Elitist Vector Evaluated Particle Swarm Optimization for Multi-mode Resource Leveling Problems *

Yan GUO 1,*; Nan LI 2; Haolan ZHANG 1; Tingting YE 1

1 Ningbo Institute Of Technology, Zhejiang University, Ningbo 315100, China
2 College of Economics and Management, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

Abstract

This paper focuses on solving a typical multi-mode resource leveling problem, in which activity duration depends on committed resources, project deadlines and other constraints. To solve this problem, we establish a multi-objective model to minimize project duration, resource requirements and resource variance. Based on the established model, a novel Elitist Vector Evaluated Particle Swarm Optimization (EVEPSO) is utilized for solving this problem. A set of experiments have been conducted based on EVEPSO method: and the empirical results indicate that EVEPSO can accurately and efficiently solve the multi-mode resource leveling problem. The computational results further suggest that project duration and minimum resource variance correlate negatively. However, there are few correlations between resource requirements and minimum resource variance.

Keywords: Project Management; Resources Leveling; Multi-objective Optimization; Particle Swarm Optimization

1 Introduction

Resource leveling problems have been investigated quite intensively in recent years since they are becoming a key challenge in various practical applications. In a traditional resource leveling problem, the duration and resource requirements of each activity are fixed. The major techniques for solving resource leveling problems can be categorized into three areas: exact procedures[1], heuristic methods[2] and evolutionary algorithms[3].

For many real-life projects in enterprise with small-batch production, such as construction projects, software development projects, and aircraft manufacturing projects, each activity is executed in one of several modes. Each mode represents a combination of its resource requirements and its duration. It is crucial to not only adjust the start time, but also select the execution mode

*Project supported by the National Nature Science Foundation of China (Grant No. 70871111) and the Zhejiang Philosophy and Social Science Project Fund (Grant No. 11JCSH03YB).
*Corresponding author.
Email address: guoyanbox@126.com (Yan GUO).
of each activity when minimizing the variation of resource utilization. This issue has affected multi-mode resource leveling problems. For a multi-mode resource leveling problem, the duration and resource requirements of each activity are not fixed. These constraints vary according to different execution mode. In order to solve multi-mode resource leveling problems, the project deadlines, resource limit as well as the constraints of traditional resource leveling problems should be taken into consideration.

Intrinsically, multi-mode resource leveling problems belongs to Multi-mode Resource-Constrained Project Scheduling Problems (MRCPSP). In a MRCPSP[4, 5, 6], the activities of a project are to be scheduled in order to minimize the project duration while they are also subject to precedence constraints and limited availability of the resources. For MRCPSP, if we only take a single resource into consideration, it is known as the Discrete Time-Cost Tradeoff Problem (DTCTP). This problem has been studied by Demeulemeester et al.[7].

Multi-objective evolutionary algorithms seem particularly suitable to solve MRCPSP because they can deal with a set of possible solutions simultaneously. One of these algorithms is Vector Evaluated Particle Swarm Optimization (VEPSO) proposed by Omkar et al.[8]. This method employs separate swarms for each of the objective and information migration between these swarms. It ensures VEPSO is able to obtain more possible solutions than multi-objective Pareto based particle swarm optimization[9, 10, 11], therefore VEPSO proves to be more appropriate for current problems. In contrast to multi-objective Pareto based particle swarm optimization, VEPSO is inferior in convergence and optimal efficiency. To solve this problem, the authors apply Pareto dominance strategy to VEPSO and propose a new novel EVEPSO method.

The remainder of this paper is organized as follows. In Section 2, we formulate the multi-mode resource leveling problem and present its mathematical model. Section 3 describes the definition of EVEPSO. In Section 4, we apply EVEPSO to the multi-mode resource leveling problem, and illustrate EVEPSO’s optimization processes. The experimental results and comparisons are reported in Section 5. Finally, Section 6 concludes the paper.

2 Problem Formulation and Its Mathematic Model

2.1 Problem formulation

The problem discussed adheres to the following assumptions:

- A project consists of \( n \) activities that are represented through using an Activity-On-Arc (AOA), where nodes represent network 'events' and arcs denote network 'activities'.

- Each activity \( T_i (i = 1, 2, ..., n) \) has to be executed in one of \( M_j (j = 1, 2, ..., m) \) modes, where each activity-mode combination has a fixed duration, and requires a single nonrenewable resource (such as money).

- (Duration, Resource) refer to a 'mode'. For example, all modes of activity \( T_i \) can be represented as \( \{(D_{i1}, R_{i1}), (D_{i2}, R_{i2}), ..., (D_{im}, R_{im})\} \). In these modes, if \( l < r \), among the constraints 
  1) \( D_{il} < D_{ir} \) and \( R_{il} > R_{ir} \), 2) \( D_{il} > D_{ir} \) and \( R_{il} < R_{ir} \), only one is tenable. It means that the longer the duration is, the lower the resource requirement is.

- Activity \( T_i \) cannot start unless all of its predecessors have been completed.

- In the implementation process of an individual activity, the resource is consumed evenly.
Based on the difference between the constraints and the objective, multi-mode resource leveling problems can be fourfold. Problem 1: it aims at minimizing the resource variance of a project in which the given deadlines and resource limit are met. Problem 2: it involves minimizing the resource variance and total resource requirement of the project without exceeding a given deadline. Problem 3: it aims at minimizing the resource variance and duration of the project without exceeding a given resource limit. Problem 4: The objective of this problem is to obtain optimal solutions that minimize the resource variance, resource requirement and duration synchronously under the conditions of unfixed deadline and resource limit.

Problem 4 is the one we discuss in this paper. It aims at minimizing the resource variance for each duration-resource combination, and is the integration of the other three problems. In this paper, the method used to solve this problem is to minimize the project duration $P_D$, total resource requirement of the project $P_R$ and resource variance $RV$ by adjusting the start time and the execution mode of each activity.

### 2.2 Mathematic model

Based on the above analysis, the mathematic model for this problem can be described as follows:

**Objectives:**

1. \( \min_P P_D = \sum_{i \in CP} \sum_{j=1}^{m} D_{ij} X_{ij} \) \hspace{1cm} (1)
2. \( \min_P P_R = \sum_{i=1}^{n} \sum_{j=1}^{m} R_{ij} X_{ij} \) \hspace{1cm} (2)
3. \( \min RV = \frac{1}{P_D} \sum_{t=1}^{PD} (R(t) - \bar{R})^2, \bar{R} = \frac{1}{P_D} \sum_{t=1}^{PD} R(t) \) \hspace{1cm} (3)

where \( CP \) denotes the critical path of the project and \( R(t) \) represents resource demand of the project on day \( t \). \( R(t) \) is determined as follows:

1. On the basis of the \( X_{ij} \) value, the duration \( D(T_i) \) and resource requirement \( RA(T_i) \) for activity \( T_i \) are calculated as

\[
D(T_i) = \sum_{j=1}^{m} D_{ij} X_{ij}, \quad RA(T_i) = \sum_{j=1}^{m} R_{ij} X_{ij}
\]  \hspace{1cm} (4)

where \( X_{ij} \) is assigned a value in \( \{0,1\} \), which is a decision variable for confirming the execution mode of each activity.

2. Compute the daily resource requirements \( R(T_i) \) of activity \( T_i \).

\[
R(T_i) = RA(T_i)/D(T_i)
\]  \hspace{1cm} (5)

3. Compute resource demand of activity \( T_i \) on day \( t \).

\[
R_t(T_i) = \begin{cases} 
R(T_i), & \text{if } ST(T_i) < t \leq FT(T_i) \\
0, & \text{if } t \leq ST(T_i) \text{ or } t > FT(T_i)
\end{cases}
\]
where $ST(T_i)$, $FT(T_i)$ are the start time and the finish time of activity $T_i$, and $R_t(T_i)$ denotes the resource demand of activity $T_i$ on day $t$.

(4) Compute resource demand of the project on day $t$.

$$R(t) = \sum_{i=1}^{n} R_t(T_i)$$

Constraints:

$$\sum_{j=1}^{m} X_{ij} = 1, (X_{ij} = 0, 1) \quad (6)$$

$$ES(T_i) \leq ST(T_i) \leq LS(T_i) \quad (7)$$

$$\max\{FT(T_b) | T_b \in PA(T_i)\} \leq ST(T_i) \quad (8)$$

Let $ES(T_i)$, $LS(T_i)$, $PA(T_i)$ be the earliest start time, the latest start time and the predecessors of activity $T_i$. Constraint (6) ensures that each activity $T_i$ is executed in only one mode. (7) and (8) respectively, represent precedence constraints.

### 3 Elitist Vector Evaluated Particle Swarm Optimization

Similar to VEPSO, EVEPSO is a co-evolutionary method that employs separate swarms. Each of the objectives and information migration between these swarms ensures an optimal solution with respect to all the objectives. The main features of EVEPSO are described in detail as follows:

The EVEPSO method supposes that $p$ swarms contain $^1Subsum$, $^2Subsum$, ..., $^pSubsum$. Each swarm’s size $n$ aims to optimize simultaneously $p$-objective functions. The $k$th swarm $^kSubsum$ aims to optimize the $k$th objective $f_k(U)$. In this paper, the left superscript $[k]$ of each variable represents that this variable is correlated to the $k$th swarm, and $k = 1, 2, ..., p$.

The updating method of the personal best position for EVEPSO is the same as VEPSO, viz. The personal best position of particles in the $k$th swarm $^k[gbest]$ is updated according to the $k$th objective. But the initialization and update method of the global best position for EVEPSO is thoroughly different from VEPSO. In EVEPSO, there is an elitist archive of non-dominated solutions. In the evolutionary process, EVEPSO should select $p$ non-dominated solutions (can be repeated) from elitist archive and use those as global best position of swarms through the method as follows. According to each non-dominated solution in an elitist archive, we have the following definitions:

(1) Let the position of non-dominated solution, which its value of the first objective $f_1(U)$ is minimum, to be the global best position of the $p$th swarm $^p[gbest]$;  

(2) Let the position of non-dominated solution, which its value of the second objective $f_2(U)$ is minimum, to be the global best position of the first swarm $^1[gbest]$;  

(3) Let the position of non-dominated solution, which its value of the third objective $f_3(U)$ is minimum, to be the global best position of the second swarm $^2[gbest]$, and so on;  

(4) Let the position of non-dominated solution, which its value of the $p$th objective $f_p(U)$ is minimum, to be the global best position of the $(p - 1)$th swarm $^{p-1}[gbest]$.
Through the above definitions, each swarm can share the information from all non-dominated solutions. Therefore, EVEPSO is able to guide the particles to Pareto front as soon as possible. EVEPSO is similar to VEPSO. Thus, in each iteration $\text{iter}$, the $a$th particle in the $k$th swarm adjusts its current position $[k]\mathbf{x}_{ai}(\text{iter})$ and velocity $[k]\mathbf{v}_{ai}(\text{iter})$ through each dimension $i$ by the personal best position $[k]\mathbf{pbest}_{ai}$ and the global best position $[k]\mathbf{gbest}_{ai}$ using equation (9) and (10):

$$
[k]\mathbf{v}_{ai}(\text{iter} + 1) = [k]\omega \times [k]\mathbf{v}_{ai}(\text{iter}) + [k]c_1 \times r_1 \times ([k]\mathbf{pbest}_{ai} - [k]\mathbf{x}_{ai}(\text{iter})) + [k]c_2 \times r_2 \times ([k]\mathbf{gbest}_{ai} - [k]\mathbf{x}_{ai}(\text{iter}))
$$

(9)

$$
[k]\mathbf{x}_{ai}(\text{iter} + 1) = [k]\mathbf{x}_{ai}(\text{iter}) + [k]\mathbf{v}_{ai}(\text{iter} + 1)
$$

(10)

where $[k]\omega$ is inertia weight, $[k]c_1$, $[k]c_2$ are the acceleration constants and $r_1, r_2$ are random real numbers drawn from $U(0, 1)$.

Here, the inertia weight $[k]\omega$ is adjusted dynamically during the optimization:

$$
[k]\omega = [k]\omega_{\text{max}} - \left(\frac{[k]\omega_{\text{max}} - [k]\omega_{\text{min}}}{\text{iter}_{\max}}\right) \times \text{iter}
$$

(11)

where $[k]\omega_{\text{max}}$ is the initial weight factor, $[k]\omega_{\text{min}}$ is the final weight factor, $\text{iter}$ is the current iteration number and $\text{iter}_{\max}$ is the maximum number of iterations.

4 EVEPSO Algorithm for Solving the Multi-mode Resource Leveling Problem

4.1 Coding design

In our experimental design, the decision variable and start time have been implemented in the coding. The EVEPSO algorithm for solving the multi-mode resource leveling problem is supported; and the scope of the solution equals to $n$-dimensional search space where $n$ is total number of activities. EVEPSO aims at three objectives - project duration, total project resource requirement and resource variance. EVEPSO employs three swarms with the same population to probe the search space and information exchanged among them.

The position of the particle $[k]\mathbf{x}_a = ([k]\mathbf{x}_{a_1}, [k]\mathbf{x}_{a_2}, ... , [k]\mathbf{x}_{ai}, ... , [k]\mathbf{x}_{an})$ corresponds to a solution for the problem, and the $i$th dimension of the position $[k]\mathbf{x}_{ai}(a = 1, 2, ... , M; i = 1, 2, ..., n)$ denotes the decision variable and start time of the $i$th activity. The coding design of particle velocity is similar to the design of particle position, which is composed of component controlled decision variable $vX_{ij}$ and component controlled the start time $vST(T_i)$.

4.2 The optimization process

**Step 1** Randomly generate three swarms $[1]\text{Subswm}, [2]\text{Subswm}, [3]\text{Subswm}$ with the same population. Then the particle position and the particle velocity of all particles are initialized. The detailed method of initialization is described as follows:

a. Initialize the decision variable in the particle position to decide the execution mode of activities. Decision variable $X_{ij}$ is randomly initialized to 0 or 1, but it must satisfy the constraint (6).
b. Initialize the start time in the particle position. \( ST(T_i) \) can be randomly initialized within \([ES(T_i), LS(T_i)]\). Then, the procedure checks whether \( ST(T_i) \) satisfies constraint (8), if not, EVEPSO randomly generates it within \([\max\{FT(PA(T)), ES(T_i)\}, LS(T_i)]\) again.

c. Initialize the particle velocity. To prevent the particle velocity \([^k]v_{ai}\) from increasing too fast, \([^k]v_{ai}\) is clamped to the maximum velocity \( v_{\max} \).

d. According to objective (1), (2) and (3), compute \( PD, PR \) and \( RV \).

**Step 2** Store non-dominated particles of all swarms in the elitist archive. Put the first particle of the first swarm into the elitist archive, and then compare each particle (is denoted by \( particle1 \)) generated subsequently with all particles in the elitist archive. The detailed rules of the comparison are described as follows:

a. If some members of the archive are dominated by \( particle1 \), all dominated solutions are deleted from the elitist archive.

b. If none of members in the archive dominates \( particle1 \), then the solution is added to the archive.

c. If at least one member of the archive dominates \( particle1 \), then the archive does not need to be updated and the iteration continues.

**Step 3** Initialize the global best position of each swarm; and the personal best position of each particle.

**Step 4** Update the position and velocity of each particle according to the evolution equations. In the multi-mode resource leveling problem, since the decision variable and start time are integer, it is necessary to modify equation (9) to make sure that both velocity and position are integer. Evolution equations can be modified as equation (12) and (13):

\[
[^k]v_{ai}(iter + 1) = \text{int}([^k] \omega \times[^k]v_{ai}(iter)) + \text{int}([^k]c_1 \times r_1 \times ([^k]pbest_{ai} -[^k]x_{ai}(iter))) + \text{int}([^k]c_2 \times r_2 \times ([^k]gbest_i -[^k]x_{ai}(iter)))
\]

\[
[^k]x_{ai}(iter + 1) = [^k]x_{ai}(iter) +[^k]v_{ai}(iter + 1)
\]

where \( \text{int}() \) represents integral function. Finally, the algorithm inspects whether the updated particle position satisfies constraints (6), (7) and (8) through the dynamic detecting method described in Step 1. If not, EVEPSO repeatedly updates the particle velocity and particle position according to equation (12) and (13).

**Step 5** Update the external archive to include non-dominated solutions from all updated swarms.

**Step 6** Update the global best position of each swarm and personal best position of each particle. If the value of the \( k \)th objective function for the current position \([^k]x_a\) is less than the value of the \( k \)th objective function for the personal best position \([^k]pbest_a\), EVEPSO assigns \([^k]x_a\) as the personal best position \([^k]pbest_a\).

According to each non-dominated solution in external archive, EVEPSO assigns the position of non-dominated solution, which the value of \( PD \) is minimum, to \([^3]gbest\); and the position of non-dominated solution, which the value of \( PR \) is minimum, to \([^1]gbest\); and the position of non-dominated solution, which the value of \( RV \) is minimum, to \([^2]gbest\).

**Step 7** Repeat Step 4 - Step 6 until the maximum iteration \( iter_{\max} \) is reached.
5 Experiments and Computational Results

5.1 Example problem

In this paper, we use the practical problem of Fig.1 proposed by Demeulemeester et al. [7].

![Example network](image)

Fig. 1: Example network

In this problem, there are 9 notes and 14 activities, and each activity has two execution modes. According to $m = 2, X_{i2} = 1 - X_{i1}$. To shorten the coding, we only put $X_{i1}$ into the coding. If $X_{i1} = 0$, it indicates that the execution mode of activity $T_i$ is $(D_{i2}, R_{i2})$; if $X_{i1} = 1$, it indicates that the execution mode of activity $T_i$ is $(D_{i1}, R_{i1})$.

Key parameters are given as follows: The population of particle in a swarm $M = 20$, the maximum iteration $\text{iter}_{\text{max}} = 10000$, the initial weight factor $\omega_{\text{max}} = 1.0$ and the final weight factor $\omega_{\text{min}} = 0.2$.

5.2 Discussion of the results

The result indicates that the optimal targets (minimum project duration $PD_{\text{min}}$, minimum resource requirement $PR_{\text{min}}$ and minimum resource variance $RV_{\text{min}}$) generated by EVEPSO after 3280th iteration equal to the optimal targets generated by EVEPSO after 10000th iteration. Then we suppose that EVEPSO is converging after 3280th iteration and generated 31 Pareto-optimal solutions.

For three optimal targets (i.e. $PD_{\text{min}} = 11, PR_{\text{min}} = 34$ and $RV_{\text{min}} = 0.025$), minimum project duration $PD_{\text{min}}$ and minimum resource requirement $PR_{\text{min}}$ are reached global optimum solutions, while minimum resource variance $RV_{\text{min}}$ are optimized extremely through EVEPSO. However, VEPSO is converging after 7488th generation, and it obtains 30 Pareto-optimal solutions, among which 18 solutions are sub-optimal solutions in contrast to Pareto optimal solutions obtained by EVEPSO.

The experimental results show that EVEPSO runs nearly 2.28 times as fast as VEPSO; and the minimum resource variance obtained by EVEPSO is reduced by 85.5% compared with the results obtained by VEPSO. It is easy to find that EVEPSO can perform well with respect to convergence and efficiency. Moreover, EVEPSO is able to produce a set of good trade-off solutions.
from which decision makers may select one satisfying with the deadline and resource limit of the project as the optimal solution.

Furthermore, Pearson’s correlation tests of the two groups’ statistics ($PD - RV$, $PR - RV$) are carried out with SPSS V13.0 software to deal with the relationship among project minimum resource variance, project duration and total project resource requirements. Before the correlation test, two groups’ statistics need to be pretreated. For $PD - RV$, in order to obtain the minimum resource variance for project duration, we only reserve the pair statistics which $RV$ is minimum if the same $PD$ is associated with more than one $RV$. For example, the minimum resource variance $RV_{min} = 0.735$ when $PD = 11$. For $PR - RV$, we perform the same pretreatment as $PD - RV$ to obtain minimum resource variance for resource requirements. The result of Pearson’s correlation test shows that Pearson correlation between project duration and minimum resource variance is $r_1 = -0.802$, and the significance $Sig_1 = 0.005 < 0.05$. It reveals that project duration and minimum resource variance are correlated negatively. The scatter figure of project duration and minimum resource variance is shown in Fig.2(a).

But Pearson correlation between total resource requirements of the project and minimum resource variance is $r_2 = 0.419$, and the significance $Sig_2 = 0.052 > 0.05$. It shows that there are few correlations between total resource requirement of the project and minimum resource variance. The scatter figure of resource requirement and minimum resource variance is shown in Fig.2(b).

![Fig. 2: (a) the scatter figure of project duration and minimum resource variance (b) the scatter figure of resource requirement and minimum resource variance](image-url)

### 6 Conclusions

This paper presents comprehensive descriptions of the multi-mode resource leveling problem, which is a new problem. Different from the traditional resource leveling problems, the duration and resource requirements of each activity are varied according to different execution mode in the multi-mode resource leveling problem. Then we have investigated a novel EVEPSO for this problem based on a practical example. The empirical results indicate that EVEPSO is superior to VEPSO in convergence efficiency, and can accurately and efficiently solve the multi-mode resource leveling problem. Finally, we discover that the project duration and minimum resource...
variance are correlated negatively; while there are few correlations between resource requirements and minimum resource variance.

It is useful for decision makers to schedule real-life projects. It is certain that the requirements and fluctuation of the resource would be higher if the project duration needs to be shortened. Conversely, if we reduce the requirements and fluctuation of the resource, the project duration would be extended. It is crucial for us to analyze the advantages and disadvantages of project duration, resource requirement and resource variance before scheduling a project.

Acknowledgement

This work is supported by the National Nature Science Foundation of China (Grant No. 70871111); the Zhejiang Philosophy and Social Science Project Fund (Grant No. 11JCSH03YB).

References