A Local Search Framework for Industrial Test Laboratory Scheduling

Extended Abstract

Florian Mischek · Nysret Musliu

Keywords Project Scheduling · Industrial Test Laboratory · Local Search

1 Introduction

In industrial test laboratories, a large number of tests have to be scheduled, requiring qualified personnel and specialized equipment. At the same time, various different constraints have to be satisfied, including project deadlines, resource availabilities and precedence relations between the tests.

This problem is related to the well-known Resource Constraint Project Scheduling Problem (RCPSP) (see e.g. [1,3]), but contains several novel features, which have to the best of our knowledge not yet been discussed elsewhere. Most notably, it requires the solver to group multiple tasks into a larger unit, which is then assigned a time slot and resources.

We provide a definition of the Test Laboratory Scheduling Problem (TLSP), together with a set of benchmarking instances (both from a real-world laboratory and randomly generated). In addition, we have implemented a Local Search framework for this problem, which supports various Metaheuristics.

2 Problem Description

In TLSP, the task is to create a schedule for a list of tasks in multiple projects. Each task can be performed in one of several modes, and has a fixed duration which depends on the chosen mode. Also, tasks require several renewable resources of different kinds (e.g. employees, test benches, equipment...). For a

Christian Doppler Laboratory for Artificial Intelligence and Optimization for Planning and Scheduling
DBAI, TU Wien
E-mail: {fmischek,musliu}@dbai.tuwien.ac.at

given mode, a certain number of units of each resource must be chosen, potentially from a restricted subset of units of that resource (a similar formulation of resource requirements can be found in [2]).

Both individual tasks and the projects they appear in can have a release date and a deadline. Within a project, tasks may have prerequisite tasks, which must be completed first.

However, individual tasks are not scheduled directly, but must first be grouped into larger units, called *jobs*. This grouping must be found by the solver as part of the solution process. The duration of a job is equal to the sum of the durations of all contained task, plus an additional fixed setup period. The resources assigned to a job must be able to satisfy the requirements of each contained task over the full length of the job. While there are other problems in the literature containing grouping or batching of smaller activities into larger units, these usually either do not deal with time slot and resource assignment at all or still assign them to the original tasks. The concept of directly scheduling the larger units is, to the best of our knowledge, not found anywhere else so far.

Solvers must also respect fixed assignments, which predefine certain jobs and some or all of their time slot and resource assignments. Further, tasks within a project can be linked, which indicates that the jobs containing those tasks must have the same employees assigned.

Soft constraints include minimizing the makespan of each project as well as respecting an additional target date, and having preferred employees over others for certain tasks. Finally, the number of employees assigned to work on each project should also be kept to a minimum.

3 Local Search Framework

We implemented a Local Search Framework for this problem, which supports a set of neighborhood operators and allows for easy implementation of various metaheuristics.

The neighborhood operators include basic moves like changing time slots, resource assignments, or mode switching for a single job, adapted to the special requirements of the problem. Additionally, it includes new moves to dynamically adapt the grouping of tasks into jobs.

An initial solution can be generated either using one of several construction heuristics, or taken from an existing base schedule.

Metaheuristic techniques such as Tabu Search, Simulated Annealing or Min-Conflict can then be easily implemented using the provided neighborhood operators.

We have deployed a preliminary implementation of these methods in a specific test laboratory, and already achieved considerable improvements in solution quality over the current practice of manual scheduling.

In future work, we expect to further improve upon these results by incorporating more complex neighborhood operators. We also plan to investigate the use of selection hyper-heuristics, to choose the best operators for a given situation.

4 Conclusion

We introduced a novel problem definition suited to model the planning of operations in an industrial test laboratory, based on real-world requirements. This problem includes several features not found in other research in the area of project scheduling so far.

Benchmark instances, both randomly-generated and real-world datasets, will be made available publicly.

For this problem, we provided a Local Search Framework, which contains innovative neighborhood operators to deal with these new features and provides baseline solutions for other work to build upon.

Acknowledgements The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

References

1. Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E.: Resource-constrained project scheduling: Notation, classification, models, and methods. European Journal of Operational Research **112**(1), 3 – 41 (1999). DOI https://doi.org/10.1016/S0377-2217(98)00204-5. URL http://www.sciencedirect.com/science/article/pii/S0377221798002045
2. Dauzère-Pérès, S., Roux, W., Lasserre, J.: Multi-resource shop scheduling with resource flexibility. European Journal of Operational Research **107**(2), 289 – 305 (1998). DOI https://doi.org/10.1016/S0377-2217(97)00341-X. URL http://www.sciencedirect.com/science/article/pii/S037722179700341X
3. Herroelen, W., Reyck, B.D., Demeulemeester, E.: Resource-constrained project scheduling: A survey of recent developments. Computers & Operations Research **25**(4), 279 – 302 (1998). DOI https://doi.org/10.1016/S0305-0548(97)00055-5. URL http://www.sciencedirect.com/science/article/pii/S0305054897000555