An Efficient Meta-heuristic Algorithm for Project Scheduling With Multiple Modes

M.H. Sebt[1*] M.R. Afshar[2] Y. Alipouri[3]

1-Introduction

The aims of MRCPSP are finding an execution mode and a feasible start time for each activity, such that makespan of the project is minimized under resource and precedence constraints.

There are three methods for solving the MRCPSP: the exact methods; heuristic approach and meta-heuristic approaches [3, 4]. However, the meta-heuristic approaches are the best existing methods for solving the MRCPSP [6, 7].

So far, many heuristic approaches were applied for solving the MRCPSP [7-12]. One of the new optimization techniques is the simulation of the swarm behavior of natural creatures and Particle Swarm Optimization (PSO) is one of them.

In this paper, we propose a Fully Informed Particle Swarm (FIPS) algorithm that is one of the best variants of the PSO, for solving the non-preemptive MRCPSP. In particular, a new fitness function is suggested to reduce the average deviation and CPU time.

2- The FIPS algorithm for solving the MRCPSP

2.1- Representation scheme

In the FIPS, a random key and the related mode list (ML) representation scheme are used as encoding schemes. Accordingly, the encoding procedure in the FIPS is as follows:

Two independent positions are considered for each particle in two different n-dimensional search spaces. The first space is employed for finding the optimal priority combination and the second one is employed for finding the optimal mode combination with one goal, which is the minimization of project makespan. Both of them must be updated simultaneously, based on formulas (1) and (2) from iteration to iteration.

$$V^{t+1}_i = V^t_i + \sum_{pm \in Ni} \varphi_k U_k^t (pb_k^t - X_i^t) \qquad (1)$$

$$X^{t+1}_i = X^t_i + V_i^{t+1} \qquad (2)$$

2.2. Generation of initial conditions (initial positions and velocities)

In this paper, the initial positions of each particle i corresponding to RK and ML are randomly generated by the ranges [0, 1] and [1, Mi] (Mi is the number of modes to be considered for activity i), respectively.

2.3. Fitness function

Determination of an appropriate fitness function is necessary for the correct operation of evolutionary algorithms.

In the MRCPSP literature, several fitness functions have been proposed of which the fitness function of Lova et al. [7] (Eq.3) is the last one. In this last fitness function (Eq.3), infeasible solutions are penalized by ERR (μ), which presents the non-renewable infeasible degree of ML (Eq. 4). Obviously, when ERR(μ) is 0, the individual i is a feasible solution.

$$f(i)=\begin{cases} 1-\dfrac{\max\_mak(P)-mak(i)}{\max\_mak(P)}, & \text{if } ERR(\mu)=0 \\ 1+\dfrac{mak(i)-min\_CP}{mak(i)}+ERR(\mu), & \text{otherwise} \end{cases} \qquad (3)$$

$$ERR(\mu)=\sum_{l=1}^{N} \max\left\{0, \frac{\sum_{j=1}^{J} n_{jml} - N_l}{N_l}\right\} \qquad (4)$$

Where mak(i) is the makepan of the individual i and max_mak(P) gives the maximal makepan of feasible solutions related to individuals of the current generation. Min CP gives the minimal critical path of the project using the minimal duration of activities.

Lova et al. [7] demonstrated that their fitness function gives better results than the other ones and removes their faults. However, calculation of CPM term in their fitness function can increase the computational effort of the algorithm. Hence, the following fitness function (Eq. 5) is proposed for the elimination of this shortcoming.

$$f(I)=\begin{cases} 1-\dfrac{T-mak(I)}{T}, & \text{if } ERR(\mu)=0 \\ 1+\dfrac{mak(I)}{T}+ERR(\mu), & \text{otherwise} \end{cases} \qquad (5)$$

where T is the upper bound on the project's makespan given by the sum of the maximal durations of activities and mak(i) gives the makespan of the individual i.

After performing a few tests, the results showed that the proposed fitness function has the better performance.

2.4 Schedule generation scheme (decoding procedure)

Schedule generation scheme (SGS), as an efficient method, transforms a representation solution of the RCPSP to a schedule [12]. There are two types of SGS: serial SGS and parallel SGS. Since the parallel SGS cannot sometimes find the optimal solution [22], we make use of the serial SGS. In the MRCPSP, because of the existence of non-renewable resources, the serial SGS must be modified. Therefore, the

[1*] Corresponding Author, Assistant professor, Department of Civil Engineering, Amirkabir University of Technology, Tehran, Iran Email Address: sebt@aut.ac.ir.

[2] Master of Science, Department of Civil Engineering, Amirkabir University of Technology, Tehran.

[3] Ph.D. candidate, Department of Civil Engineering, Amirkabir University of Technology, Tehran.

following two processes should be performed.

2.4.1 Infeasible tackling procedure

In this research, after the generation of the initial population, the infeasible tackling procedure begins to transform some of the infeasible solutions into a feasible ones. In this procedure, an activity is selected randomly and its mode is changed to a new mode. If the resultant ERR (μ) is less than the previous ERR(μ), the previous mode is replaced with the new one.

2.5.2 Multi- mode forward-backward iteration (MM-FBI) method

Using the mode improvement procedure, MM-FBI method reduces the finishing time of the activity one by one at every decision point of the partial schedule by changing its mode without changing the modes and delaying the finishing times of any other activities [11].

3- Computational experiments

In this section, using some computational experiments, the performance of the proposed FIPS for solving the MRCPSP is investigated. The proposed FIPS was programmed with Matlab R2012a, and the tests were accomplished on a laptop with an Intel® core 2 T9300 2.5 GHz processor.

3.1 Comparison with existing algorithms

The average deviation from the optimal makespans, percentage of optimally solved instances, and the average CPU time in seconds are used for comparison. Optimal values for sets J10-20 are considered as the base for calculating the average deviations. Based on the mentioned considerations, our results are compared with the existing state-of-the-art methods for solving the MRCPSP such as the GA and AIS presented by Peteghem and Vanhoucke [9], [8] denoted respectively as VPVGA and VPVAIS, EDA and SFLA presented by Wang and Fang [12], [11] denoted respectively as LCEDA and LCSFLA, the hybrid genetic algorithm developed by Lova et al. [7] denoted as LHGA, the hybrid rank-based evolutionary algorithm proposed by Elloumi and Fortemps [10] denoted as EFEA, the hybrid scatter search developed by Ranjbar et al. [19] denoted as RSS, the genetic algorithm developed by Alcaraz et al. [20] denoted as AGA and the simulated annealing proposed by Jo´zefowska et al. [5] denoted as JSA.

In Table 4, the mentioned algorithms are sorted according to the descending performance with respect to their average deviations. As it can be observed from this Table, the FIPS is the third best for sets J10 and j14 and the fourth best for set J12. However, in sets J16-J20, our algorithm gives the best performance after the VPVGA. Thus, it can be stated that as the number of activities increases, the FIPS performance improves.

Table 4. Average deviations (%) from optimal makespans (5000 generated schedules as the stopping condition)

Algorithm	J10	J12	J14	J16	J18	J20
VPVGA	0.01	0.09	0.22	0.32	0.42	0.57
Proposed FIPS	0.06	0.17	0.30	0.38	0.48	0.66
VPVAIS	0.02	0.07	0.20	0.39	0.52	0.70
LHGA	0.06	0.17	0.32	0.44	0.63	0.87
LCEDA	0.12	0.14	0.43	0.59	0.90	1.28
LCSFLA	0.10	0.21	0.46	0.58	0.94	1.40
EFEA	0.14	0.24	0.77	0.91	1.30	1.62
RSS	0.18	0.65	0.89	0.95	1.21	1.64
AGA	0.24	0.73	1.00	1.12	1.43	1.91
JSA	1.16	1.73	2.60	4.07	5.52	6.74

Table 5 presents the comparison of CPU-times for different algorithms. As it can be seen from this Table, for set J10, our proposed FIPS together with LCSFLA have the best CPU-time and for other sets, our proposed FIPS is the best algorithm after the LHGA (except for J12 and J14). Since the tests of LHGA have been carried out on the computer with a faster processor, it is faster than the proposed FIPS algorithm. Thus, it can be concluded that the FIPS speed is acceptable.

Table 5. Comparison with other proposed algorithms considering average CPU time (5000 generated schedules as the stopping condition)

Algorithm	J10	J12	J14	J16	J18	J20
LHGA[a]	0.08	0.10	0.11	0.12	0.13	0.15
Proposed FIPS[b]	0.07	0.11	0.14	0.15	0.16	0.17
VPVGA[c]	0.12	0.13	0.14	0.15	0.16	0.17
LCSFLA[d]	0.07	0.09	0.13	0.15	0.19	0.27

[a] Pentium 3GHz. [c] Pentium 2.80 GHz.
[b] T9300 2.5 GHz. [d] T7500 2.2 GHz.

Finally, with respect to the previous experimental analyses, it can be concluded that the proposed FIPS algorithm is an effective algorithm for solving the MRCPSP.

4- Conclusion

In this paper, an efficient FIPS was proposed for solving the MRCPSP. In the proposed FIPS, two positions were considered for each particle (candidate the potential solution): the first position presented the priority of activities for scheduling and the second one presented the related mode list. The multi-mode serial schedule generation scheme (MSSGS) was also adopted as the decoding procedure. In particular, for further improvement of the project makespan, the mode improvement procedure of Lova et al. was modified by defining a new criterion. Besides, a new fitness function was defined, which was effective in reducing the average deviations and CPU times. The well-known benchmark sets J10, J12, J14, J16, J18, J20 and J30 in PSPLIB were used for testing the performance of the proposed FIPS. Performed comparisons indicated that the proposed FIPS algorithm is among the most competitive algorithms for solving the MRCPSP.