CHEMICAL ENGINEERING TRANSACTIONS

VOL. 70, 2018

A publication of AIDIC
The Italian Association of Chemical Engineering
Online at www.aidic.it/cet

Guest Editors: Timothy G. Walmsley, Petar S. Varbanov, Rongxin Su, Jiří J. Klemeš
Copyright © 2018, AIDIC Servizi S.r.l.
ISBN 978-88-95608-67-9; **ISSN** 2283-9216

DOI: 10.3303/CET1870195

An S-graph Based Approach for Multi-Mode Resource-Constrained Project Scheduling with Time-Varying Resource Capacities

Olivér Ősz*, Máté Hegyháti

Széchenyi István University, Department of Information Technology, Egyetem tér 1, 9026, Győr, Hungary
osz.oliver@sze.hu

The Resource-Constrained Project Scheduling Problem (RCPSP) is a general problem class of scheduling problems. It even contains the well-known job shop scheduling problem as a special case. In these problems, jobs demand different amounts from multiple resources at once. The goal is to minimize completion time while satisfying resource capacity constraints throughout the schedule. The RCPSP has several variations and generalizations, one of which is the multi-mode problem. Multiple operation modes are given for jobs, differing in resource usages and execution times. Another generalization is where resources have time-varying capacities instead of a constant limit. The S-graph framework was developed for batch process scheduling and used successfully in various applications. The advantages of this approach motivate research to extend its capabilities to more general problem classes and apply it in various specialized case studies. The authors have previously presented an extension of the S-graph framework for RCPSP and its multi-mode generalization. In this work, a further extension is proposed for handling time-varying capacities. The method relies on a model transformation technique and recent algorithmic developments of the framework.

1. Introduction

The Resource-Constrained Project Scheduling Problem (RCPSP) is a wide problem class in scheduling, especially with its numerous generalizations. Jobs demanding multiple types of resources at once are found in various scheduling problems, including planning construction projects (Xing et al., 2016), resource-allocation in cloud computing environments (Zhao and Shen, 2015), and optimizing complex production processes.
The task is to schedule non-preemptive jobs, which are given along with their precedence network, fixed execution times, and resource usages. The resources have finite capacities, and the usages are subtracted from them when a job starts and returned when the job is finished. A schedule is infeasible if the resource usage of parallel jobs exceeds the capacity at any point in time. In the multi-mode case, there can be non-renewable resources as well, which are only consumed but not returned, and the total usage from them cannot exceed the starting capacity. In this work, the goal is to minimize completion time, for reference, other objective functions for the RCPSP and their methods are presented in the survey paper by Brucker et al. (1999).
Time-varying resource availability is a feature frequently arising in practical occurrences of RCPSP and other scheduling problems. Depending on the kind of resource, changes in its availability can be caused by planned maintenance of equipment, work shifts of workers, external dependencies, or other effects. In the problems this study considers, resource availability is given as a step function over time.
This work presents an approach based on the S-graph framework, which was originally proposed for multipurpose batch process scheduling by Sanmartí et al. (2002). The framework was later extended for various problem classes, for instance, problems involving limited waiting time constraints (Hegyháti et al., 2011), and scheduling of automated wet-etch stations (Hegyháti et al., 2014). An extension for single- and multi-mode RCPSP was presented by Ősz and Hegyháti (2017), which is hereby further extended with time-varying resource availabilities.
Related works about RCPSP are introduced in Section 2. The original S-graph based approach for RCPSP is explained in Section 3. The proposed extension to handle time-varying capacities is presented in Section 4.

Computational results are discussed in Section 5. Section 6 contains the concluding remarks and future research suggestions.

2. Related works

Researchers in project scheduling have been studying solution methods of the RCPSP and its variants since the 1960s. Early approaches relied on heuristics (Davis and Patterson, 1975), and branch-and-bound search methods to solve smaller instances to optimality (Christofides et al., 1987).
Later, as mathematical programming methods had improved, MILP approaches were also presented for RCPSPs. Alvarez-Valdés and Tamarit (1993) proposed a method which uses polyhedron optimization with lifting theorems. They introduced Minimal Resource Incompatible Sets, which are used in the S-graph based approach as well and they are explained in more detail in Section 3.2. Other MILP based approaches used vastly different modelling techniques: Mingozzi et al. (1998) used feasible sets of activities opposed to the incompatible sets mentioned before, and Kopanos et al. (2014) presented general discrete- and continuous-time formulations.
Multi-mode RCPSP is a harder problem, as it also contains assignment decisions choosing from multiple possible operation modes for each job. Zhu et al. (2006) used a model with discrete time grid and a branch-and-cut algorithm, whose performance competes well with more recent models, as Koné et al. (2011) reports based on extensive comparison.
Time-varying resource availability was introduced in the Generalized RCPSP (GRCPSP) by Herroelen and Demeulemeester (1992), along with other extensions to the original problem, such as setup times and time-varying resource usages. Demeulemeester and Herroelen (1996) presented a branch-and-bound procedure to solve GRCPSPs, and Klein (2000) proposed heuristic solution methods.
Most of the literature methods presented for RCPSP with time-varying resource availabilities only consider the single-mode case. Cheng et al. (2015) developed a precedence tree-based branch-and-bound algorithm to solve the multi-mode case with also allowing non-preemptive activity splitting (jobs can only be paused due to lack of resources). Kreter et al. (2016) further generalized the constraints for activity splitting with time-bounded breaks and resources remaining engaged during breaks. The authors proposed 3 types of binary linear models, and heuristic search procedures for optimization.
Studying solution methods for the RCPSP and its many generalizations is an important and active field of research in optimization. The case of time-varying resource capacities has only been solved with specialized methods and metaheuristics, and most approaches only consider the single-mode case. Therefore, integrating this constraint into a general-purpose scheduling framework like the S-graph, is an important advancement.

3. Modelling and solving RCPSP with S-graphs

The single-mode RCPSP was solved by extending the S-graph model with Minimal Resource Incompatible Sets (MRIS) and a new branching strategy which resolves resource conflicts between jobs. In the multi-mode case, MRIS were extended with mode assignments.

3.1 Original S-graph model

The S-graph framework (Sanmartí et al., 2002) uses a directed acyclic graph with arc weights to model partial or complete schedules. Nodes represent the start of jobs and completion of products. The arcs denote the precedence relations with weights imposing lower bounds on time differences between the associated events.
The solution process starts with the recipe graph, where arcs represent the dependencies between jobs, given by the problem instance. A recipe graph for an example with 3 products (A, B, C), each with 3 steps, is shown in Figure 1. On job nodes, the set of applicable equipment units are shown.

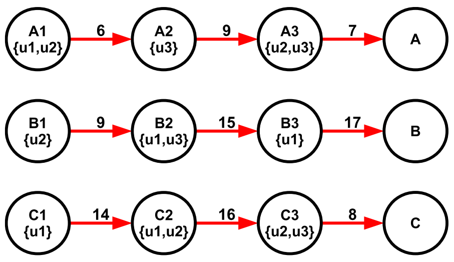


Figure 1: Recipe graph example

A branch-and-bound procedure makes assignment and sequencing decisions and introduces the necessary arcs to the graph. The longest path in the graph gives a lower bound on the makespan (completion time) for a partial schedule, or the objective value for a complete schedule. If the graph contains a directed cycle, the schedule is infeasible, along with all subproblems of it.
Figure 2 shows an S-graph model of a complete feasible schedule for the recipe graph of Figure 1. Every job is assigned to an equipment unit and schedule arcs (blue) determine the production sequence for each unit. Note that these arcs are added with Unlimited Intermediate Storage policy, which is also the case in RCPSPs.

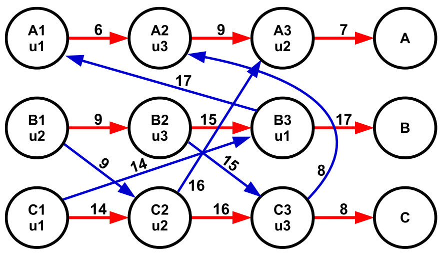
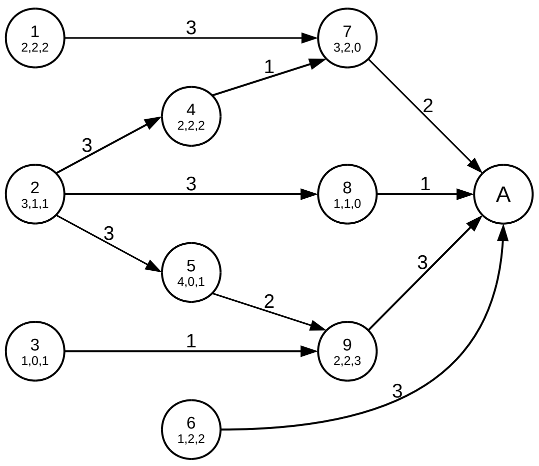


Figure 2: S-graph of a complete schedule reached from the recipe graph of Figure 1

3.2 Single-mode RCPSP with S-graphs

Equipment units are resources necessary for executing jobs, but the S-graph framework was only able to handle one-to-one assignments as is required in machine scheduling problems. In RCPSP, job sequencing must be made in order to avoid parallel jobs demanding more resource than what is available. One way to do this, is by using Minimal Resource Incompatible Sets (Alvarez-Valdés and Tamarit, 1993).
MRIS are the minimal sets of jobs that cannot be executed in parallel due to limited resource capacities. A schedule is infeasible if the set of active jobs at any time contains any of these MRIS. If for each MRIS, any two elements of it are scheduled without overlap, the schedule is feasible (assuming that the input precedence relations are satisfied).
MRIS can be generated at the start of the solution process. After that, the branch-and-bound procedure gradually resolve the sets. At each branching step, an MRIS is selected, and it is resolved in every possible way: for each ordered pair of jobs, their order is fixed by adding a schedule arc between them, which creates a subproblem from the current partial schedule. If a new path is created between any two nodes of another set, that set is resolved too. When every set is resolved, the schedule is feasible.
Branching is illustrated on a small example; whose recipe graph is in Figure 3. Instead of equipment sets, resource usages are shown below the job numbers.



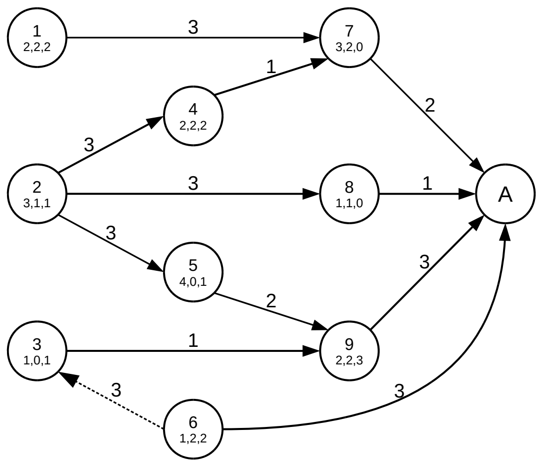
Open MRIS:

{1,2} {3,6,7}
{1,3,4} {3,7,8}
{1,3,6} {4,5}
{1,5} {4,6,8}
{1,4,6} {4,9}
{1,4,8} {5,6}
{1,6,8} {5,7}
{1,9} {5,8}
{2,3,6} {6,7,8}
{3,4,6} {6,9}
{3,5} {7,9}

Capacities:
4, 4, 4

Figure 3: Recipe graph of a single-mode RCPSP

There are 22 MRIS in the example, shown on the right of the graph. If the set {2, 3, 6} is selected at branching, that results in 9 node pairs, each offering a different resolution of the MRIS. One of them is (6, 3) for which the graph with the new schedule arc is shown in Figure 4. This decision also resolves other sets containing both 3 and 6, and also {6,9}, as a new path is created from 6 to 9, setting their precedence.



Open MRIS:

{1,2}
{1,3,4}
~~{1,3,6}~~
{1,5}
{1,4,6}
{1,4,8}
{1,6,8}
{1,9}
~~{2,3,6}~~
~~{3,4,6}~~
{3,5}
~~{3,6,7}~~
{3,7,8}
{4,5}
{4,6,8}
{4,9}
{5,6}
{5,7}
{5,8}
{6,7,8}
~~{6,9}~~
{7,9}

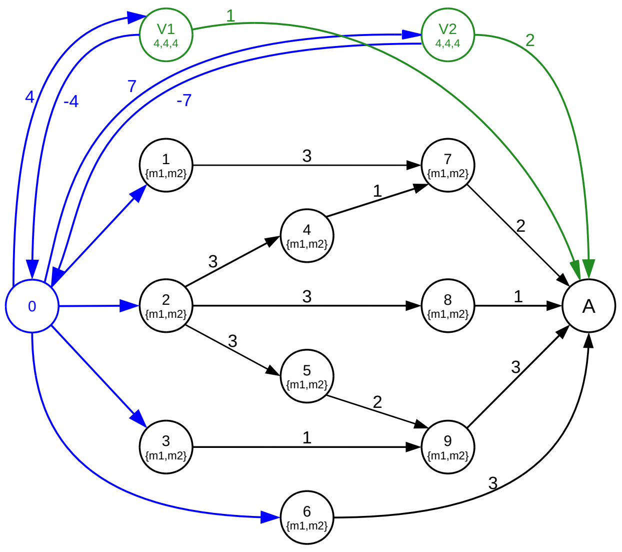
Figure 4: Adding a schedule arc between jobs 6 and 3

3.3 Extension to the multi-mode case

For multi-mode problems, MRIS are modified to contain job-mode pairs instead of jobs, as the resource usages may be different for operation modes of a job. As non-renewable resources are introduced, some MRIS only violate capacities of non-renewable resources. These job-mode pairs are not only infeasible when executed in parallel, but the mode assignments alone are infeasible. These sets cannot be resolved by introducing a precedence relation between its members. Another way of resolving sets is introduced at branching.
MRIS contain jobs with certain modes, if a different mode is assigned to any job inside the set, that set is resolved. Therefore, the multi-mode branching generates subproblems not only by adding schedule arcs but by making assignment decisions contradicting the ones in the chosen MRIS. Mode assignments can be modelled with S-graphs similarly to equipment assignments used in the original framework.

4. Model transformation for time-varying resource capacities

With the graph-based model of the S-graph framework, time is represented as time differences between events. When a new arc is added to the graph, multiple events may be shifted in the schedule. To model time-varying resource availability, virtual jobs are introduced with fixed timing, as proposed by Bartusch et al. (1988). These virtual jobs decrease the resource capacities for actual jobs, when the availability is below the maximum.
For fixing the start times of the virtual jobs, a reference node is introduced to represent the start of the time horizon. Then Zero-Wait (ZW) arcs are added between the zero node and virtual nodes.
ZW arcs in S-graphs were first introduced by Hegyháti et al. (2011). Later, a more efficient representation was used for wet-etch scheduling (Hegyháti et al., 2014), using negative weights. The latter method was used here as well. In the more general Limited-Wait case, for each recipe arc going out from a LW job, its weight is set to the processing time, and a reverse arc is added with a weight equal to the negated sum of the processing and waiting time.
The reverse arcs create directed cycles in the graph which used to indicate infeasibility in the original S-graphs. With LW arcs, cycles with negative total weight are always feasible. If the weight of a cycle is 0, it is only infeasible if no LW arcs are part of it. If a cycle with weight 0 contains LW arcs, that means the waiting time is at the maximal allowed amount. If a cycle has positive weight, it is always infeasible.
Figure 5 shows an example recipe graph with 2 virtual jobs at time 4 and 7. The sets of modes are shown at the bottom of actual job nodes. Virtual nodes show the resource usage associated with them. The processing times of the virtual jobs are equal to the duration of the resource shortage.
Virtual jobs are included in MRIS too. If the decreased capacity is not enough to perform a job, that job and the virtual job will be in an MRIS. Through sequencing decisions, the actual job will be scheduled before or after the virtual job.
The virtual jobs require a minor algorithmic modification as well. When calculating the longest path in the graph for the lower bound, paths containing a (green) recipe arc starting from a node of a virtual job, are ignored. If the longest path is one of these ignored paths, that means the makespan is lower than the starting time of the virtual job where the path starts.



Open MRIS:

{(1,m1), (2,m1)}
{(1,m1), (2,m2)}
{(1,m1), (4,m1), (6,m2)}
{(1,m1), (4,m1), (8,m1)}
{(1,m2), (3,m1)}
{(1,m2), (3,m2), (6,m1)}
{(1,m2), (5,m1)}
{(2,m1), (5,m2)}
...
{V1, (1,m1)}
{V1, (1,m2)}
{V1, (2,m1)}
{V1, (2,m2)}
{V1, (3,m1)}
...
{V2, (1,m1)}
{V2, (1,m2)}
{V2, (2,m1)}
{V2, (2,m2)}
{V2, (3,m1)}
...

Figure 5: Recipe graph of a multi-mode problem with virtual jobs

5. Computational results

The proposed method was tested on instances based on the randomly generated problems of the PSPLIB problem library (Kolisch and Hartmann, 1999). The multi-mode problem set j10 was used, where problems contain 10 jobs, each with 3 available modes, 2 renewable and 2 non-renewable resources. Job durations are given in integer time units, although the proposed method can handle arbitrary floating-point values.
The time-varying resource availability was randomly set in the same way as suggested by Hartmann (2013). The parameter PR denotes the probability of resource capacities decreased to 0 during any unit-sized interval of the time horizon. As Hartmann (2013) stated, resources are decreased at the same time in most practical applications. The recommended probabilities of 0.05 and 0.1 were used.
The tests were completed on a laptop with an Intel i7-6700HQ CPU @ 2.60GHz and 8 GB RAM.
Table 1 shows the results for parameter setting 39 of j10. Other parametrizations yielded similar results. As the probability of resource shortages increased, the average solution time increased as well. This is caused partly by the graph containing more nodes and more generated MRIS but also by the difficulty imposed by scarce resources that can be seen from the highly increased makespan.

Table 1: Solutions and execution times for j10/39

Problem		PR=0		PR=0.05		PR=0.1
instance	makespan	CPU s	makespan	CPU s	makespan	CPU s
1	21	0.81	28	1.85	27	2.85
2	18	2.83	25	4.51	25	5.55
3	28	1.40	48	27.48	66	286.23
4	17	0.46	25	1.49	29	3.49
5	30	1.38	45	12.06	50	30.51
6	24	0.70	37	3.55	31	2.12
7	17	0.26	22	0.59	23	0.79
8	23	1.87	27	1.80	32	6.53
9	24	0.84	30	3.07	64	249.37
10	31	8.77	40	9.43	39	8.92

The results show that the S-graph framework can solve problems even with relatively frequent resource disruptions under reasonable time.

6. Conclusions

The S-graph framework was extended to solve multi-mode RCPSPs with time-varying resource capacities. Based on previous work, a model transformation was presented to handle temporary disruptions in resource availabilities.

The efficiency of the approach was tested on modified instances of the well-known PSPLIB problem database. The results show that the S-graph solver needs more time with more resource disruptions introduced but that is expected for the more general problem class, and the solution times are still in acceptable range.
This extension is an important step for the S-graph framework towards better practical applications. However, there still exist constraints originating in practice, which have not yet been solved by S-graphs. These shortcomings provide future research topics, for example time-varying resource usages of jobs, deadlines, and time windows.

Acknowledgments

This research was supported by this fund: Széchenyi 2020 under the EFOP-3.6.1-16-2016-00017.

References

Bartusch M., Möhring R.H., Radermacher F.J., 1988, Scheduling project networks with resource constraints and time windows, Annals of Operations Research, 16, 201–240.
Brucker P., Drexl A., Möhring R., Neumann K., Pesch E., 1999, Resource-constrained project scheduling: Notation, classification, models, and methods, European Journal of Operational Research, 112(1), 3–41.
Cheng J., Fowler J., Kempf, K., Mason S., 2015, Multi-mode resource-constrained project scheduling problems with non-preemptive activity splitting, Computers and Operations Research, 53, 275–287.
Christofides N., Alvarez-Valdés R., Tamarit J., 1987, Project scheduling with resource constraints: A branch and bound approach, European Journal of Operational Research, 29(3), 262–273.
Davis E.W., Patterson J.H., 1975, A Comparison of Heuristic and Optimum Solutions in Resource-Constrained Project Scheduling, Management Science, 21(8), 944–955.
Demeulemeester E.L., Herroelen W.S., 1996, Modelling setup times, process batches and transfer batches using activity network logic, European Journal of Operational Research, 89(2), 355–365.
Hegyháti M., Holczinger T., Szoldatics A., Friedler F., 2011, Combinatorial approach to address batch scheduling problems with limited storage time, Chemical Engineering Transactions, 25, 495–500.
Hegyháti M., Ősz O., Kovács B., Friedler F., 2014, Scheduling of Automated Wet-Etch Stations, Chemical Engineering Transactions, 39, 433–438.
Hartmann S., 2013, Project scheduling with resource capacities and requests varying with time: a case study, Flexible Services and Manufacturing Journal, 25(1–2), 74–93.
Herroelen. W.S., Demeulemeester. E.L., 1992, Recent advances in branch-and-bound procedures for resource-constrained project scheduling problems, Proceedings of the Summer School on Scheduling Theory and Its Applications, Chateau de Bonas, France.
Klein R., 2000, Project scheduling with time-varying resource constraints, International Journal of Production Research, 38(16), 3937–3952.
Kolisch R., Hartmann S., 1999, Heuristic Algorithms for the Resource-Constrained Project Scheduling Problem: Classification and Computational Analysis, Chapter In: Węglarz J. (eds) Project Scheduling, International Series in Operations Research & Management Science, vol. 14, Springer, Boston, MA, USA
Kopanos G.M., Kyriakidis T.S., Georgiadis M.C., 2014, New continuous-time and discrete-time mathematical formulations for resource-constrained project scheduling problems, Computers & Chemical Engineering, 68, 96–106.
Kreter S., Rieck J., Zimmermann J., 2016, Models and solution procedures for the resource-constrained project scheduling problem with general temporal constraints and calendars, European Journal of Operational Research, 251(2), 387–403.
Mingozzi A., Maniezzo V., Ricciardelli S., Bianco L., 1998, An Exact Algorithm for the Resource-Constrained Project Scheduling Problem Based on a New Mathematical Formulation, Management Science, 44(5), 714–729.
Ősz O., Hegyháti M., 2017, Combinatorial approach for the multi-mode resource-constrained project scheduling problem, Joint EURO/ORSC/ECCO Conference 2017 on Combinatorial Optimization, Koper, Slovenia.
Sanmartí E., Puigjaner L., Holczinger T., Friedler F., 2002, Combinatorial framework for effective scheduling of multipurpose batch plants, AIChE Journal, 48(11), 2557–2570.
Xing Y., Song Z., Deng X., 2016, Optimizing the schedule of dispatching construction machines through artificial intelligence, Chemical Engineering Transactions, 51, 493–498.
Zhao H., Shen H., 2015, Optimization of resource schedule based on improved Particle Swarm Algorithm in cloud computing environment, Chemical Engineering Transactions, 46, 391–396.
Zhu G., Bard J.F., Yu, G., 2006, A Branch-and-Cut Procedure for the Multimode Resource-Constrained Project-Scheduling Problem, INFORMS Journal on Computing, 18(3), 377–390.