The generalised resource-constrained project scheduling problem with flexible resource profiles

Matthew Bold[1], Burak Boyaci[2], Marc Goerigk[3] and Chris Kirkbride[2]

[1] STOR-i Centre for Doctoral Training, Lancaster University, United Kingdom
m.bold1@lancaster.ac.uk
[2] Lancaster University Management School, Lancaster University, United Kingdom
[3] Network and Data Science Management, University of Siegen, Germany

Keywords: project scheduling, flexible resource profiles, generalised precedence constraints, schedule generation scheme, genetic algorithm.

1 Introduction

The classical resource-constrained project scheduling problem (RCPSP) consists of scheduling a set of activities, subject to resource and precedence constraints, in order to minimise the project makespan. The applicability of the RCPSP, however, is limited by the following two assumptions: 1. only finish-to-start, zero time-lag precedence relationships exist between activities, and 2. the resource requirements of each activity are fixed and constant throughout its duration. In practice, rarely do these assumptions hold true.

The extension of the RCPSP to include generalised precedence relationships addresses the first limiting assumptions, and is a well studied problem for which a number of exact (Bartusch, Möhring & Radermacher 1988, De Reyck & Herroelen 1998, Schutt, Feydy, Stuckey & Wallace 2013) and heuristic (Franck, Neumann & Schwindt 2001, Ballestín, Barrios & Valls 2011) solution methods have been developed. We refer to this problem as the generalised resource-constrained project scheduling problem (GRCPSP).

More recently, the resource-constrained project scheduling problem with flexible resource profiles (FRCPSP) has been introduced to address the second limiting assumption of the RCPSP. This problem assumes only that the total amount of resource that is required to complete each activity is known, and that, as well as the start time, the resource allocation for each activity must be determined. Whilst heuristic approaches have provided the most success in solving the FRCPSP (Fündeling & Trautmann 2010, Tritschler, Naber & Kolisch 2017), a number of exact mixed-integer programming (MIP) formulations have also been developed (Naber & Kolisch 2014, Naber 2017).

Here, we introduce the generalised resource-constrained project scheduling problem with flexible resource profiles (GFRCPSP) to combine these two extensions of the RCPSP into a single model. The GFRCPSP is an NP-hard problem for which realistically sized instances cannot be solved exactly, and hence, in addition to proposing a MIP formulation for this problem, we also propose a genetic algorithm (GA) based on a non-greedy serial schedule generation scheme with an unscheduling step.

2 Problem description

A project consists of a set of non-preemptive activities $V = \{0, 1, \ldots, n, n+1\}$, where 0 and $n+1$ are dummy source and sink activities. The GFRCPSP consists of determining a start time and resource profile of each activity, subject to a set of resource constraints and generalised precedence relationships, in order to minimise the project makespan.

There exist four types of generalised precedence relationship: start-to-start, start-to-finish, finish-to-start and finish-to-finish. Every generalised precedence relationship in a

project has an associated minimal or maximal time-lag, which together create a feasible time-window for the processing of each activity. In the GRCPSP, since the duration of each activity is known a priori, each of the four types of relationship can be transformed into a single type (Bartusch et al. 1988). In the GFRCPSP however, since activity durations are variables, these transformations do not apply, and each relationship type remains distinct. Maximal time-lags however, can be converted into negative minimal time-lags going in the opposite direction, which allows the set of project precedence constraints to be represented on a network, such as the one shown in Figure 1.

We define resource constraints for the GFRCPSP in the same way that Naber & Kolisch (2014) define them for the FRCPSP. The total resource allocated to each task $i \in V$ over its duration must at least satisfy its total resource requirement w_i, whilst adhering to upper and lower bounds on its per period resource allocation, $\underline{q}_i$ and $\overline{q}_i$, and a so-called *minimum block length*, $l^{\min}$ (Fündeling & Trautmann 2010), that is, the minimum number of time periods for which the resource allocation to an activity must remain constant. All resources $r \in R$ are assumed to be renewable, continuous, and have limited availability $R_r^{\max}$. It is also assumed that there are three types of resource: principle, dependent and independent. The allocation of principle resource to an activity determines the allocation of each dependent resource to that activity through a linear resource-function. The allocation of independent resources to an activity is fixed and independent of the allocation of the other resources.

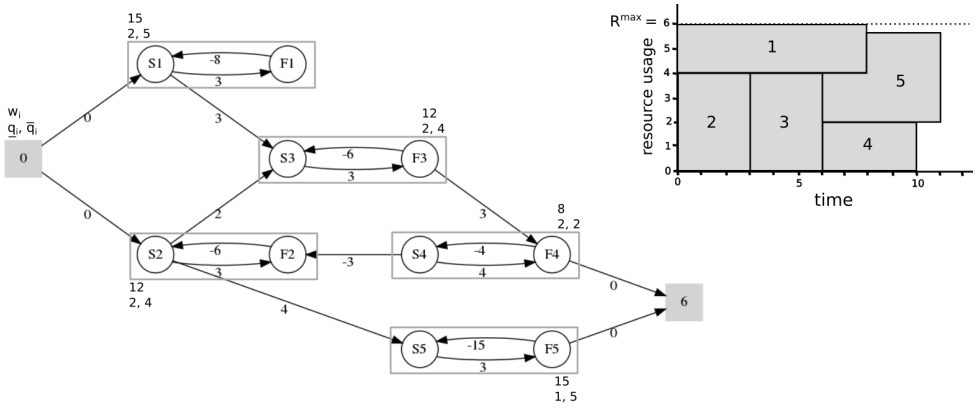

Fig. 1. An example GFRCPSP network with five non-dummy activities, a $l^{\min} = 2$, and a single resource with $R^{\max} = 6$. Arc labels indicate lower bounds on precedence relations. The chart shows an optimal solution to this problem.

3 Solution approaches

The GFRCPSP can be modelled by adapting the FP-DT3 model proposed by Naber & Kolisch (2014) for the FRCPSP. For brevity, we omit this formulation from this abstract. Solving this MIP model becomes intractable as instance sizes get larger, and hence we also propose a schedule generation scheme-based heuristic (Kolisch & Hartmann 1999) and GA for finding good solutions to realistically-sized instances in a reasonable time. We outline this scheduling heuristic and GA here.

The scheduling heuristic we propose is a non-greedy serial schedule generation scheme (SGS) with unscheduling step. This algorithm takes an activity list as input, and constructs a complete solution by scheduling activities one at a time in the given order. Each activity is started as early as possible and with as much resource as possible, subject to 'delay'

and 'greediness' parameters (Tritschler et al. 2017), which are used to encourage non-greedy scheduling. This is desirable, since an optimal solution for a given instance cannot necessarily be found using a purely greedy SGS (Fündeling & Trautmann 2010).

If a resource constraint is violated whilst scheduling an activity, the algorithm attempts to re-start the activity at the next resource-feasible start time. If a precedence constraint is violated, an unscheduling step is invoked to reschedule the activities that cause either the missed latest start or latest finish time, and start them closer to the activity with which they have a maximum time-lag with the aim of restoring the feasibility of the schedule. If the unscheduling step fails to feasibly reschedule the activities after a given number of attempts, the schedule is completed infeasibly and the total number of time periods by which precedence constraints are missed is recorded.

A GA is used to search over individual solutions, each of which consists of an activity list, and delay and greediness parameters for each activity. An initial population of individuals is generated randomly subject to precedence feasibility. Each individual is scheduled using the above heuristic. The 'fitness' of a feasible schedule is equal to its makespan, whilst the fitness of an infeasible schedule is equal to the total number of time periods by which precedence relationships are missed, plus a fixed penalty. New individuals are obtained using the adapted two-point crossover of Franck et al. (2001), which is designed to keep activities that are related by a maximal time-lag close together in the resulting activity list, thus increasing the likelihood of finding a feasible solution. The mutation operator of Hartmann (1998) is applied to each offspring solution. Having produced enough new individuals to double the original population size, the next generation is chosen using 3-tournament selection. The next generation has the same size as the original population.

4 Results and conclusions

Table 1 compares the results of the proposed GA and scheduling heuristic with the results of solving the FP-DT3-based MIP model. The two approaches are tested over five sets containing instances with 10, 20, 30, 50 and 100 activities respectively. Each of the five sets set contains 30 instances for three different values of resource strength (Kolisch, Schwindt & Sprecher 1999), resulting in a total of 450 instances. Resource strength (RS) measures the restrictiveness of the resource availability, with a smaller RS generally indicating a more challenging problem. These instances have been created using a new GFRCPSP instance generator we have developed as an extension to the ProGen/max project generator (Kolisch et al. 1999).

Table 1 shows the average percentage gap to the critical-path based lower bound of solutions found by the two solution methods. For each instance, the GA searched 50,000 schedules, whilst a limit of 2 hours was allowed for solving the MIP. To enable a fair comparison between the two solution methods, the instances for which both approaches find a feasible solution have been presented separately from those for which only the GA finds a feasible solution. There are no instances where only the MIP finds a feasible solution. These results show the MIP performing well on instances with 10 and 20 activities, but dramatically worsening over the larger test sets, as expected. In contrast, the quality of the solutions found by the GA remain roughly constant across the five test sets.

In conclusion, the GFRCPSP has been introduced to combine two existing extensions to the RCPSP. The GFRCPSP can be solved for small instances as an MIP, whilst a new scheduling heuristic and GA has been proposed for solving larger instances. Future work will include the application of this new model to a real-world scheduling problem, as well as further improvements to the metaheuristic approach proposed here, perhaps with the introduction of a local improvement step.

Test set	RS	MIP & GA			Only GA	
		#	Δ_{lb}^{MIP}	Δ_{lb}^{GA}	#	Δ_{lb}^{GA}
P10	0.05	30	10.90	13.14	0	-
	0.15	30	2.17	3.24	0	-
	0.25	30	0.49	0.49	0	-
P20	0.05	30	19.37	19.65	0	-
	0.15	30	0.25	0.49	0	-
	0.25	30	0.00	0.00	0	-
P30	0.05	28	150.81	13.43	2	10.71
	0.15	29	0.00	0.00	1	0
	0.25	30	14.56	0.00	0	-
P50	0.05	0	-	-	30	25.24
	0.15	6	617.23	0.00	24	0.00
	0.25	29	769.84	0.00	1	0.00
P100	0.05	0	-	-	30	18.81
	0.15	3	1029.66	0.00	27	0.00
	0.25	29	1442.55	0.00	1	0.00

Table 1. Average percentage gap to the critical path-based lower bound of solutions found by the MIP and GA. These values are denoted by Δ_{lb}^{MIP} and Δ_{lb}^{GA}, respectively.

References

Ballestín, F., Barrios, A. & Valls, V. (2011), 'An evolutionary algorithm for the resource-constrained project scheduling problem with minimum and maximum time lags', *Journal of Scheduling* **14**(4), 391–406.

Bartusch, M., Möhring, R. H. & Radermacher, F. J. (1988), 'Scheduling project networks with resource constraints and time windows', *Annals of Operations Research* **16**(1), 199–240.

De Reyck, B. & Herroelen, W. (1998), 'A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations', *European Journal of Operational Research* **111**(1), 152–174.

Franck, B., Neumann, K. & Schwindt, C. (2001), 'Truncated branch-and-bound, schedule-construction, and schedule-improvement procedures for resource-constrained project scheduling', *OR-Spektrum* **23**(3), 297–324.

Fündeling, C.-U. & Trautmann, N. (2010), 'A priority-rule method for project scheduling with work-content constraints', *European Journal of Operational Research* **203**(3), 568–574.

Hartmann, S. (1998), 'A competitive genetic algorithm for resource-constrained project scheduling', *Naval Research Logistics* **45**(7), 733–750.

Kolisch, R. & Hartmann, S. (1999), Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis, *in* J. Weglarz, ed., 'Project scheduling', Vol. 14, Springer, Boston, MA, pp. 147–178.

Kolisch, R., Schwindt, C. & Sprecher, A. (1999), Benchmark instances for project scheduling problems, *in* J. Weglarz, ed., 'Project scheduling', Vol. 14, Springer, Boston, MA, pp. 197–212.

Naber, A. (2017), 'Resource-constrained project scheduling with flexible resource profiles in continuous time', *Computers & Operations Research* **84**, 33–45.

Naber, A. & Kolisch, R. (2014), 'MIP models for resource-constrained project scheduling with flexible resource profiles', *European Journal of Operational Research* **239**(2), 335–348.

Schutt, A., Feydy, T., Stuckey, P. J. & Wallace, M. G. (2013), 'Solving rcpsp/max by lazy clause generation', *Journal of scheduling* **16**(3), 273–289.

Tritschler, M., Naber, A. & Kolisch, R. (2017), 'A hybrid metaheuristic for resource-constrained project scheduling with flexible resource profiles', *European Journal of Operational Research* **262**(1), 262–273.