HYBRID SOCIAL SPIDER OPTIMIZATION ALGORITHM WITH DIFFERENTIAL MUTATION OPERATOR FOR THE JOB-SHOP SCHEDULING PROBLEM

GUO ZHOU
Department of Science and Technology Teaching
China University of Political Science and Law, Beijing 100088, China

YONGQUAN ZHOU*
College of Information Science and Engineering
Guangxi University for Nationalities
Key Laboratories of Guangxi High Schools Complex System and Computational Intelligence
Nanning 530006, China

RUXIN ZHAO
School of Computer Science and Engineering
Nanjing University of Science and Technology, Nanjing 210094, China
College of Information Science and Engineering
Guangxi University for Nationalities, Nanning 530006, China

(Communicated by Frank Werner)

ABSTRACT. The job-shop scheduling problem is one of the well-known hardest combinatorial optimization problems. The problem has captured the interest of a significant number of researchers, but no efficient solution algorithm has been found yet for solving it to optimality in polynomial time. In this paper, a hybrid social-spider optimization algorithm with differential mutation operator is presented to solve the job-shop scheduling problem. To improve the exploration capabilities of the social spider optimization algorithm (SSO), we incorporate the DM operator (a mutation operator taken from differential evolution (DE) algorithm) into the framework of the female cooperative operator. The experimental results show that the proposed method is effective in solving job-shop scheduling compared to other optimization algorithms in the literature.

1. Introduction. The job-shop scheduling problem (JSSP) has been studied intensively by the operations research community because it is a well-known NP-complete combinatorial optimization problem [25]. Initially, researchers concentrated their efforts on exact methods, but the application of exact methods has been limited to only small instances. Therefore, studies of approximation methods have increasingly received attention during the last 30 years. The fact that various approximation algorithms have been developed for the JSP makes this problem a challenging optimization problem as well as an ideal platform for testing new solution approaches.

2010 Mathematics Subject Classification. Primary: 78M50, 90C27.
Key words and phrases. Social spider optimization, differential mutation operator, job-shop scheduling, metaheuristic optimization.

* Corresponding author: Yongquan Zhou.
There are many researchers made use one of the approximation methods, metaheuristic optimization algorithms [29], such as, tabu search algorithm [28, 30], genetic algorithm (GA) [31, 40, 21, 37, 17], simulated annealing (SA) [3, 41], ant colony optimization (ACO) [36, 39], particle swarm optimization (PSO) [20, 22], bat algorithm [9], discrete PSO algorithm [32], genetic PSO algorithm [1], local search algorithm [26], island model genetic algorithm [18], asexual genetic algorithm [2], island model memetic algorithm [19], Ant System parameter interactions method [10], neighborhood structure evaluation method [12], adaptive & opposite genetic algorithm for the JSSP [29]. The computational results show that the metaheuristic optimization algorithm can produce optimal or near-optimal values on almost all tested benchmark instances. Therefore, we can believe that metaheuristic optimization algorithm can be considered as an effective method for solving JSP.

However, due to the stubborn nature of JSSP, sole metaheuristic methods left a considerable space of improvements; consequently, recently, most of researchers tend to develop hybrid methods that combine the complementary strengths of different metaheuristic. Actually, to the best of our knowledge, the best-so-far method for JSSP is a tabu search [28]. An overview of JSSP techniques can be found in [43], while a comprehensive survey of them can be found in [13]. Recently, it has been shown that combining both of parallelization and hybridization in one framework is advantageous [15].

Social spider optimization (SSO) algorithm is proposed by Erik Cuevas in 2013 [7], it is a novel meta-heuristic optimization algorithm based on the simulation of social-spider whose members (spider species) maintain a set of complex of cooperative behavior, such as predation, mating, web design, and social interaction [23], [33]. The SSO algorithm is used to multilevel image thresholding, engineer optimization [27]. Especially, the ref [14] exhibits another method in view of a social insect method to explain the single goal flexible JSSP (FJSSP). This advancement focuses on the distinction between the two diverse search specialists (insects): males and females. In the proposed methodology, a few earlier guidelines are displayed to develop the underlying population with an abnormal state of value. In the proposed investigation, a few earlier standards are exhibited to build the underlying population with an abnormal state of value. Imitation comes about on the standard test cases demonstrate that social spider optimization (SSO) has a superior merging execution contrasted and single-goal existing roused optimization process. For the examination of twenty benchmark problems, the result demonstrates that the minimum make span achieves in the benchmark problem LA03 as 524 in SSO algorithm. This proposed work accomplished 92.33% exactness in SSO strategy contrasted with other optimization process and this algorithm reduces the computational time and less expensive.

In [16], to make a small step toward having an idea about its effectiveness in solving combinatorial optimization, in this work, a new SSO algorithm is proposed to solve HFSPMT problem with the objective of makespan minimization. The proposed algorithm is experimented on benchmark instances and compared with other existing swarm intelligence algorithms in the literature. The obtained results and the comparisons show that the performance of the proposed algorithm is highly competitive in terms of solution quality.

In this paper, to improve the exploration capabilities of the social spider optimization algorithm (SSO), we incorporate the DM operator (a mutation operator taken
from the deferential evolutionary (DE) algorithm) into the framework of the female cooperative operator. A social-spider optimization algorithm with differential mutation operator (SSO-DM) algorithm is presented according to the flight characteristics of social spiders, the proposed algorithm used differential mutation operator simulation to flight characteristics of social spiders and the proposed algorithm is applied to job-shop scheduling problem. The results of the proposed algorithm are compared with the results of other meta-heuristic optimization algorithm. The experimental results show that our proposed the social-spider optimization algorithm with differential mutation operator for solving the job-shop scheduling problem is more effective than some other meta-heuristic optimization algorithm.

2. The job-shop scheduling problem. The job-shop scheduling problem (JSSP) consists of \( n \) jobs and \( m \) machines. This paper considers one job consists of \( m \) operations; each operation uses one of \( m \) machines to complete one jobs work for a fixed time interval. The sequence of operations of one job should be predefined and maybe different for any job. In general, one job being processed on one machine is considered as an operation noted as \( o_{ij} \) (means \( j \)th job being processed on \( i \)th machine, where \( 1 \leq j \leq n \) and \( 1 \leq i \leq m \)), then every job has a sequence of \( m \) operations. The objective of JSSP is to find an appropriate operation permutation for all jobs that can minimize the makespan \( (C_{\text{max}}) \), which is the maximum completion time of a job in the schedule of \( n \times m \) operations.

For a \( n \times m \) JSSP, which can model by a set of \( m \) machines, denoted by \( M = \{1, 2, \ldots, m\} \), to process a set of \( n \times m \) operations, denoted by \( o = \{0, 1, 2, \ldots, n \times m + 1\} \). The operations 0 and \( n \times m + 1 \), which are dummy operations, represent the initial and the last operations, respectively. Dummy operation used to model the JSSP and need not any processing time. A precedence constraint used to let operation \( i \) to schedule after all predecessor operations included in \( P_i \) are finished. Further, one operation can be scheduled on an appointed machine that is free.

According to the description listed in Table 1, the model of the JSSP can be defined as

\[
\begin{align*}
\text{Minimize} & \quad O_{n \times m + 1}(C_{\text{max}}) \\
O_q & \leq O_i - t_i, \text{ where } i = 0, 1, 2, \ldots, n \times m + 1 \text{ and } q \in P_i \\
\sum_{i \in A(t)} \omega_{im} & \leq 1, \text{ where } m \in M \text{ and } t \geq 0 \\
O_i & \geq 0, \text{ where } i = 0, 1, 2, \ldots, n \times m + 1
\end{align*}
\]
The fitness function in Equation 1 is to minimize makespan that is the completion time of the last operation. The constraint of precedence relationship defined by Equation 2. In Equation 3, it shows that one machine can process at most one operation at a time. The finish time must be positive by the constraint stated in Equation 4.

3. SSO and SSO-DM. This section will mainly introduce SSO and SSO-DM. The description of SSO in refers [8, 5, 6], which introduces in detail the steps of SSO algorithm. SSO-DM is a new optimization algorithm proposed in this article, it is the combination of SSO and differential mutation operator, SSO-DM is described in the second part of this section.

3.1. A sample Theorem. The SSO algorithm [7] contains two different search operators (spiders): males and females. According to different gender, they were carried out in different division. The number of females ($N_f$) accounted for the entire population ($N$) of 65% ~ 90%. The number of females ($N_f$) is defined as

$$N_f = \text{floor}[(0.9 - \text{rand} \times 0.25) \times N]$$  \hspace{1cm} (5)

where ‘rand’ is a random number in the range [0,1], ‘floor’ is the integral function. The number of male spiders ($N_m$) is defined as

$$N_m = N - N_f$$  \hspace{1cm} (6)

The group $F$ stands for female population ($F = \{f_1, f_2, \ldots, f_{N_f}\}$), the group $M$ stands for male population ($M = \{m_1, m_2, \ldots, m_{N_m}\}$) and the group $S$ stands for entire population, such that $S = F \cup M$.

3.1.1. The weight of the social-spider and the object function. In SSO, each spider gets a weight ($w_i$), which represents the quality of the solution in the algorithm. The weight of social-spider is defined as follows

$$w_i = \frac{J(s_i) - \text{worst}_s}{\text{best}_s - \text{worst}_s}$$  \hspace{1cm} (7)

where $J(s_i)$ is the objective function value of the social-spider individual ($s_i$). The values of worst$_s$ and best$_s$ are defined as

$$\text{best}_s = \max_{k \in \{1,2,\ldots,N\}} (J(s_k))$$  \hspace{1cm} (8)

$$\text{worst}_s = \min_{k \in \{1,2,\ldots,N\}} (J(s_k))$$  \hspace{1cm} (9)

3.1.2. The vibration model of SSO algorithm. The vibration model mainly simulates the information exchange between social-spider, it is mainly related to the weight and distance of social-spiders. The vibrations perceived by the individual ($i$) as a result of the information transmitted by the member ($j$) are modeled as following equation

$$\text{Vib}_{ij} = w_j \times \exp (-d_{ij}^2)$$  \hspace{1cm} (10)

where $d_{ij}$ represents the Euclidean distance between individual ($i$) and individual ($j$), such that $d_{ij} = \|s_i - s_j\|$. 


3.1.3. Generation of population. The generation of the population of the SSO algorithm is defined as follows

\[
\begin{align*}
    f^0_{ij} &= p^\text{low}_j + \text{rand}(0, 1) * (p^\text{high}_j - p^\text{low}_j) \quad \text{with } i = 1, \ldots, N \quad (11) \\
m^0_{ij} &= p^\text{low}_j + \text{rand}(0, 1) * (p^\text{high}_j - p^\text{low}_j) \quad \text{with } i = 1, \ldots, N' 
\end{align*}
\]

where \( j = 1, 2, \ldots, n \), \( i \) and \( j \) are individual indices, \( f_i \) and \( m_i \) represent the position of female and male spiders respectively. Whereas zero signals the initial population, the function ‘rand’ generates a random number between 0 and 1. \( p^\text{high}_j \) and \( p^\text{low}_j \) represent the upper and lower bounds of the initial parameters respectively.

3.1.4. The cooperation model of SSO algorithm.

**Female cooperation model:** In SSO algorithm, the cooperative behavior of female social-spiders is regarded as the change of their positions, which is defined as

\[
f_{f_i}^{k+1} = \begin{cases} 
    f_{f_i}^k + \alpha * \text{Vib}_c * (s_c - f_{f_i}^k) + \beta * \text{Vib}_b * (s_b - f_{f_i}^k) + \\
    \delta * (\text{rand} - \frac{1}{2}) & r_m < \text{PF} \\
    f_{f_i}^k - \alpha * \text{Vib}_c * (s_c - f_{f_i}^k) - \beta * \text{Vib}_b * (s_b - f_{f_i}^k) + \\
    \delta * (\text{rand} - \frac{1}{2}) & r_m > \text{PF}
\end{cases} 
\]

where \( \alpha, \beta, \delta \) and ‘rand’ are random numbers between [0, 1], whereas \( k \) represents the iteration number. The individual \( s_c \) and \( s_b \) represent the nearest member to individual \( i \) that holds a higher weight of individual and the entire population \( S \) that holds a highest weight of individual respectively. The \( \text{Vib}_c \) represents the nearest member to individual \( i \) that holds a higher weight and produces the vibration \( \text{Vib}_c \). The \( \text{Vib}_b \) represents the best individual of the entire population \( S \) who produces the vibration \( \text{Vib}_b \). Since the final movement of attraction or repulsion depends on several random phenomena, the selection is modeled as a stochastic decision. For this operation, a uniform random number \( r_m \) is generated within the range [0, 1]. If \( r_m \) is smaller than a threshold, an attraction movement is generated; otherwise, a repulsion movement is produced.

**Male cooperation model:** The male population contains two different types of social-spiders: dominant members \( D \) and non-dominant members \( ND \), according to their own characteristics to perform different division of labor for male spiders. Dominant members \( D \) performs the mating operation and non-dominant members \( ND \) performs the operation to protect the food. The male population \( M = \{m_1, m_2, \ldots, m_{N_m}\} \) is arranged according to their weight value in decreasing order. Thus, the individual whose weight \( w_{N_{f+m}} \) is located in the middle is considered the median male member. Since indexes of the male population \( M \) in regard to the entire population \( S \) are increased by the number of female members \( N_f \), the median weight is indexed by \( N_{f+m} \). According to this, the position of male spiders is defined as follows

\[
m_{i}^{k+1} = \begin{cases} 
    m_{i}^k + \alpha * \text{Vib}_f * (s_f - m_{i}^k) + \delta * (\text{rand} - \frac{1}{2}) & w_{N_{f+i}} > w_{N_{f+m}} \\
    m_{i}^k + \alpha * \left( \frac{\sum_{h=1}^{N_{f+m}} m_{h} * w_{N_f + h}}{w_{N_f+m} - m_{i}^k} \right) & w_{N_{f+i}} \leq w_{N_{f+m}}
\end{cases}
\]
where the individual $s_f$ represents the nearest female individual to the male member $i$, whereas $(\sum_{h=1}^{N_m} m_h^k \cdot w_{N_f+h})/ (\sum_{h=1}^{N_m} w_{N_f+h})$ correspond to the weighted mean of the male population $M$.

3.1.5. The reproduction model of SSO algorithm. In SSO algorithm, Dominant members $D$ and female spiders perform the mating operation. Each male spider has a specific mating radius $r$, which is defined as

$$r = \frac{\sum_{j=1}^{n} (p_{j}^{\text{high}} - p_{j}^{\text{low}})}{2n} \quad (15)$$

During mating, the weight of spiders involved in mating that can affects the quality of the next generation of spiders. The influence probability $P_{s_i}$ of each spider is assigned by the roulette method, which is defined as follows

$$P_{s_i} = \frac{w_i}{\sum_{j \in T^g} w_j} \quad (16)$$

where $i \in T^g$. If the weight of the newly formed spider is greater than the lightest spider of the previous spider population, the new spider will replace the lightest spider in the spider population. Instead, the new spider will be abandoned and the spider population will not change.

3.1.6. The computational procedure of the SSO algorithm. The computational procedure for the SSO algorithm can be summarized as follows

**Step 1:** Think $N$ as the total number of n-dimensional colony members, define the number of male spiders $N_m$ and females spiders $N_f$ in the entire population $S$.

$$N_f = \text{floor}[(0.9 - \text{rand} \times 0.25) \times N] \quad (17)$$

where $N_m = N - N_f$, ‘rand’ is a random number between [0, 1] and ‘floor’ maps a real number to an integer number.

**Step 2:** Initialize randomly the female ($F = \{f_1, f_2, \ldots, f_{N_f}\}$), male ($M = \{m_1, m_2, \ldots, m_{N_m}\}$) members (where $S = \{s_1 = f_1, s_2 = f_2, \ldots, s_{N_f} = f_{N_f}, s_{N_f+1} = m_1, s_{N_f+2} = m_2, \ldots, S_N = m_{N_m}\}$) and calculate the radius of mating

$$r = \frac{\sum_{j=1}^{n} (p_{j}^{\text{high}} - p_{j}^{\text{low}})}{2n} \quad (18)$$

**Step 3:** Calculation the weight of every spider of $S$

$$w_i = \frac{J(s_i) - \text{worst}_s}{\text{best}_s - \text{worst}_s} \quad \text{for } (i = 1; i < N + 1; i++)$$

**Step 4:** Female spider’s movement according to the female cooperative operator

$$f_{k+1} = f_k - \alpha \ast \text{Vibc}_i \ast (s_c - f_k) + \beta \ast \text{Vibb}_i \ast (s_b - f_k) + \delta \ast (\text{rand} - \frac{1}{2})$$

if ($r_m \geq \text{PF}$)

$$f_{k+1} = f_k + \alpha \ast \text{Vibc}_i \ast (s_c - f_k) + \beta \ast \text{Vibb}_i \ast (s_b - f_k) + \delta \ast (\text{rand} - \frac{1}{2})$$
\[ \beta \ast \text{Vib}_i \ast (s_b - f_i) + \delta \ast (\text{rand} - \frac{1}{2}) \]

endif
endfor

Step 5: Move the male spiders according to the male cooperative operator and find the median male individual \( w_{N_f+m} \) from \( M \)

for \((i = 1; i < N_m + 1; i + +)\)
   Calculate \( \text{Vibf}_i \)
   if \((w_{N_f+i} > w_{N_f+m})\)
      \( m_i^{k+1} = m_i^k + \alpha \ast \text{Vibf}_i \ast (s_f - m_i^k) + \delta \ast (\text{rand} - \frac{1}{2}) \)
   elseif
      \( m_i^{k+1} = m_i^k + \alpha \ast \left( \frac{\sum_{h=1}^{N_m} m_h^k \ast w_{N_f+h}}{\sum_{h=1}^{N_m} w_{N_f+h}} - m_i^k \right) \)
   endif
endfor

Step 6: Perform the mating operation

for \((i = 1; i < N_m + 1; i + +)\)
   if \((m_i \in D)\)
      Find \( E^i \)
      if \((E^i \text{ is not empty})\)
         From \( s_{\text{new}} \) using the roulette method
         if \((w_{\text{new}} > w_{\text{wo}})\)
            \( s_{\text{wo}} = s_{\text{new}} \)
         endif
      endif
   endif
endfor

Step 7: If the stop criteria is met, the process is finished; otherwise go to Step 3.

3.2. The SSO-DM algorithm.

3.2.1. Differential mutation operator (DM). The differential evolution (DE) proposed by Storn and Price [34], is a stochastic search algorithm and has been successfully applied to solve optimization problems. The differential mutation operation is defined by Equation 19. The calculation formula of differential mutation operation is as follows

\[ x_i^{k+1} = x_i^k + (1 - Y) \ast (x_i^k - x_m^k) + Y \ast (x_y^k - x_j^k) \]  

(19)

where \( j, l, g \) and \( m \) are random integers, they are uniformly selected from the set \( X = \{x_1, x_2, \ldots, x_n\} \) and \( i \neq j \neq l \neq m \neq g \). In other word, the indices are mutually different. \( Y \) is defined as

\[ Y = \frac{k_{\text{cur}}}{k_{\text{max}}} \]

(20)

where \( k_{\text{cur}} \) is the current iteration and \( k_{\text{max}} \) is the number of maximum iterations.
3.2.2. The flight characteristics of social spiders. The flight is properties of many social spiders have. Social spiders can achieve rapid diffusion by means of flight behavior at a certain stage. The flight is an important way to realize long distance diffusion of spiders, the flight distance sometimes can reach hundreds of kilometers. This is also the main reason for the rapid spread of social spiders. Inspired by this feature, assume that social spiders sometimes change their position according to this characteristic[38].

3.2.3. The necessary properties and theoretical analysis of the SSO-DM algorithm.

The necessary properties (flight characteristics of social spiders): The article uses differential mutation operator to simulate the flight characteristics of social spiders. Suppose that each female social spider have flight characteristic, the calculation formula of the position change of the flying characteristic of the female social spider is as follows

\[
f_{i}^{k+1} = f_{i}^{k} + (1 - Y) \ast (f_{i}^{k} - f_{m}^{k}) + Y \ast (f_{g}^{k} - f_{j}^{k})
\]

where \(j, l, g, m\) are random integers, they are uniformly selected from the set \(F = \{f_1, f_2, \ldots, f_N\}\) and \(i \neq j \neq l \neq m \neq g\). In other word, the indices are mutually different. \(Y\) is described in Equation 20. Because the flight characteristics of social spiders are shown in a stage, so define a random number \(Z\) and a threshold value \(R\). Where \(Z\) is a random number between \([0, 1]\), \(R\) is a constant with a value of 0.5. When \(R < Z\), female social spiders perform the operation of the flight characteristics, otherwise, female social spiders perform the operation of the original algorithm.

Theoretical analysis of the SSO-DM algorithm: In the SSO-DM algorithm, \(Z\) is a control variable for the flight characteristics of female social spiders. It always obliges the female social spiders to take random executes the operation of the flight characteristics, which greatly enhances population diversity in the algorithm and promotes exploration of the search space that leads to finding diverse solutions during optimization. This mechanism is very helpful for resolving local optima stagnation even when the SSO-DM algorithm is in the exploitation phase.

The proposed SSO-DM consists essentially in a strong co-operation of the two evolutionary algorithms. The main difference with SSO is in the mutation operation. This efficient combination strategy of DM and SSO improves the global search capability, avoiding convergence to local minima and accelerates the convergence. The SSO-DM algorithm can be summarized as follow

Step 1-Step 3: The same as the SSO algorithm.

Step 4: Female spider’s movement according to the female cooperative operator.

\[
\text{for}(i = 1; i < N_f + 1; i++)
\]

Calculate Vibci and Vibt

Create \(Z\) as a random number in range(0, 1)

\(\text{if}(Z < 0.5)\)

\(f_{i}^{k+1} = f_{i}^{k} + (1 - Y) \ast (f_{i}^{k} - f_{m}^{k}) + Y \ast (f_{g}^{k} - f_{j}^{k})\)

\(\text{else}\)

\(\text{if}(r_m < PF)\)

\(f_{i}^{k+1} = f_{i}^{k} + \alpha \ast \text{Vibci} \ast (s_c - f_{i}^{k})\)
\[
\beta \ast \text{Vib}_i \ast (s_b - f_i^k) + \delta \ast (\text{rand} - \frac{1}{2}) \\
\text{else}
\]
\[
f_{i+1}^k = f_i^k - \alpha \ast \text{Vib}_c \ast (s_c - f_i^k) - \\
\beta \ast \text{Vib}_i \ast (s_b - f_i^k) + \delta \ast (\text{rand} - \frac{1}{2})
\]
\text{endif}
\text{endif}
\text{endfor}

**Step 5-Step 7:** The same as the original algorithm.

3.2.4. *The proposed algorithm for JSSP.* Some issues are in applying SSO-DM algorithm to solve JSSP. The original SSO-DM design is developed to solve continuous function. But, JSSP is a combinatorial problem, the solution space is discrete. The first issue is to find a suitable representation which the social spiders of SSO-DM can simulate an operation permutation schedule of JSSP. In the SSO-DM algorithm, each location of spiders on the communal web represents a possible solution. The vector of location can be composed of two parts: operation sequence vector and machine assignment vector. The first part is used to sequence the entire operations processed on all machines while the second part is used to assign each operation to a proper machine. The dimension of the operation sequence vector is equal to the number of all operations as well as the machine assignment. In the operation sequence, the operations of each job are expressed by its job number. The \( j \)th occurrence of a job is the \( j \)th operation in the sequence of this job.

4. *Simulation experiments and result analysis.* The paper use four instances that are taken from the OR Library [4] as test benchmarks to test our new proposed algorithm, named SSO-DM. In the 4 instances, FT20 were designed by Fisher and Thompson (FT) [11] and the rest of instances that were designed by Lawrence (LA), Applegate & Cook (ORB) and Yamada & Nakano (YN). The paper coded the SSO-DM algorithm with the environment of Matlab2012a, and simulated it with an Inter (R) Core(TM) i5-4590 CPU and 4 GB memory. The rest of this section is organized as follows: experimental setup is given in Section 4.1; the comparison of each algorithm performance is shown in Section 4.2; the result analysis in Section 4.3.

4.1. *Experimental setup.* The proposed Social Spider Optimization algorithm with Differential Mutation operator (SSO-DM) compared with swarm intelligence optimization algorithms, such as PSO [20], and evolutionary algorithms, IGA [15], DE [42], evolutionary algorithm-based ensemble model [35] using the mean and standard deviation to compare their optimal performance. Among them, the SSO is the original algorithm of SSO-DM, it does not have differential mutation operation. The parameters settings of algorithms are as follows. Population size \( N = 50 \), and the iteration number is 100. Twenty independent runs were made for the five algorithms, the results obtained by five algorithms are presented in Table 2, and the evolution curves and the variance diagram of four instances obtained by five algorithms are presented in Figure 1, 2, 3, 4 and Figure 5, 6, 7, 8, respectively.

4.2. *Comparison of each algorithm performance.* In Table 2, Name presents examples of names; Size represents dimension size; each number in the column Mean is the average global optimal value of 20 times independent operation; Best is the best global optimal value of 20 time independent operation; Worst is the worst
| Name  | Size \((n \times m)\) | Algorithm | Best   | Worst  | Mean   | Std.  |
|-------|---------------------|-----------|--------|--------|--------|-------|
| FT20  | 20*5                | PSO       | 1374.00| 1521.00| 1442.50| 42.02 |
|       |                     | IGA       | 1744.00| 2527.00| 2025.50| 198.95|
|       |                     | DE        | 1456.00| 1554.00| 1506.00| 27.64 |
|       |                     | SSO       | 1527.00| 1527.00| 1527.00| 0     |
|       |                     | SSO-DM    | 1374.00| 1374.00| 1374.00| 0     |
| LA40  | 15*15               | PSO       | 1498.00| 1732.00| 1576.05| 59.79 |
|       |                     | IGA       | 2154.00| 2803.00| 2340.25| 155.90|
|       |                     | DE        | 1691.00| 1824.00| 1767.05| 36.46 |
|       |                     | SSO       | 1834.00| 1834.00| 1834.00| 0     |
|       |                     | SSO-DM    | 1528.00| 1528.00| 1528.00| 0     |
| ORB10 | 10*10               | PSO       | 1039.00| 1263.00| 1150.05| 48.84 |
|       |                     | IGA       | 1431.00| 2121.00| 1761.25| 158.12|
|       |                     | DE        | 1190.00| 1293.00| 1244.40| 25.04 |
|       |                     | SSO       | 1345.00| 1345.00| 1345.00| 0     |
|       |                     | SSO-DM    | 1114.00| 1114.00| 1114.00| 0     |
| YN4   | 20*20               | PSO       | 1340.00| 1607.00| 1425.15| 64.84 |
|       |                     | IGA       | 1826.00| 2192.00| 1997.90| 116.48|
|       |                     | DE        | 1486.00| 1601.00| 1570.75| 26.15 |
|       |                     | SSO       | 1583.00| 1583.00| 1583.00| 0     |
|       |                     | SSO-DM    | 1492.00| 1492.00| 1492.00| 0     |

Table 2. Simulation results for FT, LA, ORB and YN

The global optimal value of 20 times independent operation; each number in the column Std. represents standard deviation value of 20 times independent operation.

In addition, we compare the running time of the algorithm in detail. The experimental results are listed in Table 3. The experimental data represent the average running time of the algorithm in this table. Its unit of time is seconds (s).

A non-parametric statistical significance proof known as the Wilcoxon rank sum test for independent samples has been conducted over the average best-so-far (Mean) data of Table 2 to Table 3, with a 5% significance level. Table 4 reports the p-values produced by Wilcoxon test for the pair-wise comparison of the average best-so-far of four groups. Such groups are constituted by SSO-DM vs. PSO, SSO-DM vs. IGA, SSO-DM vs. DE and SSO-DM vs. SSO. As a null hypothesis, it is assumed that there is no significant difference between mean values of the two algorithms. The alternative hypothesis considers a significant difference between the average best-so-far values of both approaches.

4.3 Result analysis. Seen from Table 2 above that the mean value and standard deviation produced by SSO-DM are better than the other algorithms in most cases. Figures 1, 2, 3, 4 show the evolution curves of fitness value for job-shop scheduling problem. From these Figs, can easily find that SSO-DM converges faster than other algorithms in most cases, and the values obtained by SSO-DM achieve the optimal values of benchmark problems in some cases. These show that SSO-DM has a faster convergence speed and a better precision than SSO algorithm (without differential mutation operation) and other population based algorithms. Figures 5, 6, 7, 8 show the ANOVA test for job-shop scheduling problem. From these figs, can discover that the standard deviation of SSO-DM is much smaller, and the number of outlier
is less than other algorithms. These imply that SSO-DM has a great performance with a high degree of stability. In sum, proposed SSO-DM is an algorithm with fast convergence speed, better mean value and a great performance of stability. After analyzing above, the conclusion can be easily drawn that SSO-DM has a great ability for solving job-shop scheduling problem according to the experimental results.

In Table 3, we compare the running time of the algorithm. As can be seen from the table, in most cases, the running time of DE is the shortest, followed by that of the original algorithm (without differential mutation operation). The running time of our proposed algorithm ranks third overall. The reason is that we add differential mutation operation, which increases the computational complexity of the algorithm. However, the running time of the proposed algorithm has not increased much, and it is still in the same order of magnitude as other algorithms.

Table 4 shows the experimental values of Wilcoxon's rank sum test. All p-values reported in Table 1 are less than 0.05 (5% significance level), which is a strong
evidence indicates that SSO-DM results are statistically significant and it has not occurred by coincidence.

Another finding in the results is the poor performance of IGA. In contrary to SSO-DM algorithm, there is no mechanism for significant abrupt movements in the search space and this is likely to be the reason for the poor performance of IGA. It is also worth discussing the poor performance of the DE algorithm in this section. DE algorithm has been designed based on mutation mechanism, which maintains the diversity of population and promotes exploration. However, due to the influence of the choice operation, the individual differences will gradually decrease with the increase of the number of iterations. At the same time, the decrease of individual difference also affects the diversity of the variation, which leads to the premature convergence of the algorithm, which is one of the main reasons for the poor performance of DE. Among these comparative algorithms, the optimization performance of PSO is better. In some cases, it can get the best
Table 3. Experimental results of running time of the algorithm

|        | IGA  | PSO  | DE   | SSO  | SSO-DM |
|--------|------|------|------|------|--------|
| FT20   | 4.7350 | 1.3626 | 0.6521 | 0.7993 | 0.8725 |
| LA40   | 4.2764 | 2.6175 | 1.0695 | 1.2537 | 1.2961 |
| ORB10  | 2.7547 | 1.8100 | 1.1091 | 1.3127 | 1.0230 |
| YN04   | 6.8851 | 3.0504 | 1.9461 | 2.3814 | 2.5842 |

Table 4. $p$-values produced by Wilcoxon’s test comparing SSO-DM vs. PSO, SSO-DM vs. IGA, SSO-DM vs. DE and SSO-DM vs. SSO, over the average best-so-far (Mean) values from Table 1 to Table 4.

|        | PSO      | IGA      | DE        | SSO       |
|--------|----------|----------|-----------|-----------|
| FT20   | 7.9772E-09 | 8.0065E-09 | 4.0136E-03 | 4.6826E-10 |
| LA40   | 7.9918E-09 | 7.9918E-09 | 8.0065E-09 | 4.6826E-10 |
| ORB10  | 7.9918E-09 | 8.0065E-09 | 7.9772E-09 | 4.6826E-10 |
| YN04   | 2.0993E-07 | 8.0065E-09 | 4.0289E-02 | 4.6826E-10 |

Experimental mean. However, its stability is poor and the running time of the algorithm is longer.

The reason for getting better mean value provided by the SSO-DM algorithm is that this algorithm is equipped with the parameter $Z$ that control variables for the flight characteristics of female social spiders, it always obliges the female social spiders to execute the operation of the flight characteristics. This promotes exploration ability of algorithm, which leads to finding diverse solutions during optimization. In addition, two thirds of the individuals in the SSO-DM algorithm are devoted to exploration of the search space and the rest to exploitation, this design guarantees the balance between exploration and exploitation, this is also a reason why the proposed algorithm always guiding search agents to exploitation the most promising regions of the search space, which also assist this algorithm to provide remarkable results.

5. Conclusions. Although there is a huge amount of literature on SSO, the SSO algorithm for JSSP does not have rich literature. In this paper, we proposed a new approach SSO-DM for JSSP to minimize makespan. The performance of the new approach is evaluated compare with the results obtained from PSO, IGA, DE and SSO for four instances, and obtained results by SSO-DM show the effectiveness of the proposed approach. The proposed algorithm can consider as a new method for solving JSSP.

There are a number of research directions, which can consider as useful extensions of this research. Although four classical cases used to test the performance of the proposed algorithm, a more comprehensive computational study should to test the quality of the solution of the proposed algorithm. Furthermore, applying SSO-DM to other discrete combinatorial optimization problems is also future work. The development of SSO is still ongoing. There are still many unknown areas in SSO research, such as the mathematical validation of Social Spider swarm theory.
Acknowledgments. This work is supported by National Science Foundation of China under Grants No. 61563008, the Project of China University of Political Science and Law Research Innovation under Grant No. 10818441, and Young Scholar Fund of China University of Political Science and Law Grant No. 10819144.

REFERENCES

[1] R. F. Abdel-Kader, An improved PSO algorithm with genetic and neighborhood-based diversity operators for the job shop scheduling problem, Applied Artificial Intelligence, 32 (2018), 433–462.

[2] M. Amirghasemi and R. Zamani, An effective asexual genetic algorithm for solving the job shop scheduling problem, Computers & Industrial Engineering, 83 (2015), 123–138.

[3] A. Elmi, M. Solimanpurb, S. Topaloglua and A. Elmic, A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts, Computers & Industrial Engineering, 61 (2011), 171–178.

[4] J. E. Beasley, Or-library: Distributing test problems by electronic mail, J. of the Operational Research Society, 41 (1990), 1069–1072.

[5] E. Cuevas, M. A. Díaz Cortés and D. A. O. Navarro, Advances of Evolutionary Computation: Methods and Operators, Studies in Computational Intelligence, 629, Springer, 2016, 9–33.

[6] E. Cuevas, M. Cienfuegos, R. Rojas and A. Padilla, Computational Intelligence Applications in Modeling and Control, Studies in Computational Intelligence, 575, Springer, 2015, 123–146.

[7] E. Cuevas, M. Cienfuegos, D. Zaldivar and M. Perez-Cisneros, A swarm optimization algorithm inspired in the behavior of the social-spider, Expert Systems with Applications, 40 (2013), 6374–6384.

[8] E. Cuevas, V. Osuna and D. Oliva, Evolutionary Computation Techniques: A Comparative Perspective, Studies in Computational Intelligence, 686 (2017), 65–93.

[9] T. K. Dao, T. S. Pan and J. S. Pan, Parallel bat algorithm for optimizing makespan in job shop scheduling problems, J. of Intelligent Manufacturing, 29 (2018), 451–462.

[10] N. Fighali, C. Özkalı, O. Engin A. and Fighali, Investigation of Ant System parameter interactions by using design of experiments for job-shop scheduling problems, Computers & Industrial Engineering, 56 (2009), 538–559.

[11] H. Fisher and G. L. Thompson, Probabilistic learning combinations of local job-shop scheduling rules, in Industrial Scheduling, Prentice Hall, 1963, 225–251.

[12] L. Gao, X. Li, X. Wen, C. Lu and F. Wen, A hybrid algorithm based on a new neighborhood structure evaluation method for job shop scheduling problem, Computers & Industrial Engineering, 88 (2015), 417–429.

[13] A. S. Jain and S. Meeran, Deterministic job-shop scheduling: Past, present and future, European J. of Operational Research, 113 (1999), 390–434.

[14] S. Kavitha, P. Venkumar, N. Rajini and P. Pitchipoo, An efficient social spider optimization for flexible job shop scheduling problem, J. of Advanced Manufacturing Systems, 17 (2018), 181–196.

[15] M. Kurdi, A new hybrid island model genetic algorithm for job shop scheduling problem, Computers & Industrial Engineering, 88 (2015), 273–283.

[16] M. Kurdi, A Social Spider Optimization Algorithm for Hybrid Flow Shop Scheduling with Multiprocessor Task, 12th International NCM Conference: Challenges in Industrial Engineering & Operation Management, 2018.

[17] M. Kurdi, An effective genetic algorithm with a critical-path-guided Giffler and Thompson crossover operator for job shop scheduling problem, International J. of Intelligent Systems and Applications in Engineering, 7 (2019), 13–18.

[18] M. Kurdi, An effective new island model genetic algorithm for job shop scheduling problem, Comput. Oper. Res., 67 (2016), 132–142.

[19] M. Kurdi, An improved island model memetic algorithm with a new cooperation phase for multi-objective job shop scheduling problem, Computers & Industrial Engineering, 111 (2017), 183–201.

[20] T.-L. Lin, S.-J. Horng, T.-W. Kao, Y.-H. Chen, R.-S. Run, R.-J. Chen, J.-L. Lai and I.-H. Kuo, An efficient job-shop scheduling algorithm based on particle swarm optimization, Expert Systems with Applications, 37 (2010), 2629–2636.
[21] M. Liu, Z.-J. Sun, J.-W. Yan and J.-S. Kang, An adaptive annealing genetic algorithm for the job-shop planning and scheduling problem, Expert Systems with Applications, 38 (2011), 9248–9255.
[22] S. Lu, C. Sun and Z. Lu, An improved quantum-behaved particle swarm optimization method for short-term combined economic emission hydrothermal scheduling, Energy Conversion and Management, 51 (2010), 561–571.
[23] T. B. Lubin, The Evolution of Sociality in Spiders, Advances in the Study of Behavior, 37 (2007), 83–145.
[24] A. Muthiah and R. Rajkumar, A novel algorithm for solving job-shop scheduling problem, Mechanika, 23 (2017), 610–617.
[25] B. Naderi, S. M. T. Fatemi Ghomi, M. Aminnayeri and M. Zandieh, Scheduling open shops with parallel machines to minimize total completion time, J. Comput. Appl. Math., 5 (2011), 1273–1287.
[26] Y. Nagata and I. Ono, A guided local search with iterative ejections of bottleneck operations for the job shop scheduling problem, Comput. Oper. Res., 90 (2018), 60–71.
[27] S. Ouadfel and A. Taleb-Ahmed, Social spiders optimization and flower pollination algorithm for multilevel image thresholding: A performance study, Expert Syst. with Applications, 55 (2016), 566–584.
[28] B. Peng, Z. Li and T. C. E. Cheng, A tabu search/path relinking algorithm to solve the job shop scheduling problem, Comput. Oper. Res., 53 (2015), 154–164.
[29] P. Pongchairerks, A Two-Level Metaheuristic Algorithm for the Job-Shop Scheduling Problem, Complexity, 1 (2019), 1–11.
[30] A. Ponsich and C. A. Coello Coello, A hybrid Differential Evolution-Tabu Search algorithm for the solution of Job-Shop Scheduling Problems, Applied Soft Computing, 13 (2013), 462–474.
[31] R. Qing-dao-er-ji and Y. Wang, A new hybrid genetic algorithm for job shop scheduling problem, Comput. Oper. Res., 39 (2012), 2291–2299.
[32] K. Rameshkumar and C. Rajendran, A novel discrete PSO algorithm for solving job shop scheduling problem to minimize makespan, IOP Conference Series: Materials Science and Engineering, 310 (2018), 21–43.
[33] F. Ramezani and S. Lotfi, Social-Based Algorithm (SBA), Applied Soft Computing, 13 (2013), 2837–2856.
[34] R. Storn and K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim., 11 (1997), 341–359.
[35] C. J. Tan, A. Yeoh, C. P. Lim, S. Hanoun, W. P. Wong, C. K. Loo and S. Nahavandi, Application of an evolutionary algorithm-based ensemble model to job-shop scheduling, J. of Intelligent Manufacturing, 30 (2019), 879–890.
[36] R. F. Tavares Neto and M. Godinho Filho, Literature review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research, Engineering Applications of Artificial Intelligence, 26 (2013), 150–161.
[37] W. Teekeng and A. Thammano, Modified genetic algorithm for flexible job-shop scheduling problems, Procedia Computer Science, 12 (2012), 122–128.
[38] C. M. Xiang, Observation on the flying habits of social spiders, Chinese J. of Zoology, 3 (1986), 11–11.
[39] L.-N. Xing, Y.-W. Chen, P. Wang, Q.-S. Zhao and J. Xiong, A knowledge-based ant colony optimization for flexible job shop scheduling problems, Applied Soft Computing, 10 (2010), 888–896.
[40] R. Yusof, M. Khalid, G. T. Hui and S. M. Yusof, Solving job shop scheduling problem using a hybrid parallel micro genetic algorithm, Applied Soft Computing, 11 (2011), 5782–5792.
[41] R. Zhang and C. Wu, A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective, Comput. Oper. Res., 38 (2011), 854–867.
[42] G. Zobolas, C. D. Tarantilis and G. Ioannou, A hybrid evolutionary algorithm for the job shop scheduling problem, J. of the Oper. Res. Society, 60 (2009), 221–235.
[43] G. I. Zobolas, C. D. Tarantilis and G. Ioannou, Exact, heuristic and meta-heuristic algorithms for solving job shop scheduling problems, in Metaheuristics for Scheduling in Industrial and Manufacturing Applications, Studies in Computational Intelligence, 2, Springer, Berlin, 2008, 19–40.

Received July 2018; 1st revision March 2019; final revision July 2019.
E-mail address: zhouguo@cupl.edu.cn
E-mail address: yongquanzhou@126.com
E-mail address: nnzhaoms@126.com