Application of Firefly Algorithm and Its Parameter Setting for Job Shop Scheduling

Aphirak Khadwilard[1,*], Sirikarn Chansombat[2], Thatchai Thepphakorn[2], Peeraya Thapatsuwan[2], Warattapop Chainate[3] and Pupong Pongcharoen[2]

Abstract

Job shop scheduling problem (JSSP) is one of the most famous scheduling problems, most of which are categorised into Non-deterministic Polynomial (NP) hard problem. The objectives of this paper are to i) present the application of a recent developed metaheuristic called Firefly Algorithm (FA) for solving JSSP; ii) investigate the parameter setting of the proposed algorithm; and iii) compare the FA performance using various parameter settings. The computational experiment was designed and conducted using five benchmarking JSSP datasets from a classical OR-Library. The analysis of the experimental results on the FA performance comparison between with and without using optimised parameter settings was carried out. The FA with appropriate parameters setting that got from the experiment analysis produced the best-so-far schedule better than the FA without adopting parameter settings.

Keywords: Scheduling, Job shop, Metaheuristics, Firefly Algorithm, Experimental design, Parameter setting
(Selected from 1st Symposium on Hands-on Research and Development, Chiang Mai)

[1]Department of Mechanical Engineering, Faculty of Engineering, Rajamangala University of Technology Lanna Tak.

[2] Department of Industrial Engineering, Faculty of Engineering, Naresuan University.

[3] Faculty of Liberal Arts and Science, Kasetsart University Kamphaeng Saen Campus.

[*] Corresponding author, E-mail: aphirak@rmutl.ac.th, k_aphirak@hotmail.com Received 1 August 2011; Accepted 7 December 2011

1. Introduction

Scheduling problem is a decision making process involving the allocation of resources over time to perform a set of activities or tasks. Scheduling problems in their static and deterministic forms are simple to describe and formulate, but are difficult to solve as it involves complex combinatorial optimisation. For example, if there are m machines, each of which is required to perform n independent operations. The combination can be potentially exploded up to $(n!)^m$ operational sequences. Job shop scheduling is one of the most famous scheduling problems, most of which are categorised into NP hard problem. This means that due to the combinatorial explosion, even a computer can take unacceptably large amount of time to seek a satisfied solution on even moderately large scheduling problem. Another potential issue of complexity is the assembly relationship [1,2].

Job shop scheduling problem (JSSP) is comprised of a set of independent jobs or tasks (J), each of which consists of a sequence of operations (O). Each operation is performed on machine (M) without interruption during processing time. The main purpose of JSSP is usually to find the best machine schedule for servicing all jobs in order to optimise either single criterion or multiple scheduling objectives (measures of performance) such as the minimisation of the makespan (C_{max}) or the penalty costs of tardiness and/or earliness.

Various optimisation approaches have been widely applied to solve the JSSP. Conventional methods based on mathematical model and/or full numerical search (for example, Branch and Bound [3,4] and Lagrangian Relaxation [5,6]) can guarantee the optimum solution. They have been successfully used to solve JSSP. However, these methods may highly consume computational time and resources even for solving a moderate-large problem size and therefore impractical if the computational limitation is exist. Approximation optimisation methods or metaheuristics (e.g. Tabu Search [7] and Simulated Annealing [8]), that usually conduct stochastic steps in their search process, have therefore been recently received more attention for solving a large-size problem in the last few decades. However, it does not guarantee the optimum solution.

Firefly Algorithm (FA) was recently introduced by Yang [9], who was inspired by firefly behaviours. FA has been widely applied to solve continuous mathematical functions [9, 10]. FA seems promising for dealing with combinatorial optimisation problem, but has been rarely reported. FA is a type of metaheuristic algorithm therefore quality of problem solutions depends on setting parameters in the algorithm. There is however no report on international scientific databases related to the investigation of the FA parameters' setting and its application on the JSSP. The popular Job shop scheduling problems that were used to test metaheuristic algorithm regularly are the datasets from OR-Library [11].

The objectives of this paper are to: i) presents the application of a recent developed metaheuristic called Firefly Algorithm for solving JSSP; ii) explore the parameters of the proposed algorithm; and iii) investigate the performance of the FA with different parameter setting and compare with best known solution from literature review. A job shop scheduling tool was written in modular style using Tcl/Tk programming language. The computational experiment was designed and conducted using five benchmarking datasets of the JSSP instance from the well-recognised OR-Library published by Beasley [11].

The remaining sections in this paper are organised as follows. Section 2 reviews the literature relating to job shop scheduling problems. Section 3 describes the procedures of the Firefly Algorithm (FA) and its pseudo code for solving the JSSP. Section 4 presents the experimental design and

analyses results. Finally, Section 5 draws the conclusions of the research and suggests possible further work.

2. Job shop scheduling problems (JSSP)

Shop floor scheduling problems can be classified into four main categories [12]: (i) single-machine scheduling (ii) flow-shop scheduling (iii) job-shop scheduling and (iv) open-shop scheduling. Single-machine scheduling is the simplest shop scheduling problem, in which there is only one available machine for servicing the arriving jobs. In flow-shop scheduling, jobs are processed on multiple machines in an identical sequence. Job-shop scheduling is a general case of flow-shop scheduling, in which the sequencing of each job through the machines is not necessarily identical. The open-shop scheduling is similar to the job-shop scheduling except that a job may be processed on the machines in any sequence the job needs. Since the job-shop scheduling is commonly found in many real-world businesses and/or manufacturing industry, this problem was proposed in this paper.

Although JSSP is common found but it is very difficult work to construct the best schedule within limited resources. The complexity of JSSP is increasing with the number of constraints defined and the size of search space operated. JSSP is known as one of the most difficult Non-deterministic Polynomial (NP) hard problems [13,14], in which the amount of computation required increases exponentially with problem size. The JSSP can be described as follow: a classical n-job m-machine (n×m) JSSP consists of a finite set $[J_j]_{1 \le j \le n}$ of n independent jobs or tasks that must be processed in a finite set $[M_k]_{1 \le k \le m}$ of m machines. The problem can be characterised as follows [15]: each job $j \in J$ must be processed by every machines $k \in M$; the processing of job J_j on machine M_k is called the operation O_{jk}; operation O_{jk} requires the exclusive use of machine M_k for an uninterrupted duration t_{jk}, its processing time; each job consists of a sequence of x_j operations; O_{jk} can be processed by only one machine k at a time (disjunctive constraint); each operation, which has started, runs to completion (non-preemption condition); and each machine performs operations one after another (resource/capacity constraint).

An example of two jobs to be performed three machines (2×3) job shop scheduling problem is illustrated in Table 1. In this problem, each job requires three operations to be processed on a pre-defined machine sequence. The first job (J_1) need to be initially operated on the machine M_1 for 5 time units and then sequentially processed on M_2 and M_3 for 4 and 9 time units, respectively. Likewise, the second job (J_2) has to be initially performed on M_3 for 5 time units and sequentially followed by M_1 and M_2 for 6 and 7 time units, respectively. The design task for solving JSSP is to search for the best schedule(s) for operating all pre-defined jobs in order to optimise either single or multiple scheduling objectives, which is used for identifying a goodness of schedule such as the minimisation of the makespan (C_{max}). Figure 1 shows another example on machine routings of a larger size of (3×6) job shop scheduling problem.

Table 1 An example of 2-jobs 3-machines scheduling problem with processing times.

Job (J_j)	Operation (O_{jk})	Time (t_{jk})	Machine (M_k)		
			M_1	M_2	M_3
J_1	O_{11}	5	5	-	-
	O_{12}	4	-	4	-
	O_{13}	9	-	-	9
J_2	O_{23}	5	-	-	5
	O_{21}	6	6	-	-
	O_{22}	7	-	7	-

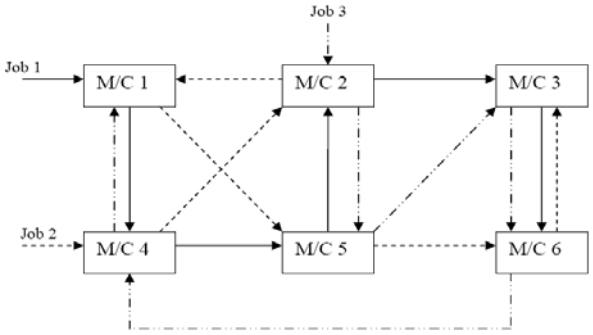

Fig. 1. Another example of the machine routings for 3-jobs 6-machines scheduling problem.

There has been a number of research works focused on the JSSP reported in the literature. The recent published research works related to JSSP by classifying those articles into four categories [16]: heuristic rules (for example, dispatching heuristics/priority rule [17,18]), mathematical programming techniques (e.g. branch and bound method, Lagrangian relaxation based approaches, queuing network model and etc. [3-6]), neighbourhood search methods (for instances, Tabu Search, Simulated Annealing [7,8]), and artificial intelligence techniques (such as, expert/knowledge based systems, artificial neural network, fuzzy logic, petri net based approaches [19-22]).

3. Firefly Algorithm for Solving Job Shop Scheduling Problems

Firefly Algorithm (FA) is a nature inspired algorithms, which is based on the flashing light of fireflies. The flashing light helps fireflies for finding mates, attracting their potential prey and protecting themselves from their predators. The swarm of fireflies will move to brighter and more attractive locations by the flashing light intensity that associated with the objective function of problem considered in order to obtain efficient optimal solutions.

The development of firefly-inspired algorithm was based on three idealised rules [9]: i) artificial fireflies are unisex so that sex is not an issue for attraction; ii) attractiveness is proportional to their flashing brightness which decreases as the distance from the other firefly increases due to the fact that the air absorbs light. Since the most attractive firefly is the brightest one, to which it convinces neighbours moving toward. In case of no brighter one, it freely moves any direction; and iii) the brightness of the flashing light can be considered as objective function to be optimised.

The main steps of the FA start from initialising a swarm of fireflies, each of which is determined the flashing light intensity. During the loop of pairwise comparison of light intensity, the firefly with lower light intensity will move toward the higher one. The moving distance depends on the attractiveness. After moving, the new firefly is evaluated and updated for the light intensity. During pairwise comparison loop, the best-so-far solution is iteratively updated. The pairwise comparison process is repeated until termination criteria are satisfied. Finally, the best-so-far solution is visualised. The pseudo code of FA applied to solve the JSSP is shown in Fig. 2. The main processes are described in the following subsections.

3.1 Population initialisation

Figs. 3-5 illustrate a typical firefly representation for the scheduling with nine operations. All operations required to produce jobs are encoded using alphanumeric strings. Each encoded operation is randomly selected and sequenced until all operations are drawn in order to create a firefly, which represents a candidate solution. This random selection is repeated to generate a swarm of fireflies with the required size. The length of slots in a firefly is equal to the total number of operations to be performed. The size of the firefly population determines the number of candidate solutions or the amount of search in the solution space.

Objective function $f(x)$, $x = (x_1, ..., x_d)^T$
Generate initial population of fireflies x_i (i = 1, 2, ..., n)
Light intensity I_i at x_i is determined by $f(x_i)$
Define light absorption coefficient γ
While t < MaxGeneration (G)
  For i = 1 : n all n fireflies
    For j = 1 : i all n fireflies
      If ($I_j > I_i$), Move firefly i towards j in d-dimension;
        Attractiveness varies with distance r_{ij} via $exp[-\gamma r_{ij}]$
        Evaluate new solutions and update light intensity
      End if
    End for j
  End for i
  Rank the fireflies and find the current best
End while
Postprocess on the best-so-far results and visualisation

Fig. 2. The pseudo code of the FA procedure adopted from [9].

3.2 Firefly evaluation

The next stage is to measure the flashing light intensity of the firefly, which depends on the problem considered. In this work, the evaluation on the goodness of schedules is measured by the makespan, which can be calculated using equation (1), where C_k is completed time of job k.

$$\text{Minimises } C_{\max} = \max(C_1, C_2, \ldots, C_k) \quad (1)$$

3.3 Distance

The distance between any two fireflies i and j at x_i and x_j, respectively, can be defined as a Cartesian distance (r_{ij}) using equation (2), where $x_{i,k}$ is the k^{th} component of the spatial coordinate x_i of the i^{th} firefly and d is the number of dimensions [9, 23].

$$r_{ij} = \| x_i - x_j \| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2} \quad (2)$$

J1:1	J1:2	J1:3	J2:1	J2:2	J2:3	J3:1	J3:2	J3:3
O1	O2	O3	O4	O5	O6	O7	O8	O9
0.45	0.52	0.99	0.27	0.09	0.01	0.33	0.85	0.78

Fig. 3. Random interval 0-1 for each job operation.

J2:3	J2:2	J2:1	J3:1	J1:1	J1:2	J3:3	J3:2	J1:3
O6	O5	O4	O7	O1	O2	O9	O8	O3
0.01	0.09	0.27	0.33	0.45	0.52	0.78	0.85	0.99

Fig. 4. Sort random number and follow with job operation.

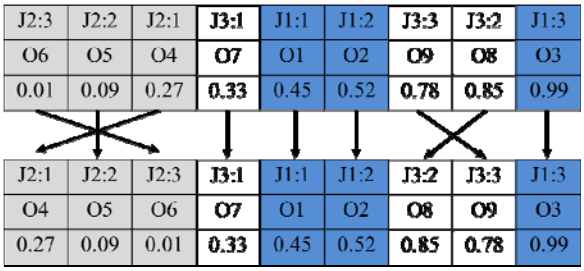

Fig. 5. Check and repair the sequence of each job.

3.4 Attractiveness

The calculation of attractiveness function of a firefly are shown in equations (3), where r is the distance between any two fireflies, β_0 is the initial attractiveness at r = 0, and γ is an absorption coefficient which controls the decrease of the light intensity [9, 23].

$$\beta_{(r)} = \beta_0 \times exp(-\gamma r^m), \text{ with } m \geq 1 \quad (3)$$

3.5 Movement

The movement of a firefly i which is attracted by a more attractive (i.e., brighter) firefly j is given by the following equation (4), where x_i is the current position or solution of a firefly, the $\beta_0 \times exp(-\gamma r_{ij}^2) \times (x_j - x_i)$ is attractiveness of a firefly to seen by adjacent fireflies. The α (rand – 1/2) is a firefly's random movement. The coefficient α is a randomisation parameter determined by the problem of interest with $\alpha \in$ [0-1], while rand is a random number obtained from the uniform distribution in the space [0,1] [9] [23].

$$x_i = x_i + \beta_0 \times exp(-\gamma r_{ij}^2) \times (x_j - x_i) + \alpha(rand - \frac{1}{2}) \quad (4)$$

Because the FA was recently developed, there have been a few research works applied the FA for solving optimisation problems, most of which has been formulated into mathematical equations. In the previous works, the settings of FA parameters, including the amount of fireflies (n), the number of generations (G), the light absorption coefficient (γ), the randomisation parameter (α) and the attractiveness value (β_0), have been defined in an ad hoc fashion. Table 2 summarises the FA parameter settings used in previous researches for solving various optimisation problems. Unfortunately, most of the work has not reported on the investigation of the appropriate setting of FA parameters via a proper statistical design and analysis. The computational experiments described in the next section were therefore proposed to identify the appropriate setting of FA parameters for solving scheduling problem.

Table 2 Examples of FA parameters' setting used in previous researches.

Authors	Problems	FA parameters			
		nG	γ	α	β_o
Apostolopoulos and Vlachos [24]	Economic emissions load dispatch	12*50	1.0	0.2	1.0
Lukasik and Zak [10]	Continuous equation	40*250	1.0	0.01	1.0
Horng and Jiang [25]	Image vector quantisation	50*200	1.0	0.01	1.0

4. Experimental design and analysis

In this research, the computational experiments were sequentially designed into two steps: identify the appropriate setting of the FA parameters and compare the results obtained from FA using the best setting identified in the previous experiment with two other results: i) adopting different parameter settings used by other research; and ii) the Best Known Solutions (BKS) from the literature [26]. Due to the limitation of computational time and resources required for computational experiments, five benchmarking instances (see Table 3) of JSSP were selected from the OR-Library [11]. The selection of those instant problems was considered from the size of searching space determined by two parameters: n jobs to be performed on m machines (n × m dimensions). All computational runs were based on personal computers with Core 2 Duo 2.67 GHz CPU with 4 GB DDR2 RAM.

Table 3 Benchmarking JSSP instance datasets selected from the OR-Library [11].

Instances	Problem size (n × m)	Number of jobs (n)	Number of machines (m)
DS6×6 (FT06)	6 × 6	6	6
DS10×5 (LA05)	10 × 5	10	5
DS15×5 (LA10)	15 × 5	15	5
DS20×5 (LA15)	20 × 5	20	5
DS15×10 (LA21)	15 × 10	15	10

4.1 Identifying appropriate setting of FA parameters

The first experiment aimed to investigate the appropriate setting of the FA parameters via experimental design and statistical analysis. Due to several parameters and levels of FA, full factional experimental design requires high computational resources and time consuming because of large number of experimental runs for each replication. Therefore, one-third fractional factorial experimental design (3^{k-1}) [27] was adopted in this work. The FA factors and their levels in this work are summarised in Table 4. Those factors are the combination of the amount of fireflies (n) and the number of maximum generations (G): nG, the light

absorption coefficient (γ), the randomisation parameter (α), and the maximum attractiveness value (β_0). Generally, the combination factor (nG) determines the amount of search (candidate solutions) in the solution space conducted by the FA. This factor is directly related with the size of the problem considered. The high value of this combination usually increases the probability of getting the best solution but requires longer computational time and resources. In this work, the computational limitations are practically imposed; this combination (nG) was therefore fixed at 2,500 in order to accommodate the computational search within the time limit. The γ factor was varied from 0 to 10, while the range of remaining factors (both α and β_0) was set between 0 to 1 [9, 23].

Table 4 FA's parameters and their levels considered.

Factors	Levels	Uncoded Values		
		Low (-1)	Medium (0)	High (+1)
nG	3	25*100	50*50	100*25
γ	3	0.1	5	10
α	3	0	0.5	1
β_0	3	0	0.5	1

The DS15 × 10 problem instance was computationally experimented with ten replications by using different random seed numbers. The computational results obtained from 270 runs (3^{4-1}*10) were analysed using a general linear model form of analysis of variance (ANOVA). Table 5 shows an ANOVA table consisting of Source of Variation (Source), Degrees of Freedom (DF), Sum of Square (SS), Mean Square (MS), and F and P values. A factor with value of $P \leq 0.05$ was considered statistically significant with a 95% confidence interval.

Table 5 Analysis of variance (ANOVA) on the FA parameters.

Source	DF	SS	MS	F	P
nG	2	141619	70809	32.54	0.000
α	2	1393894	696947	320.29	0.000
β_0	2	141845	70923	32.59	0.000
γ	2	57165	28583	13.14	0.000
Seeds	9	32975	3664	1.68	0.093
Error	252	548355	2176		
Total	269	2315854			

From Table 5, it can be seen that all FA parameters including nG, α, β_0, and γ were statistically significant in terms of the main effect with a 95% confidence interval. The most influencing factor was the α factor because of the highest F value, followed by β_0, nG, and γ, respectively. The main effect plots are shown in Fig. 6, suggesting that the main factors including nG, α, β_0, and γ should be defined at 100*25, 0.5 or 1, 0.5 or 1, and 0.1, respectively. In practical, the appropriate parameters setting of the FA obtained from main effect plots should be selected based on minimum average makespan values as follows: nG, α, β_0, and γ parameters should be set at 100*25, 0.5, 1, and 0.1, respectively.

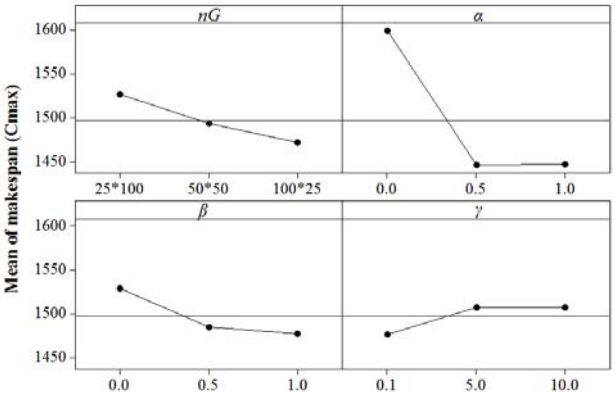

Fig. 6. The Main effect plot of FA.

4.2 Comparison on the FA performance with and without using optimised parameter setting

The aim of this experiment was to compare the results obtained from FA using the best setting identified in the previous experiment with two other results: i) adopting different parameter settings used by other researches; and ii) the Best Known Solutions (BKS) from the literature [26]. Five benchmarking instances of JSSP detailed in Table 3 were used to benchmark the FA performance in terms of minimise (Min), average (Avg), and standard deviation (SD) of the best-so-far solutions (makespan) obtained as shown in Table 6. Each parameter setting for each problem instance was repeated ten times using different random seed numbers. For a fair comparison, the amount of search (nG) in the solution space conducted by the FA search process must be similarly defined. In this case, the amount of search (nG) was fixed at 2,500 (25*100) solutions.

From the computation results shown in Table 6, the FA performance with optimised parameter setting can produced the operation schedule better than the FA with adopted other parameter setting in term of Min and Avg values of the makespan for all problems, but took longer computational time. The amount of search (nG) suggested from experimental design and analysis was 100*25, in which a large number of n must be calculated and updated the distance between any two fireflies more than a small number of n. Moreover, three of five instance problems related with small-medium sizes found the BKS, while the FA with the adopted parameters setting was found only for the DS10×5 instance. The Min and Avg values obtained from FA with optimised parameter setting on DS10×5 and DS15×5 instances were equal to the BKS. The SD of zero means that the proposed method can find the BKS in all computational runs having the use of different random seed numbers. The FA parameters setting adopted from Lukasik and Zak [10] and Horng and Jiang [25] out of condition to find the best solution. Because, those researchers define the randomisation parameter (α) less than the other settings, in which the most statistical influence parameter on this work should be set more than 0.5. Finally, it can be seen that the performance of the FA could be improved by using the appropriate parameter settings identified by adopting advance statistical tools.

5. Conclusion

Firefly Algorithm (FA) was applied to find the lowest makespan (C_{max}) of five benchmarking JSSP datasets adopted from the OR-Library. Experimental design and analysis were carried out to investigate the appropriate parameters setting of

Table 6 FA's results from different parameter settings.

Instances	Using parameter settings used by other research						Optimised parameters setting from the previous experiment			Best Known Solution
	Apostolopoulos and Vlachos [24]			Lukasik and Zak [10] Horng and Jiang [25]						
	Min	Avg	SD	Min	Avg	SD	Min	Avg	SD	
DS6×6	58	59.5	0.97	60	61.8	1.87	55	56.5	1.08	55
DS10×5	593	604.4	9.37	602	636.7	22.70	593	593	0.00	593
DS15×5	969	996	14.34	991	1040.6	32.02	958	958	0.00	958
DS20×5	1414	1491.5	36.27	1457	1537.8	43.02	1310	1366.9	32.32	1207
DS15×10	1435	1478.8	32.67	1511	1592.2	40.67	1323	1394.4	36.11	1046

the FA. The one-third fractional factorial experimental design can reduce the number of experimental runs by 66.67% compared with the conventional full factorial design. Ranges of FA parameters used by previous research were reviewed and investigated. The investigation was aimed to study the effect of the FA parameter setting on its performance before comparing the FA results between using and not using optimised parameter settings. In this research, the optimised setting of the FA parameters of nG, α, β_0, and γ parameters was suggested at 100*25, 0.5, 1, and 0.1, respectively. Moreover, the proposed algorithm with appropriate parameters setting produced the best-so-far schedule better than the FA without adopting parameter settings. It also found the best known solution in some cases. It should be noted that the appropriate parameter settings of the proposed algorithms may be case specific based on the nature and complexity of the problem domains.

6. References

[1] P. Pongcharoen, C. Hicks, P. M. Braiden and D. J. Stewardson, "Determining optimum Genetic Algorithm parameters for scheduling the manufacturing and assembly of complex products", International Journal of Production Economics, 78, 2002, pp. 311-322.

[2] P. Pongcharoen, C. Hicks and P. M. Braiden, "The development of genetic algorithms for the finite capacity scheduling of complex products, with multiple levels of product structure", European Journal of Operational Research, 152, 2004, pp. 215-225.

[3] C. Artigues and D. Feillet, "A branch and bound method for the job-shop problem with sequence dependent setup times", Annals of Operations Research, 159, 2007, pp. 135-159.

[4] C. Artigues, M.-J. Huguetand P. Lopez, "Generalized disjunctive constraint propagation for solving the job shop problem with time lags", Engineering Applications of Artificial Intelligence, 24, 2007, pp. 220-231.

[5] P. Baptiste, M. Flamini and F. Sourd, "Lagrangian bounds for just-in-time job-shop scheduling", Computers & Operations Research, 35, 2008, pp. 906-915.

[6] H. X. Chen and P. B. Luh, "An alternative framework to Lagrangian relaxation approach for job shop scheduling", European Journal of Operational Research, 149, 2003, pp. 499-512.

[7] H. Gröflin and A. Klinkert. "A new neighborhood and tabu search for the Blocking Job Shop", Discrete Applied Mathematics, 157, 2009, pp. 3643-3655.

[8] R. Zhang and C. Wu, "A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective", Computers & Operations Research, 38, 2011, pp. 854-867.

[9] X. S. Yang, "Firefly Algorithms for multimodal optimization", Lecture Notes in Computer Science, 5792, 2009,pp. 169-178.

[10] S. Lukasik and S. Zak, "Firefly Algorithm for continuous constrained optimization Tasks", Lecture Notes in Computer Science, 5796, 2009, pp. 97-106.

[11] J. E. Beasley, "OR-library: distributing test problems by electronic mail", Journal of the Operational Research Society, 41, 1990, pp. 1069-1072.

[12] X. J. Wang, C.-Y. Zhang, L. Gao and P.-G. Li, "A survey and future trend of study on multi-objective scheduling", Proceedings of the 4th International Conference on Natural Computation, 2008., pp. 382-391.

[13] G. Moslehi and M. Mahnam, "A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search", International Journal of Production Economics, 129, 2011, pp. 14-22.

[14] L. Asadzadeh and K. Zamanifar, "An agent-based parallel approach for the job shop scheduling problem with genetic algorithms", Mathematical and Computer Modelling, 52, 2010, pp. 1957-1965.

[15] K. Ripon, C. H. Tsang and S. Kwong, "An Evolutionary approach for solving the multi-objective job-shop scheduling problem", Studies in Computational Intelligence, 49, 2007, pp.165-195.

[16] A.K. Gupta and A. I. Sivakumar, "Job shop scheduling techniques in semiconductor manufacturing", International Journal of Advanced Manufacturing Technology, 27, 2006, pp. 1163-1169.

[17] O. Holthaus and C. Rajendran, "Efficient dispatching rules for scheduling in a job shop", International Journal of Production Economics, 48, 1997, pp. 87-105.

[18] R. M. Dabbas, H.-N. Chen, J. W. Fowler and D. Shunk, "A combined dispatching criteria approach to scheduling semiconductor manufacturing systems", Computers & Industrial Engineering, 39, 2001, pp. 307-324.

[19] S. Yang, D. Wang, T. Chai and G. Kendall. "An improved constraint satisfaction adaptive neural network for job-shop scheduling", Journal of Scheduling, 13, 2010, pp. 17-38.

[20] S. X. Yang and D. W. Wang, "A new adaptive neural network and heuristics hybrid approach for job-shop scheduling", Computers & Operations Research, 28, 2001, pp. 955-971.

[21] O. Bilkay, O. Anlagan and S. E. Kilic, "Job shop scheduling using fuzzy logic", International Journal of Advanced Manufacturing Technology, 23, 2004, pp. 606-619.

[22] M. Nakamura, K. Hachiman, H. Tohme, T. Okazaki and S. Tamaki, "Evolutionary computing of Petri Net structure for cyclic job shop scheduling", IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, E89-A, 2006, pp. 3235-3243.

[23] X.S. Yang. "Nature-Inspired Metaheuristic Algorithms", Luniver Press, 2008.

[24] T. Apostolopoulos and A. Vlachos, “Application of the Firefly Algorithm for solving the economic emissions load dispatch problem", International Journal of Combinatorics, 20, 2011.

[25] M. H. Horng and T. W. Jiang, "The codebook design of image vector quantization based on the firefly algorithm", Lecture Notes in Computer Science, 6423 LNAI (PART 3), 2010, pp. 438-447.

[26] 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, pp. 2629-2636.

[27] D. C. Montgomery, 2001. "Design and Analysis of Experiments, 5 ed". New York: John Wiley & Sons, 2001.