IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 11, No. 2, June 2022, pp. 423~431
ISSN: 2252-8938, DOI: 10.11591/ijai.v11.i2.pp423-431

Solving flexible job-shop scheduling problem using harmony search-based meerkat clan algorithm

Muna Mohammed Jawad[1], Muhanad Tahrir Younis[2], Ahmed T. Sadiq[3]
[1]Department of Control and Systems Engineering, University of Technology-Iraq, Baghdad, Iraq
[2]Department of Computer Science, Collage of Science, Mustansiriyah University, Baghdad, Iraq
[3]Department of Computer Science, University of Technology-Iraq, Baghdad, Iraq

Article Info

Article history:

Received Aug 14, 2021
Revised Feb 2, 2022
Accepted Feb 20, 2022

Keywords:

Flexible job shop scheduling problem
Harmony search
Meerkat clan algorithm

ABSTRACT

The classical job shop scheduling (JSS) problem can be extended by allowing processing of an operation by any machine from a given set. This type of scheduling is known as flexible job shop scheduling (FJSS) problem. It incorporates all the difficulties and complexities of its predecessor classical problem. However, it is more complex as it is required to determine the assignment of operations to the machine. Swarm intelligence techniques proved their effectiveness in solving a wide range of complex NP-Hard real world problems. One of these techniques is the meerkat clan algorithm (MCA) that has been successfully applied to various optimization problems. This paper presents a modified MCA for solving the FJSS problem. The modification is based on using harmony search (HS). The introduction of HS provides more exploitation and intensification. HS generates various solutions, which are provided to the MCA. As a result, the exploitation of the local optimum is increased, which in turn increases the convergence rate. The experimental results show that the improved method achieves higher quality schedules. Additionally, the convergence rate is speeded up compared with the standalone algorithm. This gives the proposed method the superiority over the original algorithm.

Corresponding Author:

Muhanad Tahrir Younis
Department of Computer Science, College of Sciences, Mustansiriyah University
Baghdad, Iraq
Email: mty@uomustansiriyah.edu.iq

1. INTRODUCTION

Scheduling can be defined as the process of mapping jobs to resources that aims to optimize a certain objective. Therefore, it can be seen as a decision making process [1]. Providing efficient and effective schedulers plays a primary role in the growing of many types of industries, such as administration (scheduling courses at university and human resource management), electric vehicle scheduling, the execution of applications in computer systems, project management, the production work shop, and wireless network optimization [2], and power scheduling [3], [4]. The scheduling problem in the manufacturing systems includes the allocation of operations within a given task to a set of available resources aiming at optimizing required objectives, such as makespan, lateness, flowtime, and tardiness. The literature includes various types of scheduling problems that were discussed and several performance perspectives were measured. Some examples of important scheduling problems include: Single machine scheduling, parallel machine scheduling, job shop scheduling (JSS), flow shop scheduling, job scheduling in grid and cloud computing, job scheduling in heterogeneous environments, and project scheduling [5]. A typical classical JSS problem can be defined as the assignment of a set of jobs to a set of resources in terms of optimizing some defined objectives [6]. Furthermore, some classical scheduling problems may include certain constraints on the processing of the jobs, such as its execution order [7]. On the other hand, an extension of the classical JSS problem is the flexible job shop scheduling (FJSS) problem, in which any operation is allowed to be performed by any available machine. It inherits all the complexities and difficulties of the classical JSS problem. However, it is more complex than JSP mainly due to the additional need for determining the allocation of operations to the resources. A typical FJSS problem comprises two sub-problems, namely specify the sequence of operations and resource allocation. The latter is to assign each operation to a resource from a set of resources, whereas the former is to allocate all operations to all resources to achieve high-quality schedules [8]. In terms of complexity, the FJSS problem is known to be an NP-hard problem [7], [9]. Therefore, traditional optimization approaches are not efficient to address it within a reasonable amount of time [8]. To cope with its complexity, meta-heuristic optimization algorithms can be excellent candidates as they proved their effectiveness in solving many optimization problems that share similar characteristics with FJSS problem [10]. In recent years, meta-heuristic methods refer to the class of all modern higher-level strategies [7], which include tabu search (TS), simulated annealing (SA), harmony search (HS), genetic algorithms (GA), ant colony optimization (ACO), particle swarm optimization (PSO), artificial bee colony (ABC), and firefly algorithms (FA) [10].

Kamble *et al.* [10] introduced a hybrid approach called multi-objective particle swarm optimization (MPSO) algorithm, which is based on PSO and SA, for solving FJSS problem in terms of minimizing machine idle time, total tardiness, total workload, maximal machine workload and makespan simultaneously. The method was able to deal with any breakdown that may occur in any machine by shuffling its workload through invoking a rescheduling technique. The suggested hybrid method integrated the capability of PSO in performing efficient high global search with the SA, which is very effective in avoiding getting trapped in local minimum. The results showed that the proposed approach outperformed some state-of-the-art methods selected from the literature in terms of quality of non-dominated solutions. Dehghan-Sanej *et al.* [11] addressed the problem of JSS with reverse flows under uncertainty, in a robust programming technique was employed. The authors proposed a model for solving JSS problem that is efficient in terms of computations. Moreover, the study proposed different meta-heuristics for solving JSS problem, in which different datasets with variable sizes where used, to tackle its complexity. Among these meta-heuristics, the discrete HS procedure was utilized to cope with instances from medium to large size. According to comparison done with some selected meta-heuristic methods from the literature was performed. The experimental results showed that the proposed model outperformed the other meta-heuristics involved in the study.

Al-Obaidi and Hussein [12] introduced two modifications on the cuckoo search, which were called cuckoo search-best neighbors generation (CS-BNG) and cuckoo search- iterative levy flight (CS-ILF), to solve the FJSS problem. The first modification combined the cuckoo search (CS) with the best neighbors generation (BNG) procedure to enhance the convergence rate and provide diversity. The BNG procedure mainly uses the separation of the generated solutions rather than the deleted ones. The second modification integrated CS with the iterative levy flight (ILF) procedure. The performance of CS algorithm was enhanced by repeating ILF for certain number of times in an acceptable ratio (IR). Moreover, to improve the exploration of new regions in the discrete state space, the key points of CS procedure was adapted. To analyze their performance, the modified methods were examined on some FJSP benchmark instances. The reported results showed that the two modified CS-based schedulers provided high-quality solutions. Furthermore, the convergence rate was increased by these modifications in most of the tested instances compared with the standard standalone cuckoo search.

Zhou *et al.* [13] constructed a mathematical model for FJSS problem that aimed at minimizing the maximum makespan using genetic algorithm. Through analyzing the simulated results, the study constructed a mathematical model for FJSS problem by applying genetic algorithm to obtain the optimal scheduling results with MATLAB, taking the transmission reverse gear production line of a transmission corporation as an example. To justify the achieved results, they used DELMIA/QUEST, which is a three-dimensional discrete-event simulation software to emulate the production workshop. The simulation results suggested considering the genetic algorithm as a very effective meta-heuristic method in solving scheduling problems.

Al-Obaidi *et al.* [14] applied meerkat clan algorithm (MCA) for JSS problem. A meerkat is known to have several behaviors. The suggested method used only three of them, which are baby-sitter, foraging and sentry. An outstanding performance of the MCA in solving the JSS problem was noticed as it divided the solutions into two sets, namely care and foraging sets. In general, nearly all of the operations were applied on the solutions of the foraging set. Moreover, the best solutions in care set were used to take the place of the worst solutions in foraging set, whereas the worst solution in care set was replaced by a randomly generated solution. To measure the performance based on a makespan function this algorithm was employed to solve the FJSSP, with the aim at minimizing the makspan function. The MCA algorithm provided a better diversity solution through the meerkat algorithm that depends on neighbours, which, in turn, based on the sentry behaviour with foraging and baby-sitter groups. The experimental results showed that the MCA can find optimal solution compared with cuckoo search algorithm, artificial fish search algorithm, and camel herd algorithm.

Toshev [15] presented a hybrid meta-heuristic algorithm that combined PSO algorithm with (TS). The main goal of the proposed method was solving FJSS problems. To analyze the performance of the suggested hybrid scheduler, twelve benchmark instances, which were collected from various references, were experimentally examined. A comparison among the achieved results against the results reported in the references and the results obtained by the GA were made. The comparison demonstrated that the proposed hybrid scheduler has effectively enhanced the overall performance.

Luan *et al.* [16] presented a hybrid whale optimization algorithm, called IWOA, to solve the FJSS problem in terms of minimizing makespan time. It was the pioneer work that introduced a method, which converts the position vector in the whale individual into the scheduling solution. The proposed hybrid scheduler employed three different approaches to improve the whale optimization algorithm. The first approach was the chaotic reverse learning (CRL), which was used to provide the initial population with some good solutions. The second approach included two methods, which were the nonlinear convergence factor (NFC) and adaptive weight (AW), that were employed to provide exploitation and exploration. Finally, the third approach was the variable neighborhood search (VNS), which was used as a local search that improves the quality of the achieved solutions. The work performed several experiments to examine the performance of the proposed method, which included 15 benchmarks. The reported results of the proposed scheduler were compared with a number of state-of-the-art methods. The results were very promising and indicated that the hybrid IWOA can achieve high-quality schedules in a short time.

Matrenin *et al.* [17] proposed an approach to solve optimization problems aims to increase SI algorithms' efficiency and versalility with out increasing their complexity. They applied a combination of a universal, problem-free SI algorithm with simple deterministic domain-specific heuristic algorithmsm or in other words, greedy heuristic. A hybrid of stochastic universal SI algorithm and a deterministic simple domain- specific algorithm achieves a greater synergy effect than combining two swarm or evolutionary algorithms one of the approach's applications is demonstrated on the job-shop scheduling problem. The experiments confirmed that the proposed approach allows the use of SI algorithms as a meta-optimizer that increases domain-specific heuristic algorithms' efficiency.

This paper presents an improved MCA that is based on HS. The suggested improved method is utilized to solve the problem of FJSS problem. The paper is composed of four sections that are the current section which is the introduction then a section demonstrates research method and section 3 contains Results and discussion , finally a conclusion will be denoted in section 4.

2. RESEARCH METHOD

This section presents two optimization algorithms which are MCA and HS. It starts with an explanation of MCA and its general procedure for solving an optimization problem. Then, this section continues by giving an overview about HS and how can be employed to find global optima for a given optimization problem.

2.1. Meerkat clan algorithms (MCA)

A meerkat is an animal that lives socially in packs of five to thirty individuals. As sociable animals, they exchanged the parental care and bathing responsibilities. Each pack contains a leading alpha male and leading alpha female. Each pack occupies its own home. They occasionally change their place if they do not find food or come across a stronger pack. If they have been attacked by a stronger pack, the weaker pack will try to rise or stay otherwise until they get stronger and restore their lost home [18].

A pack of meerkats additionally contains a so-called sentry, who is responsible for guarding the pack. If any risk or danger is being spotted, the sentry meerkat will notify the other members of the pack. Normally, the sentry performs his surveillance duties by monitoring from the ground, climbing a tree or from hiding in the bushes. Additionally, the sentry keeps watching the other members as they search for food as well as keeps his eye on the burrow. When danger is observed, the sentry will emit a loud sound and then the crowd will quickly escape to their hiding place. The main steps of MCA as [18]:

– Step 1: Generate the solutions randomly, which are represent the initial clan of meerkat.
– Step 2: Calculate the fitness value for the initial clan, the let the sentry equal to the best one.
– Step 3: Split the clan (solutions) into two set foraging and care.
– Step 4: Produce K neighbors from each solution in the foraging set and select the best one to be the new solution.
– Step 5: Exchange the worst solutions in the foraging set by randomly ones in the care set.

- Step 6: Drop (delete) the worst solutions in the care set and generate randomly instead of the drop ones.
- Step 7: Replace the sentry by a best solution in the foraging set (if the best one is better than the sentry).
- Step 8: Repeat from Step 4 Until termination criteria is met.

2.2. HS algorithm

The process of solving an optimization problem is closely related to the musical improvisation. To solve an optimization problem means tuning a number of decision variables that are defined in advance during the process of finding the global optimum to a given objective function. Certainly, the solution vector of an optimization problem is composed from these decision variables. The fitness of the solution vector is computed by substituting the values of the decision variables in the objective function. The values of the decision variables in the solution vector are iteratively updated until achieving the global optimum [19].

Researchers have found such an interesting connection between the above scenario and the musical improvisation. This leads to the development of a novel method, which is called HS. It was first proposed by Geem *et al.* [20]. Since its emergence, HS has been successfully applied for a wide range of complex applications despite the fact that it is relatively considered a new meta-heuristic approach. Therefore, it will be interesting to examine its performance in solving new optimization problems [21].

In HS algorithm, the solution vector is called harmony, in which the decision variables that it contains are called notes. Moreover, a harmony memory (HM) is included in HS to store N harmonies. The HS algorithm aims at minimizing and/or maximizing an objective function that contains some decision variables. The HS procedure consists of five basic steps which are:

- Step 1: This step is the initialization step, in which the initial values of the HS parameters are set. These parameters include, iterations number (NI), HM size, HMCR, PAR and BW. The goal of the optimization problem is specified in this step, i.e. to maximize or minimize the objective function f(x_i), where x_i is the solution vector that consists of N decision variables.
- Step 2: Initial values of HM is set. Initial values for x_i is selected using the lower and upper boundary ranges, as indicated in (1):

$$x_i = lowerbound\ +\ RND \times (upperbound - lowerbound) \tag{1}$$

  where RND is a randomly generated number in the range (0, 1).
- Step 3: Three major parameters, namely HMCR, PAR, and BW, are used to generate the new harmony. This harmony consists of two main steps. The first step includes generating two random numbers (a and b) in the range (0, 1). Then,. (1) is used to generate new values for x_j if (a > HMCR) and these values are copied to x_{0j}. Otherwise, the second step selects a random value from HM (x_i) if (a < HMCR). However, the values of xi are altered using (2) if (b < PAR):

$$x_{0j} = x_{newj}\ \pm bw * rand \tag{2}$$

- Step 4: The fitness function of x_{0j} is calculated and if it is better than the worst fitness in HM, it will be updated.
- Step 5: After each new harmony, HS tests the stopping condition, such as reaching the maximum number of iterations. If the stopping condition is met, the search process is terminated.

2.3. Proposed method

To get the advantages of the two optimization algorithms, we propose a hybrid method that combines HS and MCA for solving FJSS problem. The resulted method will take best characterstics of the hybrid methods. As a result, an efficient scheduler is expected, which can find high quality schedules.

2.3.1. Representation of the solutions

The typical FJSS problem consists of two parts, namely operation sequence and machine allocation. Therefore, the solution can be represented using two vectors V1 and V2, where V1 is the sequence vector, and V2 is the assignment vector. Figure 1 demonstrates how a solution of the FJSS problem with 3 jobs and 3 machines is represented.

The operation sequence is represented using the permutation-based scheme, which was first proposed by Gen *et al.* [22]. The job number is used as a symbol to name all operations of a job. The operation of a job number is determined according to the order in which the job appears in the sequence vector, that is, a job number that appears in the sequence vector is equal to the operand contained in the job. This representation always represents a feasible sequence of operations, because the priority constraint has not been broken.

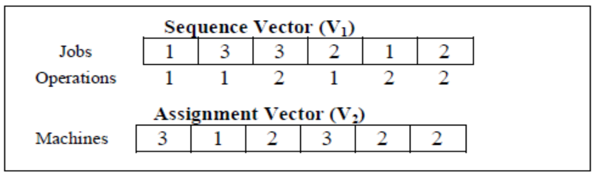

Figure 1. Representation of the solution

2.3.2. A modified meerkat clan algorithm via harmony search (MCA-HS)

The MCA is considered one of the good algorithms in terms of finding solutions with good efficiency, but it certainly does not always reach the best solution as any of the swarm intelligence algorithms. The MCA uses HS to consolidate good solutions. In the MCA, the foraging group solutions will be updated using HS, the initial population for the HS algorithm will be foraging group, and through HS processes and steps, modified and better solutions in terms of the fitness function will be obtained. This procedure will result in solutions that are quite improved in diversity. The diversity of the resulting solutions will accelerate the process of finding the best solutions or good solutions in terms of fitness function. Therefore, the modified steps will found in the end of MCA steps (inside the main loop). The MCA based on HS is listed in algorithm 1 (as seen in Appendix).

3. RESULTS AND DISCUSSION

This section presents the results of applying the basic MCA for solving the FJSS problem. Moreover, the proposed algorithm (MCA-HS) is also applied for solving the FJSS problem. This will allow us analyzing their performances. We present some adaptation for MCA and MCA-HS methods to allow them performing the search process in discrete state space of the problem by some neighborhood strategies.

MATLAB language was used to implement both MCA and MCA-HS. The implementations were ran on a computer that contains Intel® Core™ i7 2.70 GHz processor with 8 GB RAM and 1 TB hard drive. In [23] benchmark was used to examine the performance of the MCA and MCA-HS, in which three different instance sets, namely “edata”, “rdata”, and “vdata”, were selected. The set of machines in these instances, which are assignable with a particular probability distribution, has been expanded. Each instance consists of n jobs and m machines, and hence, the size of the sequence vector and assignment vector are equal to (n×m). Both MCA and MCA-HS were tested on the same instances of FJSS problem using the same parameter values, which include problem size, Meerkat population size and Harmony memory size as shown in Table 1, where max generation values are between 700 and 1,000.

Table 1. Problem and population size values

Problem size (n×m)	Meerkat population size	Harmony memory size
≤ 36	20	10
≤ 50	30	14
≤ 75	40	18
Others	60	24

For each instance, 10 independent runs of the standalone MCA and MCA-HS methods were performed. The average of these runs was calculated to provide meaningful results. These independent runs are reported in Table 2. In Table 2, LB represents the lower bound for each instance, which is reported in [18], the Average Makespan of both MCA and MCA-HS for each instance is referred as AMK, whereas RE is the relative error for MCA and MCA-HS which is computed using (3):

$$RE = \left[\frac{(AMK-LB)}{LB}\right] \tag{3}$$

For all tested instances, the reported results in Table 2 show that the makespan time achieved by MCA-HS is outperforming the standalone MCA. Moreover, the obtained makespan time is very close to the lower bound values. Additionally, the results indicate that the average relative error (ARE) of the proposed MCA-HS is less than its corresponding of the standalone MCA.

Furthermore, the results achieved from the proposed MCA-HS approach are compared with some other meta-heuristic methods that proposed in [14], [24]–[26], as shown in Table 3. The RE in the makespan time of the resulted solutions from each instance of the suggested meta-heuristics in [12], [24]–[26] is computed using (3). Our proposed algorithm (MCA-HS) has an ARE greater than the (AFSA-HS, AFSA-VND and CS-ILF) algorithms, but (MCA-HS) has an ARE less than the CHA.

Table 2. The results obtained by MCA and MCA-HS

			MCA [14]		MCA-HS	
Instances	Size	LB	AMK	RE%	AMK	RE%
edata-mt06	6×6	55	55	0	55	0
edata-mt10	10×10	871	952	0.09	950	0.09
edata-la1	10×5	609	780	0.28	653	0.07
edata-la2	10×5	655	724	0.11	678	0.04
edata-la3	10×5	550	706	0.28	598	0.09
edata-la4	10×5	568	703	0.24	607	0.07
edata-la5	10×5	503	655	0.3	543	0.08
edata-la6	15×5	833	1229	0.48	887	0.06
edata-la7	15×5	762	1136	0.49	821	0.08
edata-la8	15×5	845	1129	0.34	918	0.09
rdata-mt06	6×6	47	51	0.09	47	0
rdata-mt10	10×10	679	861	0.27	768	0.13
rdata-la1	10×5	570	789	0.38	608	0.07
rdata-la2	10×5	529	770	0.46	579	0.09
rdata-la3	10×5	477	708	0.48	498	0.04
rdata-la4	10×5	502	720	0.43	549	0.09
rdata-la5	10×5	457	667	0.46	496	0.09
rdata-la6	15×5	799	1252	0.57	891	0.12
rdata-la7	15×5	749	1123	0.5	807	0.08
rdata-la8	15×5	765	1203	0.57	797	0.04
vdata-mt06	6×6	47	49	0.04	47	0
vdata-mt10	10×10	655	854	0.3	684	0.04
vdata-la1	10×5	570	815	0.43	621	0.09
vdata-la2	10×5	529	770	0.46	609	0.15
vdata-la3	10×5	477	706	0.48	535	0.12
vdata-la4	10×5	502	706	0.41	513	0.02
vdata-la5	10×5	457	645	0.41	493	0.08
vdata-la6	15×5	799	1212	0.52	886	0.11
vdata-la7	15×5	749	1130	0.51	818	0.09
vdata-la8	15×5	765	1181	0.54	861	0.13
Average RE			0.364			0.075

Table 3. Comparing the proposed method with some methods from the literature

		MCA-HS		AFSA-HS [24]		AFSA-VND [25]		CS-ILF [**12**]		CHA [26]	
Instances	LB	AMK	RE%	AMK	RE%	AMK	RE%	AMK	RE%	AMK	RE%
edata-mt06	55	55	0	55	0	55	0	55	0	55	0
edata-mt10	871	950	0.09	912	0.05	953	0.09	979	0.12	953	0.09
edata-la1	609	653	0.07	609	0	617	0.01	634	0.04	888	0.46
edata-la2	655	678	0.04	658	0	686	0.05	694	0.06	823	0.26
edata-la3	550	598	0.09	564	0.03	568	0.03	588	0.07	732	0.33
edata-la4	568	607	0.07	590	0.04	598	0.05	619	0.09	830	0.46
edata-la5	503	543	0.08	503	0	511	0.02	526	0.05	681	0.35
edata-la6	833	887	0.06	833	0	850	0.02	861	0.03	1322	0.59
edata-la7	762	821	0.08	778	0.02	799	0.05	819	0.07	1255	0.65
edata-la8	845	918	0.09	851	0.01	860	0.02	868	0.03	1257	0.49
rdata-mt06	47	47	0	47	0	47	0	55	0.17	47	0
rdata-mt10	679	768	0.13	754	0.11	824	0.21	801	0.18	811	0.19
rdata-la1	570	608	0.07	582	0.02	590	0.04	609	0.07	665	0.17
rdata-la2	529	579	0.09	544	0.03	554	0.05	567	0.07	633	0.2
rdata-la3	477	498	0.04	488	0.02	497	0.04	512	0.07	590	0.24
rdata-la4	502	549	0.09	516	0.03	528	0.05	538	0.07	623	0.24
rdata-la5	457	496	0.09	466	0.02	476	0.04	480	0.05	568	0.24
rdata-la6	799	891	0.12	804	0.01	816	0.02	821	0.03	1044	0.31
rdata-la7	749	807	0.08	755	0.01	767	0.02	776	0.04	1005	0.34
rdata-la8	765	797	0.04	770	0.01	788	0.03	790	0.03	1061	0.39
vdata-mt06	47	47	0	47	0	47	0	48	0.02	47	0
vdata-mt10	655	684	0.04	677	0.03	814	0.14	729	0.11	715	0.09
vdata-la1	570	621	0.09	577	0.01	587	0.04	609	0.07	671	0.18
vdata-la2	529	609	0.15	538	0.02	551	0.06	564	0.07	560	0.06
vdata-la3	477	535	0.12	486	0.02	497	0.05	520	0.09	574	0.2
vdata-la4	502	513	0.02	511	0.02	528	0.05	531	0.06	641	0.28
vdata-la5	457	493	0.08	469	0.03	472	0.05	499	0.09	531	0.16
vdata-la6	799	886	0.11	807	0.01	814	0.03	821	0.03	1042	0.3
vdata-la7	749	818	0.09	755	0.01	763	0.03	773	0.03	968	0.29
vdata-la8	765	861	0.13	770	0.01	786	0.02	787	0.03	908	0.19
Average RE			0.075		0.019		0.0437		0.0647		0.2583

4. CONCLUSION

MCA is one of the most effective swarm-based approaches for handling a variety of complex real-world optimization problems. It would be interesting to enhance its performance in order to solve the FJSS problem, which is a critical issue in modern production systems. This work offered an HS-based improvement to the MCA method. The HS in MCA-HS gives the proposed scheduler extra exploitation and exploration capabilities compared to the standalone MCA. The different solutions given by HS in MCA can boost the exploitation of the local optimum, resulting in a higher convergence rate. The proposed technique improves the MCA's capacity to do local searches. In comparison to the basic MCA, the improved MCA (MCA-HS) has superior quality solutions for all examined instances. Moreover, the convergence rate in most instances is increased according to the experimental data. This distinguishes MCA-HS from the standalone MCA. Despite the fact the reported results in this work are very promising; a room for further improvements is still available. As a future work, it will be interesting to examine the performance of the proposed scheduler against the multi-objective version of FJSS problem.

REFERENCES

[1] J. A. Larco, J. C. Fransoo, and V. C. S. Wiers, “Scheduling the scheduling task: a time-management perspective on scheduling,” *Cognition, Technology and Work*, vol. 20, no. 1, pp. 1–10, Feb. 2018, doi: 10.1007/s10111-017-0443-1.

[2] P. Ganesan, M. K. Marichelvam, and G. D. Sivakumar, “A hybrid jaya algorithm to solve flexible job shop scheduling benchmark problems,” *International Journal of Pure and Applied Mathematics*, vol. 120, no. 6, pp. 6893–6904, 2018.

[3] M. L. Pinedo, *Scheduling: theory, algorithms, and systems*. Springer, 2016.

[4] Z. Pooranian, M. Shojafar, and B. Javadi, “Independent task scheduling in grid computing based on queen bee algorithm,” *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 1, no. 4, pp. 171–181, Oct. 2012, doi: 10.11591/ij-ai.v1i4.1229.

[5] M. Hazim Lokman *et al.*, “Multi-verse optimization based evolutionary programming technique for power scheduling in loss minimization scheme,” *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 8, no. 3, pp. 292–298, Dec. 2019, doi: 10.11591/ijai.v8.i3.pp292-298.

[6] M. T. Younis and S. Yang, “Hybrid meta-heuristic algorithms for independent job scheduling in grid computing,” *Applied Soft Computing*, vol. 72, pp. 498–517, Nov. 2018, doi: 10.1016/j.asoc.2018.05.032.

[7] M. Abdolrazzagh-Nezhad and S. Abdullah, “Job shop scheduling: classification, constraints and objective functions,” *International Journal of Computer and Information Engineering*, vol. 11, no. 4, pp. 429–434, 2017, [Online]. Available: https://waset.org/publications/10006691/job-shop-scheduling-classification-constraints-and-objective-functions.

[8] I. A. Chaudhry and A. A. Khan, “A research survey: review of flexible job shop scheduling techniques,” *International Transactions in Operational Research*, vol. 23, no. 3, pp. 551–591, 2016, doi: 10.1111/itor.12199.

[9] W. T. Lunardi and H. Voos, “An extended flexible job shop scheduling problem with parallel operations,” *ACM SIGAPP Applied Computing Review*, vol. 18, no. 2, pp. 46–56, Jul. 2018, doi: 10.1145/3243064.3243068.

[10] S. V Kamble, S. U. Mane, and A. J. Umbarkar, “Hybrid multi-objective particle swarm optimization for flexible job shop scheduling problem,” *International Journal of Intelligent Systems and Applications*, vol. 7, no. 4, pp. 54–61, 2015, doi: 10.5815/ijisa.2015.04.08.

[11] K. Dehghan-Sanej, M. Eghbali-Zarch, R. Tavakkoli-Moghaddam, S. M. Sajadi, and S. J. Sadjadi, “Solving a new robust reverse job shop scheduling problem by meta-heuristic algorithms,” *Engineering Applications of Artificial Intelligence*, vol. 101, May 2021, Art. no. 104207, doi: 10.1016/j.engappai.2021.104207.

[12] A. T. Saadeq Al-Obaidi and S. A. Hussein, “Two improved cuckoo search algorithms for solving the flexible job-shop scheduling problem,” *International Journal on Perceptive and Cognitive Computing*, vol. 2, no. 2, Nov. 2016, doi: 10.31436/ijpcc.v2i2.34.

[13] E. Zhou, J. Zhu, and L. Deng, “Flexible job-shop scheduling based on genetic algorithm and simulation validation,” *MATEC Web of Conferences*, vol. 100, Mar. 2017, doi: 10.1051/matecconf/201710002047.

[14] A. T. Sadiq Al-Obaidi, H. S. Abdullah, and Z. O. Ahmed, “Solving flexible job shop scheduling problem using meerkat clan algorithm,” *IRAQI JOURNAL OF SCIENCE*, vol. 59, no. 2A, pp. 754–761, Apr. 2018, doi: 10.24996/ijs.2018.59.2A.13.

[15] A. Toshev, “Particle swarm optimization and tabu search hybrid algorithm for flexible job shop scheduling problem – analysis of test results,” *Cybernetics and Information Technologies*, vol. 19, no. 4, pp. 26–44, Nov. 2019, doi: 10.2478/cait-2019-0034.

[16] F. Luan, Z. Cai, S. Wu, T. Jiang, F. Li, and J. Yang, “Improved whale algorithm for solving the flexible job shop scheduling problem,” *Mathematics*, vol. 7, no. 5, Apr. 2019, doi: 10.3390/math7050384.

[17] P. Matrenin *et al.*, “Generalized swarm intelligence algorithms with domain-specific heuristics,” *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 10, no. 1, pp. 157–165, Mar. 2021, doi: 10.11591/ijai.v10.i1.pp157-165.

[18] A. T. S. Al-Obaidi, H. S. Abdullah, and Z. O. Ahmed, “Meerkat clan algorithm: a new swarm intelligence algorithm,” *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 10, no. 1, pp. 354–360, Apr. 2018, doi: 10.11591/ijeecs.v10.i1.pp354-360.

[19] C. Anand Deva Durai, M. Azath, and J. S. C. Jeniffer, “Integrated search method for flexible job shop scheduling problem using HHS–ALNS algorithm,” *SN Computer Science*, vol. 1, no. 2, Mar. 2020, Art. no. 83, doi: 10.1007/s42979-020-0084-y.

[20] Z. W. Geem, J. H. Kim, and G. V. Loganathan, “A new heuristic optimization algorithm: harmony search,” *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[21] X. Z. Gao, V. Govindasamy, H. Xu, X. Wang, and K. Zenger, “Harmony search method: theory and applications,” *Computational Intelligence and Neuroscience*, pp. 1–10, 2015, doi: 10.1155/2015/258491.

[22] M. Gen, Y. Tsujimura, and E. Kubota, “Solving job-shop scheduling problems by genetic algorithm,” in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 1994, vol. 2, pp. 1577–1582, doi: 10.1109/ICSMC.1994.400072.

[23] D. Behnke and M. J. Geiger, “Test instances for the flexible job shop scheduling problem with work centers,” 2012.

[24] I. F. Faeq, M. G. Duaimi, and A. T. Sadiq Al-Obaidi, “An efficient artificial fish swarm algorithm with harmony search for scheduling in flexible job-shop problem,” *Journal of Theoretical and Applied Information Technology*, vol. 96, no. 8, pp. 2287–2297, 2018.

[25] A. T. Sadiq Alobaidi and S. A. Hussein, “An improved artificial fish swarm algorithm to solve flexible job shop,” in *2017 Annual Conference on New Trends in Information and Communications Technology Applications (NTICT)*, Mar. 2017, pp. 7–12, doi: 10.1109/NTICT.2017.7976155.

[26] A. T. Sadiq Al-Obaidi, H. S. Abdullah, and Z. O. Ahmed, "Camel herds algorithm: a new swarm intelligent algorithm to solve optimization problems," *International Journal on Perceptive and Cognitive Computing*, vol. 3, no. 1, May 2017, doi: 10.31436/ijpcc.v3i1.44.

APPENDIX

Algorithm 1. MCA-HS procedure

```
Input
       n              clan size
       m              foraging size   where m < n
       c              care size       n-m-1
       Frate          worst foraging rate
       Crate          worst care rate
       k              neighbor solution
       HM_Size Harmony memory size;
Output
       Best Solution
Begin
    Randomly generate a clan of (n) solutions, clan[n];
    For each individual (i), i=1..n, calculate fitness clan[i];
    Let Sentry be the best clan's solution;
    Split the clan into two sets (foraging and care)
    While the termination criteria is not met Do
             For each solution in foraging set Do
             Foraging[i] ← Best of Neighbor Generator (k, Sentry, foraging[i]);
             end for
           Exchange Frate of the worst solutions in foraging set with best Frate solutions
from the care set;
           Replace Crate worst solutions from care set with randomly generated Crate
solutions;
           Best-Foraging ← the best solutions in foraging set;
           Harmony_Memory [1..HM-Size] = updated Foraging set;
           Perform HS Algorithm on Best-Foraging and compute Best_HM
           if F(Best_HM) < F(Best-Foraging)
              Best_AF=Best_HM
           end if
           If Best-Foraging ≤ Sentry then
       Sentry=Best-Foraging
           end if
    end while
End
```

Harmony Search Algorithm

```
Input: Current Best-Foraging Solution.
Output: Best_HM
Begin
    Repeat
        Select a solution V from HM;
        Z ←Perturb V;
        if Z is better than the Best_Foraging solution Then
              Replace the solution in HM with new solution;
        end if
    Until (termination criteria is satisfied);
    Find Best_HM which has the best solution in HM;
End.
```

BIOGRAPHIES OF AUTHORS

Muna Mohammed Jawad received a B.Sc., M.Sc., and Ph.D. degree in Computer Science from the University of Technology, Computer Science Department, Iraq, 1988, 1994, and 1999 respectively. She is assistant profesor in AI and Network Management since 2012. Her research interests in most of Artificial intelligence areas. She can be contacted at email: 120012@uotechnology.edu.iq.

Muhanad Tahrir Younis received the B.Sc. degree in Computer Science from the Department of computer science, college of sciences, Mustansiriyah University, Baghdad, Iraq in July 2002, the Higher Diploma and M.Sc. degrees in Computer Science-Artificial Intelligence from the Department of computer science, University of Technology, Baghdad, Iraq in December 2003 and October 2007, respectively and the Ph.D. degree in Computer Science - Artificial Intelligence from the School of Computer Science and Informatics, De Montfort University, Leicester, United Kingdom in October 2018. His research interest covers most of AI areas. He can be contacted at email: mty@uomustansiriyah.edu.iq.

Ahmed T. Sadiq received a B.Sc., M.Sc., and Ph.D. degree in Computer Science from the University of Technology, Computer Science Department, Iraq, 1993, 1996, and 2000 respectively. He is Professor in AI since 2014. His research interests in artificial intelligence, data security, patterns recognition and data mining. He can be contacted at email: Ahmed.T.Sadiq@uotechnology.edu.iq.