Niching Estimation of Distribution Algorithm Based on Fuzzy Clustering for Multi-mode Resource-constrained Project Scheduling Problems

Omar S. Soliman and Elshimaa Elgendi

Abstract—: This paper proposes a novel niching estimation of distribution algorithm (EDA) based on fuzzy c-means (FCM) clustering for solving the multi-mode resource-constrained project scheduling problem (MRCPSP). FCM clustering is employed to partition the population into niches to avoid premature convergence of the EDA. Then, the niche capacity is determined by Boltzmann scheme according to the adaptive clearing radius and niche fitness. Besides, a restart up policy is employed based on a new diversity measure. A random walk local search is applied based on delete-then-insert operator (DIRW) to achieve a trade-off between exploration and exploitation abilities. The proposed algorithm is tested and evaluated using benchmark test problems of the project scheduling problem library PSPLIB, and compared with the standard EDA algorithm. Simulation results demonstrate the effectiveness of the proposed algorithm, and outperform the standard EDA.

Index Terms— Multi-mode resource-constrained project scheduling problems, niching estimation of distribution algorithm, Fuzzy c-means clustering, Boltzmann niching clearing scheme, Local search, Random walk

1. Introduction

The multi-mode resource-constrained project scheduling problem (MRCPSP) is a generalized version of the single-mode resource constrained project scheduling problem (RCPSP). In a multi-mode resource-constrained project, each activity can be performed in one out of a set of modes, with a mode specific duration and nonrenewable and renewable resource requirements. There are three different types of resources; renewable, nonrenewable and doubly constrained ones [1]. Renewable resources are limited on a per-period-basis, e.g. machines and manpower. Nonrenewable resources, the availability for the entire project is limited, e.g. money if the budget of the project is limited. Doubly constrained resources are limited both on a per-period-basis and on a total-project-basis, e.g. money if not only the budget of the project but also the per-period-cash-flow is limited. However, the doubly constrained resources need not be taken into account explicitly since they can be incorporated by properly enlarging the sets of the other two types of resources.

- *Omar S, Soliman is currently an Associate Professor, Faculty of Computers and Information, Cairo University, Egypt.* *E-mail: Dr.omar.soliman@gmail.com*
- *Elshimaa Elgendi is currently PhD* Candidate *, Faculty of Computers and Information, Cairo University, Egypt.* *E-mail: e.elgendi@fci-cu.edu.eg*

Finding a feasible solution of the MRCPSP is NP-complete if there is more than one non-renewable resource [2]. advantage in the solution of such problems. Optimization of multiple objectives requires that the relative importance of each objective be specified in advance which requires a prior knowledge of the possible solutions.

Therefore, evolutionary computation has gained increasing attention during the past decades for solving problems belonging to this class.
There have been several bio-inspired approaches to this problem among which a two-phase genetic local search algorithm proposed by Tseng and Chen [3] where the first phase aimed at searching globally for promising areas while the second phase aimed at searching in the promising areas more thoroughly, an efficient hybrid GA to solve the MRCPSP, in which a unique mode assignment procedure was developed and a new fitness function proposed by Lova et al. [4] and a bi-population GA to solve preemptive and non-preemptive MRCPSP that made use of two separate populations and extended the serial schedule generation scheme by introducing a mode improvement procedure proposed by Van Peteghem and Vanhoucke [5].
In addition, Van Peteghem and Vanhoucke [6] proposed an artificial immune system (AIS) to solve the MRCPSP with mode assignment list generation was translated to a multi-choice multi-dimensional knapsack problem. Wauters et al. [7] proposed a multi-agent learning approach, where an agent was placed each activity node to decide the order to visit its successors and the mode to execute the activity. Damak et al. [8] proposed a differential evolution (DE) approach to solve the MRCPSP. Elloumi and Fortemps [9] proposed a hybrid

rank-based evolutionary algorithm that transformed the MRCPSP to a bi-objective problem to cope with the potential violation of the nonrenewable resource constraints. Ranjbar et al. [10] proposed a hybrid scatter search to solve the discrete time/ resource trade-off problem and the MRCPSP. Wang and Fang [11] showed the effectiveness of using shuffled frog-leaping algorithm (SFLA) for solving the MRCPSP with the criterion to minimize the makespan. But SFLA needs a very careful choice of its parameter in order to be effective. Recently, Wang and Fang [12] designed an estimation of distribution algorithm (EDA) for solving the MRCPSP by using an encoding scheme based on the activity-mode list and a decoding scheme based on the multi-mode serial SGS.

Estimation of distribution algorithm (EDA) is a kind of stochastic population-based optimization algorithm based on statistical learning Larrañaga and Lozano [13]. But in practice it is found that, EDA still suffer from a drawback common to other evolutionary algorithms. The fast losing of diversity in the population can lead to an early convergence of the algorithm. to cope with this drawback a new hybrid fuzzy clustering based niching EDA is proposed for solving nonpreemptive MRCPSP minimizing the project makespan based on adopted fourfold: a) a restart up policy based on a proposed diversity measure is applied in the early stages of the algorithm to enhance the exploration ability of the algorithm, b) a niche clearing strategy based on Boltzmann mechanism is applied after clustering the entire population considering each cluster as a niche, c) a new local search heuristic (DIRW) based on delete-then-insert operator and random walk is applied to enhance the exploitation ability of the algorithm and d) a list of best so far found solutions is maintained to retain the convergence of the algorithm.

The remainder of this paper is organized as follows; Section 2 is devoted to the description of the MRCPSP. Section 3 deals with the proposed fuzzy clustering based niching EDA steps for MRCPSP and its features. In Sections 4 and 5, experimental results and conclusions and further work directions are given respectively.

2. Multi-mode Resource-Constrained Project Scheduling Problem

In MRCPSP, the target is to study how to allocate renewable/nonrenewable resource and schedule activities to minimize the whole project makespan. In MRCPSP, a project consists in J activities, with a dummy start node 0 and a dummy end node $J+1$. The precedence relations between activities are defined by a directed acyclic graph G. No activity may be started before all its predecessors are finished. Graph G is numerically numbered, i.e. an activity has always a higher number than all its predecessors. A is the set of all activities pairs (A_i, A_j) such that A_i directly precedes A_j. Each activity j, $j = 1, \cdots, J$ has to be executed in one of M_j modes. Preemption is not allowed. Once a mode chosen for an activity, it may not be changed. The duration of activity j executed in mode m is d_{jm}. We assume that there are R renewable and N non-renewable resources. The number of available units of renewable resource k, $k = 1, \cdots R$ is R_k^ρ and the number of available units of non-renewable resource l, $l = 1, \cdots, N$, is R_l^v. Each activity j executed in mode m requires for its processing r_{jmk}^ρ units of renewable resource k, $= 1, \cdots R$, and consumes r_{jml}^v units of non-renewable resource l, $l = 1, \cdots, N$. We assume that all activities and resources are available at the beginning of the process. The objective of the MRCPSP is to find an assignment of modes to activities as well as precedence and resource-feasible starting times for all activities, such that the makespan of the project is minimized. The mathematical formulation of the MRCPSP found in Talbot [14].

3. Hybrid Fuzzy Clustering-Based Niching EDA for MRCPSP

3.1 Proposed algorithm mechanism

In this section, the mechanism of the proposed algorithm hybrid fuzzy clustering-based niching EDA to solve the MRCPSP is described. The proposed algorithm uses activity-mode list (AML) decoding scheme described in section 3.2 below. After applying a preprocessing reduction procedure developed by Sprecher et al. [1] in project data to efficiently reduce the search space, the probability matrix is initialized using uniform distribution. Then the proposed algorithm starts generating an initial population of P individuals by sampling the probability matrix using permutation-based probability generating mechanism (PGM) developed by Wang and Fang [12]. As the MRCPSP is NP-complete, a set of infeasible solutions are introduced into the search space of MRCPSP solutions, these solutions only violate the nonrenewable resource constraints. A penalty function ($v_E(ML)$) to be minimized enforces the satisfaction of the nonrenewable resource constraints. All individuals are decoded by the multi-mode serial schedule generation scheme (MSSGS). A rank-based fitness function for the generated individuals is computed by considering both makespan and penalty as two criteria to be minimized. The population is clustered using fuzzy c-means clustering (FCM) heuristic; each cluster (niche) is evaluated by its best solution found. According to the proposed niching clearing strategy of Boltzmann scheme, *best_P* individuals are selected from niches obtained. In the early stages of the proposed algorithm, diversity is promoted in the population set by computing a diversity measure. And if its value is less than predetermined threshold, a restart up policy is applied. In the restart up policy, the list of best individuals over the previous generations (*list_best)* are kept to maintain the convergence of the algorithm and the remaining individuals (i.e., $P - list\_best$) are generated randomly using the initial probability matrices, $P_{act}(0)$ and $P_{mod}(0)$. Then the Multi-mode double-justification (MDJ) is applied to only to the selected *best_P* individuals to improve the makespan value. After that, a newly developed random walk based local search method (DIRW) is applied to the best individuals to exploit the neighborhood of the selected individuals [15]. Then, the probability matrixes $P_{act}(t)$ and $P_{mod}(t)$ are updated based on the selected *best_P* individuals, for generating offspring population. This procedure is repeated until the stopping criterion is reached. Straight forwardly, the pseudo code of the proposed algorithm is described in Algorithm 1.

ISSN 2229-5518

3.2 Solution representation

In the proposed algorithm, The encoding scheme chosen for individuals is an activity-mode list (AML), which consists of two vectors: (1) a precedence feasible activity list (AL) $\{A_{\pi_1},...,A_{\pi_i},...,A_{\pi_J}\}$; (2) a mode assignment list (ML) $\{m_{\pi_1},...,m_{\pi_i},...,m_{\pi_J}\}$ to assign every activity a mode. The m_{π_i} in the ML indicates the mode of the A_{π_i} in the AL. The EDA does not operate on a schedule but on the AML representation of a schedule.

Algorithm **1:** Proposed Niching EDA algorithm based FCM for solving MRCPSP

1. Set t = 0;
2. Initialize the probability matrixes $P_{act}(0)$ and $P_{mod}(0)$;
3. Generate the new population with P individuals by sampling the probability matrix using a permutation-based probability generating mechanism (PGM);
4. While stopping criterion is not met do
   - 4.1. Compute the *Makespan(AML)* and penalty $v_E(ML)$ for each individual using MSSGS;
   - 4.2. Compute current population diversity measure *div*;
   - 4.3. If($div \geq \delta$)
     - i. Compute the rank-based fitness assignment for each individual on basis of two criteria makespan and penalty;
     - ii. Apply FCM clustering by the proposed similarity metric;
     - iii. Select *best_P* individuals based on the proposed niching clearing strategy of Boltzmann scheme;

     Elseif(t ≤ third of maximum number of iteration)
     - i. Reinitialize the probability matrixes $P_{act}(t)$ and $P_{mod}(t)$;
     - ii. Generate $P - list\_best$ individuals by sampling the probability matrix using PGM;
     - iii. The new population will be the generated $(P - list_{best}) \cup list\_best$ individuals of the previous generations;
     - iv. t=t+1;Go to step 4;

     Else
     - i. Select best_P individuals based on ranking selection;
   - 4.4. Update the *list_best* individuals;
   - 4.5. Apply MDJ to the selected best_P individuals;
   - 4.6. Apply Local search (DIRW) to the selected *best_P* individuals;
   - 4.7. Update the probability matrixes $P_{act}(t)$ and $P_{mod}(t)$;
   - 4.8. t=t+1; Go to step 3;
5. Return the best found solution.

3.3 Multi-mode serial schedule generation scheme

After population individuals are generated, the serial schedule generation scheme (SGS) is used to evaluate the solution vector, i.e., the solution vector of the MRCPSP is translated into a schedule [9]. The multi-mode serial schedule generation scheme (MSSGS) for the MRCPSP adopted by Wang and Fang [12] is employed to calculate two objective functions, the makespan (Makespan(AML)) and the penalty function ($v_E(ML)$) for each generated schedule. A penalty measure is the aggregation measure $v_E(ML)$ considers only the positive value of nonrenewable resource excess over the availability. That is, $v_E(ML)$ is given by (1):

$$v_E(ML) = \sum_{l=1}^{N} \max\left\{0, \frac{\sum_{j=1}^{J} r^v_{jm_j l} - R^v_l}{R^v_l}\right\} \quad (1)$$

It is noticeable that the penalty $v_E(ML) = 0$ if the AML is nonrenewable resources feasible.

For infeasible solution, its penalty value $v_E(ML)$ is calculated as in equation (1) and its makespan value as a sum of all activities durations associated with the latest mode list as in (2).

$$\text{Makespan(AML)} = \sum_{j=1}^{J} d_{A_{\pi_j} m_{\pi_j}} \quad (2)$$

For feasible solution, surly the penalty $v_E(ML)$ equals to 0 and the makespan equals to the finish time of the end activity of the entire project as in (3).

$$Makespan(AML) = FT_{\pi_J} \quad (3)$$

Where FT_{π_J} is the scheduled finish time of the end activity in the produced schedule.

3.4 Fitness function

To compute the fitness value, the rank-based fitness assignment method for multi-objective genetic algorithms (MOGAs) conceived by Fonseca and Fleming [16] is adopted. Each individual is assigned a rank value depending on its position within the population, and considering the criteria to be optimized. At a given population t, an individual i is dominated by Nt individuals in the considered population. Its rank R will be Ri,t=1+Nt and its fitness value would be computed by the following formula:

$$f(AML_i) = \frac{1}{R_{i,t}}$$

Note that non-dominated individual would have a rank equal to one. Therefore, the fitness value will be, in its turn, equal to one. Other individuals would have greater rank and would then be assigned a fitness value less than one. Also, for a given individual, this metric may varies through populations because of the population distribution changes. Also, small rank of a given individual means small number of solutions dominating that individual.

3.5 Population diversity measure

Invariably, the population diversity diminishes as the EDA evolves over generations. This results in search stagnation. To overcome this problem and also to reduce the computational effort to be used in clustering a homogenous population, a new diversity measure of the current population is developed. This measure of population diversity is based on averaging the Euclidean distance between individuals from their centroid. The diversity measure from the individuals of the current population is computed in the following way:

Step1: compute the centroid $v=(v_1,v_2)$ of the current population where v_1 and v_2 are the average values of the penalty and makespan over the whole population respectively.

Step2: calculate the diversity measure *div* is calculated as follows:

$$div=\frac{1}{2P}\left(\sum_{k=1}^{P}\left(\frac{|v_E(ML_k)-v_1|}{v_1}\right)+\sum_{k=1}^{P}\left(\frac{|Makespan(AML_k)-v_2|}{v_2}\right)\right)$$

Hence, *div* gives a diversity measure with a value between zero and one. Clearly, a large *div* value indicates that the population is highly diverse. A value close to one indicates a very diverse population where each activity is set at different positions and no similar blocks of activities exist among the individuals. On the other hand, a value close to zero means that the population is homogenous, i.e., all individuals are very similar or identical

3.6 Fuzzy c-means clustering

In order to avoid premature convergence, the proposed algorithm is enhanced by the use of an adaptive grid, relying on fuzzy c-means clustering. FCM is a kind of soft clustering method and primarily cluster data set which allows one piece of data to belong to two or more clusters. FCM works by partitioning data sets into *c* fuzzy groups by using membership degrees of cases which are computed for clusters and finds a cluster center in each group such that the cost function of dissimilarity measure is minimized [17]. This paper briefly introduces fuzzy clustering methodology and the related membership function construction. In this context, *c* can be chosen as the worst rank value (the largest).

Each data point $X_i=(Makespan(AML_i),v_E(ML_i))$ has a degree of membership $u_j(X_i)$ in the cluster number *j*, given by:

$$u_{ij}=\frac{(\Delta_{ij})^{-\frac{1}{\tau-1}}}{\sum_{k=1}^{c}(\Delta_{ij})^{-\frac{1}{\tau-1}}} \qquad (4)$$

where Δ_{ij} represents the distance between the data point X_i and c_j, c_j is the cluster *j* centroid and τ corresponds to the fuzzyfication parameter which determine the degree of fuzziness (weighting exponent) in the clusters, it would be set to 2 in this paper.

The j-th cluster centroid, c_j, is determined by computing the mean of all the points in that cluster weighted by their degree of belonging to or membership of the data points in that cluster;

$$c_j=\frac{\sum_{i=1}^{P}(u_{ij})^{\tau}X_i}{\sum_{i=1}^{P}(u_{ij})^{\tau}} \qquad (5)$$

Given a value of the parameter τ=2 and the number of clusters *c*; The FCM clustering algorithm works as follows:

Step1: Assign randomly to each point membership degrees for the *c* clusters so that the membership degrees for each data point in all the *c* clusters satisfy $\sum_{j=1}^{c}u_{ij}=1$, $i=1,2,\ldots,P$

Step2: Compute the centroid of each cluster using (5).

Step3: Compute the membership degree for each data point in each cluster using (4).

Step4: If a convergence criterion is not met, go to 2).

The performance index of a FCM is defined by:

$$idx=\sum_{j=1}^{c}\sum_{i=1}^{P}(u_{ij})^{\tau}\Delta_{ij}$$

The clustering goal is to find an iterative fuzzy partitioning that minimizes the performance index idx.

3.7 Selection of the niche capacity

To implement the selection procedure of the hybrid fuzzy clustering based niching EDA, after the individuals are grouped into many clusters (niches) by FCM when diversity of the population is greater than δ. Then, the capacity w_i is defined as the maximum number of winners that will be selected from cluster *i* [18]. w_i is calculated as follows:

$$w_i=best\_P\left[\gamma_t\frac{f_i}{\sum_{j=1}^{c_t}f_j}+(1-\gamma_t)/n_i\right],\ \gamma_t=\left[\frac{e^{\left(\frac{t}{T}\right)}-1}{e-1}\right]^2$$

where *best_P* is the number of selected individual in each iteration, *t* is the number of generated schedules, γ_t is the Boltzmann weight after generating *t* schedules, *T* the maximum number of generated schedules, n_i is number of individuals of cluster *i* in the current iteration and f_i is the maximum fitness value of *i*-th niche in the current population. The Boltzmann scheme can provide a good balance between the diversity preservation and algorithm efficiency. This selection procedure guarantees to select the dominant individuals from each cluster so the convergence of the algorithm is maintained. Also, selection of some dominated individuals enhances the exploration ability of the algorithm.

3.8 Local search

After the *best_P* individuals are selected from the population, multi-mode double-justification (MDJ) is applied to the *best_P* individuals to improve their makespan values. MDJ iteratively employs the MSSGS forward and backward scheduling until no further improvement in the makespan of project can be found. More details of MDJ could be found in Wang and Fang [12].

To enhance the exploitation ability, a new local search heuristic (DIRW) based on the delete-then insert operator and random walk developed by Soliman and Elgendi [15] is applied. Delete-then-insert operator is applied on each activity on the AL which deletes the activity from its current position and then inserts it in a random eligible position. Accepting this move is based on acceptance rate *RW*. DIRW enhance the exploitation ability by exploring the neighborhood of the individual Algorithm 2.

Algorithm 2: The delete-then-insert random walk Local search (DIRW).

```
Procedure DIRW Local Search (AML)
Begin
   AL' = AL;
   ML' = ML;
  //activity at first position is the start activity which has no
  predecessor
   j=2;
  while(j<J-1) do
   {
     Generate uniformly distributed random number q,
     q ∈ [0,1]
     if(q < RW)
       Pos = position of A_j in AL'
        if (v_E(ML') = 0)
          ML'_pos = min_m(d_jm);
       else
          ML'_pos = min_m( Σ_{l=1}^{N} r^v_jml );
       endif
     Delete A_j from position pos in AL';
     Maxpos=max(π_k|(AL'_πk, A_j) ∈ A)+1;
     Minpos= min(π_k|(A_j, AL'_πk) ∈ A)-1;
     Insert A_j in a random location between Maxpos and
     Minpos on AL';
     AL = AL';
     ML = ML';
     endif
     j=j+1
   }
  endwhile
End
```

In the procedure of DIRW, the nonrenewable resource consumption for infeasible representation may be reduced by assigning the mode with minimum cumulative nonrenewable resources consumption to the currently modified activity, and enhance the nonrenewable resource utilization for feasible representations by assigning the mode with minimum duration to the currently modified activity.

4. Computational Results

4.1. Test problems

In this section, the results of a computational experiment concerning the implementations of the hybrid fuzzy clustering based niching EDA for solving MRCPSP is presented. The experiments have been performed on an Intel(R) Celeron(R)-based DELL with 2.19 GHz clock-pulse and 1 Gb RAM. The proposed algorithm has been coded and compiled in MATLAB R2007b. the program has run on standard MRCPSP test instances available in the project scheduling problem library PSPLIB developed by Kolisch and Sprecher [19]. Optimal solutions for J10-20 instances sets are available, whereas no optimal solutions were found for J30 instances; hence, heuristic lower bounds are provided for the latter. In Table 1, the number of instances for which at least one feasible solution exists is presented. Each instance of J10-20 and J30 contains three modes (i.e., $M_j = 3, j = 1,2,..., J$; $M_0 = 1$; $M_{J+1} = 1$), two renewable and two nonrenewable resources. The duration of activity j in mode m_j varies from 1 to 10.

Table1: Number of instances

J	10	12	14	16	18	20	30
Number of Instances	536	547	551	550	552	554	552

4.2. Configuration of the algorithm

In this section, the numerical configuration of the proposed algorithm is reported through a numerical investigation.
Following Hartmann [2], the performances of the algorithm have been investigated on the set J20 since it is the hardest standard set for MRCPSP and 5000 generated schedules as a stopping condition. The proposed hybrid fuzzy clustering based niching EDA contains five key parameters: the population size of each generation (P), the number of selected individuals to update the probability matrix (best_P), the learning speed (α), diversity measure threshold (δ) and local search acceptance rate (*RW*).

The average relative deviation (ARD) value is the following average deviation value for N instances.

$$ARD = \frac{1}{N}\sum_{i=1}^{N}\frac{(Makespan_i - Optimum_i)}{Optimum_i}$$

where $Makespan_i$ is the makespan of the ith feasible instance obtained by the proposed algorithm; $Optimum_i$ is the optimal makespan of ith feasible instance.

Extensive testing revealed that the best configuration results consist in the following issues: the fuzzy clustering improvement procedure deserves to be applied; the best algorithmic parameters are a P = 100, best_P = 10, α = 0.1. The best δ and *RW* values are 0.5 and 0.7, respectively, at which a balance between exploitation and exploration abilities occurred.

In fact, a high value of δ implies that the restart up policy is frequently applied, and this enhances the exploration ability of the algorithm and hence prevent from falling in local minimum and also finding a near optimum solution faster. But very high value of δ implies the decrease of convergence of the proposed algorithm.

4.3. Comparisons with standard EDA

To compare the proposed algorithm with the standard EDA algorithm developed by Wang and Fang [12], both algorithms have been coded and compiled in MATLAB R2007b. So, results displayed in Table 2 using 60 s CPU time as a stopping criterion. From Table 2, it can be seen that the proposed algorithm not only has a higher optimal and feasible rates than classical EDA but also has lower average deviation with less computation resource for all problem sets.

Table 2: Comparison with EDA [12] (60 s CPU time).

J	Av.dev (%)		Feasible rate (%)		Optimal rate (%)	
	The proposed algorithm	EDA [16]	The proposed algorithm	EDA [16]	The proposed algorithm	EDA [16]
10	**0.044**	0.067	**100**	**100**	**98.6**	98.2
12	**0.062**	0.086	**100**	**100**	**98.3**	97.8
14	**0.103**	0.247	**100**	**100**	**97.4**	94.6
16	**0.131**	0.403	**100**	**100**	**96.9**	91.5
18	**0.293**	0.548	**100**	**100**	**93.7**	88.6
20	**0.642**	0.771	**100**	**100**	**86.7**	84.1
30	**11.91**	13.90	**88.10**	80.1	n/a	n/a

Note: The bold values denote the best results among the results obtained by all the algorithms.

Next, the variants leading to the best configurations of the hybrid fuzzy clustering based niching EDA with DIRW local search on the different set instances with 5000 computed schedules stopping criteria with other algorithms based is compared on the PSPLIB to further show the effectiveness of the proposed work. The compared algorithm includes the classical EDA developed by Wang and Fang [12]. The Pseudo-code and the parameters of these algorithms can be found from the reference. Note that all the compared algorithms adopt activity list and mode list as their encoding type. The results are displayed in table 3 below.

As reported in table 3, the hybrid fuzzy clustering based niching EDA outperforms the EDA [12] over all data sets. EDA [12] needs effort to find and track the most promising area at the beginning of the search procedure. The proposed algorithm shows more exploration ability due to applying the restart up policy and FCM embedded to the algorithm.

Table 3: Comparison with EDA [12] (5000 computed schedules)

Av.dev (%)	J10	J12	J14	J16	J18	J20
Proposed algorithm	**0.08**	**0.14**	**0.22**	**0.40**	**0.72**	**1.08**
EDA [12]	0.12	**0.14**	0.43	0.59	0.90	1.28

Note: The bold values denote the best results among the results obtained by all the algorithms.

According to the above experimental results and comparison between the proposed algorithm and the slandered EDA algorithms, as reported in tables 2&3 ; the proposed algorithm is effective in solving the MRCPSP.

5. Conclusion

This paper introduced a hybrid niching EDA based on fuzzy clustering for solving the multi-mode resource-constrained project scheduling problem. A critical problem when dealing with EDA is the premature convergence. To avoid premature convergence, FCM clustering is used to divide the population into clusters (niches). Then, the set of best solutions is selected from each cluster based on a cluster niche clearing scheme based on Boltzmann mechanism. By adopting a restart up policy and fuzzy c-means clustering method, the exploration ability is enhanced in a way it could be tracked effectively to keep finding optimum or near-optimum solution. By applying the DIRW local search and multi-mode double justification improvement; the exploitation could be enhanced as well. Simulation results based on the PSPLIB benchmarks and comparisons with some existing algorithms demonstrated the effectiveness of the proposed algorithm. The further work is to apply the proposed algorithm to other scheduling problems since it looks like promising.

References

[1] A Sprecher, S Hartmann and A Drexl, “An exact algorithm for project scheduling with multiple modes”, OR Spektrum, 19(3), pp. 195–203, 1997.

[2] R Kolisch and A Drexl, “Local search for nonpreemptive multi-mode resource-constrained project scheduling,” *Iie Trans, 29*, pp. 987–999, 1997.

[3] L Y Tseng and S C Chen, “Two-phase genetic local search algorithm for the multimode resource-constrained project scheduling problem,” Trans Evol Comp., 13, pp. 848 –857, 2009.

[4] A Lova, P Tormos, M Cervantes and F Barber, “An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes,” *Int J Pprod Econ., 117(2)*, pp. 302–316, 2009.

[5] V Van Peteghem and M Vanhoucke, “A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem,” *Eur J Oper Res., 201(2)*, pp. 409 – 418, 2010.

[6] V Van Peteghem and M Vanhoucke, “An Artificial Immune System for the Multi-Mode Resource-Constrained Project Scheduling Problem,” *vol. 5482 of Lect Notes Comput Sc. Springer Berlin /Heidelberg*, pp. 85–96, 2009.

[7] T Wauters, K Verbeeck, G Vanden Berghe and P De Causmaecker, “A multi-agent learning approach for the multi-mode resource-constrained project scheduling problem,” Decker, ; Sichman, ; Sierra, ; Castelfranchi, (eds.), Autonomous Agents and Multiagent Systems, Hungary, 10-15 May 2009, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems* (AAMAS 2009), pp. 1-8, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org)

[8] N Damak, B Jarboui, P Siarry and T Loukil, “Differential evolution for solving multi-mode resource-constrained project scheduling problems,” *Comput Oper Res, 36(9)*, pp. 2653 – 2659, 2009.

[9] S Elloumi and P Fortemps, “A hybrid rank-based evolutionary algorithm applied to multi-mode resource-constrained project scheduling problem,” *Eur J Oper Res., 205(1)*, pp. 31 – 41, 2010.

[10] M Ranjbar, B De Reyck and F Kianfar, “A hybrid scatter search for the discrete time/resource trade-off problem in project scheduling,” *Eur J Oper Res., 193(1)*, pp. 35–48,2009.

[11] L Wang and C Fang, “An effective shuffled frog-leaping algorithm for multi-mode resource-constrained project

scheduling problem," *Inform Sciences*, *181(20)*, pp. 4804 – 4822, 2011,*Special Issue on Interpretable Fuzzy Systems.*

[12] L Wang and C Fang, "An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem," *Comput Oper Res.*, *39(2)*, pp. 449 – 460, 2012.

[13] P. Larrañaga and J.A. Lozano, "Estimation Of Distribution Algorithms: A New Tool For Evolutionary Computation. Genetic Algorithms And Evolutionary Computation," Kluwer Academic Publishers, 2002.

[14] FB Talbot, "Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case," *Manage Sci.*, *28(10)*, pp. 1197–1210, 1982.

[15] OS Soliman and E. Elgendi, "A hybrid estimation of Distribution Algorithm with Random Walk local Search for Multi-mode Resource-Constrained Project Scheduling problems," International Journal of Computer Trends and Technology (IJCTT). 8(2), pp. 57-64, 2014.

[16] CM Fonseca and PJ Fleming, "Genetic algorithm for multiobjective optimization, formulation, discussion and generalization," In Forrest, S., ed., Genetic Algorithms: Proceeding of the Fifth International Conference. CA. pp. 416-423, 1993.

[17] JC Bezdek, R. Ehrlich and W. Full, "FCM: The fuzzy C-means clustering algorithm," Computers and Geosciences, pp. 10: 91–203, 1984.

[18] EL Yu and PN Suganthan, "Ensemble of niching algorithms," Information Sciences, 180(15), pp. 2815–2833, 2010.

[19] R Kolisch, A Sprecher. "PSPLIB-a project scheduling problem library," *Eur J Oper Res*, *96(1)*, pp. 205–16, 1997.