Information Sciences 432 (2018) 146–163

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

On the use of genetic programming to evolve priority rules for resource constrained project scheduling problems

Shelvin Chand[a], Quang Huynh[a], Hemant Singh[a,*], Tapabrata Ray[a], Markus Wagner[b]

[a] School of Engineering & IT, University of New South Wales, Canberra ACT, Australia
[b] School of Computer Science, University of Adelaide, Adelaide SA, Australia

ARTICLE INFO

Article history:
Received 5 June 2017
Revised 4 December 2017
Accepted 7 December 2017
Available online 8 December 2017

Keywords:
Resource constrained project scheduling
Genetic programming
Heuristic evolution
Evolutionary computation
Generation hyper-heuristics

ABSTRACT

Resource constrained project scheduling is critical in logistic and planning operations across a range of industries. Most businesses rely on *priority rules* to determine the order in which the activities required for the project should be executed. However, the design of such rules is non-trivial. Even with significant knowledge and experience, human experts are understandably limited in terms of the possibilities they can consider. This paper introduces a genetic programming based hyper-heuristic (GPHH) for producing efficient priority rules targeting the resource constrained project scheduling problem (RCPSP). For performance analysis of the proposed approach, a series of experiments are conducted on the standard PSPLib instances with up to 120 activities. The evolved priority rules are then compared against the existing state-of-the-art priority rules to demonstrate the efficacy of our approach. The experimental results indicate that our GPHH is capable of producing reusable priority rules which significantly out-perform the best human designed priority rules.

1. Introduction

Classical project scheduling problem (PSP) [28] usually involves a set of non-preempt-able and precedence related activities that need to be scheduled. The structure of the project can be represented by an activity on node (AoN) graph which is a directed, acyclic and transitively reduced graph in which the nodes represent the activities and the arcs represent the precedence constraints. The precedence constraints impose that certain activities cannot start until all their predecessor activities have been completed. The classical PSP can be solved to optimality (makespan) in polynomial time using the Critical Path Method [18].

The resource constrained project scheduling problem (RCPSP) extends the classical project scheduling by taking into account constraints on the resources required to complete the activities. RCPSP is an important aspect of manufacturing and industrial operations, as it has direct impact on productivity. Therefore, it comes as no surprise that tools to handle this challenging problem are well sought after by practitioners for a wide range of applications, e.g. manufacturing systems [20], construction engineering and software development [5].

* Corresponding author.
E-mail addresses: shelvin.chand@student.adfa.edu.au (S. Chand), quang.huynh@student.adfa.edu.au (Q. Huynh), h.singh@adfa.edu.au (H. Singh), t.ray@adfa.edu.au (T. Ray), markus.wagner@adelaide.edu.au (M. Wagner).

https://doi.org/10.1016/j.ins.2017.12.013
0020-0255/

Several approaches have been proposed for solving RCPSP, ranging from classical priority rules to stochastic search algorithms [20–22]. While optimization algorithms which operate on the solution space directly have the benefit of producing good solutions, they often lack the generalization, flexibility and intuitiveness needed for real-world problems. For this reason a number of businesses prefer to rely on simple priority rules for generating schedules. A priority rule, as the name suggests, assigns priority values to the activities within the project which then determine the order in which these activities will be processed/scheduled. Priority rules provide a scheduling approach which is more understandable and usable for the project participants with different roles, from the project manager to the shop floor operators. Another advantage of using priority rules is the good scalability which is often a weak point of optimization approaches [29]. In fact, a number of optimization methods within literature employ priority rules to generate good starting solutions [13]. Last but not the least, in highly dynamic environments where quick decisions need to be made, priority rules offer a more practical choice over time consuming optimization methods.

However, the design of good priority rules is non-trivial. Evidently, even with significant knowledge and experience, human experts are limited in terms of the possibilities they can consider; an issue that becomes increasingly aggravated as the number of activities grow. Therefore, a key open problem in the field is how to design rules that have good generalization capability when applied to unseen cases. Recently, the use of genetic programming (GP) [25] has gained traction for design of efficient priority and dispatch rules in the domain of job-shop scheduling [3,29]. A GP operates using the principles of recombination, mutation and natural selection to evolve programs. Through such systematic exploration of the large (unbound) search space of possible rules, it is able to produce efficient re-usable priority rules in contrast to the meta-heuristic approaches which only provide one disposable solution. Research on heuristic evolution has also been done in some other areas such as routing [27] and bin packing [6]. However, the potential of GP for evolving rules has been scarcely explored in the field of RCPSP, except for a few foundational studies discussed in the next section. In this study, we aim to address this gap by studying automated evolution of priority rules for RCPSP using a GP approach. In order to demonstrate the benefits of the proposed approach, we also compare the evolved rules against the existing priority rules from the literature. More specifically, the key intended contributions of this paper are:

- Development and study of a GP hyper-heuristic (GPHH) framework to evolve priority rules for the classical RCPSP, using a diverse set of attributes.
- Comparison of two representation schemes (arithmetic, decision-based) for priority rules and two schedule generation schemes (serial, parallel).
- Performance characterization of the evolved rules and benchmarking with the existing rules in the literature on a large set of standard PSPLib instances [23] with up to 120 activities.

The remainder of the paper is organized as follows. Section 2 gives background on the problem and the existing literature. Section 3 gives details on the proposed GP approach for evolving rules. Section 4 provides a discussion on the results obtained while Section 5 summarizes the findings of this paper and highlights some future research directions.

2. Background and related work

2.1. Problem definition

The RCPSP involves a project with M activities which need to be scheduled while considering two types of constraints:

- *Precedence constraints*: These represent the interdependence between the activities. If activity j is a successor of activity i then activity i must be completed before activity j can be started.
- *Resource constraints*: These represent realistic limitations on resources such as manpower, budget, etc. Each project is assigned a set of K renewable resources where each resource k is available in R_k units for the entire duration of the project. Each activity may require one or more of these resources to be completed. While scheduling the activities, the daily resource usage for resource k can not exceed R_k units.

Each activity j takes d_j time units to complete. The overall goal of the problem is to minimize the *makespan*, i.e., the total duration from the start of first to the end of last scheduled activity. The problem is NP-hard [2] and therefore the exact methods [7] are practical only for relatively small instances. For larger instances, heuristic/meta-heuristic methods such as genetic algorithm [37], particle swarm optimization [24], simulated annealing [34] and hybrid search [1] etc. are used to obtain good (but not necessarily optimal) solutions.

2.2. GPHH for evolving priority heuristics

Priority heuristics are often used for solving combinatorial optimization problems as they are simple and computationally efficient. They are used by decision makers to decide on which action to perform next based on a “priority value” among the candidates. Instead of relying on human experts to design new heuristics, researchers are now resorting to genetic programming based hyper-heuristics (GPHH) to automate this process. GPHH’s usually operate on a set of problem attributes and mathematical operators to construct heuristics. This can be both time saving as well as help in discovering rules which may not be easy to construct manually.