sustainability

Article

Energy-Efficient Scheduling Problem Using an Effective Hybrid Multi-Objective Evolutionary Algorithm

Lvjiang Yin [1,2], Xinyu Li [1], Chao Lu [1] and Liang Gao [1,*]

[1] The State Key Lab of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; yinlvjiang@hust.edu.cn (L.Y.); lixinyu@hust.edu.cn (X.L.); luchao@hust.edu.cn (C.L.)

[2] School of Economics and Management, Hubei University of Automotive Technology, Shiyan 442002, China

[*] Correspondences: gaoliang@hust.edu.cn; Tel./Fax: +86-27-8755-7742

Academic Editor: Tomonobu Senjyu
Received: 1 September 2016; Accepted: 25 November 2016; Published: 7 December 2016

Abstract: Nowadays, manufacturing enterprises face the challenge of just-in-time (JIT) production and energy saving. Therefore, study of JIT production and energy consumption is necessary and important in manufacturing sectors. Moreover, energy saving can be attained by the operational method and turn off/on idle machine method, which also increases the complexity of problem solving. Thus, most researchers still focus on small scale problems with one objective: a single machine environment. However, the scheduling problem is a multi-objective optimization problem in real applications. In this paper, a single machine scheduling model with controllable processing and sequence dependence setup times is developed for minimizing the total earliness/tardiness (E/T), cost, and energy consumption simultaneously. An effective multi-objective evolutionary algorithm called local multi-objective evolutionary algorithm (LMOEA) is presented to tackle this multi-objective scheduling problem. To accommodate the characteristic of the problem, a new solution representation is proposed, which can convert discrete combinational problems into continuous problems. Additionally, a multiple local search strategy with self-adaptive mechanism is introduced into the proposed algorithm to enhance the exploitation ability. The performance of the proposed algorithm is evaluated by instances with comparison to other multi-objective meta-heuristics such as Nondominated Sorting Genetic Algorithm II (NSGA-II), Strength Pareto Evolutionary Algorithm 2 (SPEA2), Multiobjective Particle Swarm Optimization (OMOPSO), and Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D). Experimental results demonstrate that the proposed LMOEA algorithm outperforms its counterparts for this kind of scheduling problems.

Keywords: multi-objective evolutionary algorithm; energy efficient scheduling; controllable processing times; single machine scheduling; just-in-time

1. Introduction

In general, most scheduling problems usually involve multiple objectives like cost, tardiness, and earliness due to demand of practical production and these objectives are often conflicting with each other, which is a challenging task for solving the optimal solution [1]. Thus, study of the multi-objective scheduling problem is very important in terms of theoretical significance and applying value. In recent years, many manufacturing companies and researchers have started to focus on the minimum energy consumption [2]. Just-in-time (JIT) production is another key issue because most companies are concerned not only with reducing inventory cost but also with delivering on time. It is noted that there is a remarkable relation between JIT scheduling problems and controllable processing times, since the scheduling environment can approach JIT philosophy by controlling the process times [3]. The controllable processing times mean that each operation can be processed in a reasonable shorter or longer time by allocating the available resources like fuel, equipment, catalyzes, manpower and so on. The mentioned-above factors (i.e., energy consumption and JIT) directly affect the production efficiency. Therefore, energy consumption and JIT production should be taken into account simultaneously in manufacturing sectors.

The energy consumption problem is driven by the fact that the pressing campaign of sustainable development is now widely accepted in real production systems. "Rising energy consumption has triggered some negative impacts on environment which has led to legislative restrictions on manufacturing companies" [4]. Obviously, the current trend of green manufacturing has brought about new pressures and challenges for manufacturing enterprises. Consequently, "enterprises have to reduce energy consumption so as to save cost and become more environmentally friendly". Most researches on minimizing energy consumption largely concentrate on more energy efficient machines or machine process plans. However, the majority of energy consumption is not directly related to the production of components [5]. Drake et al. [6] further proved that there existed significant energy consumption when a machine is idle but still runs. Therefore, minimization of energy consumption can be realized by turning off/on the machine between scheduled jobs rather than keeping the machine idle. Compared with machine or process redesign, this turn off/on strategy only needs a little financial investment and can easily be implemented.

JIT production also plays a key role in manufacturing sectors since a scheduling scenario based on JIT philosophy can effectively reduce cost and improve reputation of companies among the customers [7]. JIT philosophy in manufacturing systems has been studied widely in the scheduling field, especially for those with earliness and tardiness (E/T) problems. In the JIT environment, tasks which are finished before their due date may incur earliness penalties such as inventory cost while a tardy task can cause customer dissatisfaction, lost sales and loss of reputation. Therefore, a desirable schedule is one where all jobs are completed exactly on their own due dates. In reality, JIT scheduling can be achieved by controlling job processing times. Although job processing times in most literatures are fixed positive parameters, it is now accepted that in practice the scheduler can change processing times by increasing or reducing additional resources. Therefore, controllable processing times can try to ensure that all jobs can be processed on their own due dates, which can increase profit by squeezing inventory cost and improving customer satisfaction.

Most conventional scheduling problems use production efficiency, cost and quality as their preeminent optimization objectives. However, because of increasing costs of energy and environmental pollution, "low-carbon scheduling" as a novel scheduling model has received increasing attention from scholars and engineers [8]. Based on the discussion above, many researchers and manufacturing companies realize the importance of energy efficiency. To the best of the authors' knowledge, the single machine scheduling problem with controllable processing and setup times, including energy consumption conception, still has not been studied in the previous researches reported. Furthermore, this addressed scheduling is not only a multi-objective optimization problem (MOP) in nature but also a NP-hard problem. Multi-objective evolutionary algorithm (MOEA) is very suitable to solve this kind of problem because MOEA can obtain the non-dominated solutions in a single run and is also applied successfully in the optimization field [9–14]. This motivates us to develop a new multi-objective scheduling model and propose a MOEA to cope with this type of problem.

For a single-machine scheduling environment, Mouzon et al. [15] proposed a multi-objective mathematical programming model and several algorithms for a single Computer numerical control (CNC) machine scheduling problem with the goals of reducing both energy consumption and total completion time. Mouzon and Yildirim [16] applied a greedy randomized adaptive search algorithm to a single-machine multi-objective optimization scheduling problem with the objective of minimizing the total energy consumption and total tardiness. Rager et al. [2] proposed an evolutionary algorithm for energy-oriented parallel machine scheduling. For the flow-shop environment, Fang and

Uhan et al. [17,18] presented some new mathematical programming models of its scheduling problem that consider "peak power load, energy consumption, and associated carbon footprint in addition to cycle time". Bruzzone et al. [19] proposed an approach that relies on a mixed-integer programming model in which the reference schedule was modified by an advanced planning and scheduling (APS) system to consider energy consumption without changing the jobs' assignments and sequencing. Dai et al. [20] proposed an energy-efficient model for flexible flow-shop scheduling and developed a genetic-simulated annealing algorithm to solve it. Luo et al. [21] proposed a new ant colony optimization meta-heuristic for production efficiency and electric power cost for a hybrid flow shop scheduling problem. Keller et al. [22] developed a heuristic method for the hybrid flow-shop scheduling problem considering energy flexibility. For the job-shop scheduling environment, Liu et al. [23] developed a multi-objective scheduling method for the classical job-shop scheduling problem (JSP) with total energy consumption and total weighted tardiness as objectives. Kang et al. [24,25] to address a selected problem two approaches are investigated, i.e., traditional scheduling algorithms and genetic algorithms based multi-objective optimization, both approaches are compared based given key performance indicators, computational time required and the quality of generated schedule.

In this work we mainly study on a single machine scheduling with controllable processing and setup times for minimizing production costs (i.e., cost and total E/T) and energy consumption. These objectives are important Key Performance Indicators (KPIs) for enterprises. The purpose of our current study is to develop a MOEA called local MOEA (LMOEA) to obtain trade-offs between the three objectives closer towards true Pareto front. According to the features of the addressed problem and evolutionary algorithm, this proposed LMOEA uses a new encode mechanism which can convert discrete optimization problems into continuous optimization problems, and adopts a hybrid local search with three different strategies to enhance exploitation ability.

The remainder of the paper is organized as follows. In Section 2, we give a review on scheduling problems considering energy consumption and JIT scheduling with controllable processing time. In Section 3, we describe a formal definition of the addressed problem. Then, we proposed a MOEA with multiple local search operators for handling the scheduling problem in Section 4. The experimental results of our proposed LMOEA are presented and analyzed in Section 5. In Section 6, we give a case study. Concluding remarks and suggestions for several future research subjects are given in Section 7.

2. Related Works

Study on energy consumption and JIT production scheduling with controllable processing times has been quite limited. This section gives a review on energy consumption and JIT production scheduling with controllable processing times, respectively.

2.1. Energy Consumption in the Scheduling Problem

Here, we briefly state some researches on different scheduling environments for minimizing energy consumption in the following paragraphs.

For a single machine scheduling environment, Mouzon et al. [15] employed operational methods and dispatching rules to reduce energy consumption significantly. Mouzon and Yildirim [16] formulated a mathematical model and proposed a multi-objective greedy random adaptive search algorithm for minimizing the total tardiness and energy consumption. Afterward, Yildirim and Mouzon [26] proposed a mathematical model to minimize total energy consumption and total completion time, and utilize a multi-objective genetic algorithm to optimize the two objectives. Shrouf et al. [27] addressed the scheduling problem to reduce production cost considering fluctuating energy prices during short periods.

For a parameter optimization problem, Lu et al. [28] developed a model for minimizing unit production energy consumption and machining precision value. He et al. [29] proposed a model to reduce energy consumption and the makespan, then a heuristic algorithm is developed to obtain the optimal or approximate optimal solutions based on tabu search strategy.

For a flow shop scheduling, Fang et al. [17] presented a multi-objective mathematical formulation with the objective to minimize makespan, the peak total energy consumption, and carbon footprint. Bruzzonea et al. [18] presented an approach to amend the schedule of jobs in a flexible flow shop so as to regulate the maximum peak power constraint. Xiang et al. [30] presented an improved genetic algorithm to solve the hybrid flow shop scheduling considering energy consumption.

For the other scheduling environments, Zhang et al. [31] presented a new programming mathematical model which considers the energy consumption and the schedule efficiency simultaneously in flexible manufacturing system. Recently, study on scheduling with environmentally-oriented objectives is few but growing. For example, Liu et al. [32] the epsilon-archived genetic algorithm to minimize the total tardiness and carbon dioxide emission in a batch scheduling problem. They the first attempt to handle the batch scheduling problem for minimizing carbon dioxide emissions and total weighted tardiness simultaneously.

2.2. JIT Production in Scheduling Problem

JIT production is usually associated with the E/T scheduling problem. There also exists a close relationship between E/T scheduling and controllable processing times. Therefore, some works on single machine E/T scheduling problem are reviewed in the following paragraphs.

The E/T scheduling problem on a single machine has been studied by many researchers so far. For example, Mason et al. [33] addressed a single machine E/T scheduling with unrestrictive common due dates. Lenstra et al. [34] considered the non-preemptive single machine scheduling problem concerning E/T penalties. Yano and Kim [35] considered a single machine scheduling problem for minimizing sum of weighted tardiness and earliness penalties. Valenta et al. [36,37] proposed a beam search algorithm for handling a single machine E/T scheduling problem with job independent penalties and no idle time. Khorshidian et al. [38] concerns with a JIT single machine scheduling with machine idle time. Weng et al. [39] considered a dynamic hybrid JIT scheduling problem involving continual arrival of new jobs. Benmansour et al. [40] considered an E/T scheduling with periodic preventive maintenance. Sourd et al. [41] proposed a faster branch and bound algorithm to solve an E/T scheduling. Baker and Kenneth [42] considered a single E/T stochastic scheduling problem. There are a few literatures on the JIT scheduling problem with controllable processing times. For example, Kayvanfar et al. [3] proposed a single machine scheduling problem with controllable processing times to minimize the total E/T. They presented an efficient heuristic called NBC-NBE approach for the JIT problem. Shabtay and Steiner [43] addressed a single machine E/T scheduling problem with due date and controllable processing times.

Based on the previous research described above, we are able to state that the research of energy-oriented scheduling with controllable processing times is very important in terms of theoretical significance and applying value. In the present context, we first attempt to develop a mathematical model to minimize total energy consumption, cost and total E/T simultaneously. An effective MOEA called LMOEA is also proposed for solving this scheduling problem. The proposed LMOEA has several features as follows: (1) a new solution representation is presented, which can help convert discrete combinational problems into continuous problems; (2) a local search strategy, combining three different local search strategies with self-adaptive scheme, is introduced into our proposal for improving search efficiency.

3. Problem Formulation

In this section, a mathematical model of the scheduling problem considered is given. Before describing this model, the notations and the assumptions are listed as follows.

3.1. Notations and Assumptions

The notations employed in the addressed problem are as follows.

N number of jobs.
J$_j$ the *j*th job.
J$_{(i)}$ job placed in the *i*th position in a processing sequence.
Π a processing sequence of jobs.
p_j normal processing time of job *j*.
p_j^c crash (minimum allowable) processing time of job j.
p_j^e expansion (maximum allowable) processing time of job *j*.
m_j^c maximum amount of compression for job *j*, namely $m_j^c = p_j - p_j^c$.
m_j^e maximum amount of expansion for job *j*, namely $m_j^e = p_j^e - p_j$.
x_j^c amount of compression of job *j*.
x_j^e amount of expansion of job *j*.
p_j^a actual processing time of job *j*, namely $p_j^a = p_j - x_j^c + x_j^e$.
a_j unit cost of compression of job *j*.
b_j unit cost of expansion of job *j*.
d_j due date of job *j*.
C_j completion time of job *j*.
T_j tardiness of job *j*, namely $T_j = \max(0, C_j - d_j)$.
E_j earliness of job *j*, namely $E_j = \max(0, d_j - C_j)$.
t_j unit tardiness penalty factor for job *j*.
e_j unit earliness penalty factor for job *j*.
S_j start time of job *j*.
s_{ij} setup time between job *i* and job *j*.
M an arbitrary big positive number.

The job processing times are controllable and the setup times of jobs are sequence-dependent in this scheduling. Considering a set of *n* jobs to be performed on a single machine, some assumptions are necessary as follows.

- Setup time for a job is zero if it is the first order in a processing sequence.
- Machine is idle when there exists setup time s_{ij} between job *i* and job *j*.
- Each machine can process only one job at a time.
- No job preemption is allowed.
- The processing time is discrete.
- The normal processing time could be compressed by an amount of x_j^c which requires a unit cost of compression.
- The normal processing time could be expanded by an amount of x_j^e which requires a unit cost of expansion.
- The job with normal processing time will incur no extra processing cost.

3.2. Mathematical Model

The considered scheduling problem contains three objectives namely E/T, penalty cost, and energy consumption. The mathematical model consists of the production model and energy consumption model. Minimization of total E/T and cost is objective of production model. Reducing energy consumption is objective of energy consumption model. The production model and energy consumption model are given in the following subsections, respectively.

3.2.1. Production Model

In the production model, we present a bi-objective mathematical model to minimize total E/T and penalty cost simultaneously. The model is formulated as follows.

$$\min f_1 = ET = \sum_{j=1}^{n} \left(t_{(j)} T_{(j)} + e_{(j)} E_{(j)} \right) \tag{1}$$

$$\min f_2 = cost = \sum_{j=1}^{n} \left(a_{(j)} x^c_{(j)} + b_{(j)} x^e_{(j)} \right) \tag{2}$$

$$S_{(j)} + p_{(j)} - x^c_{(j)} + x^e_{(j)} + s_{(j)(j+1)} \leq S_{(j+1)}, j = 1, \ldots, n-1 \tag{3}$$

$$S_{(j)} + p_{(j)} - x^c_{(j)} + x^e_{(j)} - d_{(j)} \leq T_{(j)}, j = 1, \ldots, n \tag{4}$$

$$d_{(j)} - S_{(j)} - p_{(j)} + x^c_{(j)} - x^e_{(j)} \leq E_{(j)}, j = 1, \ldots, n \tag{5}$$

$$m^c_{(j)} \geq x^c_{(j)}, j = 1, \ldots, n \tag{6}$$

$$m^e_{(j)} \geq x^e_{(j)}, j = 1, \ldots, n \tag{7}$$

$$M\lambda \geq x^c_{(j)}, j = 1, \ldots, n \tag{8}$$

$$M(1-\lambda) \geq x^e_{(j)}, j = 1, \ldots, n \tag{9}$$

$$T_{(j)} \geq 0, E_{(j)} \geq 0,\ x^c_{(j)} \geq 0,\ x^e_{(j)} \geq 0,\ S_{(j)} \geq 0,\ \lambda = 0\ or\ 1 \tag{10}$$

where Equation (1) denotes the total sum of tardiness and earliness; Equation (2) represents the total penalty cost of compression and expansion. Constraint (3) defines that the sum of the completion time of the job in the *j*th position and the setup time of the jobs between the *j*th and (*j* + 1)th position is less than or equal to the start time of the job in the (*j* + 1)th position. Constraints (4) and (5) define the tardiness and earliness of the job which must be minimized. Constraints (6) and (7) restrict the amount of compression and expansion of each job, respectively. Constraints (8) and (9) simultaneously guarantee that only one of compression and expansion for a job is applied to this scheduling problem, which implies that the processing time is either compressed or expanded for each job. Constraint (10) presents non-negativity of variables.

3.2.2. Energy Consumption Model

The energy consumption model is developed in this subsection. For the sake of simplicity of computation, we just consider energy consumption on scheduling process. Therefore, a power model on a scheduling can be divided into three types of energy. They are energy consumption during idle time, energy consumption when turning on/off machine (if it is necessary), and energy consumption when processing the job.

The idle power is denoted by P_{idle}; the additional energy power during processing job is defined by P_{run}; the total energy requirement when switching off and on machine is denoted by E_{sc}. When a machine stays idle for a long time, the machine should be switched off so as to save energy. If the total amount of energy required (namely, E_{sc}) for turning off and on the machine exceeds the energy demand during the idle time between the jobs in the *j*th and (*j* + 1)th position (denoted by $E_{idle}^{(\mathrm{j})(\mathrm{j}+1)} = T_B \times P_{idle}$), it is better to keep the machine idle. The breakeven duration T_B is used to determine whether the machine is closed or not when the machine is idle. The T_B is defined as follows:

$$T_B = \frac{E_{sc}}{P_{idle}} = \frac{(P_{close} + P_{start}) \times T_{sc}}{P_{idle}} \tag{11}$$

where P_{close} and P_{start} is average power required to switch off/on machine and T_{sc} is duration time during turning off/on machine, so E_{sc} can be regarded as a constant value for a schedule. If T_{idle}

(T_{idle} = $s_{(j)(j+1)}$ = $S_{(j+1)} - C_{(j)}$, $j = 1, \ldots, n - 1$) is greater than T_B, it is favorable to turn the machine off, or vice versa.

Assuming that the energy levels are constant when job is processed on the machine, the energy consumption on the idle machine can be formulated as $E_{idle} = P_{idle} \times T_{makespan}$, where $T_{makespan}$ is a completion time for a schedule, namely makespan. Additional energy consumption to process the jobs is $E_{run} = P_{run} \times T_{run}$, where $T_{run} = \sum p_j^a$, $j = 1, \ldots, n$. The total energy saving for a given schedule can be defined as follows.

$$E_{save} = \sum_{j=1}^{n-1} \lambda_{(j)} \left[P_{idle} \times \left(S_{(j+1)} - C_{(j)} \right) - E_{sc} \right] \tag{12}$$

$$\lambda_{(j)} = \begin{cases} 1, \; if \; S_{(j+1)} - C_{(j)} \geq T_B \\ 0, \; otherwise \end{cases}$$

Thereby, the objective function of total energy consumption can be regarded as following formula.

$$\min f_3 = energy = E_{idle} + E_{run} \; - E_{save} \tag{13}$$

To further illustrate the total energy consumption by the turn off/on machine and operational method, Figure 1 gives an example with three jobs for the calculation of the total energy consumption, the area that the red line and the axis encircled expresses energy consumption. The job processing times and setup times are given in Table 1. The processing and idle energy power are 2 kW and 1 kW, respectively. The total energy required to startup and shutdown E_{sc} is 2 kWh. To briefly illustrate energy calculation, a fixed or actual processing time is employed in this scheduling problem considered, even though the job processing times are controllable. For a given job sequence $\pi = (J_1, J_3, J_2)$ as shown in Figure 1 the total idle energy consumption is $E_{idle} = P_{idle} \times T_{makespan} = 1 \times 11 = 11$. Additional energy consumption during processing the jobs is $E_{run} = P_{run} \times T_{run} = 2 \times 6 = 12$. According to Equation (12) the idle time T_{idle} between J_1 and J_3 ($T_{idle} = S_{(j+1)} - C_{(j)} = 2 - 1 = 1$) is smaller than a predefined breakeven duration $T_B = 2$, machine should keep running. Similarly, the idle time ($T_{idle} = 4$) between J_3 and J_2 is greater than $T_B = 2$, so machine should be switched off so as to save energy. The energy saved can be calculated as following formula. $E_{save} = 1 \times (1 \times (9 - 5) - 2) = 2$. The total energy consumption for this schedule is $f_3 = E_{idle} + E_{run} - E_{save} = 11 + 12 - 2 = 21$. Therefore, energy consumption is associated not only with the operational method but also closely with the turn on/off method. It can be reduced by switching off and on machine between scheduled tasks instead of leaving machine idle.

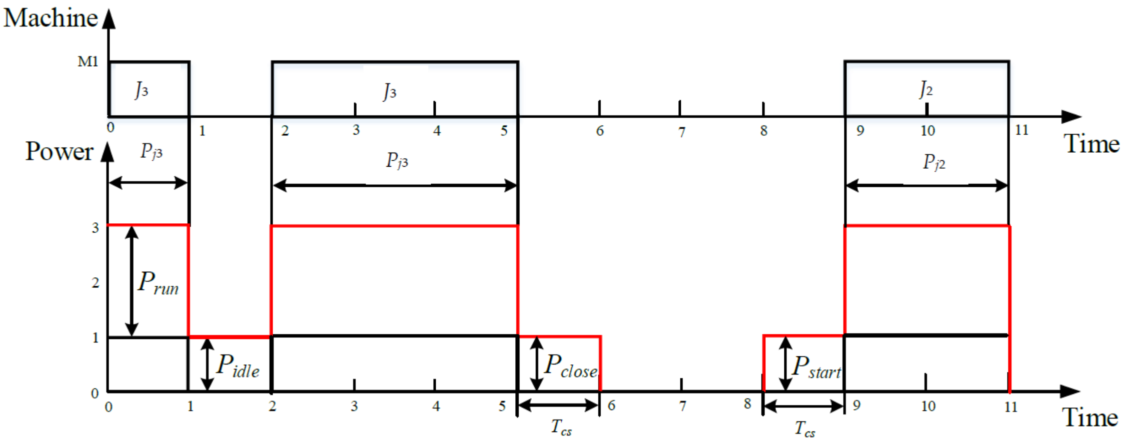


Figure 1. Gantt chart and its corresponding power profile. The area that the red line and the axis encircled expresses energy consumption.

Table 1. Three-job instance.

Job	Setup Times			Actual Processing Times
	1	**2**	**3**	
1	-	1	1	1
2	1	-	2	2
3	5	4	-	3

In summary, the three objective functions for the optimization of total E/T, cost, and total energy consumption in this scheduling model can be defined as follows.

$$\begin{array}{l} \min f(x) = [f_1(x), f_2(x), f_3(x)]^T \\ f_1(x) = ET \\ f_2(x) = cost \\ f_3(x) = energy \end{array} \tag{14}$$

4. The Proposed LMOEA for Multi-Objective Scheduling with Controllable Processing Times

In this section, firstly we state the basic background on multi-objective optimization, and then describe the framework of the proposed LMOEA. Finally, some main operators are presented in detail for optimizing the scheduling problem.

4.1. Background on Multi-Objective Optimization

To better understand our proposed LMOEA for solving the above problem, we begin with a brief introduction of basic concept of MOEA. Without loss of generality, we assume that there are several objectives to be minimized simultaneously. A multi-objective optimization problem (MOP) is composed of multiple conflicting objectives. In general, a MOP can be defined in the following form [44].

$$\begin{array}{l} \min f(x) = [f_1(x), f_2(x), \ldots, f_{\mathrm{k}}(x)]^{\mathrm{T}} \\ x \in \Omega \\ s.t. \begin{cases} g_i(x) \geq 0, i = 1, \ldots, \mathrm{q} \\ h_j(x) = 0, j = 1, \ldots, \mathrm{p} \end{cases} \end{array} \tag{15}$$

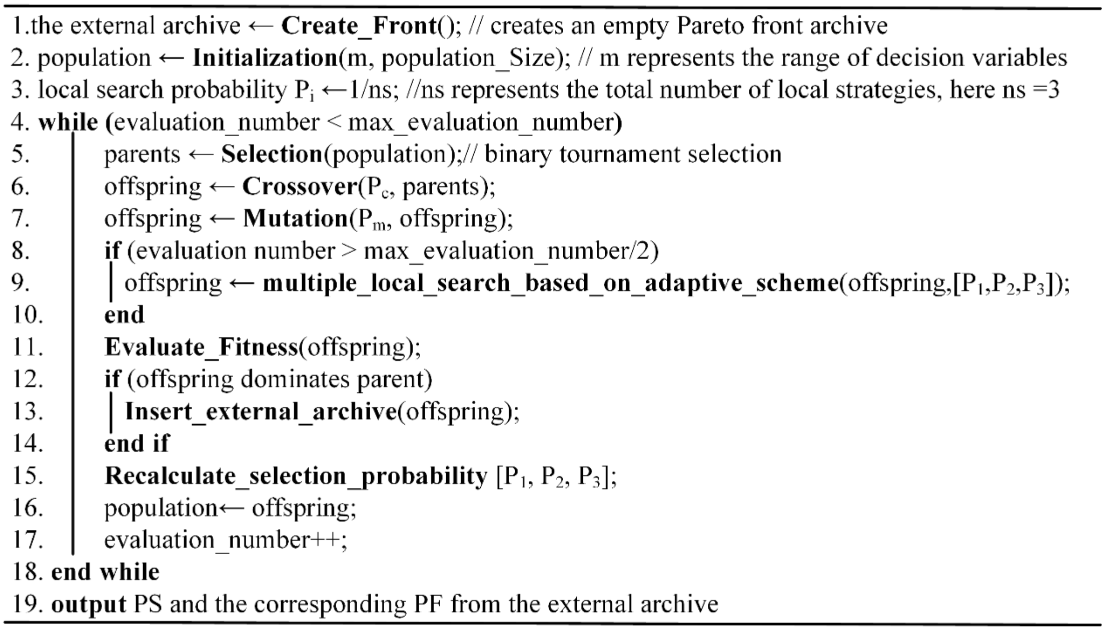
where x is called a decision vector and Ω is the search space. $f(x)$ constitutes k individual objective functions.

Let ***a*** and ***b*** ϵ Ω, a vector ***a*** = $(\mathrm{a}_1, \mathrm{a}_2, \ldots, \mathrm{a}_\mathrm{n})^T$ is said to dominate another vector ***b*** $(\mathrm{b}_1, \mathrm{b}_2, \ldots, \mathrm{b}_\mathrm{n})^T$ (denoted by $a \prec b$) if and only if $f_i(a) \leq f_i(b)$ for each i ϵ {1, ... , k} and $f_i(a) \leq f_i(b)$ for at least index j ϵ {1, ... , k}. A solution x* ϵ Ω is called a Pareto optimal solution in Equation (15) if there is not any a solution x ϵ Ω that dominates x*. The corresponding objective function is called Pareto optimal front vector f (x*). That is to say, for a Pareto optimal solution, the improvement of any objective must cause the deterioration of at least another objective. The set of all Pareto optimal solutions is called Pareto set (*PS**), while the set of all Pareto optimal front vectors is called the Pareto optimal front (*PF**). The main goal of multi-objective optimization is to find *PF**. However, in general a Pareto front consists of a large number of points. Therefore, a good Pareto front should contain a limited number of points which should be as close as possible to the *PF** and should be uniformly spread as well.

4.2. Framework of the Proposed Algorithm

As described in the previous section, the goal of this work is to develop a hybrid local MOEA (LMOEA) for above multi-objective scheduling problem. In the proposed LMOEA, a more suitable crossover and a more intensive local search are used to keep a better balance between the exploration and exploitation in search space.

We present two key strategies in the proposed LMOEA for solving this scheduling problem. (1) a new encoding scheme based on a real number to convert discrete problems into continuous problems; (2) the multiple local search strategies based self-adaptive selection scheme to make the candidate solution approach toward an optimal solution. The first strategy is used in solution representation to encode and decode solutions. The second strategy is performed on offspring solutions after crossover and mutation operation so as to exploit better solution in offspring's neighborhood. The pseudocode of the proposed LMOEA is shown in Figure 2.



```
1.the external archive ← Create_Front(); // creates an empty Pareto front archive
2. population ← Initialization(m, population_Size); // m represents the range of decision variables
3. local search probability $P_i$ ←1/ns; //ns represents the total number of local strategies, here ns =3
4. while (evaluation_number < max_evaluation_number)
5.      parents ← Selection(population);// binary tournament selection
6.      offspring ← Crossover($P_c$, parents);
7.      offspring ← Mutation($P_m$, offspring);
8.      if (evaluation number > max_evaluation_number/2)
9.          offspring ← multiple_local_search_based_on_adaptive_scheme(offspring,[$P_1$,$P_2$,$P_3$]);
10.     end
11.     Evaluate_Fitness(offspring);
12.     if (offspring dominates parent)
13.         Insert_external_archive(offspring);
14.     end if
15.     Recalculate_selection_probability [$P_1$, $P_2$, $P_3$];
16.     population← offspring;
17.     evaluation_number++;
18. end while
19. output PS and the corresponding PF from the external archive
```

Figure 2. The pseudocode of the proposed local multi-objective evolutionary algorithm (LMOEA).

Section 4.2.2 gives detailed illustration on line 6 of Figure 2. Section 4.2.3 also provides some explanations on line 7. Section 4.2.4 describes a concrete principle of the local search on line 8–10 in Figure 2. The calculation of each local search strategy rate on line 15 is also stated in Section 4.2.4. It is pointed out that the local search strategy takes much computation time during search process, even though it has a positive effect on the behavior of the LMOEA. So we execute the local search strategy in later stage (namely half of max function evaluation number) to establish a balance between the time consumption and search accuracy. Section 4.2.5 gives a detailed description on line 12–14. The external archive is used to store non-dominated solutions found so far. To maintain the diversity of the no-dominated solutions in the external archive, the crowding distance technique is adopted to discard the solutions with worse crowding distance when the number of non-dominated solutions exceeds the size of the external archive.

Additionally, there are three crucial differences between our proposal and the previous researches on the hybrid MOEAs.

- According to the no free lunch theorem, one algorithm cannot obtain a better result than all the others on all problems. Therefore, the proposed MOEA combines multiple techniques including harmony search (HS), genetic algorithm (GA), and differential evolution (DE) strategy. Even though the three nature-inspired algorithms are all population-based metaheuristics, they have different search manners or directions in search space, which can improve the diversity of the population and stability of the algorithm. The reasons why we adopt these three strategies in our proposal are explained in Section 3.2.1.
- The proposed algorithm is based on multiple techniques consisting of HS, GA, and DE, so it needs to use a new hybrid selection scheme to automatically select an appropriate operation

during the search process. This new hybrid selection mechanism is featured because the static and adaptive selection scheme can switch alternately. The aim of this hybrid selection mechanism is to avoid adopting the same search operator over the course of the iteration, which means the search process get trapped in the local optima. Therefore, by integrating the advantages of static and adaptive selection scheme, this hybrid selection can help the algorithm to mitigate the local optima and search for preferable solutions in unexplored areas.

In the following subsections, the ingredients of our proposed algorithm are detailed, including a new representation for this scheduling problem, crossover operator, mutation operator, the multiple local search approaches based on self-adaptive mechanism, and replacement strategy.

4.2.1. Representation

One of the important issues in applying MOEA to the considered problem lies in its solution representation. According to the feature of problem and algorithm considered, we present a new encoding scheme (solution representation) based on the real number for this scheduling problem in this work. This solution representation is different from literatures [45,46]. In [45], the length of solution representation developed by them is $2n$ (n is the number of jobs), while our proposal is n. In [46], they adopted the largest position value (LPV) rule to denote order of job process in solution representation. Whereas our representation contains more information. This encoding scheme we proposed has three advantages as follows.

(1) Chromosome structure is simpler. Even though this proposed encoding scheme only consists of one dimensional structure, it contains two contents namely amount of compression or expansion of job processing times and job sequence.

(2) Problem solving is simpler. Discrete optimization problems can be converted to continuous optimization problems by adoption of the proposed encoding scheme.

(3) Constraint handling mechanism is simpler even unnecessary. The algorithm with the conventional encoding scheme needs to employ some special recombination operators to avoid unfeasible solution. However, this approach with the proposed encoding mechanism can generate feasible solutions.

The main principle of this encoding scheme is as follows. For a given n jobs, n real values, denoting amount of compression or expansion and job sequence, are randomly created from a uniform distribution in its own range (m^c, m^e). The integer part, obtained by rounding off this real value, stands for the actual amount of compression or expansion of job processing times. To distinguish compression and expansion of job processing time, the positive integer represents actual amount of expansion and the negative integer denotes actual amount of compression. It is noted that the positive and negative number are generated with equal probability. The fractional part of a real number implies the processing order of jobs on a single machine. To further describe the proposed encoding scheme, Figure 3 illustrates the solution representation. For a given single machine scheduling problem with four jobs, we assume that its maximum amount of compression and expansion are $m^c = (-3, -4, -2, -4)$ and $m^e = (2, 3, 2, 4)$, respectively. One chromosome vector $(-1.325, 2.420, -1.761, 3.067)$ is randomly generated in the range of compression or expansion amount. This chromosome can be translated into two meanings: (1) actual amount vector of compression or expansion of job processing times $m = (-1, 2, -2, 3)$; (2) the sequence of jobs namely $\pi = (J_4, J_1, J_2, J_3)$. In detail, the integer part $(-1, 2, -2, 3)$ is obtained by rounding off the real number and represents actual amount of compression or expansion of job processing times. Among them, the integer value "-1" in the first position of the integer part means that amount of compression of J_1 is one unit time. Similarly, the value "3" in the fourth position denotes that amount of expansion of J_4 is three unit time. The fractional part (0.325, 0.420, 0.761, 0.067) implies job processing sequence. The fractional part numbers are sorted based on non-descending order, namely $0.067 < 0.325 < 0.420 < 0.761$. Accordingly, the corresponding job processing sequence is (J_4, J_1, J_2, J_3). Using this encoding scheme, it is easy to recognize the actual amount of compression or expansion of jobs and processing sequence of jobs. Therefore, this decoding mechanism is very simple and efficient based on the above analysis.

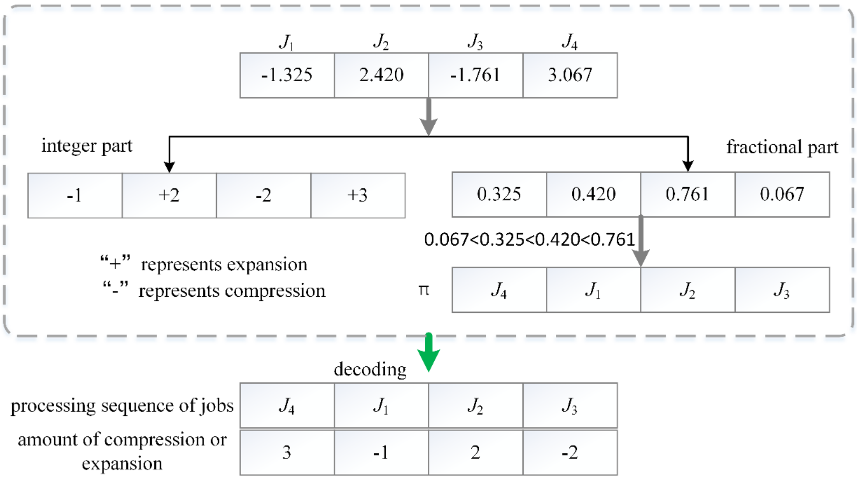


Figure 3. Solution representation.

4.2.2. Crossover Operator

Crossover is one of the most vital operators and is utilized to generate the new offspring solutions. It is reasonable that an efficient crossover operator not only update the population but also inherit elite genes of their parents to the offspring individuals. In the past years, many types of crossover operators have been proposed for solving continuous optimization problems, such as uniform crossover [47], simulated binary crossover (SBX) [48], blend crossover (BLX-α) [49], single point crossover, et al. As we know, arithmetic crossovers and permutation-based crossover are more efficient for discrete problems, but they are not suitable for the encoding scheme in this study. In addition, the effectiveness of the crossover operators depends on the specific problems. According to the experimental study on different crossovers for all test instances in Section 5.4, finally SBX is applied to our proposed LMOEA in this paper.

4.2.3. Mutation Operator

Mutation is also an important operator in MOEA, which can fine-tune some genes with a small probability. Moreover, it helps the algorithm to escape from local optimum as well. In this work, the mutation operator is composed of two kinds of mutation techniques. The goal of the first mutation technique is to adjust compression or expansion amount of job processing times in the range. The second mutation technique called swap mutation operator is used to swap order of jobs. The first mutation technique can achieve its objective by changing integer value in its range. The other technique can exchange the job processing order by swamping two genes' fractional parts. When performing a mutation operator on an individual, either the first mutation or the second mutation operator is chosen with the possibility of 0.5. To detail the process of this mutation operator, an example of how to implement the hybrid mutation is presented in Figure 4. For the first mutation operator as shown in Figure 4a, firstly a mutation position is selected from uniform distribution between 1 and 4. We assume that the second gene is chosen, and then a new integer is generated randomly from its range. For the second mutation operator as shown in Figure 4b, when the two genes are randomly selected out (e.g., the second and fourth gene), their fractional parts are swamped. More specifically, the original fractional values of the second and fourth genes are 0.420 and 0.067, respectively. The original sequence of jobs is $\pi = (J_4, J_1, J_2, J_3)$ according to the above proposed encoding scheme. After applying the second mutation operation, the fractional values of the second and fourth genes become 0.067 and 0.420, respectively. Accordingly, the sequence of jobs also becomes $\pi = (J_2, J_1, J_4, J_3)$ and the corresponding compression or expansion amount of job processing times is $m = (2, -1, 3, -2)$.

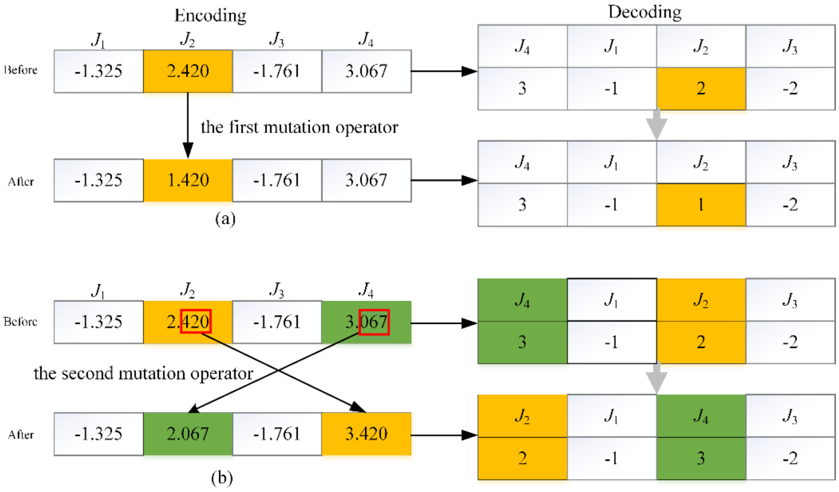


Figure 4. Mutation operator: (**a**) the first mutation operator; (**b**) the second mutation operator. The colored cell denotes mutation position, the red box denotes the job exchange part.

4.2.4. Local Search Procedure

A local search strategy can effectively enhance exploitation ability of the algorithm. In this research, we present three different local search strategies based on self-adaptive selection mechanism to obtain better approximate solutions. For a given schedule, in Strategy 1 (S1), we insert the fractional part randomly chosen into another job position with the biggest setup time to reduce the energy consumption. Strategy 2 (S2) is another local search technique that may minimize the total tardiness/earliness by swapping the largest lateness or earliness of a job with the smaller one. Strategy 3 (S3) can be used to decrease cost by approaching the actual processing times closer toward the corresponding normal processing times. The flow chart of the multiple local search is presented in Figure 5.

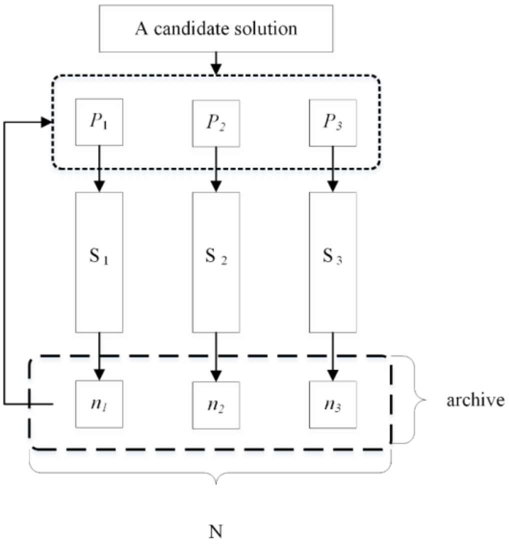


Figure 5. Flow chart of multiple local search.

To select an appropriate local search operator for a given schedule, a self-adaptive selection mechanism is utilized. Selection probability of each local search strategy can be obtained by recording the kind of local operator performed by each solution. In initialization, each local search strategy is chosen with the possibility of 1/3. Afterward, when a local search strategy is chosen, the strategy will be assigned to a new solution. Let P_i denote the selection probability of the *i*th strategy and P_1 is probability for S1, P_2 for S2, P_3 for S3. The self-adaptive selection mechanism can be described as follows. Firstly, if the external archive storing non-dominated solution found so far is updated, compute the selection rate of each local strategy $P_i = n_i / |\mathrm{N}|$, where n_i is the number of non-dominated solutions obtained by the *i*th local strategy in the external archive. |N| represents the current size of external archive. Then select a strategy by using the roulette wheel approach. To avoid this situation of all candidate solutions adopting the same local search operator during search process, each local strategy will be assigned with the possibility of 1/3 when this situation occurs. Based on the above analysis, this multiple local operator based on self-adaptive mechanism is very simple and effective for solving complex problems. We also investigate the effect of the hybridization of local search on performance of the proposed algorithm in Section 5.6.

4.2.5. Replacement Strategy

Multi-objective optimization is different from mono-objective optimization. In multi-objective optimization, each solution is associated with a rank equal to its non-dominance level (e.g., 1 for the best level, 2 for the second best level and so on). Then within each level or rank, a crowding distance technique, which indicates the sum of distances to the closest individual along each objective, is used to define an ordering among solutions. To achieve wide spread of the obtained Pareto fronts, a trial solution with large crowding distance is preferred to one with small crowding distance. Our replacement method is based on rank and crowding distance. That is, replace the solution in the external archive if it is dominated by the offspring or both are non-dominate level and the one has the worst crowding distance in the population consisting of solutions from external archive plus offspring. Otherwise the offspring will be discarded. However, at the beginning of update, the external archive is empty and the *PS* in initial population is directly inserted into the archive as presented in Figure 6.

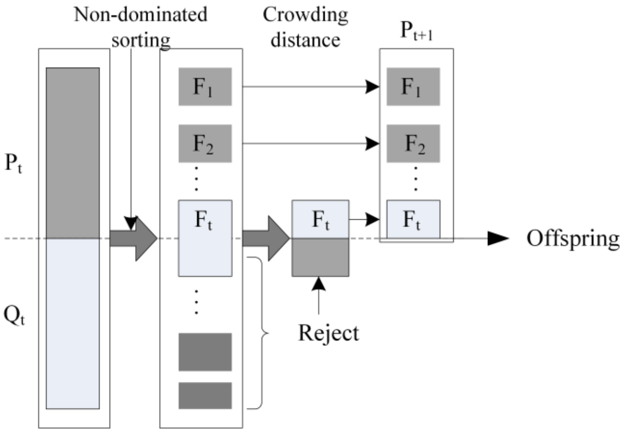


Figure 6. Construction of population P_{t+1}.

5. Experimental Studies

In the section we conduct a set of computational experiments to evaluate the performance of the proposed LMOEA, which was coded in java and run on a Intel Core i5 1.6 GHz PC with 1 GB memory.

This section is devoted to measuring the performance of the proposed algorithm LMOEA for the addressed problems. In this section, the empirical studies contain the following four aspects.

(1) Efficiency comparisons with different crossover in LMOEA for all problems in Section 5.4.

(2) The best choice of crossover rate and mutation rate for all test instances in Section 5.5.

(3) Performance analysis on the local search strategy of the LMOEA for the considered problems in Section 5.6.

(4) Performance comparisons with other MOEAs for the scheduling problems in Section 5.7.

In the following subsections, test problems, performance metrics and parameter settings are described at first, and then the experimental studies are further investigated step by step.

5.1. Test Instances

Three levels of problem size (i.e., small, medium and large) depending on the number of jobs and stages are considered in this experiment. The due date of each job is $d_j = k \times p_j, j = 1, 2, \ldots, n$ where k is control factor. The value of k will be set as $1, 2, \ldots, 5$ which corresponds to the trend of less tight due dates. The time unit is minute. The other parameters of data are shown in Table 2. The following tables record the results obtained by different MOEAs for each instance, where the instance with n jobs and k parameter is denoted as symbol "Problem_n_k". For example, "Problem_10_1" represents the addressed problem is featured by 10 jobs and k equal to 1.

Table 2. Data set distribution.

Input Variables	Value
number of jobs (n)	10, 50, 100
normal processing time (p_j)	~DU (20, 80)
crash processing time (p_j^c)	~DU ($-p_j/3$, 0)
expansion processing time (p_j^e)	~DU (0, $p_j/5$)
due time (d_j)	$d_j = k \times p_j$, k = 1, 2, 3, 4, 5.
unit penalty cost of compression (a_j)	~U (0.5, 2.5)
unit penalty cost of expansion (b_j)	~U (0.5, 2.5)
penalty factor of earliness and tardiness(e_j or t_j)	0.1
power during idle (p_{idle})	2.2 kW
additional power when processing job (p_{run})	7.5 kW
total energy when turning off and on machine (E_{sc})	6.72 kWh
setup times between adjacent jobs (s_{ij})	~DU (0, 10)

5.2. Performance Metrics

To measure the Pareto front (PF) obtained by these proposed algorithms, some metrics such as Spread [44,50], Generational Distance (GD) [51], and Inverse Generational Distance (IGD) [52] should be employed as below.

(1) Spread. The metric is a diversity indicator that measures the extent of spread achieved among the front found. The definition of Spread in [44] was used for bi-objective problems. As to problems with three or more objectives, the modified Spread is different, as it given in Equation (16) [50]. The metric is defined as:

$$Spread = \frac{\sum_{j=1}^{N-0} df_j + \sum_{i=1}^{n_{PF}} \left| d'_i - \overline{d'} \right|}{\sum_{j=1}^{N-0} df_j + n_{PF} \times \overline{d'}} \tag{16}$$

where df_j denotes the Euclidean distance of each point in PF to its closest point in PF*, n_{PF} is the number of vectors in PF*, d'_i is the Euclidean distance between the extreme solutions in the ith objective and the boundary solutions of the PF* obtained, and $\overline{d'}$ is the average of all d'_i. N is the number of PF* found. If the metric value is zero, then all the members of Pareto-optimal front are evenly spaced. A smaller value of the metric indicates a better distribution and diversity.

(2) Generational Distance (GD). The GD metric indicates how far the PF found is from the PF*. This metric is formulated as:

$$GD = \sqrt{\sum_{i=1}^{|EXA|} l_i^2} / |EXA| \quad (17)$$

where $|EXA|$ is the number of Pareto front points found so far, l_i is the Euclidean distance between the ith member of *PF* obtained and the nearest member of the *PF**. A low GD value is desirable, which denotes a good convergence performance.

(3) Inverse Generational Distance (IGD). It is a variant of the GD but represents a combined or comprehensive indicator. It measures the distances between each solution consisting of the optimal Pareto front and obtained front. It can be defined as follows:

$$IGD = \sqrt{\sum_{i=1}^{N} Dist_i^2} / H \quad (18)$$

where H is the number of the optimal Pareto front and $Dist_i$ is the Euclidean distance between each point of that front and the nearest member of the approximation. Fronts with lower IGD value are desirable.

It should be noted that the true *PF** of the considered problem maybe is unknown, so we regard the Pareto front of all the algorithms in all runs as the Pareto optimal front.

5.3. Parameters Settings

For the proposed LMOEA, the population size and external size are set to 100 and the number of evaluation function is set to 25,000. 30 independent runs are conducted for each algorithm on each test instance.

The optimal results are highlighted with bold in the tables. Since all candidate MOEAs are stochastic algorithms, the following statistical analysis is necessary for providing confidential comparisons. Firstly, the kolmogorov-smirnov test is conducted to check whether the results follow the normal distribution or not. If the results are not normal distributed, the *t*-test will be used to test the result of each algorithm; otherwise, a Wilcoxon rank sum test is applied to the mean values. The two statistical tests executed on the pairs of the algorithms are to measure the significant differences between the results obtained by different algorithms. The confidence level for all tests is set to 95% (corresponding to a *p*-value below 0.05). The sign "+" indicates that our proposed LMOEA algorithm performs significantly better than the second best algorithm on average. While "−" represents that the performance of LMOEA algorithm is significantly worse than that of the best algorithm. The "=" sign denotes that there is no significant difference between LMOEA algorithm and the best or second best MOEA. The bottom rows of these tables list the ratio of dominant performance for the corresponding metric. For example, 3/15 indicates the corresponding algorithm outperform its rivals on 3 out of 15 problems for a given metric.

Before solving these scheduling instances, we should select an appropriate crossover operator, crossover and mutation rate for LMOEA in the following Sections 5.4 and 5.5.

5.4. Comparison among Different Crossovers

To determine an appropriate crossover operator among SBX, BLX-α, uniform crossover, and single point crossover, we execute experiments on all instances. There are four versions of LMOEA only adopting SBX, BLX-α, uniform crossover and single point crossover operator. The crossover and mutation rate in these four versions are set to 0.9 and $1/n$ (n is the number of jobs), respectively. We conduct 30 independent runs for each MOEA on each instance and compare average value of IGD, GD, and Spread metric over all runs for different crossover operators as shown in Figure 7. According to Section 5.2, the smaller the metric is, the better the performance of the algorithm is. We can clearly observe that the proposed LMOEA with only SBX outperforms its counterparts in terms of the IGD, GD, and Spread metric value. Therefore, the SBX (crossover and mutation distribution index are both 20) operator, regarded as a crossover operator, is introduced into the LMOEA for solving these scheduling instances.

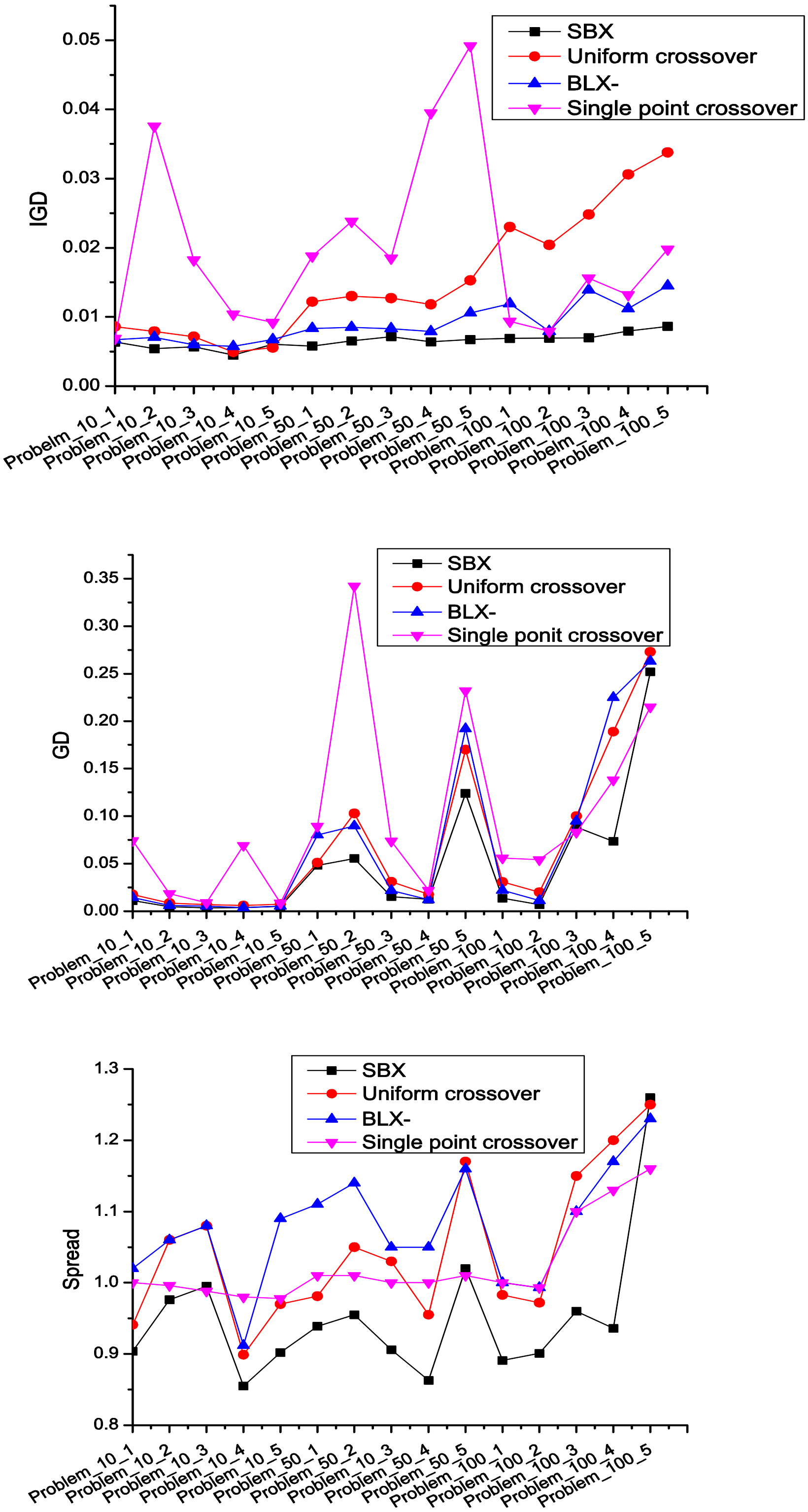


Figure 7. Average Inverse Generational Distance (IGD), Generational Distance (GD), and Spread value for different crossovers on all test instances.

5.5. The Best Choice of Crossover and Mutation Probability

The quality of the algorithm is significantly influenced by the value of crossover and mutation rate. In this subsection we study the performance of the different parameters of LMOEA on all test instances.

5.5.1. Selection of Crossover Probability

The crossover probability denoted by symbol P_c, it is generated from 0 to 1.0 in increments of 0.1, and the mutation probability P_m is $1/n$ (n is the number of jobs) in this experiment. The IGD metric is utilized to test the performance of the proposed algorithm for different P_c on each instance since this IGD metric is a comprehensive indicator. We conduct 30 independent runs for each crossover rate on each test instance. The IGD trajectories against different P_c are plotted in Figure 8. The Y axis of each subfigure represents the IGD value and the X axis denotes the different P_c. The red circle indicates the mean IGD value for a crossover rate combination. It can be observed that the IGD curves of small size problems with 10 jobs are stable with the increase of P_c, which implies that the performance of the proposed algorithm is not sensitive to the P_c. It is observed from Figure 8 that the algorithm is more stable with the decreasing of the box's length. In terms of descriptive statistics, a box plot is a convenient way of graphically depicting groups of numerical data through their quartiles. Box plots may have lines extending vertically from the boxes indicating variability outside the upper and lower quartiles, hence the terms box plot diagram. Outliers maybe plotted as individual points. Box plots are non-parametric: its display variation in samples of a statistical population without making any assumptions of the underlying statistical distribution. The spacing between the different parts of the box indicate the degree of dispersion (spread) and skewness in the data, and show outliers. In addition to the points of themselves, they allow one to visually estimate various L-estimators, notably the interquartile range, midhinge, range, mid-range, and trimean. Box plots can be drawn either horizontally or vertically. Concerning medium and large scale problems, the IGD value gradually decreases with the increase of P_c. It means that crossover with a large probability can help to improve the performance of the proposed algorithm for coping with these scheduling problems. Therefore, the range (0.7, 1) for P_c is recommended for decision makers when solving these scheduling instances.

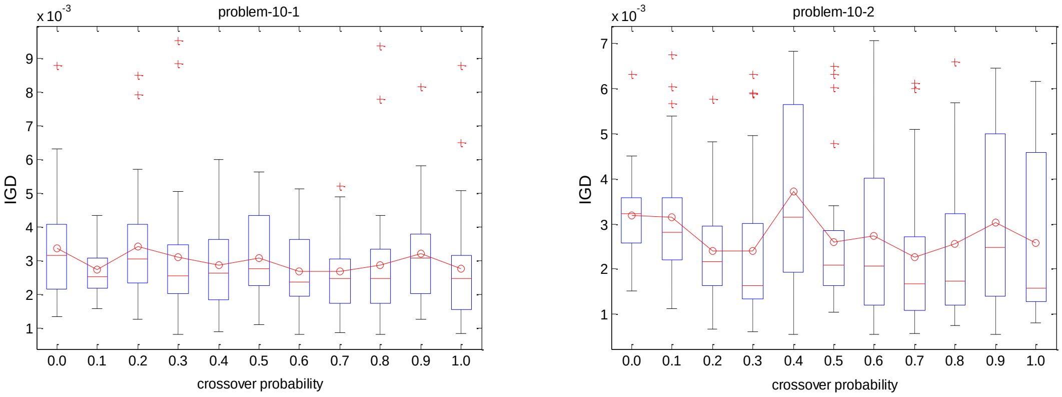


Figure 8. *Cont.*

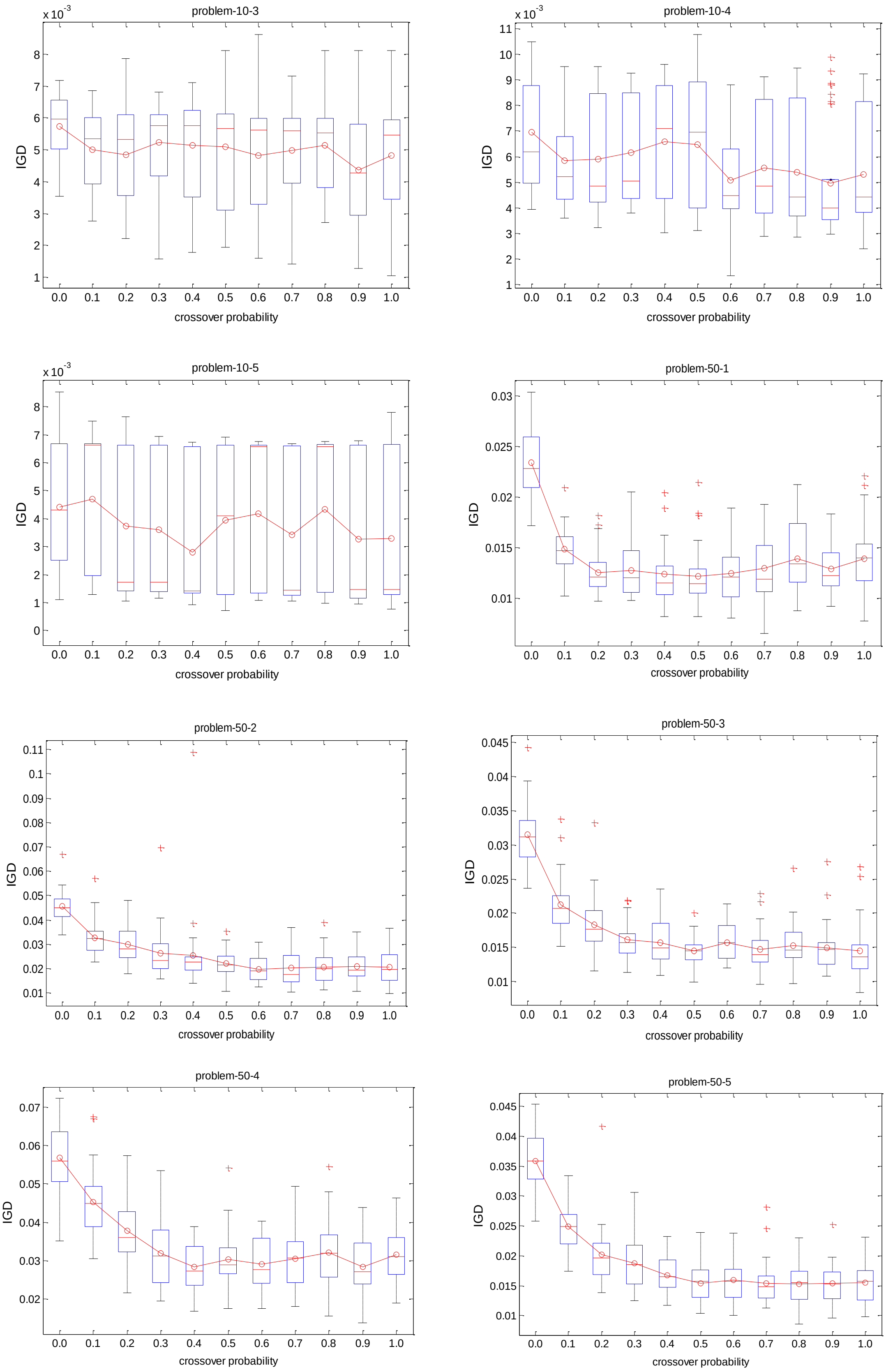


Figure 8. *Cont.*

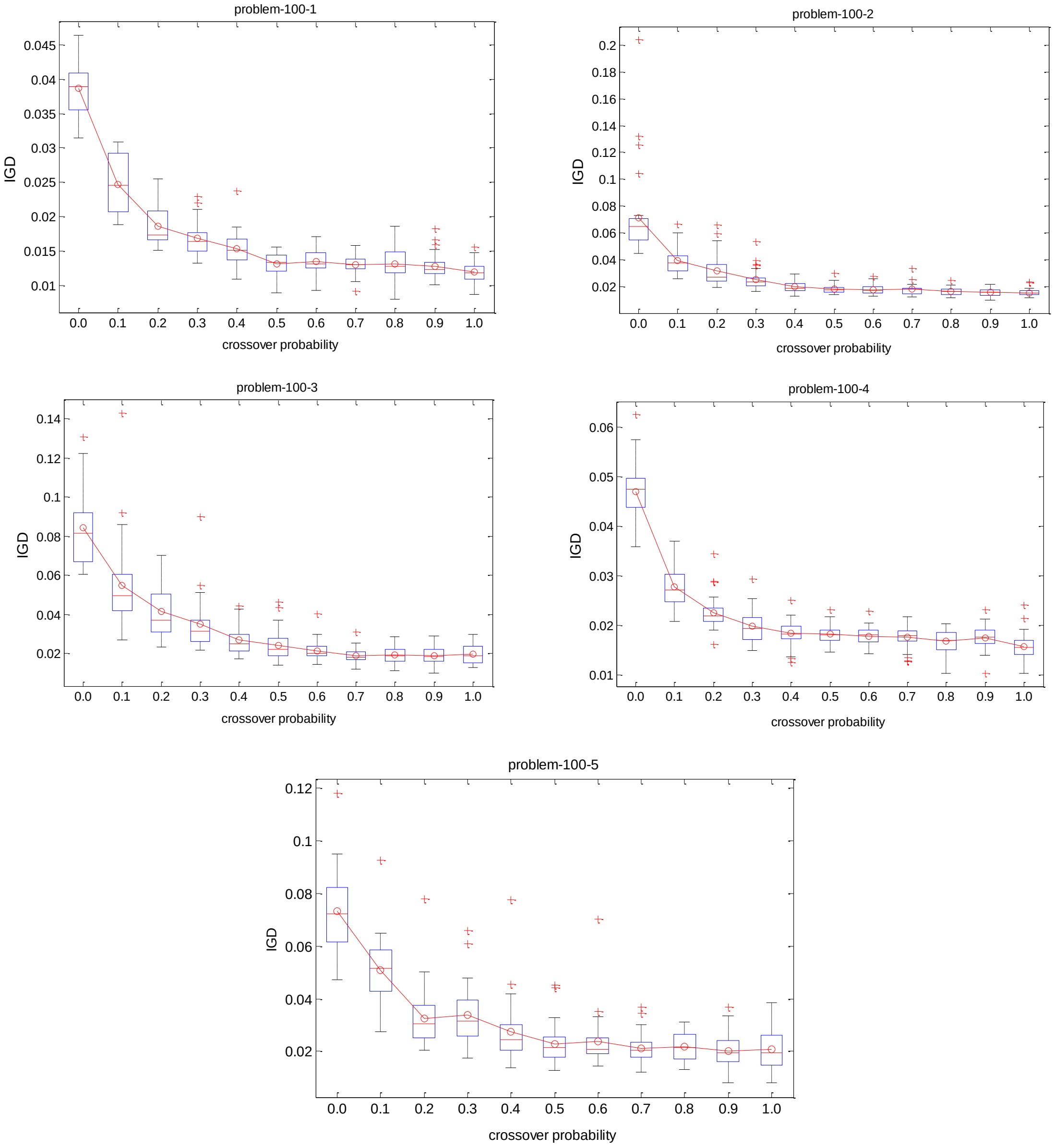


Figure 8. Sensitivity study on crossover probability. The red circle indicates the mean IGD value for a crossover rate combination, the red + denote the abnormal value of the IGD.

5.5.2. Selection of Mutation Probability

We further investigate these scheduling problems affected by mutation probability in this experiment. Here, crossover rate P_c is fixed as 0.9 and mutation rate P_m is updated from 0 to 0.1 with step of 0.01. We can run the same type of experiment to determine the optimal mutation probability. The mean IGD results are plotted in Figure 9. The vertical axis of each subfigure denotes the IGD value and the horizontal axis represents the variation of P_m. These scheduling problems can be divided into two categories. The first one is characterized as that a relatively stable IGD trend, indicating that the value of P_m has little effect on the performance of the proposed algorithm. The instances with 10 jobs and problem_100_4 belong to this category. The second category, which consists of the rest medium and large instances except problem_100_4, is featured by a relatively high IGD value at the beginning, and then the IGD value decreases to a low level when P_m is 0.04 and remains stable later. From the Figure 9, the value 0.1 for P_m is recommended.

In the following empirical studies, crossover and mutation probability in the proposed algorithm are set to 0.9 and 0.1, respectively.

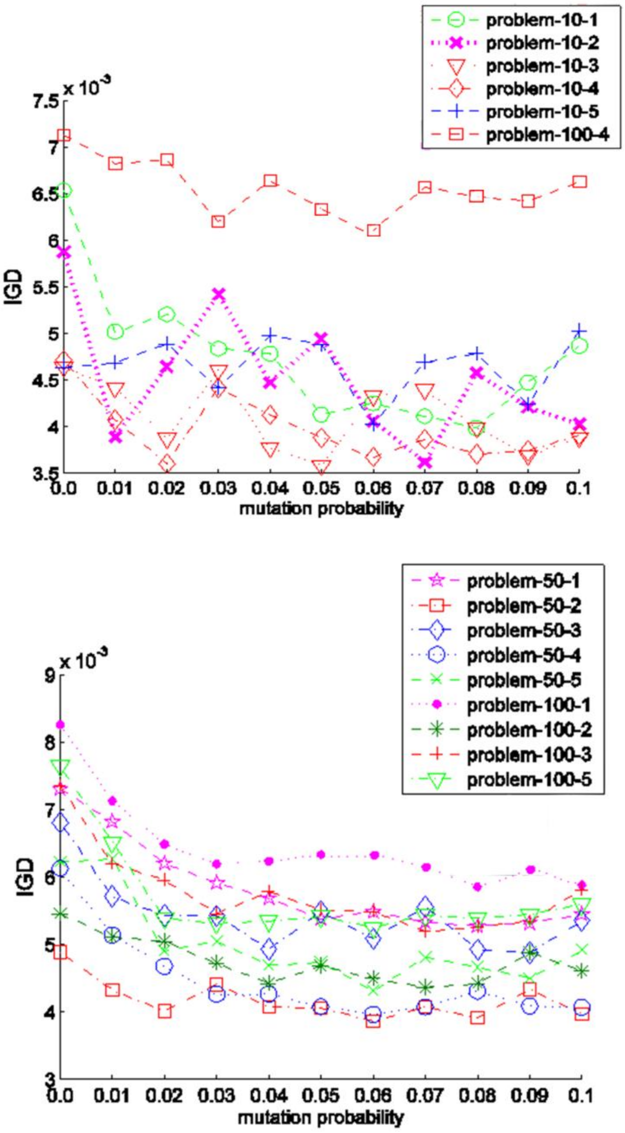


Figure 9. Sensitivity study on mutation probability.

5.6. Efficiency of Each Local Search Strategy

To prove the effectiveness of multiple local search strategies based on self-adaptive mechanism in LMOEA, we generate three variations of the LMOEA by adopting only one local search strategy and compare the behavior of each local strategy on the 15 problems. In this experiment, we use MOEA$_{S1}$ to denote LMOEA with only S1. Similarly, MOEA$_{S2}$ indicates LMOEA using only S2, and MOEA$_{S3}$ represents LMOEA with only S3. Tables 3–5 show mean and standard deviation metrics on different algorithms in 30 independent runs.

From Tables 3–5 it can be observed that MOEA$_{S3}$ obtains significantly better results on 9 out of 15 problems with comparison to MOEA$_{S1}$ and MOEA$_{S2}$ in terms of GD metric, while MOEA$_{S1}$ and MOEA$_{S2}$ can achieve the better GD values on two and four problems, respectively. Concerning Spread metric, the MOEA$_{S1}$ performs better than the MOEA$_{S2}$ and MOEA$_{S3}$ for most instances. However, MOEA$_{S2}$ is superior to MOEA$_{S1}$ and MOEA$_{S3}$ with regard to IGD metric. Therefore, it can be concluded that none of the three local strategies has a dominant performance for all instances.

When these three local search strategies are incorporated into our proposed LMOEA, we can observe that the proposed LMOEA wins the best IGD value on 11 out of 15 test instances. It means that the multiple strategies based on self-adaptive scheme can help to improve comprehensive performance (convergence and diversity). With regard to the GD metric, the LMOEA obtains the best value on 13 out of 15 test problems, which suggests our proposal can generate better *PF* toward *PF** than the others with only single local strategy. That is to say, multiple local strategies based on self-adaptive scheme can enhance search efficiency since the candidate solution can select a proper local strategy to search optimal solutions. The proposed LMOEA with the multiple local search strategy is also superior to its three variations in terms of Spread metric.

Therefore, we can conclude that combination of these local search strategies can help to improve the search diversity and the selection of appropriate local operators can enhance the search efficiency. Besides, the results computed by the proposed LMOEA are more stable, which implies that the integration of multiple local search techniques based on self-adaptive mechanism can strengthen the robustness of the LMOEA.

Table 3. Mean and standard deviation of GD obtained by different MOEAs with different local search strategies.

Problem	**MOEA$_{S1}$**	**MOEA$_{S2}$**	**MOEA$_{S3}$**	**LMOEA**
	Mean/Std.	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**
Problem_10_1	$3.21 \times 10^{-2}/5.2 \times 10^{-2}$	$6.79 \times 10^{-3}/3.9 \times 10^{-3}$	$\mathbf{5.01 \times 10^{-3}/3.6 \times 10^{-3}}$	$6.47 \times 10^{-3}/8.0 \times 10^{-3}$ −
Problem_10_2	$4.45 \times 10^{-3}/3.9 \times 10^{-3}$	$7.52 \times 10^{-3}/7.3 \times 10^{-3}$	$6.38 \times 10^{-3}/4.2 \times 10^{-3}$	$\mathbf{4.26 \times 10^{-3}/1.5 \times 10^{-3}}$ +
Problem_10_3	$2.99 \times 10^{-2}/1.7 \times 10^{-2}$	$1.79 \times 10^{-2}/1.2 \times 10^{-2}$	$1.81 \times 10^{-2}/1.2 \times 10^{-2}$	$\mathbf{3.94 \times 10^{-3}/1.2 \times 10^{-2}}$ +
Problem_10_4	$1.21 \times 10^{-2}/3.8 \times 10^{-3}$	$1.48 \times 10^{-2}/1.2 \times 10^{-2}$	$1.69 \times 10^{-2}/1.3 \times 10^{-2}$	$\mathbf{4.44 \times 10^{-3}/1.1 \times 10^{-2}}$ +
Problem_10_5	$1.81 \times 10^{-2}/1.6 \times 10^{-2}$	$8.54 \times 10^{-3}/3.1 \times 10^{-3}$	$9.44 \times 10^{-3}/4.8 \times 10^{-3}$	$\mathbf{4.93 \times 10^{-3}/3.1 \times 10^{-3}}$ +
Problem_50_1	$4.73 \times 10^{-1}/8.4 \times 10^{-1}$	$2.42 \times 10^{-1}/6.4 \times 10^{-1}$	$8.27 \times 10^{-2}/1.9 \times 10^{-1}$	$\mathbf{7.07 \times 10^{-2}/1.3 \times 10^{-1}}$ +
Problem_50_2	$9.04 \times 10^{-1}/9.0 \times 10^{-1}$	$8.36 \times 10^{-2}/2.9 \times 10^{-1}$	$1.88 \times 10^{-2}/3.9 \times 10^{-1}$	$\mathbf{7.38 \times 10^{-2}/1.5 \times 10^{-1}}$ +
Problem_50_3	$1.12 \times 10^{-1}/1.2 \times 10^{-1}$	$3.52 \times 10^{-2}/7.8 \times 10^{-2}$	$1.34 \times 10^{-2}/2.0 \times 10^{-2}$	$\mathbf{4.11 \times 10^{-3}/3.0 \times 10^{-3}}$ +
Problem_50_4	$4.90 \times 10^{-1}/6.2 \times 10^{-1}$	$1.01 \times 10^{-1}/2.2 \times 10^{-1}$	$9.53 \times 10^{-2}/2.5 \times 10^{-1}$	$\mathbf{3.67 \times 10^{-3}/3.9 \times 10^{-2}}$ +
Problem_50_5	$4.64 \times 10^{-1}/9.3 \times 10^{-1}$	$1.94 \times 10^{-1}/4.1 \times 10^{-1}$	$\mathbf{1.59 \times 10^{-1}/3.3 \times 10^{-1}}$	$1.60 \times 10^{-1}/5.3 \times 10^{-1}$ −
Problem_100_1	$1.48 \times 10^{-1}/1.2 \times 10^{-1}$	$2.49 \times 10^{-2}/4.6 \times 10^{-2}$	$2.38 \times 10^{-2}/4.3 \times 10^{-2}$	$\mathbf{1.56 \times 10^{-2}/2.4 \times 10^{-2}}$ +
Problem_100_2	$1.01/1.2 \times 10^{-1}$	$9.80 \times 10^{-1}/1.5$	$4.38 \times 10^{-1}/8.6 \times 10^{-1}$	$\mathbf{1.53 \times 10^{-2}/3.4 \times 10^{-1}}$ +
Problem_100_3	$1.12/2.2 \times 10^{-1}$	$7.15 \times 10^{-1}/1.5$	$4.30 \times 10^{-1}/1.2$	$\mathbf{7.56 \times 10^{-2}/6.4 \times 10^{-2}}$ +
Problem_100_4	$1.08/2.3 \times 10^{-1}$	$5.50 \times 10^{-1}/1.1$	$2.66 \times 10^{-1}/7.2 \times 10^{-1}$	$\mathbf{6.12 \times 10^{-2}/8.2 \times 10^{-2}}$ +
Problem_100_5	$6.89 \times 10^{-1}/7.5 \times 10^{-1}$	$3.43 \times 10^{-1}/7.0 \times 10^{-1}$	$5.34 \times 10^{-1}/1.10$	$\mathbf{2.27 \times 10^{-1}/2.8 \times 10^{-1}}$ +
Hit rate	0/15	0/15	2/15	13/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

Table 4. Mean and standard deviation of Spread obtained by different MOEAs with different local search strategies.

Problem	**MOEA$_{S1}$**	**MOEA$_{S2}$**	**MOEA$_{S3}$**	**LMOEA**
	Mean/Std.	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**
Problem_10_1	$1.03/1.5 \times 10^{-1}$	$1.12/1.5 \times 10^{-1}$	$1.11/1.8 \times 10^{-1}$	$\mathbf{9.03 \times 10^{-1}/1.3 \times 10^{-2}}$ +
Problem_10_2	$\mathbf{1.06/1.2 \times 10^{-1}}$	$1.26/2.1 \times 10^{-1}$	$1.25/1.8 \times 10^{-1}$	$1.09/1.6 \times 10^{-1}$ −
Problem_10_3	$\mathbf{1.01/6.5 \times 10^{-2}}$	$1.13/1.5 \times 10^{-1}$	$1.21/1.5 \times 10^{-1}$	$1.03/1.3 \times 10^{-1}$ −
Problem_10_4	$1.32/1.1 \times 10^{-1}$	$1.42/1.4 \times 10^{-1}$	$1.36/1.3 \times 10^{-1}$	$\mathbf{8.45 \times 10^{-1}/5.3 \times 10^{-2}}$ +
Problem_10_5	$1.02/6.3\times 10^{-2}$	$1.25/1.6 \times 10^{-1}$	$1.23/1.7 \times 10^{-1}$	$\mathbf{9.11 \times 10^{-1}/6.2 \times 10^{-2}}$ +
Problem_50_1	$1.24/1.7 \times 10^{-1}$	$1.16/1.5 \times 10^{-1}$	$\mathbf{9.02 \times 10^{-1}/1.9 \times 10^{-1}}$	$9.50 \times 10^{-1}/2.2 \times 10^{-2}$ −
Problem_50_2	$1.25/1.8 \times 10^{-1}$	$1.11/2.1 \times 10^{-1}$	$1.11/2.3 \times 10^{-1}$	$\mathbf{1.01/2.1 \times 10^{-1}}$ +
Problem_50_3	$1.12/1.6 \times 10^{-1}$	$1.04/1.6 \times 10^{-1}$	$1.00/1.5 \times 10^{-1}$	$\mathbf{9.63 \times 10^{-1}/6.7 \times 10^{-2}}$ +
Problem_50_4	$1.17/1.4 \times 10^{-1}$	$1.02/2.1 \times 10^{-1}$	$1.06/1.6 \times 10^{-1}$	$\mathbf{9.02 \times 10^{-1}/8.4 \times 10^{-2}}$ +
Problem_50_5	$1.23/1.6 \times 10^{-1}$	$1.14/2.0 \times 10^{-1}$	$1.08/1.8 \times 10^{-1}$	$\mathbf{1.05/1.7 \times 10^{-1}}$ +
Problem_100_1	$9.57 \times 10^{-1}/2.9 \times 10^{-1}$	$9.81 \times 10^{-1}/2.0 \times 10^{-1}$	$9.66 \times 10^{-1}/1.8 \times 10^{-1}$	$\mathbf{9.15 \times 10^{-1}/6.3 \times 10^{-2}}$ +
Problem_100_2	$\mathbf{9.01 \times 10^{-1}/3.8 \times 10^{-1}}$	$1.21/2.1 \times 10^{-1}$	$1.15/2.0 \times 10^{-1}$	$1.01/2.4 \times 10^{-1}$ −
Problem_100_3	$1.35/1.4 \times 10^{-1}$	$1.18/1.7 \times 10^{-1}$	$1.13/2.1 \times 10^{-1}$	$\mathbf{9.16 \times 10^{-1}/1.7 \times 10^{-1}}$ +
Problem_100_4	$1.37/1.4 \times 10^{-1}$	$1.26/1.5 \times 10^{-1}$	$1.14/2.1 \times 10^{-1}$	$\mathbf{9.55 \times 10^{-1}/8.5 \times 10^{-2}}$ +
Problem_100_5	$1.30/1.3 \times 10^{-1}$	$\mathbf{1.12/1.9 \times 10^{-1}}$	$1.15/2.2 \times 10^{-1}$	$1.22/1.5\times 10^{-1}$ −
Hit rate	3/15	1/15	1/15	10/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

Table 5. Mean and standard deviation of IGD obtained by different MOEAs with different local search strategies.

Problem	**MOEA$_{S1}$**	**MOEA$_{S2}$**	**MOEA$_{S3}$**	**LMOEA**
	Mean/Std.	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**
Problem_10_1	$5.61 \times 10^{-3}/3.1 \times 10^{-3}$	$\mathbf{4.61 \times 10^{-3}/2.2 \times 10^{-3}}$	$4.94 \times 10^{-3}/2.3 \times 10^{-3}$	$5.57 \times 10^{-3}/2.4 \times 10^{-3}$ −
Problem_10_2	$5.21 \times 10^{-3}/2.8 \times 10^{-3}$	$5.33 \times 10^{-3}/2.5 \times 10^{-3}$	$5.34 \times 10^{-3}/3.1 \times 10^{-3}$	$\mathbf{4.55 \times 10^{-3}/2.6 \times 10^{-3}}$ +
Problem_10_3	$7.30 \times 10^{-3}/1.8 \times 10^{-3}$	$6.00 \times 10^{-3}/1.7 \times 10^{-3}$	$6.24 \times 10^{-3}/1.9 \times 10^{-3}$	$\mathbf{5.97 \times 10^{-3}/1.7 \times 10^{-3}}$ +
Problem_10_4	$9.06 \times 10^{-3}/3.7 \times 10^{-3}$	$7.91 \times 10^{-3}/4.1 \times 10^{-3}$	$8.68 \times 10^{-3}/3.9 \times 10^{-3}$	$\mathbf{6.87 \times 10^{-3}/3.8 \times 10^{-3}}$ +
Problem_10_5	$5.69 \times 10^{-3}/3.4 \times 10^{-3}$	$5.66 \times 10^{-3}/3.5 \times 10^{-3}$	$\mathbf{5.35 \times 10^{-3}/3.4 \times 10^{-3}}$	$5.79 \times 10^{-3}/3.4 \times 10^{-3}$ −
Problem_50_1	$4.52 \times 10^{-2}/8.5 \times 10^{-3}$	$2.11 \times 10^{-2}/7.5 \times 10^{-3}$	$1.93 \times 10^{-2}/5.8 \times 10^{-3}$	$\mathbf{7.46 \times 10^{-3}/3.7 \times 10^{-3}}$ +
Problem_50_2	$3.56 \times 10^{-2}/5.2 \times 10^{-3}$	$2.09 \times 10^{-2}/7.7 \times 10^{-3}$	$2.45 \times 10^{-2}/9.8 \times 10^{-3}$	$\mathbf{7.60 \times 10^{-3}/4.5 \times 10^{-3}}$ +
Problem_50_3	$1.24 \times 10^{-2}/1.5 \times 10^{-3}$	$1.34 \times 10^{-2}/3.2 \times 10^{-3}$	$1.33 \times 10^{-2}/4.7 \times 10^{-3}$	$\mathbf{7.53 \times 10^{-3}/3.2 \times 10^{-3}}$ +
Problem_50_4	$2.98 \times 10^{-2}/4.1 \times 10^{-3}$	$1.53 \times 10^{-2}/3.1 \times 10^{-3}$	$1.62 \times 10^{-2}/3.1 \times 10^{-3}$	$\mathbf{7.58 \times 10^{-3}/2.2 \times 10^{-3}}$ +
Problem_50_5	$5.47 \times 10^{-2}/1.2 \times 10^{-2}$	$2.27 \times 10^{-2}/9.4 \times 10^{-3}$	$2.52 \times 10^{-2}/9.1 \times 10^{-3}$	$\mathbf{8.03 \times 10^{-3}/6.8 \times 10^{-3}}$ +
Problem_100_1	$3.27 \times 10^{-2}/3.9 \times 10^{-3}$	$1.67 \times 10^{-2}/1.5 \times 10^{-3}$	$1.58 \times 10^{-2}/2.1 \times 10^{-3}$	$\mathbf{1.38 \times 10^{-2}/2.3 \times 10^{-3}}$ +
Problem_100_2	$8.15 \times 10^{-2}/1.4 \times 10^{-2}$	$2.24 \times 10^{-2}/6.8 \times 10^{-3}$	$2.26 \times 10^{-2}/6.9 \times 10^{-3}$	$\mathbf{2.18 \times 10^{-2}/6.0 \times 10^{-3}}$ +
Problem_100_3	$7.00 \times 10^{-2}/1.1 \times 10^{-2}$	$2.32 \times 10^{-2}/6.5 \times 10^{-3}$	$2.02 \times 10^{-2}/5.9 \times 10^{-3}$	$\mathbf{7.08 \times 10^{-3}/6.4 \times 10^{-3}}$ +
Problem_100_4	$1.93 \times 10^{-2}/3.6 \times 10^{-3}$	$2.06 \times 10^{-2}/4.2 \times 10^{-3}$	$5.45 \times 10^{-2}/6.5 \times 10^{-3}$	$\mathbf{8.56 \times 10^{-3}/4.3 \times 10^{-3}}$ +
Problem_100_5	$8.37 \times 10^{-2}/1.2 \times 10^{-2}$	$2.23 \times 10^{-2}/6.2 \times 10^{-3}$	$2.34 \times 10^{-2}/6.5 \times 10^{-3}$	$\mathbf{8.80 \times 10^{-3}/6.4 \times 10^{-3}}$ +
Hit rate	0/15	1/15	1/15	13/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

5.7. LMOEA against Other MOEAs

To assess the performance of the proposed algorithm for scheduling problems, it is compared with other MOEAs such as NSGA-II, SPEA2, OMPSO, and MOEA/D in this section. First, the compared MOEAs are described as follows.

(1) NSGA-II is one of the most popular MOEA proposed by Deb et al. [44]. The characteristic of NSGA-II is that it uses a fast non-dominated sorting and crowding distance estimation procedure. The fast non-dominated sorting technique is used to assign the parent and offspring population to different levels of non-dominated solution fronts. A crowding distance strategy is employed to maintain diversity of the population.

(2) SPEA2 was presented by Zitler et al. [51]. In SPEA2 algorithm each individual has a fitness value which is the sum of its strength raw fitness and density estimation based on the distance to the *k*th nearest neighbor. A new population is from the non-dominated solutions in both the original population and the external archive.

(3) OMOPSO is a multi-objective particle swarm optimization algorithm proposed by Sierra and Coello Coello [52]. This proposal employs two external archives: one for storing the leaders for carrying out the flight and the other one for storing the final solutions. The crowding factor is used to remove the list of leaders whenever the external archive is full of final solutions. Only the leaders with the best crowding values are maintained. Additionally, the authors presented a scheme in which they divide the population into three different subpopulation.

(4) MOEA/D invented by Zhang and Hui. [53] decomposes a multi-objective problem into a number of scalar sub-problems and optimizes them simultaneously. MOEA/D have received widely attention from evolutionary computation fields due to its fast convergence and good diversity.

To make a fair comparison in this experiment, the function evaluation number of all MOEAs is set to be 25,000 and all MOEAs use the same encoding scheme. Additionally, the initial population is randomly generated for the above MOEAs. The other parameters are shown in Table 6. 30 independent runs are conducted for each algorithm on each test instance.

Table 6. Parameter settings in MOEAs.

Algorithm	Parameters			
LMOEA	population size: 100	crossover rate: 0.9	mutation rate: 0.1	archive size: 100
NSGA-II	population size: 100	crossover rate: 0.9	mutation rate: 0.1	
SPEA2	population size: 100	crossover rate: 0.9	mutation rate: 0.1	archive size: 100
OMOPSO	swarm size: 100		mutation rate: 0.1	leader size: 100
MOEA/D	population size: 500	CR = 1.0, F = 0.5, nr = 2, $\delta = 0.9$	mutation rate: 0.1	neighborhood size T: 20

Tables 7–9 present the statistical results of the GD, Spread, and IGD metrics. From these tables, we can clearly observe that the proposed LMOEA outperforms its counterparts for solving these scheduling instances. Especially on the comprehensive metric IGD and convergence metric GD, the outperformances of the proposal are significant. More specifically, Table 7 reveals that the proposed LMOEA is significantly better than the other four MOEAs for the GD metric on 8 out of 15 instances. SPEA2 can obtain better GD values for three instances, and MOEA/D and NSGA-II for two, respectively. It should be noted that OMOPSO fails to get better results on any one instance. We also observe from Table 8 that LMOEA is superior to NSGA-II and OMOPSO for the Spread metric. In addition, LMOEA is competitive to SPEA2 and MOEA/D with regard to Spread metric. From Table 9, LMOEA achieves significantly better results than the compared MOEAs in terms of the IGD metric. It should be pointed out that none of the MOEAs show a consistent and good behavior for all cases. However, compared with other MOEAs, the LMOEA is a very suitable tool for solving most instances since LMOEA shows a very good performance on this type of scheduling problems.

Table 7. Mean and standard deviation value of GD metric of each algorithm.

Problem	NSGA-II	SPEA2	OMOPSO	MOEA/D	LMOEA
	Mean/Std.	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**
Problem_10_1	$7.0 \times 10^{-3}/8.7 \times 10^{-3}$	$1.14 \times 10^{-2}/2.0 \times 10^{-2}$	$2.46 \times 10^{-2}/1.4 \times 10^{-2}$	$6.78 \times 10^{-2}/7.7 \times 10^{-3}$	$\mathbf{6.66 \times 10^{-3}/5.8 \times 10^{-3}}$ +
Problem_10_2	$1.03 \times 10^{-2}/1.0 \times 10^{-2}$	$8.93 \times 10^{-3}/1.1 \times 10^{-2}$	$4.38 \times 10^{-2}/2.6 \times 10^{-2}$	$3.23 \times 10^{-2}/1.2 \times 10^{-2}$	$\mathbf{5.01 \times 10^{-3}/6.3 \times 10^{-3}}$ +
Problem_10_3	$4.29 \times 10^{-3}/3.3 \times 10^{-3}$	$\mathbf{3.77 \times 10^{-3}}/2.5 \times 10^{-3}$	$1.25 \times 10^{-2}/8.4 \times 10^{-3}$	$4.29 \times 10^{-3}/3.3 \times 10^{-3}$	$4.21 \times 10^{-3}/\mathbf{1.7 \times 10^{-3}}$ −
Problem_10_4	$7.01 \times 10^{-3}/3.8 \times 10^{-3}$	$6.53 \times 10^{-3}/4.3 \times 10^{-3}$	$1.32 \times 10^{-2}/8.5 \times 10^{-3}$	$6.31 \times 10^{-3}/3.7 \times 10^{-3}$	$\mathbf{5.27 \times 10^{-3}/1.9 \times 10^{-3}}$ +
Problem_10_5	$5.41 \times 10^{-3}/\mathbf{1.3 \times 10^{-3}}$	$\mathbf{4.01 \times 10^{-3}}/1.9 \times 10^{-3}$	$8.98 \times 10^{-3}/4.7 \times 10^{-3}$	$5.65 \times 10^{-3}/2.7 \times 10^{-3}$	$5.40 \times 10^{-3}/2.2 \times 10^{-3}$ −
Problem_50_1	$5.15 \times 10^{-2}/8.1 \times 10^{-2}$	$3.22 \times 10^{-2}/6.6 \times 10^{-2}$	$3.03 \times 10^{-1}/3.3 \times 10^{-1}$	$\mathbf{3.14 \times 10^{-2}/6.0 \times 10^{-2}}$	$8.67 \times 10^{-2}/1.1 \times 10^{-1}$ −
Problem_50_2	$3.95 \times 10^{-2}/3.2 \times 10^{-2}$	$8.36 \times 10^{-2}/2.9 \times 10^{-1}$	$2.30 \times 10^{-1}/2.1 \times 10^{-1}$	$4.86 \times 10^{-2}/3.6 \times 10^{-2}$	$\mathbf{3.22 \times 10^{-2}/3.5 \times 10^{-2}}$ +
Problem_50_3	$\mathbf{1.33 \times 10^{-2}/1.9 \times 10^{-2}}$	$3.52 \times 10^{-2}/7.8 \times 10^{-2}$	$1.13 \times 10^{-1}/1.0 \times 10^{-1}$	$2.33 \times 10^{-2}/2.1 \times 10^{-2}$	$2.03 \times 10^{-2}/3.2 \times 10^{-2}$ −
Problem_50_4	$1.00 \times 10^{-1}/1.3 \times 10^{-1}$	$1.01 \times 10^{-1}/2.2 \times 10^{-1}$	$6.12 \times 10^{-1}/5.8 \times 10^{-1}$	$1.12 \times 10^{-1}/1.2 \times 10^{-1}$	$\mathbf{2.06 \times 10^{-2}/3.9 \times 10^{-2}}$ +
Problem_50_5	$\mathbf{3.76 \times 10^{-2}/4.5 \times 10^{-2}}$	$1.94 \times 10^{-1}/4.1 \times 10^{-1}$	$1.50 \times 10^{-1}/1.4 \times 10^{-1}$	$1.76 \times 10^{-1}/3.5 \times 10^{-1}$	$1.44 \times 10^{-1}/1.8 \times 10^{-1}$ −
Problem_100_1	$6.25 \times 10^{-3}/5.6 \times 10^{-3}$	$6.19 \times 10^{-3}/1.7 \times 10^{-3}$	$2.94 \times 10^{-2}/1.5 \times 10^{-2}$	$6.01 \times 10^{-3}/5.2 \times 10^{-3}$	$\mathbf{5.62 \times 10^{-3}/2.6 \times 10^{-3}}$ +
Problem_100_2	$5.21 \times 10^{-3}/4.4 \times 10^{-3}$	$4.40 \times 10^{-3}/1.8 \times 10^{-3}$	$2.08 \times 10^{-2}/7.2 \times 10^{-3}$	$5.03 \times 10^{-3}/4.1 \times 10^{-3}$	$\mathbf{4.16 \times 10^{-3}/1.2 \times 10^{-3}}$ +
Problem_100_3	$9.81 \times 10^{-3}/8.4 \times 10^{-3}$	$\mathbf{2.42 \times 10^{-3}}/1.8 \times 10^{-3}$	$2.96 \times 10^{-2}/1.0 \times 10^{-2}$	$3.18 \times 10^{-3}/\mathbf{1.4 \times 10^{-3}}$	$5.78 \times 10^{-3}/1.8 \times 10^{-3}$ −
Problem_100_4	$5.85 \times 10^{-3}/4.2 \times 10^{-3}$	$4.53 \times 10^{-3}/1.8 \times 10^{-3}$	$1.91 \times 10^{-2}/6.8 \times 10^{-3}$	$5.85 \times 10^{-3}/4.2 \times 10^{-3}$	$\mathbf{4.06 \times 10^{-3}/9.6 \times 10^{-4}}$ +
Problem_100_5	$7.67 \times 10^{-3}/8.5 \times 10^{-3}$	$4.97 \times 10^{-3}/2.5 \times 10^{-3}$	$9.06 \times 10^{-2}/4.5 \times 10^{-2}$	$\mathbf{3.69 \times 10^{-3}}/4.5 \times 10^{-2}$	$2.28 \times 10^{-1}/\mathbf{1.2 \times 10^{-2}}$ −
Hit rate	2/15	3/15	0/15	2/15	8/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

Table 8. Mean and standard deviation of Spread metric of each algorithm.

Problem	NSGA-II	SPEA2	OMOPSO	MOEA/D	LMOEA
	Mean/Std.	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**	**Mean/Std.**
Problem_10_1	$8.38 \times 10^{-1}/1.2 \times 10^{-1}$	$8.12 \times 10^{-1}/1.6 \times 10^{-1}$	$9.51 \times 10^{-1}/1.3 \times 10^{-1}$	$7.46 \times 10^{-1}/1.2 \times 10^{-1}$	$8.96 \times 10^{-1}/1.5 \times 10^{-1}$ −
Problem_10_2	$1.03/2.2 \times 10^{-1}$	$8.61 \times 10^{-1}/2.3 \times 10^{-1}$	$1.12/1.5 \times 10^{-1}$	$8.45 \times 10^{-1}/1.8 \times 10^{-1}$	$1.03/2.0 \times 10^{-1}$ −
Problem_10_3	$9.87 \times 10^{-1}/7.1 \times 10^{-2}$	$9.43 \times 10^{-1}/9.7 \times 10^{-1}$	$9.79 \times 10^{-1}/1.2 \times 10^{-1}$	$9.23 \times 10^{-1}/8.6 \times 10^{-2}$	$9.80 \times 10^{-1}/6.5 \times 10^{-2}$ −
Problem_10_4	$9.73 \times 10^{-1}/8.2 \times 10^{-2}$	$8.50 \times 10^{-1}/7.9 \times 10^{-1}$	$9.80 \times 10^{-1}/1.4 \times 10^{-1}$	$9.17 \times 10^{-1}/8.9 \times 10^{-2}$	$8.66 \times 10^{-1}/8.5 \times 10^{-2}$ −
Problem_10_5	$8.87 \times 10^{-1}/7.2 \times 10^{-2}$	$9.12 \times 10^{-1}/9.1 \times 10^{-1}$	$9.22 \times 10^{-1}/9.1 \times 10^{-2}$	$8.84 \times 10^{-1}/9.2 \times 10^{-2}$	$\mathbf{8.76 \times 10^{-1}/9.3 \times 10^{-2}}$ +
Problem_50_1	$9.71 \times 10^{-1}/2.1 \times 10^{-1}$	$1.05/3.2 \times 10^{-1}$	$1.08/2.0 \times 10^{-1}$	$1.03/2.2 \times 10^{-1}$	$1.02/3.5 \times 10^{-1}$ −
Problem_50_2	$9.28 \times 10^{-1}/1.4 \times 10^{-1}$	$7.72 \times 10^{-1}/2.2 \times 10^{-1}$	$1.10/2.5 \times 10^{-1}$	$9.68 \times 10^{-1}/2.7 \times 10^{-1}$	$9.66 \times 10^{-1}/2.9 \times 10^{-2}$ −
Problem_50_3	$8.45 \times 10^{-1}/1.3 \times 10^{-1}$	$9.04 \times 10^{-1}/3.2 \times 10^{-1}$	$1.02/2.2 \times 10^{-1}$	$9.27 \times 10^{-1}/3.1 \times 10^{-1}$	$\mathbf{8.26 \times 10^{-1}/2.6 \times 10^{-1}}$ +
Problem_50_4	$9.88 \times 10^{-1}/2.4 \times 10^{-1}$	$8.94 \times 10^{-1}/2.7 \times 10^{-1}$	$1.18/2.1 \times 10^{-1}$	$9.94 \times 10^{-1}/3.4 \times 10^{-1}$	$\mathbf{8.50 \times 10^{-1}/2.8 \times 10^{-1}}$ +
Problem_50_5	$9.38 \times 10^{-1}/1.8 \times 10^{-1}$	$9.48 \times 10^{-1}/3.1 \times 10^{-1}$	$1.00/1.7 \times 10^{-1}$	$9.42 \times 10^{-1}/3.1 \times 10^{-1}$	$\mathbf{9.19 \times 10^{-1}/3.2 \times 10^{-1}}$ +
Problem_100_1	$9.06 \times 10^{-1}/1.6\text{e}{-01}$	$7.57 \times 10^{-1}/1.8 \times 10^{-1}$	$9.66 \times 10^{-1}/1.8 \times 10^{-1}$	$8.64 \times 10^{-1}/1.8 \times 10^{-1}$	$8.10 \times 10^{-1}/1.8 \times 10^{-1}$ −
Problem_100_2	$8.66 \times 10^{-1}/9.0 \times 10^{-2}$	$7.39 \times 10^{-1}/9.7 \times 10^{-1}$	$8.62 \times 10^{-1}/9.3 \times 10^{-2}$	$7.14 \times 10^{-1}/8.8 \times 10^{-2}$	$8.06 \times 10^{-1}/6.1 \times 10^{-2}$ −
Problem_100_3	$8.55 \times 10^{-1}/6.1 \times 10^{-2}$	$7.37 \times 10^{-1}/7.7 \times 10^{-2}$	$8.29 \times 10^{-1}/9.1 \times 10^{-2}$	$7.88 \times 10^{-1}/7.9 \times 10^{-1}$	$7.65 \times 10^{-1}/7.3 \times 10^{-2}$ −
Problem_100_4	$1.06/2.8 \times 10^{-1}$	$8.38 \times 10^{-1}/2.9 \times 10^{-1}$	$8.47 \times 10^{-1}/8.2 \times 10^{-2}$	$8.64 \times 10^{-1}/3.1 \times 10^{-1}$	$8.44 \times 10^{-1}/2.9 \times 10^{-1}$ −
Problem_100_5	$1.04/2.8 \times 10^{-1}$	$9.26 \times 10^{-1}/2.9 \times 10^{-1}$	$8.53 \times 10^{-1}/9.9 \times 10^{-2}$	$9.26 \times 10^{-1}/2.9 \times 10^{-1}$	$1.10/1.5 \times 10^{-1}$ −
Hit rate	1/15	5/15	1/15	4/15	4/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

Table 9. Mean and standard deviation value of IGD metric of each algorithm.

Problem	NSGA-II	SPEA2	OMOPSO	MOEA/D	LMOEA
	Mean/Std.	Mean/Std.	Mean/Std.	Mean/Std.	Mean/Std.
Problem_10_1	$4.95 \times 10^{-3}/2.0 \times 10^{-3}$	$5.07 \times 10^{-3}/1.8 \times 10^{-3}$	$5.80 \times 10^{-3}/1.7 \times 10^{-3}$	$5.11 \times 10^{-3}/1.8 \times 10^{-3}$	$\mathbf{4.34 \times 10^{-3}/1.8 \times 10^{-3}}$ +
Problem_10_2	$6.19 \times 10^{-3}/2.3 \times 10^{-3}$	$7.01 \times 10^{-3}/2.6 \times 10^{-3}$	$6.57 \times 10^{-3}/2.6 \times 10^{-3}$	$6.82 \times 10^{-3}/2.5 \times 10^{-3}$	$\mathbf{5.41 \times 10^{-3}/2.2 \times 10^{-3}}$ +
Problem_10_3	$4.64 \times 10^{-3}/2.1 \times 10^{-3}$	$5.46 \times 10^{-3}/1.7 \times 10^{-3}$	$5.89 \times 10^{-3}/2.1 \times 10^{-3}$	$5.23 \times 10^{-3}/1.6 \times 10^{-3}$	$\mathbf{3.66 \times 10^{-3}/1.2 \times 10^{-3}}$ +
Problem_10_4	$4.00 \times 10^{-3}/1.2 \times 10^{-3}$	$5.17 \times 10^{-3}/1.8 \times 10^{-3}$	$4.88 \times 10^{-3}/1.3 \times 10^{-3}$	$4.51 \times 10^{-3}/1.9 \times 10^{-3}$	$\mathbf{3.57 \times 10^{-3}/1.3 \times 10^{-3}}$ +
Problem_10_5	$4.57 \times 10^{-3}/1.2 \times 10^{-3}$	$5.55 \times 10^{-3}/1.5 \times 10^{-3}$	$5.34 \times 10^{-3}/1.4 \times 10^{-3}$	$5.12 \times 10^{-3}/1.2 \times 10^{-3}$	$\mathbf{4.53 \times 10^{-3}/1.9 \times 10^{-3}}$ +
Problem_50_1	$6.02 \times 10^{-3}/7.1 \times 10^{-4}$	$7.35 \times 10^{-3}/7.5 \times 10^{-4}$	$9.14 \times 10^{-3}/1.1 \times 10^{-3}$	$6.35 \times 10^{-3}/7.2 \times 10^{-4}$	$\mathbf{5.78 \times 10^{-3}/9.8 \times 10^{-4}}$ +
Problem_50_2	$5.07 \times 10^{-3}/1.1 \times 10^{-3}$	$6.07 \times 10^{-3}/7.0 \times 10^{-4}$	$7.67 \times 10^{-3}/1.3 \times 10^{-3}$	$5.13 \times 10^{-3}/1.5 \times 10^{-3}$	$\mathbf{4.69 \times 10^{-3}/6.4 \times 10^{-4}}$ +
Problem_50_3	$4.87 \times 10^{-3}/7.8 \times 10^{-4}$	$6.18 \times 10^{-3}/6.1 \times 10^{-4}$	$8.42 \times 10^{-3}/1.8 \times 10^{-3}$	$6.12 \times 10^{-3}/5.7 \times 10^{-4}$	$\mathbf{4.54 \times 10^{-3}/5.1 \times 10^{-4}}$ +
Problem_50_4	$5.83 \times 10^{-3}/9.1 \times 10^{-4}$	$6.65 \times 10^{-3}/7.4 \times 10^{-4}$	$1.06 \times 10^{-3}/1.7 \times 10^{-3}$	$6.71 \times 10^{-3}/7.2 \times 10^{-4}$	$\mathbf{5.40 \times 10^{-3}/8.3 \times 10^{-4}}$ +
Problem_50_5	$5.28 \times 10^{-3}/8.4 \times 10^{-4}$	$6.64 \times 10^{-3}/5.6 \times 10^{-4}$	$7.60 \times 10^{-3}/1.0 \times 10^{-3}$	$6.51 \times 10^{-3}/5.1 \times 10^{-4}$	$\mathbf{5.02 \times 10^{-3}/1.1 \times 10^{-4}}$ +
Problem_100_1	$7.40 \times 10^{-3}/1.4 \times 10^{-3}$	$9.76 \times 10^{-3}/9.4 \times 10^{-4}$	$8.44 \times 10^{-3}/1.1 \times 10^{-3}$	$8.03 \times 10^{-3}/7.3 \times 10^{-4}$	$\mathbf{6.70 \times 10^{-3}/1.0 \times 10^{-3}}$ +
Problem_100_2	$1.10 \times 10^{-2}/2.7 \times 10^{-3}$	$1.28 \times 10^{-2}/2.3 \times 10^{-3}$	$\mathbf{9.27 \times 10^{-3}/1.0 \times 10^{-3}}$	$3.01 \times 10^{-3}/2.7 \times 10^{-3}$	$9.56 \times 10^{-3}/2.3 \times 10^{-3}$ −
Problem_100_3	$8.07 \times 10^{-3}/1.3 \times 10^{-3}$	$9.75 \times 10^{-3}/8.9 \times 10^{-4}$	$9.27 \times 10^{-3}/1.2 \times 10^{-3}$	$7.87 \times 10^{-4}/5.9 \times 10^{-4}$	$\mathbf{7.12 \times 10^{-3}/1.1 \times 10^{-3}}$ +
Problem_100_4	$8.36 \times 10^{-3}/1.3 \times 10^{-3}$	$1.01 \times 10^{-2}/9.9 \times 10^{-4}$	$8.70 \times 10^{-3}/9.6 \times 10^{-4}$	$2.12 \times 10^{-2}/1.9 \times 10^{-3}$	$\mathbf{7.66 \times 10^{-3}/1.1 \times 10^{-3}}$ +
Problem_100_5	$6.85 \times 10^{-3}/8.7 \times 10^{-4}$	$8.36 \times 10^{-3}/1.2 \times 10^{-3}$	$8.51 \times 10^{-3}/1.1 \times 10^{-3}$	$7.34 \times 10^{-3}/1.7 \times 10^{-3}$	$\mathbf{6.49 \times 10^{-3}/1.1 \times 10^{-3}}$ +
Hit rate	0/15	0/15	1/15	0/15	14/15

+ and − denote that the performance of the LMOEA is significantly better than and worse than other algorithms, respectively. And the best values are highlighted in bold.

The average and standard deviation CPU-time of each algorithm has been reported in Table 10, and the unit of time is second. In order to visualize the performance of five algorithms, the CPU-time costs on all test instances are selected for graphical representation for the Figure 10, vertical axis taken logarithm. The figure illustrates and confirms some conclusions are derived from the numerical analysis with the performance criteria. It can be observed that SPEA2 is the fastest and OMOPSO is the slowest in solving all the problems. When there are more machines at each stage, CPU-time of each algorithm will be added accordingly, nevertheless LMOEA CPU-time also grows slower than other algorithms. The reason lies in the procedure of local search strategy construction. In this research, we present three different local search strategies (S1, S2 and S3) based on self-adaptive selection mechanism to obtain better approximate solutions, more details have been shown in Section 4.2.4. Consequently, LMOEA can benefit from a local search strategy which aims at enhancing exploitation ability of the algorithm. In contrast, OMOPSO does not include any heuristic information for reducing running time.

Table 10. Mean and standard deviation CPU-time of each algorithm (unit is second).

Problem	NSGA-II	SPEA2	OMOPSO	MOEA/D	LMOEA
	Mean/Std.	Mean/Std.	Mean/Std.	Mean/Std.	Mean/Std.
Problem_10_1	0.661/0.375	**0.642**/0.321	17.076/1.719	1.292/**0.206**	7.348/0.528
Problem_10_2	0.717/0.310	**0.615**/0.276	15.422/1.288	1.242/**0.219**	7.313/0.658
Problem_10_3	0.697/0.321	**0.658**/0.398	14.801/1.302	1.238/**0.212**	7.154/0.672
Problem_10_4	0.702/0.329	**0.635**/0.293	15.845/1.494	1.250/**0.247**	7.631/0.911
Problem_10_5	0.691/0.322	**0.639**/0.277	15.617/1.581	1.255/**0.206**	7.211/0.638
Problem_50_1	2.161/0.511	**1.295/0.339**	119.246/5.678	2.988/0.374	9.060/0.890
Problem_50_2	2.219/0.480	**1.290/0.340**	129.345/8.984	3.233/0.448	9.182/0.727
Problem_50_3	2.134/0.447	**1.318/0.276**	117.909/6.094	3.500/0.447	9.022/0.703
Problem_50_4	2.189/0.386	**1.283**/0.375	119.640/6.532	3.448/**0.372**	8.632/0.629
Problem_50_5	2.180/0.363	**1.285**/0.340	115.436/7.438	3.433/**0.335**	8.596/0.882
Problem_100_1	5.606/0.481	**2.780/0.370**	389.005/15.972	6.954/0.794	11.616/0.831
Problem_100_2	5.493/**0.348**	**2.863**/0.376	387.852/15.519	6.192/0.488	11.609/0.921
Problem_100_3	5.545/0.467	**2.749/0.366**	387.411/16.465	6.196/0.434	11.683/0.921
Problem_100_4	5.728/0.539	**2.793/0.446**	418.133/24.103	6.231/0.461	11.538/0.923
Problem_100_5	5.542/0.408	**2.746/0.390**	395.403/16.745	6.472/0.530	11.757/1.029

The best values are highlighted in bold.

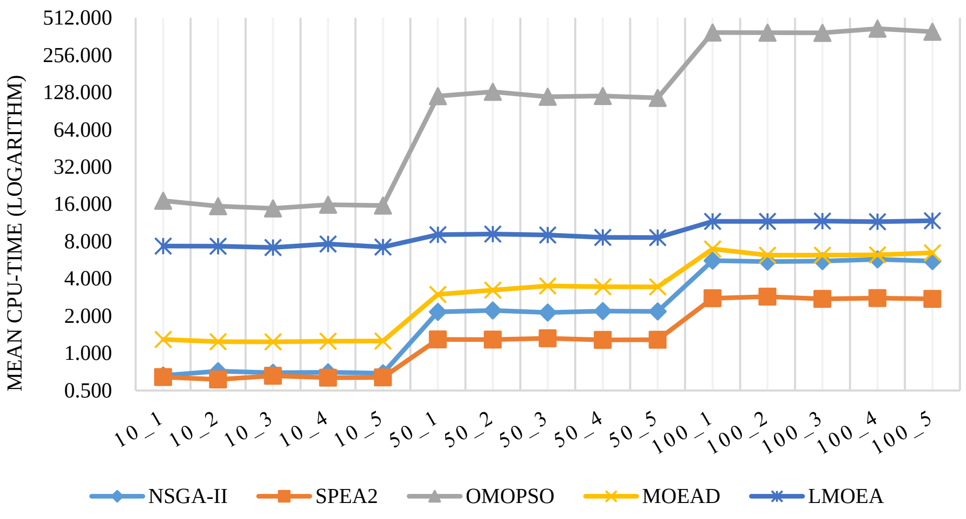
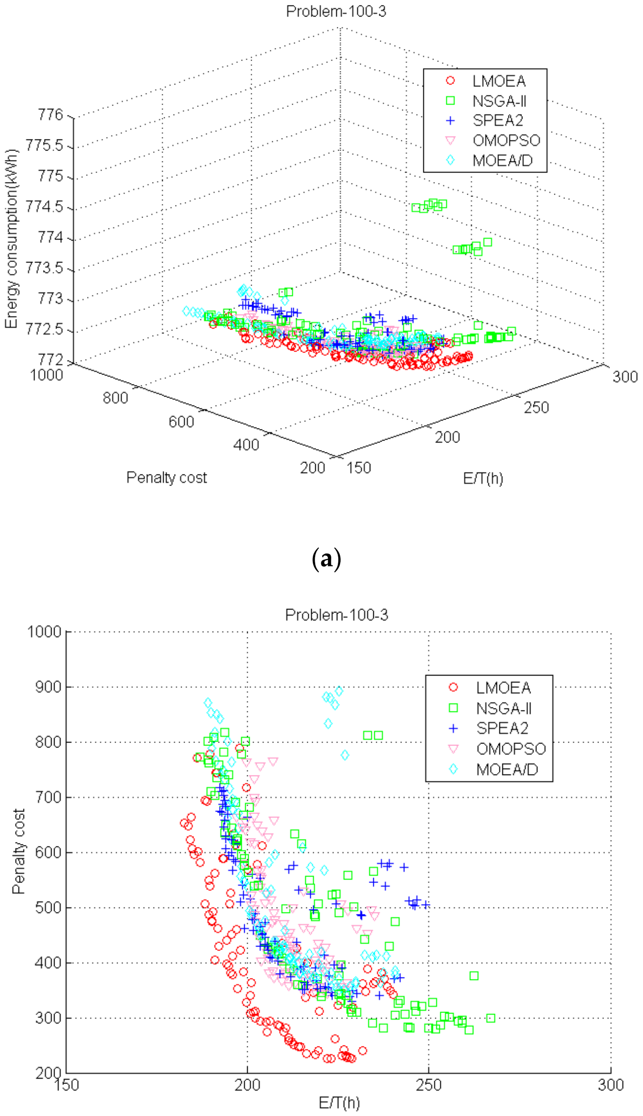


Figure 10. The CPU-time costs on all test instances.

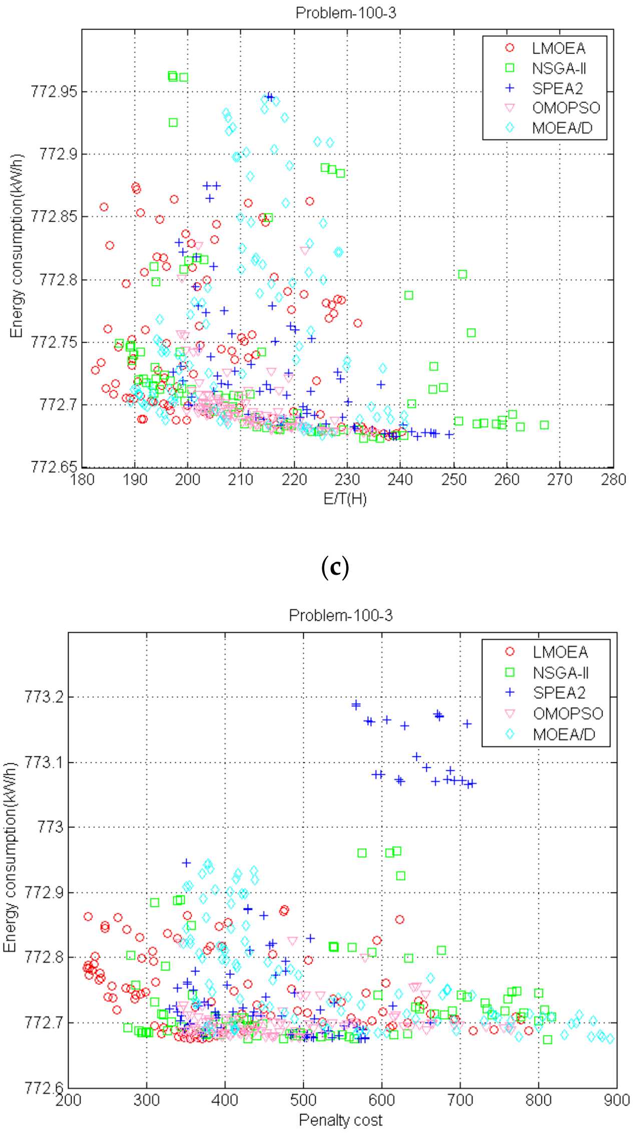
To give a graphical overview of the performance of these algorithms, Figure 11 shows the *PF* approximations obtained by different MOEAs in the run with the best IGD value for the problem_100_3 from different perspectives. In all the figures, the total E/T represents f_1 objective; penalty cost represents f_2 and the total energy consumption denotes f_3. Figure 11a presents *PF* in 3D space and it is hard to judge which MOEA is the best one among them. However, it is evident from Figure 11b that the proposed LMOEA reaches better approximations toward *PF** than the other MOEAs in 2D space, which denotes the LMOEA has good convergence performance. It also indicates the two objectives of the total E/T and cost are conflicted with each other. As shown in Figure 11c,d, even though the SPEA2, OMOPSO and LMOEA can approach closer toward *PF** than NSGA-II and MOEA/D, the proposed LMOEA can cover a wider range than SPEA2 and OMOPSO. It implies that *PF* obtained by LMOEA has good diversity. SPEA2 also covers a large area, but it tends to converge to local areas. The outperformance of LMOEA can be attributed to multiple local search strategies based upon adaptive mechanism, by which LMOEA can search preferable solutions in its neighborhood.



(**a**)

(**b**)

Figure 11. *Cont.*



(**c**)

(**d**)

Figure 11. Final solutions obtained by different MOEAs with the best IGD values from different perspectives. (**a**) Pareto front of ET, Penalty cost and Energy consumption, (**b**) Pareto front of ET and Penalty cost, (**c**) Pareto front of ET and Energy consumption, (**d**) Pareto front of Penalty cost and Energy consumption.

According to the results in the above experimental studies, the LMOEA is a promising approach to cope with this kind of scheduling. Additionally, several observations are also drawn.

(1) Different crossover operators have different impacts on the performance of the proposed LMOEA. The SBX operator is the best choice among the compared crossovers in LMOEA for solving the scheduling instances.

(2) Compared with the adoption of only one local search strategy, integration of multiple local search techniques with self-adaptive mechanism can significantly improve the behavior of the LMOEA.

(3) The objectives of total E/T, penalty cost and energy consumption are in conflict with each other according to Figure 11.

(4) The proposed LMOEA can perform better than its rivals such as NSGA-II, OMOPSO, SPEA2, and MOEA/D for most scheduling instances.

6. Case Study

Consider the following single machine scheduling problem with controllable processing and setup times from a realistic welding workshop in Henan Weihua Heavy Machinery Co., Ltd, Changyuan, China. This welding shop is required to process 12 types of jobs that should be passed through only one machine. The relative data including Machine power, available machines, job number, processing times, setup times is provided in Tables 11–14.

Table 11. Machine power of manufacturing process.

Machine Processing Power (kW)	Power with Setup Time (kW)	Power When Turning On/Off (kW)
26.5	3.8	10.5

Table 12. Available machine in the welding workshop.

Machine Type	Available Machine Number	Manufacture Description
TIG welding machine	1	Slicing of cover slab and gusset

Table 13. The processing times of job on tungsten inert gas (TIG) welding machine.

Job Type	Size (cm)	Job No.	Normal Processing Time (min) (*p*)	Maximum Amount of Extension of Processing Time (m_j^e)	Unit Cost of Expansion Time (b_j)	Maximum Amount of Compression of Processing Time (m_j^c)	Unit Cost of Compression Time (a_j)	Due Time (d_j)
Y6ws01A1	105	1	16	1	2	2	1.5	80
Y6ws01A1	115	2	18	2	2	3	1.5	120
Y6ws01A1	125	3	20	3	2	4	1.5	170
Y6ws01A1	135	4	22	4	2	5	1.5	190
L1ws01A2	105	5	16	1	2	2	1.5	170
L1ws01A2	115	6	18	2	2	3	1.5	130
L1ws01A2	125	7	20	3	2	4	1.5	180
L1ws01A2	135	8	22	4	2	5	1.5	220
G5ws01A3	105	9	16	1	2	2	1.5	230
G5ws01A3	115	10	18	2	2	3	1.5	200
G5ws01A3	125	11	20	3	2	4	1.5	120
G5ws01A3	135	12	22	4	2	5	1.5	150

Table 14. The setup times between two consecutive jobs.

Setup Time (min)	Job1	Job2	Job3	Job4	Job5	Job6	Job7	Job8	Job9	Job10	Job11	Job12
Job1	-	6	5	7	5	8	6	9	8	10	9	9
Job2	4	-	4	5	6	7	9	5	10	8	5	7
Job3	4	5	-	5	6	3	6	10	8	7	5	4
Job4	5	5	4	-	5	4	8	5	6	10	12	5
Job5	8	11	6	9	-	8	7	9	7	4	8	14
Job6	5	4	8	9	8	-	11	18	5	6	5	4
Job7	6	4	11	8	7	5	-	9	6	6	7	8
Job8	9	6	5	7	3	10	15	-	4	5	5	9
Job9	5	6	7	13	11	6	5	8	-	12	8	10
Job10	10	7	7	14	9	8	5	8	5	-	4	10
Job11	4	6	9	5	6	7	8	11	3	7	-	15
Job12	5	6	8	4	7	9	7	5	15	8	14	-

The Pareto front obtained by the proposed LMOEA is presented in Figure 12. It is obviously observed that the proposed LMOEA is capable of providing high quality and good diversity solutions.

Point A, B and C represent three extreme objective values, respectively. These points can be expressed as follows: A (813.8, 16.5, 6625.6), B (850.8, 0, 6639), and C (1213.60, 0, 6411). The above extreme objectives correspond to the following three non-dominated solutions.

Solution at point A: $\pi = \{J_5, J_{11}, J_1, J_2, J_3, J_6, J_{12}, J_7, J_{10}, J_9, J_8, J_4\}$, m= (0, 0, 0, 0, −3, 0, −3, 0, 0, 0, 0, 1,)
Solution at point B: $\pi = \{J_5, J_{11}, J_1, J_2, J_6, J_{12}, J_3, J_4, J_{10}, J_7, J_9, J_8\}$, m= (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,)
Solution at point C: $\pi = \{J_4, J_7, J_2, J_1, J_3, J_6, J_{12}, J_8, J_5, J_{10}, J_{11}, J_9\}$ m= (0, 0, 0, −1, 0, 0, 0, 0, 0, 0, 0, 0,)

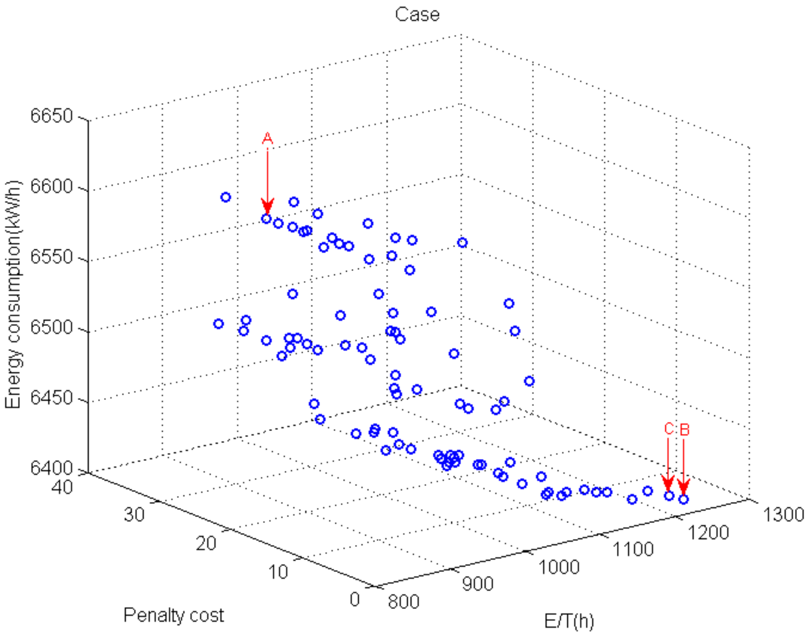


Figure 12. Pareto front obtained by the proposed LMOEA. Point A, B and C represent three extreme objective values.

The computed results indicate that the proposed Energy-efficient scheduling problems model and corresponding local MOEA can indeed optimize the decision target, effectively solving the process-based problem. However, it is not possible to obtain a dispatch plan that simultaneously optimizes the penalty cost, total process energy consumption and E/T because optimizing one indicator inevitably worsens the others.

7. Conclusions

This paper developed a multi-objective mathematical model considering both productivity (i.e., E/T and cost) and energy consumption on a single machine environment. In this new model, controllable processing times are employed to approach toward JIT production, and the energy consumption can be reduced by operational and the turn off/on method. The proposed algorithm LMOEA is utilized to solve this scheduling problem. This algorithm LMOEA is different from other MOEAs where three different local search strategy based on adaptive scheme is to improve search performance of the algorithm. There are still some limitations. Firstly, the heuristic information based on problem property has not been introduced into the LMOEA. Secondly, the proposed algorithm contains some parameters, which affect the performance of the algorithm.

Experimental studies are conducted to measure the performance of the proposed LMOEA algorithm for scheduling instances. First, a SBX operator is chosen as an appropriate crossover operator in the LMOEA by comparing different crossovers on all instances, and then the impact of the crossover and mutation probability is also analyzed. Finally, the computational experiments demonstrate that the proposed LMOEA outperforms its rivals such as NSGA-II, SPEA2, OMOPSO, and MOEA/D for most scheduling problems. In addition, the main contributions of this work are as follows:

(1) A multi-objective scheduling model, which considers three objectives, is constructed. A hybrid MOEA with multiple local search strategy is developed to effectively solve this kind of the scheduling.
(2) A new solution representation is provided for this kind of the scheduling. It also offers a new idea in solving scheduling problems.
(3) In this work, the mutation operator composed of two types of mutation techniques is proposed. The first mutation technique can adjust amount of compression of job processing times. The second mutation technique can exchange order of jobs. It implies the mutation operator can helps the algorithm to escape from local optimum.
(4) According to the above parametric experiment, both parameters Pc and Pm have critical effect on the performance of the LMOEA for most instances. The range (0.7, 1) for Pc and 0.1 for Pm are recommended for decision makers when solving these scheduling instances.
(5) Multiple local strategies based on adaptive selection scheme can improve the performance of the LMOEA on instances.

With respect to future work, the proposed algorithm should be evaluated on different scheduling environments such as job and open shop scheduling environment to validate its more general applicability. Another issue for future research is to improve the search efficiency of the algorithm by incorporating current new techniques such as deep learning. In addition, operation speed associated with processing times will be considered in future research.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (NSFC) under Grant no. 51375004, National Key Technology Support Program under Grant no. 2015BAF01B04, Youth Science & Technology Chenguang Program of Wuhan under Grant no. 2015070404010187 and the Fundamental Research Funds for the Central Universities, HUST under Grant no. 2015TS061.

Author Contributions: Lvjiang Yin analyzed the data and wrote the manuscript; Xinyu Li and Chao Lu contributed to the design of the concept and algorithms; Liang Gao designed the whole experiment and was in charge of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhou, M.; Pan, Y.; Chen, Z.; Yang, W. Optimizing green production strategies: An integrated approach. *Comput. Ind. Eng.* **2013**, *65*, 517–528. [CrossRef]
2. Rager, M.; Gahm, C.; Denz, F. Energy-oriented scheduling based on evolutionary algorithms. *Comput. Oper. Res.* **2015**, *54*, 218–231. [CrossRef]
3. Kayvanfar, V.; Mahdavi, I.; Komaki, G.M. Single machine scheduling with controllable processing times to minimize total tardiness and earliness. *Comput. Ind. Eng.* **2013**, *65*, 166–175. [CrossRef]
4. Dietmair, A.; Verl, A. Energy consumption forecasting and optimisation for tool machines. *Mod. Mach. Sci. J.* **2009**, *14*, 62–67. [CrossRef]
5. Dahmus, J.B.; Gutowski, T.G. An Environmental Analysis of Machining. In Proceedings of IMECE 2004, 2004 ASME International Mechanical Engineering Congress and Exposition, Anaheim, CA, USA, 13–19 November 2004.
6. Drake, R.; Yildirim, M.B.; Twomey, J.; Whitman, L.; Ahmad, J.; Lodhia, P. Data collection framework on energy consumption in manufacturing. In Proceedings of the IIE Annual Conference and Expo, Orlando, FL, USA, 20–24 May 2006.
7. Kayvanfar, V.; Mahdavi, I.; Komaki, G.M. A drastic hybrid heuristic algorithm to approach to JIT policy considering controllable processing times. *Int. J. Adv. Manuf. Technol.* **2013**, *69*, 257–267. [CrossRef]
8. Gahm, C.; Denz, F.; Dirr, M.; Tuma, A. Energy-efficient scheduling in manufacturing companies: A review and research framework. *Eur. J. Oper. Res.* **2016**, *248*, 744–757. [CrossRef]

9. Gao, K.Z.; Suganthan, P.N.; Pan, Q.K.; Chua, T.J.; Cai, T.X.; Chong, C.S. Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. *Inform. Sci.* **2014**, *289*, 76–90. [CrossRef]
10. García-Nájera, A.; Bullinaria, J.A.; Gutiérrez-Andrade, M.A. An evolutionary approach for multi-objective vehicle routing problems with backhauls. *Comput. Ind. Eng.* **2015**, *81*, 90–108. [CrossRef]
11. Gen, M.; Lin, L. Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey. *J. Intell. Manuf.* **2014**, *25*, 849–866. [CrossRef]
12. Hao, X.; Gen, M.; Lin, L.; Suer, G. Effective multiobjective eda for bi-criteria stochastic job-shop scheduling problem. *J. Intell. Manuf.* **2015**. [CrossRef]
13. Zhang, G.; Shao, X.; Li, P.; Gao, L. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Comput. Ind. Eng.* **2009**, *56*, 1309–1318. [CrossRef]
14. Zheng, J.-N.; Chien, C.-F.; Gen, M. Multi-objective multi-population biased random-key genetic algorithm for the 3-D container loading problem. *Comput. Ind. Eng.* **2014**. [CrossRef]
15. Mouzon, G.; Yildirim, M.B.; Twomey, J. Operational methods for minimization of energy consumption of manufacturing equipment. *Int. J. Prod. Res.* **2007**, *45*, 4247–4271. [CrossRef]
16. Mouzon, G.; Yildirim, M.B. A framework to minimize total energy consumption and total tardiness on a single machine. *Int. J. Sustain. Eng.* **2008**, *1*, 105–116. [CrossRef]
17. Fang, K.; Uhan, N.; Zhao, F.; Sutherland, J.W. A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *J. Manuf. Syst.* **2011**, *30*, 234–240. [CrossRef]
18. Fang, K.; Uhan, N.A.; Zhao, F.; Sutherland, J.W. Flow shop scheduling with peak power consumption constraints. *Ann. Oper. Res.* **2013**, *206*, 115–145. [CrossRef]
19. Bruzzone, A.A.G.; Anghinolfi, D.; Paolucci, M.; Tonelli, F. Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. *Cirp Ann.-Manuf. Technol.* **2012**, *61*, 459–462. [CrossRef]
20. Dai, M.; Tang, D.; Giret, A.; Salido, M.A.; Li, W.D. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robot. Comput.-Integr. Manuf.* **2013**, *29*, 418–429. [CrossRef]
21. Luo, H.; Du, B.; Huang, G.Q.; Chen, H.; Li, X. Hybrid flow shop scheduling considering machine electricity consumption cost. *Int. J. Prod. Econ.* **2013**, *146*, 423–439. [CrossRef]
22. Keller, K.; Schonborn, C.; Reinhart, G. Energy-orientated machine scheduling for hybrid flow shop. *Procedia CIPR* **2015**, *29*, 156–161. [CrossRef]
23. Liu, Y.; Dong, H.B.; Lohse, N.; Petrovic, S.; Gindy, N. An investigation into minimising total energy consumption and total weighted tardiness in job shops. *J. Clean. Prod.* **2014**, *65*, 87–96. [CrossRef]
24. Kang, P.S.; Duffy, A.; Shires, N.; Smith, T.; Novels, M. Lean Cables—A Step towards Competitive, Sustainable and Profitable Processes. In Proceedings of the 63rd International Wire and Cable Symposium, Providence, RI, USA, 9–12 November 2014.
25. Kang, P.S.; Erhart, A.; Todeschini, V.; Duffy, A. Comparative Study of Two Scheduling Approaches to Resolve Scheduling Problem for a Wire and Cable Manufacturing Process. Available online: https://www.researchgate.net/publication/291312939_Comparative_Study_of_Two_Scheduling_Approaches_to_Resolve_Scheduling_Problem_for_a_Wire_and_Cable_Manufacturing_Process (accessed on 15 November 2015).
26. Yildirim, M.B.; Mouzon, G. Single-machine sustainable production planning to minimize total energy consumption and total completion time using a multiple objective genetic algorithm. *IEEE Trans. Eng. Manag.* **2012**, *59*, 585–597. [CrossRef]
27. Shrouf, F.; Ordieres-Mere, J.; Garcia-Sanchez, A.; Ortega-Mier, M. Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *J. Clean. Prod.* **2014**, *67*, 197–207. [CrossRef]
28. Lu, C.; Gao, L.; Li, X.; Chen, P. Energy-efficient multi-pass turning operation using multi-objective backtracking search algorithm. *J. Clean. Prod.* **2016**, *137*, 1516–1531. [CrossRef]
29. He, Y.; Liu, F.; Cao, H.-J.; Li, C.-J. A bi-objective model for jop-shop scheduling problem to minimize both energy consumption and makespan. *J. Cent. South Univ. Technol.* **2005**, *12*, 167–171. [CrossRef]
30. Liu, X.; Zou, F.; Zhang, X. Mathematical Model and Genetic Optimization for Hybrid Flow Shop Scheduling Problem Based on Energy Consumption. In Proceedings of 2008 Control and Decision Conference, Yantai, China, 2–4 July 2008.
31. Zhang, L.; Li, X.; Gao, L.; Zhang, G. Dynamic rescheduling in fms that is simultaneously considering energy consumption and schedule efficiency. *Int. J. Adv. Manuf. Technol.* **2013**, *87*, 1387–1399. [CrossRef]

32. Liu, C.H. Approximate trade-off between minimisation of total weighted tardiness and minimisation of carbon dioxide (CO_2) emissions in bi-criteria batch scheduling problem. *Int. J. Comput. Integr. Manuf.* **2014**, *27*, 759–771. [CrossRef]
33. Mason, S.J.; Jin, S.; Jampani, J. A moving block heuristic for minimizing earliness and tardiness on a single machine with unrestrictive common due dates. *J. Manuf. Syst.* **2005**, *24*, 328–338. [CrossRef]
34. Lenstra, J.K.; Rinnooy Kan, A.H.G.; Brucker, P. Complexity of machine scheduling problems. In *Annals of Discrete Mathematics*; Elsevier: Toronto, ON, Canada, 1977; Volume 1, pp. 343–362.
35. Yano, C.A.; Kim, Y.-D. Algorithms for a class of single-machine weighted tardiness and earliness problems. *Eur. J. Oper. Res.* **1991**, *52*, 167–178. [CrossRef]
36. Valente, J.M.S. Beam search heuristics for quadratic earliness and tardiness scheduling. *J. Oper. Res. Soc.* **2010**, *61*, 620–631. [CrossRef]
37. Valente, J.M.S.; Alves, R. Beam search algorithms for the early/tardy scheduling problem with release dates. *J. Manuf. Syst.* **2005**, *24*, 35–46. [CrossRef]
38. Khorshidian, H.; Javadian, N.; Zandieh, M.; Rezaeian, J.; Rahmani, K. A genetic algorithm for jit single machine scheduling with preemption and machine idle time. *Expert Syst. Appl.* **2011**, *38*, 7911–7918. [CrossRef]
39. Weng, W.; Wei, X.; Fujimura, S. Dynamic routing strategies for jit production in hybrid flow shops. *Comput. Oper. Res.* **2012**, *39*, 3316–3324. [CrossRef]
40. Benmansour, R.; Allaoui, H.; Artiba, A.; Hanafi, S. Minimizing the weighted sum of maximum earliness and maximum tardiness costs on a single machine with periodic preventive maintenance. *Comput. Oper. Res.* **2014**, *47*, 106–113. [CrossRef]
41. Sourd, F.; Kedad-Sidhoum, S. A faster branch-and-bound algorithm for the earliness-tardiness scheduling problem. *J. Sched.* **2008**, *11*, 49–58. [CrossRef]
42. Baker, K.R. Minimizing earliness and tardiness costs in stochastic scheduling. *Eur. J. Oper. Res.* **2014**, *236*, 445–452. [CrossRef]
43. Shabtay, D.; Steiner, G. The single-machine earliness-tardiness scheduling problem with due date assignment and resource-dependent processing times. *Ann. Oper. Res.* **2008**, *159*, 25–40. [CrossRef]
44. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
45. Nearchou, A.C. Scheduling with controllable processing times and compression costs using population-based heuristics. *Int. J. Prod. Res.* **2010**, *48*, 7043–7062. [CrossRef]
46. Wang, L.; Pan, Q.-K.; Tasgetiren, M.F. A hybrid harmony search algorithm for the blocking permutation flow shop scheduling problem. *Comput. Ind. Eng.* **2011**, *61*, 76–83. [CrossRef]
47. Spears, W.M.; De Jong, K.A. On the virtues of parameterized uniform crossover. In Proceedings of the 4th International Conference on Genetic Algorithms, San Diego, CA, USA, 13–16 July 1991.
48. Deb, K.; Agrawal, R.B. Simulated binary crossover for continuous search space. *Complex Syst.* **1995**, *9*, 115–148.
49. Eshelman, L.J. Chapter real-coded genetic algorithms and interval-schemata. *Found. Genet. Algorithm.* **1993**, *2*, 187–202.
50. Shen, X.N.; Yao, X. Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems. *Inform. Sci.* **2015**, *298*, 198–224. [CrossRef]
51. Zitzler, E.; Laumanns, M.; Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In *Evolutionary Methods Design, Optimization, Control Application in Industrial Problem, Proceedings of the EUROGEN 2001 Conference, Athens, Creece, 19–21 September 2001*; International Center for Numerical Methords in Engineering (CIMNE): Barcelona, Spain, 2001; pp. 95–100.
52. Sierra, M.; Coello Coello, C. Improving PSO-based multi-objective optimization using crowding, mutation and $\in$-dominance. In *Evolutionary Multi-Criterion Optimization*; Coello Coello, C., Hernández Aguirre, A., Zitzler, E., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 505–519.
53. Qingfu, Z.; Hui, L. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evolut. Comput.* **2007**, *11*, 712–731. [CrossRef]