# Probabilistic Model Enhanced Genetic Algorithm for Multi-Mode Resource Constrained Project Scheduling Problem

Mayowa Ayodele
Robert Gordon University
Aberdeen, Scotland
m.m.ayodele@rgu.ac.uk

John McCall
Robert Gordon University
Aberdeen, Scotland
j.mccall@rgu.ac.uk

Olivier Regnier-Coudert
Robert Gordon University
Aberdeen, Scotland
o.regnier-coudert@rgu.ac.uk

## ABSTRACT

The Multi-mode Resource Constrained Project Scheduling Problem (MRCPSP) is composed of two interacting sub-problems which are the mode assignment problem and the scheduling problem. Solving these sub-problems in isolation pose a challenge due to the interaction that exists between them. We present a unified approach for applying a combination of algorithms to the sub-problems of the MRCPSP. The use of Genetic algorithms and Estimation of Distribution Algorithms aims to exploit efficiently and synchronously the distinct search spaces presented by the two sub-problems.

## Categories and Subject Descriptors

I.6.5 [**SIMULATION AND MODELING**]: Model Development—*Modeling methodologies*; I.2.8 [**ARTIFICIAL INTELLIGENCE**]: Problem Solving, Control Methods, and Search—*Scheduling, Heuristic methods*

## Keywords

Probabilistic Model, Genetic Algorithm, Project Scheduling, Hybrid, Estimation of Distribution Algorithm

## 1. INTRODUCTION

The Multi-mode Resource Constrained Project Scheduling Problem (MRCPSP) entails allocating a mode of execution, start and finish times to all activities of a project. This problem combines mode assignment with scheduling. The most common objective of this problem is to minimise the total project duration (makespan). Genetic Algorithms (GAs) have been very widely applied to the MRCPSP and the experimental review by Van Peteghem and Vanhoucke in [4] shows that GA produces the best results for larger data instances of the MRCPSP. Estimation of Distribution Algorithms (EDAs) on the other hand have much fewer applications. This may be attributed to the fact that the activity ordering aspect of the MRCPSP is naturally represented as permutations. Permutation search spaces are more complex to model in EDAs [1]. In this work, we respectively apply EDA and GA to the mode assignment and activity ordering sub-problems of the MRCPSP. We explain how these algorithms are combined to solve the MRCPSP.


*GECCO '15 July 11-15, 2015, Madrid, Spain*

ACM ISBN 978-1-4503-3488-4/15/07.
DOI: http://dx.doi.org/10.1145/2739482.2765535

## 2. PROBLEM DEFINITION

In the MRCPSP, there are a set of activities $act_i, i \in [1, n]$, a set of renewable resources $A$ and a set of non-renewable resources $B$. We denote by $\alpha max_r$, the per period availability of the renewable resource $r$, $r \in [1, |A|]$ and by $\beta max_l$ the overall availability of the non-renewable resource $l$, $l \in [1, |B|]$. Activities $act_1$ and $act_n$ are dummy activities that respectively represent the start and end of a project. Every activity $act_i$ with the exception of $act_1$ has a set of predecessor(s) $Pred_i$. We denote as $pred_{i,j}$ its $j$-th predecessor, $j \in [1, |Pred_i|]$. Finally, $act_i$ can be performed in mode $mode_{i,k}$, $k \in [1, m_i]$, where $m_i$ is the number of possible modes for $act_i$. Each mode $mode_{i,k}$ is composed of an integer vector of renewable resources $(\alpha_{i,k,1}, ..., \alpha_{i,k,|A|})$, an integer vector of non-renewable resources $(\beta_{i,k,1}, ..., \beta_{i,k,|B|})$ and is associated with an execution time $t_{i,k}$. The objective of the MRCPSP is to determine for each activity $act_i$, its start time $st(act_i)$, finish time $ft(act_i) = st(act_i) + t_{i,k}$ and the mode $mode_{i,k}$ in which it is executed, so that the finish time of the last activity $ft(act_n)$ is minimised, subject to:

$$\sum_{act_a}^{n} \beta_{a,k_a,l} \leq \beta max_l \forall\, l, l \in [1, |B|] \tag{1}$$

$\forall\, act_i$

$$ft(pred_{i,j}) \leq st(act_i)\ \forall\ pred_{i,j} \in Pred_i \tag{2}$$

Let $C_i$ be the set of activities that clash with $act_i$ such that $C_i = \{a \in A : (ft(act_a) > st(act_i) \wedge st(act_a) < ft(act_i))\}$, then

$$\alpha_{i,k,r} + \textstyle\sum_{a \in C_i} \alpha_{a,k_a,r} \leq \alpha max_r \forall r, r \in [1, |A|] \tag{3}$$

We describe the non-renewable resource, precedence and renewable resource constraints in (1), (2) and (3) respectively. Note that in (1) and (3), $\beta_{a,k_a,l}$ is the amount of non-renewable resource $l$ required by $act_i$ performed in its allocated mode $k_a$ while $\alpha_{a,k_a,r}$ is the amount of renewable resource $r$ required by $act_i$ performed in its allocated mode $k_a$.

# 3. PROPOSED METHOD

We divide the MRCPSP into two sub-problems: the "mode assignment" and "activity ordering" problems as recommended in [4]. One advantage of dividing the problem this way is that we are able to use different representations for the sub-problems. We use the integer representation for the mode assignment and permutation representation for the activity ordering sub-problem. This makes it easier for standard genetic operators to be applied. Van Peteghem and Vanhoucke [4] considers the mode assignment sub-problem as the more important sub-problem and recommends it as an area for further research. This is because it determines whether or not a solution is feasible and contributes significantly to the overall makespan. For this reason, we concentrate on the mode assignment aspect of the problem.

The standard GA is not well suited for solving the mode assignment sub-problem as it is limited in its ability to learn parameter interactions in the problem. Van Peteghem and Vanhoucke [3] incorporated a mode improvement local search method to improve the performance of the GA. EDAs on the other hand are able to learn the interactions that exists between the parameters of a problem. We therefore modify the approach presented in [3] by changing the way modes are allocated to activities.

The GA presented in [3] is called the bi-population GA (BPGA). According to the review in [4], the BPGA amongst other GAs performed best on most of the test problems in terms of the ability of the algorithm to find solutions with minimal makespan. The BPGA uses two different populations of solutions $POP_L$ and $POP_R$. Solutions in $POP_L$ are left-justified (i.e the makespan is calculated by scheduling activities as early as possible based on the ordering of activities) while solutions in $POP_R$ are right-justified (i.e the makespan is calculated by scheduling activities as late as possible). The algorithm begins with $POP_L$ and iteratively alternate between both populations. Solutions produced by one population are fed into the other. Algorithmic details of this approach can be found in [3].

In algorithm 1, we specifically describe how new solutions are produced at each generation after hybridising the GA from [3] with an EDA. Activity orderings are generated as suggested in [3] and thus offspring orderings are always feasible. Each solution in $POP_L$ consists of an activity ordering and a mode assignment. The activity ordering of each solution is selected as one parent $p1_s$ while the other parent $p2_s$ is selected using tournament selection. The two solutions produced by breading those parents are denoted as $c1_s$ and $c2_s$.

## 3.1 Modeling and Sampling Mode Assignments

The EDA used is the Univariate Marginal Distribution Algorithm which builds a completely new probabilistic model at each generation [2]. A model is continually sampled for a whole generation. This will save on the extra effort lost to the use of local search improvement techniques.

The probabilistic matrix is created from the best $sSize$ solutions in population $P$ as shown in (Alg. 1, lines 2 - 3). We create a new mode assignment $c_m$ (which can be $c1_m$ or $c2_m$) by sampling $M_P$.

Let $\mu_{ik}$ be the probability of assigning mode $k$ to $act_i$.

$$\text{Then, } \mu_{ik} = \frac{\sum_{q=1}^{sSize}[mode_{i,k} = k]}{sSize} \qquad (4)$$

**Algorithm 1** Generation of $POP_R$ using GA and EDA mechanisms for the MRCPSP

1: initialise $POP_R = \emptyset$
2: select best $sSize$ solutions from $POP_L$ to form $S$, where $sSize < |POP_L|$
3: build probabilistic model $M_p$ from mode assignments of solutions in $S$
4: **for** $q = 1$ to $|POP_L|$ **do**
5:     sample $M_p$ to produce two mode solutions $c1_m$ and $c2_m$
6:     define $p1_s$ as the activity ordering of the $q{-}th$ solution in $POP_L$
7:     define $p2_s$ as the activity ordering of solution $p2$ selected from $POP_L$ using tournament selection
8:     perform crossover on $p1_s$ and $p2_s$ to produce activity orderings $c1_s$ and $c2_s$
9:     perform mode mutation on $c1_s$, $c2_s$ and ordering mutation on $c1_m$ and $c2_m$
10:     combine $c1_s$ with $c1_m$ and $c2_s$ with $c2_m$ to produce $c1$ and $c2$
11:     evaluate $c1$ and $c2$, set $c$ as the best of $c1$ and $c2$
12:     insert $c$ in $POP_R$
13: **end for**

# 4. CONCLUSIONS

This work demonstrates how the MRCPSP may be split up into two sub-problems so that distinct methods can be applied to each. This allows algorithms adapted to each sub-problems to be used in a unified way. In this work, a GA is used to improve the quality of the activity orderings while an EDA focuses on generating mode assignments. Future work should focus on applying alternative EDAs such as Population-Based Incremental Learning and multivariate approaches.

# 5. REFERENCES


[1] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. *Progress in Artificial Intelligence*, 1(1):103–117, 2012.

[2] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[3] V. Van Peteghem and M. Vanhoucke. A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. *European Journal of Operational Research*, 201(2):409–418, 2010.

[4] V. Van Peteghem and M. Vanhoucke. An experimental investigation of metaheuristics for the multi-mode resource-constrained project scheduling problem on new dataset instances. *European Journal of Operational Research*, 235(1):62–72, 2014.