Future Generation Computer Systems 86 (2018) 211–221

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

# Evolving priority rules for resource constrained project scheduling problem with genetic programming

Mateja Đumić [a,*], Dominik Šišejković [b], Rebeka Čorić [a], Domagoj Jakobović [c]

[a] *Department of Mathematics, University of Osijek, Trg Ljudevita Gaja 6, Osijek, Croatia*
[b] *Institute for Communication Technologies and Embedded Systems, RWTH Aachen University, Germany*
[c] *Faculty of Electrical Engineering and Computing, Zagreb, Croatia*

## HIGHLIGHTS

- A scheduling procedure consisting of a meta-algorithm and priority function is shown.
- A genetic programming approach is used for generating the priority function.
- A comparison of three different approaches of meta-algorithm is made.
- Performances of the suggested approaches are compared with other priority rules.
- Results have shown adaptability of this approach for various optimization criteria.

## ARTICLE INFO

*Article history:*
Received 6 November 2017
Received in revised form 28 March 2018
Accepted 11 April 2018
Available online 22 April 2018

*MSC:*
00-01
99-00

*Keywords:*
Genetic programming
Resource constrained scheduling
Hyper-heuristics

## ABSTRACT

The main task of scheduling is the allocation of limited resources to activities over time periods to optimize one or several criteria. The scheduling algorithms are devised mainly by the experts in the appropriate fields and evaluated over synthetic benchmarks or real-life problem instances. Since many variants of the same scheduling problem may appear in practice, and there are many scheduling algorithms to choose from, the task of designing or selecting an appropriate scheduling algorithm is far from trivial. Recently, hyper-heuristic approaches have been proven useful in many scheduling domains, where machine learning is applied to develop a customized scheduling method. This paper is concerned with the resource constrained project scheduling problem (RCPSP) and the development of scheduling heuristics based on Genetic programming (GP). The results show that this approach is a viable option when there is a need for a customized scheduling method in a dynamic environment, allowing the automated development of a suitable scheduling heuristic.

## 1. Introduction

Scheduling is a process that deals with the allocation of resources to tasks over given time periods. It is used on everyday basis in many manufacturing and services industries. The task of scheduling is to optimize one or more objectives. Often, the process includes some limitations like the capacity of vehicles, working time of employees, limited funds etc.

The focus of this work is on one of the scheduling problems with limitations—the resource constrained project scheduling problem (RCPSP). RCPSP is a problem with two kinds of constraints—precedence and resource. One job can require other activities to be completed before it starts with processing. All jobs need resources in various amounts, and resources are limited. The goal of solving RCPSP is to schedule all jobs in a given project in way that all constraints are met, and one or more criteria are optimized.

Many real life problems can be formulated as RCPSP, and due to the fact that RCPSP is very hard to solve, many solving methods were developed. The decision of which solving method to use must be made depending on the problem characteristics. Usually, solving methods can be divided into two groups—exact methods and heuristics. Exact methods are impractical on problems which have a large number of jobs to schedule, so the heuristic methods are mostly used. Some heuristic methods show better results on specific problem instances, but no heuristics that are good for all problems exist (proven by the No free lunch theorem [1]). Because of that there are some authors that try to combine different heuristic approaches. A common example of this approach are genetic algorithms combined with local search methods.

* Corresponding author.
*E-mail addresses:* mdjumic@mathos.hr (M. Đumić), sisejkovic@ice.rwth-aachen.de (D. Šišejković), rcoric@mathos.hr (R. Čorić), domagoj.jakobovic@fer.hr (D. Jakobović).

https://doi.org/10.1016/j.future.2018.04.029
0167-739X/© 2018 Elsevier B.V. All rights reserved.

Since heuristic methods give good results only when applied to problems containing specific characteristics, thereby being application specific, there is a need to rise to a higher level of solving—searching the space of solving methods, and not searching the solution space. This is why hyper-heuristics are being developed. Genetic programming (GP) is one of algorithms that can be used as a hyper-heuristic which produces new heuristics. This paper demonstrates the use of GP as a hyper-heuristic to evolve appropriate scheduling heuristics for the RCPSP.

The paper is organized as follows: in Section 2 a definition of RCPSP is given, while in Section 3 a brief overview of methods for solving RCPSP is given. Section 4 describes how GP can be used in evolving new scheduling heuristics in form of priority rules. The results are presented in Section 5, and short conclusion and future research directions are given in Section 6.

## 2. RCPSP

In general, the resource constrained project scheduling problem considers activities and resources. Activities have known durations and resource demands, while resources are of limited availability. Additionally, activities are linked by precedence relations. The problem consists of finding a feasible schedule with minimal total duration by assigning start times to each activity such that the precedence and resource constraints are satisfied.

All *activities* constituting the project are defined by the set $A = \{A_0, \ldots, A_{n+1}\}$. By convention the activities $A_0$ and $A_{n+1}$ represent the start and the end of a schedule. These activities are usually referred to as dummy activities. Duration of activity $A_i$ is $p_i$, and duration of both dummy activities is 0. An activity cannot be interrupted once it is started. This property is referred to as not allowing *preemption*.

The *precedence relation* is given by the set $E$ which defines pairs such that $(A_i, A_j) \in E$ means that activity $A_i$ precedes activity $A_j$. Additionally, we assume that $A_0$ precedes all other activities and that $A_{n+1}$ succeeds all other activities.

Generally, resources can be categorized as *renewable*, *non-renewable* and *doubly-constrained* [2]. In this article we shall work exclusively with renewable resources which are available at any given time with full replenishment capacity. Each activity has demands on resources and in order for an activity to execute, all demands have to be satisfied. In the most of cases, goal of the problem is to minimize makespan, i.e. total project duration time, but objective function can be anything else like profit maximization, the uniform use of resource etc.

To make problem easier to solve for each activity in the problem instance a variety of properties can be calculated to define the *time-frame* in which a specific activity can be scheduled. Let the set of all activities that can be scheduled at a given time $t$ be defined as $E(t)$—the set of eligible activities at time $t$. Then we can define $E(t)$ as follows:

$$E(t) = \{A_j : A_j \in A, ES_j + 1 \leq t \leq LF_j\}, \tag{1}$$

where $ES_j$ stays for the earliest start time of activity $A_j$, and $LF_j$ for the latest finish time of activity $A_j$. i.e. each activity $A_j$ can be scheduled and executed in the given time frame between its earlier start and latest finish time, where we assume integer time values. More about how to calculate this time-frame for each activity can be found in [3] with remark that the aforementioned calculation procedures assume unlimited resources and rely only on the precedence relations.

The concrete difficulty of a RCPSP instance depends upon many different parameters of which the following are mentioned as most important in literature [4]: network complexity (NC) and the effect of resources: resource factor (RF) and resource strength (RS). Experiments conducted by *Kolisch et al.* [5] show that a negative but very weak correlation exists between the network complexity and the project execution time, while a great magnitude of a positive correlation between the resource factor and execution time exists as for a negative correlation between the resource strength and the execution time.

## 3. Previous work

RCPSP was introduced in 1963. by Kelley. Even though it is a more than 50 years old problem, it is still interesting and new solving methods are still being developed. According to the computational complexity theory, the RCPSP is one of the most intractable combinatorial optimization problems and belongs to the class of *NP-hard* problems [6]. RCPSP solving methods can generally be divided into exact and heuristic approaches [4].

Exact solving approaches search the complete space of feasible solutions and therefore guarantee optimality [7]. But the search space is often of impractical size which makes such approaches almost useless for a very large number of problem instances [8]. However, in literature there are few examples of exact algorithms that produce good results on small problem instances like mathematical planning [9] and numerical methods like dynamical programming [10]. Exact algorithms can be divided into the following four main categories: Integer Programming [3,11], Implicit Enumeration [12,13], Branch-and-bound [14,15], Dynamic Programming [16].

As exact methods are generally not applicable for larger problem instances, many different heuristic approaches have been developed. Heuristics differ from exact methods by searching only a part of the solution space which offers possible better performance in a given time frame but not optimality. Nevertheless, in most cases generating a feasible and good enough solution is far more important than optimality. Therefore heuristic approaches are a popular and useful option for solving the RCPSP.

Known scheduling heuristics for solving the resource constrained project scheduling problems can further be classified into two categories: *priority rule-based methods* (or constructive heuristics) and *metaheuristic-based approaches* (or improvement heuristics) [17].

The first class of methods always starts with no scheduled activity. Construction of a complete schedule is controlled by combination of schedule generation scheme [18] and *priority rules* [17]. The general idea is that the SGS builds a schedule from scratch taking resource and precedence constraints into account. During the building process, the SGS upgrades *partial schedules* until all activities are scheduled and a complete feasible schedule is generated. Which activity SGS is going to schedule next depends on priority rule. In the literature, two different schedule generation schemes are available: the *serial* SGS and the *parallel* SGS. Both schemes generate feasible schedules but differ in the way activities and time is handled throughout the procedures. More about SGS can be found in [18].

The second class of methods is applied to initial complete solutions with the goal of achieving improvement in terms of a selected criterion. The main representatives of this category are genetic algorithms [19,20], tabu search, simulated annealing [21], ant colony optimization [22] and others.

One of the heuristic methods, that is not mentioned yet, is Genetic programming (GP). In the last 15 years GP was used in scheduling and achieved good results. GP has been used for wide variety of environments like single machine scheduling [23,24], multiple machine scheduling [25], job shop scheduling [26–28], unrelated machine environment [29]. In scheduling GP is mostly used for evolving dispatching rules (priority functions) and recently Branke et al. [30] brought the survey of GP approaches used in scheduling. Also in literature we can find comparison of