ARTÍCULO

Multi-Mode Resource Constrained Project Scheduling: Scheduling Schemes, Priority Rules and Mode Selection Rules

Antonio Lova, Pilar Tormos, Federico Barber

Universidad Politécnica de Valencia
Camino de Vera s/n
Valencia, 46022
{allova,ptormos}@eio.upv.es
fbarber@dsic.upv.es

Abstract

This work deals with the multi-mode resource-constrained project scheduling problem. In this problem, activities of the project may be executed in more than one operating mode and renewable resource constraints exist. Each activity operation mode has a different duration and requires different amount of renewable resources. The objective function considered is the minimisation of the project completion time. Heuristics based on priority rule are considered as solution procedures for this problem. Both, parallel and serial scheduling generation schemes are described in the multi-mode context. On the basis of a well-known set of project instances, the new heuristics based on priority rules are evaluated against the best ones published. Finally, multi-pass heuristics based on priority rules that outperforms the deterministic multi-pass heuristic previously published are designed.

Keywords: Project Management and Scheduling, Multi-mode, Resource-Constrained Project Scheduling, Heuristics based on priority rule

1. Introduction

The classical resource-constrained project scheduling problem (RCPSP) involves the scheduling of a project to minimise its total duration subject to precedence relationships and constant availability constraints on the required set of renewable resources (as machines or manpower). In this case each activity may be characterised by a unique duration and a singular collection of resource requirements that have to be available each time period the activity is being executed.

Recently efforts have been addressed to formulate and to solve multi-mode resource-constrained project scheduling problem (MRCPSP) where activity duration is a discrete function of resources.

The problem is one of the most general and most difficult project scheduling problems and as a generalisation of the well-known job shop problem it belongs to the class of the NP-Hard problems [Blazewicz et al.83]. The currently most powerful optimisation procedures are unable to optimally solve highly resource-constrained projects with more than 20 activities and three execution modes per

activity within reasonable computation times[Sprecher et al.98]. Hence, in practice heuristics to generate near-optimal schedules for large and highly constrained projects are needed.

Heuristics based on priority rule have been one of the most important solution techniques for resource constrained project scheduling problems with renewable resources. This is due to the fact they are easy to understand and to implement, they are fast in terms of computational effort, and obtain acceptable results even for large sized projects [Kolisch 96]. Furthermore they are contained in commercial software packages. However, the solution quality obtained by such proprietary heuristics often is not satisfying [Maroto et al.99]. Since modern packages provide the possibility of including own procedures comfortably by incorporated programming languages, more efficient, easy to implement priority rule-based procedures are of great interest for project management software developers and users[Klein 00].

On the other hand and although priority rule based methods to solve the MRCPSP do not always give the best results, they are indispensable when solving large problem instances quickly. Therefore, good priority rule-based methods are needed to determine the initial solutions for metaheuristic procedures such as Simulated Annealing (SA), Tabu Search (TS) and Genetic Algorithms (GA). However, to the best of our knowledge, there are no studies that simultaneously analyse the effect on the project completion time of the components of the heuristics based on priority rule for the MRCPSP: Schedule Generation Schemes, Priority Rule and Mode Selection Rules. With this idea in mind, the main objective of this work is to fill the existing gap evaluating the effect of both scheduling generations schemes proposed in the RCPSP (serial and parallel), priority rules and mode selection rules in the context of Multi-mode Resource-Constrained Project Scheduling (MRCPSP).

The outline of the remaining is as follows: section 2 shows a review of the MRCPSP and the solving methods. In section 3 the components of the heuristics based on priority rule for the MRCPSP are described: the serial and the parallel scheduling generation scheme, the priority rules and the mode selection rules. In addition, a generalisation of the best mode selection rule discussed in the literature is reported. In section 4 the performance of the heuristics developed is evaluated against the best published deterministic heuristics based on priority rules. Results show that the deterministic heuristics developed in this work greatly outperforms the ones previously published. The main conclusions are drawn in section 5.

2. Problem formulation and solution procedures

A project can be depicted by an acyclic activity-on-node graph where activities are numerically labelled such that successor activities have higher numbers (labels) than all their predecessors. Associated with each activity is a set of possible durations and the corresponding resource requirements which would permit the activity to be completed in the stated durations. Each duration-resource combination is a "mode". If resources are available in limited quantities each time period, the resources are considered renewable.

The problem considered in this paper deals with the resource constrained project scheduling in which activities (j=1...J) may have more than one execution mode and renewable resource constraints exist. Different activity execution modes (m=1...Mj) require different amounts or different types of resources and represent alternative ways of realising an activity. Each activity mode thus specifies a non preemptable unique activity duration (d_{jm}) and a set of requirements for renewable resources. The objective to consider is to minimise the project completion time.

To model the multi-mode multiple resource-constrained project scheduling problem (MRCPSP) with renewable constrained resources, we assume that activities are topologically ordered, i.e. each activity j has an activity number (label) that is larger than the number of all its immediate predecessors $i \in Pj$. We also assume that we have a unique dummy start and finish activities, j=1 and j=J, each only performable in a single mode together with zero duration and zero resource requirement, respectively. For all other activities we assume that modes are sorted in the order of non-decreasing duration.

Given an upper bound T of the project´s makespan (e.g. $\sum_{j=1}^{J} max\{d_{jm} \mid m = 1,...,Mj\}$), the earliest and latest finish times, EFT_j and LFT_j can be calculated performing a traditional forward pass assigning to each activity its shortest mode and a traditional backward pass after assigning LFT_J=T. Defining the variables of the model as binary variables x_{jmt} which equal 1 if activity j is scheduled in mode m ($1 \le m \le Mj$) to finish at t, 0 otherwise, the problem can be formulated as follows[Talbot 82]:

$$Minimise \sum_{m=1}^{M_J} \sum_{t=EFTJ}^{LFT_J} t\, x_{Jmt} \quad (1)$$

$$\sum_{m=1}^{M_j} \sum_{t=EFT_j}^{LFT_j} x_{jmt} = 1, \quad j = 1,...,J \quad (2)$$

$$\sum_{m=1}^{M_i} \sum_{t=EFT_I}^{LFT_i} t\, x_{imt} \le \sum_{m=1}^{M_j} \sum_{t=EFT_j}^{LFT_j} (t - d_{jm}) x_{jmt}, \quad j = 2,...,J, \quad i \in P_j \quad (3)$$

$$\sum_{j=2}^{J-1} \sum_{m=1}^{M_j} r_{jmk} \sum_{\tau=t}^{t+d_{jm}-1} x_{jm\tau} \le R_k, \quad k \in R, \quad t = 1,...,T \quad (4)$$

$$x_{jmt} \in \{0,1\}, \quad j=1,...,J, \quad m=1,...,M_j, \quad t=EFT_j,...,LFT_j \quad (5)$$

Constraint set (2) ensures that each activity j is performed in one of its modes and is finished within its time window $[EFT_j,...,LFT_j]$. Constraints (3) insure that precedence relationships are maintained. Here P_j is the set of all immediate predecessor activities of activity j and d_{jm} is the duration of activity j operating in mode m. Renewable resource restrictions are enforced by constraints (4). Resource k is available in R_{kt} units in period t. Activity j required the use of r_{jmk} units of renewable resource k when operating in mode m. The definition of all decision variables as binary is considered in constraints (5). Finally although other objectives could be considered [Slowinski 89] the objective function considered in this work is expression (1), that is, to minimise the project completion time.

The feasible scheduled start time (S_j) and the feasible scheduled finish time (F_j) of each activity with the minimum feasible completion time of the project are obtained as solution of this model. This model can be solved to optimality using the exact method of Talbot that was the first to present an enumeration scheme to solve the MRCPSP. The computational experience reported in this work for a series of test problem indicates that the methodology can provide optimal solutions to small problems and good heuristic solutions to larger problems in a cost-effective manner. Additional exact approaches for solving this problem have been proposed by [Patterson et al.89], [Sprecher et al.97], [Sprecher et al.98] and [Demeulemeester et al.00].

[Patterson et al.89], propose an enumerative type of branch and bound algorithm based on the generation of a precedence tree that guides the search for solutions. Sprecher et al. [Sprecher et al.97] extend the concept of delaying alternatives introduced by Demeulemesteer and Herroelen [Demeulemeester et al.92] for the single-mode RCPSP, and define the concepts of delay and mode alternatives. [Sprecher et al.98] present an exact procedure of the branch and bound type in which the enumeration scheme is enhanced by search tree reduction schemes. On the other hand, Demeulemeester et al. [Demeulemeester et al.00] present a depth-first branch-and-bound procedure for the discrete time/resource trade-off problem in project networks. In this work, it is considered that in real-life projects, it often occurs that only one renewable resource is available in limited amount throughout the project.

Despite the encouraging results obtained in the mentioned works, it has to be recalled that exact algorithms in general fail to solve problems with more than 20 activities. Hence, even for small problems they are of limited use leaving heuristics as alternative.

Heuristic solution procedures have been proposed by Drexl and Grünewald [Drexl et al.93], Özdamar and Ulusoy [Özdamar et al. 94], Boctor [Boctor 93] [Boctor 96a] [Boctor 96b], Kolisch and Drexl [Kolisch et al. 96], Özdamar [Özdamar 99] and more recently by Hartmann [Hartmann 01] and Lova and Tormos [Lova et al.01b]. In addition the solution procedure of Talbot [Talbot 82] as well as that devised by Patterson et al. [Patterson et al.89] can be applied as heuristics truncating the search procedure by imposing time limits.

Drexl and Grünewald [Drexl et al.93] develop models for formulating nonpreemptive project scheduling problems with general resource availabilities and requirements as well as multi-mode time resource tradeoffs. For the solution of this model, a stochastic scheduling method is presented which outperforms traditional deterministic scheduling rules.

Özdamar and Ulusoy [Özdamar et al. 94] present a constraint-based approach to solve the MRCPSP with |NR|=1. They employ a parallel scheduling method to decide via so-called essential conditions which activity-mode pairs have to be scheduled. For each combination the increase over a lower bound of the project´s makespan is calculated and the combination that induces the smallest increase over the lower bound is then scheduled.

Kolisch and Drexl [Kolisch et al. 97] propose a local search heuristic with a multi-pass approach consisting of three phases. Phase I finds an initial feasible solution, Phase II, derives an improved

project completion time with a near-optimal RCPSP heuristic and in Phase III, they generate and evaluate 30 schedules for the best mode assignment obtained in phase II with the sampling heuristic developed in Kolisch and Drexl [Kolisch et al. 96]. As renewable and nonrenewable resources are considered in these problems, the computational experience developed shows that the truncated branch and bound procedure of Talbot [Talbot 82] and the heuristic of Drexl and Grünewald [Drexl et al.93] are not able to find a feasible solution to all the project instances considered. Nevertheless, the heuristic of Kolisch and Drexl [Kolisch et al. 97] is presented as the first method able to find a feasible project schedule to all project instances.

Metaheuristic solution strategies for the MRCPSP have been developed by Boctor [Boctor 93], Mori and Tseng [Mori et al.97], Özdamar [Özdamar 99] and Hartmann [Hartmann 01]. In Boctor [Boctor 93] a SA algorithm with the minimum slack as priority rule considering renewable resources is presented. The algorithm is able to handle single-mode and multi-mode problems and to optimise different objective functions. Mori and Tseng [Mori et al.97] propose a GA and compare it against the stochastic scheduling method developed by Drexl and Gruenewald [Drexl et al.93]. Özdamar [Özdamar 99] presents a GA based on an encoding which is made up by a sequence of priority rules and a mode assignment. Finally, Hartmann [Hartmann 01] presents a GA based on an activity list representation successfully employed by Hartmann [Hartmann 98] for the RCPSP. The two latter metaheuristics solve the MRCPSP considering both renewable and nonrenewable resources.

In Lova and Tormos [Lova et al.01b], a hybrid heuristic method that combines random sampling with a new Multimode Backward-Forward method (that reduces the project completion time) is developed for the multi-mode resource constrained project scheduling with both renewable and nonrenewable resources types. This heuristic solution technique greatly outperforms the heuristics and metaheuristics previously published.

On the other hand, the work of Boctor [Boctor 96a] is the unique that describes and analyse the single-pass heuristics based on priority rule for the MRCPSP when only renewable resource types are considered. In this work, the author proposes several heuristics based on priority rule with a time oriented scheduling approach that can be considered a modified parallel scheduling scheme. Several priority rules to sort the activities in the decision set and to select the mode of the activities are evaluated. Among the best heuristics reported in the computational experience, the ones based on the Minimum Latest Finish Time and the Minimum Slack rules (with a dynamic approach) with the shortest duration as criterion to select the activity mode have the best performance. In Boctor [Boctor 96b] two-pass heuristics based on forward and backward scheduling and a heuristic that enumerates schedulable combinations of activities and chooses from them the one having the best value for an evaluation criterion are proposed. The latter heuristic outperforms the deterministic heuristics previously appeared in the open literature for this problem.

3. Heuristics based on priority rule for the MRCPSP

Priority rule-based heuristics for the MRCPSP combine scheduling generation schemes (SGS), mode selection rules and priority rules in order to construct a specific algorithm. Two different schemes can be distinguished: the Serial Scheme and the Parallel Scheme. Both build feasible project schedules by stepwise extension of a partial schedule -a schedule where only a subset of activities has been scheduled. Activities that, in each step, compete for resources are sorted by a priority rule and a mode selection rule is applied in order to assign an execution mode to each activity which determines the duration and resources consumption.

3.1. Schedule Generation Schemes (SGS)

The **Serial Schedule Generation Scheme (S-SGS)** was proposed by Kelley [Kelley 63] for the RCPSP and we generalise it to the MRCPSP with renewable resources. It is an activity oriented scheme and consists of *J* stages, in each of which *one activity* with its assigned *mode* is selected and scheduled at the earliest precedence and resource feasible completion time. There are two disjoint activity sets associated with each stage: the scheduled set (Sn) that includes the activities already scheduled, and the decision set (Dn) that consists of the activities eligible for scheduling, that is, unscheduled activities all of whose predecessors are in Sn.

In each stage, an activity is selected from Dn according to the order established by a *priority rule* and an execution mode is chosen according to a *mode selection rule.* This activity is scheduled as soon as possible taking into account the precedence relationships of the activities and the capacity of the resources. The activity selected is removed from Dn and added to Sn. Additionally, Dn is updated with the immediate successors of the activity just scheduled all of whose predecessors have been

scheduled. The S-SGS finishes when all activities are in Sn.

The **Parallel Schedule Generation Scheme (P-SGS)** was proposed by Kelley [Kelley 63] for the RCPSP and we generalise it to the MRCPSP with renewable resources. It is a time oriented scheme and consists of at most *J* stages. In each stage *a set of activities* -which might be empty- is scheduled. Each stage *j* has associated a schedule time t_j where $t_m < t_j$ for $m < j$, and three disjoint sets of activities: the decision set (Dj), the complete set (Cj) and the in-process set (Pj).

On account of the schedule time t_j, the set of scheduled activities -from the serial scheme- is now divided into two subsets. The set Cj with the activities already scheduled and completed up to the schedule time, and the set Pj that includes the activities already scheduled but not completed yet. Additionally, we have the set Dj that in contrast with the serial method contains all yet unscheduled activities that are available for scheduling w.r.t. precedence and resource constraints.

The partial schedule obtained each iteration is composed of the set Cj∪Pj. The schedule time in a stage is the earliest finish time of the activities scheduled during all earlier stages.

Each stage is made up of two steps:

Step 1. The new schedule time is determined and activities with a finish time equal to the new schedule time are removed from Pj and put into Cj. This may place a number of activities into Dj.

Step 2. One activity from Dj is selected with a *priority rule* and one mode of this activity is selected with a *mode selection rule*. The selected activity is scheduled to start at the current schedule time taking into account the availability of the resources. Afterwards, this activity is removed from Dj and put into Pj. Step 2 is repeated until Dj is empty, i.e. activities were scheduled or are no longer available for scheduling w.r.t. resource constraints.

The parallel scheme finishes when all activities are in the set Cj∪Pj.

There are few studies analysing the performance of the heuristics based on priority rules depending on the schedule generation scheme, and all of them in the RCPSP context. Valls, Perez and Quintanilla [Valls et al.92], after a computational experience carried out with projects of 1000 and 1500 activities, conclude that heuristics based on priority rule frequently obtain better performance with the parallel scheduling scheme. Nevertheless, the serial scheme can outperform the parallel scheme in projects with certain characteristics. In fact, Kolisch [Kolisch 96] points out that the serial scheduling scheme is superior for large sized projects and for instances that are only moderately resource-constrained when minimising makespan in the RCPSP. More recently, Lova and Tormos [Lova et al.01a] perform an analysis of these types of heuristics in the multiproject context concluding that the parallel scheduling scheme is superior than the serial scheme when the time criteria mean project delay and multiproject duration are minimised.
However, there are no works that for the MRCPSP analyse the performance of the Scheduling Generation Schemes when the same priority rule and the same mode selection rule are used and only in the works of Boctor [Boctor 96a][Boctor 96b] a time oriented scheduling generation scheme is used.

3.2. Priority rules

Activities in D are sorted each iteration of the SGS according to the priority rule value. Seven priority rules for the MPCPSP are evaluated by Boctor [Boctor 96a]: Minimum Latest Finish Time (LFT), Minimum Total Slack (SLK), Maximum remaining work (RWK) considered as the sum of the shortest possible duration of the activity and all its successors, Maximum number of subsequent candidates (CAN) considering a subsequent candidate to an activity *x* is an activity which becomes or remains schedulable if *x* is scheduled to start at the considered period, Maximum number of immediate successors (NIS), Longest Processing Time (LPM) and Shortest Processing Time (SPM). In the computational experience reported in this work the rules that obtain the best performance regardless of the mode selection rule used are LFT, SLK, RWK and CAN. If ties occur the activity with the smallest activity number is selected. It is important to note that the rules LFT, SLK and CAN have been coded with a *dynamic* approach (the priority values are updated in each stage of the scheduling process). Nevertheless, the rules NIS, LPM and SPM are coded as *static* priority rules (priority values are calculated only once before starting the scheduling process). Obviously, the dynamic rules require higher computational effort (CPU-time) than the static rules in order to calculate dynamically the priority values. We have tested that a heuristic based on priority rule with a dynamic approach requires five times more computational effort than when a static heuristic based on priority rule is used. Therefore, good static rules (with low

computational effort) are preferred to be included in more sophisticated heuristics based on random sampling or in metaheuristics.

3.3. Mode selection rules

In a stage of a SGS when an activity is selected to be scheduled, a mode selection rule is necessary for determining the duration and the resource requirements of the activity. In [Boctor 96a] three mode selection rules are tested: Shortest feasible mode (SFM), Least criticality ratio or least critical resource (LCR) and Least resource proportion (LRP). As a main conclusion of the computational experience, it is pointed out that the rule SFM with the priority rules previously described greatly outperforms the heuristics based on the mode selection rules LCR or the rule LRP.

The three mode selection rules are used by [Boctor 96a] with the time oriented scheduling generation scheme developed in that work. Nevertheless, an extension of the SFM rule to the S-SGS is needed.

When the S-SGS is used the rule SFM rule applied with the P-SGS can be easily generalised instead using the Earliest Feasible Finish Time (EFFT). This mode selection rule selects for each activity the execution mode such that it is scheduled with the feasible finish time as early as possible. Therefore, in the S-SGS the following activities to be processed could be executed earlier. If ties occur (an activity has several feasible modes with the same minimum value of feasible finish time) the mode with the highest duration is selected. This selection implies that other activities in the serial scheduling process can be executed earlier. The use of the EFFT mode selection rule in the P-SGS produces the same results than when the rule SFM is applied.

4. Computational experience

A set of instances is necessary in order to compare the performance of the heuristics based on priority rule for the MRCPSP with renewable resources. In http://129.187.106.231/psplib/dataob.html a set of test problems is available. This set consists of two main subsets of randomly generated problems and used in Boctor [Boctor 93][Boctor 96a][Boctor 96b]. The first subset contains 120 instances of 50 non-dummy activities. To generate these instances, 50 activity networks with an average of two immediate successors for each activity were generated. Then each network was used to generate six different problems with one, two or four resource types. The second main subset, composed of 120 instances of 100 non-dummy activities was generated similarly. For each of these 240 instances, activity data was generated in the following manner. The number of execution modes is derived from a uniform distribution from 1 to 4. The execution time of the first (shortest) mode duration was derived from a uniform distribution from 1 to 15 while the execution time of consecutive modes is derived from the interval: (preceding mode duration, preceding mode duration + 15/number of modes). To determine the resource requirements corresponding to each mode, we used a uniform distribution from 1 to 5 with the restriction that a weighted average of resource consumption is about the same for all execution modes. Finally, resource capacities were calculated as average resource requirement plus 40% of the difference between the resource utilisation peak (when activities are executed in the shortest mode and at their early start date) and the average requirement.

The main evaluation criterion is the average deviation with respect to the lower bound that supposes the critical path length (calculated by using the shortest mode duration of the project activities). Other evaluation criteria to be used are the maximum percentage increase above the critical path length and the minimum percentage increase above the critical path length. When comparing the performance obtained by the heuristics developed to evaluate the statistical significance, the Wilcoxon signed rank test is applied. This non-parametric test has the advantage that we do not need to make any assumption about the distribution of these results.

Our computational experience has been performed on a Pentium-based IBM-compatible personal computer with 600 MHz clock-pulse and 64 MB RAM. The heuristic proposed in this work has been coded in Visual C++ ver 1.0 under Windows 98.

4.1. Single-pass heuristics based on priority rule

As we have described in the previous section, a heuristic based on priority rule for the MRCPSP with renewable resources has tree components: a SGS, a priority rule in order to sort Dn and a mode selection rule. Table 1 presents the priority rules considered within our computational experiments (14 priority rules). In the first column, the name of each rule is given. The second column (Extr.) denotes whether the rule selects an activity with the smallest (min) or the largest (max) priority value p_j. The last column shows how the priority value p_j of an activity j is computed.

The majority of the priority rules shown in table 1 have been tested for the RCPSP being the rules LFT

or SLK the ones that in general obtain the best performance. In the MRCPSP with renewable resources, only SLK, LFT, NIS, RWK, LPT and SPT have been tested by Boctor [Boctor 93], (all with a time oriented scheduling generation scheme) being the rules SLK, LFT and RWK the ones with the best performance. In addition to the classical priority rules shown in table 1, we have included a combined rule where the priority value p_j is calculated by adding the values of two well known priority rules: the rules LFT and LST. We have called this new priority rule Latest start and finish time (LSTLFT).

Furthermore, the rules described in table 1 have been coded with a static approach that is, the priority values are calculated only once before starting the scheduling process. In this way the construction of feasible schedules is very fast and the best heuristics can be included in more sophisticated techniques as random sampling heuristics or metaheuristics. For each heuristic tested we use the rule Earliest Feasible Finish Time (EFFT) as mode selection rule that can be used in both P-SGS and S-SGS.

Considering the combination of the 14 priority rules with the two scheduling generation schemes: P-SGS and S-SGS, we will analyse the performance of 28 single-pass heuristics based on priority rules. If ties occur, the FCFS rule is used. The results of these heuristics are shown in tables 2 and 3.

If the P-SGS is the scheduling scheme used, from the performance analysis of the priority rules, we can conclude that those with the best performance are LSTLFT, LST, LFT and RWK with a percentage deviation with respect to the lower bound of 38.3%, 38.8%, 38.9% and 39.4% respectively (table 2). The priority rule with the lowest average deviation is LSTLFT. The difference between this result and those achieved by the rules LST and LFT is not statistically significant (according to the Wilcoxon signed rank test) and a p-value of 0.2327 and 0.0978 respectively is obtained. The LSTLFT rule (proposed in this work) has a better performance than the rule RWK with a p-value of 0.0059. The next best priority rules are FREE and SLK that have the same average deviation (40.8%). The remaining priority rules have significant worse results than the one previously analysed.

Table 3 shows the performance of the priority rules tested when the S-SGS is used. The mean ordering is similar to the one reported for the P-SGS. Again the best priority rules are LSTLFT, LST, LFT and RWK with a percentage deviation with respect to the lower bound of 34.3%, 34.7%, 34.7% and 35.5% respectively. Once more the rule LSTLFT is the one with the lowest average deviation. The rules LSTLFT and LST are not statistically different (according to the Wilcoxon signed rank test), but the rule LSTLFT outperforms the rule LFT with a p-value of 0.0193. With the S-SGS, these three priority rules have a performance significantly better than the rule RWK with a p-value of 0.012 when the LST and the RWK are considered. The next best priority rules are again the rules FREE and the SLK yielding an increase in the average deviation with respect to the rule RWK of 3.8%. The remaining priority rules have a performance significantly worse than the previously analysed.

Obviously, in the MRCPSP the S-SGS requires a computational effort higher than that of P-SGS due to the fact that in the S-SGS for each selected activity of Dn, all modes are tested in order to select that with the minimum feasible finish time. For a given priority rule when the P-SGS is used the mean CPU-time in seconds is around 0.002 seconds while it is of 0.003 seconds when the S-SGS is used.

Despite this increase in the computational requirements the results reported in table 4 show that the S-SGS leads to greatly better results than those yielded by the P-SGS for nearly all priority rules tested. Indeed this better performance is maintained for all the priority rules evaluated except for the rule SPT for which the difference of using S-SGS or P-SGS is not statistically significant (although the average deviation is lower when the S-SGS is used). When analysing the ranking of the priority rules it is remarkable that the same set of rules are in the top both for P-SGS and S-SGS. These rules are LSTLFT, RWK, LST and LFT that when applied with the S-SGS obtain an average deviation that is significantly lower than that achieved when the P-SGS is used. Concretely, the reductions are 3.9%, 3.9%, 4.1% and 4.2% respectively.

The performance of the best single-pass heuristics based on priority rules is compared with the best single-pass heuristics developed by Boctor [Boctor 96a]. In table 5 the minimum, maximum, standard deviation and mean deviation with respect to the lower bound are shown for each heuristic when applied to each set of instances with different characteristics. The best single-pass heuristics based on priority rule analysed in Boctor [Boctor 96a] are SLK, RWK, CAN, LFT which obtain an average deviation with respect to the lower bound of 39.2.%, 39.3%, 39.7% and 39.8% respectively. Considering the same context, that is assuming a time-oriented scheduling scheme (in our case P-SGS) the best

rules are LSTLFT, LST, LFT and RWK yielding average deviations of 38.3%, 38.8%, 38.9% and 39.4% respectively. These rules jointly with the rules LSTLFT and the EFFT as mode selection rules outperform the best single-pass heuristic obtained in [Boctor 96a]. According to the results reported in the analysis carried out, the best single-pass heuristic based on priority rule is S-SGS/LSTLFT/EFFT with an average deviation of 34.3%, that is 4.9% lower than the average deviation of the best heuristic analysed in Boctor [Boctor 96a].

Finally on account of the project characteristics, the heuristic S-SGS/LSTLFT/EFFT reduces the average deviation with respect to the heuristic SLK by Boctor [Boctor 96a] in 6%, 3.7% and 4% when instances with 50 activities and 1, 2 or 4 resource types respectively are considered. When project instances have 100 activities requiring 1, 2 or 4 resource types, the reduction in the average deviation is 6.1%, 5.1% and 4% respectively.

4.2. A multi-pass approach

The low computational effort needed in the single-pass heuristics based on priority rule has brought about the idea of performing several passes. There are many possibilities to combine SGS and priority rules into a multi-pass method. Each combination leads to different algorithms all of which report as solution the schedule with the feasible schedule with the lowest completion time. In this work among all the possibilities that can be analysed for obtaining a multi-pass method, we report those with the best performance.

We start the analysis of the multi-pass heuristic based on priority rule analysing the two-pass heuristics based on priority rules. In table 6 the minimum, maximum, standard deviation, mean deviation with respect to the lower bound and mean CPU-time in seconds are reported for different combination of SGS and priority rule. We have performed all the combinations between the four best priority rules (in all cases using S-SGS). In addition, we have considered the two-pass heuristics built by considering both SGS for each of the four best priority rules. The results are drawn in table 6. There are seven two-pass heuristics that obtain an average deviation with respect to the lower bound lower than 34%. The heuristic built by combining the rules LST and LSTLFT with S-SGS obtains the lowest average deviation (33.3%).

In table 7 the performance of the seven best two-pass heuristics of table 6 and the best two-pass heuristics based on priority rule developed in Boctor [Boctor 96b] are reported. Boctor [Boctor 96b] use the same priority rule with two scheduling approaches: Forward and Backward scheduling. Both approaches are different depending on the consideration of when an activity is schedulable. In Forward scheduling an activity is schedulable when all its predecessors have been scheduled while in Backward scheduling an activity is schedulable when all its successors have been scheduled. The heuristics analysed in [Boctor 96b] are based on the rules RWK, SLK, CAN and LFT that obtains average deviation of 36.7%, 36.8%, 36.9% and 37.3% respectively (table 7). A noteworthy result is that the best two-pass heuristic tested by Boctor [Boctor 96b] has an average deviation with respect to the lower bound that is 2.4% higher than the best single-pass heuristic developed in this work.

The best two-pass heuristic obtained in this work reduces the average deviation regarding the best two-pass heuristic of [Boctor 96b] in a range that varies from 3.4% when the two-pass heuristic S-SGS/LSTLFT/LFT is applied, to 2.9% when the heuristic used is either the combination of P/S-SGS/LSTLFT or S-SGS/LSTLFT/LST.

As we have pointed out, a multi-pass heuristic can be obtained with the combination between the 28 single-pass heuristics based on priority rule analysed in this work. This implies that there are many combinations that leads to three-pass, four-pass, and so on heuristics. From all combinations we have tested the four-pass heuristics built by considering S-SGS with the rules LSTLFT, LFT, LST and RWK as well as those four-pass heuristics considering the same rules but with P-SGS.

The results are conclusive, using the four-pass heuristic based on P-SGS the average deviation is 36.1% with a mean CPU-time of 0.008 seconds, whereas the average deviation is 32.6% for the same rules and S-SGS (the CPU-time is 0.012 seconds), that is 3.5% lower than one based on P-SGS.

Additionally, eight-pass heuristics can be considered by selecting the best schedules generated by the rules LSTLFT, LFT, LST and RWK using both S-SGS and P-SGS. The average deviation with respect to the lower bound of this heuristic is 32% with a mean CPU-time of 0.02 seconds.

In table 8, the performance of the best deterministic heuristic developed in [Boctor 96b] with the best single-pass heuristic (BEST1) S-SGS/LSTLFT/EFFT, the best two-pass heuristics

(BEST2) S-SGS/LSTLFT and S-SGS/LFT, the best four-pass heuristic (BEST4) S-SGS/LSTLFT, S-SGS/LFT, S-SGS/LST and S-SGS/RWK, as well as the eight-pass heuristic (BEST8) P/S-SGS/LSTLFT, P/S-SGS/LFT, P/S-SGS/LST and P/S-SGS/RWK are shown.

From the results reported can be concluded that the heuristic BEST1 obtains on average the same performance than that of Boctor [Boctor 96b] (with a lightly lower average deviation), whereas the remaining three multi-pass heuristics developed in this work outperform the more sophisticated heuristic by Boctor [Boctor 96b].

5. Conclusions

In this work heuristics based on priority rule for the MRCPSP with renewable resources are analysed. These methods are very important in the construction of more sophisticated heuristics as random sampling techniques or metaheuristics. Hence, additional efforts in order to obtain better heuristics based on priority rule are justified. Three components of this type of heuristics are analysed: scheduling generation scheme, priority rule and mode selection rule.

Based on a well-known set of 240 randomly generated project instances, it is shown that the single-pass and multi-pass heuristics based on priority rules developed in this work greatly outperforms the ones previously published. The serial schedule generation scheme greatly outperforms the parallel scheme with the majority of the priority rules tested justifying the higher computational effort required by the former. Finally, a multi-pass method that combines eight heuristics based on priority rule obtains the lowest average deviation with respect to the critical path length (32.0%), thus being the best deterministic heuristic for this problem. Furthermore the low computational requirements make it a powerful heuristic for the hard problem considered in this work.

The study carried out in this work can be extended to problems where the resource availability varies within time and in contexts where several projects must be concurrently executed.

Acknowledgements

We are grateful to the anonymous referees for their constructive comments and suggestions.
This work has been partially supported by the research projects TIN2004-06354-C02- 01 (Min. de Educación y Ciencia, Spain-FEDER), FOM-70022/T05 (Min. de Fomento, Spain) and by the Future and Emerging Technologies Unit of EC (IST priority-6th FP) under contract nº.FP6-021235-2 (project ARRIVAL).

References

[Blazewicz et al.83] Blazewicz, J., Lenstra J. and Rinnooy Kan A.H.G., Scheduling subject to resource constraints: classification and complexity, Discrete Applied Mathematics 5 (1983) 11-24.

[Boctor 93] Boctor, F.F., Heuristics for scheduling projects with resource restrictions and several resource-duration modes, International Journal of Production Research 31 (1993) 2547-2558.

[Boctor 96a] Boctor F.F., A new and efficient heuristic for scheduling projects with resource restrictions and multiple execution modes, European Journal of Operational Research 90 (1996) 349-361.

[Boctor 96b] Boctor F.F., Resource-constrained project scheduling by simulated annealing, International Journal of Production Research 34 (1996) 2335-2351.

[Demeulemeester et al.92] Demeulemeester, E. and Herroelen, W., A branch-and-bound procedure for the multiple resource-constrained project scheduling problem, Management Science 38 (1992) 1803-1818.

[Demeulemeester et al.00] Demeulemeester, E., De Reyck, B. and Herroelen, W., The discrete time/resource trade-off problem in project networks: a branch-and-bound approach, IIE Transactions 32 (2000) 1059-1069.

[Drexl et al.93] Drexl A. and Grünewald, J., Nonpreemptive multi-mode resource-constrained project scheduling, IIE Transactions 25 (1993) 74-81.

[Hartmann 98] Hartmann S., A competitive genetic algorithm for resource-constrained project scheduling. Naval Research Logistics 45 (1998) 733-750.

[Hartmann 01] Hartmann, S., Project scheduling with multiple modes: a genetic algorithm, Annals of Operations Research 102 (2001)111-135.

[Kelley 63] Kelley, J.E., The critical-path method: Resources planning and scheduling, in: Industrial Scheduling, Prentice Hall, New Jersey, 1963, pp. 347-365.

[Klein 00] Klein, R., Bidirectional planning: improving priority rule-based heuristics for scheduling resource-constrained projects 127 (2000) 619-638.

[Kolisch 96] Kolisch, R. Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation, European Journal of Operational Research 90/2 (1996) 320-333.

[Kolisch et al. 96] Kolisch, R. and Drexl, A.,

Adaptive search for solving hard project scheduling problems, Naval Research Logistics 43 (1996) 23-40.

[Kolisch et al. 97] Kolisch, R. and Drexl, A., Local search for nonpreemptive multi-mode resource-constrained project scheduling, IIE Transactions 29 (1997) 987-999.

[Lova et al.01a] Lova, A. and Tormos P., Analysis of Scheduling Schemes and Heuristic Rules Performance in Resource-Constrained Multiproject Scheduling, Annals of Operations Research 102 (2001) 263-286.

[Lova et al.01b] Lova A. and P. Tormos , A robust heuristic method for scheduling resource-constrained projects with multiple execution modes. In: Artiba, A., Martel, A. (Eds.), Proceedings of the International Conference on Industrial Engineering and Production Management, Université Laval and Facultés Universitaires Catholiques de Mons, Québec, Vol 2,(2001), pp. 778-787.

[Maroto et al.99] Maroto, C., Tormos, P. And Lova, A, The evolution of software quality in project scheduling. IN: Weglarz, J: (Ed.), Handbook on Recent Advances in Project Scheduling. Kluwer, Dordrecht, 1999, pp. 239-260.

[Mori et al.97] Mori, M and Tseng, C.C., A genetic algorithm for multi-mode resource constrained project scheduling problem, European Journal of Operational Research 100 (1997) 134-141.

[Özdamar 99] Özdamar L., A genetic algorithm approach to a general category project scheduling problem, IEEE Transactions on systems, man and cybernetics – part C 29 (1999) 44-59.

[Özdamar et al. 94] Özdamar, L. and Ulusoy G., A local constraint based analysis approach to project scheduling under general resource constraints, European Journal of Operational Research 79 (1994) 287-298.

[Patterson et al.89] Patterson, J.H., R. Slowinski, F.B. Talbot and Weglarz, J., An algorithm for a general class of precedence and resource constrained scheduling problems, In R. Slowinski and J.Weglarz, (eds), Advances in project scheduling (Amsterdam: Elsevier), chapter1, 1989, pp. 3-28

[Slowinski 89] Slowinski, R., Multiobjective project scheduling under multiple-category resource constraints., In R. Slowinski and J.Weglarz, (eds), Advances in project scheduling (Amsterdam: Elsevier), chapter7, 1989, pp. 151-167

[Sprecher et al.97] Sprecher, A, Hartmann, S. and Drexl, A., An exact algorithm for project scheduling with multiple modes, OR Spektrum 19 (1997) 195-203.

[Sprecher et al.98] Sprecher, A. and Drexl, A., Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm, European Journal of Operational Research 107 (1998) 431-450.

[Talbot 82] Talbot, B., Resource-constrained project scheduling with time-resource tradeoffs: the nonpreemptive case, Management Science 28 (1982) 1197-1210.

[Valls et al.92] Valls, V., M.A. Perez and M.S. Quintanilla, Heuristic performance in large resource-constrained projects, Working Paper, Nr.2-92. Departament d'Estadística i Investigació Operativa, Universitat de Valencia, Spain,1992.

Table 1. Priority rules tested

Priority rules	Extr.	Priority value p_j^*
Activity number (AN)	min	j
Earliest finish time (EFT)	min	EFT_j
Earliest start time (EST)	min	EST_j
Greatest rank positional weight (GRPW)	max	$d_j + (\Sigma\, d_i \;\forall\; i \in IS_j)$
Greatest resource demand (GRD)	max	$d_j\, (\Sigma\, r_{jk})$
Longest processing time (LPT)	max	d_j
Latest finish time (LFT)	min	LFT_j
Latest start and finish time (LSTLFT)	min	$LST_j + LFT_j$
Latest start time (LST)	min	LST_j
Minimum free slack (FREE)	min	$\min (EST_i \;\forall\; i \in IS_j) - EFT_j$
Minimum total slack (SLK)	min	$LST_j - EST_j$
Number of immediate successors (NIS)	max	$\lvert IS_j \rvert$
Remaining work (RWK)	max	$d_j + (\Sigma \min d_i \;\forall\; i \in AS_j)$
Shortest processing time (SPT)	min	d_j

* All parameters are calculated using the shortest duration mode for each activity.
(IS_j denotes the set of immediate successor activities of j, and AS_j all the successor activities of j)

Table 2. Performance of the heuristics based on priority rule tested (P-SGS)

	Average deviation with respect to the lower bound (%)			
Priority rules*	Min	Max	St.Dev	Mean
Latest start and finish time (LSTLFT)	**12.2**	**65.4**	**12.5**	**38.3**
Latest start time (LST)	**12.2**	**71.7**	**12.9**	**38.8**
Latest finish time (LFT)	**16.2**	**70.1**	**12.4**	**38.9**
Remaining work (RWK)	**8.8**	**68.7**	**12.3**	**39.4**
Minimum free slack (FREE)	15.7	76.9	13.3	40.8
Minimum total slack (SLK)	14.8	72.8	13.4	40.8
Activity number (AN)	19.3	73.9	12.9	43.5
Earliest start time (EST)	20.0	74.7	13.0	44.0
Number of immediate successors (NIS)	14.8	74.5	12.6	45.3
Longest processing time (LPT)	17.6	90.0	13.2	45.7
Greatest rank positional weight (GRPW)	16.6	80.1	12.9	45.8
Greatest resource demand (GRD)	18.7	84.6	13.3	45.9
Earliest finish time (EFT)	18.3	82.1	11.7	47.5
Shortest processing time (SPT)	19.3	80.4	12.4	48.3

* Mean CPU-time (seconds): 0.002

Table 3. Performance of the heuristics based on priority rule tested (S-SGS)

	Average deviation with respect to the lower bound (%)			
Priority rules*	Min	Max	St.Dev	Mean
Latest start and finish time (LSTLFT)	**7.9**	**62.0**	**13.2**	**34.3**
Latest finish time (LFT)	**7.9**	**64.8**	**12.9**	**34.7**
Latest start time (LST)	**9.9**	**61.2**	**13.2**	**34.7**
Remaining work (RWK)	**8.8**	**63.2**	**13.3**	**35.5**
Minimum free slack (FREE)	10.5	65.5	13.1	39.3
Minimum total slack (SLK)	9.9	68.3	13.4	39.4
Activity number (AN)	17.6	72.6	12.2	41.7
Earliest start time (EST)	17.0	72.2	12.5	42.5
Greatest rank positional weight (GRPW)	15.1	79.2	12.9	44.3
Longest processing time (LPT)	18.9	89.1	13.2	44.4
Number of immediate successors (NIS)	15.8	76.7	12.5	44.4
Greatest resource demand (GRD)	18.3	80.6	13.2	44.9
Earliest finish time (EFT)	21.0	75.6	11.6	45.3
Shortest processing time (SPT)	13.8	86.5	13.0	47.9

* Mean CPU-time (seconds): 0.003

Table 4. Parallel vs Serial Schedule Generation Scheme

Priority rules	Average deviation with respect to the lower bound (%) Mean		
	P	S	Diff.
Latest start and finish time (LSTLFT)	**38.2**	**34.3**	**3.9***
Latest finish time (LFT)	**38.9**	**34.7**	**4.2***
Latest start time (LST)	**38.8**	**34.7**	**4.1***
Remaining work (RWK)	**39.4**	**35.5**	**3.9***
Minimum free slack (FREE)	40.8	39.3	1.5*
Minimum total slack (SLK)	40.8	39.4	1.4*
Activity number (AN)	43.5	41.7	1.8*
Earliest start time (EST)	44.0	42.5	1.5*
Greatest rank positional weight (GRPW)	45.8	44.3	1.5*
Longest processing time (LPT)	45.7	44.4	1.3*
Number of immediate successors (NIS)	45.3	44.4	0.9*
Greatest resource demand (GRD)	45.9	44.9	1.0*
Earliest finish time (EFT)	47.5	45.3	2.2*
Shortest processing time (SPT)	48.3	47.9	0.4

* The S-SGS has a significantly better performance than the P-SGS according to the Wilcoxon signed rank test with a significant level of 0.00001

Table 5. Performance of single-pass heuristics based on priority rule

	Deviation with respect to the lower bound (%)											
	Best single-pass heuristics in [Boctor 96a]				Best single-pass heuristics in this paper							
	LFT	CAN	RWK	SLK	P-SGS (RWK)	P-SGS (LFT)	P-SGS (LST)	P-SGS (LSTLFT)	S-SGS (RWK)	S-SGS (LST)	S-SGS (LFT)	S-SGS (LSTLFT)
Set 1: 50 activities and 1 resource type	**25.8**	**26.0**	**26.9**	**25.5**	**26.6**	**25.4**	**24.9**	**25.1**	**20.5**	**20.0**	**20.6**	**19.5**
	16.8	10.5	8.9	12.2	8.8	16.2	12.2	12.2	9.4	9.9	7.9	7.9
	37.7	46.8	51.1	43.9	51.4	37.6	43.4	43.4	35.1	36.7	36.0	36.7
Set 2: 50 activities and 2 resource types	**39.2**	**40.6**	**40.0**	**39.5**	**39.1**	**38.1**	**39.7**	**38.8**	**36.7**	**36.3**	**35.8**	**35.8**
	25.8	27.0	24.2	24.7	22.3	24.4	26.9	20.1	20.0	22.3	21.4	19.5
	52.6	55.8	57.8	52.2	57.7	55.3	52.1	52.1	55.8	51.1	47.4	49.2
Set 3: 50 activities and 4 resource types	**51.9**	**51.0**	**50.5**	**51.6**	**51.2**	**51.8**	**51.3**	**50.3**	**48.6**	**48.0**	**48.1**	**47.6**
	39.8	36.9	35.6	35.6	37.3	36.1	35.5	35.5	30.1	32.3	31.8	32.3
	68.1	65.7	62.9	71.7	64.5	70.1	71.7	63.8	62.6	59.7	64.8	60.1
Set 4: 100 activities and 1 resource type	**26.1**	**25.2**	**25.1**	**24.2**	**25.3**	**25.3**	**23.4**	**23.4**	**20.1**	**18.5**	**18.9**	**18.1**
	18.1	17.3	17.5	14.1	16.8	17.6	15.3	15.5	8.8	10.9	11.3	10.1
	37.1	36.6	34.3	34.1	36.2	37.1	33.6	33.3	30.7	27.8	27.0	24.8
Set 5: 100 activities and 2 resource types	**42.5**	**42.4**	**42.1**	**42.0**	**42.9**	**41.6**	**41.1**	**41.1**	**38.5**	**37.1**	**36.9**	**36.9**
	28.8	28.1	28.7	29.4	28.5	26.9	30.4	30.4	26.1	22.3	25.7	22.3
	52.6	61.4	59.1	55.2	62.5	54.3	56.1	49.1	50.8	51.9	48.2	47.5
Set 6: 100 activities and 4 resource types	**52.2**	**52.2**	**51.1**	**51.8**	**51.2**	**51.0**	**51.3**	**51.3**	**48.7**	**48.3**	**47.8**	**47.8**
	43.0	40.3	41.3	41.2	41.6	39.0	41.1	41.1	39.1	39.2	38.9	40.3
	71.4	69.9	71.1	69.6	68.7	67.3	65.3	65.4	63.2	61.2	64.8	62.0
All problems Average deviation	**39.8**	**39.7**	**39.3**	**39.2**	**39.4**	**38.9**	**38.8**	**38.3**	**35.5**	**34.7**	**34.7**	**34.3**
Minimum percentage increase	16.8	10.5	8.9	12.2	8.8	16.2	12.2	12.2	8.8	9.9	7.9	7.9
Maximum percentage increase	71.4	69.9	71.1	71.7	68.7	70.1	71.7	65.4	63.2	61.2	64.8	62.0

Table 6. Two-pass heuristics based on priority rule

		Deviation with respect to the lower bound (%)			
Best of	CPU-t (sec.)	Min	Max	St.Dev	Mean
S-SGS (LFT) and S-SGS (LSTLFT)	**0.0061**	**7.9**	**62.0**	**12.9**	**33.3**
S-SGS (RWK) and S-SGS (LSTLFT)	**0.0062**	**7.9**	**58.8**	**13.0**	**33.4**
S-SGS (LFT) and S-SGS (LST)	**0.0063**	**7.9**	**61.2**	**12.9**	**33.4**
S-SGS (LFT) and S-SGS (RWK)	**0.0065**	**7.9**	**60.5**	**12.8**	**33.5**
S-SGS (LST) and S-SGS (RWK)	**0.0065**	**8.8**	**60.0**	**12.9**	**33.6**
S-SGS (LST) and S-SGS (LSTLFT)	**0.0068**	**7.9**	**61.2**	**13.1**	**33.8**
P-SGS (LSTLFT) and S-SGS (LSTLFT)	**0.0053**	**7.9**	**62.0**	**13.0**	**33.8**
P-SGS (LFT) and S-SGS (LFT)	0.0054	7.9	64.8	12.6	34.1
P-SGS (LST) and S-SGS (LST)	0.0052	9.9	61.2	13.0	34.2
P-SGS (RWK) and S-SGS (RWK)	0.0053	8.8	59.6	12.8	34.7
P-SGS (FREE) and S-SGS (FREE)	0.0054	10.5	65.3	12.8	37.7
P-SGS (SLK) and S-SGS (SLK)	0.0050	9.9	67.9	13.1	37.9
P-SGS (AN) and S-SGS (AN)	0.0053	17.6	70.8	12.4	40.5
P-SGS (EST) and S-SGS (EST)	0.0053	17.0	72.2	12.6	41.3
P-SGS (NIS) and S-SGS (NIS)	0.0055	14.8	73.6	12.5	42.9
P-SGS (GRPW) and S-SGS (GRPW)	0.0054	15.1	74.2	12.7	43.1
P-SGS (LPT) and S-SGS (LPT)	0.0053	17.6	89.1	13.1	43.3
P-SGS (GRD) and S-SGS (GRD)	0.0052	18.3	80.6	13.2	43.7
P-SGS (EFT) and S-SGS (EFT)	0.0053	18.3	73.3	11.4	44.0
P-SGS (SPT) and S-SGS (SPT)	0.0054	13.8	80.4	12.6	45.6

Table 7. Performance of two-pass heuristics based on priority rule

	Deviation with respect to the lower bound (%)										
	Best forward and backward heuristics in [Boctor 96b]				Best two-pass heuristics in this paper						
	LFT	CAN	SLK	RWK	P/S-SGS (LSTLFT)	S-SGS (LSTLFT/ LST)	S-SGS (LST/ RWK)	S-SGS (LFT/ RWK)	S-SGS (LFT/ LST)	S-SGS (LSTLFT/ RWK)	S-SGS (LSTLFT/ LFT)
Set 1: 50 activities and 1 resource type	**22.0**	**22.4**	**21.9**	**22.8**	**19.4**	**19.1**	**18.8**	**18.8**	**18.8**	**18.5**	**18.6**
	12.1	10.5	10.9	8.9	7.9	7.9	9.4	7.9	7.9	7.9	7.9
	37.6	32.9	35.9	41.8	36.7	36.7	32.0	32.0	32.8	32.0	34.0
Set 2: 50 activities and 2 resource types	**36.5**	**36.6**	**36.3**	**35.6**	**34.3**	**35.4**	**35.2**	**34.2**	**34.5**	**34.8**	**34.3**
	25.8	18.2	22.5	18.2	19.5	19.5	20.0	20.0	21.4	19.5	19.5
	46.4	49.4	50.5	52.6	49.0	48.0	48.0	46.4	47.4	49.2	46.4
Set 3: 50 activities and 4 resource types	**49.4**	**48.3**	**49.2**	**48.1**	**46.7**	**47.0**	**46.3**	**46.5**	**46.4**	**46.2**	**46.3**
	38.2	36.9	35.6	35.6	32.3	32.3	30.1	30.1	31.8	30.1	31.8
	65.2	60.9	67.9	59.8	60.1	58.9	59.3	59.8	58.9	58.9	60.1
Set 4: 100 activities and 1 resource type	**23.9**	**23.5**	**22.7**	**22.9**	**18.0**	**17.7**	**18.0**	**18.3**	**17.8**	**17.6**	**17.6**
	14.9	15.8	14.1	14.9	10.1	10.1	8.8	8.8	10.9	8.8	10.1
	37.0	35.1	33.4	34.3	24.8	24.8	27.8	25.4	25.5	24.6	24.8
Set 5: 100 activities and 2 resource types	**40.7**	**40.4**	**40.3**	**40.5**	**36.7**	**36.2**	**36.4**	**36.4**	**36.0**	**36.4**	**35.9**
	28.8	23.7	28.0	27.4	22.3	22.3	22.3	25.7	22.3	22.3	22.3
	52.0	55.4	55.0	54.3	47.5	47.5	50.8	48.2	48.2	47.5	47.5
Set 6: 100 activities and 4 resource types	**50.6**	**50.5**	**50.6**	**50.3**	**47.4**	**47.5**	**47.1**	**46.7**	**47.0**	**46.8**	**46.8**
	40.5	40.3	41.2	41.3	40.3	39.2	39.1	38.9	38.9	39.1	38.9
	65.6	67.3	66.7	69.3	62.0	61.2	60.0	60.5	61.2	58.7	62.0
All problems Average deviation	**37.3**	**36.9**	**36.8**	**36.7**	**33.8**	**33.8**	**33.6**	**33.5**	**33.4**	**33.4**	**33.3**
Minimum percentage increase	12.1	10.5	10.9	8.9	7.9	7.9	8.8	7.9	7.9	7.9	7.9
Maximum percentage increase	65.6	67.3	67.9	69.3	62.0	61.2	60.0	60.5	61.2	58.9	62.0

Table 8. Performance of the best heuristics based on priority rule -deviation with respect to the lower bound (%)

	Heuristic based on activity-mode combinations [Boctor 96b]	Best multi-pass heuristics based on priority rule in this paper			
		BEST1	BEST2	BEST4	BEST8
Set 1: 50 activities and 1 resource type	**20.2**	**19.5**	**18.6**	**17.9**	**17.8**
	11.3	7.9	7.9	7.9	7.9
	30.4	36.7	34.0	32.0	32.0
Set 2: 50 activities and 2 resource types	**33.9**	**35.8**	**34.3**	**33.6**	**32.3**
	18.8	19.5	19.5	19.5	19.5
	46.0	49.2	46.4	46.4	44.5
Set 3: 50 activities and 4 resource types	**46.4**	**47.6**	**46.3**	**45.3**	**44.2**
	32.8	32.3	31.8	30.1	30.1
	60.3	60.1	60.1	58.5	58.5
Set 4: 100 activities and 1 resource type	**20.4**	**18.1**	**17.6**	**17.2**	**17.0**
	12.8	10.1	10.1	8.8	8.8
	31.1	24.8	24.8	24.6	24.6
Set 5: 100 activities and 2 resource types	**37.4**	**36.9**	**35.9**	**35.5**	**35.3**
	27.4	22.3	22.3	22.3	22.3
	48.5	47.5	47.5	47.5	47.5
Set 6: 100 activities and 4 resource types	**47.8**	**47.8**	**46.8**	**46.0**	**45.7**
	39.7	40.3	38.9	38.9	38.9
	60.2	62.0	62.0	58.7	57.7
All problems Average deviation	**34.4**	**34.3**	**33.3**	**32.6**	**32.0**
Minimum percentage increase	11.3	7.9	7.9	7.9	7.9
Maximum percentage increase	60.3	62.0	62.0	58.7	58.5