Computers & Industrial Engineering 91 (2016) 109–128

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

A Lévy flight embedded particle swarm optimization for multi-objective parallel-machine scheduling with learning and adapting considerations

S.H. Pakzad-Moghaddam

School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran

ARTICLE INFO

Article history:
Received 4 February 2015
Received in revised form 29 October 2015
Accepted 31 October 2015
Available online 11 November 2015

Keywords:
Parallel-machine scheduling
Learning effect
Adapting ability
Possibilistic programming
Particle swarm optimization
Lévy flight

ABSTRACT

The study at hand is devoted to schedule jobs on uniform parallel processors which are capable of adapting as well as learning. The problem is formulated into a bi-objective mixed integer mathematical model in which several parameters are supposed to follow triangular possibility distributions. Converting the aforementioned model, an auxiliary equivalent mixed integer crisp one is constructed through an interactive possiblistic programming approach. Due to the NP-hardness of the problem, swarm intelligence is hired by applying a highly modified Particle Swarm Optimization (PSO) method. In the proposed evolutionary method, called Lévy Flight Embedded Particle Swarm Optimization (LFEPSO), uniformly distributed walks are replaced by Lévy flights. In order to check the validity of the model and the performance of the proposed LFEPSO, twenty-six data sets are generated on a random basis. As the numerical results indicate, embedding Lévy flights made a tremendous improvement towards solving the local optima problem of the traditional PSO. Compared to the exact solution method, LFEPSO has shown an outstanding performance in scheduling 8–500 jobs on 3–50 parallel processors. In addition, numerical results reveal the significant role of considering the adapting ability in accurately modeling real-world and huge-sized problems.

1. Introduction

Applying parallel machines is an indispensable part of numerous industrial activities. Regarding such activities, many studies have been carried out to schedule jobs on parallel machines (see for example, Cheng, Kang, & Ng, 2007; Cheng & Sin, 1990; Elvikis & T'kindt, 2014; Li & Yang, 2009; Yeh, Lai, Lee, & Chuang, 2014). In this line of research, the effect of learning has been proved to play an important role in the accuracy of the modeling procedure. In such a way that neglecting the effect of learning is observed to invalidate the results in some cases. Such industrial cases are addressed in many previous studies (see for example, Biskup, 2008; Cheng, Wu, & Lee, 2008; Eren, 2009; Huang, Wang, & Ji, 2014; Mosheiov, 2001a, 2001b).

Several applications for applying learning curves were addressed by Heizer and Render (1999) such as internal labor forecasting, establishing costs and budgets, external purchasing and subcontracting. As another example, Webb (1994) referred to the semiconductor industry, where efficiency gains cause price to drop of about 10–30% per year. Also, processing of memory chips and circuit boards or running bottling plants, offset printing, highly customized products, the production of high-end electric tools, maintenance of airplanes and pimping cars are accounted as other applications of learning in industrial activities (Lu, Wei, & Wang, 2012).

Recently, Pakzad-Moghaddam, Mina, and Tavakkoli-Moghaddam (2014), elaborated the fact that in many real-life situations, operators are under the influence of their adapting/lingering capabilities in addition to the common learning/deteriorating effects. They defined adapting ability of an operator as his/her ability to adapt to unfamiliar jobs in addition to his/her experience acquired by performing similar jobs. Car-part industry is introduced by Pakzad-Moghaddam et al. (2014) as an example of manufacturing environments which is under the impression of learning and adapting at the same time. Similarly, performing such activities on uniform parallel machines is an example for the under-study scheduling problem. Which is about scheduling jobs to get processed on several parallel processors. The processors' processing rates increase as they learn and adapt. This study is aimed at minimizing machine hiring cost and maximum completion time simultaneously. This research is carried out to bridge the following gaps in the literature of scheduling:

- The impression of adapting ability on a single machine was first addressed by Pakzad-Moghaddam et al. (2014). To this day, adaptable parallel processors had never been studied in the

E-mail address: hpakzad@ut.ac.ir

http://dx.doi.org/10.1016/j.cie.2015.10.019
0360-8352/

literature of scheduling. To this end, in the current study, agility of the parallel processors increases as they learn from their experiences and adapt to new situations.

- This paper pioneers in addressing the adapting ratios as imprecise parameters. Hence, an efficient possibilistic programming approach is adopted from Torabi and Hassini (2008). This approach is applied to find compromised solutions regarding the ambiguous nature of the problem.
- A highly modified and well-organized meta-heuristic called Lévy Flight Embedded Particle Swarm Optimization is designed to solve the developed parallel machine scheduling problem. In order to model the original inspiration of PSO accurately, the simulated movement patterns are improved by applying Lévy walks. The main idea is to avoid particles getting trapped in local optima by accurately simulating the movement patterns of their real-world inspirations (i.e. birds and fishes). As the result of which, getting trapped in local optima is not a major concern for real birds/fishes towards their destination. In addition, particles are intelligently designed to implement an efficient search procedure.
- The proposed parallel machine scheduling problem is formulated into a bi-objective model in which both machine hiring cost and maximum completion time are aimed to get minimized, simultaneously.

The remainder of this paper is organized as follow. Notations which are required to present the mathematical formulations are given in Section 2. In Section 3, the problem is formulated into a Multiple Objective Possibilistic Mixed Integer Linear Programming (MOPMILP) model. First, problem environment, its definition and explanations for the problem's objectives and constraints are provided. In the next section, a possibilistic programming approach is applied in order to deal with the ambiguousness of the problem. Section 5 is devoted to organizing an efficient evolutionary method which is designed to deal with the NP-hardness of the problem at hand. Numerical results are presented in Section 6. Final conclusions and possible avenues for continuing this line of research are provided in the last section.

2. Notations

This section is devoted to address the notations which are required to present the rest of the study at hand. They are categorized into the two following categories. First is the general notations which are employed in describing the applied interactive possibilistic programming approach and the modified meta-heuristic method (i.e. LFEPSO). Indices, parameters and variables that the mathematical model is formulated upon are addressed in the second category.

2.1. General notations

$h, \widehat{h} = 1, \ldots, 4$	Objective number
$\widehat{\beta}$	Minimum acceptable possibility or the minimum acceptable degree of feasibility
$w = (w^p, w^m, w^o)$	Most pessimistic, possible and optimistic values weights which are calculated to convert fuzzy parameters into crisp numbers
θ_h	Relative importance weight of the h-th objective's degree of satisfaction
$\widehat{\gamma}$	Compensation coefficient which is applied in the possibilistic programming approach
z_h	Value calculated for objective h
μ_h	Degree of satisfaction obtained for objective h
λ_0	The minimum degree of satisfaction among those are calculated for all four objectives
υ	Any feasible solution to the proposed MOPMILP model
Ω	Set of all feasible solutions to the proposed MOPMILP model
$M^j_{g_i}$	Elements of the comparison pairwise matrix in the form of triangular fuzzy numbers (comparing objective j to i)
S_i	Fuzzy synthetic extent calculated for objective i
$d'(\mu_h)$	Relative weights calculated for the objectives' degree of possibility
$d(\mu_h)$	Normalized relative weights calculated for the objectives' degree of possibility
c_1, c_2	Personal and social awareness of particles respectively
r_1, r_2	Random numbers which are applied in the velocity update equation of the traditional PSO
w_t	Particles' inertia weight in time t
$x_{id}(t)/V_{id}(t)$	Position/velocity of particle i in time t concerning its d-th dimension
$Levy(\beta_i)$	A levy distribution with parameter β_i
$p^a_j(t)$	The actual time which is spent to process job j, where the current time equals to t
e^T_{lj}	The total effect made on job j as the result of processing job l
$CE_j/VE_j(t)/E_j(t)$	The constant, variable and total effects made respectively on the processing rate of job j at time t
$R_j(t)$	The maturated processing rate of job j at time t
$\widehat{t}_j$	The amount of time before maturation while processing job j
t'_j	The hypothetical amount of time which is required to finish job j, if its maturation limit is neglected
t^*_j	An auxiliary variable which is calculated via Eq. (54)
t^{real}_j	The actual time required to execute job j considering its maturation limit

2.2. Mathematical model's notations

Indices, parameters and decision variables which are required to form the mathematical representation of the proposed parallel machine scheduling problem with learning and adapting abilities, are listed as follows.

2.2.1. Indices

i, j	job	$i, j = 1, 2, \ldots, N$
k	machine	$k = 1, 2, \ldots, K$
t	time period	$t = 1, 2, \ldots, T$

2.2.2. Parameters

l	Time steps
λ_{jk}	Normal processing rate of job j for processor k
$\widetilde{e^c_{ijk}} = \left(e^{c^p}_{ijk,\widehat{\beta}}, e^{c^m}_{ijk,\widehat{\beta}}, e^{c^o}_{ijk,\widehat{\beta}}\right)$	Time-independent effect for processor k made by job i on job j and its corresponding most pessimistic, possible and optimistic values
$\widetilde{e^r_{ijk}} = \left(e^{r^p}_{ijk,\widehat{\beta}}, e^{r^m}_{ijk,\widehat{\beta}}, e^{r^o}_{ijk,\widehat{\beta}}\right)$	Time-dependent effect for processor k made by job i on job j and its corresponding most pessimistic, possible and optimistic values
g^U_{jk}	Upper bound for the total completion rate of job j on processor k
$\widetilde{\gamma_k} = (\gamma^p_k, \gamma^m_k, \gamma^o_k)$	Cost of hiring processor k and its corresponding most pessimistic, possible and optimistic values
$bigm \cong \infty$	Large positive number

2.2.3. Variables

$\rho_{jt} = \begin{cases} 1 \\ 0 \end{cases}$	binary	If a processor executes job j at time period t Other wise
$\alpha_{jk} = \begin{cases} 1 \\ 0 \end{cases}$	binary	If job j is processed by processor k Other wise
$x_{ij} = \begin{cases} 1 \\ 0 \end{cases}$	binary	If job j is scheduled to be placed right after job i Other wise
$y_{ij} = \begin{cases} 1 \\ 0 \end{cases}$	binary	If job j is scheduled to be placed after job i Other wise
$\eta_k = \begin{cases} 1 \\ 0 \end{cases}$	binary	If processor k is available for processing Other wise
$B_{ijt_0} = \begin{cases} (l \times \widetilde{e^c_{ijk}} \times y_{ij}) \\ +\sum_{t=1}^{t_0}(l^2 \times \widetilde{e^r_{ijk}} \times \beta_{it}) \\ 0 \end{cases}$	positive	If $\exists k \alpha_{ik} = \alpha_{jk}$ Other wise
C_j	positive	Completion time of job j
St_j	positive	Start time of processing job j
E_{jt}	positive	Total effect made on job j in time t before maturation
R_{jt}	positive	Maturated form of the total effect on job j in time t
$\widehat{\rho}_{jtk}$	positive	Representative for the product of ρ_{jt} and α_{jk}
$\widehat{c}_{ij}$	positive	Representative for the product of C_j and x_{ij}
$\widehat{R}_{jt}$	positive	Representative for the product of R_{jt} and β_{jt}
C_{max}	positive	Maximum completion time
$\widetilde{MHC} = (MHC^p, MHC^m, MHC^o)$	positive	Machine hiring cost and its corresponding most pessimistic, possible and optimistic values
π	positive	Dummy completion time

3. Problem formulation

The problem and its environment are first described. Then, an overview of the objectives and constraints is given. Finally, the mathematical representation for the given objectives and constraints is presented in this section.

3.1. Problem environment

In many industrial cases, processors are under the impression of learning. In general, we learn by experiencing or by following others' experiences. In either way, the experience is helpful in handling situations similar to the one in which the experience is obtained. Back to the scheduling, performing a job requires a variety of skills, which depend on the operator's amount of experience. On the other hand, agility of an operator in performing a hypothetical job depends on two matters as follows:

(a) The amount of time spent by the operator performing similar jobs i.e. experience.
(b) His/her ability in adapting to the job's special characteristics i.e. adapting ability.

Now imagine a new job with completely unfamiliar characteristics. In fact, the operators' experience is useless in performing such an unfamiliar job. Therefore, its processing time only depends on the adapting ability of the operator, which is required to adapt to the unfamiliar characteristics of the new job. In general, different jobs have some features in common as well as some unique characteristics which are bounded to a job itself. With this regard, taking the operator's adapting ability into account is as important as considering the operator's experience.

Due to the complexity and ambiguous nature of learning and adapting parameters which are adopted from the previous work of Pakzad-Moghaddam et al. (2014), they are assumed to be imprecise in nature. Concerning the multiplicity of constitutive elements of a machine's hiring cost (including its hidden and environmental costs) and their ambiguous nature, it is also needed to be treated as a fuzzy parameter. In this regard, the pattern of triangular fuzzy number is employed in order to formulate their imprecise nature. In the literature, the triangular possibility distribution is mentioned as the most common tool for modeling the fuzziness of imprecise parameters (Liang, 2006; Zimmermann, 1978).

3.2. Problem definition

Despite the previous studies carried out to schedule jobs on parallel machines, this work is devoted to schedule parallel machines/operators which are capable of adapting as well as learning. In other words, while operators usually learn how to do their job by performing a set of similar jobs, their agility in performing newly-assigned jobs (with unique characteristics) depend on their adapting ability. Why sometimes an amateur operator does a better job rather than an experienced one, while performing jobs with absolutely unique characteristics? This actually happens due to their different adapting abilities. As well, agility of the operator in performing similar jobs mostly depends on the amount of his/her obtained experience. In this study, Learning and adapting effects are supposed to have maturation limits. A set of candidate machines are available for processing desired jobs. Since the machines are not identical, each has a specific hiring cost of its own. As well, they have their differences in processing jobs, including processing, learning and adapting rates. Despite their differences, all machines are capable of processing all jobs. All jobs and machines are available from the beginning. The amount of process required on a job is not supposed to split among machines. Therefore each job should be performed on exactly one machine. Finally it is up to the decision maker to find the optimum sub-set of machines and the optimum job schedules. The aforementioned parallel machine scheduling problem is formulated into a multi-objective mixed-integer mathematical model with linear terms, through this section.

3.3. Objectives

Which machines to hire? Regarding machine's total hiring cost and maximum completion time which clearly act in opposite directions, it is a very complex decision to make. As the first step toward making this complex decision, a bi-objective mathematical model is formulated. Mathematical representations of the aforementioned objectives are given in the proposed mathematical model.

3.4. Constraints

Formulating the described scheduling problem into an appropriate mathematical model requires applying a variety of constraints. This guarantees the applicability of the obtained solutions. Pakzad-Moghaddam et al. (2014) proposed an analytical integration which is required to get solved in order to calculate the maximum completion time in scheduling environments with adapting/lingering considerations. In the problem formulation section of the work of Pakzad-Moghaddam et al. (2014), the proposed analytical integration is replaced with a numerical one to make it possible to solve the model by using common Mixed-Integer Programming (MIP) solvers. The same technique is hired in the current study to formulate a parallel-machine scheduling problem with adapting considerations. The applied constraints are categorized into three categories as follows.

(a) Such as any scheduling mathematical model, basic constraints are needed in order to find feasible values for start and completion times respecting the job schedule. These constraints are formulated by applying variable ρ_{jt}. This specific variable is applied in order to calculate the aforementioned integral numerically.
(b) The aforementioned numerical integration is formulated through the second group of constraints. Besides, these constraints are applied to take learning and adapting rates as well as maturation limit of the processing rates into consideration.
(c) Furthermore, the third categorized group of constraints are called logical constraints which are used to enforce binary and auxiliary variables to act the way they are expected to.

3.5. Mathematical formulation

Objectives

$$\min\ C_{\max} \tag{1}$$

$$\min\ \widetilde{MHC} = \sum_{k=1}^{K} \widetilde{\gamma_k} \times \eta_k \tag{2}$$

Subjected to

$$l.t.\rho_{jt} + bigm.(1-\rho_{jt}) > St_j \quad \forall j,t \tag{3}$$

$$C_j + bigm \cdot (1-\rho_{jt}) \geqslant l.t.\rho_{jt} \quad \forall j,t \tag{4}$$

$$\widehat{C}_{ij} + bigm \times (1-x_{ij}) \geqslant C_i \quad \forall i \neq j \tag{5}$$

$$C_i + bigm \times (1-x_{ij}) \geqslant \widehat{C}_{ij} \quad \forall i \neq j \tag{6}$$

$$\widehat{C}_{ij} \leqslant bigm \times x_{ij} \quad \forall i \neq j \tag{7}$$

$$St_j = \sum_{i \neq j} \widehat{C}_{ij} \quad \forall j \tag{8}$$

$$\pi + bigm \cdot \left(1 - \sum_{j=1}^{N} \rho_{jt}\right) \geqslant l.t \quad \forall t \tag{9}$$

$$l.t + bigm \cdot \sum_{j=1}^{N} \rho_{jt} \geqslant \pi \quad \forall t \tag{10}$$

$$C_{\max} \geqslant l. \sum_{t=1}^{T} \sum_{j=1}^{N} \widehat{\rho}_{jtk} \quad \forall k \tag{11}$$

$$B_{ijt_0} \leqslant l \times \widetilde{e^c_{ijk}} \times y_{ij} + \sum_{t=1}^{t_0} (l^2 \times \widetilde{e^r_{ijk}} \times \rho_{it}) + bigm \cdot (2 - \alpha_{ik} - \alpha_{jk}) \quad \forall i,j,k,t_0 \tag{12}$$

$$E_{jt} = \sum_{i=1}^{N} B_{ijt} \quad \forall j,t \tag{13}$$

$$R_{jt} \leqslant l \times \sum_{k=1}^{K} (\alpha_{jk} \times \lambda_{jk}) + E_{jt} \quad \forall j,t \tag{14}$$

$$R_{jt} \leqslant l \times \sum_{k=1}^{K} (\alpha_{jk} \times g^U_{jk}) \quad \forall j,t \tag{15}$$

$$\widehat{R}_{jt} + bigm \times (1-\rho_{jt}) \geqslant R_{jt} \quad \forall j,t \tag{16}$$

$$R_{jt} + bigm \times (1-\rho_{jt}) \geqslant \widehat{R}_{jt} \quad \forall j,t \tag{17}$$

$$\widehat{R}_{jt} \leqslant bigm \times \rho_{jt} \quad \forall j,t \tag{18}$$

$$1 - \frac{l \times \sum_{k=1}^{K} \left(\alpha_{jk} \times g^U_{jk}\right)}{2} < \sum_{t=1}^{T} \widehat{R}_{jt} \quad \forall j \tag{19}$$

$$1 + \frac{l \times \sum_{k=1}^{K} \left(\alpha_{jk} \times g^U_{jk}\right)}{2} \geqslant \sum_{t=1}^{T} \widehat{R}_{jt} \quad \forall j \tag{20}$$

$$x_{ij} + 1 \geqslant \rho_{it} + \rho_{j(t+1)} \quad \forall i \neq j, t < T \tag{21}$$

$$\sum_{i=1}^{N} \sum_{j \neq i}^{N} x_{ij} = N - 1 \tag{22}$$

$$y_{ij} \geqslant x_{ij} \quad \forall i \neq j \tag{23}$$

$$y_{ij} + 1 \geqslant y_{ik} + x_{kj} \quad \forall i \neq j, j \neq k, k \neq i \tag{24}$$

$$y_{ij} + y_{ji} = 1 \quad \forall i \neq j \tag{25}$$

$$\widehat{\rho}_{jtk} + bigm \times (1-\alpha_{jk}) \geqslant \rho_{jt} \quad \forall j,t,k \tag{26}$$

$$\sum_{k} \alpha_{ik} = 1 \quad \forall i \tag{27}$$

$$\sum_{i} \alpha_{ik} \leqslant bigm \times \eta_k \quad \forall k \tag{28}$$

Maximum completion time and machine hiring cost are minimized through Eqs. (1) and (2), respectively. When a job is started and not finished yet, so it is expected to be under execution at the time. This is formulated through Eqs. (3) and (4). Based on Eqs. (5)–(7), $\widehat{C}_{ij}$ equals to zero unless x_{ij} is one. In the case that, x_{ij} is one so $\widehat{C}_{ij}$ equals to C_i. Hence, $\widehat{C}_{ij}$ represents the product of C_i and x_{ij} as it is supposed to do. Furthermore, a job starts immediately after its previous job is done. Therefore, Eq. (8) is embedded to this aim. π is the dummy maximum completion time of jobs in the case that, they are scheduled to be processed on a single machine. In this regard, Eqs. (9) and (10) are employed to guide π toward feasible values. Based on Eq. (9), all jobs are not thoroughly completed when a job is still under execution. Due to Eq. (10) all jobs are done, when there is no job left to be processed. Eq. (11) is embedded to calculate the maximum completion time of jobs. Based on Eq. (11), the maximum of jobs' completion times equals to the maximum busy time of machines. A machine's busy time is calculated as the summation of the time units, in which the machine is occupied. Eqs. (3)–(11) are from the first category.

Learning and adapting abilities are formulated through Eqs. (12) and (13). Since different machines with different learning and adapting rates are available for processing, the total change made in jobs' completion rates depends on that which machine is in charge of processing each job. With this regard, α_{jk} is a newly-defined decision variable to find the optimum job/machine assignment choice. Based on Eq. (12), the amount of learning acquired as the result of processing job i is taken into account in calculating the total effect made on job j ($\widehat{B_{ijt_0}}$), if job j is going to be processed after job i on the same machine. Regarding to the adapting ability of its processor, total effect (i.e. learning and adapting) made on job j till t_0 is calculated through Eq. (13). The Eqs. (14) and (15) are applied in order to model the maturation of machines' learning and adapting abilities. $\widehat{R}_{jt}$ is calculated through Eqs. (16)–(18) in the same way as $\widehat{C}_{ij}$ is calculated through Eqs. (5)–(7). Hiring the same technique as it was applied in the previous work of Pakzad-Moghaddam et al. (2014), integral of the processing rate of job j over its execution interval is approximated to the linear term of $\sum_{t=1}^{T}\widehat{R}_{jt}$. Based on Eqs. (19) and (20), job j is completed if and only if, $\sum_{t=1}^{T}\widehat{R}_{jt}$ is approximately one. The second category contains Eqs. (12)–(20).

The third category of constraints are as follows. According to Eqs. (21) and (22), the processer does not switch from job i to j, unless there is a time (t) in which job i is under execution while job j is going to be processed afterwards. Based on Eqs. (23) and (24), job j is scheduled to get processed after job i, if job j is processed right after job i or there is a third job (say job k) which is scheduled to be processed after job i and before job j. According to the next equation, job j could not be processed before and after job i. $\widehat{\rho}_{jtk}$ is calculated via Eq. (26). As mentioned earlier, each job must be processed on exactly one machine which must be hired at the time. Eqs. (27) and (28) are applied to guarantee that no feasible solution violates these constraints.

4. Applied interactive possibilistic programming approach

Flexible and possibilistic mathematical programming are known as two major classes of fuzzy mathematical programming (Inuiguchi & Ramik, 2000). Flexible programming is applied to deal with vague and yet flexible problems (Kumar, Vrat, & Shankar, 2006; Liang, 2006b; Mula, Poler, & Garcia, 2006; Selim & Ozkarahan, in press). On the other hand, possibilistic programming provides the appropriate tools for treating ambiguous coefficients in objective functions and constraints (Hsu & Wang, 2001; Lai & Hwang, 1992; Lai & Hwang, 1993; Liang, 2006a; Wang & Liang, 2005). The degree of desired event occurrence are aimed to get maximized through possibilistic programming. This method of fuzzy mathematical programming takes advantage of the available historical data to determine possibility distributions objectively.

Due to the ambiguousness of $\widetilde{e^c_{ijk}}$, $\widetilde{e^r_{ijk}}$ and $\widetilde{\gamma_k}$ a possiblistic programming approach is applied. So far, a MOPMILP model is developed for the parallel machine scheduling problem with learning and adapting considerations. Torabi and Hassini (2008) proposed a two-phase approach to deal with such MOPMILP models. In the first phase, they converted their MOPMILP model into an equivalent auxiliary crisp multiple objective mixed integer linear model. Then an interactive fuzzy programming approach is developed in the second phase to deal with the crisp model. In this study, the same two-phase approach is applied to find a preferred compromise solution for the proposed parallel machine scheduling problem.

4.1. The auxiliary crisp multi-objective mixed integer model

The Machine Hiring Cost (MHC) makes an imprecise objective function which has a triangular possibility distribution as well as its imprecise coefficients (i.e. $\widetilde{\gamma_k}$). In this regard, it is geometrically defined by three following points: $(MHC^p, 0)$, $(MHC^m, 1)$ and $(MHC^o, 0)$. Lai and Hwang (1992) like many other researches attempted to minimize fuzzy objectives by pushing their three prominent points towards the left. They proposed three simultaneous crisp objectives in order to minimize the original imprecise one. As the result of adopting their approach, following three objectives are obtained.

$$\begin{aligned}\min\ \widetilde{MHC} &= \min(MHC^p, MHC^m, MHC^o)\\ \Rightarrow &\begin{cases}\min z_2 = MHC^m = \sum_{k=1}^{K}\gamma_k^m \times \eta_k\\ \max z_3 = MHC^m - MHC^p = \sum_{k=1}^{K}(\gamma_k^m - \gamma_k^p)\times\eta_k\\ \min z_4 = MHC^o - MHC^m = \sum_{k=1}^{K}(\gamma_k^o - \gamma_k^m)\times\eta_k\end{cases}\end{aligned} \tag{29}$$

Approaching three abovementioned objectives simultaneously, pushes the triangular equivalent of the original imprecise objective towards the left. Pushing it towards the left, make the fuzzy objective approaches to its optimum value.

With regard to the imprecise parameters, the weighted average method (Lai & Hwang, 1992; Liang, 2006a; Wang & Liang, 2005) is applied to convert these parameters into crisp numbers. Therefore the equivalent weighted average value for the imprecise parameters are as follows:

$$\begin{aligned}\widetilde{e^c_{ijk}} &= w^p.e^{c^p}_{ijk,\widehat{\beta}} + w^m.e^{c^m}_{ijk,\widehat{\beta}} + w^o.e^{c^o}_{ijk,\widehat{\beta}}\\ \widetilde{e^r_{ijk}} &= w^p.e^{r^p}_{ijk,\widehat{\beta}} + w^m.e^{r^m}_{ijk,\widehat{\beta}} + w^o.e^{r^o}_{ijk,\widehat{\beta}} \quad \forall i \neq j, k, t_0\end{aligned} \tag{30}$$

where $\widehat{\beta}$ is the minimum acceptable possibility or the minimum acceptable degree of feasibility. w^p, w^m and w^o are applied in order to represent relative importance weights of the most pessimistic, possible and optimistic values. It is up to the decision maker to determine appropriate values for these weights as well as $\widehat{\beta}$. Besides, Lai and Hwang (1992) denoted $(w^p, w^m, w^o, \widehat{\beta}) = (1/6, 4/6, 1/6, 0.5)$ as their most likely values which were also applied by several relevant studies (Liang, 2006a; Wang & Liang, 2005). Hereafter, considering the first objective as ($\min z_1 = c_{\max}$), following mathematical representation is concluded for the MOPMILP model.

$$\begin{aligned} &\min \quad z = [z_1, z_2, -z_3, z_4] \\ &s.t. \quad \forall \upsilon \in \Omega \end{aligned} \tag{31}$$

where the set of all feasible solutions satisfying Eqs. (3)–(28) is denoted by Ω. As well, υ is a representative for any feasible solution to the proposed MOPMILP model.

4.2. The interactive fuzzy programming solution approach

Fuzzy programming approaches are vastly applied to find efficient preferred solutions for multi-objective linear problems. Among single-phase fuzzy programming approaches, TH approach developed by Torabi and Hassini (2008), is applied to solve the proposed MOPMILP model. The TH method is chosen due to its ability in finding efficient and compromised solutions with generally balanced satisfaction degrees.

4.2.1. Finding ideal solutions

Positive and negative ideal solutions for z_2, z_3 and z_4 objectives are modified using the following equations.

$$z_2^{PIS} = \min(MHC^m) = \min(\gamma_k^m, \forall k),$$

$$z_2^{NIS} = \max(MHC^m) = \sum_{k=1}^{K} \gamma_k^m$$

$$z_3^{PIS} = \max(MHC^m - MHC^p) = \sum_{k=1}^{K} (\gamma_k^m - \gamma_k^p),$$

$$z_3^{NIS} = \min(MHC^m - MHC^p) = \min\left(\gamma_k^m - \gamma_k^p, \forall k\right)$$

$$z_4^{PIS} = \min(MHC^o - MHC^m) = \min(\gamma_k^o - \gamma_k^m, \forall k),$$

$$z_4^{NIS} = \max(MHC^o - MHC^m) = \sum_{k=1}^{K} (\gamma_k^o - \gamma_k^m)$$

The variable η_k must satisfy the following constraint in order to avoid the final solution jumping out of the feasible region $\left(\sum_{k=1}^{K} \eta_k \geqslant 1\right)$. Regarding this constraint, the single machine with lowest hiring cost is hired, in order to minimize only the machine hiring cost. In this case, since all jobs are processed sequentially on a single machine, they require a considerable amount of time to get fully processed. Hiring more machines results in higher machine hiring costs and lower maximum completion times. Hence, in order to minimize z_2, z_3 and z_4 the single machine with the smallest coefficient is hired. On the other hand, hiring all machines gives the maximum possible values for z_2, z_3 and z_4. In order to find the positive ideal solution concerning z_1 the following single objective model is proposed:

$$\begin{aligned} &z_1^{PIS} = \min C_{\max} \\ &s.t. \quad \forall \upsilon \in \Omega \end{aligned} \tag{32}$$

After defining z_1^{PIS} through solving the aforementioned model, its negative ideal solution is estimated as follows:

$$z_1^{NIS} = \max(c_{\max}) = \max\left(z_1(\upsilon_2^*), z_1(\upsilon_3^*), z_1(\upsilon_4^*)\right) \tag{33}$$

where $\upsilon_2^*, \upsilon_3^*$ and υ_4^* are the positive ideal solutions concerning z_2, z_3 and z_4 respectively.

4.2.2. Modeling linear membership functions concerning each objective

The degrees of satisfaction for both minimization and maximization objectives are calculated by applying the appropriate formulation adopted form the work of Torabi and Hassini (2008). The membership functions are schematically outlined through Figs. 1 and 2.

The linear mathematical formulation of the aforementioned degrees of satisfaction are as follows:

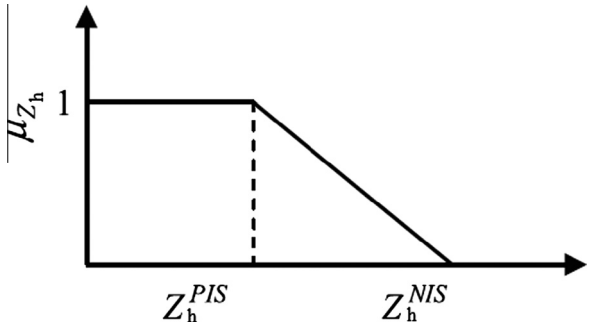

Fig. 1. The linear membership function for h-th objective, $h = 1, 2, 4$.

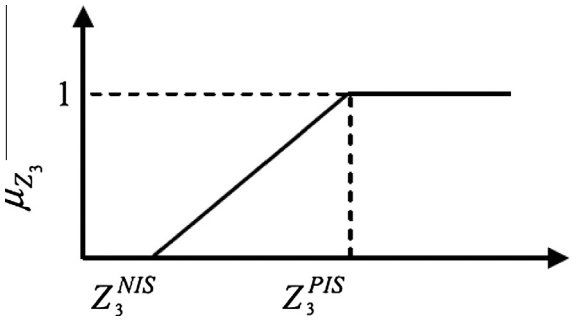

Fig. 2. The linear membership function for the third objective.

$$\mu_h(\upsilon) \leqslant \frac{z_h^{NIS} - z_h(\upsilon)}{z_h^{NIS} - z_h^{PIS}} \quad h = 1, 2, 3, 4 \tag{34}$$

4.2.3. Converting the auxiliary MOPMILP model into a single objective MILP one

In this step, the same technique as which is applied by Torabi and Hassini (2008), is employed to convert the proposed MOPMILP model into the following MILP one.

$$\begin{aligned} \max \quad & \lambda(\upsilon) = \widehat{\gamma} \cdot \lambda_0 + (1 - \widehat{\gamma}) \sum_{h=1}^{4} \theta_h \cdot \mu_h(\upsilon) \\ s.t. \quad & \lambda_0 \leqslant \mu_h(\upsilon) \leqslant \frac{z_h^{NIS} - z_h(\upsilon)}{z_h^{NIS} - z_h^{PIS}} \quad h = 1, 2, 3, 4 \\ & \upsilon \in \Omega \end{aligned} \tag{35}$$

$\widehat{\gamma}$ is proposed by Torabi and Hassini (2008), as a compensation coefficient which is applied to control the minimum satisfaction level of objectives as well as their compromise degree. θ_h denotes the relative importance weight of the h-th objective's degree of satisfaction which is defined regarding the experts' opinion. Analytic Hierarchy Process (AHP) technique is applied in this step to determine θ_h values.

4.3. The Fuzzy extended Analytic Hierarchy Process (FAHP)

In order to deal with the imprecise nature of the weighting procedure the linguistic variables are treated as fuzzy numbers. With this regard, a Fuzzy AHP approach is applied to this aim. Firstly, experts are asked to give their opinions by using the terms depicted in Table 1. Hence, the linguistic variables are set up to their primary values concerning the experts' opinions. Secondly, this information is applied to fill the comparison pairwise matrix which will lead us to the final relative importance weights (θ_h).

As well as many previous studies such as Kahraman, Cebeci, and Ruan (2004), Erensal, Oncan, and Demircan (2005), Bozbura,

Table 1
Triangular fuzzy conversion scale.

Linguistic scale		Triangular fuzzy scale	Triangular fuzzy reciprocal scale
Importance	Difficulty		
Just equal	Just equal	(1, 1, 1)	(1, 1, 1)
Equally Important (EI)	Equally Difficult (ED)	(1/2, 1, 3/2)	(2/3, 1, 2)
Weakly More Important (WMI)	Weakly More Difficult (WMD)	(1, 3/2, 2)	(1/2, 2/3, 1)
Strongly More Important (SMI)	Strongly More Difficult (SMD)	(3/2, 2, 5/2)	(2/5, 1/2, 2/3)
Very Strongly More Important (VSMI)	Very Strongly More Difficult (VSMD)	(2, 5/2, 3)	(1/3, 2/5, 1/2)
Absolutely More Important (AMI)	Absolutely More Difficult (AMD)	(5/2, 3, 7/2)	(2/7, 1/3, 2/5)

Beskese, and Kahraman (2007) and Celik (2009) the theoretical basics of Chang's extent analysis on FAHP is hired in the current study, to determine the relative importance weights.

5. A Lévy flight embedded particle swarm optimization

Firstly, the under study optimization problem is formulated into a fuzzy multi-objective mathematical model. Secondly the MOPMILP model with imprecise parameters is converted into an equivalent crisp mixed integer linear model through an interactive possibilistic programming approach. Although the original MOPMILP and its auxiliary MILP models both are linearized to avoid having non-linear terms, they are accounted as NP-hard mathematical models.

The problem of minimizing make-span with a learning effect on parallel machines is of NP-hard optimization problems (Mosheiov, 2001b) which are not expected to be solved in a reasonable amount of time by the means of exact methods. Hence, the uniform parallel machine scheduling problem at hand, which is extended to have adapting abilities as well as cost considerations, is definitely NP-hard too. Regarding the NP-hardness of the proposed parallel machine scheduling problem, a highly efficient and well-organized meta-heuristic optimization method, called Lévy Flight Embedded Particle Swarm Optimization is developed.

PSO is a swarm intelligence-based optimization method, in which the social behavior of particles (such as birds and fishes) in approaching to their desired goals is simulated. The algorithm borrows its intelligence from the social behavior of bird flocking and fish schooling (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995). Regarding the semi-stochastic birds' movements in search for better places, PSO also takes advantage of a semi-stochastic search mechanism. Particles' movement pattern is simulated in PSO via the following mathematical formulae over particles' positions and their velocities.

$$V_{id}(t+1) = V_{id}(t) + c_1 r_1 \left(x_{id}^{pbest}(t) - x_{id}(t)\right) + c_2 r_2 \left(x_{d}^{gbest}(t) - x_{id}(t)\right); \quad (r_1, r_2) \in Uniform[0\ 1] \tag{36}$$

$$x_{id}(t+1) = x_{id}(t) + V_{id}(t+1) \tag{37}$$

where $x_{id}(t)$ and $V_{id}(t)$ denote position and velocity of the i-th particle in time t concerning its d-th dimension. c_1 and c_2 represent personal and social awareness of particles respectively. The stochastic nature of PSO derives from r_1 and r_2 parameters which guarantee the randomness of particles' walk towards their personal and global best experiences. Getting trapped in local optima is known as the major deficiency of the basic PSO. Hence, Shi and Eberhart (1998) added the inertia weight to Eq. (36) to control the balance between the algorithm's exploration and exploitation.

$$V_{id}(t+1) = w_t \times V_{id}(t) + c_1 r_1 \left(x_{id}^{pbest}(t) - x_{id}(t)\right) + c_2 r_2 \left(x_{d}^{gbest}(t) - x_{id}(t)\right) \tag{38}$$

where w_t denotes the particles' inertia weight in time t. However regarding the insufficient diversity of population, PSO still faces the threat of getting stuck in local optima while dealing with complex optimization problems. Therefore, improving the convergence behavior of PSO has been of particular interest in many recent studies such as (Jana & Sil, 2014; Haklı & Uğuz, 2014; Higashi & Iba, 2003; Tang & Zhao, 2009; Wang, Liu, Li, & Zeng, 2007; Wu & Zhong, 2009). They attempted to avoid getting trapped in local optima by applying various mutation strategies. To this aim in the current study, particles are supposed to do their random walks towards their personal and global best experiences using Lévy flights. The main idea in PSO is to guide solutions towards optimum values as well as birds who take advantage of their natural instinct to get to their desired destination. With this regard, the PSO algorithm including its velocity and position updating equations is inspired by the intelligent social behavior of bird flocking. However despite particles in PSO, getting trapped in local optima is not the major concern for real birds along the way to their destination. What does cause the algorithm to act inferior compared to its original real-world inspiration? And what sometimes does hold the algorithm back from fulfilling the purpose that it's been designed to achieve at the first place? Do the birds actually take uniformly distributed random walks? "On the other hand, various studies have shown that flight behavior of many animals and insects has demonstrated the typical characteristics of Lévy flights" (Brown, Liebovitch, & Glendon, 2007; Pavlyukevich, 2007a; Pavlyukevich, 2007b; Reynolds & Frye, 2007; Yang & Deb, 2009). Embedding Lévy flights enables PSO to simulate birds' movement pattern much more precisely than the basic one. Therefore it come to mind that applying Lévy flights instead of uniformly distributed walks might be a great step towards solving the local optima problem. As well as, previous studies such as Lee and Yao (2004) and Richer and Blackwell (2006), in which Lévy walks are adopted in order to escape from a local optimum. "This might improve algorithm performance by allowing escape from a local optimum, and the adoption of Lévy mutation in evolutionary programming has been shown to be advantageous" (Lee & Yao, 2004). Richer and Blackwell (2006), also attempted to avoid particles getting trapped in local optima by replacing the uniform distributions used to determine the attraction of each particle to the local and global bests with a Lévy distribution.

Basically Lévy flight is a Markov chain in which the steps are drawn randomly from a Lévy distribution (Yang, 2010). Firstly, an efficient method is required to approximate Lévy flights using the Lévy distribution. The Mantegna algorithm has been found the most efficient method to generate Lévy distributed values among those were tested by Leccardi (2005). Mantegna implemented Lévy flights using random noises which are drawn from a symmetric Lévy stable distribution (Leccardi, 2005). Therefore in the current study, the Mantegna algorithm is applied to generate Lévy flights (Mantegna, 1994). Applying Eq. (39) is a common way to model birds' movement pattern via Lévy flights. For instance, Kanagaraj, Ponnambalam, and Jawahar (2013) also used Eq. (39) to generate new solutions by performing Lévy flights.

$$V_{id}(t+1) = \alpha_d \times Levy(\beta); x_{id}(t+1) = x_{id}(t) + V_{id}(t+1)$$
$$\Rightarrow x_{id}(t+1) = x_{id}(t) + \alpha_d \times Levy(\beta) \quad (39)$$

where α_d denotes the bird's step size regarding d-th dimension. In order to enable particles to use their personal and social awareness in taking Lévy distributed random walks, Eqs. (38) and (39) are combined to make Eq. (40).

$$V_{id}(t+1) = w_t \times V_{id}(t) + c_1 \times Levy(\beta_1) \times (x_{id}^{pbest}(t) - x_{id}(t)) + c_2 \times Levy(\beta_2) \times (x_d^{gbest}(t) - x_{id}(t)) \quad (40)$$

$$Levy(\beta_i) \sim \frac{u_i}{|v_i|^{\frac{1}{\beta_i}}} \quad (41)$$

Applying Eq. (40) leads particles to take Lévy steps, proportional to their distances from their personal and global best experiences. An approach similar to the one applied by Yang and Deb (2011) is employed in LFEPSO to take Lévy distributed steps towards best known solutions. u_i and v_i are drawn from normal distributions as follows.

$$u_i \sim N(0, \sigma_{u_i}^2) \text{ and } v_i \sim N\left(0, \sigma_{v_i}^2\right)$$
$$\sigma_{u_i} = \left\{ \frac{\Gamma(1+\beta_i)\sin\left(\frac{\pi\beta_i}{2}\right)}{\Gamma\left[\frac{(1+\beta_i)}{2}\right]\beta 2^{\frac{\beta_i-1}{2}}} \right\}^{1/\beta_i} \quad \text{and} \quad \sigma_{v_i} = 1 \quad (42)$$

Here Γ is the standard Gamma function and distribution parameter β_i belongs to [0.3 1.99]. Applying Eqs. (40)–(42) makes a great progress toward accurately modeling the original inspiration (i.e. social behavior and movement pattern of bird flocking).

5.1. Solution structure

A feasible solution to the proposed MILP model contains of job schedules on parallel machines. Regarding a random solution, some machines may not be utilized at all. A permutation of $N+M$ is applied as follows in order to represent any feasible solution to the problem.

υ:	x_1	x_2	…	x_{N+M-1}	x_{N+M}

x_r denotes the r-th number in the solution, which is either a representative for a job or a machine. Definition of x_r is as follows:

$$x_r = \begin{cases} job\ r & x_r = r \Rightarrow 1 \leqslant x_r \leqslant N \\ machine\ r & x_r = N + r \Rightarrow N < x_r \leqslant N + M \end{cases} \quad x_r = 1, 2, \ldots, N+M$$

Jobs which are placed between two machines (say, $x_{\widehat{r}}$ and x_r) are scheduled to be processed on the later one (x_r) in the same order as they are following in the solution. In other words, a sequence of jobs is scheduled to be processed on the closest machine placed after the sequence. For example consider 8 jobs which are about to get processed on a subset of 3 uniform parallel machines. A possible solution to this problem is as follows:

υ:	6	1	7	4	9	10	8	3	2	5	11

where jobs are denoted by their numbers while machines are shown by their numbers plus 8 (9, 10 and 11). Therefore, the first machine is employed to perform the following sequence of jobs, 6-1-7-4 as well as the third machine which is hired to execute jobs 8-3-2-5. Since no job is assigned to the second machine, it is not required to be hired.

Since all jobs are needed to be processed on exactly one machine, last position of a feasible solution must be always occupied by a machine. In this regard and without the loss of generality, last position of a feasible solution is supposed to get a fixed value of $M+N$. In other words, always $x_{M+N} = M+N$. Consequently the solution size is reduced to $(M+N-1)$ positions which should be occupied by a permutation of $(M+N-1)$.

5.2. Feasibility step

Updating the position of a particle using Eqs. (40) and (37), disturbs its feasibility. In fact, updated particles usually contain a lot of decimal numbers which are not acceptable as a representative for neither a job nor a machine. In order to deal with these decimal numbers a feasibility step is applied as follows:

$$x_{\widehat{r}}(\upsilon') = x_r(\upsilon) \Rightarrow x_r(\upsilon'') = \widehat{r} \quad (43)$$

where υ is the initial infeasible solution, containing decimal numbers. Firstly υ is sorted into υ', in which $\widehat{r} < r \Rightarrow x_{\widehat{r}}(\upsilon') < x_r(\upsilon')$. Secondly Eq. (43) is applied in order to find the feasible counterpart (υ'') of the initial solution υ. For instance consider the following initial solution.

υ:	2.412	5.318	6.215	4.915	0.597	6.617	8.363	4.902

υ' is formed as follows by sorting the initial solution.

υ':	0.597	2.412	4.902	4.915	5.318	6.215	6.617	8.363
$[\upsilon']$:	1	2	3	4	5	6	7	8

where $[\upsilon']$ is applied to demonstrate the position of decimal numbers in sorted solution υ'. Finally the initial infeasible solution is converted to the following feasible one:

υ'':	2	5	6	4	1	7	8	3

The proposed LFEPSO is summarized into the following Pseudo code (see Fig. 3).

5.3. Fitness evaluation

As mentioned before, Pakzad-Moghaddam et al. (2014) applied an analytical integration to calculate the maximum completion time in scheduling environments with adapting/lingering considerations. Following calculations are presented to solve the proposed integral analytically. The same integral is formulated in the mathematical model but in a different way. A discrete approach is applied in the mathematical formulation section in order to solve the same integral numerically. Hence, the numerical integration which is formulated in Section 3.5 is the same as the integral equation which is going to get solved in this section analytically. Pakzad-Moghaddam et al. (2014), formulated special equations to calculate the total effect made on job j at time t, where the processor is capable of adapting as well as learning. Regarding the under-study uniform parallel machine scheduling problem, those equations are adopted and reformulated as follows.

$$e_{lj}^T(t) = \sum_k \left(\alpha_{jk} \times \alpha_{lk} \times \left[\widetilde{e_{ljk}^c} + p_l^a(t) \times \widetilde{e_{ljk}^r} \right] \right) \quad j, l = 1, 2, \ldots, n \quad (44)$$

where $p_l^a(t)$ represents the actual time which is spent to process job l, where the current time equals to t. e_{lj}^T represents the total effect

1. Load original parallel machine scheduling problem's data.
2. Assign values to the parameters of the LFEPSO algorithm.
3. Initialize the algorithm as follows.
   3.1. Create the initial swarm containing permutations of M+N-1.
   # where M is the number of machines and N is the number of jobs.
   3.2. Sort and evaluate the swarm.
   3.3. Initialize particles' personal best position which is where they currently are.
   3.4. Call the position of the best particle, the global best.
4. Repeat until the stop condition is met (main loop).
   4.1. Regarding Eq. (40), update particles' velocity vectors, taking Lévy walks around their personal best positions and the current global best.
   4.2. According to Eq. (37), update particles' position, using their updated velocity vectors.
   4.3. Apply the feasibility step to move particles to the feasible region.
   4.4. Sort and evaluate the swarm.
   4.5. Update particles' personal best positions.
   4.6. Use the best position obtained so far as the new global best.

Fig. 3. Pseudo code of the proposed LFEPSO.

made on job j as the result of processing job l. α_{jk}, $\widetilde{e^c_{ljk}}$ and $\widetilde{e^r_{ljk}}$ are borrowed from the mathematical model's notations.

$$CE_j = \sum_{J_l=1}^{J_j} e^T_{lj}(St_j) \quad j,l = 1,2,\ldots,n \tag{45}$$

$$VE_j(t) = \sum_k \alpha_{jk} \times \widetilde{e^r_{jjk}} \times (t - St_j) \quad j = 1,2,\ldots,n, t \geqslant St_j \tag{46}$$

$$E_j(t) = CE_j + VE_j(t) \quad j = 1,2,\ldots,n, t \geqslant St_j \tag{47}$$

$CE_j, VE_j(t)$ and $E_j(t)$ are the constant, variable and total effects made respectively on the processing rate of job j at time t (Pakzad-Moghaddam et al., 2014). t indicates the current time which is supposed to be greater than the start time of job $j(St_j)$. In other words, Eqs. (46) and (47) are applied when job j is already started. Applying a continuous approach, Eqs. (44)–(47) do the same job as Eqs. (12) and (13), which must be applied where a discrete approach is hired to calculate the integral numerically. Considering Eqs. (12) and (13), $p^a_l(t)$ is approximated to $\sum_{t_0=1}^{t} \rho_{lt_0}$ as the result of using the discrete approach. In addition, $E_j(t)$ denotes the total effect made on the processing rate of job j at time t which increases constantly as time goes on. Similarly, E_{jt} is a notation which has been applied in the mathematical formulation section. As well, E_{jt} demonstrates the total effect made on the processing rate of job j at time step t which increases step by step while time step (i.e. t) is increasing.

Eq. (48) is also adopted from the previous work of Pakzad-Moghaddam et al. (2014), and reformulated to model the learning/adapting maturation limits for the proposed parallel machine scheduling problem.

$$R_j(t) = \min\left\{\sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j + VE_j(t), \sum_k \left(\alpha_{jk} \times g^U_{jk}\right)\right\} \times j = 1,2,\ldots,n \tag{48}$$

where $R_j(t)$ represents the maturated processing rate of job j. λ_{jk} and g^U_{jk} are the normal processing rate and maturation limit of job j on machine k. Eq. (48) is considered as the continuous and nonlinear equivalent of Eqs. (14) and (15). As well, $R_j(t)$ and R_{jt} are both representatives for the maturated processing rates while $R_j(t)$ is free to change constantly as R_{jt} changes step by step. Afterwards, in order to calculate job's completion times Eq. (49) is applied.

$$\int_{St_j}^{C_j} R_j(t) \times dt = 1 \tag{49}$$

Please note that Eq. (49) (i.e. the analytic function) is approximated to Eqs. (19) and (20) to make it possible to solve the integral equation numerically. Hereafter, the following procedure is applied to calculate the completion time of job $j(C_j)$, based on Eq. (49). Eq. (50) is proposed to anticipate the maturation time of job j.

$$\sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j + VE_j\left(\widehat{t}_j\right) = \sum_k \left(\alpha_{jk} \times g^U_{jk}\right)$$
$$\Rightarrow \widehat{t}_j = \frac{\sum_k \left(\alpha_{jk} \times g^U_{jk}\right) - \left[\sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j\right]}{\sum_k \left(\alpha_{jk} \times e^r_{jjk}\right)} \quad j = 1,2,\ldots,n \tag{50}$$

Based on Eq. (50), the processing rate of job j will reach its maturation limit if the operator continues to execute the job for $\widehat{t}_j$ units of time (e.g. hours). On the other hand, the following equation is applied to approximate the amount of time required to execute job j, where its maturation limit is neglected $\left(t'_j\right)$.

$$\int_{St_j}^{C_j} \left[\sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j + \sum_k (\alpha_{jk} \times \widetilde{e^r_{jjk}} \times (t - St_j))\right] \times dt$$
$$= 1 \overset{t'_j = C_j - St_j / t'' = t - St_j}{\rightarrow} \int_0^{t'_j} \left[\sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j + \sum_k \left(\alpha_{jk} \times \widetilde{e^r_{jjk}} \times t''\right)\right] \times dt''$$
$$= 1 \overset{r^c_j = \sum_k (\alpha_{jk} \times \lambda_{jk}) + CE_j \,\&\, r^v_j = \sum_k \left(\alpha_{jk} \times \widetilde{e^r_{jjk}}\right)}{\rightarrow} r^c_j \times t'_j + r^v_j \times \left(t'^2_j/2\right) = 1$$
$$j = 1,2,\ldots,n \tag{51}$$

Hence, t'_j calculates as follows.

$$t'_j = \frac{\sqrt{\left(r^c_j\right)^2 + 2r^v_j} - r^c_j}{r^v_j} \quad j = 1,2,\ldots,n \tag{52}$$

Eq. (53) is applied in order to calculate t^{real}_j, based on the calculated values for $\widehat{t}_j$ and t'_j.

$$t^{real}_j = \begin{cases} t'_j & t'_j \leqslant \widehat{t}_j \\ \dfrac{1 - \left(r^c_j \times \widehat{t}_j + r^v_j \times \widehat{t}^2_j/2\right)}{\sum_k \left(\alpha_{jk} \times g^U_{jk}\right)} + \widehat{t}_j & t'_j > \widehat{t}_j \end{cases} \quad j = 1,2,\ldots,n \tag{53}$$

where t^{real}_j denotes the actual time required to execute job j considering its maturation limit. The two following equations are auxiliary

equivalents for Eq. (53). Eqs. (54) and (55) are much easier than Eq. (53) to implement.

$$t_j^* = \max\left(0, \min\left(t'_j, \widehat{t}_j\right)\right) \quad j = 1,2,\ldots,n \tag{54}$$

$$t_j^{real} = \frac{1 - \left(r_j^c \times t_j^* + r_j^v \times t_j^{*2}/2\right)}{\sum_k \left(\alpha_{jk} \times g_{jk}^U\right)} + t_j^* \quad j = 1,2,\ldots,n \tag{55}$$

Finally the precise completion time for job j, calculates as follows.

$$C_j = St_j + t_j^{real} \quad j = 1,2,\ldots,n \tag{56}$$

Start and completion times will be computed sequentially using Eqs. (50), (52), (54)–(56). While z_1 is the maximum completion time among all jobs, Eq. (34) is employed to find the degree of satisfaction for z_1 i.e. $\mu_1(\upsilon)$. At last, Eqs. (29) and (34) are applied in order to calculate z_2, z_3 and z_4 objective values and their corresponding degrees of satisfaction.

6. Experimental results

6.1. Random data set generation

Various small to large-sized test problems are generated in order to investigate the validity of the proposed MOPMILP model and its auxiliary MILP one. The probabilistic distributions, applied to generate random test problems are depicted in Table 2. The symmetrical triangular possibility distribution which was firstly applied by Selim and Ozkarahan (in press) is employed in this study to generate fuzzy numbers for the parameters which are recognized imprecise in nature. In this regard, the most possible values for such parameters are generated using an appropriate probability distribution. Then the generated most possible values are multiplied with 0.8 and 1.2 to produce the corresponding most pessimistic and optimistic values, respectively.

6.2. Solving a small-sized instance via GAMS IDE/CPLEX, PSO and LFEPSO

Sub-section 6.2 is an illustration section, which is prepared to show the problem data, the stage by stage solutions and final answers. True purpose of this section is to clarify the presented problem and methodologies regarding their complex nature.

6.2.1. Sample problem's data

An instance with 5 jobs and 2 machines is generated to test the validity of the presented mathematical model and the performance of the proposed meta-heuristic methods. Problem's data are given in Tables 3 and 4.

6.2.2. Implementation of the modeling procedure and the possibilistic programming approach

Positive and negative ideal solutions for $h = 2$, 3 and 4 and their corresponding objective values are calculated as follows.

Then, the mathematical model which is presented in Eq. (32) is solved via GAMS in order to find the positive ideal solution concerning the maximum completion time. In order to implement the numerical integration, the time step (i.e. l) is decided to be 0.5. Obtained results are depicted in Table 6.

After finding the exact optimum solution (including the optimum job to machine assignment and schedules) the fitness evaluation procedure which is described through sub-section 5.3 is also applied to calculate the maximum completion time precisely. The precise and analytically calculated start and completion times are as follows (see Table 7).

For example fitness evaluation procedure for calculating C_3, is described in details. First, CE_3 is calculated using Eqs. (44) and (45). After defining the value of CE_3 which equals to 4.6533, Eq. (50) is employed to find that $\widehat{t}_3 = -2.2863$. Afterwards, r_3^c, r_3^v and t'_3 are calculated using Eqs. (51) and (52). They respectively equal to 4.8314, 1.9028 and 0.1992. In order to calculate t_3^*, Eq. (54) is implemented as follows.

$$t_3^* = \max\left(0, \min\left(t'_3, \widehat{t}_3\right)\right) = \max(0, \min(0.1992, -2.2863)) = \max(0, -2.2863) = 0 \tag{57}$$

Applying Eq. (55), t_3^{real} equals to 2.079. According to Eq. (56), C_3 calculates as follows.

$$C_3 = 7.6224 + 2.079 = 9.7014 \tag{58}$$

Which is the minimum possible value for the exact maximum completion time of all jobs. Based on Table 5, the numerical integral method approximates this maximum completion time to $C^{\max} = 20 \times l = 10$ using time step ($l = 0.5$). It should be noted that, time step is elaborately assumed to be 0.5 in order to illustrate the approximation. But usually much smaller time steps are applied in order to increase the accuracy of the exact results. Also the negative ideal value for the first objective calculates using Eq. (33). So z_1^{NIS} equals to 14.4444. Based on Eq. (34) the satisfaction degrees formulations are as follows.

$$\mu_1(\upsilon) \leqslant \frac{14.444 - z_1(\upsilon)}{4.743}; \quad \mu_2(\upsilon) \leqslant \frac{15.921 - z_2(\upsilon)}{10.242}$$
$$\mu_3(\upsilon) \leqslant \frac{z_3(\upsilon) - 1.135}{2.049}; \quad \mu_4(\upsilon) \leqslant \frac{3.184 - z_4(\upsilon)}{2.049} \tag{59}$$

Table 2
Probabilistic distributions which are applied to generate the random data set.

Parameter	γ_k^m	λ_{jk}	$e_{ijk}^{c^m} \mid i \neq j$	$e_{iik}^{c^m}$	$e_{ijk}^{r^m} \mid i \neq j$	$e_{iik}^{r^m}$	$g_{jk}^{U^m}$
Distribution	Uniform (5,11)	Uniform (0.05,0.45)	Uniform (0.6,1.8)	Uniform (2.4,7.2)	Uniform (0.3,0.55)	Uniform (1.2,2.2)	Uniform (0.2,0.6)

Table 3
Values of λ_{jk}, g_{jk}^U and $\widetilde{\gamma_k}$'s fuzzy triangular numbers for the sample problem.

i	$k = 1$					$k = 2$				
	λ_{jk}	g_{jk}^U	γ_k^p	γ_k^m	γ_k^o	λ_{jk}	g_{jk}^U	γ_k^p	γ_k^m	γ_k^o
1	0.135	0.471	8.193	10.241	12.289	0.18	0.448	4.544	5.68	6.816
2	0.124	0.499				0.18	0.485			
3	0.156	0.416				0.178	0.481			
4	0.161	0.411				0.135	0.419			
5	0.13	0.491				0.105	0.425			

Table 4
Values of fuzzy triangular numbers for $\widetilde{e^c_{ijk}}$ and $\widetilde{e^r_{ijk}}$ regarding the sample problem.

i	j	$k=1$						$k=2$					
		$e^{c^p}_{ijk,0.5}$	$e^{c^m}_{ijk,0.5}$	$e^{c^o}_{ijk,0.5}$	$e^{r^p}_{ijk,0.5}$	$e^{r^m}_{ijk,0.5}$	$e^{r^o}_{ijk,0.5}$	$e^{c^p}_{ijk,0.5}$	$e^{c^m}_{ijk,0.5}$	$e^{c^o}_{ijk,0.5}$	$e^{r^p}_{ijk,0.5}$	$e^{r^m}_{ijk,0.5}$	$e^{r^o}_{ijk,0.5}$
1	1	2.13	2.66	3.19	1.594	1.992	2.39	2.61	3.26	3.91	1.498	1.872	2.247
1	2	1.43	1.79	2.15	0.435	0.544	0.653	1.43	1.78	2.14	0.418	0.522	0.627
1	3	1.03	1.29	1.55	0.262	0.328	0.393	0.49	0.62	0.74	0.323	0.403	0.484
1	4	1.3	1.62	1.94	0.346	0.433	0.52	0.89	1.12	1.34	0.253	0.316	0.379
1	5	0.96	1.2	1.44	0.309	0.387	0.464	1.37	1.71	2.05	0.319	0.398	0.478
2	1	1.06	1.33	1.6	0.33	0.413	0.496	1.01	1.26	1.51	0.42	0.525	0.631
2	2	3.84	4.8	5.77	1.379	1.723	2.068	5.32	6.65	7.98	1.361	1.701	2.042
2	3	1.2	1.5	1.8	0.315	0.394	0.473	0.93	1.17	1.4	0.243	0.304	0.364
2	4	0.8	1.01	1.21	0.344	0.43	0.516	1.28	1.6	1.92	0.312	0.39	0.468
2	5	1	1.25	1.49	0.307	0.384	0.46	1.19	1.49	1.79	0.251	0.313	0.376
3	1	1.23	1.53	1.84	0.41	0.512	0.615	1.4	1.75	2.1	0.28	0.35	0.42
3	2	0.8	1	1.2	0.326	0.407	0.489	0.69	0.86	1.03	0.295	0.369	0.443
3	3	2.51	3.14	3.76	1.224	1.53	1.836	5.57	6.97	8.36	1.522	1.903	2.283
3	4	0.74	0.93	1.12	0.395	0.494	0.592	1.27	1.59	1.91	0.287	0.359	0.43
3	5	1.32	1.65	1.97	0.355	0.444	0.532	1.19	1.49	1.78	0.315	0.394	0.473
4	1	0.97	1.21	1.46	0.318	0.398	0.477	1.05	1.32	1.58	0.3	0.375	0.449
4	2	0.65	0.81	0.97	0.281	0.352	0.422	0.51	0.64	0.77	0.347	0.433	0.52
4	3	0.82	1.03	1.23	0.308	0.386	0.463	0.72	0.9	1.08	0.341	0.427	0.512
4	4	1.94	2.43	2.91	1.056	1.32	1.584	3.66	4.57	5.49	1.123	1.404	1.684
4	5	1.17	1.47	1.76	0.413	0.516	0.619	1.39	1.74	2.08	0.395	0.494	0.592
5	1	0.51	0.63	0.76	0.388	0.485	0.582	1.26	1.57	1.88	0.339	0.424	0.509
5	2	1.08	1.35	1.62	0.305	0.381	0.457	0.91	1.14	1.37	0.355	0.444	0.532
5	3	0.62	0.77	0.93	0.403	0.504	0.605	0.85	1.06	1.28	0.316	0.395	0.474
5	4	1.25	1.56	1.87	0.365	0.456	0.548	0.85	1.06	1.27	0.403	0.503	0.604
5	5	4.49	5.61	6.73	1.119	1.399	1.678	3.88	4.85	5.82	1.092	1.365	1.638

Table 5
Positive and negative ideal solutions for $h = 2$, 3 and 4.

h	PIS	NIS
2	5.679	15.921
3	3.184	1.135
4	1.135	3.184

Regarding the expert's opinion, relative importance/weights of the degrees of satisfaction are calculated using a FAHP method. Remainder of this sub-section is devoted to present the step by step implementation procedure of the FAHP method. In order to weight the proposed objective functions (i.e. time and cost objectives) a FAHP method is applied. Steps of the applied FAHP method adopted from the theoretical basics of Chang's extent analysis are described in sub-section 4.3. The method is implemented through the following example. The Hypothetical comparison pairwise matrix is given in Table 8. The matrix is established regarding the experts' opinions and is extended to contain $\sum_{j=1}^{4} M^j_{g_i}$ in its last column.

Hiring the FAHP method adopted from the previous work of Celik (2009),θ_h values are calculated as follows.

$$\left[\sum_{i=1}^{4}\sum_{j=1}^{4} M^j_{g_i}\right]^{-1} = \left(\frac{1}{23.5}, \frac{1}{16.497}, \frac{1}{12.001}\right) \tag{60}$$

Obtained values for the fuzzy synthetic extents are depicted through Table 9.

Table 10 is employed to demonstrate the concluded degrees of possibility and objectives' relative importance weights.

Table 7
start and completion time in z_1^{PIS} computed by applying the analytical integration procedure.

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
St_j	0	5.5613	7.6224	0	6.2007
C_j	5.5613	7.6224	9.7014	6.2007	8.2361

Hereafter, the concluded normalized relative importance weights depicted in the last column of Table 10, are applied as θ_h values in solving the mathematical model presented in Eq. (35). As it was advised by Torabi and Hassini (2008), the compensation coefficient is 0.4 ($\widehat{\gamma} = 0.4$). Finally the auxiliary crisp mathematical model is formulated based on Eq. (35).

$$\begin{aligned}
\max\quad & \lambda(\upsilon) = 0.4\lambda_0 + 0.174\mu_1(\upsilon) + 0.162\mu_2(\upsilon) \\
& \quad + 0.126\mu_3(\upsilon) + 0.138\mu_4(\upsilon) \\
s.t.\quad & \lambda_0 \leqslant \mu_1(\upsilon) \leqslant \frac{14.444 - z_1(\upsilon)}{4.743}; \quad \lambda_0 \leqslant \mu_2(\upsilon) \leqslant \frac{15.921 - z_2(\upsilon)}{10.242} \\
& \lambda_0 \leqslant \mu_3(\upsilon) \leqslant \frac{z_3(\upsilon) - 1.135}{2.049}; \quad \lambda_0 \leqslant \mu_4(\upsilon) \leqslant \frac{3.184 - z_4(\upsilon)}{2.049} \\
& \upsilon \in \Omega
\end{aligned} \tag{61}$$

6.2.3. Optimum solution

Solving the aforementioned crisp model, leads again to the positive ideal solution concerning the first objective (z_1^{PIS}). For

Table 6
z_1^{PIS} found by GAMS IDE/CPLEX.

$j=1$				$j=2$				$j=3$				$j=4$				$j=5$			
Processor	Position	St_j	C_j	Processor	Position	St_j	C_j	Processor	Position	St_j	C_j	Processor	Position	St_j	C_j	Processor	Position	St_j	C_j
2	1	0	12	2	2	12	16	2	3	16	20	1	1	0	13	1	2	13	17

Table 8
Extended comparison pairwise matrix.

Objective/criteria	μ_1	μ_2	μ_3	μ_4	$\sum_{\hat{h}=1}^{4} M_{g_h}^{\hat{h}}$
μ_1	(1,1,1)	(1/2,1,3/2)	(1,3/2,2)	(1,3/2,2)	(3.5,5,6.5)
μ_2	(2/3,1,2)	(1,1,1)	(1,3/2,2)	(1/2,1,3/2)	(3.167,4.5,6.5)
μ_3	(1/2,2/3,1)	(2/3,1,2)	(1,1,1)	(2/3,1,2)	(2.667,3.33,5)
μ_4	(1/2,2/3,1)	(1/2,2/3,1)	(1/2,1,3/2)	(1,1,1)	(3.167,3.667,5.5)

Table 9
Fuzzy synthetic extents computed regarding the objective functions.

Fuzzy synthetic extents	S_1	S_2	S_3	S_4
Triangular fuzzy numbers	(0.149,0.303,0.542)	(0.135,0.273,0.542)	(0.113,0.202,0.417)	(0.113,0.222,0.458)

Table 10
Degrees of possibility, initial and normalized objectives' relative importance weights.

S_h	$S_{\hat{h}}$					
	S_1	S_2	S_3	S_4	$d'(\mu_h)$	$\theta_h = d(\mu_h)$
S_1	1	1	1	1	1	0.2901
S_2	0.929	1	1	1	0.929	0.2695
S_3	0.726	0.799	1	0.938	0.726	0.2106
S_4	0.792	0.864	1	1	0.792	0.2298
Formulae	$V(S_h \geqslant S_{\hat{h}})$				$d'(\mu_h) = Min\left(V\left(s_h \geqslant s_{\hat{h}}\right)\right)$	$\frac{d'(\mu_h)}{\sum_{\hat{h}} d'\left(\mu_{\hat{h}}\right)}$

Table 11
Start and completion times regarding z_1^{PIS}, which are computed by applying the analytical integration procedure.

Solution	4	5	6	1	2	3	7
Decoded solution	$j=4$	$j=5$	$k=1$	$j=1$	$j=2$	$j=3$	$k=2$

example, optimum solution found by both meta-heuristics is presented in Table 11.

Due to Table 11, first machine is responsible to process jobs 4-5 as well as the second machine which is decided to process jobs 1-2-3 right in this order. Hence, the optimum results are as follows (see Table 12).

Al three solvers have accomplished to find the aforementioned solution but with different computational times. LFEPSO was the fastest which converged to the optimum solution in 0.502 s. PSO and GAMS also solved the problem in 0.898 and 1.114 s respectively. In order to increase the accuracy of exact results, optimum solutions which are found by the exact solver are evaluated again using the analytical integration evaluation procedure.

6.3. GAMS IDE/CPLEX versus PSO and LFEPSO

In order to test the validity of the proposed mathematical model and the performance of the designed optimization methods (i.e. PSO and LFEPSO), twenty-six instances are generated. Attributes of the aforementioned instances and their optimization results are shown in Table 13.

In the randomly-generated test problems, a variety of 8–500 jobs are executed on a wide range of machines. The number of parallel processors varies from 3 to 50 machines. Objective values (i.e. λ) obtained by each optimization method and their required runtimes are reported in Table 13. The optimization methods are forced to terminate their optimization procedure immediately after 5000 s, unless their results converge to the global optimum or a local one before their time is up. Accordingly the exact method has shown itself to be incapable of converging whilst optimizing instances with more than 150 jobs. In fact, the exact method requires more time/memory to converge when solving such problems. But both PSO and LFEPSO are observed to be sufficiently fast to find the global/a local optimum solution, regarding the available resources (i.e. time and memory). All these processing have been carried on a computer with a core™ i7-2640M chip processer and a 6 GB RAM. It is noteworthy, while the exact method was capable of solving only 61% of the instances in 5000 s, PSO and LFEPSO managed to solve all twenty-six test problems in 1213 and 984 s respectively. In this regard, the proposed meta-heuristic methods are found incredibly efficient confronting real-world scheduling problems with plenty of jobs executed by lots of machines. Table 14 also yields the promising ability of both meta-heuristics, reaching optimum solutions in considerably low computational times.

Compared to LFEPSO, the exact method moderately improved the quality of the final solutions by 2.8% in average. However, as

Table 12
Obtained optimum results.

Variable	z_1	z_2	z_3	z_4	μ_1	μ_2	μ_3	μ_4	λ_0	λ
Optimum value	9.7014	15.921	1.135	3.184	1	0	1	0	0	0.3004

Table 13
Results of optimizing twenty-six instances by the exact and evolutionary methods.

Problem name	Number of jobs	Number of machines	GAMS IDE/CPLEX		PSO		LFEPSO	
			λ	Runtime	λ	Runtime	λ	Runtime
P01	8	3	0.5845	1.993	0.5845	1.0129	0.5845	0.7226
P02		4	0.5756	4.448	0.5614	1.1073	0.5756	1.1922
P03	16	3	0.6046	11.085	0.5661	1.1332	0.5844	1.2472
P04		4	0.5886	15.587	0.5700	1.6398	0.5766	1.3481
P05	24	3	0.4994	106.908	0.4994	22.4198	0.4994	10.5391
P06		4	0.6241	131.350	0.5992	17.1173	0.6125	8.9734
P07	32	3	0.5144	359.040	0.4895	32.2668	0.5052	17.7847
P08		4	0.5809	248.819	0.5498	29.2341	0.5659	11.3649
P09	40	3	0.5765	532.439	0.5495	52.1513	0.5566	19.0352
P10		4	0.593	639.809	0.5637	64.0429	0.5934	23.7821
P11	50	5	0.6516	899.140	0.5839	85.0905	0.6042	31.5268
P12		10	0.6123	772.254	0.5584	93.6424	0.5737	27.8619
P13	100	5	0.6059	3110.518	0.5703	277.6009	0.5703	138.4722
P14		10	0.6337	3144.315	0.5713	365.6541	0.6008	153.0458
P15	150	5	0.6002	4983.058	0.5375	263.231	0.5889	218.9096
P16		10	0.5690	4658.824	0.5303	268.0573	0.5629	125.9263
P17	200	5	–		0.4887	443.0161	0.5396	435.0421
P18		10	–		0.4923	163.2143	0.5531	245.7476
P19	250	5	–		0.5245	1150.5010	0.5617	493.4752
P20		10	–		0.4822	463.5304	0.5623	398.7888
P21	300	25	–		0.4582	865.8966	0.5416	682.9682
P22		50	–		0.4653	361.7087	0.5727	491.1437
P23	400	25	–		0.4409	957.4972	0.5691	652.4411
P24		50	–		0.4447	961.2134	0.5176	747.7195
P25	500	25	–		0.4184	1212.8221	0.5496	983.1739
P26		50	–		0.4231	830.2055	0.5212	876.5575
Minimum			0.4994	1.993	0.4184	1.0129	0.4994	0.7226
Mean			0.5883	1232.474	0.5201	345.5769	0.5630	261.4917
Maximum			0.6516	5083.058	0.5992	1212.8221	0.6125	983.1739

Table 14
Relative objective values and runtimes.

Problem name	*GAMS/LFEPSO*		*PSO/LFEPSO*	
	λ	Runtime	λ	Runtime
P01	1	2.761277	1	1.401662
P02	1	3.731107	0.975483	0.928787
P03	1.034568	8.887915	0.968802	0.908595
P04	1.020814	11.56291	0.988613	1.216379
P05	1	10.14401	1	2.127297
P06	1.019922	14.63777	0.979183	1.90756
P07	1.018359	20.19349	0.96901	1.814781
P08	1.026511	21.89367	0.97159	2.572315
P09	1.035826	27.97131	0.987305	2.73973
P10	1	26.90298	0.950725	2.692903
P11	1.07861	28.51988	0.966402	2.698989
P12	1.068679	27.71722	0.97459	3.360948
P13	1.062587	22.46306	1	2.004741
P14	1.054851	20.54493	0.950899	2.389181
P15	1.020801	23.2199	0.914184	1.202464
P16	1.010884	36.99643	0.942086	2.128626
P17	–		0.905693	1.018329
P18	–		0.890289	0.664154
P19	–		0.933772	2.331427
P20	–		0.858007	1.162346
P21	–		0.846012	1.267843
P22	–		0.812467	0.736461
P23	–		0.774732	1.467561
P24	–		0.859158	1.285527
P25	–		0.761281	1.233578
P26	–		0.811781	0.94712
Minimum	1	2.761277	0.761281	0.664154
Mean	1.028276	19.25924	0.922772	1.700358
Maximum	1.07861	36.99643	1	3.360948

for the computational time, the exact method averagely consumed 19.26 times more than the computational time consumed by the proposed LFEPSO. In addition, Embedding Lévy flights averagely enabled PSO to improve the objective value by 7.72% while decreasing its required computational time by 70.03%. Considering both the objective value and the required runtime, LFEPSO accomplished to outperform PSO in solving more than 80% of the instances. Regarding the rest 20%, objective value is improved remarkably but it took more time for LFEPSO to reach these optimal solutions. Please notify, that low runtimes should not be always accounted as an accomplishment. Such low runtimes along with low objective values, often are a hint of getting trapped in local optima, which has been previously addressed as a major deficiency of traditional PSO. Approaching to better solutions in such cases, promises the outstanding ability of Lévy walks in solving the local optima problem. The performance of GAMS, PSO and LFEPSO are compared schematically through Figs. 4 and 5.

As depicted in Fig. 5, λ values obtained by GAMS and PSO play the role of upper and lower bounds respectively, for the values found by the proposed LFEPSO. Fig. 6, reveals the undeniable superiority of the proposed meta-heuristics dealing with the complexity of real-world scheduling problems with huge dimensions. In order to observe how performance of the LFEPSO is under the impression of its Lévy walks, Figs. 6 and 7 are presented as follows.

Embedding Lévy flights not only enabled PSO to reach better solutions in all twenty-six cases, but also its required runtimes have been decreased astonishingly in most cases. The larger the problem, the more effective the Lévy flight is. Improvement made as the result of applying Lévy flights, regarding both objective values and required computational times is more significant for the large-sized problems. The increasing gap, demonstrates the high efficiency of the proposed LFEPSO facing complex and large-sized problems. The evolution schemes for PSO and LFEPSO solving P21 to P26 are depicted through Figs. 8–13.

Figs. 8–13 also demonstrate the astonishing improvement made in PSO as the result of replacing uniformly distributed walks with Lévy flights.

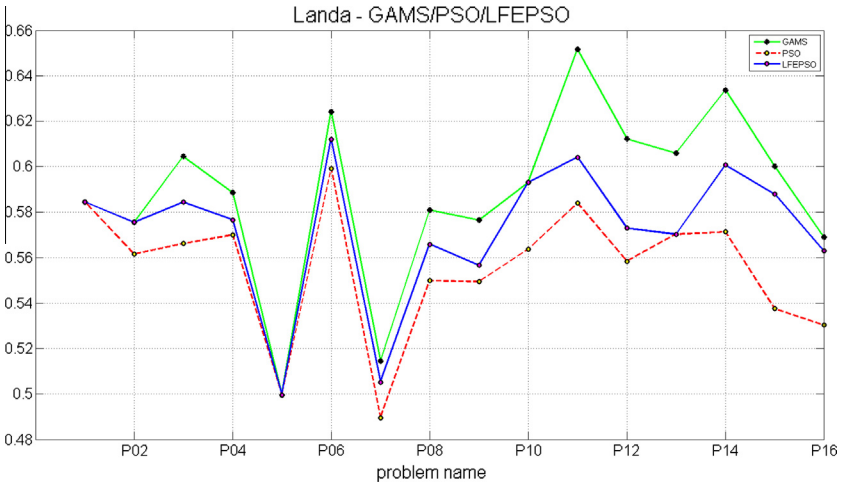

Fig. 4. Comparison of the exact and the meta-heuristic solution methods in terms of λ.

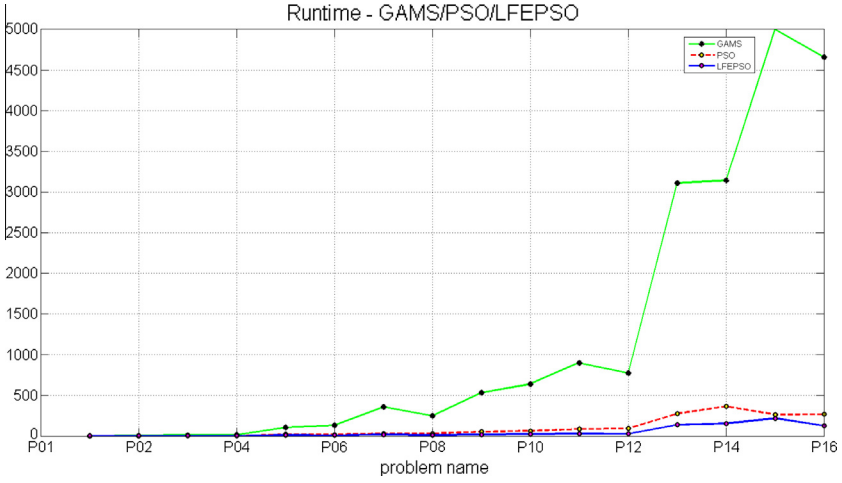

Fig. 5. Comparison of the exact and the meta-heuristic solution methods in terms of runtime.

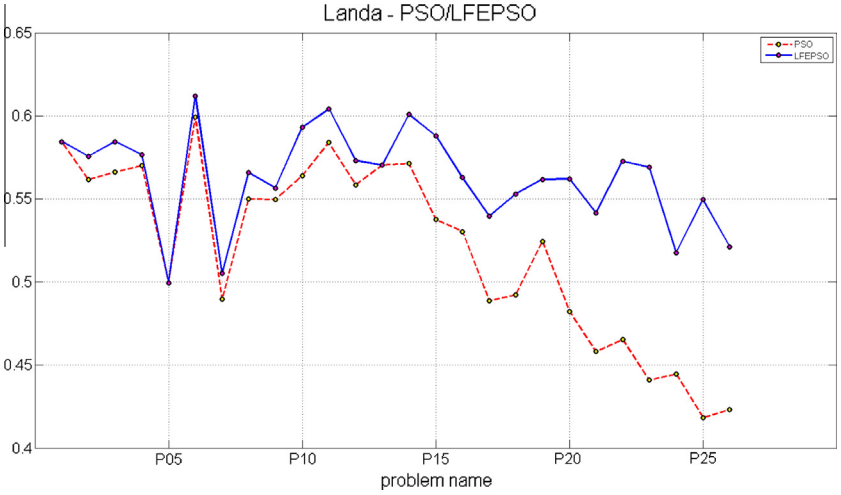

Fig. 6. Comparison of the meta-heuristic solution methods in terms of λ.

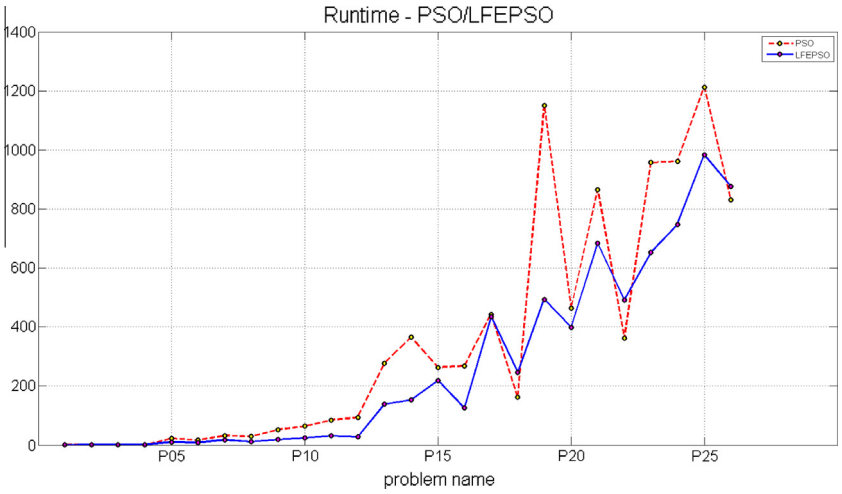

Fig. 7. Comparison of the meta-heuristic solution methods in terms of runtime.

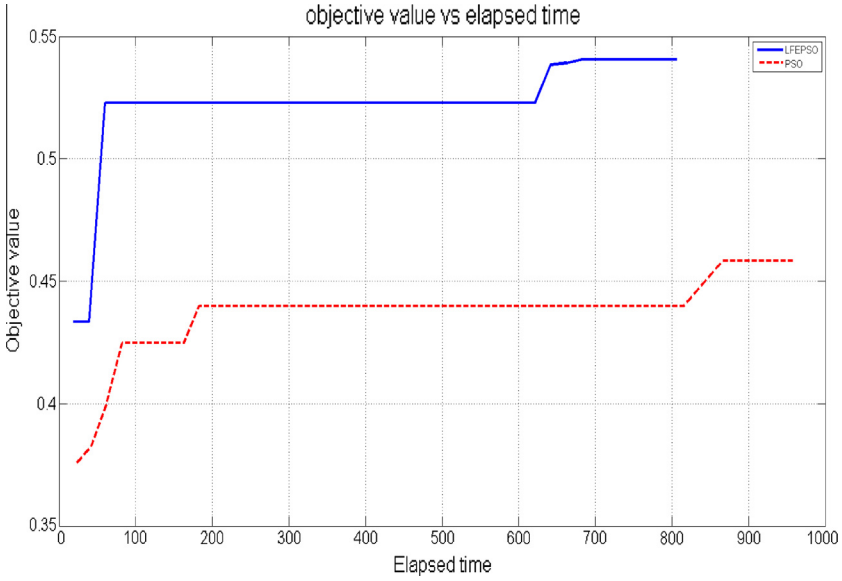

Fig. 8. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 300$ and $m = 25$.

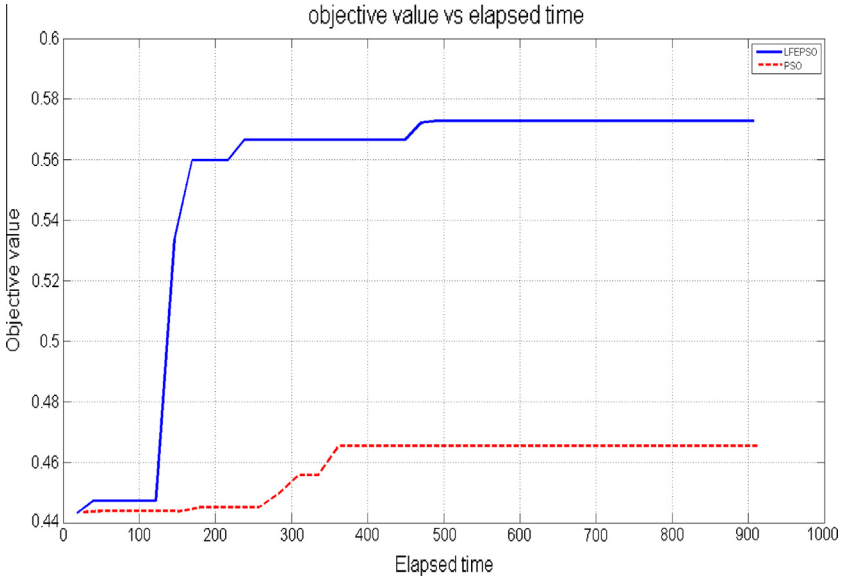

Fig. 9. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 300$ and $m = 50$.

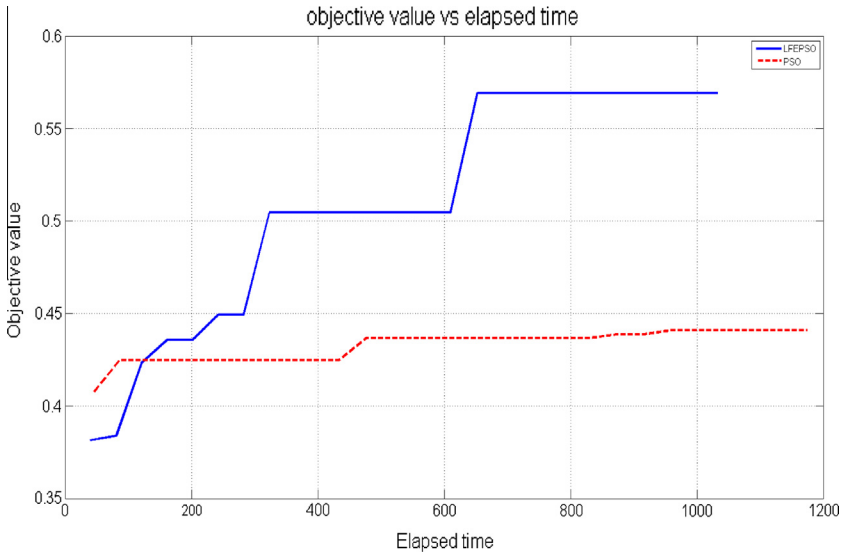

Fig. 10. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 400$ and $m = 25$.

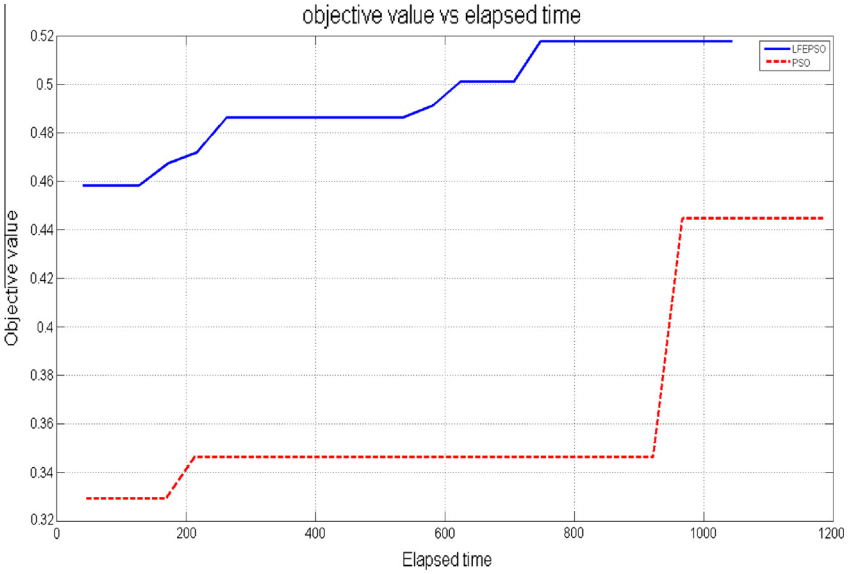

Fig. 11. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 400$ and $m = 50$.

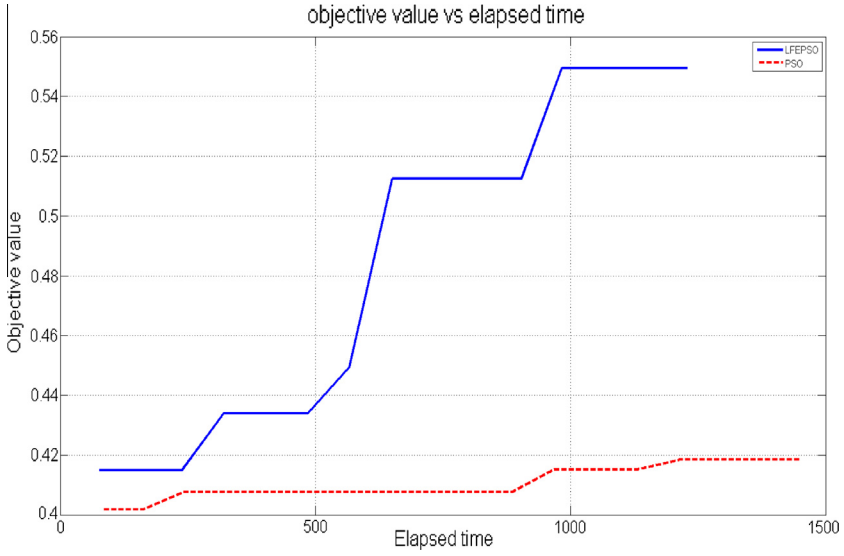

Fig. 12. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 500$ and $m = 25$.

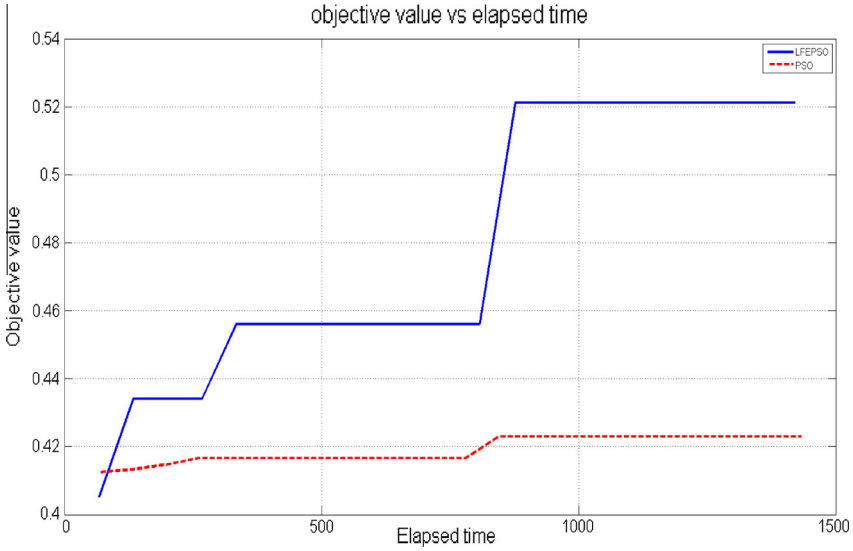

Fig. 13. Evolution schemes depicted for PSO and LFEPSO scheduling n jobs on m parallel processors. $n = 500$ and $m = 50$.

Table 15
Detailed objective values obtained as the result of considering/neglecting the adapting ability.

Problem name	With adapting ability					Without adapting ability				
	Machine hiring cost				C^S_{max}	Machine hiring cost				C^N_{max}
	MHC^p	MHC^m	MHC^o	MHC		MHC^p	MHC^m	MHC^o	MHC	
P01	20.639	18.188	13.877	17.878	9.25	20.639	18.188	13.877	17.878	11.235
P02	14.293	11.721	8.42	11.599	9.3137	14.2306	11.6698	8.3832	11.548	11.198
P03	16.156	13.781	10.43	13.618	17.927	17.5962	15.0095	11.359	14.832	21.786
P04	20.559	15.791	13.036	16.126	18.034	23.7637	18.2525	15.068	18.640	21.578
P05	18.734	15.675	13.929	15.893	26.798	20.2588	16.9508	15.062	17.187	33.251
P06	17.851	14.555	9.72	14.298	27.438	18.6571	15.2122	10.158	14.944	34.907
P07	20.864	17.021	13.088	17.006	35.665	20.864	17.021	13.088	17.006	44.944
P08	22.405	18.537	13.543	18.349	36.759	24.9957	20.6805	15.109	20.471	46.213
P09	18.465	14.86	9.351	14.542	45.259	22.5082	18.1138	11.398	17.727	57.206
P10	21.276	17.66	12.318	17.372	49.367	21.2156	17.6099	12.283	17.323	56.993
P11	21.487	17.048	12.822	17.083	57.757	31.7507	25.1913	18.946	25.243	50.961
P12	50.765	38.705	25.879	38.577	39.434	50.2921	38.3444	25.637	38.218	41.2899
P13	18.537	24.416	13.237	21.573	89.394	18.537	24.416	13.237	21.573	114.396
P14	43.739	37.083	27.48	36.591	67.199	48.5934	41.1987	30.529	40.653	82.214
P15	33.875	28.263	18.804	27.621	118.683	35.1620	29.3368	19.518	28.671	201.61
P16	52.309	42.467	30.699	42.146	113.861	28.5185	23.1527	16.736	22.977	190.369
P17	24.875	21.071	15.294	20.742	156.314	18.6066	15.7612	11.439	15.515	262.704
P18	51.855	42.484	31.134	42.154	216.496	60.8850	49.8821	36.555	49.494	175.764
P19	31.246	23.938	16.684	23.947	240.998	32.3079	24.7515	17.251	24.760	306.235
P20	51.066	40.702	29.371	40.540	203.227	51.0166	40.6626	29.342	40.501	297.655
P21	134.24	105.31	79.001	105.74	185.283	140.016	109.841	82.400	110.29	203.43
P22	259.39	209.52	151.18	208.10	126.27	254.332	205.435	148.23	204.05	195.466
P23	130.58	103.35	78.179	103.69	162.48	133.136	105.373	79.709	105.72	200.951
P24	213.44	169.15	126.23	169.37	282.22	225.501	178.708	133.36	178.94	294.573
P25	132.67	108.4	78.702	107.49	332.56	95.7601	78.2422	56.806	77.589	434.058
P26	198.38	158.04	112.81	157.22	368.91	296.285	236.036	168.48	234.81	303.642

6.4. Measuring the effect of adapting ability on the accuracy of the results

Adapting abilities are once neglected in the twenty-six sample problems in order to measure the impact of considering/neglecting the processors' adapting ability on the accuracy of the results. Obtained results are provided in Table 15. The weighted average method (Lai & Hwang, 1992; Liang, 2006a; Wang & Liang, 2005) is applied to calculate the crisp equivalent for the machine hiring costs which are supposed to follow a triangular possibility distribution. Calculated crisp machine hiring costs are also provided in Table 15.

Based on this table, both objective values (i.e. MHC and C_{max}) increase as the number of jobs goes up. Since it takes more time and cost to process an increasing number of jobs, the observed incremental trend is definitely justified. On the other hand, new machine alternatives provide new opportunities which might lead to lower maximum completion time and machine hiring costs. In fact an increasing number of machines usually offers new and probably cheaper processors to take care of jobs. In order to compare the effect of considering/neglecting the adapting ability of the processors, Table 16 is given as follows.

Processors are underestimated as the result of neglecting their adapting ability. In fact, they will be deemed to operate with a

Table 16
The effect of considering/neglecting the adapting ability on the obtained results.

Problem name	With adapting ability		Without adapting ability		MHC error	C_{max} error
	MHC^S	C^S_{max}	MHC^N	C^N_{max}	$\left\|\frac{MHC^S-MHC^N}{MHC^S}\right\| \times 100$	$\left\|\frac{C^S_{max}-C^N_{max}}{C^S_{max}}\right\| \times 100$
P01	17.878	9.252	17.878	11.235	0	21.45946
P02	11.5995	9.3137	11.5488	11.198	0.436401	20.23149
P03	13.6183	17.927	14.8323	21.786	8.914513	21.52619
P04	16.1265	18.034	18.6403	21.578	15.58828	19.65177
P05	15.8938	26.798	17.1874	33.251	8.139374	24.08016
P06	14.2985	27.438	14.9441	34.907	4.515742	27.22137
P07	17.006	35.665	17.006	44.944	0	26.0171
P08	18.3493	36.759	20.4711	46.213	11.56345	25.71887
P09	14.5426	45.259	17.7270	57.206	21.89686	26.39696
P10	17.3723	49.367	17.3230	56.993	0.283469	15.44757
P11	17.0835	57.757	25.2438	50.961	47.76739	11.76654
P12	38.5773	39.434	38.2180	41.2899	0.931428	4.706345
P13	21.573	89.394	21.573	114.396	0	27.96832
P14	36.5918	67.199	40.6530	82.214	11.09874	22.34408
P15	27.6218	118.683	28.6713	201.61	3.799439	69.87269
P16	42.146	113.861	22.9777	190.369	45.48063	67.19421
P17	20.7421	156.314	15.5152	262.704	25.19952	68.06172
P18	42.1541	216.496	49.4948	175.764	17.414	18.8142
P19	23.947	240.998	24.7608	306.235	3.398652	27.06952
P20	40.5408	203.227	40.501	297.655	0.096685	46.4643
P21	105.746	185.283	110.29	203.43	4.303156	9.794207
P22	208.108	126.27	204.05	195.466	1.949637	54.80003
P23	103.693	162.48	105.72	200.951	1.957783	23.67738
P24	169.378	282.22	178.94	294.573	5.650966	4.377082
P25	107.495	332.56	77.589	434.058	27.82084	30.52021
P26	157.225	368.91	234.81	303.642	49.35259	17.69212
Minimum	11.5995	9.252	11.548	11.198	0	4.377082
Mean	50.7426	116.803	53.330	142.101	12.21383	28.18746
Maximum	208.108	368.91	234.81	434.058	49.35259	69.87269

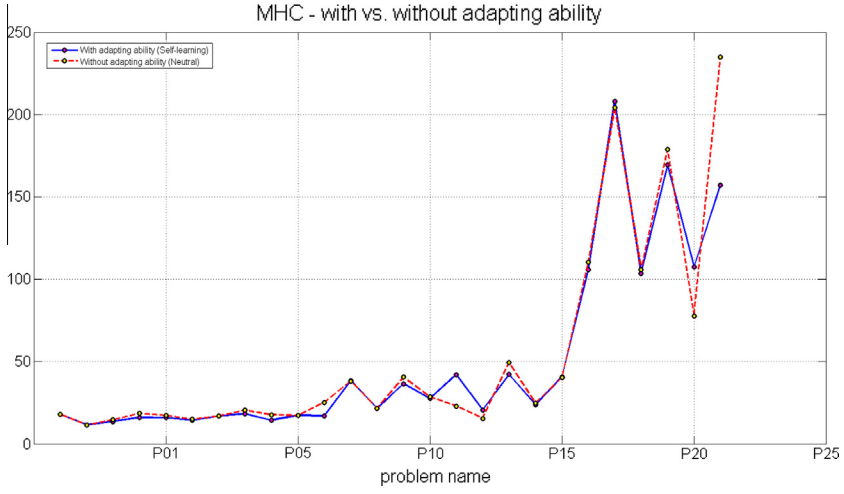

Fig. 14. MHC values obtained with and without considering the adapting ability.

slower pace than which they really do. In such cases, the obtained completion times are expected to have unrealistic large values. It also forces the manufacturer to hire more machines to reduce the overestimated completion times. Therefore, the optimum values for both objectives are expected to increase by mistake as the result of neglecting the adapting ability. Comparing θ_1 (i.e. the C_{max} importance weight) to $\sum_{i=2}^{4}\theta_i$ (i.e. the total machine hiring cost related weights), minimizing *MHC* is accounted as the first priority in the decision making procedure. With this regard, neglecting the adapting ability often increases C_{max} more significantly than *MHC*. Objective Errors are provided in the two last columns of Table 16. Neglecting the processors' adapting ability made *MHC* and C_{max} to increase by 12.21% up to 49.35% and 28.19% up to 69.87% respectively. For further comparison, obtained results are depicted schematically through Figs. 14 and 15.

Tables 15 and 16 and Figs. 14 and 15, demonstrate the significant gaps made as the results of neglecting the ability of operators to adapt. They also yield that taking the adapting ability of the processors into consideration is a great step towards accurate modeling. Due to Figs. 14 and 15, the objective errors are much more evident for large-sized problems. The incremental trend of the objective errors indicates the undeniable role of considering the adapting ability in the validity of the formulated mathematical models especially for modeling real-world and huge-sized problems.

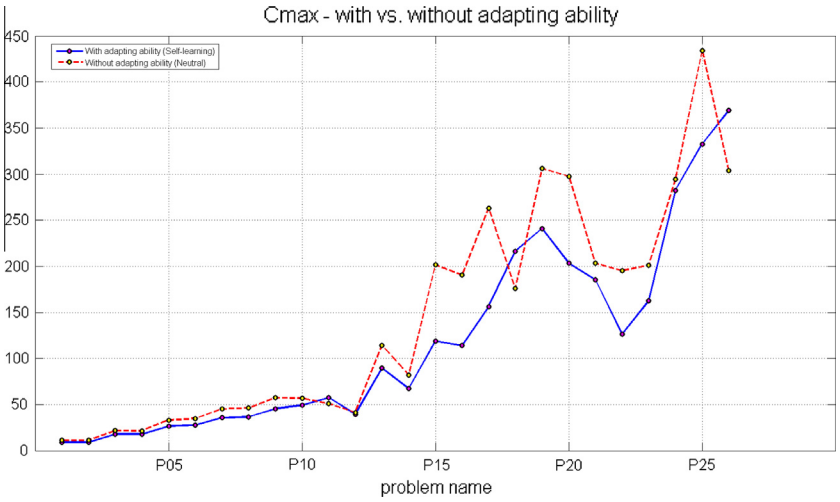

Fig. 15. C_{max} values obtained with and without considering the adapting ability.

7. Conclusions and outlook on future work

The proposed uniform parallel machine scheduling problem is formulated into a bi-objective mixed integer mathematical model with several imprecise parameters. To deal with the ambiguousness of the parameters, an interactive possiblistic programming approach is applied. Then, the aforementioned model is converted to a mixed integer crisp mathematical model. Since optimizing such NP-hard problems in real-world sizes requires some low order algorithms, an evolutionary method is developed. The traditional PSO is highly modified in order to lend itself well to the complicated structure of the problem. Also, Lévy flights are embedded in the proposed meta-heuristic method (i.e. LFEPSO) in order to avoid particles getting trapped in local optima. Compared to the GAMS IDE/CPLEX solver, LFEPSO accomplished to find promising solutions in a considerable short amount of time. Furthermore, concerning both objective values and runtime, LFEPSO outperformed the traditional PSO, 80% of the time. While PSO got stuck in a local optima for the rest 20%, LFEPSO managed to escape from such local optima and find much better solutions. Obviously, it took a little more computational time for LFEPSO to reach these superior solutions. Due to the significant improvements made in the quality of such solutions, consuming a little more time is fairly acceptable. Also, as the numerical results show, neglecting the adapting ability of the processors, results in undeniable gaps, especially for large-sized problems. As for future research directions, there would be worthwhile results in extending this study to other manufacturing systems such as flow shop and job shop. Also, future studies are required to investigate the problem considering further scheduling criteria and assumptions, namely earliness, tardiness, setup cost/time considerations and so on. Also, it is of special interest to implement the findings of this paper in other problem environments such as considering learning rates in unprintable, autonomous systems. Just after a little adjustment, same modeling and optimization techniques can be applied in such environments. Proper timing technique is required to provide enough data, in order to calculate special ratios (i.e. $\widetilde{e^{c}_{jjk}}$ and $\widetilde{e^{r}_{jjk}}$) by using regression methods in such working environments.

References

Biskup, D. (2008). A state-of-the-art review on scheduling with learning effects. *European Journal of Operational Research, 188*(2), 315–329.

Bozbura, F. T., Beskese, A., & Kahraman, C. (2007). Prioritization of human capital measurement indicators using fuzzy AHP. *Expert Systems with Applications, 32*(4), 1100–1112.

Brown, C., Liebovitch, L. S., & Glendon, R. (2007). L'evy flights in Dobe Ju/'hoansi foraging patterns. *Human Ecology, 35*, 129–138.

Celik, M., Deha Er, I., & Ozok, A. F. (2009). Application of fuzzy extended AHP methodology on shipping registry selection: The case of Turkish maritime industry. *Expert Systems with Applications, 36*(1), 190–198.

Cheng, T. C. E., Kang, L. Y., & Ng, C. T. (2007). Due-date assignment and parallel-machine scheduling with deteriorating jobs. *Journal of the Operational Research Society, 58*(8), 1103–1108.

Cheng, T. C. E., & Sin, C. C. S. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research, 47*(3), 271–292.

Cheng, T. C. E., Wu, C. C., & Lee, W. C. (2008). Some scheduling problems with deteriorating jobs and learning effect. *Computers & Industrial Engineering, 54*(4), 972–982.

Eberhart, R. C., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *International symposium on micromachine and human science* (pp. 39–434).

Eren, T. (2009). A bicriteria parallel machine scheduling with a learning effect of setup and removal times. *Applied Mathematical Modelling, 33*(2), 1141–1150.

Erensal, Y. C., Oncan, T., & Demircan, M. L. (2005). Determining key capabilities in technology management using fuzzy analytic hierarchy process: A case study of Turkey. *Information Sciences, 176*(18), 2755–2770.

Elvikis, D., & T'kindt, V. (2014). Two-agent scheduling on uniform parallel machines with min-max criteria. *Annals of Operations Research, 213*(1), 79–94.

Haklı, H., & Uğuz, H. (2014). A novel particle swarm optimization algorithm with Levy flight. *Applied Soft Computing, 23*, 333–345.

Heizer, J., & Render, B. (1999). *Operations management* (5th ed.). Englewood Cliffs, NJ: Prentice Hall.

Higashi, N., & Iba, H. (2003). Particle swarm optimization with Gaussian mutation. In *Proc. IEEE swarm intelligence symposium, Indianapolis* (pp. 72–79).

Hsu, H. M., & Wang, W. P. (2001). Possibilistic programming in production planning of assemble-to-order environments. *Fuzzy Sets and Systems, 119*, 59–70.

Huang, X., Wang, M. Z., & Ji, P. (2014). Parallel machines scheduling with deteriorating and learning effects. *Optimization Letters, 8*(2), 493–500.

Inuiguchi, M., & Ramik, J. (2000). Possibilistic linear programming: A brief review of fuzzy mathematical programming and a comparison with stochastic programming in portfolio selection problem. *Fuzzy Sets and Systems, 111*, 3–28.

Jana, N. D., & Sil, J. (2014). Particle swarm optimization with Lévy flight and adaptive polynomial mutation in gbest particle. In *Recent advances in intelligent informatics* (pp. 275–282). Springer International Publishing.

Kahraman, C., Cebeci, U., & Ruan, D. (2004). Multi-attribute comparison of catering service companies using FAHP: The case of Turkey. *International Journal of Production Economics, 87*(2), 171–184.

Kanagaraj, G., Ponnambalam, S. G., & Jawahar, N. (2013). A hybrid cuckoo search and genetic algorithm for reliability–redundancy allocation problems. *Computers & Industrial Engineering, 66*(4), 1115–1124.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *IEEE international joint conference on neural networks* (pp. 1942–1948). IEEE Press.

Kumar, M., Vrat, P., & Shankar, R. (2006). A fuzzy programming approach for vendor selection problem in a supply chain. *International Journal of Production Economics, 101*, 273–285.

Lai, Y. J., & Hwang, C. L. (1992). A new approach to some possibilistic linear programming problems. *Fuzzy Sets and Systems, 49*, 121–133.

Lai, Y. J., & Hwang, C. L. (1993). Possibilistic linear programming for managing interest rate risk. *Fuzzy Sets and Systems, 54*, 135–146.

Leccardi, M. (2005). Comparison of three algorithms for Lèvy noise generation. In *Proceedings of fifth EUROMECH nonlinear dynamics conference*.

Lee, C. Y., & Yao, X. (2004). Evolutionary programming using mutations based on the Lévy probability distribution. *IEEE Transactions on Evolutionary Computation, 8* (1), 1–13.

Li, K., & Yang, S. L. (2009). Non-identical parallel-machine scheduling research with minimizing total weighted completion times: Models, relaxations and algorithms. *Applied Mathematical Modelling, 33*(4), 2145–2158.

Liang, T. F. (2006a). Distribution planning decisions using interactive fuzzy multi-objective linear programming. *Fuzzy Sets and Systems, 157*, 1303–1316.

Liang, T. F. (2006b). Integrating production–transportation planning decision with fuzzy multiple goals in supply chains. *International Journal of Production Research*, preview article.

Lu, Y. Y., Wei, C. M., & Wang, J. B. (2012). Several single-machine scheduling problems with general learning effects. *Applied Mathematical Modelling, 36*(11), 5650–5656.

Mantegna, R. N. (1994). Fast, accurate algorithm for numerical simulation of Levy stable stochastic processes. *Physical Review E, 49*, 4677–4683.

Mosheiov, G. (2001a). Parallel machine scheduling with a learning effect. *Journal of the Operational Research Society*, 1165–1169.

Mosheiov, G. (2001b). Scheduling problems with a learning effect. *European Journal of Operational Research, 132*(3), 687–693.

Mula, J., Poler, R., & Garcia, J. P. (2006). MRP with flexible constraints: A fuzzy mathematical programming approach. *Fuzzy Sets and Systems, 157*, 74–97.

Pakzad-Moghaddam, S. H., Mina, H., & Tavakkoli-Moghaddam, R. (2014). An approach for modeling a new single machine scheduling problem with deteriorating and learning effects. *Computers & Industrial Engineering, 78*, 33–43.

Pavlyukevich, I. (2007a). L'evy flights, non-local search and simulated annealing. *Journal of Computational Physics, 226*, 1830–1844.

Pavlyukevich, I. (2007b). Cooling down L'evy flights. *Journal of Physics A: Mathematical and Theoretical, 40*, 12299–12313.

Reynolds, A. M., & Frye, M. A. (2007). Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search. *PLoS One, 2*, e354.

Richer, T. J., & Blackwell, T. M. (2006). The Lévy particle swarm. In *IEEE congress on evolutionary computation, 2006 (CEC 2006)* (pp. 808–815). IEEE.

Selim, H., & Ozkarahan, I. (in press). A supply chain distribution network design model: An interactive fuzzy goal programming-based solution approach. *The International Journal of Advanced Manufacturing Technology*.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In *The 1998 IEEE international conference on evolutionary computation proceedings, Anchorage, AK* (pp. 69–73).

Tang, J., & Zhao, X. (2009). Particle swarm optimization with adaptive mutation. In *WASE international conference on information engineering*.

Torabi, S. A., & Hassini, E. (2008). An interactive possibilistic programming approach for multiple objective supply chain master planning. *Fuzzy Sets and Systems, 159* (2), 193–214.

Wang, R. C., & Liang, T. F. (2005). Applying possibilistic linear programming to aggregate production planning. *International Journal of Production Economics, 98*, 328–341.

Wang, H., Liu, Y., Li, C. H., & Zeng, S. Y. (2007). A hybrid particle swarm algorithm with Cauchy mutation. In *Proc. of IEEE swarm intelligence symposium* (pp. 356–360).

Webb, G. K. (1994). Integrated circuit (IC) pricing. *High Technology Management Research, 5*(II), 247–260.

Wu, X., & Zhong, M. (2009). Particle swarm optimization based on power mutation. In *ISECS international colloquium on computing, communication, control, and management*.

Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver Press.

Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. In *World congress on nature & biologically inspired computing, 2009 (NaBIC 2009)* (pp. 210–214). IEEE.

Yang, X.-S., & Deb, S. (2011). Multiobjective cuckoo search for design optimization. *Computers & Operations Research*.

Yeh, W. C., Lai, P. J., Lee, W. C., & Chuang, M. C. (2014). Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects. *Information Sciences, 269*, 142–158.

Zimmermann, H. J. (1978). Fuzzy programming and linear programming with several objective functions. *Fuzzy Sets and Systems, 1*, 45–55.