Solving large, long-horizon resource constrained multi project scheduling problems with genetic algorithms

Brendan Hill[1], Adam Scholz[1], Lachlan Brown[1], and Dr Ana Novak[2]

[1] School of Mathematics and Statistics, University of Melbourne, Australia
`brendan.hill@gmail.com`, `adam.scholz@unimelb.edu.au`,
`l.brown25@student.unimelb.edu.au`
[2] Joint and Operations Analysis Division, Defence Science and Technology Group, Australia
`ana.novak@dst.defence.gov.au`

Keywords: RCMPSP, priority rules, genetic algorithms, large-scale/long-horizon.

1 Introduction

In this paper we adapt the *priority rule* based approach to the *Resource Constrained Multi-Project Scheduling Problem* (RCMPSP) (Villafáñez, F.A. *et. al.* 2018) for large-scale, long-horizon scheduling problems, combining the expressiveness of the priority rule framework with genetic algorithms (GA) to overcome the limitations of current methods. An efficient scheduling algorithm is described which runs in approximately linear time, enabling the use of sampling-intensive search methods in the priority rule space.

Although the algorithm can be applied to any long horizon scheduling problem, we evaluate the method in the context of a simulation of recruits flowing through military training schools. Recruit training can take years, and ensuring sufficient supply of the right people at the right time requires planning over a long horizon. Integral to this planning effort is an understanding of the yearly capacity of each training facility in progressing students through the relevant training syllabus.

We find that the the problem can be usefully framed as a RCMPSP where lessons are activities, and infrastructure and instructor specifications are resource requirements, and so the long range optimisation of training activities can be defined in project scheduling terms. Previous work in this area by the research team involved randomly sampling priorities and heuristic based optimisation of training activities along with ongoing work with Mixed Integer Linear Programming (MILP). The RCMPSP is NP-hard as it is a generalisation of the Resource Constrained Project Scheduling Problem (RCPSP) (Blazewicz, J. *et. al.* 1983). Current approaches tend to focus on heuristic methods, in particular, priority rule based methods such as found in Vázquez, E.P.*et. al.* (2013); Kurtulus IB. and Davis, E.W. (1982). Here, we adopt this representation and apply search methods over the space of possible priority rules to optimise over long-horizon objective, and find that a *random key genetic algorithm* ("RKGA") method outperforms baseline methods considered.

2 Problem formulation

A training school defines a syllabus for each student consisting of lessons of varying durations, dependencies on previous lessons and resource requirements (both infrastructure and instructor based). We define each individual student as a *project* and each of their lessons as an *activity*, possibly dependent on earlier activities, and specifying local *resource requirements* which must be satisfied in accordance with the global resource constraints of the training facility. We model over $Y = 20$ years where $\Delta = 20$ students per year are introduced, spread evenly throughout the year, defining the *release time* of each initial activity in each project.

The aim is to maximize average yearly throughput of students over the horizon, i.e. average number of *projects completed per year*. As such we adopt as our primary objective measure z_1 the number of projects completed within the horizon $\mathcal{G}(h)$ for some feasible schedule h. Additionally we consider the degree of partial completion of remaining projects $\overline{\mathcal{G}}(h)$ and hence the average remaining progress, z_2 is added. Since $z_1 \in \mathbb{N}$ and necessarily $z_2 \in [0, 1)$ this forms a lexicographical ordering within the multi-objective function:

$$\max_h z(h) = z_1 + z_2 = |\mathcal{G}(h)| + \frac{1}{|\overline{\mathcal{G}}(h)|} \sum_{p \in \overline{\mathcal{G}}(h)} \frac{\text{\# Activities completed in project } p}{\text{Total activities in project } p} \tag{1}$$

The modelling involves 20 projects per year over 20 years, each with 500 activities resulting in 200000 activities in total. We assumed 250 active training days per year with 10 hours periods of 30 minute blocks for a total of 100000 discrete time steps. Activities which could not be scheduled within the horizon were discarded.

3 The standard Priority Rule Scheduling Algorithm (PRSA)

The high volume of activities to be scheduled, combined with the need to apply sampling intensive methods, necessitates the implementation of a computationally efficient algorithm. We have adapted the *priority rule scheduling algorithm* (PRSA) from Vázquez, E.P.*et. al.* (2013) which accepts as input a set of activities $\mathcal{A}$ with durations and resource constraints, and a priority rule $\phi : \mathcal{A} \to \mathbb{R}$. Activities with no dependencies are added to the priority queue $\mathcal{Q}$ first. Then, activities are initialized when one or more dependencies are scheduled, and queued when all dependencies have been added to the schedule. Among the queued activities, we iteratively select the highest priority activity $a^* = argmax\{\phi(a) : a \in \mathcal{Q}\}$ to be scheduled. We then identify the earliest time a^* can be added to the schedule in accordance with global resource constraints, the release time of the activity and completion time of all dependencies (*). If a time cannot be found within the horizon, a^* and all subsequent dependencies are discarded. This continues until $\mathcal{Q} = \emptyset$.

A feature unique to our application is that many activities are defined by the same lesson, with the same resource requirements. This means that the earliest schedulable time for each lesson can be tracked and updated, avoiding the need to search the entire horizon in step (*) - a significant computational saving. The generation of a single schedule with 200000 activities for a fixed ϕ took on average 80ms on an i7 machine with 32GB RAM. A review of the runtime performance of extreme problem sizes up to 50M activities suggested that the algorithm had approximately linear time complexity in the number of activities.

4 Searching the priority rule space

For this PRSA algorithm, a priority rule ϕ must define the priority value for each activity, specifying the sequence in which activities are added to the schedule. Given that many activities relate to the same underlying lesson type, we define the priority rule in terms of these lessons.

4.1 Representations of priority rules space

We considered two representations of the priority rule space, firstly the direct representation, where $\alpha_i \in \mathbb{R}$ defines the priority value of lesson i and hence all related activities:

$$\phi = [\alpha_1, \ldots, \alpha_n]$$

Secondly we developed a feature-based representation of the activities, where the feature vector ζ_i defines the duration, resource requirements and other properties of lesson i. Then the actual priority rule was the weighted linear combinations β of the lesson features,where β_j defines the weighting of feature j:

$$\phi = \beta^T [\zeta_1, \ldots, \zeta_n]$$

Let $h_{\phi,x} = PRSA(\phi, x)$ be the schedule resulting from applying $PRSA$ to problem instance x with priority rule ϕ. Now, the problem is identifying $\phi^* = argmax_{\phi} z(h_{\phi,x})$.

4.2 Searching with Random Key Genetic Algorithm and baseline methods

We develop a stochastic optimizer based on the *Random Key Genetic Algorithm* (RKGA) (Gonçalves, J.F. and Resende, M.G. 2011) as follows. The priority rule ϕ is taken to be the chromosome, with a random key representation $\phi_i \in [0, 1)$, where the ranks within ϕ defined the relative priorities of activities to schedule. The k-point crossover method (Umbarkar, A.J. and Sheth, P.D. 2015) with $\gamma = 0.05$ point-wise mutation probability in each chromosome element in each generation were used, with a population size of 50. The population was evolved using GA until the function valuation limit was exceeded to provide a fair comparison with other sampling methods.

For comparison, we also applied three baseline methods:

- **NOPRIORITY**: Fixes $\phi := [0, \ldots, 0]$ to remove all prioritization of activities and default to tie-breaking rules
- **RANDOM**: Samples uniformly at random $\phi \in [0, 1]^n$ up to function evaluation limit
- **N/M**: Deterministic search with Nelder-Mead optimisation over $\phi \in \mathbb{R}^n$

The RANDOM, N/M and RKGA methods were each evaluated for both the direct representation, and the feature based representation ("**FEAT-***").

5 Preliminary experimentation

We generated 100 random problem instances which had comparable complexity to real world training facility problems in terms of numbers of activities, resources, durations and interdependencies between activities. Each search method was applied to each problem instance with a function evaluation limit of 10000. Early indications showed that RKGA consistently outperformed other methods and so a multi-start, longer-running version of this was implemented to identify the "**BESTKNOWN**" solution for each problem instance, in the absence of an exact solution or approximation bound. Then, the objective value of each search method for each instance, relative to the BESTKNOWN, was extracted as the performance measure, and the distribution of this measure over all problem instances was computed. The results are shown in Figure 1.

6 Discussion and future work

While all methods outperformed NOPRIORITY, we note that RKGA found the best solution in 92% of cases, obtaining on average 96.6% of best known objective value within the function evaluation limit, and showed significant potential for further gains with a greater limit. We observed that Nelder-Mead frequently converged on a local minimum significantly far from the best known, and may be unsuited given the high dimensionality.

Contrary to our expectations the feature based methods ("FEAT-*") performed consistently worse on average than their direct representation counterparts, possible because the

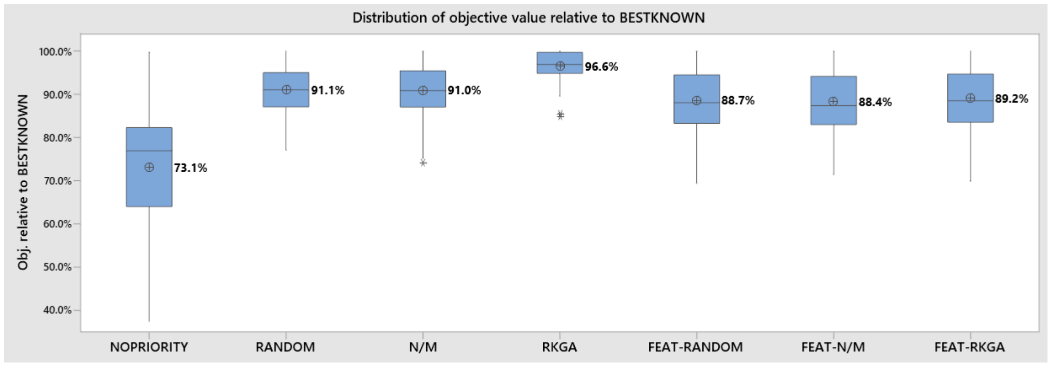

Fig. 1. Distribution of objective values relative to best known for each method (means labelled)

linear combination of raw features as $\phi = \beta^T[\zeta_1, \dots, \zeta_n]$ weights features independently and does not encapsulate feature combinations. Identifying a more expressive feature set is an important task for future work.

In parallel, we are developing a MILP formulation of the same problem but operating over a much shorter horizon e.g. 1 week, with an approximate objective function. A key research question is to compare the MILP vs RKGA models on a common set of problem instances and evaluate over a long horizon. This will allow us to directly compare the value of long-horizon metaheuristic solutions v.s. a sequence of short-horizon exact solutions. A further task of interest is to evaluate the RKGA method over a standard, public set of RCMPSP instances for direct comparison with other methods in literature.

We conclude that the RCMPSP representation is highly applicable to the long-horizon training scheduling problem. The PRSA developed is extremely efficient for large-scale RCMPSP instances scaling up to millions of activities. This enables the use of genetic algorithms and other sampling intensive metaheuristics in the automated search for efficient priority rules, while optimising directly with respect to the long-horizon objective function.

Acknowledgements

We wish to thank Prof. Terry Caelli for his insightful comments and editorial feedback.

References

Blazewicz, J., Lenstra, J.K., Kan, A H.G. Rinnooy, 1983, “Scheduling subject to resource constraints : Classification and Complexity”, *Discrete Applied Mathematics*, Vol. 05, pp. 11-24

Gonçalves, J.F. and Resende, M.G., 2011, “Biased random-key genetic algorithms for combinatorial optimization”, *Journal of Heuristics*, Vol. 17(5), pp.487-525.

Kurtulus IB. and Davis, E.W., 1982, “Multi-project scheduling: Categorization of heuristic rules performance”,*Management Science*, Vol. 28(2), pp.161-172.

Umbarkar, A.J. and Sheth, P.D., 2015, “Crossover operators in genetic algorithms: A review”, *ICTACT journal on soft computing*, Vol. 06(1).

Vázquez, E.P., Calvo, M.P. and Ordóñez, P.M., 2015, “Learning process on priority rules to solve the RCMPSP”, *Journal of Intelligent Manufacturing*, Vol. 26(1), pp.123-138.

Villafáñez, F.A., Poza, D., López-Paredes, A. and Pajares, J., 2018, “A unified nomenclature for project scheduling problems (RCPSP and RCMPSP)”. *Dirección y Organización*, Vol. 64, pp. 56-60.