INSTITUTE OF
COMPUTER SCIENCE

HELMUT SCHMIDT
UNIVERSITÄT
Universität der Bundeswehr Hamburg

RESEARCH
PAPER

№ 17-01

A MODE REDUCTION TECHNIQUE FOR THE MULTI-MODE RESOURCE-CONSTRAINED PROJECT SCHEDULING PROBLEM

Christian Stürck
Helmut Schmidt University Hamburg
christian.stuerck@hsu-hh.de

SEPTEMBER 2017

HSU Institute of Computer Science Research Paper Series • ISSN 2198-3968

A Mode Reduction Technique for the Multi-Mode Resource-Constrained Project Scheduling Problem

Christian Stürck

Institute for Operations Research
Helmut Schmidt University
Hamburg, Germany
E-Mail: christian.stuerck@hsu-hh.de

Abstract

Recently, new benchmark instances (MMLIB) have been presented by Van Peteghem and Vanhoucke (2014). These instances have been designed in such way that the well known pre-processing procedure of Sprecher et al. (1997) does not have any impact on any instance. We developed a technique that could be applied to the new data set to reduce the number of modes per activity. Computational results show that the procedure is highly effective in reducing the number of modes. A reduction was possible for 2,026 of the 4,320 MMLIB instances.

Keywords: Multi-Mode Resource-Constrained Project Scheduling Problem, MRCPSP, Mode Reduction, Pre-processing, MMLIB

1 Introduction

In the field of project scheduling the multi-mode resource-constrained project scheduling problem (MRCPSP) is well known and intensively studied. The objective is to minimise the makespan of the project. Each activity has to be assigned to an execution mode as well as a starting time. Precedence constraints have to be considered as well as non-renewable and renewable resource constraints. A more detailed problem description is given by Mika et al. (2015).

Being a generalisation of the RCPSP the MRCPSP is $\mathcal{NP}$-hard. Since the modes consume non-renewable resources even finding a feasible mode assignment

is $\mathcal{NP}$-complete if the instance contains more than one non-renewable resource (Kolisch and Drexl (1997)).

To reduce the complexity of the problem, pre-processing procedures have been presented in the literature. Sprecher et al. (1997) proposed a pre-processing procedure which was able to eliminate inefficient modes as well as redundant non-renewable resources. It has shown to be very effective on the PSPLIB instances (Kolisch and Sprecher (1997)). However, Van Peteghem and Vanhoucke (2014) – the authors of the most recently presented benchmark data set MMLIB – used a repair function during the design of the library to ensure that this procedure has no effect on the instances at all.

This work presents a new mode reduction technique which is effective even on the MMLIB instances. The next section outlines the design of the proposed mode reduction technique. In section 3 combinations of the technique with existing approaches are proposed. The effectiveness of the procedure is tested on the MMLIB data set in section 4. Conclusions are drawn in section 5.

2 A mode reduction technique for the MRCPSP

Earliest starting times ES_i, latest starting times LS_i, earliest completion times EC_i and latest completion times LC_i for activities $i \in A$ are used in heuristics as well as in exact methods. For some of the mathematical models (e.g. the model of Talbot (1982)) these values have to be determined upfront. The values of ES_i and EC_i can be computed with the critical path method (CPM) of Kelley (1963), using the mode with the shortest duration for each activity. For the computation of LS_i and LC_i an upper bound T is needed. This can be a known feasible makespan or, if no feasible makespan is known, the sum of the modes with the maximum duration can be taken as worst case approximation (Mika et al. (2015)).

With an upper bound T the values of LS_i and LC_i can be derived using CPM (backward pass) as well. Therefore, the values of LS_i and LC_i depend on the value of T. If T is taken from a feasible solution x and a better solution x' is found, all LS_i and LC_i values can be decreased by the difference of the makespans of x and x'.

The proposed mode reduction technique is based on the dependency of LC_i on T. Thus, to apply the procedure a known feasible makespan of solution x is needed. Based on x the latest completion time $LC_i^{feasible}$ can be derived. Each mode m of each activity $i \in A$ is investigated, wether condition (1) holds:

$$LC_i^{feasible} \geq ES_i + d_{i,m} \; . \tag{1}$$

If inequality (1) is violated for a mode m and m is part of the mode vector, it is impossible to achieve a makespan which is equal or less than the makespan

of x. Therefore, this mode can be be eliminated. Since we know that the optimal solution belongs to the set of feasible solutions and each solution has a makespan either equal or greater than the optimal solution, we call these modes *non-optimal modes*. Executing an activity in a *non-optimal mode* may lead to good solution, but never to the optimal one. Term (1) can be seen as an generalisation of *Bounding Rule 1* from Sprecher et al. (1997) which stated that a partial schedule with an activity that extends its latest completion time cannot be completed with a makespan equal to T or less. If constraint programming is used, Term (1) can be redefined as constraint propagation.

To illustrate this an example is given Figure 1. With a given upper bound $T = 6$ of a feasible solution the $LC_i^{feasible}$ values can be derived. Activity i has two modes with a duration of $d_{i,1} = 2$ and $d_{i,2} = 5$. Using CPM the earliest starting and latest completion time can be computed as $ES_i = 2$ and $LC_i^{feasible} = 6$.

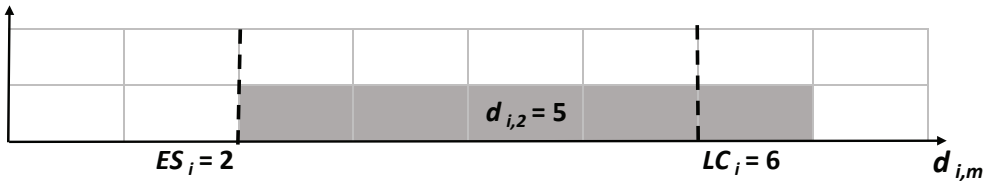

Figure 1: Example of a *non-optimal* mode

While for the first mode condition (1) holds, for the second mode it does not. Therefore, it is a *non-optimal* mode and can be excluded from any further investigation for the optimal solution. This is illustrated in Figure 1.

There are different ways to apply the mode reduction:

- The technique can be used in a heuristic each time a better solution is found.
- If an exact method is used the approach can be applied if the method stops after a given time with a feasible solution and then starts again with updated modes and LC_i values. An alternative would be to run an exact method from a solution that was generated with heuristic methods.
- Since only the makespan is required, the best known solutions from a database can be used. This comes with the disadvantage that a procedure is no longer independent from external knowledge. Nevertheless, if the aim is finding new best known solutions this approach is very promising.

In the following the integration of the mode reduction technique into other pre-processing elements is considered.

3 Integration of the presented approach into known pre-processing procedures

This approach can be combined with other techniques to reduce the complexity of the MRCPSP even further. To show this two examples are given: the pre-processing procedure of Sprecher et al. (1997) and the calculation of new earliest starting times (Zhu et al. (2006); Stürck and Gerhards (2018)).

3.1 Integration in a pre-processing procedure

Sprecher et al. (1997) presented a pre-processing procedure for the MRCPSP. The authors of the MMLIB stated that this procedures does not have any impact on the MMLIB (Van Peteghem and Vanhoucke (2014)). We tested this and we can verify this statement – neither a mode nor a resource was excluded while applying the pre-processing procedure on the MMLIB datasets.

Sprecher et al. (1997) reduced the complexity in their approach by identifying the following three cases:

- *Non-executable modes*: Modes that are exceeding the given renewable resources.
- *Inefficient modes*: Modes that are dominated by at least one mode of that activity in their duration as well as their resource consumption.
- *Redundant non-renewable resources*: Non-renewable resources that do not restrict the instance at all.

These three eliminations were used iteratively until no more reductions occur, since an elimination of a *redundant non-renewable resource* could lead to an *inefficient mode* and vice versa.

The presented mode reduction technique can be included in the pre-processing procedure in the following way:

1. Delete all *non-executable modes*.
2. Delete the *non-optimal modes*.
3. Delete all *redundant non-renewable resources*.
4. Delete all *inefficient modes*.
5. If any mode has been reduced, restart at step 3.

Step 1 will never have an impact on the MMLIB instances, since no *non-executable* modes exist. To maintain a generally valid procedure this step is still part of the pre-processing. The presented mode reduction technique can be found in step 2. The reduction of modes may lead to *redundant non-renewable resources* in step 3 which subsequently could result in *inefficient modes*. The loop is restarted each time a mode was reduced. If a better makespan has been found, all LC_i values can be updated and the loop can be restarted with Step 2.

3.2 New earliest starting times

In the work of Zhu et al. (2006) as well as Stürck and Gerhards (2018) new earliest starting times are computed. While Zhu et al. (2006) use heuristics to determine new starting times, in the approach of Stürck and Gerhards (2018) each starting time is calculated with mathematical programming techniques. The basic idea is the following: since CPM only takes the precedence constraints into account and ignores all resource restrictions, a starting time ES_i derived from CPM may never be feasible.

A small example: If an activity 3 has two predecessors 1 and 2 with $ES_1 = 0$ and $ES_2 = 0$ and the modes with the minimal duration $d_{i,m}$ are $d_{1,1} = 1$ and $d_{2,1} = 1$, the earliest starting time of activity $i = 3$ is $ES_3 = 1$. However, if the resource constraints do not allow activity 1 and 2 to be executed in parallel, it is impossible that $ES_3 = 1$ can be achieved and ES_3 can be updated to $\widehat{ES_3} = 2$.

If an earliest starting time is updated, it can be taken into account when excluding the non-optimal modes. Inequality (1) can be tested for each mode with the new calculated starting time instead of the one derived from CPM alone. This could lead to more modes that can be excluded.

4 Computational experiments

The presented pre-processing procedure was applied to the MMLIB instances to test its effectiveness. As a measure of effectiveness we use the number of reduced modes. We argue that with a lower number of modes the search space is decreased. Hence, it could be easier to identify good quality solutions. The experiments were carried out on a PC with an Intel i-7-6700K CPU at 4.00 GHz. The algorithm was implemented in C#. The computational time was below one second for each instance.

To test the technique the makespan of the best known solution (*BKS*) of each MMLIB instance[1] was taken as upper bound T. Based on T the critical path

[1]Derived from the website *www.mmlib.eu* in June 2017.

method was used to compute the $LC_i^{feasible}$ values. Then each mode was tested. If a mode violated inequality (1) it was deleted as *non-optimal* mode.

	MMLIB50	MMLIB100	MMLIB+
Total number of instances	540	540	3,240
Number of instances with at least one *non-optimal mode*	352	347	1,327
Average reduction of *non-optimal modes*	17.69%	13.45%	13,52%
Maximal reduction	44.00%	32.33%	55.78%

Table 1: *Non-optimal modes* of the MMLIB after using the best known solutions

The results of the computational experiments are displayed in Table 1. They show that if the best known solutions are used *non-optimal modes* can be found in 65.19% / 64.26% / 40.96% of the MMLIB50 / MMLIB100 / MMLIB+ instances, respectively. Due to the fact that the first mode has the shortest duration it was always kept for each instance. Furthermore, after deleting all *non-optimal modes* the remaining modes have been tested for their feasibility. A feasibility check has been made by a MIP-based feasible mode assignment as presented in Gerhards et al. (2017). A feasible mode assignment was found for every instance. Thus, the technique did not lead to infeasibility.

To investigate on which instances the technique has an impact, we take a more detailed look on the instances with *non-optimal* modes. Due to the fact that the makespans of the best known solutions were used as value for T for the reduction, we introduce the parameter Δ^{BKS}. It is determined as the absolute gap of the BKS and the critical path lower bound: $\Delta^{BKS} = BKS - CPLB$.

	MMLIB50	MMLIB100	MMLIB+
Minimal Δ^{BKS} of an instance without any *non-optimal* modes	4	5	4
Maximal Δ^{BKS} of an instance with *non-optimal* modes	8	8	27
Average Δ^{BKS} of an instance with *non-optimal* modes	1.09	0.79	6.64
Average Δ^{BKS} of the data set	6.36	7.83	37.09

Table 2: Minimal, maximal and average values of Δ^{BKS}

Table 2 shows the investigation of the Δ^{BKS} of all instances. The first row shows the minimal value of Δ^{BKS} of all instances without any *non-optimal* modes. This shows that every instance with a $\Delta^{BKS} < 4$ has *non-optimal* modes. The third row

shows that instances with *non-optimal* modes have an distinctively lower average value of Δ^{BKS} than the average value of the data set. Thus, Δ^{BKS} has an impact on the number of *non-optimal* modes.

The maximal values of Δ^{BKS} of the instances with *non-optimal* modes are presented in the second row. The maximal value of 27 was obtained from an instance with nine modes. It contained an activity i, whose first mode has a duration of $d_{i,1} = 1$ and its ninth mode a duration of $d_{i,9} = 30$. This activity was part of the critical path. Since the LC_i values are calculated with the shortest mode, the gap between these two modes with 29 was higher than the Δ^{BKS} with 27, which resulted in the reduction of the ninth mode. This shows that the structure of an instance (critical path, gap of the duration of the modes) has an impact on the number of *non-optimal* modes as well.

The efficiency of the mode reduction technique was highly depending on the quality of the makespan. Especially for instances with a makespan close to the critical path lower bound a high number of *non-optimal modes* was found. The computational time for all instances was within milliseconds. The computational experiments show that the technique is very fast and has an impact on nearly half of the MMLIB instances.

5 Conclusions

This work presented a mode reduction technique for the MRCPSP. Two different ways of integrating it into existing pre-processing procedures were proposed. The computational experiments on the MMLIB were able to display the effectiveness of the procedure. Using the best known solutions a reduction was possible for 2,026 of the 4,320 MMLIB instances. The computation was very fast. A feasible mode assignment proved that after the reduction still feasible mode combinations exist which do not exceed the non-renewable resources. An analysis of the instances showed that the technique is most efficient, if the gap between the makespan and the critical path lower bound is small.

First computational results of implementing this mode reduction technique within a procedure for solving the MRCPSP are very promising: several new best known solutions were found using this mode reduction technique. Further experiments have to be made to test the impact on different solution approaches for the MRCPSP. Additionally, the combinations with other pre-processing elements have to be investigated.

References

P. Gerhards, C. Stürck, and A. Fink. An adaptive large neighbourhood search as a matheuristic for the multi-mode resource-constrained project scheduling problem. *European Journal of Industrial Engineering*, to appear, 2017.

J. E. Kelley. The critical-path method: Resources planning and scheduling. *Industrial Scheduling*, 13(1):347–365, 1963.

R. Kolisch and A. Drexl. Local search for nonpreemptive multi-mode resource-constrained project scheduling. *IIE Transactions*, 29(11):987–999, 1997.

R. Kolisch and A. Sprecher. PSPLIB – a project scheduling problem library: OR software – ORSEP operations research software exchange program. *European Journal of Operational Research*, 96(1):205–216, 1997.

M. Mika, G. Waligóra, and J. Weglarz. Overview and state of the art. In C. Schwindt and J. Zimmermann, editors, *Handbook on Project Management and Scheduling Vol. 1*, pages 445–490. Springer, Cham, 2015.

A. Sprecher, S. Hartmann, and A. Drexl. An exact algorithm for project scheduling with multiple modes. *Operations-Research-Spektrum*, 19(3):195–203, 1997.

C. Stürck and P. Gerhards. Providing lower bounds for the multi-mode resource-constrained project scheduling problem. In *Operations Research Proceedings 2016*, pages 551–557. Springer, 2018.

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

V. Van Peteghem and M. Vanhoucke. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. *European Journal of Operational Research*, 235(1): 62–72, 2014.

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

HELMUT SCHMIDT UNIVERSITÄT
Universität der Bundeswehr Hamburg

IMPRINT

SERIES

HSU Institute of Computer Science Research Paper Series
ISSN 2198-3968

EDITOR

Prof. Dr. Andreas Fink

Institute of Computer Science
Helmut-Schmidt-Universität Hamburg

Holstenhofweg 85
22043 Hamburg, Germany

http://ifi.hsu-hh.de