doi.org/10.15407/usim.2019.05.038
UDK 658.5

T. WITKOWSKI, D.Sc., Prof., Faculty of Production Engineering ,
Warsaw University of Technology,
Pl. Politechniki 1, 00-661, Warsaw, Poland,
tadeusz.witkowski@pw.edu.pl

THE DABC AND TLBO ALGORITHMS FOR SOLVE JOB SHOP SCHEDULING PROBLEM

This paper shows use Discrete Artificial Bee Colony (DABC) and Teaching-Learning-Based Optimization (TLBO) algorithms for solving the job shop scheduling problem (JSSP) in order to minimize makespan (C_{max} value). The Job Shop Scheduling Problem is one of the most difficult problems as it is classified as an NP-hard one. Stochastic search techniques, such as evolutionary algorithms, are used to find a good solution. Our objective is to estimate efficiency of DABC and TLBO algorithms on many tests of JSSP problems.

Keywords: *Discrete Artificial Bee Colony algorithm; Teaching-Learning-Based Optimization; job shop scheduling problem; makespan.*

1. Introduction

The job shop scheduling problem has become a classic scheduling problem and it is mainly related to industrial engineering, though it is also required in other branches. The study of scheduling problem is carried out taking inputs from various streams like computer science, operations research, management and manufacturing. These problems belong to the family of NP-hard, in which they cannot be solved in polynomial time.

Scheduling is assigning a set of tasks on resources in a time period, taking into account the time, capability and capacity constraints. Many studies have been done to solve this problem or to determine the nearest approach to the solution. Commonly used scheduling techniques include the following [1] :

- Exact Algorithms (e.g. Branch and Bounds Methods, Linear Programming, Dynamic Programming);
- Approximation Algorithms [(Artificial Intelligence Algorithms (e.g. artificial neural network), Local Search Algorithms (e.g. greedy randomized adaptive search procedure), Evolutionary Algorithms (e.g. genetic algorithm), Swarm Optimization Algorithms (e.g. bee colony algorithm)].

Hence, a variety of heuristics and metaheuristics procedures such as taboo search, simulated annealing and genetic algorithm have been applied to solve these problems and find optimal or near optimal schedule in a reasonable time [2].

In this paper, the discrete artificial bee colony algorithm (DABC) and teaching-learning-based optimization (TLBO) algorithm are proposed for solving the job shop scheduling problem with the aim of minimizing makespan (C_{max}).

2. Job Shop Scheduling Problem

The task of production scheduling consists in the temporal planning of the processing of a given set of orders. The processing of an order corresponds to the production of a particular product. It is accomplished by the execution of a set of operations in a predefined sequence on certain resources subject to several constraints. The result

of scheduling is a schedule showing the temporal assignment of operations of orders to the resources to be used.

Each operation can be performed by some machines with different processing times. The difficulty is to find a good assignment of an operation to a machine in order to obtain a schedule which minimizes the total elapsed time C_{max}. The structure of the scheduling problem can be described as follows [2].

Consider a set of N jobs $\{J_i\}$, $1 \le j \le N$, and a set of machines $\{M_k\}$, $1 \le k \le M$, where M is the total number of machines existing in the shop. All jobs are independent of one another.

Each job J_j has an operating sequence, called G_j; each operating sequence G_j is an ordered series of operations O_{ij}, indicating the position of the operation in the technological sequence of the job; the realization of each operation O_{ij} requires a resource, i.e. a machine selected from a set of machines, $\{M_k\}$ (for FJSSP problem).

This implying the existence of an assignment problem: there is a predefined set of processing times; for a given machine M_k, and a given operation O_{ij}, the processing time is denoted by $P_{ij\,k}$; an operation which has started runs to completion (non-preemption condition); each machine can perform operations one after another (resource constraints).

The time required to complete the whole job is C_{max}.

Our objective is to determine the set of completion times for each operation which minimizes C_{max}.

The flexible job shop scheduling problem (FJSSP) is an extended traditional JSSP problem. It breakthroughs the restriction of unique resources and allows each operation to be processed by several different machines, thus making the FJSSP problem to correspond with actual production situation more accurately.

3. Metaheuristics for Solving JSSP Problem

In this study, a recently developed Discrete Artificial Bee Colony (DABC) algorithm and Teaching-Learning-Based Optimization (TLBO) method are proposed to solve the job shop scheduling problems to minimize the makespan.

3.1. *DABC Algorithm*

The classical artificial bee colony (ABC) algorithm proposed by Dervis Karaboga [3] is one of population algorithm often used for constrained optimization problems [4-7]. The considered problem is reformulated so as to take the form of optimizing two functions, the objective function and the constraint violation function [3]. Detailed pseudo-code of the ABC algorithm is given below [8, 9].

1: Initialize the population of solutions X_i:

$$X_i = \{x_{i1}, x_{i2}, ..., x_{ij}, ..., x_{iDN}\}, \quad (1)$$

$$x_{ij} = lb_j + \text{rand}(0,1)\cdot(ub_j - lb_j), \quad (2)$$

where: $i = 1, ..., SN$ (source number), $j = 1, ..., DN$ (dimension number), lb — lower boundary, ub — upper boundary;

2: Evaluate the population;

3: cycle=1;

4: repeat;

5: Produce new solutions v_{ij} for the employed bees by using and evaluate them, according to (2.2) in [2]:

$$v_{ij} = x_{ij} + \text{rand}(0,1)\cdot(x_{ij} - x_{kj}), \quad (3)$$

where $k \in \{1, 2,..., BN\}$ and $j \in \{1, 2,..., DN\}$ are randomly chosen indexes, $k \neq i$;

6: Apply the greedy selection process [10];

7: Calculate the probability values p_i for the solutions x_{ij} with formula (2.1) in [3]:

$$p_i = \frac{\text{fit}_i}{\sum_{i=1}^{SN} \text{fit}_i}, \quad (4)$$

where: fit_i — fitness function value of i-th solution (food source);

8: Produce the new solutions v_{ij} for the onlookers from the x_{ij} solutions elected depending on p_i and evaluate them;

9: Apply the greedy selection process [10];

10: Determine the abandoned solution for the scout, if exists, and replace it with a new randomly produced solution x_{ij} according to the equation 2;

11: Memorize the best solution achieved so far;

12: cycle = cycle + 1;

13: go step 4 until cycle = MCN, (MCN — maximum cycle number).

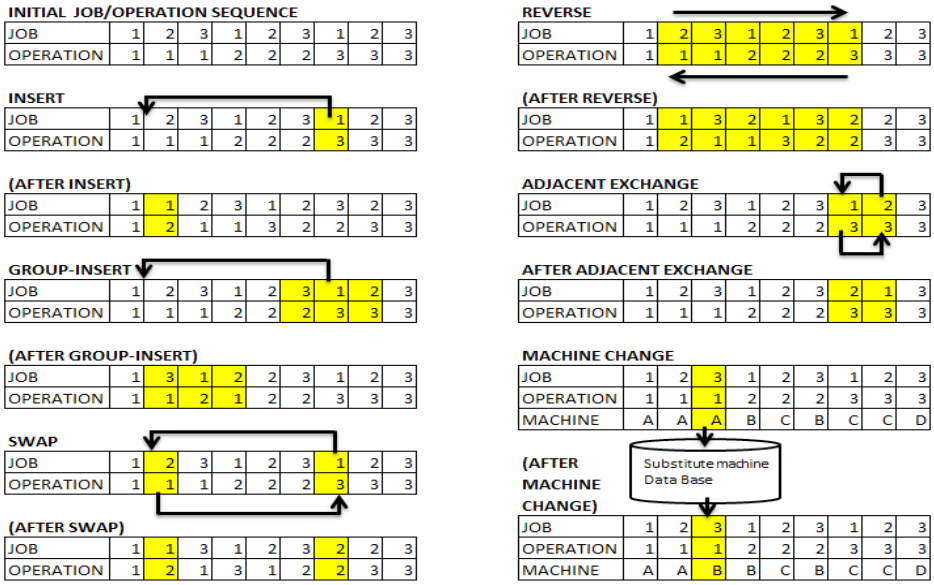

Fig. 1. Mutation strategies for creating new neighboring solution by DABC algorithm

As the basic ABC algorithm was originally designed for continuous function optimization, in order to make it applicable for solving the problem, a discrete version of the ABC algorithm (DABC) is considered. Composite mutation strategies are proposed to enable the DABC to explore the new search space and solve the permutation flow shop scheduling problem. We consider each discrete job permutation as a food source and apply discrete operations to generate a new neighborhood food source (FS) for different bees to make artificial bee colony algorithm suitable for JSP. Each FS is a permutation of operations.

In order to generate good diversity neighboring solutions several mutation strategies are proposed to enable the DABC to solve the JSSP problem (Figure 1). We use INSERT and SWAP, REVERSE (INVERSE) and ADJACENT EXCHANGE mutations which are commonly used in DABC [10, 11] and use two additional strategies: GROUP-INSERT, which acts like INSERT mutation with possibility to insert group of adjacent operations and MACHINE CHANGE mutation for FJSSP.

These operations based on [9, 11] are given in [12] and are illustrated on the Fig. 1.

The control parameters of proposed Discrete Artificial Bee Colony are: *SN* — source number, *MITN* — maximum improvement trial number and one typed by a user termination criteria: *MCN* — maximum cycle number, computation time limit or optimal makespan deviation in percentage. Except of control parameters, input data that describes job scheduling problem is loaded into program: manufacturing program, operation routing matrix, group-tech machine routing matrix (for FJSSP), candidate machine list (for FJSSP), unit processing times matrix and re-tooling times matrix. Initial population consists of schedule generated based on a random permutation of operations (priority rule).

Whenever during employed or onlooker bee phase a new food source (schedule) is produced, for each bee new neighboring solutions are generated using proposed mutations strategies (Figure 1), then new solutions are evaluated, the fitness function is calculated — the smaller makespan, the higher value of the fitness function. Then the original food source (schedule) and a new schedules (from neighborhood) participate in the greedy selection process. One local winner with the highest fitness function value is chosen for each bee. Whenever scout bee occurs, a new schedule is generated based on a random priority rule for food source with MITN criteria met. Then the

best global solution for whole population is memorized, number of cycles increases by one, and termination criteria value is updated. If the termination criteria is met, the algorithm stops and the best found solution is saved and printed.

3.2. *TLBO Algorithm*

Effective optimization method, based on the processes of teaching and learning was proposed by Rao and Kalyankar for optimization the engineering design problems [13]. Teaching-learning-based optimization (TLBO) population-based metaheuristic search algorithm is inspired by the teaching and learning process in a classroom.

This method works based on effect action of teacher on students. TLBO relies on generating initial population of solutions, which is used to move to the global solution. The population is treated as a group of students. The whole process is divided into two main phases: the transfer of knowledge by the teacher (teaching phase) and the phase of the exchange of knowledge among students through interaction with each other (learning phase). Detailed pseudo-code of the TLBO algorithm is given below:

1: Initialize the population of solutions X_i:

$$X_i = \{x_{i1}, x_{i2}, ..., x_{ij}, ..., x_{iDN}\}, \quad (5)$$

$$x_{ij} = lb_j + \text{rand}(0,1) \cdot (ub_j - lb_j), \quad (6)$$

where: $i = 1, ..., P$ (population size), $j = 1, ..., DN$ (dimension number), lb — lower boundary, ub — upper boundary;

2: Evaluate the population;

3: cycle=1;

4: repeat;

5: Find the best solution which will be defined as teacher;

6: Teaching phase:

produce new solutions X_{new} for all P students in group by using and evaluate them:

$$X_{new,j} = X_{old,j} + \text{rand}(0,1) \cdot (X_{teacher,j} - T_F \cdot x_{k,j}), \quad (7)$$

where: T_F — teaching factor,

T_F = round (1 + rand (0,1));

7: Apply the greedy selection process [10];

8: Learning phase: select two random different students X_i and X_j, where $i \neq j$:

$$X_{new,j} = X_{old,j} + \text{rand}(0,1) \cdot \text{abs}(X_i - X_j); \quad (8)$$

9: Apply the greedy selection process [10, 12];

10: cycle=cycle+1;

11: go step 4 until cycle=*MCN*, (*MCN* — maximum cycle number)

In [13] were designed some effective metaheuristics to improve TLBO algorithm for effective optimization of more complex problems. Our TLBO was improved in some of them.

Some different strategies are utilized in a hybrid way to generate population with certain quality and diversity. For the machine assignments, the following three rules are applied to generate the initial assignments.

Rule-1: Random rule. Randomly select a machine from the candidate machines set for each operation, and then place it at the position in the machine assignment vector.

Rule-2: Global minimum processing time rule [14]

Rule-3: Local minimum processing time rule [8]. In our TLBO, 20% of individuals are generated by rule-1, 10% are generated by rule-2 and rest 60% by rule-3.

As for the operation sequence, the following three rules are applied to initialize the population.

Rule-4: Random rule. Randomly generate the sequence of the operations on each machine.

Rule-5: Most time remaining rule [8]. Sequence the jobs in the order of non-increasing remaining time, that is, the job with the most remaining time will be selected first.

Rule-6: Minimum time remaining rule [8]. Sequence the jobs in the order of increasing remaining time, that is, the job with the shortest remaining time will be selected first.

Rule-7: Most number of operations remaining rule [8]. The job with largest remaining operations unprocessed has a high priority to be selected.

In our TLBO, 10% individuals are generated by rule-4, other individuals generated 30% for each rule (rule-5, rule-6, rule-7).

In order to generate good diversity neighboring

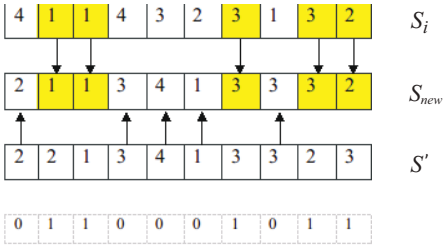

Fig. 2. Procedure of uniform crossover for machine assignment by TLBO [15]

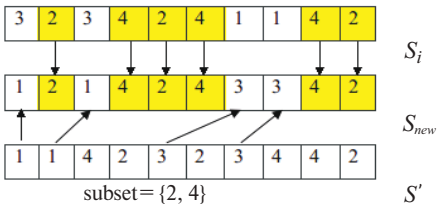

Fig. 3. Procedure of MPOX crossover for operation sequence by TLBO [16]

solutions different exploitation search procedures to evolve the machine assignment and operation sequence respectively are used.

For the operation sequence, a modified precedence operation crossover (MPOX) is used [16].

In learning phase an additional parameter α (between 0% and 100%) is introduced. In each generation, all the individuals rank from the best to the worst and top $P \cdot \alpha$ individuals which are defined the best students; and only this part of all students will take part in the learning phase. This imitates that only smart students improve themselves by interaction. Also, the number of students is not limited to one student.

Modified TLBO used extra phase local search which imitates the process that the teacher improves himself by enhancing his own knowledge. The local search is performed on the teacher for tl times. Greedy selection is used to update the teacher. Two local search operators are developed for the solution, namely local search for machine assignment and local search for operation sequence.

Local search for machine assignment S_i:

1: Generate an integer $i = \text{rand}(1, Z)$, where Z is the total number of operations;

2: Randomly select i positions from the machine assignment vector of S_i;

3: For each selected position, replace the machine with a different machine that is randomly chosen from the candidate machine set to generate a new solution S_{new}. Then evaluate it.

Local search for operation sequence S_i:

1: Randomly select two jobs J_1 and J_2, and record the positions of the two jobs;

2: Fill J_1 into the positions of J_2 from left to right, and fill J_2 into the positions of J_1 to generate a new solution S_{new}. Evaluate the solution.

The control parameters of proposed TLBO are: P — size of the population, α — percent of the best students (solutions), tl — times of local search (maximum cycle number, computation time limit), MCN — maximum cycle number, computation time limit or optimal makespan deviation in percentage, L — times of interactions between students.

Modified TLBO algorithm can be used not only for JSSP problem but for FJSSP too.

4. Experiment Results

This section describes the computational experiments to evaluate the performance of the proposed algorithms. The proposed algorithms are tested on 10 job shop scheduling problems and 10 times independently. The experiments performed on PC with processor Intel® Core™ i7-3770 CPU @ 3.40GHz and RAM: 16GB. The DABC algorithm was coded in Java and the TLBO algorithm was coded in the GNU Octave, which could have a little influence on the differences in recalculation time of the algorithms.

Computational experiment 1
with DABC — fixed control parameters

In this section theoretical JSSP problems by DABC with $SN = 1000$, $MITN = 750$, $MCN = 3000$ are investigated. The proposed algorithm is tested on 10 job shop scheduling bench mark problems, outcomes are given in Table 1 [12].

Computational experiment 2
with DABC — adjusted control parameters

In the experiment, control parameters values were adjusted for each problem accordingly to formulas give in [5], but "number of operations" is used instead of fixed value "10":

$$SN = 5 \cdot n, \quad (9)$$

$$MCN = n \cdot m \cdot o, \quad (10)$$

$$MITN = 2 \cdot n \cdot m, \quad (11)$$

where: n — number of jobs, m — number of machines, o — number of operations for a job. The proposed algorithm is tested on 10 job shop scheduling bench mark problems, outcomes are given in Table 2 [12].

Table 1. **Theoretical JSSPs solved by DABC, fixed parameters**

Problem	Performance of DABC algorithm (C_{max} value)			
	Optimal makespan	Best found makespan	Deviation from the optimal makespan, %	Computation time [s]
FT6	55	55	0	127
FT10	930	967	4	260
FT20	1165	1216	4	220
La02	655	655	0	151
La19	842	863	2	233
La21	1046	1102	5	383
La27	1235	1318	7	481
La30	1355	1404	4	475
La40	1222	1345	10	532
SWV11	2983	3844	29	1074

Table 2. **Theoretical JSSPs solved by DABC, adjusted parameters**

Problem	Performance of DABC algorithm (C_{max} value)			
	Optimal makespan	Best found makespan	Deviation from the optimal makespan, %	Computation time [s]
FT6	55	55	0	0,4
FT10	930	1005	8	4,9
FT20	1165	1275	10	4,8
La02	655	676	3	0,8
La19	842	885	5	4,9
La21	1046	1152	10	13,4
La27	1235	1402	14	30
La30	1355	1404	9	30
La40	1222	1367	12	53
SWV11	2983	3944	32	434

*Average values of best found makespan — 10 runs for each problem

Computational experiment 1
with TLBO — fixed control parameters

In this section theoretical FSSP problems by TLBO with P = 400, α = 15%, tl = 15 times, L = 20 and MCN = 10 times are investigated. The proposed algorithm is tested on 10 job shop scheduling bench mark problems, outcomes are given in Table 3 [12].

Computational experiment 2
with TLBO — adjusted control parameters

Control parameters values were adjusted for each problem depend on the size of scheduling problem: number of jobs n and number of machines m, only parameter of the best students α is used instead of fixed value 15%.

$$P = 5 \cdot n; \quad (12)$$

$$tl = n + m; \quad (13)$$

$$MCN = n + m; \quad (14)$$

$$L = n + m. \quad (15)$$

The proposed algorithm is tested on 10 job shop scheduling benchmark problems, outcomes are given in Table 4 [12].

Table 3. **Theoretical JSSPs solved by TLBO, fixed parameters**

Problem	Performance of DABC algorithm (C_{max} value)			
	Optimal	Average best found makespan*	Best found makespan	Average computation time* [s]
FT6	55	55	55	15
FT10	930	974	948	751,5
FT20	1165	1240	1204	769,5
La02	655	671	655	260,3
La19	842	879	863	779
La21	1046	1124	1092	1630,4
La27	1235	1346	11315	2211
La30	1355	1409	1291	2818,8
La40	1222	1362	1326	3402,6
SWV11	2983	3698	3577	15306,6

*Average values of best found makespan — 10 runs for each problem

Results comparison for JSSP test problems

In [17] comparison of SPT, FIFO, and GA taking into account makespan values and computational time are included. Only makespan values are compared in this work because of hardware differences. The results comparison is given in Table 5 (C_{max} values) and Table 6 (optimal solution deviation).

Table 4. **Theoretical JSSPs solved by TLBO, adjusted parameters**

Problem	Performance of TLBO algorithm (C_{max} value)			
	Optimal makespan	Average best found makespan*	Best found makespan	Average computation time* [s]
FT6	55	55	55	8
FT10	930	1022	980	151
FT20	1165	1240	1225	379,6
La02	655	699	686	30,8
La19	842	905	894	144,5
La21	1046	1152	1145	746,5
La27	1235	1387	1373	1744
La30	1355	1422	1411	2110,5
La40	1222	1362	1326	3402,6
SWV11	2983	3493	3472	36104,2

*Average values of best found makespan — 10 runs for each problem

Table 5. **Theoretical JSSPs results comparison — makespan values [12]**

Problem	Deviation from the optimal makespan				
	GA [18]	TLBO –1 [12]	TLBO –2 [12]	DABC –1 [12]	DABC –2 [12]
FT6	57	55	55	55	55
FT10	974	953	980	967	1005
FT20	1198	1204	1225	1216	1275
La02	668	667	686	655	675
La19	876	873	984	863	885
La21	1098	1110	1145	1102	1152
La27	1350	1341	1373	1318	1402
La30	1362	1398	1411	1404	1461
La40	1289	1326	1326	1345	1367
SWV11	3330	3577	3472	3844	3944

Computational experiment with TLBO, DABC, ANN and GA for a real production system

The input data are the matrix of the groups of technologically interchangeable machines, the matrix of technological routes, the matrix of operations with an accuracy of group of technologically interchangeable machines, the matrix of the processing times t_{ij} i-th of an operation, the matrix of the processing times of a setup of machines before proceeding j-th operation and i-th part. The date set contains 10 parts which need to be processed by 27 machines and 160 operations. The objective is to minimize C_{max} value for FJSSP problem (with serial type production flow).

Analyzing the effectiveness of the algorithms is a difficult task. For example, in table 7 we can see that for FJSSP problem (with serial production flow) ANN [19], GRASP [20], TLBO [12] and

Table 6. **Theoretical JSSPs results comparison — makespan values [12]**

Problem	Deviation from the optimal makespan, %				
	GA	TLBO –1	TLBO –2	DABC –1	DABC –2
FT6	4	0	0	0	0
FT10	5	2	5	4	6
FT20	3	3	5	4	9
La02	2	2	5	0	3
La19	4	4	6	2	5
La21	5	6	9	5	10
La27	9	9	11	7	14
La30	1	3	4	4	8
La40	5	9	9	10	12
SWV11	12	20	16	29	32

Table 7. Values of makespan for a real production system

Problem	Algorithms				
	GA [18]	ANN [19]	GRASP [20]	DABC [12]	TLBO [12]
FT6	57636	50242,4*	50242,2*	50242,2*	50242,2*

*Average values of best found makespan — 10 runs for each problem

DABC [12] algorithms gives better C_{max} values than the genetic algorithm [18]. But GA [18] was not as thoroughly tested as was done in case of GRASP, DABC and TLBO.

Conclusion

This work examined the JSSP and FJSSP problems. Computational experiments for JSSP test problems shows that results given by the DABC with current control parameters values and TLBO are close to optimum results GA algorithm [18] for FT6, ..., La27 problems (table 5 and 6). The TLBO with appropriate parameters setting that were achieved from the experimental analysis produced the best-so-far schedule better than TLBO without adopting parameters settings. Further experiments with DABC and TLBO algorithms for various termination criteria and different control parameters , shows TLBO demonstrated better C_{max} values then DABC and worse then GA [18] for La and SWV11 problems. Computational experiment shows that results given by the DABC and TLBO for flexible job shop scheduling problems (real production systems) generated optimal C_{max} values.

REFERENCES

1. Mesghouni, K., Hammadi, S., Borne, P., 2004. "Evolutionary Algorithms for Job Shop Scheduling". J. Appl. Math. Comput. Sci., Vol. 14, No. 1, pp. 91–103.
2. Blazewicz, J., et al., 2007. Handbook on Scheduling; From Theory to Application, Springer, Berlin, Heidelberg, New York, 280 p.
3. Karaboga, D., Basturk, B., 2007. "Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems", in: P. Melin et al. (eds.), IFSA 2007, LNAI 4529, Springer-Verlag Berlin Heidelberg. pp. 789–798.
4. Li, J., Pan, Q., Gao, K., 2011. "Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems", Int. J. Advance Manufacturing Technology, Vol. 55, No 9-12, pp. 1159-1169.
5. Li, J. Pan, Q. Xie, S. Wang, S., 2011. "A Hybrid Artificial Bee Colony Algorithm for Flexible Job Shop Scheduling Problems", Int. J. of Computers, Communications & Control, Vol. 6, No 2, pp. 286-296.
6. Onder, E., Ozdemir, M., Yildirim, B.F., 2013. "Combinatorial Optimization Using Artificial Bee Colony Algorithm And Particle Swarm Optimization Supported Genetic Algorithm", Kafkas University Journal of Economics and Administrative Sciences Faculty, Vol. 4, Issue 6, pp. 59-70.
7. Yurtkuran, A., Emel, E., 2016. "An Enhanced Artificial Bee Colony Algorithm with Solution Acceptance Rule and Probabilistic Metasearch", Computational Intelligence and Neuroscience, Vol. 2016, Article ID 8085953, 13 p.
8. Lei, D., 2010. "A genetic algorithm for flexible job shop scheduling problem with fuzzy processing time", Int. J. Prod. Res., Vol. 48, No 10, pp. 2995–3013.
9. Li, X., Yin, M., 2012. "A discrete artificial bee colony algorithm with composite mutation strategies for permutation flow shop scheduling problem", Scientia Iranica, Vol. 19, No 6, pp. 1921–1935.
10. Greedy algorithm, in: Paul E. Black, Dictionary of Algorithms and Data Structures, National Institute of Standards and Technology, US.
11. Rao, R.V., Savsani, V.J., Vakharia, D.P., 2011. "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems", Computer-Aided Design, Vol. 43, Issue 3, pp. 303-315.
12. Witkowski, T., Krzyzanowski, P., Vasylishyna, S., 2016. "Comparison of DABC and TLBO Metaheuristics for Solve Job Shop Scheduling Problem", 12 International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD 2016), 13-15 August, Changsha.
13. Rao, R.V., Kalyankar, V.D., 2013. "Parameter optimization of modern machining processes using teaching–learning-based optimization algorithm", Engineering Applications of Artificial Intelligence, Vol. 26, Issue 1, pp. 524–531.
14. Pezzella, F., Morganti, G., Ciaschetti, G., 2008. "A genetic algorithm for the flexible job-shop scheduling problem", Comput. Oper. Res., Vol.35, Issue 10, pp. 3202–3212.
15. Xu, Y., Wang, L., Wang, S.Y., Liu, M., 2015. "An effective teaching-learning based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time", Neurocomputing, Vol.148, pp. 260-268.
16. Zhang, G., Shao, K., Li, P.,Gao, L., 2009. "An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem", Computers & Industrial Engineering, Vol. 56(4), pp. 1309-1318.

17. Kloud, T., Koblasa, F., 2011. “Solving job shop scheduling with the computer simulation”, The International Journal of Transport & Logistics, 9th Special Issue, pp. 775–785.
18. Witkowski, T., Strojny, G., Antczak, P., 2007. “The Application Of Neural Networks For Flexible Job Shop Problem”. Int. J. of Factory Automation, Robotics, and Soft Computing, Int. Society for Advanced Research, Palermo, Italy, Issue 2, pp. 116-121.
19. Witkowski, T., Antczak, A., Antczak, P., 2010. “Comparison of Optimality and Robustness between SA, TS and GRASP Metaheuristics in FJSP Problem”, Lecture Notes in Computer Science, Springer-Verlag Berlin, Vol. 6215, pp. 319-328.
20. Witkowski, T., Elzway, S., Antczak, A., Antczak, P., 2007. “Representation of Solutions and Genetic Operators for Flexible Job Shop Problem”, Communications in Computer and Information Science, Advanced Intelligent Computing Theories and Applications. Springer-Verlag, Berlin Heidelberg , Vol. 2, pp. 256-265.

Тадеуш Вітковський, доктор техн. наук, професор,
Варшавська політехніка,
Площа Політехніки, 1, 00-661,
Варшава, Польща
tadeusz.witkowski@pw.edu.pl

ЗАСТОСУВАННЯ АЛГОРИТМІВ DABC ТА TLBO ДО ЗАДАЧІ ПЛАНУВАННЯ РОБОТИ ЦЕХУ

Вступ. Задача (календарного) планування роботи цеху (ЗПРЦ, *Job Shop Scheduling Problem — JSSP*) є класичною задачею теорії розкладів. Вона пов’язана, головним чином, з промисловим виробництвом, хоча знаходить застосування і в інших галузях. Теорія розкладів знаходиться на перехресті таких дисциплін, як інформатика, дослідження операцій, управління та виробництво. Задачі теорії розкладів у загальному випадку є *NP*-повними, тобто не існує методу отримання їх розв’язку за поліноміальний час.

JSSP оптимально призначає кожній технологічній операції ресурс та початок часу виконання, щоб мінімізувати загальну тривалість виконання. Для вирішення цієї проблеми (або для визначення найкращого підходу) проведено багато досліджень.

У цьому дослідженні до *JSSP* застосовано алгоритм дискретної штучної бджолиної колонії (ДШБК, *Discrete Artificial Bee Colony — DABC*) та метод оптимізації на основі викладання/навчання (ООВН, *Teaching-Learning-Based Optimization — TLBO*).

Мета дослідження — оцінка ефективності алгоритму *DABC* та методу *TLBO* на багатьох тестах задачі планування роботи цеху.

Методи. Для пошуку ефективного рішення використовуються стохастичні методи пошуку, такі як еволюційні алгоритми, з якими порівнюються методи дискретної штучної бджолиної колонії та оптимізації на основі викладання/навчання.

Результати. Показано використання алгоритмів дискретної штучної бджолиної колонії та методу оптимізації на основі викладання/навчання для отримання розв'язку задачі календарного планування з метою мінімізації часу виконання (значення C_{max}).

Висновки. Проведено порівняння методу дискретної штучної бджолиної колонії та оптимізації на основі викладання/навчання. Обчислювальні тестові експерименти показують, що результати, отримані за *DABC* із значеннями поточних параметрів управління та *TLBO*, близькі до оптимальних результатів відомого генетичного алгоритму.

Експерименти з алгоритмами *DABC* та *TLBO* для різних параметрів та критеріїв зупинки показують, що *TLBO* продемонстрував кращі значення C_{max}, ніж *DABC*, і гірші, ніж генетичний алгоритм. Обчислювальний експеримент показує, що результати, отримані *DABC* та *TLBO* для задач планування потокової лінії (реальні виробничі системи), дали оптимальні значення C_{max}.

Ключові слова: *алгоритм дискретної штучної бджолиної колонії; оптимізація на основі викладання/навчання; задача календарного планування роботи цеху; теорія розкладу; час виконання.*

Тадеуш Витковски, доктор техн. наук, профессор,
Варшавская политехника,
Пл. Политехники, 1, Варшава, Польша,
tadeusz.witkowski@pw.edu.pl

ИСПОЛЬЗОВАНИЕ АЛГОРИТМОВ DABC И TLBO В ЗАДАЧЕ ПЛАНИРОВАНИЯ РАБОТЫ ЦЕХА

Введение. Задача (календарного) планирования работы цеха (ЗПРЦ, *Job Shop Scheduling Problem — JSSP*) — классическая задача теории расписаний. Она связана, главным образом, с промышленным производством, но находит применение и в других отраслях. Теория расписаний находится на перекрестке таких дисциплин, как информатика, исследование операций, управление и производство. Задачи теории расписаний в общем случае являются *NP*-полными, т.е. не существует метода получения их решения за полиномиальное время.

JSSP оптимально назначает каждой технологической операции ресурс и начало времени выполнения, чтобы минимизировать общее время выполнения. Для решения этой проблемы (или для определения наилучшего подхода) проведено много исследований.

В этом исследовании к *JSSP* применен алгоритм дискретной искусственной пчелиной колонии (ДИПК, *Discrete Artificial Bee Colony — DABC*) и метод оптимизации на основе преподавания/обучения (ООПО, *Teaching-Learning-Based Optimization — TLBO*).

Цель исследования — оценка эффективности алгоритма *DABC* и метода *TLBO* на многих тестах задачи планирования работы цеха.

Методы. Для поиска эффективного решения используются стохастические методы поиска, такие как эволюционные алгоритмы, с которыми сравниваются методы дискретной искусственной пчелиной колонии и оптимизации на основе преподавания/обучения.

Результаты. Показано использование алгоритмов преподавания/обучения для решения задач календарного планирования с критерием минимизации времени выполнения (значения C_{max}).

Выводы. Выполнено сравнение методов дискретной штучной пчелиной колони и оптимизации на основе преподавания/обучения. Вычислительные тестовые эксперименты показывают, что результаты, полученные с помощью *DABC* со значениями текущих параметров управления и *TLBO*, близки оптимальным результатам известного генетического алгоритма.

Эксперименты с алгоритмами *DABC* и *TLBO* для разных параметров и критериев остановки показывают, что *TLBO* демонстрирует лучшие значения C_{max}, чем *DABC*, и хуже, чем генетический алгоритм. Вычислительный эксперимент показывает, что результаты, полученные *DABC* и *TLBO* для задач планирования потоковой линии (реальные производственные системы), дали оптимальные значения C_{max}.

Ключевые слова: *алгоритм дискретной искусственной пчелиной колонии; оптимизация на основе преподавания/обучения; задача календарного планирования работы цеха; теория расписаний; время выполнения.*