly to the MRCPSP, assuming that there are two kinds of limitations: limited availability of renewable and nonrenewable resources, and a deadline for the entire project. The objective is to minimize the weighted sum of squared deviation from the assumed resource usage level for all renewable resources. Two heuristics: simulated annealing and random sampling are developed for this problem and compared on the basis of a computational experiment with instances with 10 and 20 activities. The obtained results show that simulated annealing performs better for the projects with deadlines close to optimal makespans, whereas for instances with later deadlines random sampling surprisingly outperforms simulated annealing.

## 7. Discrete–continuous project scheduling

In Sections 4.1.2, 4.2.1, 5, and 6 we consider models where duration of an activity is defined by its processing mode. In other words, activity durations depend on the numbers of units of discrete resources allotted to this activity. In many practical situations resources may be allotted to activities in arbitrary amounts from a given interval, i.e. are continuous ones (see Section 3.1.1).

Situations of this type occur when, e.g. activities are processed by parallel processing units driven by a common (electric, pneumatic, hydraulic) power source, like commonly supplied grinding or mixing machines, electrolytic tanks, or refueling terminals. Also in computer systems, where multiple processors share a common primary memory, if it is a paged-virtual memory system and the number of pages goes into hundreds, then primary memory can be treated as a continuous resource (Węglarz, 1980b). On the other hand, the processors themselves can be considered as a continuous resource in scalable (SPP) or massively parallel (MPP) systems when the number of them is huge (hundreds or even thousands). Another example concerning the forging process in steel plants was analyzed in (Janiak, 1991a). Forgings are preheated by gas up to an appropriate temperature in forge furnaces. Gas flow intensity, limited for the whole battery of forge furnaces, is a continuous resource. Observe, that in such situations we may speak of uncountable amount of activity processing modes.

As mentioned in Section 3.2.2, two activity processing models appear in the literature. When continuous resources are taken into account, the processing time vs. resource amount model defines the activity duration as a function of the amount of a continuous resource allotted to this activity. This model is a straightforward generalization of the discrete time/resource trade-off model, as mentioned in Section 4.2.2. It is implicitly assumed that the resource amount allocated to an activity does not change during its execution. Within this approach, the existence of some polynomially solvable cases of machine scheduling problems for linear functions are proved (Janiak, 1991b). In the second model – the processing rate vs. resource amount model, the processing rate of an activity is a function of the amount of a continuous resource allotted to this activity at a time. In this case, the amount of the continuous resource allotted to an activity may change during its execution. A fundamental result for this model and a renewable resource can be found in (Węglarz, 1976), whereas for a doubly constrained resource in (Węglarz, 1981). In these papers activities are assumed to be continuously-preemptable. From between the two models, the processing rate vs. resource amount model is more natural in the majority of practical situations, since it reflects directly the temporary nature of renewable resources. As examples, functions like rotational speed vs. electric current, or progress rate vs. number of primary memory pages allotted to a program can be given. This temporary character is vital as it is often ignored in practice in the case of some doubly constrained resources, which can be then treated as nonrenewable ones. Money is a good example of such a resource, where usually only the total consumption is taken into account, whereas it also has a temporary nature as it may be limited in a given period. Even the most typical renewable continuous resource, which is power, is also, in general, doubly constrained, since its consumption, i.e. energy, is also limited. Moreover, the processing rate vs. resource amount model enables to perform a deeper analysis of the properties of optimal schedules, and can even lead to analytical results in some cases. Because of that, it is sometimes reasonable to treat a discrete resource as a continuous one in order to use this model. Such an approach may be applied when there are sufficiently many allotments of the discrete resource for processing an activity, e.g. in SPP or MPP systems.

A special case of the processing rate vs. resource amount model with continuous resources is considered in Leachman (1983), Leachman et al. (1990) and Kis (2005, 2006). The processing rate of an activity, called intensity, is proportional to the relative amount of resources allotted to an activity at a time. The objective function considered by Leachman et al. (1990) is the project duration. A heuristic approach to solving this problem is proposed. New resource allocation is calculated each time when the amount of the resources that can be allocated changes due to activity completion or to the change of the total resource amount available. At each decision point a priority list is generated on the basis of the projected lateness of the activities. The resources are allocated to activities with respect to this priority list. Two algorithms are compared. In the first one, any reduction in the level of resources already committed to an activity is not allowed, while the second algorithm allows both reduction and increase of the activity intensity level during the activity execution. In the model proposed by Kis (2006) it is assumed that variable intensity activities are connected by feeding precedence constraints, the known time horizon is divided into a discrete number of units, and the objective is to minimize the violation of resource constraints. Feeding precedence constraints allow some overlap in the execution of the connected activities, and capture the flow of material or information between them. For this problem, an exact algorithm based on the B&C technique is developed. Some new polyhedral results are obtained, that can be used to strengthen the formulation, and computational results are summarized.

*Discrete–continuous project scheduling problems* arise when activities simultaneously require for their execution discrete and continuous resources. The problem is, in general, to find a schedule of activities satisfying precedence and discrete resource constraints, and, simultaneously, an allocation of continuous resources to activities such that the considered objective is optimized. Such a problem with one discrete resource being a set of parallel machines was studied in (Węglarz, 1979) for preemptable activities, and in (Józefowska and Węglarz, 1998) for nonpreemptable, independent activities. In both the papers there is one continuous, renewable resource, whose total amount available at a time is limited. The processing rate of an activity is a continuous increasing function of the amount of the continuous resource allotted to the activity at a time. The objective is to minimize the makespan. The methodology presented in (Józefowska and Węglarz, 1998) was next generalized in (Józefowska et al., 1999) for the case of precedence-related activities and an arbitrary number of discrete resources. Discrete-continuous project scheduling problems to minimize the makespan are then considered in (Waligóra, 2010), where heuristic approaches to the problems are proposed. The first results concerning the NPV maximization are presented in (Waligóra, 2008a), and the general methodology for discrete–continuous problems with discounted cash flows and various payment models is described in (Waligóra, submitted for publication). A comprehensive survey on models and algorithms for discrete–continuous project scheduling can be found in (Waligóra, 2008b).

Properties of optimal solutions to the discussed problems depend on the processing rate functions. In the case of convex functions, the problem becomes easy, since consecutive completion of activities, each one using the total available amount of the continuous resource during its execution leads to optimal schedules. In the case of concave processing rate functions, it is desirable to perform activities in parallel. The solution methodology proposed for concave processing rate functions in (Józefowska et al., 1999) consists in two steps. In the first step a feasible sequence of activity sets is constructed, and in the second step an optimal continuous resource allocation is calculated by solving a convex mathematical programming problem. The activity sets constructed in the first step consist of activities that may be processed in parallel without violating precedence or discrete resource constraints. A sequence of sets is feasible if each activity belongs to at least one set, and if any activity belongs to more than one set then the sets occur consecutively in the sequence (it is then guaranteed that all activities are scheduled without preemption). Unfortunately, in order to find an optimal schedule all feasible sequences have to be examined in general, and in consequence this approach is computationally inefficient. Moreover, for larger problem sizes, when the number of variables in the problem grows rapidly, solving the mathematical programming problem in the second step requires using specialized nonlinear solvers, and may take quite a long time. Therefore heuristic approaches to allocating the continuous resource are proposed in (Waligóra, 2010).

In (Józefowska et al., 2000) another heuristic approach is proposed, where continuous resource allotments are discretized. As a result, the classical discrete MRCPSP is obtained in a version without nonrenewable resources. The number of modes for the activities depends on the discretization level, i.e. the number of discretized allotments. It should be stressed that in this approach the processing rate functions of activities need not be concave, but may be arbitrary continuous, increasing functions. The resulting MRCPSP can be solved using one of the numerous existing methods, e.g. efficient metaheuristics. This will generate an approximate schedule for the original discrete–continuous resource-constrained project scheduling problem, however the calculations will be much less time consuming with no need for solving the nonlinear mathematical programming problem.

Thus, the importance of the discretization idea follows from the fact that it allows to convert a discrete–continuous scheduling problem into a purely discrete one. On the other hand, as mentioned above, it can be sometimes useful to continuize a discrete resource, in order to take advantage of the processing rate vs. resource amount model and the results proved for it. Discretization and continuization are opposite approaches, and the decision on using them depends on the knowledge about the problem, and particularly, processing rate functions.

## 8. Preemptibility discussion

Although we basically restrict ourselves to nonpreemptable activities, in this section we present a short discussion on their preemptibility, as it is an important feature in some applications.

As mentioned in Section 3.2.8, in the classical models of project scheduling problems it is assumed that activities are nonpreemptable. However, in some project environments it might be possible that the nonpreemption assumption can be relaxed. If this is the case in the classical RCPSP, the resulting problem is then called *preemptive resource-constrained project scheduling problem* (PRCPSP).

It is a typical assumption of the PRCPSP that activities can only be preempted at integer time instants, and they can be restarted later at no additional cost. In other words, an activity $A_i$ of duration $d_i$ may be split in at most $d_i$ duration units. However, in the most general case, activities are continuously-preemptable. For example, in computing projects where activities are processes in a computer system, they can be preempted at arbitrary time moments. Since the analyses of the problem with discretely- and continuously-preemptable activities are very much different, the next two subsections will be devoted to each of the two cases separately.

### 8.1. Discretely-preemptable activities

The classical PRCPSP is extensively analyzed in (Demeulemeester and Herroelen, 1996), where the problem formulation, as well as a B&B exact algorithm to solve it are presented. A numerical example for the algorithm is provided, and some computational results are reported. The authors conclude that the introduction of preemption to the RCPSP has little effect on the project makespan when constant resource availabilities are defined. The only exception is a problem with variable resource availability levels. This conclusion caused that the PRCPSP had not been further analyzed for several years, although various project scheduling problems with discretely-preemptable activities were considered in several papers (Bianco et al., 1999; Valls et al., 1999; Schwindt and Trautmann, 2000; Brucker and Knust, 2001; Franck et al., 2001). Demeulemeester and Herroelen (2002) extensively discuss the PRCPSP and recall the results from (1996).

However, Ballestín et al. (2008) resume the analysis of the PRCPSP. They base on the observation given in (Kolisch et al., 1995) that the Patterson instance set (Patterson, 1984), used for experiments performed by Demeulemeester and Herroelen (1996), suffers from two drawbacks which might affect the obtained results. For this reason, the authors put in doubt the conclusion stated by Demeulemeester and Herroelen (1996, 2002), and they reevaluate the usefulness of preemption for decreasing the project length in the RCPSP, but they limit the number of preemptions to a maximum of one per activity. The resulting problem is called 1_PRCPSP, as a special case of the $m$_PRCPSP where activities are allowed to be preempted at most $m$ times at any integer time instant. The authors show in the paper how three basic elements of many heuristics for the RCPSP – codification, serial SGS decoding rule, and double justification – can be adapted to deal with preemption. Computational experiments performed on the standard PSPLIB instances with 30 and 120 activities support the conclusion that preemption does help to decrease project length when compared to the nonpreemptive case, even assuming at most one preemption per activity.

The same authors in (Ballestín et al., 2009) further investigate the PRCPSP. They propose a new model that covers most practical applications of discrete activity preemption. The problem introduced is called Maxnint_PRCPSP, in which activity $A_i$ can be preempted a given maximum number of $maxnint_i$ of times. The authors develop an evolutionary metaheuristic for solving the problem. Computational experiments performed the on standard PSPLIB instances with 30 and 120 activities study the difference in makespan between allowing $m$ preemptions per activity, $m = 0, 1, 2, \ldots$. In the second part of the paper the usefulness of preemption in the presence of due dates is analyzed.

Vanhoucke and Debels (2008) investigate the impact of within-activity fast tracking on the PRCSPS, which allows the execution of preemptive subparts of an activity in parallel. The fast tracking option removes precedence relations between subparts of preempted activities and increases the number of execution scenarios. The within-activity fast tracking option is inspired by the idea that activities are often executed by groups of resources (with a fixed availability), but the total work can often be done by multiple groups (in parallel). The authors define the resulting problem as the preemptive resource-constrained project scheduling problem with fast tracking (PRCPSP-FT).

All the papers mentioned above tackle single-mode problems. To the best of our knowledge, the multi-mode version of the PRCPSP has only been considered in four papers.

Nudtasomboon and Randhawa (1997) were the first to include activity preemption into a multi-mode RCPSP. They assume that activities may be preempted an arbitrary number of times at integer time instants. The main contribution of the paper is the formulation of a zero-one integer programming model of the multi-mode RCPSP, which includes many important characteristics of project scheduling: activity preemption, renewable and nonrenewable resources, variation in resource availability, time/cost and time/resource trade-offs, and multiple objectives. Solution algorithms are presented and evaluated for three single-objective problems: with makespan minimization, cost minimization, and resource levelling, as well as for a multi-objective problem combining those three criteria.

Prashant Reddy et al. (2001) study a version of the problem with renewable resources only. The paper addresses the use of a Petri net as a modelling and scheduling tool in this context. The benefits of Petri nets in project scheduling are discussed. The authors propose some extensions of Petri nets to suit scheduling of activities in a decision CPM. They also propose the use of a

P-matrix for token movements in Petri nets. A genetic algorithm is used to find a better solution. Petri-net-aided software including genetic algorithm based search and heuristics is described to deal with a multi-mode, multi-constrained scheduling problem with preemption of activities.

In (Buddhakulsomsiri and Kim, 2006) the authors introduced a new model of the MRCPSP in which activities can be preempted under situations where resources may be temporarily not available. All resources considered are renewable, and each resource unit may not be available at all times due to resource vacations which are known in advance, and assignments to other finite duration activities. Activities can only be preempted at discrete points in time, and mode switching is not allowed when activities are resumed after preemption. A designed experiment is conducted that investigates project makespan improvement when activity preemption is permitted in various project scenarios, where different project scenarios are defined by parameters that have been used in the literature. A B&B procedure is applied to solve a number of small project scheduling problems with and without activity preemption. The results show that, in the presence of resource vacations and temporary resource unavailability, activity preemption can significantly improve the optimal project makespan in many scenarios, especially when resources are tight. The results also show that the makespan improvement is primarily dependent on those parameters that impact resource utilization.

In the follow-on paper (Buddhakulsomsiri and Kim, 2007) the authors develop a heuristic approach to the preemptive MRCPSP mentioned above. A new concept, called moving resource strength, is developed to help identify project situations where activity preemption is likely to be beneficial during scheduling. The moving resource strength concept is implemented in priority rule-based heuristics to control activity preemption when scheduling. Multiple comparisons of the performance of combination of activity-mode priority rules used in the heuristics are provided. Computational experiments demonstrate the effectiveness of the heuristic in reducing project makespan, and minimizing the number of preemptions.

To the best of our knowledge, the most general case of the preemptive MRCPSP in which mode switching is allowed after preemption, has not been considered in any paper up to now.

### *8.2. Continuously-preemptable activities*

The results obtained under this assumption for the processing rate vs. resource amount activity model, and continuous or discrete and continuous resources have been already reported in Section 7.

When only discrete resources are concerned, a so-called one-phase method is proposed in (Węglarz et al., 1977). This method was originally elaborated for the case of discrete and renewable resources, preemptable and dependent activities executed in single modes, and the minimization of project duration. It was shown that the optimal solution of the problem can be obtained by solving an appropriate linear programming (LP) problem. A specialized, automatic revised simplex method (ARSME) is presented in (Węglarz et al., 1977) to solve the problem. The method was basically developed for the single-mode case but its idea can be extended to multiple processing modes as well. An important assumption for the ARSME method is that a feasible ordering of nodes in the AoA precedence graph is given, i.e. such an ordering in which node $i$ occurs not later than node $j$, if $i < j$. Obviously, a feasible ordering of nodes can be found in $O(n^2)$ steps, but it is not unique except for a uniconnected AoA network. In general, an optimal schedule can be found by applying the ARSME method to every feasible ordering of nodes in an AoA network. Since the number of such orderings grows exponentially with the number of activities, an efficient heuristic called NODORD was presented in (Słowiński, 1978) for finding a node ordering. The one-phase method is next generalized in (Słowiński, 1980) for many other project scheduling problems, including multiple processing modes of activities. It is worth emphasizing that under this approach, as opposed to the cases discussed in Section 8.1, mode switching is allowed after preemption, more precisely, each activity may use each of its processing modes at any time of its execution.

Simultaneously, another approach, called two-phase method, was developed which allows to find an optimal schedule for a given feasible ordering of nodes in the precedence AoA graph. In the first phase a linear programming (LP) problem is solved, while in the second phase an optimal schedule is constructed using the solution of the LP problem. The main difference between these two approaches is the transformation into an LP problem, which is polynomial in the case of the two-phase method and exponential in the one-phase approach. It follows from the fact that in the one-phase approach variables denote processing times of resource-feasible subsets of activities, whereas in the two-phase approach variables are defined as parts of activities processed on particular machines. The two-phase method, moreover, deals with 0–1 resource requirements. This method was formulated by Błażewicz et al. (1976a,b), and independently described in a work by Lawler and Labetoulle (1978). It is worth mentioning, that nearly 30 years later, an almost optimal heuristic for the problem of scheduling preemptable, dependent tasks on parallel, identical machines to minimize the makespan was presented in (Józefowska et al., 2004). In the proposed approach a specialized heuristic, called H-LEV-NOD and based on the NODORD algorithm, is applied to find a node ordering in an AoA graph. Then the two-phase method is used to construct an optimal schedule for a given node ordering. A computational experiment performed on 35722 problem instances shows that the presented heuristic finds optimal solutions in 99.9% of instances.

The preemptive (and also nonpreemptive) RCPSP with continuously-preemptable activities was tackled in (Damay et al., 2007) where algorithms based on linear programming are developed for the problem. In the presented model each variable is associated to a subset of independent activities (antichains). The properties of the model are first investigated, in particular, conditions are given that allow a solution of the linear program to be a feasible schedule. From these properties, an algorithm based on neighbourhood search is derived. One neighbour solution is obtained through one Simplex pivoting, if this pivoting preserves feasibility. Methods to get out of local minima are provided. The solving methods are tested on the PSPLIB instances in a preemptive setting and prove efficient. They are used when preemption is forbidden with less success, and this difference is discussed. Some concluding remarks are given.

## 9. Conclusions and future work

The main goal of this paper has been to review the results across project scheduling problems which have not been comprehensively discussed yet within a common terminological and methodological framework. An important effect of such a survey is the exposure of problems which are worth to become a subject of future research. Below we present some conclusions and possible directions for future work.

Let us firstly notice that problems in which resources of categories different from at least one point of view, i.e. resource constraints, divisibility, or preemptibility (see Section 3.1), are simultaneously required to process project activities, have not been extensively studied within the project scheduling framework up to now. The only exception is the MRCPSP in which renewable, nonrenewable, and doubly constrained resources occur simultaneously, but this is just the case where only discrete resources are involved. Project scheduling problems under both discrete and continuous resources have only been considered in a few papers listed in Section 7. However, problems in which activities simultaneously require preemptable and nonpreemptable resources have not yet been studied in the project scheduling literature at all. As it is well-known from the computer resource management or operating systems theory, where such situations are commonly met, the main problem defined in such systems is coping with deadlocks. Still, optimization of the resource allocation in deadlock-free schedules is also a challenging problem. Moreover, under nonpreemptable resources, preemption is no longer just a feature of a set of activities, but we may talk about preemptive or nonpreemptive schedules. This issue has not been addressed in the project scheduling literature, either.

Let us now pass to some conclusions and directions for future research concerning the multi-mode problems considered in this survey.

In the context of the trade-off problems, discussed in Section 4, the research area is still quite open. As mentioned in Section 4.1.2, very few papers have dealt with the DTCTP under financial objectives, especially the NPV maximization. The DTCTP with discounted cash flows, as a combination of the RCPSPDCF and the DTCTP problems, seems, however, to be a very interesting problem to study, taking into account the combination between the time/cost trade-off and the cash flows occurring in a project. Furthermore, the DTRTP has also been considered in few papers, as indicated in Section 4.2.1. A tabu search, a genetic algorithm, and a hybrid scatter search/path relinking method are the only three approaches to the DTRTP up to now. Besides, the last approach is applied to the DTRTP with multiple resources (MDTRTP) which, in fact, becomes a special case of the MRCPSP with the absence of nonrenewable resources, and thus, it can be attacked by several existing efficient methods devoted to the general MRCPSP.

The MRCPSP is undoubtedly the most extensively studied problem from among all the problems considered in this survey. Many heuristic and several exact approaches, as well as lower bounds, for the basic formulation, some special cases, and various extensions have been developed. However, it does not exclude further research in this area. One of possible directions for future work is to develop new (or to improve the existing) exact or heuristic algorithms which will perform faster and/or give better results. Maybe it is also possible to improve algorithms computing the lower bounds. Another interesting question worth to be examined is: which project parameters, which of their settings, and in which way may cause an instance of the problem to be easy or difficult to solve optimally. This knowledge may by further used to develop better algorithms. Moreover, new extensions and special cases of the basic MRCPSP formulation which reflect real-life situations can be considered, e.g. by incorporating some special resource categories discussed in Section 3.1.3.2.

Finally, we feel the need to make a comment on computational experiments described in the MRCPSP literature. The current state is that no common standard has been elaborated indicating how computational experiments should be performed, which data sets should be used, which measures should be considered, which other results and how should be used for comparison, etc. Thus, in our opinion, there is a need to establish a set of rules according to which computational experiments should be carried out. This would increase the reliability of the obtained results. A good idea would be to collect all the data sets and data generators used in the literature at one site. The data files should be coded in one standard and extensible format, which does not depend on the software and hardware environment. The XML format seems to be a good choice in this case, which may make the data files to be more readable, easy to understand, and to extend with additional data. Moreover, it would also be useful to build a repository of algorithms that could be used for comparison with newly developed approaches. Then, these algorithms could be used in computational experiments performed in the same environment. Obviously, such algorithms should also be coded in a standard language that does not depend on the software and hardware environment (i.e. Java or ANSI C).

As far as the MRCPSPDCF is concerned, let us stress that there are still very few approaches to the problem. As it has been pointed out in Section 6.1.1, genetic algorithms, tabu search and simulated annealing, as well as ant colony optimization, are the only approaches known form the literature to the general MRCPSPDCF. More importantly, no exact procedure for the MRCPSPDCF has been proposed up to now, whereas at least a modification of one of the B&B approaches to the MRCPSP comes to one's mind. Also the multi-mode version of the payment scheduling problem (PSP) has only been considered in just three papers (see Section 6.1.3). Thus, the research area concerning problems with multiple processing modes and financial objectives seems still to be wide open. The same applies to multi-mode problems with other objectives, e.g. resource-based objectives which have only been tacked in three papers, too (Section 6.2).

Possible future works in the field of the preemptive MRCPSP are also worth mentioning. As it has been said in Section 8.1, this problem has only been studied in four papers. Moreover, in all of the papers it is assumed that mode switching is not allowed after preemption, i.e. each activity has to be continued using the previously chosen processing mode when it is resumed after preemption. Let us stress that, generally, this assumption does not have to hold, which leads to the most general case of the preemptive MRCPSP where each activity may use each of its processing modes at any time of its execution, as it has been showed in Section 8.2 for continuously-preemptable activities. Besides, let us notice that the PRCPSP and its multi-mode version have only been analyzed with the makespan minimization objective in all papers but one, pointed out in Section 8.1. Analyses of the preemptive project scheduling problems under other objectives are certainly another direction for further research.

Analyses of discrete–continuous project scheduling problems discussed in Section 7 can also be extended in several directions. These include extensions of the considered class of the problems encompassing more general classes of (i) activities – by taking into account, e.g. ready times and due dates (or deadlines), setup times, or generalized precedence constraints, (ii) discrete resources – by including, e.g. nonrenewable, variable in time, or partially renewable resources, (iii) continuous resources – by studying doubly constrained resources, or adding continuous resources of other categories, and (iv) criteria – by considering many other time-based, resource-based, or financial objectives, from among which earliness and tardiness costs, mean flow time, or resource levelling seem to have wide practical applications. Other directions for further research in the context of discrete–continuous project scheduling include analysis of problems with multiple modes where, e.g. processing demands and processing rate functions of activities depend on their discrete resource requirements, as well as problems with both positive and negative cash flows under the NPV maximization.

## Acknowledgments

This work has been funded by the Polish Ministry of Science and Higher Education as a research project in the years 2009–2012, Grant No. N N519 403437.

We express our gratitude to two anonymous referees for their valuable comments.

## References


Ahn, T., Erenguc, S.S., 1998. The resource-constrained project scheduling problem with multiple crashable modes: a heuristic procedure. European Journal of Operational Research 107 (2), 250–259.

Akkan, C., Drexl, A., Kimms, A., 2005. Network decomposition-based benchmark results for the discrete time-cost trade-off problem. European Journal of Operational Research 165 (2), 339–358.

Alcaraz, J., Maroto, C., Ruiz, R., 2003. Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms. Journal of the Operational Research Society 54 (6), 614–626.

Allahverdi, A., Gupta, J.N.D., Aldowaisan, T., 1999. A review of scheduling research involving setup considerations. OMEGA The International Journal of Management Science 27 (2), 219–239.

Allahverdi, A., Ng, C.T., Cheng, T.C.E., Kovalyov, M.Y., 2008. A survey of scheduling problems with setup times or costs. European Journal of Operational Research 187 (3), 985–1032.

Artigues, C., 2005. The resource-constrained project scheduling problem. In: Artigues, C., Demassey, S., Néron, E. (Eds.), Resource-constrained project scheduling: models, algorithms, extensions and applications. ISTE-Wiley, London, pp. 21–35.

Artigues, C., Roubellat, F., 2000. A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and multiple modes. European Journal of Operational Research 127 (2), 297–316.

Artigues, C., Roubellat, F., 2001. A Petri net model and a general method for on and off-line multi-resource shop floor scheduling with setup times. International Journal of Production Economics 74 (1-3), 63–75.

Ballestín, F., Valls, V., Quintanilla, S., 2008. Pre-emption in resource-constrained project scheduling. European Journal of Operational Research 189 (3), 1136–1152.

Ballestín, F., Valls, V., Quintanilla, S., 2009. Scheduling projects with limited number of preemptions. Computers and Operations Research 36 (11), 2913–2925.

Ballestín, F., Blanco, R., 2010. Theoretical and practical fundamentals for multi-objective optimisation in resource-constrained project scheduling problems. Computers and Operations Research, doi:10.1016/j.cor.2010.02.004.

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.

Baptiste, P., Le Pape, C., Nuijten, W., 2001. Constraint-based Scheduling: Applying Constraint Programming to Scheduling Problems. Kluwer, Norwell.

Barrios, A., Ballestín, F., Valls, V., 2009. A double genetic algorithm for the MRCPSP/max. Computers and Operations Research, doi: 10.1016/j.cor.2009.09.019.

Bartels, J.H., Zimmermann, J., 2009. Scheduling tests in automotive R&D projects. European Journal of Operational Research 193 (3), 805–819.

Bedworth, D.D., 1973. Industrial Systems: Planning, Analysis, and Control. The Ronald Press, New York.

Bellenguez, O., Néron, E., 2005. Lower bounds for the multi-skill project scheduling problem with hierarchical level of skills. In: Burke, E.K., Trick, M. (Eds.), Lecture Notes in Computer Science (3616): Practice and Theory of Automated Timetabling V. Springer, Berlin, pp. 229–243.

Bellenguez-Morineau, O., 2008. Methods to solve multi-skill project scheduling problem. 4OR: A Quarterly Journal of Operations Research 6 (1), 85–88.

Bellenguez-Morineau, O., Néron, E., 2008. Multi-mode and multi-skill project scheduling problem. In: Artigues, C., Demassey, S., Néron, E. (Eds.), Resource-Constrained Project Scheduling: Models, Algorithms, Extensions and Applications. ISTE-Wiley, London, pp. 149–160.

Bianco, L., Caramia, M., Dell'Olmo, P., 1999. Solving a preemptive project scheduling problem with coloring techniques. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms, and Applications. Kluwer Academic Publishers, Boston, pp. 135–145.

Bianco, L., Dell'Olmo, P., Speranza, M.G., 1998. Heuristics for multi-mode scheduling problems with dedicated resources. European Journal of Operational Research 107 (2), 260–271.

Billaut, J.C., Moukrim, A., Sanlaville, E., 2008. Flexibility and Robustness in Scheduling. ISTE-Wiley, London.

Błażewicz, J., Cellary, W., Słowiński, R., Węglarz, J., 1976a. Deterministyczne problemy szeregowania zadań na równoległych procesorach. Część I – Zbiory zadań niezależnych. Podstawy Sterowania 6 (2), 155–178.

Błażewicz, J., Cellary, W., Słowiński, R., Węglarz, J., 1976b. Deterministyczne problemy szeregowania zadań na równoległych procesorach. Część II – Zbiory zadań zależnych. Podstawy Sterowania 6 (3), 297–320.

Błażewicz, J., Cellary, W., Słowiński, R., Węglarz, J., 1986. Scheduling Under Resource Constraints: Deterministic Models. J.C. Baltzer AG Science Publishers, Basel.

Błażewicz, J., Ecker, K., Pesch, E., Schmidt, G., Węglarz, J., 2007. Handbook on Scheduling: From Theory to Applications. Springer Verlag, Berlin.

Błażewicz, J., Lenstra, J.K., Rinnooy Kan, A.H.G., 1983. Scheduling subject to resource constraints: classification and complexity. Discrete Applied Mathematics 5 (1), 11–24.

Boctor, F.F., 1993. Heuristics for scheduling projects with resource restrictions and several resource-duration modes. International Journal of Production Research 31 (11), 2547–2558.

Boctor, F.F., 1996a. A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes. European Journal of Operational Research 90 (2), 349–361.

Boctor, F.F., 1996b. Resource-constrained project scheduling by simulated annealing. International Journal of Production Research 34 (8), 2335–2351.

de Boer, R., Resource-constrained multi-project management, a hierarchical decision support system. Ph.D. Thesis, University of Twente, 1998.

Boschetti, M., Maniezzo, V., 2009. Benders decomposition, Lagrangean relaxation and metaheuristic design. Journal of Heuristics 15 (3), 283–312.

Böttcher, J., Drexl, A., Kolisch, R., Salewski, F., 1999. Project scheduling under partially renewable resource constraints. Management Science 45 (4), 543–559.

Bouleimen, K., Lecocq, H., 2003. A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version. European Journal of Operational Research 149 (2), 268–281.

Brinkman, K., Neumann, K., 1996. Heuristic procedures for resource-constrained project scheduling with minimal and maximal time lags: the minimum project duration and resource leveling problem. Journal of Decision Systems 5, 129–156.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E., 1999. Resource-constrained project scheduling: notation, classification, models and methods. European Journal of Operational Research 112 (1), 3–41.

Brucker, P., Knust, S., 2001. Resource-constrained project scheduling and timetabling. Lecture Notes in Computer Science 2079, 277–293.

Brucker, P., Knust, S., 2003. Lower bounds for resource-constrained project scheduling problems. European Journal of Operational Research 149 (2), 302–313.

Buddhakulsomsiri, J., Kim, D.S., 2006. Properties of multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting. European Journal of Operational Research 175 (1), 279–295.

Buddhakulsomsiri, J., Kim, D.S., 2007. Priority rule-based heuristic for multi-mode resource-constrained project scheduling problems with resource vacations and activity splitting. European Journal of Operational Research 178 (2), 374–390.

Burgess, A.R., Killebrew, J.B., 1962. Variation in activity level on a cyclic arrow diagram. Journal of Industrial Engineering 13 (2), 76–83.

Calhoun, K.M., Deckro, R.F., Moore, J.T., Chrissis, J.W., Van Hove, J.C., 2002. Planning and re-planning in project and production scheduling. Omega 30 (3), 155–170.

Chen, W.N., Zhang, J., Chung, H.S.H., Huang, R.Z., Liu, O., 2010. Optimizing discounted cash flows in project scheduling – an ant colony optimization approach. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 40 (1), 64–77.

Chiang, C.W., Huang, Y.Q., Wang, W.Y., 2008. Ant colony optimization with parameter adaptation for multi-mode resource-constrained project scheduling. Journal of Intelligent and Fuzzy Systems 29 (4–5), 345–358.

Christofides, N., Alvarez-Valdes, R., Tamarit, J.M., 1987. Project scheduling with resource constraints: a branch-and-bound approach. European Journal of Operational Research 29 (3), 262–273.

Chyu, C.C., Chen, A.H.L., Lin, X.H., 2005. A hybrid ant colony approach to multi-mode resource-constrained project scheduling problems with nonrenewable types. In: Proceedings of First International Conference on Operations and Supply Chain Management, Bali, 2005.

Coffman Jr., E.G., Denning, P.J., 1973. Operating Systems Theory. Prentice-Hall, Englewood Cliffs, NJ.

Damak, J., Jarboui, B., Siarry, P., Loukil, T., 2009. Differential evolution for solving multi-mode resource-constrained project scheduling problems. Computers and Operations Research 36 (9), 2653–2659.

Damay, J., Quilliot, A., Sanlaville, E., 2007. Linear programming based algorithms for preemptive and nonpreemptive RCPSP. European Journal of Operational Research 182 (3), 1012–1022.

Dayanand, N., Padman, R., 1997. On modeling progress payments in project networks. Journal of the Operational Research Society 48 (9), 906–918.

De, P., Dunne, E.J., Gosh, J.B., Wells, C.E., 1995. The discrete time-cost trade-off problem revisited. European Journal of Operational Research 81 (2), 225–238.

Demeulemeester, E.L., Dodin, B., Herroelen, W.S., 1993. A random network activity generator. Operations Research 41 (5), 972–980.

Demeulemeester, E.L., Herroelen, W.S., 1992. A branch-and-bound procedure for the multiple resource-constrained project scheduling problem. Management Science 38 (12), 1803–1818.

Demeulemeester, E.L., Herroelen, W.S., 1996. An efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem. European Journal of Operational Research 90 (2), 334–348.

Demeulemeester, E.L., Herroelen, W.S., 2002. Project Scheduling – A Research Handbook. Kluwer Academic Publishers, Boston.

Demeulemeester, E., De Reyck, B., Herroelen, W., 2000. The discrete time/resource trade-off problem in project networks: a branch-and-bound approach. IIE Transactions 32 (11), 1059–1069.

De Reyck, B., Demeulemeester, E., Herroelen, W., 1998. Local search methods for the discrete time/resource trade-off problem in project networks. Naval Research Logistic Quarterly 45 (6), 553–578.

De Reyck, B., Herroelen, W.S., 1998. A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 111 (1), 152–174.

De Reyck, B., Herroelen, W.S., 1999. The multi-mode resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 119 (2), 538–556.

Dornorf, U., 2002. Project Scheduling with Time Windows: From Theory to Applications. Physica, Heidelberg.

Drexl, A., Grünewald, J., 1993. Nonpreemptive multi-mode resource-constrained project scheduling. IIE Transactions 25 (5), 74–81.
Drezet, L.E., 2008. RCPSP with financial costs. In: Artigues, C., Demassey, S., Néron, E. (Eds.), Resource-Constrained Project Scheduling: Models, Algorithms, Extensions and Applications. ISTE-Wiley, London, pp. 213–226.
Duin, C.W., Van Der Sluis, E., 2006. On the complexity of adjacent resource scheduling. Journal of Scheduling 9 (1), 49–62.
Elloumi, S., Fortemps, P., Teghem, J., Loukil, T., 2006. A new bi-objective algorithm using clustering heuristics to solve the multi-mode resource-constrained project scheduling problem. In: Proceedings of the 25th workshop of the UK Planning and Scheduling Special Interest Group (PlanSIG 2006), Nottingham, UK, p. 113–120.
Elmaghraby, S.E., 1977. Activity Networks: Project Planning and Control by Network Models. Wiley, New York.
Elmaghraby, S.E., 1995. Activity nets: a guided tour through some recent developments. European Journal of Operational Research 82 (3), 383–408.
Elmaghraby, S.E., Kamburowski, J., 1989. The analysis of activity networks under generalized precedence relations (GPR). Parts I and II. OR Reports No. 231 and 232, NC State University, Raleigh NC.
Elmaghraby, S.E., Kamburowski, J., 1992. The analysis of activity networks under generalized precedence relations (GPRs). Management Science 38 (9), 1245–1263.
Erenguc, S.S., Ahn, T., Conway, D.G., 2001. The resource constrained project scheduling problem with multiple crashable modes: an exact solution method. Naval Research Logistics 48 (2), 107–127.
Erenguc, S.S., Tufekci, S., Zappe, C.J., 1993. Solving time/cost trade-off problems with discounted cash flows using generalized Benders decomposition. Naval Research Logistics 40 (1), 25–50.
Fischetti, M., Lodi, A., 2003. Local branching. Mathematical Programming 98 (1–3), 23–47.
Fondahl, J.W., 1962. A noncomputer approach to the critical path method for the construction industry, second ed., Department of Civil Engineering, Stanford University, Stanford, California.
Franck, B., Neumann, K., Schwindt, C., 2001. Project scheduling with calendars. OR Spektrum 23 (3), 325–334.
Gagnon, M., Boctor, F.F., d'Avignon, G., 2005. A tabu search algorithm for the multiple mode resource constrained project scheduling problem. In: Proceedings of the ASAC 2005, Toronto, Canada, pp. 89–102.
Hadjiconstantinou, E., Klerides, E., 2009. A new path-based cutting plane approach for the discrete time-cost trade-off problem. Computational Management Science, doi: 10.1007/s10287-009-0115-6.
Hartmann, S., 1998. A competitive genetic algorithm for resource-constrained project scheduling. Naval Research Logistics 45 (7), 733–750.
Hartmann, S., 2001. Project scheduling with multiple modes: a genetic algorithm. Annals of Operations Research 102 (1–4), 111–135.
Hartmann, S., Drexl, A., 1998. Project scheduling with multiple modes: a comparison of exact algorithms. Networks 32 (4), 283–297.
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 (2), 394–407.
Hartmann, S., Sprecher, A., 1996. A note on ''Hierarchical models for multi-project planning and scheduling''. European Journal of Operational Research 94 (2), 377–383.
Hazir, Ö, Haouari, M., Erel, E., 2010. Discrete time/cost trade-off problem: a decomposition-based solution algorithm for the budget version. Computers and Operations Research 37 (4), 649–655.
He, Z., Wang, N., Jia, T., Xu, Y., 2009. Simulated annealing and tabu search for multi-mode project payment scheduling. European Journal of Operational Research 198 (3), 688–696.
He, Z., Xu, Y., 2008. Multi-mode project payment scheduling problems with bonus-penalty structure. European Journal of Operational Research 189 (3), 1191–1207.
Heilmann, R., 2001. Resource-constrained project scheduling: a heuristic for the multi-mode case. OR Spektrum 23 (3), 335–357.
Heilmann, R., 2003. 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 144 (2), 348–365.
Herroelen, W.S., De Reyck, B., Demeulemeester, E.L., 1998. Resource-constrained project scheduling: a survey of recent developments. Computers and Operations Research 25 (4), 279–302.
Herroelen, W.S., Leus, R., 2004. Robust and reactive project scheduling: a review and classification of procedures. International Journal of Production Research 42 (8), 1599–1620.
Herroelen, W.S., Leus, R., 2005. Project scheduling under uncertainty, survey and research potentials. European Journal of Operational Research 65 (2), 289–306.
Herroelen, W.S., Van Dommelen, P., Demeulemeester, E.L., 1997. Project network models with discounted cash flows: a guided tour through recent developments. European Journal of Operational Research 100 (1), 97–121.
Hsu, C.C., Kim, D.S., 2005. A new heuristic for the multi-mode resource investment problem. Journal of the Operational Research Society 56 (4), 406–413.
Icmeli, O., Erenguc, S.S., Zappe, C.J., 1993. Project scheduling problems: a survey. International Journal of Operations and Production Management 13 (11), 80–91.
Icmeli, O., Erenguc, S.S., 1996. The resource constrained time/cost trade-off project scheduling problem with discounted cash flows. Journal of Operations Management 14 (3), 255–275.
Icmeli-Tukel, O., Rom, W.O., 1997. Ensuring quality in resource constrained project scheduling. European Journal of Operational Research 103 (3), 483–496.
ILOG. ILOG CPLEX 7.5: Reference Manual. ILOG Inc.: Mountain View; 2002.
Janiak, A., 1991a. Scheduling a resource allocation problems in some flow type manufacturing processes. In: Fandel, G., Zapfel, G. (Eds.), Modern Production Concepts. Springer, Berlin, pp. 404–415.
Janiak, A., 1991b. Single machine scheduling problem with a common deadline and resource dependent release dates. European Journal of Operational Research 53 (3), 317–325.
Jarboui, B., Damak, N., Siarry, P., Rebai, A., 2008. A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems. Applied Mathematics and Computation 195 (1), 299–308.
Jędrzejowicz, P., Ratajczak, E., 2006. Population learning algorithm for the resource-constrained project scheduling. In: Józefowska, J., Węglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Springer, Berlin, pp. 275–296.
Józefowska, J., Mika, M., Różycki, R., Waligóra, G., Węglarz, J., 1999. Project scheduling under discrete and continuous resources. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms, and Applications. Kluwer Academic Publishers, Boston, pp. 289–308.
Józefowska, J., Mika, M., Różycki, R., Waligóra, G., Węglarz, J., 2000. Solving the discrete-continuous project scheduling problem via its discretization. Mathematical Methods of Operations Research 52 (3), 489–499.
Józefowska, J., Mika, M., Różycki, R., Waligóra, G., Węglarz, J., 2001. Simulated annealing for multi-mode resource-constrained project scheduling. Annals of Operations Research 102 (1–4), 137–155.
Józefowska, J., Mika, M., Różycki, R., Waligóra, G., Węglarz, J., 2004. An almost optimal heuristic for preemptive $C_{max}$ scheduling of dependent tasks on parallel identical machines. Annals of Operations Research 129 (1–4), 205–216.
Józefowska, J., Węglarz, J., 1998. On a methodology for discrete-continuous scheduling problems. European Journal of Operational Research 107 (2), 338–353.
Kavlak, N., Ulusoy, G., Sivrikaya-Şerifoğlu, F., Birbil, I., 2009. Client-contractor bargaining on net present value in project scheduling with limited resources. Naval Research Logistics 56 (2), 93–112.
Kelley Jr., J.E., 1963. The critical path method: resources planning and scheduling. In: Muth, J.F., Thompson, G.L. (Eds.), Industrial Scheduling. Prentice-Hall, New Jersey, pp. 347–365.
Kelley, J.E., Jr., Walker, M.R., 1959. Critical path planning and scheduling. In: Proceedings of the Eastern Joint Computer Conference, Boston, MA, pp. 160–173.
Kimms, A., 2001. Mathematical Programming and Financial Objectives for Scheduling Projects. Kluwer Academic Publishers, Dordrecht.
Kis, T., 2005. A branch-and-cut algorithm for scheduling of projects with variable-intensity activities. Mathematical Programming 103 (3), 139–515.
Kis, T., 2006. RCPS with variable intensity activities and feeding precedence constraints. In: Józefowska, J., Węglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Springer, Berlin, pp. 105–129.
Knotts, G., Dror, M., Hartman, B.C., 2000. Agent-based project scheduling. IIE Transactions 32 (5), 387–401.
Knust, S., 2010. Scheduling non-professional table-tennis leagues. European Journal of Operational Research 200 (2), 358–367.
Knust, S., von Thaden, M., 2006. Balanced home-away assignments. Discrete Optimization 3 (4), 354–365.
Kolisch, R., 1995. Project Scheduling under Resource Constraints: Efficient Heuristics for Several Problem Classes. Physisca-Verlag, Heidelberg.
Kolisch, R., 1996. Serial and parallel resource-constrained project scheduling methods revisited: theory and computation. European Journal of Operational Research 90 (2), 320–333.
Kolisch, R., Drexl, A., 1997. Local search for nonpreemptive multi-mode resource-constrained project scheduling. IIE Transactions 29 (11), 987–999.
Kolisch, R., Hartmann, S., 1999. Heuristic algorithms for the resource-constrained project scheduling problem: classification and computational analysis. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms, and Applications. Kluwer Academic Publishers, Boston, pp. 147–178.
Kolisch, R., Hartmann, S., 2006. Experimental investigation of heuristics for resource-constrained project scheduling: an update. European Journal of Operational Research 174 (1), 23–37.
Kolisch, R., Padman, R., 2001. An integrated survey of deterministic project scheduling. OMEGA The International Journal of Management Science 29 (3), 249–272.
Kolisch, R., Sprecher, A., 1997. PSPLIB – A project scheduling problem library. European Journal of Operational Research 96 (1), 205–216.
Kolisch, R., Sprecher, A., Drexl, A., 1995. Characterization and generation of a general class of resource-constrained project scheduling problems. Management Science 41 (10), 1693–1703.
Krishnamoorthy, M., Deo, N., 1979. Complexity of the minimum-dummy-activities problem in a PERT network. Networks 9 (3), 189–194.
Lawler, E.L., Labetoulle, J., 1978. On preemptive scheduling of unrelated parallel processors by linear programming. Journal of the ACM 25 (4), 612–619.
Leachman, R.C., 1983. Multiple resource leveling in construction systems through variation of activity intensities. Naval Research Logistics Quarterly 30 (3), 187–198.
Leachman, R.C., Dincerler, A., Kim, S., 1990. Resource-constrained scheduling of projects with variable-intensity activities. IIE Transactions 22 (1), 31–39.

Li, K.Y., Willis, R.J., 1992. An iterative scheduling technique for resource-constrained project scheduling. European Journal of Operational Research 56 (3), 370–379.
Li, H., Womer, K., 2008. Modeling the supply chain configuration problem with resource constraints. International Journal of Project Management 26 (6), 646–654.
Li, H., Womer, K., 2009. Scheduling projects with multi-skilled personnel by a hybrid MILP/CP Benders decomposition algorithm. Journal of Scheduling 12 (3), 281–298.
Liu, Z., Wang, H., 2006. Heuristic algorithm for the RCPSP with the objective of minimizing activities' cost. Journal of Systems Engineering and Electronics 17 (1), 96–102.
Lova, A., Tormos, P., Barber, F., 2006. Multi-mode resource constrained project scheduling: scheduling schemes, priority rules and mode selection rules. Inteligencia Artificial 10 (30), 69–86.
Lova, A., Tormos, P., Cervantes, M., Barber, F., 2009. An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. International Journal of Production Economics 117 (2), 302–316.
Malcolm, D.G., Roseboom, J.H., Clark, C.E., Fazar, W., 1959. Application of a technique for research and development program evaluation. Operations Research 7 (5), 646–669.
Maniezzo, V., Mingozzi, A., 1999. A heuristic procedure for the multi-mode project scheduling problem based on Bender's decomposition. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Dordrecht, pp. 179–196.
Maroto, C., Tormos, P., 1994. Project management: an evaluation of software quality. International Transactions in Operational Research 1 (2), 209–221.
Mika, M., Waligóra, G., Węglarz, J., 2005. Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. European Journal of Operational Research 164 (3), 639–668.
Mika, M., Waligóra, G., Węglarz, J., 2003. A metaheuristic approach to scheduling workflow jobs on a grid. In: Nabrzyski, J., Schopf, J.M., Węglarz, J. (Eds.), Grid Resource Management: State of the Art and Future Trends. Kluwer Academic Publishers, Boston, pp. 295–318.
Mika, M., Waligóra, G., Węglarz, J., 2006. Modelling setup times in project scheduling. In: Józefowska, J., Węglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Springer, Berlin, pp. 131–163.
Mika, M., Waligóra, G., Węglarz, J., 2008. Tabu search for multi-mode resource-constrained project scheduling with schedule-dependent setup times. European Journal of Operational Research 187 (3), 1238–1250.
Mika, M., Węglarz, J., 2004. Heurystyczne algorytmy równoważenia obcia¸żenia w problemach rozdziału zasobów z wieloma sposobami wykonywania czynności. In: Gessing, R., Szkodny, T. (Eds.), Automatyzacja Procesów Dyskretnych: Optymalizacja dyskretna; Robotyka i sterowniki programowalne, Wydawnictwa Naukowo Techniczne, Warszawa, pp. 161–168.
Mingozzi, A., Maniezzo, V., Ricciardelli, S., Bianco, L., 1998. An exact algorithm for project scheduling with resource constraints based on a new mathematical formulation. Management Science 44 (5), 714–729.
Moder, J.J., Phillips, C.R., Davis, E.W., 1983. Project Management with CPM, PERT and Precedence Diagramming, Van Nostrand Reinhold Company, New York.
Möhring, R.H., 1984. Minimizing costs of resource requirements in project networks subject to a fixed completion time. Operations Research 32 (1), 89–120.
Mori, M., Tseng, C.C., 1997. A genetic algorithm for multi-mode resource constrained project scheduling problem. European Journal of Operational Research 100 (1), 134–141.
Néron, E., 2002. Lower bounds for the multi-skill project scheduling problem. In: Proceedings of the Eighth International Workshop on Project Management and Scheduling (PMS2002), Valencia, Spain, pp. 274–277.
Neumann, K., Schwindt, C., Zimmermann, J., 2002. Project Scheduling with Time Windows and Scarce Resources: Temporal and Resource-Constrained Project Scheduling with Regular and Nonregular Objective Functions. Springer, Berlin.
Neumann, K., Schwindt, C., Zimmermann, J., 2006. Resource-constrained project scheduling with time windows. In: Józefowska, J., Węglarz, J. (Eds.), Perspectives in Modern Project Scheduling. Springer, Berlin, pp. 375–407.
Nonobe, K., Ibaraki, T., 2002. Formulation and tabu search algorithm for the resource constrained project scheduling problem. In: Ribeiro, C.C., Hansen, P. (Eds.), Essays and Surveys in Metaheuristics. Kluwer, Boston, pp. 557–588.
Nudtasomboon, N., Randhawa, S.U., 1997. Resource-constrained project scheduling with renewable and nonrenewable resources and time-resource trade-offs. Computers and Industrial Engineering 32 (1), 227–242.
Özdamar, L., 1998. On scheduling project activities with variable expenditure rates. IIE Transactions 30 (8), 695–704.
Özdamar, L., 1999. A genetic algorithm approach to a general category project scheduling problem. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 29 (1), 44–59.
Özdamar, L., Dündar, H., 1997. A flexible heuristic for a multi-mode capital constrained project scheduling problem with probabilistic cash inflows. Computers and Operations Research 24 (12), 1187–1200.
Özdamar, L., Ulusoy, G., 1994. A local constraint based analysis approach to project scheduling under general resource constraints. European Journal of Operational Research 79 (2), 287–298.
Özdamar, L., Ulusoy, G., 1995. A survey on the resource-constrained project scheduling problem. IIE Transactions 27 (5), 574–586.
Patterson, J.H., 1984. A comparison of exact procedures for solving the multiple constrained resource project scheduling problem. Management Science 30 (7), 854–867.
Patterson, J.H., Słowiński, R., Talbot, F.B., Węglarz, J., 1989. An algorithm for a general class of precedence and resource constrained scheduling problems. In: Słowiński, R., Węglarz, J. (Eds.), Advances in Project Scheduling. Elsevier, Amsterdam, pp. 3–28.
Patterson, J.H., Talbot, F.B., Słowiński, R., Węglarz, J., 1990. Computational experience with a backtracking algorithm for solving a general class of precedence and resource-constrained scheduling problems. European Journal of Operational Research 49 (1), 68–79.
Paulus, J.J., Hurink, J., 2006. Adjacent-resource scheduling: why spatial resources are so hard to incorporate. Electronic Notes in Discrete Mathematics 25, 113–116.
Pedersen, C.R., Rasmussen, R.V., Andersen, K.A., 2007. Solving a large-scale precedence constrained scheduling problem with elastic jobs using tabu search. Computers and Operations Research 34 (7), 2025–2042.
Peng, W.L., Wang, C.G., 2008. A multi-mode resource-constrained discrete time-cost trade-off problem and its genetic algorithm based solution. International Journal of Project Management 27 (6), 600–609.
Pesch, E., 1999. Lower bounds in different problem classes of project schedules with resource constraints. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Dordrecht, pp. 53–76.
Prashant Reddy, J., Kumanan, S., Krishnaiah Chetty, O.V., 2001. Application of Petri nets and a genetic algorithm to multi-mode multi-resource constrained project scheduling. International Journal of Advanced Manufacturing Technology 17 (4), 305–314.
Pritsker, A.A.B., Watters, L.J., Wolfe, P.M., 1969. Multiproject scheduling with limited resources: a zero-one programming approach. Management Science 16 (1), 93–107.
Ranjbar, M., Kianfar, F., 2007. Solving the discrete time/resource trade-off problem in project scheduling with genetic algorithms. Applied Mathematics and Computation 191 (2), 451–456.
Ranjbar, M., De Reyck, B., Kianfar, F., 2009. A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling. European Journal of Operational Research 193 (1), 35–48.
Roy, B., 1962. Graphes et ordonnancements. Revue Française de Recherche Opérationelle 25, 323–326.
Russell, A.H., 1970. Cash flows in networks. Management Science 16 (5), 357–373.
Sabzehparvar, M., Seyed-Hosseini, S.M., 2008. A mathematical model for the multi-mode resource-constrained project scheduling problem with mode-dependent time lags. Journal of Supercomputing 44 (3), 257–273.
Sabzehparvar, M., Seyed-Hosseini, S.M., Nouri, S., 2008. A mathematical model for the multi-mode resource investment problem. Journal of Industrial Engineering International 4 (7), 25–32.
Salewski, F., Schrimer, A., Drexl, A., 1997. Project scheduling under resource and mode identity constraints: model, complexity, methods, and application. European Journal of Operational Research 102 (1), 88–110.
Schultmann, F., Rentz, O., 2001. Environment-oriented project scheduling for the dismantling of buildings. OR Spektrum 23 (1), 51–78.
Schwindt, C., Trautmann, N., 2000. Batch scheduling in process industries: an application of resource-constrained project scheduling. OR Spektrum 22 (4), 501–524.
Schwindt, C., Trautmann, N., 2003. Scheduling the production of rolling ingots: industrial context, model, and solution method. International Transactions in Operational Research 10 (6), 547–563.
Seifi, M., Tavakkoli-Moghaddam, R., 2008. A new bi-objective model for a multi-mode resource-constrained project scheduling problem with discounted cash flows and four payments model. International Journal of Engineering, Transactions A: Basics 21 (4), 347–360.
Shewchuk, J.P., Chang, T.C., 1995. Resource-constrained job scheduling with recyclable resources. European Journal of Operational Research 81 (2), 364–375.
Słowiński, R., 1978. A node ordering heuristic for network scheduling under multiple resource constraints. Foundations of Control Engineering 3 (1), 19–27.
Słowiński, R., 1980. Two approaches to problems of resource allocation among project activities – a comparative study. Journal of the Operational Research Society 31 (8), 711–723.
Słowiński, R., 1981. Multi-objective network scheduling with efficient use of renewable and nonrenewable resources. European Journal of Operational Research 7 (3), 265–273.
Słowiński, R., 1989. Multi-objective project scheduling under multiple-category resource constraints. In: Słowiński, R., Węglarz, J. (Eds.), Advances in Project Scheduling. Elsevier, Amsterdam, pp. 151–167.
Speranza, M.G., Vercellis, C., 1993. Hierarchical models for multi-project planning and scheduling. European Journal of Operational Research 64 (2), 312–325.
Sprecher, A., 1994. Resource-constrained project scheduling: exact methods for the multi-mode case. Springer, Berlin.
Sprecher, A., Drexl, A., 1998. Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm. European Journal of Operational Research 107 (2), 431–450.
Sprecher, A., Hartmann, S., Drexl, A., 1997. An exact algorithm for project scheduling with multiple modes. OR Spektrum 19 (3), 195–203.
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 (1), 94–102.

Sung, C.S., Lim, S.K., 1994. A project activity scheduling problem with net present value measure. International Journal of Production Economics 37 (2–3), 177–187.

Talbot, F.B., 1982. Resource-constrained project scheduling with time-resource trade-offs: the nonpreemptive case. Management Science 28 (10), 1197–1210.

Tareghian, H.R., Taheri, S.H., 2006. On the discrete time, cost and quality trade-off problem. Applied Mathematics and Computation 181 (2), 1305–1312.

Tareghian, H.R., Taheri, S.H., 2007. A solution procedure for the discrete time, cost and quality trade-off problem using electromagnetic scatter search. Applied Mathematics and Computation 190 (2), 1136–1145.

Tiwari, V., Patterson, J.H., Mabert, V.A., 2009. Scheduling projects with heterogeneous resources to meet time and quality objectives. European Journal of Operational Research 193 (3), 780–790.

Toroslu, I.H., 2003. Personnel assignment problem with hierarchical ordering constraints. Computers and Industrial Engineering 45 (3), 493–510.

Tseng, L.Y., Chen, S.C., 2009. Two-phase genetic local search algorithm for the multi-mode resource-constrained project scheduling problem. IEEE Transactions on Evolutionary Computation 13 (4), 848–857.

Ulusoy, G., Cebelli, S., 2000. An equitable approach to the payment scheduling problem in project management. European Journal of Operational Research 127 (2), 262–278.

Ulusoy, G., Sivrikaya-Şerifoğlu, F., Şahin, S., 2001. Four payment models for the multi-mode resource constrained project scheduling problem with discounted cash flows. Annals of Operations Research 102 (1–4), 237–261.

Ulusoy, G., Özdamar, L., 1995. A heuristic scheduling algorithm for improving the duration and net present value of a project. International Journal of Operations and Production Management 15 (1), 89–98.

Valls, V., Ballestín, F., Quintanilla, S., 2004. A population-based approach to the resource-constrained project scheduling problem. Annals of Operations Research 131 (1–4), 305–324.

Valls, V., Ballestín, F., Quintanilla, S., 2005. Justification and RCPSP: a technique that pays. European Journal of Operational Research 165 (2), 375–386.

Valls, V., Laguna, M., Lino, P., Pérez, Á., Quintanilla, S., 1999. Project scheduling with stochastic activity interruptions. In: Węglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms and Applications. Kluwer, Dordrecht, pp. 333–353.

Valls, V., Pérez, Á, Quintanilla, S., 2009. Skilled workforce scheduling in service centres. European Journal of Operational Research 193 (3), 791–804.

Valls, V., Quintanilla, S., Ballestín, F., 2003. Resource-constrained project scheduling: a critical activity reordering heuristic. European Journal of Operational Research 149 (2), 282–301.

Van Hove, J.C., Deckro, R.F., 1998. Multi-modal project scheduling with generalized precedence constraints. In: Babarasoğlu, G., Karabati, S., Özdamar, L., Ulusoy, G. (Eds.), Proceedings of the Sixth International Workshop on Project Management and Scheduling, Istanbul, pp. 137–140.

Van Peteghem, V., Vanhoucke, M., 2010. A genetic algorithm for the preemptive and nonpreemptive multi-mode resource-constrained project scheduling problem. European Journal of Operational Research 201 (2), 409–418.

Vanhoucke, M., Demeulemeester, E., Herroelen, W., 2002. Discrete time/cost trade-offs in project scheduling with time-switch constraints. Journal of the Operational Research Society 53 (7), 741–751.

Vanhoucke, M., 2005. New computational results for the discrete time/cost trade-off problem with time-switch constraints. European Journal of Operational Research 165 (2), 359–374.

Vanhoucke, M., Debels, D., 2007. The discrete time/cost trade-off problem: extensions and heuristic procedures. Journal of Scheduling 10 (4–5), 311–326.

Vanhoucke, M., Debels, D., 2008. The impact of various activity assumptions on the lead time and resource utilization of resource-constrained projects. Computers and Industrial Engineering 54 (1), 140–154.

Waligóra, G., 2008a. Discrete-continuous project scheduling with discounted cash flows – a tabu search approach. Computers and Operations Research 35 (7), 2141–2153.

Waligóra, G., 2008b. Discrete-Continuous Project Scheduling – Models and Algorithms. Habilitation Thesis, Poznań University of Technology, Poznań.

Waligóra, G., 2010. Heuristic approaches to discrete-continuous project scheduling problems to minimize the makespan. Computational Optimization and Applications, in press.

Waligóra, G., submitted for publication. Methodology for discrete-continuous project scheduling with discounted cash inflows and various payment models. European Journal of Operational Research.

Węglarz, J., Błażewicz, J., Cellary, W., Słowiński, R., 1977. Algorithm 520. An automatic revised simplex method for constrained resource network scheduling. ACM Transactions on Mathematical Software 3 (3), 295–300.

Węglarz, J., 1976. Time-optimal control of resource allocation in a complex of operations framework. IEEE Transactions on Systems, Man and Cybernetics 6 (11), 783–788.

Węglarz, J., 1979. Project scheduling with discrete and continuous resources. IEEE Transactions on Systems, Man and Cybernetics 9 (10), 644–650.

Węglarz, J., 1980a. On certain models of resources allocation problems. Kybernetes 9 (1), 61–66.

Węglarz, J., 1980b. Multiprocessor scheduling with memory allocation – a deterministic approach. IEEE Transactions on Computers 29 (8), 703–709.

Węglarz, J., 1981. Project scheduling with continuously-divisible, doubly constrained resources. Management Science 27 (9), 1040–1052.

Węglarz, J., 1991. Synthesis problems in allocating continuous, doubly constrained resources among dynamic activities. In: Bradley, H.E. (Ed.), Operational Research '90: Selected Papers from the 12th IFORS International Conference on Operational Research, Athens, Greece, 25–29.06.1990, Pergamon Press, Oxford, 1991, pp. 715–724.

Zapata, J.C., Hodge, B.M., Reklaitis, G.V., 2008. The multi-mode resource constrained multi-project scheduling problem: alternative formulations. AIChE Journal 54 (8), 2101–2119.

Zhang, H., Tam, C.M., Li, H., 2006. Multi-mode project scheduling based on particle swarm optimization. Computer-Aided Civil and Infrastructure Engineering 21 (2), 93–103.

Zhu, G., Bard, J.F., Yu, G., 2006. A branch-and-cut procedure for the multi-mode resource-constrained project scheduling problem. INFORMS Journal on Computing 18 (3), 377–390.