European Journal of Operational Research 235 (2014) 62–72

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances

Vincent Van Peteghem [a,b], Mario Vanhoucke [b,c,d,*]

[a] EDHEC Business School, 24 Avenue Gustave Delory, 59057 Roubaix, France
[b] Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Gent, Belgium
[c] Technology and Operations Management Area, Vlerick Business School, Reep 1, 9000 Gent, Belgium
[d] Department of Management Science and Innovation, University College London, Gower Street, London WC1E 6BT, United Kingdom

ARTICLE INFO

Article history:
Received 29 June 2011
Accepted 6 October 2013
Available online 24 October 2013

Keywords:
Project scheduling
Multi-mode
Metaheuristics

ABSTRACT

In this paper, an overview is presented of the existing metaheuristic solution procedures to solve the multi-mode resource-constrained-project scheduling problem, in which multiple execution modes are available for each of the activities of the project. A fair comparison is made between the different metaheuristic algorithms on the existing benchmark datasets and on a newly generated dataset. Computational results are provided and recommendations for future research are formulated.

1. Introduction

Resource-constrained project scheduling has been a well-known and extensively studied research topic for the past decades. The optimization problem minimizes the makespan of the project, subject to precedence relations between the activities and resource constraints. When introducing different modes to each activity, with for every mode a different duration and different resource requirements, the problem is generalized to the Multi-Mode Resource-Constrained Project Scheduling Problem (MRCPSP).

In the MRCPSP, three different categories of resources can be distinguished (Slowinski, 1981): renewable resources, which are limited per time-unit (e.g. manpower, machines), nonrenewable resources, which are limited for the entire project (e.g. budget) and doubly constrained resources, which are limited both per time-unit and for the total project duration (e.g. cash-flow per time-unit). Since doubly constrained resources can be considered as a combination of renewable and nonrenewable resources, we do not consider them explicitly and do not take them into account in this study. In the remainder of this paper, we will refer to the MRCPSP/R if only renewable resources are considered while the general term MRCPSP is used for the multi-mode scheduling problem with both renewable and nonrenewable resources. Moreover, in this paper only discrete resources are considered. These resources can be allocated in discrete amounts to activities from a given finite set of possible allocations (Weglarz, Jozefowska, Mika, & Waligora, 2011).

As the MRCPSP is a generalization of the RCPSP, the MRCPSP is known to be NP-hard (Blazewicz, Lenstra, & Rinnooy Kan, 1983). Moreover, if there is more than one nonrenewable resource, the problem of finding a feasible solution for the MRCPSP is NP-complete (Kolisch, 1995). The problem is denoted as $m,\ 1T|cpm,disk,mu|C_{max}$ using the classification scheme of Demeulemeester and Herroelen (2002) and is denoted as $MPS|prec|C_{max}$ by Brucker, Drexl, Möhring, Neumann, and Pesch (1999).

In the MRCPSP, a set A of pairs of activities between which a precedence relationship exists, and a set N of n activities, where each activity $i \in N = \{1,\ldots,n\}$, can be performed in different execution modes, is given. The set $M_i = \{1,\ldots,|M_i|\}$ determines the possible execution modes for activity i, while m_i indicates the chosen mode for activity i. The duration of activity i, when executed in mode m_i, is d_{im_i}. Each mode m_i also requires $r^{\rho}_{im_ik}$ units for each resource in the set R^{ρ} of renewable resource types. For each renewable resource $k \in R^{\rho} = \{1,\ldots,|R^{\rho}|\}$, the availability a_k is constant throughout the project horizon. Activity i, executed in mode m_i, will also use r_{im_il} nonrenewable resource units of the total available nonrenewable resource a^{ν}_l, with $l \in R^{\nu}$ and R^{ν} the set of nonrenewable resources. The aim of the MRCPSP is to select exactly one

* Corresponding author at: Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Gent, Belgium. Tel.: +32 92643569.
E-mail addresses: vincent.vanpeteghem@edhec.edu (V. Van Peteghem), mario.vanhoucke@ugent.be (M. Vanhoucke).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.10.012

mode for each activity in order to schedule the project with a minimal makespan, subject to the resource and precedence constraints.

Talbot (1982) presented the mathematical programming formulation of this problem, while the MRCPSP can be formulated conceptually as follows:

$$\text{minimize } f_n \tag{1}$$

$$\text{subject to } f_i + d_{jm_j} \leqslant f_j \quad \forall (i,j) \in A \tag{2}$$

$$\sum_{i \in S(t)} r^{\rho}_{im_i k} \leqslant a^{\rho}_k \quad \forall k \in R^{\rho}, \ \forall t \tag{3}$$

$$\sum_{i=1}^{n} r^{v}_{im_i l} \leqslant a^{v}_l \quad \forall l \in R^{v}, \tag{4}$$

$$m_i \in M_i \quad \forall i \in N \tag{5}$$

$$f_0 = 0 \tag{6}$$

$$f_i \in \text{int}^+ \quad \forall i \in N \tag{7}$$

where $S(t)$ denotes the set of activities in progress in period $]t-1,t]$ and f_i the finish time of the ith activity. The objective of the MRCPSP is to find a mode and a start time for each activity such that the makespan is minimized and the schedule is feasible with respect to the precedence constraints and the renewable and nonrenewable resource constraints. If the schedule is not feasible with respect to one of the precedence or resource constraints, the solution is called infeasible.

Several exact, heuristic and metaheuristic solution methods for the MRCPSP have been proposed in the literature in recent years. Exact solution procedures have been proposed by Slowinski (1980), Talbot (1982), Patterson, Slowinski, Talbot, and Weglarz (1989), Speranza and Vercellis (1993), Sprecher (1994), Sprecher, Hartmann, and Drexl (1997), Sprecher and Drexl (1998), Hartmann and Drexl (1998) and Zhu, Bard, and Tu (2006). Except from the last, all procedures are applications of branch-and-bound algorithms, for which an overview is made by Hartmann and Drexl (1998). However, none of these exact procedures can be used for solving large-sized realistic projects, since they are unable to find an optimal solution in a reasonable computation time. Therefore, different single-pass heuristic and metaheuristic procedures were presented.

For the MRCPSP/R (with renewable resource only), heuristic procedures were proposed by Elmaghraby (1977); Boctor (1993, 1996a); Knotts, Dror, and Hartman (2000); Artigues and Roubellat (2000) and Lova, Tormos, and Barber (2006). Heuristic solution procedures for the general MRCPSP (with renewable and nonrenewable resources) were presented by Talbot (1982); Drexl and Grünewald (1993); Özdamar and Ulusoy (1994) and Kolisch and Drexl (1997).

The increasing interest in operations research for metaheuristics during the recent years has also resulted in the development of several metaheuristic solution procedures for the MRCPSP. A wide variety of metaheuristic strategies, solution representations and schedule generation schemes were used to develop the most efficient algorithms. Since the methods are tested on different benchmark datasets using different stop criteria, a fair comparison between each of these procedures is difficult. Moreover, the current benchmark datasets show several shortcomings, which could lead to biased results. For an extensive overview of the available exact, heuristic and metaheuristic approaches to the MRCPSP, the reader is referred to the overview paper of Weglarz et al. (2011).

The objective of this paper is threefold: first, an overview of the available metaheuristics is given and a classification is made based on the characteristics of each method. Second, a new dataset will be proposed in order to deal with the shortcomings of the current benchmark datasets. Finally, an extended computational comparison is performed to compare the metaheuristics and to evaluate the impact of the network and resource parameters of the project on the efficiency of the procedures.

The remainder of this paper is organized as follows: in Section 2, an overview is given of all the metaheuristics solution procedures for the MRCPSP. In Section 3, the shortcomings of the current benchmark datasets are mentioned and a new benchmark dataset is proposed. In Section 4, the computational experiments are reported as well as the results of the different metaheuristics on the current and new benchmark datasets. In the last section, overall conclusions and suggestions for future research are presented.

2. Metaheuristics for the MRCPSP

This section gives an overview of the current available metaheuristics from literature. In Section 2.1, we briefly explain the preprocessing process as proposed by Sprecher et al. (1997). In Section 2.2, an overview of the different classification criteria, as mentioned in Kolisch and Hartmann (1999), is given and the available algorithms are classified according to these criteria. Finally, in Section 2.3, an overview is given of the methodology used to code each of the metaheuristics in order to make a fair comparison for these procedures.

2.1. Preprocessing

Sprecher et al. (1997) developed a preprocessing procedure, which can be applied before the solution procedure is started, to reduce the search space in which the solution procedure will operate. This reduction procedure excludes those modes which are inefficient or non-executable and those resources which are redundant. As defined by Sprecher et al. (1997), a mode is called *inefficient* if there is another mode of the same activity with the same or smaller duration and no more requirements for all resources. A mode is called *non-executable* if its execution would violate the renewable or nonrenewable resource constraints in any schedule. A nonrenewable resource is called *redundant* if the sum of the maximal requests for that nonrenewable resource does not exceed its availability. Excluding these modes or nonrenewable resources does not affect the set of feasible or optimal schedules and reduces the search space of the problem.

2.2. Classification criteria

In order to make a classification of the available metaheuristics, the procedures are categorized based on four classification criteria: the metaheuristic strategy, the schedule representation, the mode representation and the schedule generation scheme. In what follows, each of these criteria is briefly discussed.

Metaheuristic strategy. Several metaheuristic strategies to solve a scheduling problem are available. For an overview of these metaheuristic strategies we refer to Glover and Kochenberger (2003) and Talbi (2009). For the MRCPSP the following eight strategies were used in literature: genetic algorithm (GA), scatter search (SS), simulated annealing (SA), particle swarm optimization (PSO), ant colony optimization (ACO), differential evolution algorithm (DEA), the multi-agent learning algorithm (MAL) and the estimation of distribution algorithm (EOD).

Schedule representation. Kolisch and Hartmann (1999) distinguished five different schedule representations in the RCPSP literature, from which the activity-list (AL) representation and the random key (RK) representation are the most widespread. In both representations, a priority structure between the activities is embedded. In the AL representation, the position of an

activity in the AL determines the relative priority of that activity versus the other activities, while in the RK representation the sequence in which the activities are scheduled is based on the priority value attributed to each activity.

Mode representation. Two mode representations can be distinguished in the literature: the mode list (ML) representation and the mode vector (MV) representation. In the mode list representation, the list represents the execution modes of the activities in ascending order, i.e. the first number in the list indicates the mode in which the first activity will be executed, the second number the execution mode of the second activity, etc. In the mode vector representation, the mode indicated in the *i*th position of the mode vector represents the execution mode of the activity placed in the *i*th position of the activity list. A mode vector is therefore always represented in combination with an activity list.

Schedule generation scheme. A schedule generation scheme (SGS) translates the schedule representation into a schedule. Two commonly used types of SGSs are the serial SGS (Kelley, 1963) and the parallel SGS (Bedworth & Bailey, 1982). Kolisch and Sprecher (1996) have shown that it is sometimes impossible to reach an optimal solution with the parallel SGS. Nevertheless, both schemes are used in the solution procedures currently available in the literature.

Local search procedures. Several local search (LS) procedures have been proposed in literature and used in the different algorithms. These procedures can be classified into several categories:

1. *Improving quality of initial population*: These procedures are focussing on the feasibility and quality of the initial population elements and try to improve the overall quality of the initial population elements. Therefore, simple (as the procedure of Hartmann (2001)) or more advanced (as the Minimum Normalized Resources procedure of Lova, Tormos, Cervantes, & Barber (2009)) procedures are developed.
2. *Improving feasibility*: These procedures are focussing on the improvement of the feasibility of a population element. Examples are the massive mutation procedure of Lova et al. (2009) and the mode shift procedure of Józefowska, Mika, Rózycki, Waligóra, and Weglarz (2001).
3. *Improving makespan*: These procedures are focussing on the improvement of the makespan of a population element. Examples are the single-pass improvement of Hartmann (2001) or the critical path improvement method of Van Peteghem and Vanhoucke (2011).
4. *Forward–backward*: Some algorithms are also using the forward–backward improvement method (Tormos & Lova, 2001) or an extended or modified version. This specific local search tries to improve the makespan by transforming left-justified schedules (where all activities are scheduled as soon as possible) into right-justified schedules (where all activities are scheduled as late as possible) and vice versa until no further improvements can be found.

In Table 1, an overview of the different metaheuristic algorithms is given in chronological order. This overview paper contains all metaheuristic solution procedures published in international peer reviewed journals before January 1, 2013. For each solution procedure, the name of the author(s) and the abbreviation which will be used in the remainder of the paper to refer to the procedure is given. The indication R or RNR in the third column indicates if the procedure is applicable on datasets with only renewable resources (R) or with both renewable and nonrenewable (RNR) resources. The information in the fourth, fifth, sixth and seventh column indicates the metaheuristic strategy (MS), the schedule representation (SR), the mode representation (MR), the schedule generation scheme (SGS) and the type of local search (LS) used to solve the problem instances, respectively. The algorithmic details of each of the procedures can be found in the respective papers. However, on the website http://www.projectmanagement.UGent.be an extensive summary of the details of each of these algorithms can be found.

2.3. Methodology

In order to compare each of the procedures presented in Table 1 on the same computer and within the same stop criteria, each algorithm described above has been coded. In this section, we will explain the methodology that has been followed in order to obtain a fair and realistic reproduction of the original codes.

Each paper is studied by three undergraduate students and one PhD student. For each procedure, three independently coded programs were available, each reviewed by a PhD student to look whether the structure of the program corresponded to the steps presented in the original paper. In case the content of the paper was insufficient to interpret the algorithm, other papers and solution procedures of the author were analyzed. If necessary, the following two adaptations were made. First, since not all algorithms included the preprocessing method of Sprecher et al. (1997), the preprocessing procedure was added to each solution procedure in order to make a fair comparison based on the same solution space. Second, procedures which are only applicable to datasets with renewable resources (more in particular the algorithms *BOCT*96 and *MORI*97), were transferred to a version in which nonrenewable resources are incorporated. In order to deal with possible infeasible solutions, the penalty function as introduced by Alcaraz, Maroto, and Ruiz (2003) was used in the code.

From the three independently coded programs, the program that obtained the results that were the closed to the results presented in the original paper (taken into account the stopping criteria) was selected and submitted to a second control phase, in which the code was reviewed again in detail. An overview of all coded solution procedures and the obtained results is given in Table 2. For every procedure, the different datasets for which results are presented in the original papers and the stop criterium which is applied are mentioned. All results were obtained on the same computer and indicate the average percentage deviation from the optimal solution, unless otherwise indicated.

The table shows that all procedures, except the procedures HART01 and ZHAN06, obtain results which are equivalent or slightly better than the results obtained in the original papers. This can be due to randomness, the incorporation of the preprocessing process or the competitive nature of the coding process. For the procedures VANP10, WAUT11, VANP11 and COEL11, the results were obtained from the authors.

The different procedures were used to test the performance of each algorithm. These computational tests were performed on both the available dataset PSPLIB and on a new dataset, which is described in detail in the next section. For each subset and each algorithm, a full factor design was set up in order to obtain the best results for each individual solution procedure.

3. A new dataset for the MRCPSP

Most of the metaheuristics mentioned in Table 1 were tested on two well-known benchmark datasets, the PSPLIB dataset (Kolisch, Sprecher, & Drexl, 1995) and the Boctor dataset (Boctor, 1993). However, the two datasets show some important shortcomings given the recent evolution in the development of metaheuristic search procedures. Therefore, we propose a new benchmark

Table 1
Classification metaheuristics.

Author	Abbr.	R/RNR	MS	SR	MR	SGS	LS
Slowinski et al. (1994)	SLOW94	RNR	SA	AL	ML	P	1
Boctor (1996b)	BOCT96	R	SA	AL	ML	S	–
Mori and Tseng (1997)	MORI97	R	GA	RK	ML	S	–
Özdamar (1999)	OZDA99	RNR	GA	RK	ML	P	–
Józefowska et al. (2001)	JOZE01	RNR	SA	AL	ML	S	1, 2
Hartmann (2001)	HART01	RNR	GA	AL	MV	S	1, 3
Bouleimen and Lecocq (2003)	BOUL03	RNR	SA	AL	MV	S	–
Alcaraz et al. (2003)	ALCA03	RNR	GA	AL	MV	S	1, 4
Zhang et al. (2006)	ZHAN06	RNR	PS	RK	ML	S	1, 2
Jarboui et al. (2008)	JARB08	RNR	PS	RK	ML	S	1, 3
Ranjbar et al. (2009)	RANJ09	RNR	SS	AL	MV	S	1, 4
Chiang et al. (2008)	CHIA08	RNR	ACO	RK	ML	P	1, 4
Lova et al. (2009)	LOVA09	RNR	GA	AL	MV	P/S	1, 2, 3, 4
Tseng and Chen (2009)	TSEN09	RNR	GA	AL	MV	S	1, 4
Damak et al. (2009)	DAMA09	RNR	DEA	RK	ML	S	–
Van Peteghem and Vanhoucke (2010)	VANP10	RNR	GA	RK	ML	S	1, 2, 3, 4
Elloumi and Fortemps (2010)	ELLO10	RNR	GA	AL	MV	S	1
Wauters et al. (2011)	WAUT11	RNR	MAL	AL	MV	S	3, 4
Van Peteghem and Vanhoucke (2011)	VANP11	RNR	SS	RK	ML	S	1, 2, 3, 4
Coelho and Vanhoucke (2011)	COEL11	RNR	GA	AL	–	S	4
Wang and Fang (2012)	WANG12	RNR	EOD	AL	MV	S	1, 2, 3, 4

Table 2
Results of all procedures in the literature (avg. % deviation from optimal solution).

Procedure	Dataset	Stop criterium	Result paper	Our result
SLOW94	Own	–	0.02	0.02
BOCT96	Boctor50[a]	–	25.70	25.13
	Boctor100[a]	–	27.20	26.82
MORI97	Own	*Dataset not available*		
OZDA99	J10	6000 schedules	0.38	0.34
JOZE01	J10	5000 schedules	1.16	0.99
	J20	5000 schedules	6.74	6.65
HART01	J10	6000 schedules	0.10	0.13
BOUL03	J10	50 seconds	0.21	0.18
	J20	50 seconds	2.10	2.09
ALCA03	J10	5000 schedules	0.24	0.24
	J20	5000 schedules	1.91	1.90
	Boctor50[a]	5000 schedules	26.52	26.48
	Boctor100[a]	5000 schedules	29.16	29.10
ZHAN06	J10	unknown/1 seconds	0.11	0.12
	J20	unknown/1 seconds	1.79	2.41
JARB08	J10	150 milliseconds	0.03	0.03
	J20	150 milliseconds	1.10	1.08
RANJ08	J10	5000 schedules	0.18	0.17
	J20	5000 schedules	1.64	1.31
CHIA08	J10	5000 schedules	0.16	0.15
	J20	5000 schedules	1.44	1.42
	J30	5000 schedules	2.42	2.21
LOVA09	J10	5000 schedules	0.06	0.04
	J20	5000 schedules	0.87	0.86
	J30[a]	5000 schedules	14.77	14.58
	Boctor50[a]	5000 schedules	23.70	23.65
	Boctor100[a]	5000 schedules	24.85	24.63
TSEN09	J10	5000 schedules	0.33	0.32
	J20	5000 schedules	1.71	1.47
	J30[a]	5000 schedules	18.33	17.06
DAMA09	J10	150 milliseconds/act	0.09	0.08
	J20	150 milliseconds/act	0.70	0.69
ELLO10	J10	5000 schedules	0.14	0.12
	J20	5000 schedules	1.62	1.51
VANP10	*Results obtained from authors*			
WAUT11	*Results obtained from authors*			
VANP11	*Results obtained from authors*			
COEL11	*Results obtained from authors*			
WANG12	J10	5000 schedules	0.12	0.09
	J20	5000 schedules	1.28	1.28

[a] Deviation from minimal critical path.

dataset which overcomes the disadvantages of the current datasets and which can be used as a benchmark dataset for further research. In Section 3.1, we make an analysis of the current benchmark datasets and explain their shortcomings, while in Section 3.2 the indicators of the newly developed and generated benchmark dataset are proposed.

Table 3
Overview of the characteristics of PSPLIB and Boctor dataset.

	PSPLIB	Boctor
Number of subsets	7	2
Number of instances/subset	640	120
Number of feasible instances/subset	549 (avg.)	120 (avg.)
Number of activities	10, 12, 14, 16, 18, 20, 30	50, 100
Number of renewable resources	2	1, 2, 4
Number of nonrenewable resources	2	0
Number of modes	3	1, 2, 4

3.1. Analysis of the current benchmark datasets

In this section, we make an analysis of the two current benchmark datasets, the PSPLIB dataset and the Boctor dataset. In Table 3, an overview of the input parameters of both datasets is given.

PSPLIB. The PSPLIB dataset is generated with the project generator ProGen (Kolisch et al., 1995) and is available in the project scheduling problem library PSPLIB from the ftp server of the Technische Universität München (http://129.187.106.231/psplib/). The dataset contains 7 subsets: the datasets J10, J12, J14, J16, J18, J20 and J30, containing 640 project instances with 10, 12, 14, 16, 18, 20 and 30 activities, respectively. Not for all instances, however, a feasible solution can be found. The projects in the dataset use 2 renewable and 2 nonrenewable resources and have 3 execution modes for each activity.

Boctor. This dataset, proposed by Boctor (1993), contains 2 subsets of 120 instances. There is one set of 50 activities (Boctor50) and one set of 100 activities (Boctor100). For each project, one, two or four renewable resource types are used and one, two or four execution modes per activity are determined. The Boctor dataset can also be downloaded from the PSPLIB server.

Several project parameters have been introduced in the literature for describing the characteristics of a project network and the resource scarceness. The coefficient of network complexity

(CNC, Pascoe, 1966), the order strength (OS, Mastor, 1970) and the I2 indicator (Vanhoucke, Coelho, Debels, Maenhout, & Tavares, 2008) are examples of network topology measures. The resource factor (RF, Pascoe, 1966) and resource strength (RS,Kolisch et al., 1995) are examples of resource scarceness measures.

An overview of the values of these instance characteristics for the J30 and Boctor100 dataset is given in Table 4. In this table, the average value, the minimum value and the maximum value for the different project characteristics (OS, CNC and I2) and the different resource characteristics (RS and RF) are presented. In Fig. 1, a frequency graph is given for the values of both the order strength and resource strength. As can be seen, the range for the values of the order strength (for J30 and Boctor100) and the resource strength (for Boctor100) is rather limited.

Despite the diverse range of the resource parameters (the RS varies between 0.25 and 1) and the incorporation of both renewable and nonrenewable resources in the J30 dataset, we believe that the instances of the PSPLIB dataset have four major shortcomings to stimulate further research for the MRCPSP. A first shortcoming lies in the inability to report feasible solutions for all problem instances. As an example, only 552 instances from the 640 generated instances have a possible feasible solution. A similar shortcoming holds for the J10 to J20 datasets. This results in an average of 549 feasible instances over all PSPLIB subsets. (see Alcaraz et al., 2003; Van Peteghem & Vanhoucke, 2010). A second drawback is the small range of OS values (between 0.35 and 0.60) which implies a low diversity in the complexity of the project topology networks. A third shortcoming lies in the observation that most solution procedures are able to solve the project instances to near optimality, which leaves little room to reach major improvements with newly developed solution procedures. Future instances should contain projects with more activities (>30), more mode combinations per activity (>3) or more renewable or nonrenewable resources. Finally, the generation of activity modes should be done differently, in order to guarantee that all modes are efficient. In the J30 dataset, three modes have been generated per activity. However, a number of these modes can be deleted by the preprocessing procedure of Sprecher et al. (1997), leading to only 2.88 modes per activity on average (see Van Peteghem & Vanhoucke, 2010). Moreover, the deletion of inefficient modes leads to a change of the resource parameters of the project instance, which could possibly lead to biased results.

The main advantage of the Boctor100 dataset is the large number of activities per project instance. However, three major concerns about this dataset can be specified. First, only renewable resources are taken into account and nonrenewable resources are neglected. Second, the average resource strength per project is not larger than 0.25, which means that the renewable resources are almost not restricted. Finally, the order strength of the projects varies between 0.8 and 0.95, which means that the projects are mainly serial.

Table 4
Overview of the characteristics of PSPLIB and Boctor.

	J30			Boctor100		
	Avg.	Min	Max	Avg.	Min	Max
Project characteristics						
OS	0.46	0.34	0.61	0.87	0.79	0.93
CNC	1.81	1.81	1.81	1.59	1.44	1.92
I2	0.29	0.23	0.43	0.53	0.40	0.62
Resource characteristics						
Resource strength (R)	0.62	0.25	1	0.15	0.06	0.25
Resource factor (R)	0.75	0.5	1	0.88	0.67	1
Resource strength (NR)	0.62	0.25	1	–	–	–
Resource factor (NR)	0.75	0.5	1	–	–	–

In the next section, the new MMLIB dataset for the MRCPSP will be presented. This dataset will cover most of the shortcomings mentioned in this section.

3.2. The MMLIB dataset for the MRCPSP

3.2.1. Generation conditions

In order to overcome the shortcomings of the PSPLIB and Boctor datasets, the following conditions were taken into account while generating the dataset:

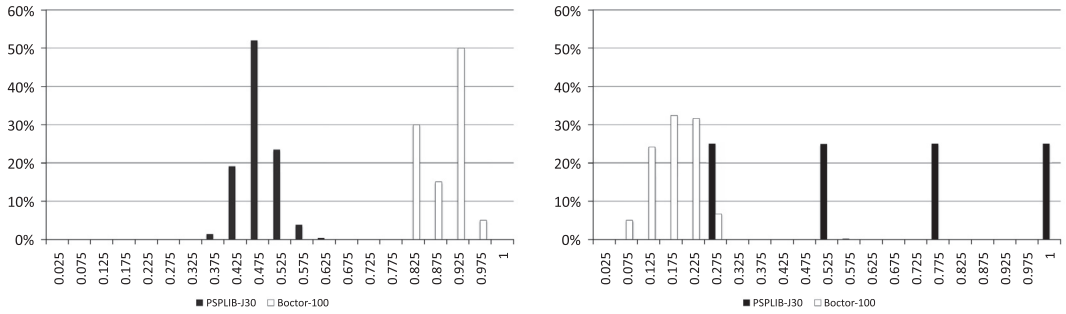
1. The generated instances are diverse with respect to the project (OS) and resource characteristics (RS and RF).
2. Every instance has at least one feasible solution.
3. No modes can be excluded.
4. Both renewable and nonrenewable resources are taken into account.

3.2.2. Dataset generation

For the generation of the instances, we have used the RanGen project scheduling instances generator developed by Vanhoucke et al. (2008) and extended to projects with multiple modes. However, in a limited number of cases it is not possible to fullfill the conditions mentioned in the previous section. In order to meet these conditions, two repair functions are used under specific project settings.

3.2.2.1. Repair function 1. During the generation of the instances, it was noted that for projects with low values for the nonrenewable resource strength ($RS^{\nu} = 0.25$) and resource factor ($RF^{\nu} = 0.50$), many infeasible projects were generated. This problem was also cited by Demeulemeester, Vanhoucke, and Herroelen (2003) who stated that the feasibility of the problem cannot be assured for low values of the resource strength and will lead to many infeasible modes, that is, modes for which the resource demand exceeds the availability a^{ν}. Since the generation of the resource demands per activity occurs randomly, only in a limited number of cases a feasible project is generated. Moreover, this number decreases for an increasing number of activities per project. In order to avoid this specific generation problem, a repair function is applied on the generated instances in order to obtain projects with a RS^{ν} equal to 0.25 and RF^{ν} equal to 0.50. In the first phase, instances are generated using the RanGen instance generator and extended to projects with multiple modes, as is also done for the instances with other project characteristics. However, in the second phase, the nonrenewable resource demand information was deleted and replaced by new data, obtained according to the following steps:

- Firstly, a large set of small (sub)projects with 10 activities and a $RS^{\nu} = 0.25$ and a $RF^{\nu} = 0.50$ is generated until in total 1000 subprojects (with 10 activities) are obtained for which at least 1 feasible solution (w.r.t. the nonrenewable resource demand) is found.
- Secondly, for each generated project with 50 or 100 activities, respectively 5 or 10 subprojects are randomly selected from the set of feasible subprojects.
- Thirdly, from each activity of these subprojects the nonrenewable resource demand information is then randomly assigned to one of the activities of the new project. For example, the nonrenewable data information of activity 4 of subproject 2 is then randomly assigned to activity 43 of the new project. This is done till all activities of the new project have obtained nonrenewable resource demand information from one of the subprojects. Only the nonrenewable resource demand information is used, other project data of the subprojects is not considered. The total



(a) Order strength (b) Resource strength

Fig. 1. Frequency table for the project instance characteristics.

nonrenewable resource availability is obtained by taking the sum of all nonrenewable resource availabilities of the selected 5 (10) subprojects.

This repair function is based on the mathematical formulation of the nonrenewable resource strength, which states that RS^v is equal to

$$RS^v = \frac{a^v - r^v_{min}}{r^v_{max} - r^v_{min}} \tag{8}$$

with r^v_{min} and r^v_{max} the minimum and maximum nonrenewable resource consumption which can be obtained by cumulating the consumptions obtained when performing each activity in the mode having minimum and maximum consumptions.

The idea of this repair function can be explained using the following example. Two projects A and B are randomly generated, each with 5 activities per project and 2 modes per activity. For these two projects, the information about the two nonrenewable resources is given in Table 5. It can easily be calculated that the RF^v for both projects is 0.5 and that the RS^v is equal to 0.25. By choosing for each activity the first mode, a feasible mode assignment can be found.

In order to generate a new project C with 10 activities, the nonrenewable resource data from each activity of project A and B is now randomly assigned to one of the activities of project C. In Table 5, the activity in project C to which each activity of project A and B is assigned is given between brackets in the first column. The nonrenewable resource availability of project C is equal to 9 and 12, which is the sum of the nonrenewable resource availabilities of A and B. It can be calculated that the RF^v of project C remains 0.5 and that the RS^v is still equal to 0.25.

3.2.2.2. Repair function 2. Many inefficient modes were generated during the data generation process. Modes are labelled as *inefficient* if its duration is not shorter and the resource demand for each resource is not less than that of another mode of the same activity. Since all projects are generated randomly, it is straightforward that inefficient modes are generated. Therefore, a repair function is used in order to deal with the problem of inefficient modes. If an inefficient mode is detected, the resource requirement for one of the nonrenewable resources is set to a value lower than the original nonrenewable resource demand. This is only possible if the nonrenewable resource demand is not zero or if the nonrenewable resource demand is larger than the minimal resource demand for that activity, otherwise the nonrenewable resource strength will be affected (as can be derived from Eq. (8)). In case no changes can be performed, a new project is generated.

By applying these 2 repair functions, we were able to generate a set of project instances which fulfills all conditions set in Section 3.2.1.

Table 5
Nonrenewable resource information of projects A and B.

i	m_i	$r^v_{im_i1}$	$r^v_{im_i1}$
Project A			
1 (5)	1	2	0
	2	0	9
2 (2)	1	3	0
	2	2	0
3 (7)	1	0	1
	2	0	7
4 (8)	1	0	2
	2	5	0
5 (3)	1	0	3
	2	4	0
	a^v	5	6
Project B			
1 (9)	1	0	2
	2	7	0
2 (4)	1	3	0
	2	0	5
3 (1)	1	0	2
	2	0	4
4 (10)	1	0	2
	2	5	0
5 (6)	1	1	0
	2	0	5
	a^v	4	6

3.2.3. Dataset characteristics

The dataset generation resulted in the design of a new dataset MMLIB with three subsets: the MMLIB50 and MMLIB100 sets, which have project characteristics similar to the PSPLIB set and take the shortcomings mentioned in the previous section into account, and the MMLIB + dataset, which also takes more resources and more modes into account. The motivation to generate the new dataset comes from the idea that the complexity of the current benchmark datasets should be increased. The MMLIB50 and MMLIB100 are generated in order to increase the complexity of the scheduling problem (i.e. more activities), the MMLIB + dataset is generated to increase the complexity of both the scheduling problem (i.e. more activities) and the mode assignment problem (i.e. more possible nonrenewable mode combinations).

MMLIB50 and MMLIB100. The MMLIB50 and MMLIB100 dataset contains 50 and 100 activities, respectively. Each project activity has 2 renewable and 2 nonrenewable resources and for each activity, 3 modes were defined. The order strength is set at 0.25, 0.50 or 0.75. The resource strength of both the renewable and nonrenewable resource strength is set at 0.25, 0.50 or 0.75 and the resource factor is set at 0.50 or 1 for both renewable

and nonrenewable resources. Using 5 instances for each problem class, each dataset contains 540 instances. The MMLIB50 and MMLIB100 datasets can be considered as similar to the PSPLIB datasets, since most parameters in both datasets are equal. However, some important differences should be noticed: firstly, the order strength of the projects in the MMLIB50 and MMLIB100 dataset is more diverse than the order strength of the projects in the PSPLIB dataset. Secondly, the MMLIB50 and MMLIB100 datasets do not contain projects with a resource strength equal to 1. Thirdly, all the instances have a feasible solution and finally, no modes can be deleted during the preprocessing procedure.

MMLIB+. The MMLIB + dataset contains projects with 50 and 100 activities. Each project activity has 2 or 4 renewable and nonrenewable resources. For each activity, 3, 6 or 9 modes are defined. The order strength is set at 0.25, 0.50 or 0.75. The resource strength of both the renewable and nonrenewable resource strength is set at 0.25, 0.50 or 0.75. In order to keep the number of instances to a reasonable level, the renewable and nonrenewable resource factor is set at 1. Using 5 instances for each problem class, the MMLIB + dataset contains 3240 instances.

3.2.4. Download

The MMLIB dataset and the best known solutions for each problem instance are available online. They can be downloaded from the website http://www.projectmanagement.UGent.be. This website will be updated regularly, and we call upon researchers to report the solutions of their procedures when this leads to an improvement.

4. Computational comparison

In this section, a computational comparison is made between the different metaheuristic solution procedures. In Section 4.1, the design of the tests is explained, while in Section 4.2, the results of the tests on the PSPLIB and Boctor dataset (Section 4.2.1) and MMLIB dataset (Section 4.2.2) are presented.

4.1. Test design

4.1.1. Problem instances

The tests which are performed in this computational comparison are executed on the following two sets of problem instances.

1. First, the metaheuristic solution procedures are executed on the instances of the J10, J20, J30, Boct50 and Boct100 dataset.
2. Secondly, the procedures are also tested on the instances of the MMLIB dataset, namely the MMLIB50, MMLIB100 and MMLIB+.

In order to keep the number of tests under control, the instances of the J12, J14, J16 and J18 subsets of the PSPLIB dataset are not included in the comparison.

4.1.2. Stop criterion

In order to make a fair comparison between the different solution procedures, the evaluation is stopped after a predefined number of generated schedules. According to Kolisch and Hartmann (2006), the advantage of the number of schedules as stop criterion is twofold: first, it is *platform independent* and second, future studies can easily make use of the *benchmark results* by applying the same stop criterion. However, the stop criterion also has a few shortcomings: first, it cannot be applied to all different heuristic strategies. Second, the required time to compute one schedule might differ between metaheuristics. Nevertheless, Kolisch and Hartmann (2006) conclude that limiting the number of schedules is the best criterion available for a broad comparison, which motivated us to use this stop criterion in all computational experiments.

In order to measure the number of generated schedules, the definition of one schedule should be defined. In their RCPSP review paper, Kolisch and Hartmann (2006) state that one schedule corresponds to (at most) one start time assignment per activity, as done by a SGS. However, measuring the number of schedules according to this rule means that for every mode change in a local search procedure a new schedule should be counted. Therefore, Lova et al. (2009) define the number of generated schedules as the sum of times each activity of the project has obtained a feasible start time divided by the number of activities of the project. This means that each change of the schedule, i.e. by changing either the start time or the finish time of one activity, is counted as $1/|N|$ schedules. Assume a project with eight activities, each with three modes. Suppose that the SGS generates a schedule based on an activity and mode list (8 start times are assigned) and that a local search procedure has also analyzed the two other (feasible) modes for three of the activities (without affecting the start times of the other activities). This means that the procedure has generated and analyzed $(8 + 2 \times 3)/8 = 1,75$ schedules.

In the remainder of this paper, the definition of Lova et al. (2009) is used to define the number of schedules.

4.2. Experimental results

4.2.1. Results of the PSPLIB and Boctor dataset

The results for the first set of instances obtained after 5000 schedules can be found in Table 6. In the first column, the reference to the authors is given, while in the second column the metaheuristic strategy (MS) is indicated. In the following columns, the results for the J10, J20, J30, Boctor50 and Boctor100 sets are given. For the J10 and J20 set, the average deviation from the optimal solution is given. For the J30 and Boctor dataset, the deviation from the minimal critical path-based lower bound is reported. If a solution procedure was not able to obtain a feasible solution for all project instances, the percentage of instances for which a feasible solution was found is indicated between brackets. As can be seen in the table, not all heuristics were able to obtain a feasible solution for all project instances. The metaheuristics are sorted with respect to decreasing average deviations for the Boctor100 dataset, except for schedules which were not able to obtain feasible results for all datasets.

4.2.1.1. Best heuristic. The best results for a specific dataset are displayed in bold. In order to determine the best heuristic, the concept of dominance as defined in Kolisch and Hartmann (2006) is used: a heuristic a is dominated by a heuristic b if a has for at least one subset a higher average deviation than b without having for any of the other subsets a lower average deviation. Table 6 reveals that the procedures of Lova et al. (2009); Van Peteghem and Vanhoucke (2010) and Van Peteghem and Vanhoucke (2011) are not dominated by other heuristics. For the PSPLIB dataset, the procedure of Van Peteghem and Vanhoucke (2011) dominates all other metaheuristics.

4.2.2. Results of the MMLIB dataset

The results of the computational tests on the MMLIB50, MMLIB100 and MMLIB+ are given in the Tables 7–9. In these tables, performance measures for the three stop criterion (1000, 5000 and 50,000 schedules, respectively) are shown. For every stop criterion, the average deviation from the minimal critical path is indicated. However, when the algorithm was not able to obtain a feasible solution for all project instances in the set, the percentage

Table 6
Average deviation from optimum/critical path lower bound for PSPLIB and Boctor instances after 5000 schedules.

Author	MS	J10	J20	J30	Boct50	Boct100
SLOW94	SA	(*44.96*)	(*50.18*)	(*48.55*)	31.70	38.14
BOCT96	SA	0.27	(*99.10*)	(*93.66*)	25.13	26.82
BOUL03	SA	2.65	(*99.64*)	(*99.64*)	28.53	31.49
MORE97	GA	2.68	13.55	24.99	41.28	53.81
OZDA99	GA	0.44	6.05	27.38	37.81	47.21
JARB08	PS	0.22	2.44	18.14	30.31	37.82
CHIA08	ACO	0.15	2.42	15.18	31.64	36.21
ZHAN06	PS	0.31	3.17	18.63	28.97	32.83
ELLO10	GA	0.12	1.51	16.16	27.56	31.68
WAUT11	MAL	0.05	0.70	13.91	26.56	30.31
COEL11	GA	0.07	0.80	14.44	25.11	30.03
DAMA09	DEA	0.74	1.62	15.43	25.43	29.14
ALCA03	GA	0.24	1.96	21.83	26.48	29.10
WANG12	EOD	0.09	1.28	15.55	26.52	28.98
TSEN09	GA	0.32	1.47	17.06	26.60	27.12
HART01	GA	0.20	1.61	15.96	24.73	26.47
RANJ09	SS	0.17	1.31	16.21	24.10	25.84
JOZE01	SA	0.99	6.65	18.64	24.44	25.70
VANP11	SS	**0.00**	**0.32**	**13.66**	23.79	25.11
VANP10	GA	0.01	0.57	13.75	**23.41**	24.67
LOVA09	GA	0.04	0.89	14.58	23.65	**24.63**

(*Number*): % of feasible solutions found.
Number: average % deviation from minimal CPBLB.

of instances for which a feasible solution is found, is shown. This percentage is indicated between brackets.

As can be seen in Table 7, the solution procedures of Slowinski, Soniewicki, and Weglarz (1994), Boctor (1996b), Bouleimen and Lecocq (2003), Özdamar (1999) and Coelho and Vanhoucke (2011) do not succeed in obtaining a feasible schedule for all project instances of the MMLIB50 dataset after 50,000 schedules. This is mainly due to the fact that these metaheuristics have not included a search procedure to enhance the feasibility of an infeasible solution vector or to the increased complexity of the projects. The procedure of Boctor (1996b) was designed to solve the MRCPSP/R, while the procedure of Bouleimen and Lecocq (2003) was originally designed to solve the RCPSP. The procedure of Coelho and Vanhoucke (2011) produces infeasible solutions due to the limitations in the SAT-solver. The number of procedures which are not able to generate feasible solutions for all instances further increases for MMLIB100 (Table 8) and MMLIB+ (Table 9), mainly due to the increased complexity of the instances (more activities and more modes per activity). As can be seen in Table 9, the feasibility rate of some procedures significantly increases for an increasing number of schedules evaluated (Boctor, 1996b; Mori & Tseng, 1997), while other metaheuristic solution procedures do not succeed in increasing this rate (Slowinski et al., 1994; Özdamar, 1999; Bouleimen & Lecocq, 2003; Ranjbar, De Reyck, & Kianfar, 2009; Coelho & Vanhoucke, 2011).

4.2.2.1. Best heuristic. Based on the results, it can be stated that the procedure of Van Peteghem and Vanhoucke (2011) dominates all other metaheuristics for the MMLIB-datasets.

4.3. Performance of metaheuristics

A closer look on the results shows that the main difference in the performance of the different algorithms is mainly due to the characteristics of the local search procedures used in the algorithm. Algorithms that make use of more advanced initial population procedures (LS1) perform better, just as the algorithms that make use of the feasibility improvement (LS2) and the makespan improvement local search (LS3). The use of the forward–backward local search procedure does not distinguish between a good or bad performing metaheuristic, as opposed to what can be found in literature (Kolisch & Hartmann, 2006).

This general analysis can be proven by the multivariate regression analysis that is performed. In Table 10, the results are shown for a regression analysis with the different algorithmic components (local search procedures) as independent variables and the deviation of the result from the best known solution as the dependent variable (R^2 = .342). All independent variables are dummy variables. The values for the different local searches indicate if the specific local search is used (value = 1) or not (value = 0). It should be noted that for the initial population local search, only the more advanced local search procedures, such as the minimum normalized resources procedure of Lova et al. (2009) and the controlled mode assignment procedure of Van Peteghem and Vanhoucke (2011), are taken into account (see Section 2.2).

Table 7
Solutions for MMLIB50 after 1000, 5000 and 50,000 schedules.

Author	MS	*Number of schedules*		
		1000	5000	50,000
SLOW94	SA	(*21.85*)	(*23.52*)	(*25.00*)
BOCT96	SA	(*67.59*)	(*69.63*)	(*78.52*)
BOUL03	SA	(*72.96*)	(*77.78*)	(*82.78*)
COEL11	GA	(*83.33*)	(*83.33*)	(*83.15*)
OZDA99	GA	(*83.15*)	(*83.33*)	(*83.33*)
CHIA08	ACO	(*83.33*)	(*83.33*)	(*83.33*)
MORI97	GA	(*72.22*)	(*83.52*)	46.91
ALCA03	GA	56.06	43.05	40.69
WAUT11	MAL	(*89.07*)	(*94.63*)	33.49
JARB08	PS	49.98	38.86	32.01
ZHAN06	PS	49.25	35.94	30.23
TSEN09	GA	65.17	37.92	29.44
WANG12	EOD	43.17	31.95	28.76
RANJ09	SS	38.49	32.16	28.55
JOZE01	SA	49.06	33.81	27.81
ELLO10	GA	43.84	32.47	26.92
HART01	GA	35.40	30.61	26.81
LOVA09	GA	34.16	28.59	26.69
DAMA09	DEA	46.19	32.46	26.27
VANP10	GA	34.07	27.12	24.93
VANP11	SS	**28.17**	**25.45**	**23.79**

(*Number*): % of feasible solutions found.
Number: average % deviation from minimal CPBLB.

Table 8
Solutions for MMLIB100 after 1000, 5000 and 50,000 schedules.

Authors	MS	*Number of schedules*		
		1000	5000	50,000
SLOW94	SA	(*18.89*)	(*19.81*)	(*21.48*)
BOCT96	SA	(*66.67*)	(*66.67*)	(*66.67*)
BOUL03	SA	(*66.67*)	(*67.04*)	(*67.41*)
OZDA99	GA	(*66.67*)	(*67.59*)	(*67.59*)
COEL11	GA	(*83.33*)	(*83.33*)	(*81.85*)
MORI97	GA	(*67.04*)	(*69.63*)	(*83.33*)
CHIA08	ACO	(*83.33*)	(*83.33*)	(*83.33*)
WAUT11	MAL	(*83.15*)	(*83.89*)	(*96.85*)
ALCA03	GA	61.80	52.67	46.68
JARB08	PS	(*91.85*)	49.41	40.23
TSEN09	GA	72.95	66.04	37.04
ZHAN06	PS	57.42	44.05	35.35
RANJ09	SS	45.16	37.00	34.17
WANG12	EOD	52.94	38.55	30.45
JOZE01	SA	53.97	39.05	30.27
ELLO10	GA	56.21	40.22	30.02
HART01	GA	39.96	33.98	29.04
DAMA09	DEA	52.31	36.87	28.92
LOVA09	GA	36.29	31.01	27.89
VANP10	GA	37.58	29.55	25.63
VANP11	SS	**29.77**	**26.51**	**24.02**

(*Number*): % of feasible solutions found.
Number: average % deviation from minimal CPBLB.

Table 9
Solutions for MMLIB+ after 1000, 5000 and 50,000 schedules.

Authors	MS	*Number of schedules*	
		5000	50,000
SLOW94	SA	(*4.78*)	(*4.78*)
COEL11	GA	(*50.00*)	(*53.90*)
BOCT96	SA	(*54.48*)	(*65.77*)
CHIA08	ACO	(*66.67*)	(*66.67*)
OZDA99	GA	(*68.64*)	(*68.77*)
BOUL03	SA	(*67.96*)	(*70.59*)
MORI97	GA	(*72.59*)	(*97.22*)
VANP10	GA	(*97.59*)	(*97.59*)
ZHAN06	PS	(*99.81*)	(*99.85*)
ALCA03	GA	177.55	164.87
TSEN09	GA	183.02	142.14
JARB08	PS	(*94.29*)	137.99
WAUT11	MAL	145.78	132.88
RANJ09	SS	131.45	131.07
HART01	GA	132.01	111.45
WANG12	EOD	144.84	110.43
ELLO10	GA	130.06	106.35
DAMA09	DEA	126.69	103.75
JOZE01	SA	121.09	103.19
LOVA09	GA	114.07	102.73
VANP11	SS	**101.45**	**92.76**

(*Number*): % of feasible solutions found.
Number: average % deviation from minimal CPBLB.

As can be seen in the table, all parameters are significant ($p < 0.001$).The influence of the first three local searches is straightforward: the initial population local search (LS1), the improving feasibility local search (LS2) and the improving makespan local search (LS3) all have a negative coefficient and thus, a positive and significant impact on the results. The result for the forward–backward local search (LS4) is opposite as expected (Kolisch & Hartmann, 2006). As shown in the analysis, the positive coefficient indicates that worse results will be obtained if this local search is used. However, this is not in line with previous research, where this local search procedure has already proven being successful (see Valls, Ballestin, & Quintanilla, 2005). A closer look to the literature also shows that the introduction of the forward–backward procedure results in a better performance of the individual algorithm. However, since not all of the algorithms are using this method and some of them not using it are also obtaining good results, the forward–backward procedure is no conditione sine qua non for obtaining good results.

A specific remark is dedicated to the DEA-procedure, as proposed by Damak, Jarboui, Siarry, and Loukil (2009). This algorithm does not use any specific local search procedure, but is performing quite good: 3rd position for MMLIB50, 4th for MMLIB100 and 4th for MMLIB+ (after 50,000 schedules). This indicates that this search strategy is very promising for the problem under study and probably will also perform well for other (project) scheduling problems. The choice for a specific metaheuristic search strategy does not consistently lead to better results. This can be seen by the results of the different genetic algorithms (the search strategy that is used the most to solve the MRCPSP): while the genetic algorithms of Lova et al. (2009) and Van Peteghem and Vanhoucke (2010) perform very well on both the PSPLIB and MMLIB dataset, the genetic algorithms of Mori and Tseng (1997) and Özdamar (1999) generate solutions of low quality. The same remark can be made for other search strategies, such as the simulated annealing and scatter search.

Table 10
Influence of the algorithmic components on the deviation.

Algorithmic component	Coefficient
Constant	.217*
LS1 – Initial population	–.173*
LS2 – Feasibility improvement	–.064*
LS3 – Makespan improvement	–.047*
LS4 – Forward–backward	.157*

Based on these results, it can be stated that the use of specific heuristic components have a positive and significant influence on the results obtained by a specific heuristic.

4.4. Influence of the project parameters

In this section, the influence of the different project parameters is studied and evaluated. First, a multivariate linear regression analysis assesses the influence of the renewable and nonrenewable resource strength (RS^{ρ} and RS^{ν}), the order strength (OS), the number of modes ($|M|$) and the number of renewable ($|R^{\rho}|$) and nonrenewable ($|R^{\nu}|$) resources (the independent variables) on the deviation from the best known solution after 50,000 schedules (the dependent variable). The analysis is performed on the MMLIB + dataset truncated after 5000 schedules and is performed on each algorithm that obtains feasible solutions for all project instances. The results of this analysis are provided in Table 11. For each method, the resulting R^2-value, the constant (Const), and the coefficient for each problem parameter are provided. An asterisk (*) indicates that the coefficient is significant at the 1% level of confidence. If no value is indicated, the coefficient is not significant at the 5% level of confidence.

The resulting R^2-values range between .185 and .597, which means that the 6 problem parameters are not able to predict the performance of a heuristic on a specific project instance. As can be seen, the resource strength (both renewable and nonrenewable) and the number of modes are the only parameters for which all values are significant at the 5% level. For the number of modes, the coefficients are all highly significant and positive, which means that an increase in the number of modes results in a higher deviation from the best known solution, mainly due to the increased number of feasible mode combinations.

For the renewable resource strength RS^{ρ}, the coefficients are mainly negative which indicates that for an increasing renewable resource strength, the deviation with the best known solution becomes smaller. The higher the value of the coefficient, the more sensitive the heuristic is in terms of the resource scarceness of the project instance. The most sensitive heuristic is the one of Lova et al. (2009) (see also below), while the most insensitive heuristic is the one of Van Peteghem and Vanhoucke (2011). Most heuristics also have a positive coefficient for the nonrenewable resource strength RS^{ν}, which is opposite to the renewable resource strength. It means that the higher the nonrenewable resource strength becomes (thus the easier it becomes to find a feasible solution), the larger the deviation with the best known solution will be. This seems counter-intuitive, but can be explained by the increased search space that comes with an increased number of feasible mode combinations: since more mode combinations will lead to a feasible solution, the risk of selecting a mode combination which leads to a poor schedule becomes higher.

The same conclusion can be made for the OS coefficient. All significant OS coefficients are negative, indicating that with an increasing value of the order strength, the average performance of the heuristics increases. This in line with the findings of Hartmann and Kolisch (2000), who stated that more precedence relations between activities lower the number of possible activity sequences and thus the size of the solution space. The high value of the OS-coefficient for the heuristic of Józefowska et al. (2001) indicates the this procedure becomes more performant if the complexity of the project network increases.

The influence of the number of nonrenewable resources is rather low, ranging from -.020 to 0.004, while the influence of

Table 11
Results of the multivariate regression on the MMLIB + dataset.

Procedure	R^2	Const	OS	RS^{ρ}	RS^{v}	$\|M\|$	$\|R^{\rho}\|$	$\|R^{v}\|$
VANP11	.185	.019*	−.012	−.009*	−.022*	.007*	.002	.003*
LOVA09	.402	−.089*		.119*	.112*	.017*		
JOZE01	.350	.161*	−.147*	−.064*	−.038*	.017*		−.005*
DAMA09	.229	.134*	−.043	−.075*	−.014	.015*	.005*	.003
ELLO10	.471		−.073*	−.024*	.301*	.028*		−.012*
WANG12	.587	.152*	−.024*	−.070*	−.133*	.035*	.005*	
HART01	.452	.055*	−.030*	−.038*	.082*	.025*	.005	.004*
RANJ09	.465		−.037	.019*	.104*	.028*		−.007*
WAUT11	.420	.048*	−.048*	−.085*	.093*	.031*		
TSEN09	.597	−.052		−.073*	.653*	.053*		−.020*
ALCA03	.471	.080*		−.039*	−.331*	.046*		−.018*

the number of renewable resources on the solution quality is negligible (only in 3 cases the coefficient is significant). Finally, the constant-values are significant at the 1% level for all heuristics and give an indication of the heuristic performance: the lower the value of constant, the better – on average – the performance. It can be noticed that the procedure of Lova et al. (2009) has a better (negative) value for the constant coefficient than the procedure of Van Peteghem and Vanhoucke (2011), although the latter has an overall better performance.

In the remainder of this section, the focus is on the effects of the renewable and nonrenewable resource strength. In Table 12, the average deviations from the best known solutions after 5000 iterations for each parameter setting of the MMLIB + dataset, respectively, is given. The number between brackets (if indicated) indicates the relative position of the procedure under the defined parameter setting. The table only indicates a result if all solutions found are feasible.

For the renewable resource strength, the results are in line with the results obtained from the multivariate regression analysis: the procedure of Van Peteghem and Vanhoucke (2011) performs the best under all parameter settings (indicated in bold). It can also be noticed that the algorithm of Lova et al. (2006) performs very well for low values of RS^{ρ}, while its performance diminishes for higher values. This is in line with the rather high coefficient values found for RS^{ρ} in the multivariate regression analysis. The inverse is true for the procedures of Józefowska et al. (2001) and Damak et al., 2009 which perform very well for high values of RS^{ρ}.

For the nonrenewable resource strength RS^{v}, the number of procedures able to find 100% feasible solutions increases for an increasing RS^{v}, i.e. for a declining resource availability: the number increases from 11 (RS^{v} = .25) to 16 (RS^{v} = .75). This is obviously a consequence of the increased availability of nonrenewable resources and the decreased complexity to find a feasible solution.

Table 12
Average deviation from best known solution after 5000 schedules.

Procedure	RS^{ρ}			RS^{v}		
	0.25	0.50	0.75	0.25	0.50	0.75
BOCT96						.322
MORI97					.505	.814
OZDA99					.353	.593
JOZE01	.167(3)	.158(3)	.135(2)	.141	.162(3)	.159(3)
HART01	.218	.216	.199	.182	.227	.224
BOUL03					.508	.815
ALCA03	.466	.455	.447	.389	.425	.555
ZHAN06		.319	.320		.295	.419
JARB08					.326	.375
RANJ09	.209	.203	.219	.174	.231	.226
LOVA09	.100(2)	.125(2)	.160	.096(2)	.137(2)	.152(2)
TSEN09	.518	.494	.482	.357	.454	.683
DAMA09	.193	.192	.155(3)	.174	.198	.167
ELLO10	.214	.226	.202	.140(3)	.210	.291
VANP11	**.058**(1)	**.047**(1)	**.053**(1)	**.056**(1)	**.059**(1)	**.044**(1)
WAUT11	.269	.284	.311	.263	.291	.309
WANG12	.276	.261	.236	.284	.275	.216

5. Discussion and conclusions

This research has given an overview of the metaheuristic solution procedures available in the literature to solve the multi-mode resource-constrained project scheduling problem and has made a fair comparison between these procedures. Moreover, a new benchmark dataset is proposed. Researchers are encouraged to use this dataset to compare the results of their solution procedures with other procedures.

Based on the results, the following three conclusions can be made. First, the MRCPSP can be divided into two subproblems: the mode assignment problem, whose aim is to generate a feasible mode assignment list, and a scheduling problem, whose aim is to minimize the makespan of the problem. Some procedures mainly focused on the scheduling problem, by which these procedures were not able to generate feasible solutions for all project instances. Moreover, procedures using a thought-out mode assignment procedure to generate the initial population, such as the minimum normalized resources procedure of Lova et al. (2009) or the controlled mode assignment procedure of Van Peteghem and Vanhoucke (2011), have an advantage compared to other methods. Future research will have to focus on this mode assignment problem, since for an increasing number of activities and an increasing number of modes the complexity of this problem also increases.

Second, the good performance of some procedures is mainly due to the applied MRCPSP-specific local search procedures rather than to the followed metaheuristic search strategy. This conclusion is supported by computational tests executed by several authors comparing the effectiveness of both their algorithm and a similar solution procedure without the proposed local search method. Lova et al. (2009) proved that their hybrid genetic algorithm with an efficient improvement method clearly outperforms a simple genetic algorithm. Similar results are found by Hartmann (2001), who showed that the use of the single-pass improvement method clearly improves the performance of the proposed genetic algorithm. Moreover, the use of problem specific information in the local search process increases the efficiency of the procedure significantly.

A final contribution is made by making a fair comparison between the different available metaheuristic solution procedures. We believe that the introduction of the new MMLIB dataset opens the possibility to compare new solution procedures with the currently available methods. We hope that the introduction of this new dataset will also stimulate the development of new ideas and techniques to solve the MRCPSP.

Acknowledgments

The computational resources and services used in this work were provided by Ghent University, the Hercules Foundation and the Flemish Government.

References

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*, 614–626.

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

Bedworth, D., & Bailey, J. (1982). *Integrated production control systems – Management, analysis, design*. New York: Wiley.

Blazewicz, J., Lenstra, J., & Rinnooy Kan, A. (1983). Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics, 5*, 11–24.

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

Boctor, F. (1996a). An adaption of the simulated annealing for solving resource-constrained project scheduling problems. *International Journal of Production Research, 34*, 2335–2351.

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

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*, 268–281.

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*, 3–41.

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

Coelho, J., & Vanhoucke, M. (2011). Multi-mode resource-constrained project scheduling using RCPSP and SAT solvers. *European Journal of Operational Research, 213*, 73–82.

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

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

Demeulemeester, E., Vanhoucke, M., & Herroelen, W. (2003). A random network generator for activity-on-the-node networks. *Journal of Scheduling, 6*, 13–34.

Drexl, A., & Grünewald, J. (1993). Nonpreemptive multi-mode resource-constrained project scheduling. *IIE Transactions, 25*, 74–81.

Elloumi, S., & Fortemps, P. (2010). A hybrid rank-based evolutionary algorithm applied to multi-mode resource-constrained project scheduling problem. *European Journal of Operational Research, 205*, 31–41.

Elmaghraby, S. (1977). *Activity networks: Project planning and control by network models*. New York: John Wiley and Sons.

Glover, F., & Kochenberger, G. A. (2003). *Handbook of metaheuristics*. Kluwer Academic Publishers.

Hartmann, S. (2001). Project scheduling with multiple modes: A genetic algorithm. *Annals of Operations Research, 102*, 111–135.

Hartmann, S., & Drexl, A. (1998). Project scheduling with multiple modes: A comparison of exact algorithms. *Networks, 32*, 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*, 394–407.

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*, 299–308.

Józefowska, J., Mika, M., Rózycki, R., Waligóra, G., & Weglarz, J. (2001). Simulated annealing for multi-mode resource-constrained project scheduling. *Annals of Operations Research, 102*, 137–155.

Kelley, J. Jr., (1963). *The critical-path method: Resources planning and scheduling*. New Jersey: Prentice-Hall.

Knotts, G., Dror, M., & Hartman, B. (2000). Agent-based project scheduling. *IIE Transactions, 32*, 387–401.

Kolisch, R. (1995). *Project scheduling under resource constraints – Efficient heuristics for several problem classes*. Ph.D. thesis Physica, Heidelberg.

Kolisch, R., & Drexl, A. (1997). Local search for nonpreemptive multi-mode resource-constrained project scheduling. *IIE Transactions, 29*, 987–999.

Kolisch, R., & Hartmann, S. (1999). Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis. *European Journal of Operational Research, 112*, 3–41.

Kolisch, R., & Hartmann, S. (2006). Experimental investigation of heuristics for resource-constrained project scheduling: An update. *European Journal of Operational Research, 174*, 23–37.

Kolisch, R., & Sprecher, A. (1996). PSPLIB – A project scheduling problem library. *European Journal of Operational Research, 96*, 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*, 1693–1703.

Lova, A., Tormos, P., & Barber, F. (2006). Multi-mode resource constrained project scheduling: Scheduling schemes, priority rules and mode selection rules. *Inteligencia Artificial, 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*, 302–316.

Mastor, A. (1970). An experimental and comparative evaluation of production line balancing techniques. *Management Science, 16*, 728–746.

Mori, M., & Tseng, C. (1997). A genetic algorithm for the multi-mode resource constrained project scheduling problem. *European Journal of Operational Research, 100*, 134–141.

Özdamar, L. (1999). A genetic algorithm approach to a general category project scheduling problem. *IEEE Transactions on Systems, Management and Cybernetics, 29*, 44–59.

Özdamar, L., & Ulusoy, G. (1994). A local constraint based analysis approach to project scheduling under general resource constraints. *European Journal of Operational Research, 79*, 287–298.

Pascoe, T. (1966). Allocation of resources – CPM. *Revue Française de Recherche Opérationnelle, 38*, 31–38.

Patterson, J., Slowinski, R., Talbot, F., & Weglarz, J. (1989). An algorithm for a general class of precedence and resource constrained scheduling problem. In R. Slowinsky & J. Weglarz (Eds.), *Advances in project scheduling* (pp. 3–28). Amsterdam: Elsevier.

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*, 35–48.

Slowinski, R. (1980). Two approaches to problems of resource allocation among project activities – A comparative study. *Journal of Operational Research Society, 8*, 711–723.

Slowinski, R. (1981). Multiobjective network scheduling with efficient use of renewable and non-renewable resources. *European Journal of Operational Research, 7*, 265–273.

Slowinski, R., Soniewicki, B., & Weglarz, J. (1994). DSS for multiobjective project scheduling. *European Journal of Operational Research, 79*, 220–229.

Speranza, M., & Vercellis, C. (1993). Hierarchical models for multi-project planning and scheduling. *European Journal of Operational Research, 64*, 312–325.

Sprecher, A. (1994). *Resource-constrained project scheduling: Exact methods for the multi-mode case*. Springer.

Sprecher, A., & Drexl, A. (1998). Multi-mode resource-constrained project scheduling with a simple, general and powerful sequencing algorithm. *European Journal of Operational Research, 107*, 431–450.

Sprecher, A., Hartmann, S., & Drexl, A. (1997). An exact algorithm for project scheduling with multiple modes. *OR Spektrum, 19*, 195–203.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. John Wiley and Sons, Inc.

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

Tormos, P., & Lova, A. (2001). A competitive heuristic solution technique for resource-constrained project scheduling. *Annals of Operations Research, 102*, 65–81.

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

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

Vanhoucke, M., Coelho, J., Debels, D., Maenhout, B., & Tavares, L. (2008). An evaluation of the adequacy of project network generators with systematically sampled networks. *European Journal of Operational Research, 187*, 511–524.

Van Peteghem, V., & Vanhoucke, M. (2011). Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. *Journal of Heuristics* http://dx.doi.org/10.1007/s10732-010-9152-0.

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

Wang, L., & Fang, C. (2012). An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem. *Computers & Operations Research, 39*, 449–460.

Wauters, T., Verbeeck, K., Vanden Berghe, G., & De Causmaecker, P. (2011). Learning agents for the multi-mode project scheduling problem. *Journal of the Operational Research Society, 62*, 281–290.

Weglarz, J., Jozefowska, J., Mika, M., & Waligora, G. (2011). Project scheduling with finite or infinite number of activity processing modes – A survey. *European Journal of Operational Research, 208*, 177–205.

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

Zhu, G., Bard, J., & Tu, G. (2006). A branch-and-cut procedure for the multimode resource-constrained project-scheduling problem. *Journal on Computing, 18*, 377–390.