Iranian Journal of Management Studies (IJMS) *http://ijms.ut.ac.ir/*
Vol. 12, No. 1, Winter 2019 *Print ISSN: 2008-7055*
pp. 91-119 *Online ISSN: 2345-3745*
DOI: 10.22059/ijms.2018.261960.673213

An Energy-efficient Mathematical Model for the Resource-constrained Project Scheduling Problem: An Evolutionary Algorithm

Amir Hossein Hosseinian, Vahid Baradaran*
Department of Industrial Engineering, Islamic Azad University, Tehran North Branch, Tehran, Iran

(Received: July 13, 2018; Revised: November 22, 2018; Accepted: December 2, 2018)

Abstract

In this paper, we propose an energy-efficient mathematical model for the resource-constrained project scheduling problem to optimize makespan and consumption of energy, simultaneously. In the proposed model, resources are speed-scaling machines. The problem is NP-hard in the strong sense. Therefore, a multi-objective fruit fly optimization algorithm (MOFOA) is developed. The MOFOA uses the VIKOR as a multi-criteria decision making (MCDM) method to rank solutions in vision-based search procedure. The proposed algorithm is applied to small, medium and large size problems to evaluate its performance. Comprehensive numerical tests are conducted to evaluate the performance of the MOFOA in comparison to three other meta-heuristics in terms of convergence, diversity and computation time. The experimental results significantly show that the proposed algorithm can surpass other methods in terms of most of the metrics. Besides, the results of meta-heuristics are compared with the outputs of GAMS software for small size problems.

Keywords

Energy-efficient scheduling, Multi-objective optimization, Project scheduling, MCDM

* **Corresponding Author, Email:** v_baradaran@iau-tnb.ac.ir

Introduction

Energy has always been a crucial necessity for sustaining human life. The International Energy Agency (2015) announced that the global consumption of energy will increase by 37% in the next twenty years. Moreover, the production and consumption of energy lead to the emission of greenhouse gases to the atmosphere (Fang et al., 2011; Lu et al., 2017). This has led to global concerns. There are several energy-efficient strategies for production projects such as using energy-efficient machines (resources) for the execution of projects. However, a considerable amount of investment is required to use energy-efficient machines (Mori et al., 2011). Scheduling strategies can help project managers to save energy and to decrease costs. Scheduling concentrates on allocating valuable and finite resources to a set of activities. This paper addresses an energy-efficient resource-constrained project scheduling problem (EE-RCPSP) to optimize makespan and total energy consumption, simultaneously. In this problem, resources are considered as speed-scaling machines. Speed-scaling machines are able to process activities at different speed levels. The RCPSP aims at scheduling the activities of a project with respect to precedence relations and resource constraints (Zareei and Hassan-Pour, 2015). In the EE-RCPSP, the required workload of each activity depends on the activity itself and the resource on which it is executed. The RCPSP is categorized as an NP-hard optimization problem. This means that due to complexity of the problem, exact methods may fail to find optimal solutions of large-size test problems in a reasonable computation time (Blazewicz et al., 1983). Therefore, a multi-objective fruit fly optimization algorithm (MOFOA) is developed for the EE-RCPSP. The MOFOA can find appropriate schedules for large projects in a relatively short computation time. We listed the major contributions of this research as follows:

1. This research tackles the energy-efficient RCPSP, where the objectives are the minimization of makespan and total energy consumption, simultaneously.
2. An efficient multi-objective fruit fly optimization algorithm (MOFOA) with a new solution representation is proposed. The

proposed algorithm uses the VIKOR[1] as a multi-criteria decision making (MCDM) method to choose the best solution in each sub-swarm.

3. The input parameters of the MOFOA is tuned by the Taguchi method.
4. The outputs of the MOFOA is compared with the optimal solutions obtained by GAMS and three other meta-heuristics.

The rest of the paper is structured as follows. Section 2 reviews the most relevant studies in the literature. Section 3 explains the problem, and it contains the EE-RCPSP formulation. Section 4 describes the proposed algorithm in detail. The computational results obtained by implementing algorithms are reported in Section 5. Section 6 summarizes the paper and provides some conclusions.

Literature review

In this section, we review several studies on the Energy-efficient scheduling problems. There are two common energy-efficient strategies in the literature, namely the speed-scaling approach and the power-down mechanism. In the speed-scaling approach, it is possible to adjust processing speeds of resources where it is necessary. By increasing the processing speed, the consumption of energy grows, while the required processing time decreases. Hence, the appropriate selection of processing speeds for resources will result in significant energy savings. According to the power-down strategy, if a resource remains idle for a period of time, it is possible to save energy by shutting the resource down (Mouzon and Yildirim, 2008). Fang et al. (2011) presented a mathematical formulation for flow shop scheduling problem considering peak power load, energy consumption and carbon footprint. Bampis et al. (2012) used power-down mechanism to minimize total energy consumption for a speed-scaling single machine scheduling problem. Luo et al. (2013) studied a hybrid flow shop (HFS) scheduling to improve the production efficiency. The proposed algorithm considers the production efficiency and electric power cost (EPC) with time-of-use (TOU) electricity prices, simultaneously. Dai et al. (2013) modeled an energy-efficient bi-objective flexible flow shop scheduling problem

1. VIseKriterijumska Optimizacija I Kompromisno Resenje (VIKOR)

(FFSP). They considered power-down mechanism in their proposed model to optimize makespan and energy consumption. Shrouf et al. (2014) studied a scheduling problem, where a single machine is available for production. Liu et al. (2014) proposed a bi-objective job shop scheduling problem to minimize total electricity consumption and total weighted tardiness. The Non-dominated Sorting Genetic Algorithm II (NSGA-II) was hired to find the Pareto front. Merkert et al. (2015) clarified the differences between energy efficiency and demand-side management. Fang et al. (2016) studied speed-scaling single machine scheduling problem with time-of-use electricity prices. A bi-objective formulation was proposed by Ding et al. (2016) for the speed-scaling permutation flow shop scheduling problem. Gahm et al. (2016) developed a research framework for energy-efficient scheduling (EES) to improve energy efficiency. Mansouri et al. (2016) introduced a mixed-integer linear multi-objective model for shop floor scheduling with sequence-dependent setup times. Tang et al. (2016) addressed a dynamic flexible flow shop scheduling problem. Wang et al. (2016) studied a bi-objective machine batch scheduling problem with non-identical job sizes, the time-of-use (TOU) electricity prices and different consumption rates of machines. Yan et al. (2016) proposed a multi-level optimization approach for energy-efficient flexible flow shop scheduling problem. Zhang and Chiong (2016) presented a bi-objective mathematical formulation for the job shop scheduling problem based on the machine speed scaling framework. Che et al. (2017) proposed a mixed-integer linear programming (MILP) model for energy-conscious unrelated parallel machine scheduling problem considering time-of-use electricity pricing scheme. In this problem, the electricity price varies during a day. Liu et al. (2017) addressed the fuzzy flow shop scheduling problem, where the setup times of machines depend on their prior states. Lu et al. (2017) proposed a mathematical model for the flow shop scheduling problem. Their proposed model is energy-efficient and includes controllable transportation times. Mokhtari and Hasani (2017) modeled an energy-efficient multi-objective flexible job shop scheduling problem to optimize the makespan, the availability of the system, and the total energy costs of production and maintenance. Che et al. (2017) developed a mixed-integer programming model for the single machine scheduling problem

with power-down mechanism. In another study, a dynamic scheduling approach was proposed by Zhai et al. (2017) to minimize the electricity cost of a flow shop with a grid-integrated wind turbine. They utilized time series models to obtain updated wind speed and electricity prices. Based on the studies reviewed in this section, none of the previous research projects have proposed an energy-efficient mathematical formulation for the RCPSP. Therefore, we propose an energy-efficient bi-objective formulation for the resource-constrained project scheduling problem. The objectives of the proposed model are the minimization of both the completion time and the total energy consumption of the project, simultaneously. Besides, the fruit fly optimization algorithm has not been used in previous studies to solve energy-efficient models. Thus, a multi-objective fruit fly optimization algorithm is developed to solve the proposed model.

Problem definition

In this research, an energy-efficient resource-constrained project scheduling problem (EE-RCPSP) is studied. In this problem, there is a set of interrelated activities to be processed by a set of unrelated renewable resources. We present the project as a graph *G(J, A)*, where *J* is the set of nodes, representing activities, and *A* represents a set of arcs, showing precedence constraints with no time-lags. For each activity, there is a set of immediate predecessors. The activities are numbered as $j= 0, 1, 2, \ldots, N+1$. The activities 0 and $N+1$ are dummy start and finish activities, respectively. These activities have zero durations and they have no consumption of resources. Activities require a certain amount of resources in each period and they have only one execution mode. Moreover, they are non-preemptive. A limited amount of resource is available in each period. Allocation and transfer times of resources are negligible. Each resource can only process one activity in each period. Resources are able to process activities at different speed levels. There are *L* speed levels for each resource. Suppose that v_l ($1\leq l \leq L$) denotes the functioning speed of a resource at l^{th} speed level. All resources have the same operating speed at level *L*. In the EE-RCPSP, we assume that $v_1 < v_2 < \ldots < v_l$. The workload of activity *j* ($1\leq j \leq N$) on resource *k* ($1\leq k \leq K$) *k*, denoted as w_{jk}, depends on the activity itself and the resource on which it is

executed. Let d_j stand for the actual processing time required by activity j. d_j can be computed by $d_j = max_k \left(\frac{w_{jk}}{v_l}\right)$. Suppose that g_{jkl} represents the energy consumption rate for the execution of activity j by resource k at speed level l. The actual energy consumption required by resource k to execute activity j at speed level l is denoted as e_{jkl}, which is calculated by $e_{jkl} = g_{jkl} \times \left(\frac{w_{jk}}{v_l}\right)$.

If $v_{l1} > v_{l2}$ ($l1, l2 \in \{1, \ldots, L\}$) then $e_{jkl1} > e_{jkl2}$. Using higher processing speed will result in more energy consumption. However, less time is required to accomplish the activity. The EE-RCPSP aims at assigning resources to activities and choosing proper speed levels for activities in order to optimize the makespan and total energy consumption, simultaneously. We define the following notations to understand the proposed model.

Sets

J	Set of activities ($j, j = 0, \ldots, N+1$)
Γ	Set of renewable resources ($k = 1, \ldots, K$)
H	Set of time periods ($t, t = 0, \ldots, T$)
SL	Set of speed levels ($l = 1, \ldots, L$)
Pj	Set of predecessors of activity j

Parameters

v_l	Processing speed of a resource functioning at speed level l
w_{jk}	Workload of activity j on resource k
g_{jkl}	Energy consumption rate for performing activity j by resource k at speed level l.
r_{jk}	The required amount of resource k for activity j
R_k	The available amount of resource k

Variables

d_j	The actual processing time required by activity j
e_{jkl}	The actual energy consumption required by resource k to execute activity j at speed level l
FT_j	Finish time of activity j
C_{max}	Project completion time (Makespan)
E	Total energy consumption
X_{it}	Equals 1 if activity j is started in period t, otherwise it equals 0
U_{jkl}	Equals 1 if activity j is assigned to resource k at processing speed v_l, otherwise it equals 0

In the following lines, the problem is formulated as a bi-objective mixed-integer linear programming model:

$$Min \quad C_{\max} \tag{1}$$

$$Min \quad E \tag{2}$$

Subject to:

$$\sum_{t=0}^{T} X_{jt} = 1 \qquad \forall j \in J \tag{3}$$

$$d_j = \max_k (w_{jk} / v_l) \qquad \forall j \in J \tag{4}$$

$$FT_j \le FT_{j'} - d_{j'} \qquad \forall j, j' \in J, j \in P_{j'} \tag{5}$$

$$\sum_{j=0}^{N+1} \sum_{t=t'}^{t'+d_j-1} r_{jk}.X_{jt} \le R_k \qquad \forall k \in \Gamma \tag{6}$$

$$\sum_{l=1}^{L} U_{jkl} = 1 \qquad \forall j \in J, \forall k \in \Gamma \tag{7}$$

$$e_{jkl} = g_{jkl} \times \left(w_{jk} / v_l \right) \qquad \forall j \in J, \forall k \in \Gamma, \forall l \in SL \tag{8}$$

$$C_{\max} \ge \sum_{j=0}^{N+1} \sum_{l=1}^{L} \frac{w_{jk}}{v_l} \times U_{jkl} \qquad \forall k \in \Gamma \tag{9}$$

$$E \ge \sum_{j=0}^{N+1} \sum_{k=1}^{K} \sum_{l=1}^{L} e_{jkl} \times U_{jkl} \tag{10}$$

$$d_j, e_{jkl}, FT_j, C_{\max}, E \ge 0 \tag{11}$$

$$X_{jt}, U_{jkl} \in \{0,1\} \tag{12}$$

The objective function (1) is the minimization of the project completion time. The objective function (2) is the minimization of project energy consumption. Constraint (3) ensures that activities are allowed to start just once. Equation (4) computes the actual processing times of activities. Constraint (5) guarantees the precedence constraints of the project. Constraint (6) satisfies the resource limitations. Constraint (7) implies that a single speed level can be chosen for processing activities. Equation (8) calculates the actual energy consumption required by resource k to perform activity j at speed level l. Constraint (9) implies that the makespan must be greater than or equal to the maximum completion times on all resources. Constraint (10) defines the total energy consumption of the project. Constraints (11) and (12) define the feasible scope of decision variables.

MOFOA for the EE-RCPSP

This section proposes a multi-objective fruit fly optimization algorithm (MOFOA) to solve the EE-RCPSP. One of the advantages of the fruit fly optimization algorithm (FOA) in comparison to some of the other swarm intelligence methods is that the FOA is a simple method to implement since it requires less calculation and it does not have many input parameters to adjust. The FOA has a strong expanding capability in its smell-based search procedure. Despite having a strong smell-based search procedure, the vision-based search procedure sometimes entraps the FOA in a local optima (Huang et al., 2017). Hence, in this paper, the operators of the MOFOA for searching the solution space are developed, and a multi-swarm strategy is used to avoid the premature convergence of the FOA to local optima. The procedures of the MOFOA are described in the following sub-sections.

Solution representation and decoding procedure

In the MOFOA, each fruit fly is considered as a solution of the EE-RCPSP. The key point for solving the EE-RCPSP is to determine on which resources each activity is processed. Besides, it is important to determine the appropriate processing speed levels for the resources. Therefore, a feasible solution consists of three vectors, namely an activity list, a resource list, and a speed vector. The activity list is

feasible based on the precedence constraints of the project and it shows the order of activities entering the scheduling process. On an activity list, activities must be positioned after all their predecessors. A resource assignment vector determines the resources assigned to each activity. A speed vector shows the speed levels of resources assigned to activities. Figure 1 shows a sample project with eight non-dummy activities. There are four available resources to execute project activities. Suppose that resources have three processing speed levels. Figure 2 illustrates a solution for this project. According to Figure 2, the resources "2" and "4" are assigned to activity "1". Both resources are set to operate at speed level "2" to execute this activity. We have used the serial schedule generation scheme (S-SGS) (Kadri and Boctor, 2018), as a decoding procedure, to produce schedules for the EE-RCPSP.

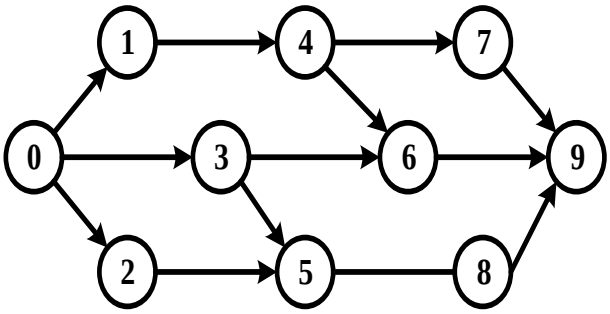

Fig. 1. The AON network of the example

Activity list	2	1	4	3	6	5	8	7
Resource assignment vector	1	2 , 4	3	3	4	2	1	1 , 3
Speed vector	1	2	2	1	1	3	3	2

Fig. 2. A feasible solution for the EE-RCPSP

Initialization

The MOFOA uses multiple-swarm procedure to construct population. The number of swarms in the population is denoted as *NS*. The locations of fruit fly swarms are considered as solutions. As mentioned in Section 4.1, each solution is represented as a $3\times N$ vector, where N is the number of project activities. Therefore, the MOFOA generates *NS* vectors in the initialization phase. The MOFOA stores non-dominated solutions in a Pareto archive (PA).

Smell-based search

The MOFOA uses the smell-based search as its main search procedure. In this procedure, S fruit flies are produced in the surrounding of each swarm. These fruit flies form the sub-swarm. The smell-based search procedure is implemented by using search operators (Wang and Zheng, 2018). Fang and Wang (2012) proposed an operator which swaps two adjacent activities without precedence relations. This operator is used to explore the neighborhood for finding new activity lists. Suppose that activity j has been selected randomly to be changed with regard to its assigned resources. For the resource assignment list, a new operator is designed, which reassigns an appropriate resource to activity j. Another operator is developed to change the speed level of the resources assigned to the chosen activity.

Vision-based search

After generating S fruit flies in the smell-based search, the solutions in each sub-swarms are evaluated. Then, the best found solution takes the place of the central location of the sub-swarm if more appropriate outputs are acquired. Therefore, one best solution has to be chosen from S+1 solutions. Opricovic and Tzeng (2007) proposed the VIKOR method as a compromise multi-attribute decision making (MADM) for optimization problems, where there are multiple criteria. The VIKOR determines how good an individual is among a set of non-dominated solutions. Therefore, the VIKOR is employed to choose the best solution in each sub-swarm. Based on the VIKOR, the alternatives are ranked based on the measure of closeness to the ideal solution. Figure 3 shows the procedure of the VIKOR method when objectives are needed to be minimized.

Step 1. Determine the best (f_c^*) and the worst (f_c^-) values of all criteria among solutions. f_c^* and f_c^- are obtained by Equations (13) and (14) respectively, if the c^{th} metric represents a cost:

$$f_c^* = \min_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1, 2, ..., \rho \qquad \forall c = 1, 2, ..., CR \tag{13}$$

$$f_c^- = \max_{\gamma} f_{\gamma c} \qquad \forall \gamma = 1, 2, ..., \rho \qquad \forall c = 1, 2, ..., CR \tag{14}$$

Step 2. Calculate the maximum group utility (ξ_γ) and the minimum individual regret of the opponent (τ_γ) values by the following formulas:

$$\xi_\gamma = \sum_{c=1}^{CR} \omega_c \left(f_c^* - f_{\gamma c} \right) / \left(f_c^* - f_c^- \right) \qquad \forall \gamma = 1, 2, ..., \rho \tag{15}$$

$$\tau_\gamma = \max_{c} \left[\omega_c \left(f_c^* - f_{\gamma c} \right) / \left(f_c^* - f_c^- \right) \right] \tag{16}$$

Where, ω_c is the weight of the c^{th} criterion. In this study, the weights of the criteria are the same.

Step 3. Compute the VIKOR index for each alternative (Q_γ) ($\forall \gamma$= 1, 2, …, ρ) using Equation (17):

$$Q_\gamma = \upsilon \left(\xi_\gamma - \xi^* \right) / \left(\xi^- - \xi^* \right) + (1 - \upsilon) \left(\tau_\gamma - \tau^* \right) / \left(\tau^- - \tau^* \right) \qquad \forall \gamma = 1, 2, ..., \rho \tag{17}$$

Where, υ and $(1-\upsilon)$ represent the weights for the strategies of maximum group utility and individual regret, respectively. In this paper, υ is set to 0.5. In Equation (17), $\xi^* = \min \xi_\gamma$, $\xi^- = \max \xi_\gamma$, $\tau^* = \min \tau_\gamma$, and $\tau^- = \max \tau_\gamma$.

Step 4. Rank the alternatives (solutions) by sorting the values ξ , τ and Q in ascending order. Three ranking lists are obtained by sorting these values.

Step 5. Propose as a compromise solution, the alternative with the minimum VIKOR index. The solution with the least VIKOR index is the best-ranked solution.

Fig. 3. Procedure of the VIKOR

After the smell-based search, there are *NS* new fruit fly swarms. Therefore, the MOFOA updates the population by selecting *NS* solutions from 2×*NS* solutions. In this respect, the non-dominated sorting technique proposed by Deb et al., (2002) is used to sort the candidate solutions in order to find non-dominated solutions for the Pareto front.

Elitism in the MOFOA

In each iteration, the Pareto archive (PA) is updated by the newly found non-dominated solutions. In this respect, the newly found non-dominated solutions are compared with the individuals existing in the PA. If a newly found non-dominated solution (*y*) is not dominated by the solutions in the PA, it will be added to the PA. The number of elite solutions (*NE*) in the PA is limited. The solutions of the PA which are dominated by the solution *y* will be removed from the PA.

Procedures of the MOFOA

Figure 4 illustrates the flowchart of the MOFOA. Operators of the smell-based search are used to find the Pareto optimal solutions. In the vision-based search procedure, the MOFOA uses the VIKOR and the non-dominated sorting method for multi-objective evaluations. The Pareto archive is updated in each iteration by the newly found non-dominated solutions. The MOFOA stops when the maximum number of iterations (*MaxIt*) is reached.

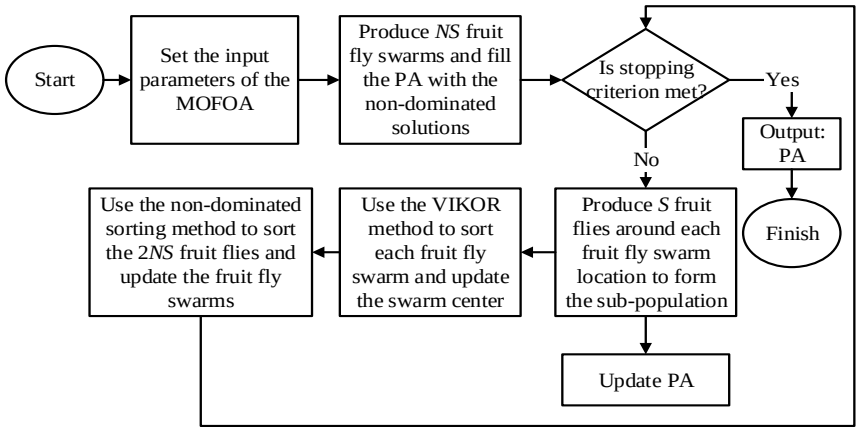

Fig. 4. Flowchart of the MOFOA

Experimental results

In this section, the performance of the MOFOA is compared with three meta-heuristics such as the NSGA-II, SPEA-II and PESA-II. All algorithms have been coded in the Matlab R2017b software. We have run the algorithms on a personal computer with Intel Core 2 Quad processor Q8200 (4M Cache, 2.33 GHz, 1333 MHz FSB) and 4GB memory.

Test problems

To evaluate the proposed model, 200 precedence networks have been taken randomly from the library of project scheduling problem (PSPLIB) (Kolisch and Sprecher, 1996) with 30, 60, 90 and 120 non-dummy activities. These standard test instances contain the activities, standard durations, precedence relations, required amount of resources for activities and the available amount of resources. However, some new data are needed for our proposed mathematical model. The required data are generated as follows:

1. The workload of activity j on resource k (w_{jk}) is following a uniform distribution on the interval $[5, 50]$.
2. The basic energy consumption rate for activity j on resource k (g_{jk}) is uniformly chosen on the interval $[4, 18]$.
3. The rate of energy consumption for activity j on resource k with speed v_l is calculated by $g_{jkl} = g_{jk}.v_l^{\alpha} (\alpha > 1)$.
4. Based on the energy consumption rate, the actual energy consumption of activity j on resource k with speed v_l is obtained as $e_{jkl} = w_{jk}.g_{jk}.v_l^{\alpha-1}$. Since $\alpha - 1$ is more than zero and $w_{jk}.g_{jk}$ is constant for activity j on resource k, the faster v_l is, the higher e_{jkl} will be. In this study, we set $\alpha = 3$.
5. We set $L = 4$, and $v_l \in \{0.75, 1, 1.25, 1.5\}$ for speed levels of processing.

Test problems are categorized into three sizes of small, medium and large. Test instances with 30 activities (J=30) are considered as small size problems. Test problems with 60 (J=60) activities are medium size problems and test problems with 90 (J=90) and 120 (J=120) activities are large size problems.

Performance measures

It is challenging to evaluate an algorithm when the objectives conflict with each other (Mehdizadeh et al., 2018). Five multi-objective performance measures are used to evaluate the algorithms:

- **Number of non-dominated solutions (*NOS*):** The number of non-dominated solutions on the Pareto front is a metric to evaluate a multi-objective optimization algorithm. The higher the number of non-dominated solutions, the more alternatives are available for decision makers to choose from (Pargar et al., 2018).
- **Mean ideal distance (*MID*):** This metric measures the distance between the non-dominated solutions of the Pareto front obtained by an algorithm and the ideal point. This metric is calculated by Eq. (18) (Zitzler and Thiele, 1998):

$$MID = \frac{\sum_{i=1}^{NOS} \sqrt{\left(OFV_1^i - OFV_1^*\right)^2 + \left(OFV_2^i - OFV_2^*\right)^2}}{NOS} \tag{18}$$

where, NOS represents the number of non-dominated solutions obtained by an algorithm. OFV_1^i and OFV_2^i denote the values of the first and the second objective function for solution i, respectively. Ideal points of the first and the second objective functions are represented as OFV_1^* and OFV_2^*, respectively. Lower values of the *MID* metric imply better performance of an algorithm.

- **Spacing metric (*SM*):** This metric evaluates the distribution of non-dominated solutions obtained by an algorithm. This metric is computed using Equation (19) (Schott, 1995):

$$SM = \sqrt{\frac{1}{NOS-1} \sum_{i=1}^{NOS} \left(D_i - \overline{D}\right)^2} \tag{19}$$

where, D_i is the Euclidean distance between solutions of the Pareto front. D_i is obtained by Equation (20):

$$D_i = \min_{i'} \left\{ \left| OFV_1^i - OFV_1^{i'} \right| + \left| OFV_2^i - OFV_2^{i'} \right| \right\}; \quad i, i' = 1, 2, ..., NOS \tag{20}$$

$\bar{D}$ denotes the average of D_is, calculated as $\bar{D} = \sum_{i=1}^{NOS} D_i / NOS$. OFV_1 and OFV_2 denote the values of the first and the second objective functions, respectively. Lower values of this metric indicate that solutions are more uniformly distributed.

- **Diversification metric (*DM*):** The extension of the Pareto front is measured by the diversification metric (*DM*). Higher values of *DM* imply that solutions have better diversity. This performance measure is obtained by Equation (21) (Zitzler et al., 2000):

$$DM = \sqrt{\left(\max_{i=1:NOS} OFV_1^i - \min_{i=1:NOS} OFV_1^i\right)^2 + \left(\max_{i=1:NOS} OFV_2^i - \min_{i=1:NOS} OFV_2^i\right)^2} \quad (21)$$

- **Computation time (CPU time):** The computation time is a well-known metric to assess the efficiency of an algorithm in terms of rapidity.

Table 1. Parameters and their levels

Algorithm	Parameter	Symbol	Factor level			
			1	**2**	**3**	**4**
MOFOA	Number of swarms	*NS*	10	50	100	200
	Size of sub-swam	*S*	5	10	15	20
	Number of elite solutions (Size of PA)	*NE*	5	10	20	30
	Maximum number of iterations	*MaxIt*	100	150	200	300
NSGA-II	Crossover rate	p_c	0.75	0.80	0.85	0.90
	Mutation rate	p_m	0.10	0.15	0.20	0.25
	Number of solutions in population	*Npop*	10	50	100	200
	Maximum number of iterations	*MaxIt*	100	150	200	300
SPEA-II	Crossover rate	p_c	0.75	0.80	0.85	0.90
	Mutation rate	p_m	0.10	0.15	0.20	0.25
	Number of solutions in population	*Npop*	10	50	100	200
	Maximum number of iterations	*MaxIt*	100	150	200	300
	Archive size	*ARS*	5	10	15	20
PESA-II	Crossover rate	p_c	0.75	0.80	0.85	0.90
	Mutation rate	p_m	0.10	0.15	0.20	0.25
	Number of solutions in population	*Npop*	10	50	100	200
	Maximum number of iterations	*MaxIt*	100	150	200	300
	Archive size	*ARS*	5	10	15	20

Calibrating parameters of algorithms

In this paper, the Taguchi method – as a design of experiments (DOE) technique (Afruzi et al., 2014) – is used to tune parameters of algorithms. Taguchi proposed a statistic called Signal-to-Noise (*S/N*) ratio to evaluate a process. "Signal" represents mean response variable, with its greater value being desirable. On the other hand, "Noise" depicts standard deviation and smaller values of it are preferred. Therefore, greater values of *S/N* ratio will lead to variability reduction. Table 1 shows the control factors of the MOFOA, NSGA-II, SPEA-II and PESA-II methods.

Rahmati et al. (2013) presented a response variable known as the multi-objective coefficient of variation (*MOCV*) for the Taguchi method. The *MOCV* incorporates both conversion and diversity by considering the *MID* and *DM* metrics. The *MID* estimates the convergence rate of the algorithms, while *DM* measures the diversity of methods. The *MOCV* is obtained as follows:

$$MOCV = \frac{MID}{DM} \tag{22}$$

Five test instances are randomly chosen and each problem is run for ten replicates to obtain reliable results. Therefore, 50 results are obtained for each experiment. The result of each problem is equal to the best result among ten runs of that problem. To compute the *MOCV*, the results of *MID* and *DM* should be converted to relative percentage difference (*RPD*). Equation (23) is used to calculate the *RPD* (Gao et al., 2013):

$$RPD = \frac{\left|\mathrm{Alg}_{sol} - Best_{sol}\right|}{Best_{sol}} \times 100 \tag{23}$$

where Alg_{sol} denotes the value of metric acquired by an algorithm, while $Best_{sol}$ represents the best obtained value of a metric. For each experiment, the average of *RPD*s ($\overline{RPD}$) are computed. In the next step, $MOCV = \overline{RPD}(MID)/\overline{RPD}(DM)$ is calculated for all experiments. Figures 5 to 8 show the *S/N* ratio plots for parameters of the MOFOA, NSGA-II, SPEA-II and PESA-II, respectively. Optimal values of parameters have been reported in Table 2.

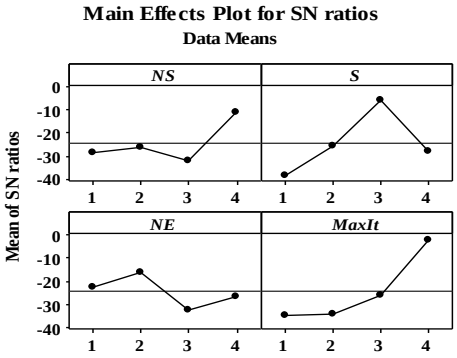

Fig. 5. The *S*/*N* ratio plot of the MOFOA in the Taguchi methodology

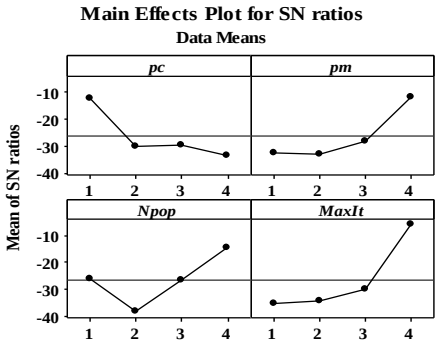

Fig. 6. The *S*/*N* ratio plot of the NSGA-II in the Taguchi methodology

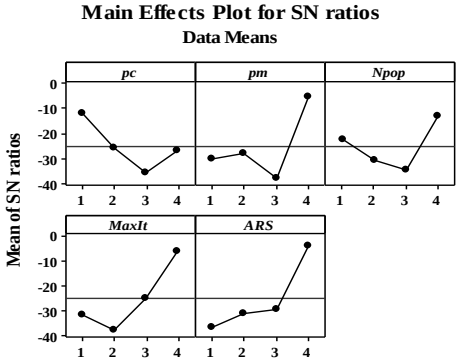

Fig. 7. The *S*/*N* ratio plot of the SPEA-II in the Taguchi methodology

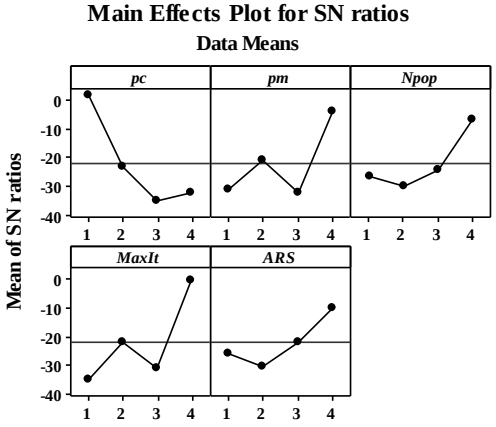

Fig. 8. The *S/N* ratio plot of the PESA-II in the Taguchi methodology

Table 2. Optimal values of parameters

Algorithms	Parameters' values				
MOFOA	NS = 200	S = 15	NE = 10	MaxIt = 300	
NSGA-II	$p_c = 0.75$	$p_m = 0.25$	Npop = 200	MaxIt = 300	
SPEA-II	$p_c = 0.75$	$p_m = 0.25$	Npop = 200	MaxIt = 300	ARS = 20
PESA-II	$p_c = 0.75$	$p_m = 0.25$	Npop = 200	MaxIt = 300	ARS = 20

Computational results

The performance of the MOFOA is compared with the NSGA-II, SPEA-II and PESA-II in solving test problems. Each algorithm has been run for ten times to remove uncertainties. Table 3 shows the average values of performance measures obtained by these algorithms. Based on the outputs summarized in Table 3, the following outlines have been obtained:

1. The MOFOA has been more successful in finding higher number of non-dominated solutions. Therefore, the decision makers have more options.
2. The convergence of the MOFOA to the optimal Pareto front is considerably better than the convergence of other algorithms.
3. The diversity of the solutions obtained by the MOFOA is more than the diversity of the solutions acquired by the NSGA-II, SPEA-II and PESA-II.

4. The computation time of the PESA-II is far less than the other methods.

Figures 9 to 13 show the best results obtained by algorithms in terms of performance measures. According to these figures, the MOFOA has achieved better results in terms of NOS, MID, SM and DM in comparison with other method. In terms of CPU time, the PESA-II needed less time to solve test problems.

Table 3. Comparison of algorithms in terms of performance measures

Number of activities	Algorithm	Performance measure				
		NOS	*MID*	*SM*	*DM*	CPU time
J=30	MOFOA	19.40	50.06	0.92	4974.32	41.38
	NSGA-II	12.86	314.19	2.84	3138.64	45.07
	SPEA-II	14.84	299.51	2.05	3328.71	57.03
	PESA-II	11.01	266.58	1.86	3607.67	35.39
J=60	MOFOA	32.27	135.83	3.42	5507.13	92.26
	NSGA-II	23.71	467.69	5.38	4273.92	98.01
	SPEA-II	24.53	431.44	5.24	4826.04	123.28
	PESA-II	20.18	401.47	4.88	5119.93	79.47
J=90	MOFOA	41.01	658.86	12.39	9834.10	132.06
	NSGA-II	32.36	784.13	26.37	8199.08	141.44
	SPEA-II	30.44	721.77	25.04	8312.26	179.42
	PESA-II	36.11	706.19	24.51	8956.40	111.58
J=120	MOFOA	49.87	2473.97	53.72	10872.34	173.64
	NSGA-II	40.28	4235.35	81.34	9101.17	188.58
	SPEA-II	43.96	4014.22	79.16	9574.99	236.76
	PESA-II	45.52	3856.79	73.39	10029.85	151.01

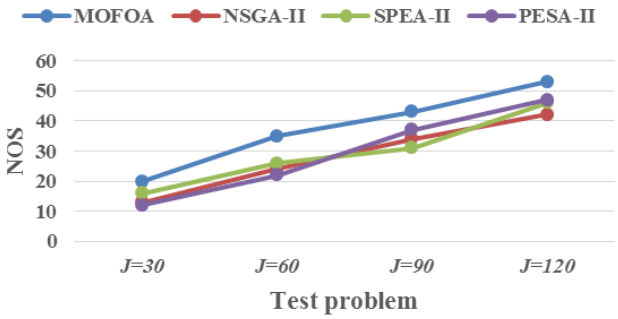

Fig. 9. Comparison of algorithms in terms of *NOS*

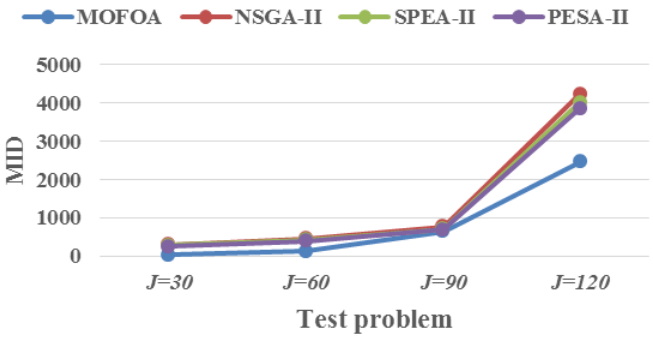

Fig. 10. Comparison of algorithms in terms of *MID*

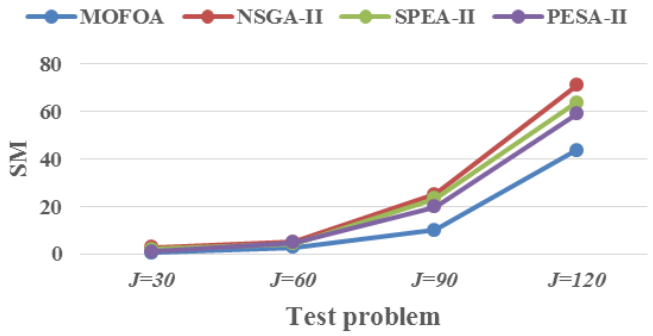

Fig. 11. Comparison of algorithms in terms of *SM*

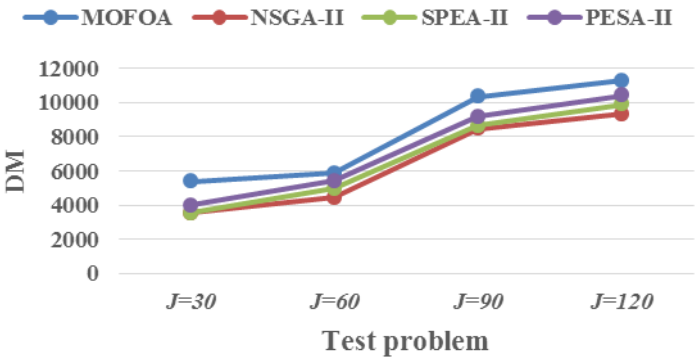

Fig. 12. Comparison of algorithms in terms of *DM*

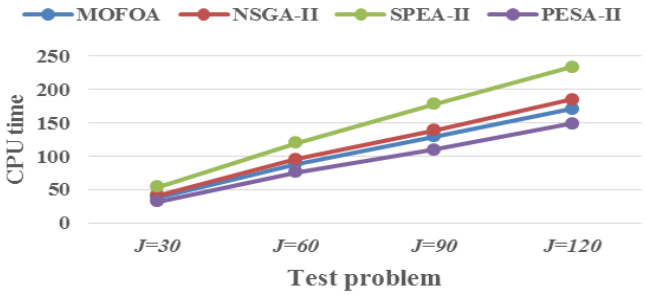

Fig. 13. Comparison of algorithms in terms of CPU time

To validate the results obtained by the algorithms, the outputs of three algorithms are compared with the optimal objective function values obtained by the GAMS software (version 24.1.2). The comparisons have been made for small size problems with J=30 activities. In this respect, twenty test problems with 30 activities have been chosen randomly. Table 4 reports the best results obtained by algorithms in ten runs. C_{max} and E represent the best values of objectives found by algorithms. As shown in Table 4, the MOFOA has achieved the best values for the first and the second objective functions in most of the problems.

Table 4. The best results obtained by each algorithm

Prob.	MOFOA		NSGA-II		SPEA-II		PESA-II		GAMS	
	C_{max}	E	C_{max}	E	C_{max}	E	C_{max}	E	C_{max}	E
1	60	6370.39	60	6370.39	68	6994.22	60	6370.39	60	6370.39
2	62	6231.18	62	6231.18	69	7074.45	62	6231.18	62	6231.18
3	53	6816.54	63	7352.31	53	6816.54	53	6816.54	53	6816.54
4	51	7119.89	51	7119.89	62	7558.48	57	7583.39	51	7119.89
5	63	7067.086	56	6591.47	56	6591.47	56	6591.47	56	6591.47
6	74	6094.072	71	6525.77	64	6081.56	74	6204.37	64	6081.56
7	59	6372.85	59	6372.85	59	6372.85	59	6372.85	59	6372.85
8	37	7948.84	37	7948.84	53	8261.03	48	8290.63	37	7948.84
9	73	5503.33	73	5503.33	73	5503.33	73	5503.33	73	5503.33
10	61	6310.81	61	6310.81	73	6553.95	61	6310.81	61	6310.81
11	71	5577.56	71	5577.56	71	5577.56	71	5577.56	71	5577.56
12	46	7319.99	46	7319.99	46	7319.99	46	7319.99	46	7319.99
13	38	7881.47	50	8445.50	53	8397.91	38	7881.47	38	7881.47
14	48	7268.41	48	7268.41	48	7268.41	48	7268.41	48	7268.41
15	68	5754.23	80	5948.42	68	5754.23	68	5754.23	68	5754.23
16	74	5438.4	74	5438.40	74	5438.40	74	5438.40	74	5438.40
17	49	7158.49	60	7623.80	49	7158.49	49	7158.49	49	7158.49
18	55	6708.71	55	6708.71	55	6708.71	55	6708.71	55	6708.71
19	65	6037.65	73	6468.03	65	6037.65	74	6180.80	65	6037.65
20	66	6014.18	78	6047.49	66	6014.18	66	6014.18	66	6014.18
Avg.	58.65	6549.70	61.40	6658.66	61.25	6674.17	59.60	6578.86	57.80	6525.30

Table 5 reports the gaps between the best found solutions of meta-heuristics and optimal solutions obtained by the GAMS software. The optimal solutions found by the GAMS software have been achieved in around 2000 seconds. The relative gaps between the optimal solutions and the solutions of evolutionary algorithms are obtained as follows (Bashiri et al., 2012):

$$GAP = \frac{OFV^{alg} - OFV^{*}}{OFV^{*}} \tag{24}$$

where OFV^{alg} and OFV^{*} are the objective function values obtained by the evolutionary algorithm and GAMS, respectively. As reported in Table 5, the MOFOA has been successful in finding optimal solutions in 90% of problems.

Table 5. Comparison of meta-heuristics and GAMS

Prob.	GAP(MOFOA)		GAP(NSGA-II)		GAP(SPEA-II)		GAP(PESA-II)	
	C_{max}	E	C_{max}	E	C_{max}	E	C_{max}	E
1	0.00	0.00	0.00	0.00	0.13	0.10	0.00	0.00
2	0.00	0.00	0.00	0.00	0.11	0.14	0.00	0.00
3	0.00	0.00	0.19	0.08	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.22	0.06	0.12	0.07
5	0.13	0.07	0.00	0.00	0.00	0.00	0.00	0.00
6	0.16	0.01	0.11	0.07	0.00	0.00	0.16	0.02
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.43	0.04	0.30	0.04
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.20	0.04	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	0.00	0.00	0.32	0.07	0.39	0.07	0.00	0.00
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.00	0.00	0.18	0.03	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17	0.00	0.00	0.22	0.07	0.00	0.00	0.00	0.00
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19	0.00	0.00	0.12	0.07	0.00	0.00	0.14	0.02
20	0.00	0.00	0.18	0.01	0.00	0.00	0.00	0.00

To have a reliable comparison between performances of algorithms, a one-way analysis of variances (ANOVA) at a 95% confidence interval is conducted. The following hypothesis test has been considered to determine whether the differences between algorithms are statistically significant or not. The null hypothesis (H_0) is rejected if *P*-value of hypothesis test is smaller than 0.05.

H_0; The outputs of algorithms are not significantly different.

H_1; The outputs of algorithms are significantly different. (25)

Tables 6, 7, 8, 9 and 10 report the results of ANOVA tests of algorithms in terms of the *NOS*, *MID*, *SM*, *DM* and CPU time, respectively. To save space, only the results for large size problems (*J* =120) have been reported in these tables. According to the results, there is a significant difference between the performances of the algorithms in terms of the *NOS*, *MID*, *SM*, and *DM* metrics, while there is no significant difference between the performances of algorithms in terms of CPU time.

Table 6. ANOVA test for the difference of algorithms in terms of *NOS*

Source	SS	MS	F	P-Value	Results
Columns	1334.25	444.75	7.97	0.0034	H_0 is rejected.
Error	669.50	55.792			
Total	2003.75				

Table 7. ANOVA test for the difference of algorithms in terms of *MID*

Source	SS	MS	F	P-Value	Results
Columns	75542890	2518096.30	224.99	8.26e-011	H_0 is rejected.
Error	134306.90	11192.20			
Total	7688595.90				

Table 8. ANOVA test for the difference of algorithms in terms of *SM*

Source	SS	MS	F	P-Value	Results
Columns	1543.01	514.33	7.15	0.0052	H_0 is rejected.
Error	863.70	71.97			
Total	2406.71				

Table 9. ANOVA test for the difference of algorithms in terms of *DM*

Source	SS	MS	F	P-Value	Results
Columns	1.01e+007	3367645.60	6.06	0.0094	H_0 is rejected.
Error	6.66e+006	555743.70			
Total	1.67e+007				

Table 10. ANOVA test for the difference of algorithms in terms of CPU time

Source	SS	MS	F	P-Value	Results
Columns	2934.80	978.26	3.40	0.13	H_0 is not rejected.
Error	1151.85	287.96			
Total	4086.65				

Conclusions

This paper investigated an energy-efficient speed-scaling resource-constrained project scheduling problem to optimize both project completion time and total energy consumption. To tackle the problem which is strongly NP-hard, a multi-objective fruit fly optimization algorithm (MOFOA) was developed. This method uses a new solution representation consisting of activity vector, resource vector and speed vector. In each iteration, the MOFOA utilizes the VIKOR method to choose the best solutions (fruit flies) in each sub-swarm. The MOFOA was used to solve a considerable number of test problems available on the PSPLIB with 30, 60, 90 and 120 activities. To evaluate the performance of the MOFOA, three other meta-heuristics were employed to solve the same test problems. The algorithms have been compared in terms of several performance measures. Computational results have significantly demonstrated that the MOFOA has been more successful than other methods in terms of most of the metrics. To validate the outputs of the MOFOA, the solutions of this algorithm were compared to the optimal solutions of small size problems provided by the GAMS software. These comparisons show that the proposed algorithm has found the optimal solutions in most of the cases. There are some suggestions to extend the current research:

- Considering power-down mechanism as another energy-efficient mechanism for the RCPSP.
- Using other multi-criteria decision-making techniques in the MOFOA to rank fruit flies in each sub-swarm.

References

Afruzi, E., Najafi, A. A., Roghanian, E., & Mazinani, M. (2014). A Multi-Objective Imperialist Competitive Algorithm for solving discrete time, cost and quality trade-off problems with mode-identity and resource-constrained situations. *Computers & Operations Research*, *50*, 80-96.

Bampis, E., Dürr, C., Kacem, F., & Milis, I. (2012). Speed-scaling with power down scheduling for agreeable deadlines. *Sustainable Computing: Informatics and Systems*, *2*(4), 184-189.

Bashiri, M., Badri, H., & Talebi, J. (2012). A new approach to tactical and strategic planning in production–distribution networks. *Applied Mathematical Modelling*, *36*, 1703-1717.

Blazewicz, J., Lenstra, J. K., & Kan, A. (1983). Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics*, *5*, 11-24.

Che, A., Wu, X., Peng, J., & Yan, P. (2017). Energy-efficient bi-objective single-machine scheduling with power-down mechanism. *Computers & Operations research*, *85*, 172-183.

Che, A., Zhang, S., & Wu, X. (2017). Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs. *Journal of Cleaner Production*, *156*, 688-697.

Dai, M., Tang, D., Giret, A., Salido, M. A., & Li, W. D. (2013). Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer-Integrated Manufacturing*, *29*(5), 418-429.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*, 182-197.

Ding, J. Y., Song, S. J., & Wu, C. (2016). Carbon-efficient scheduling of flow shops by multi-objective optimization. *European Journal of Operational Research*, *248*(3), 758-771.

Fang, K., Uhan, N. A., Zhao, F., & Sutherland, J.W. (2011). A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, *30*(4), 234-240.

Fang, K., Uhan, N. A., Zhao, F., & Sutherland, J. W. (2016). Scheduling on a single machine under time-of-use electricity tariffs. *Annals of Operations Research*, *238*(1-2), 199-227.

Fang, C., & Wang, L. (2012). An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem. *Computers & Operations Research*, *39* (5), 890-901.

Gahm, C., Denz, F., Dirr, M., & Tuma, A. (2016). Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, *248*(3), 744-757.

Gao, J., Chen, R., & Deng, W. (2013). An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Production Research*, *51*, 641-651.

Huang, L., Wang, G. C., Bai, T., & Wang, Z. (2017). An improved fruit fly optimization algorithm for solving traveling salesman problem. *Frontiers of Information Technology & Electronic Engineering*, *18*(10), 1525-1533.

Kadri, R. L., & Boctor, F. F. (2018). An efficient genetic algorithm to solve the resource-constrained project scheduling problem with transfer times: The single mode case. *European Journal of Operational Research*, *265*(2), 454-462.

Kolisch, R., & Sprecher, A. (1996). PSPLIB - A project scheduling problem library. *European Journal of Operational Research*, *96*(1), 205-216.

Liu, G. S., Zhou, Y., & Yang, H. D. (2017). Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time. *Journal of Cleaner Production*, *147*, 470-484.

Liu, Y., Dong, H., Lohse, N., Petrovic, S., & Gindy, N. (2014). An investigation into minimising total energy consumption and total weighted tardiness in job shops. *Journal of Cleaner Production*, *65*, 87-96.

Lu, C., Gao, L., Li, X., Pan, Q., & Wang, Q. (2017). Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm. *Journal of Cleaner Production*, *144*, 228-238.

Luo, H., Du, B., Huang, G. Q., Chen, H., & Li, X. (2013). Hybrid flow shop scheduling considering machine electricity consumption cost. *International Journal of Production Economics*, *146*(2), 423-439.

Mansouri, S. A., Aktas, E., & Besikci, U. (2016). Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption. *European Journal of Operational Research*, *248*(3), 772-788.

Mehdizadeh, E., Niaki, S.T.A, Hemati, M. (2018). A bi-objective aggregate production planning problem with learning effect and machine deterioration: Modeling and solution. *Computers & Operations research*, *91*, 21-36.

Merkert, L., Harjunkoski, I., Isaksson, A., Saynevirta, S., Saarela, A., & Sand, G. (2015). Scheduling and energy – Industrial challenges and opportunities. *Computers & Chemical Engineering*, *72*, 183-198.

Mokhtari, H., & Hasani, A. (2017). An energy-efficient multi-objective optimization for flexible job-shop scheduling problem. *Computers & Chemical Engineering*, *104*, 339-352.

Mori, M., Fujishima, M., Inamasu, Y., & Oda, Y. (2011). A study on energy efficiency improvement for machine tools. *CIRP Annals – Manufacturing Technology*, *60*(1), 145-148.

Mouzon, G., & Yildirim, M. B. (2008). A framework to minimize total energy consumption and total tardiness on a single machine. *International Journal of Sustainable Engineering*, *1*(2), 105-116.

Opricovic, S., & Tzeng, G. H. (2007). Extended VIKOR method in comparison with outranking methods. *European Journal of Operational Research*, *178*, 514–529.

Pargar, F., Zandieh, M., Kauppila, O., & Kujala, J. (2018). The effective of worker learning on scheduling jobs in a hybrid flow shop: A bi-objective approach. *Journal of Systems Science and Systems Engineering*, *27*(3), 265-291.

Rahmati, S. H. A., Hajipour, V., & Niaki, S. T. A. (2013). A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem. *Applied Soft Computing*, *13*, 1728-1740.

Schott, J. R. (1995). *Fault tolerant design using single and multicriteria genetic algorithms optimization* (unpublished master's thesis). Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA.

Shrouf, F., Ordieres-Mere, J., Garcia-Sanchez, A., & Ortega-Mier, M. (2014). Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *Journal of Cleaner Production*, *67*, 197-207.

Tang, D., Dai, M., Salido, M. A., & Giret, A. (2016). Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization. *Computers in Industry, 81*, 82-95.

Wang, S., Liu, M., Chu, F., & Chu, C. (2016). Bi-objective optimization of a single machine batch scheduling problem with energy cost consideration. *Journal of Cleaner Production*, *137*, 1205-1215.

Wang, L., & Zheng, X. L. (2018). A knowledge-guided multi-objective fruit fly optimization algorithm for the multi-skill resource constrained project scheduling problem. *Swarm and Evolutionary Computation*, *38*, 54-63.

Yan, J., Li, L., Zhao, F., Zhang, F., Zhao, Q. (2016). A multi-level optimization approach for energy-efficient flexible flow shop scheduling. *Journal of Cleaner Production*, *137*, 1543-1552.

Zareei, M., & Hassan-Pour, H. A. (2015). A multi-objective resource-constrained optimization of time-cost trade-off problems in scheduling project. *Iranian Journal of Management Studies*, *8*(4), 653-685.

Zhai, Y., Biel, K., Zhao, F., & Sutherland, J. W. (2017). Dynamic scheduling of a flow shop with on-site wind generation for energy cost reduction under real time electricity pricing. *CIRP Annals*, *66*(1), 41-44.

Zhang, R., & Chiong, R. (2016). Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production*, *112*, 3361-3375.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multi objective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, *8*, 125–148.

Zitzler, E., & Thiele, L. (1998). Multi-objective optimization using evolutionary algorithms—a comparative case study. *International Conference on Parallel Problem Solving from Nature*. Springer, Berlin, Heidelberg, pp. 292–301, DOI: https://doi.org/10.1007/BFb0056872.