Frontiers in Scientific Research and Technology

Online ISSN: 2682-2970
Print ISSN: 2682-2962
http://fsrt.journals.ekb.eg

FSRT J 7 (2023) 64 - 69 10.21608/FSRT.2023.234065.1106

Implementation of a Binary Kidney-Inspired Algorithm Based on a Parallel Method for Solving the Job Shop Scheduling Problem

Wael Mohamed Fawaz Abdel-Rehim

Computer Science Department, Faculty of Computers and Information, Suez University, Suez, Egypt.

ARTICLE INFO

Article history:
Received 4 September 2023
Received in revised form 10 September 2023
Accepted 13 September 2023
Available online 16 September 2023

Keywords
Job Shop Scheduling Problem (JSSP),
Optimization,
MATLAB,
Meta-heuristics,
Kidney-inspired algorithm.

ABSTRACT

This paper proposes a binary kidney-inspired algorithm (KA) to tackle the job shop scheduling problem (JSSP), and its solution is essential in manufacturing, especially in real industrial engineering. The job shop scheduling problem is a computer science optimization problem, and it is among the most significant and challenging issues in the field of production scheduling. The proposed algorithm is based on a parallel method with many threads. This algorithm is checked using certain job shop benchmark problems. The findings are compared with those retrieved using two techniques: a genetic algorithm (GA) and a particle swarm optimization (PSO) method. These techniques indicate that applying the binary kidney-inspired in parallel is an efficient algorithm for solving the JSSP, shows remarkable competitiveness, and considerably accelerates speedups, especially in large-scale instances. The achieved results are based on four threads; the speedup is 3.13 for the FT06 instance, while the execution time is 2.24 seconds.

1. Introduction

Recent research indicates that nature-inspired [1] and bio-inspired meta-heuristic algorithms are the most preferable option of all meta-heuristic algorithms. Explicitly solving many optimization problems. The Kidney-inspired algorithm (KA) is an optimization technique empowered by the kidney mechanism of human body exploitation and exploratory capacities [2]. The job-shop scheduling problem (JSSP) is among the majority of significant combinatorial optimization issues among typical production scheduling problems [3-6]. The definition is given below: *M* machines and *N* jobs are presented such that each job includes *M* tasks, each of fixed processing time and requires a given machine, and each job reaches each machine only once. The primary purpose of the JSSP is to reduce the maximum execution time of all jobs (makespan). It is common knowledge that the JSSP problem is NP-hard.

Kennedy and Eberhart [1] originally proposed the particle swarm optimization (PSO) algorithm for solving optimization problems. This algorithm has been successfully applied in many research and application areas [7,8], and it has been applied to various combinatorial optimization issues, such as job-shop scheduling problems. Zelenka [9] solved the JSSP using the PSO in parallel based on a MATLAB distributed computing server.

* Corresponding author at Suez University

E-mail addresses: w.fawaz@suezuni.edu.eg (Wael Mohamed Fawaz Abdel-Rehim)

Syarif et al. [10] show various advances in the genetic algorithm (GA) to solve difficult optimization problems in the lot from inventory and production design issues. Kuczapski et al. [11] proposed a method of enhancing genetic algorithms for the JSSP by generating near-optimal initial populations. In [12], X. Tian and X. Liu proposed a hybrid heuristic algorithm for solving the JSSP by incorporating improved GA, modified rumor PSO, and fine-grained local search methods.

The majority of recent research on binary KA focuses on continuous optimization problems, while research on scheduling problems is few. However, more research is needed on scheduling problems. Moreover, more research needs to be shown in the JSSP and KA areas. The kidney-inspired algorithm has successfully been applied to numerous combinatorial optimization issues, for instance, solving the traveling salesman problem [13]. The results show that their algorithm outputs outstanding solutions to typical benchmark issues.

The binary KA structure is almost essentially parallel, making implementing it efficiently using a parallel computing architecture easier. The KA algorithm is an evolutionary calculation technique computationally more straightforward and can converge more quickly than other approaches to solve JSSP. Moreover, in this work, the KA is implemented to solve the JSSP using a parallel method, and the utilized tool is MATLAB, which provides maximum efficiency.

The following contributions are as follows:

- Implementing an efficient algorithm, namely, binary KA to resolve the JSSP problem based on a parallel method.

- Evaluating the proposed algorithm with some benchmark test problems, it has demonstrated exceptional performance compared to existing algorithms.
- Demonstrating the binary KA's effectiveness and efficiency in solving the JSSP by comparing its performance with other metaheuristic algorithms, such as particle swarm optimization and genetic algorithm.
- Presenting the parallel execution of the binary KA using MATLAB enables the algorithm to benefit from the power of multi-core processors and significantly improve its computational efficiency.

The rest of the paper is organized as follows: Section 2 provides an overview of related work. Section 3 presents the proposed methodology, while Section 4 outlines the experimental results. Lastly, Section 5 presents the conclusion of the proposed work.

2. Related Work

It is a significant optimization problem in operations research and computer science. The classical job shop scheduling problem is among the largest critical and difficult production scheduling issues. Moreover, it is a subfield of industrial production scheduling issues. It has been the study of intensive research over the past 50 years in both academic and industrial where the goal is to minimize the makespan, which is the total time taken to accomplish all jobs on all machines. JSSP can be applicable to production processing and affects the time and cost of manufacturing. Its difficulty indicates that the famous 10 × 10 instance formulated by Thompson and Muth [14] remained an issue for more than 20 years because of the high combinational complexity. Several heuristics and evolutionary techniques have recently been used to seek near-optimal solutions for problems with reasonable calculation time. Tabu Search, Ant Colony Optimization, Simulated Annealing, genetic algorithm, and particle swarm optimization were most frequently adopted to solve the JSSP [15-21]. The drawback of the PSO method is that, like metaheuristics methods, it does not guarantee that an optimal solution is always found. However, the performance of KA has been evaluated in eight test functions compared to well-known algorithms, such as GA and PSO. Moreover, the binary KA achieved better convergence [2].

Taqi and Ali [22] presented a kidney-inspired algorithm, which is one of the modern approaches employed to enhance classification accuracy in a multi-algorithm problem for feature selection. Abdullah and Jaddi [23] proposed a dual-population KA system that can be used for water quality prediction and cancer diagnosis. Homaid *et al*. [24] proposed an application using KA for pairwise testing, which was used to reduce the cost of assessing software. The authors classified KA as a Metaheuristic algorithm. Du *et al.* [25] used a KA algorithm to control the performance of a terminal sliding mode controller to ensure accurate tracking of the motion of a vehicle. Abdullah and Jaddi [26] propose a dual-population KA system that can be used for water quality prediction and cancer diagnosis.

3. The Proposed Binary Kidney-Inspired Algorithm

The kidney-inspired algorithm mimics the human urinary system, applying critical physiological processes of filtration, reabsorption, and secretion [2]. The proposed binary KA algorithm depends on the traditional kidney-inspired algorithm. The kidney-inspired algorithm begins with a population of solutions that are randomized, formed and set; in the population, the solution's objective function is calculated, then the optimal solution is chosen. The movement of solutions that imitates the generation of new solutions denotes Eq. (1) [2]:

$$Sol_{i+1} = Sol_i + \beta \times (Sol_{best} - Sol_i) \quad (1)$$

A solution in the i^{th} iteration is denoted as Sol_i, and the best solution is represented as Sol_{best} and uses a random number β ranging between [0, 1]. The KA algorithm handles filtration by transferring the solutions to any W or FB. This transfer is accomplished by a threshold known as the filtration rate (f_r), and it is computed using the Eq. (2) [2]:

$$f_r = \alpha \times \frac{\sum_{i=1}^{p} f(x_i)}{p} \quad (2)$$

α is a constant value ranging between [0, 1], the solution's objective function x in the i^{th} repetition is denoted by $f(x_i)$, and p is the population size. The pseudo-code algorithm inspired by the kidney that describes the entire process performed by the algorithm [2] as follows:

Algorithm: The Proposed Pseudo-Code of the Kidney-Inspired Algorithm

```
Set the population
Evaluate the solute in the population.
Set the best solute, Sol_best
Set filtration rate, fr, Eq.(2)
Set waste, W
Set filter blood, FB.
Set number of iterations, numOfIte
Set ite ← 0
Do while (ite<numOfIte)
    For all Sol_i
        Generate new Sol_i, Eq. (1)
        Check the Sol_i, using fr
        If Sol_i, assigned to W
            Apply reabsorption and generate Sol_new, Eq. (1)
            If Sol_new cannot be a part of FB
                Remove Sol_i from W (excretion)
                Insert a random Sol into W to replace Sol
            End if
            Sol_new is reabsorbed
        Else
        If it is better than the Sol_worst in FB
            Sol_worst is secreted
        Else
            Sol_i is secreted
        End if
    End if
    End for
    Rank the Sol_s from FB and update the Sol_best
    Merge W and FB
End while
Return Sol_best
```

The algorithm consists of three main phases: filtration, reabsorption, and secretion, which are inspired by the kidney's filtration of blood and reabsorption and secretion of ions and other substances. A number of binarization methods for adapting continuous meta-heuristic optimization algorithms to resolve binary issues have been proposed [7].

Finally, the binary KA algorithm could well be modified to resolve the job-shop scheduling problem. Some researchers used binary KA in many applications [13,21]. In particular, the binary KA is utilized to work well with scheduling optimization problems. Moreover, because scheduling optimization problems have discrete solution spaces, applying binary KA to JSSP successfully presents significant challenges.

4. Experimental Results

The binary KA is implemented to solve the JSSP problem based on a parallel method using MATLAB programming language. The outcomes are compared using serial and parallel by using a parallel MATLAB with four, three, two, and one threads. The experimental outcomes are presented on a laptop with a Core i5 2.50 GHz CPU, 3M of Cache, 4GB of RAM, and MATLAB 8.1 (R2013a).

In our experimentations, several types of benchmark instances with different sizes, from small-size instances with six jobs and six machines to large-sized instances with 30 jobs and 10 machines, have been chosen for computation. FT06, FT10, and FT20 (also known as MT06, Mt10, and MT20) are three well-known benchmark JSSP problems formulated by Muth and Thompson [14]. Moreover, LA06, LA26, and LA31 are three instances of three different sizes; the cited source is [27] Contributed by Lawrence and available through the OR-Library [28]. Table 1 shows that the binary KA algorithm returns nearly optimal solutions to the JSSP problem, and the best-known solutions are marked in bold fonts. Moreover, this table demonstrates the instance's name and size (number of jobs × number of operations), the optimal solution, and the results collected by applying the GA, PSO, and the binary KA algorithm.

Table 1: Comparison between optimal solution, GA, PSO, and the proposed binary KA algorithm makespans

Instances (size)	Optimal Solution	GA Algorithm	PSO Algorithm	Binary KA Algorithm
FT06 (6×6)	55	55	55	**55**
LA01 (10×5)	666	666	666	**666**
LA06 (15×5)	926	926	926	**926**
FT10 (10×10)	930	951	939	935
FT20 (20×5)	1165	1178	1187	1177
LA12 (20×5)	1039	1039	1039	**1039**
LA26 (20×10)	1218	1391	1226	**1218**
LA31 (30×10)	1784	1749	1784	**1784**

The proposed algorithm is run ten times for each test problem. The makespans results obtained by the binary KA method are superior and converge to the optimal solution more quickly than those acquired by the GA and PSO algorithms. It is quite evident that our algorithm more converges to the optimal solution, where the GA and PSO reached the optimal solution for 50% and 62.5%, respectively, of its solved test problems, while the binary KA reached the optimal solution for 75% and the major remaining problems binary KA reached near to optimal solution than GA and PSO. Moreover, the Gantt chart results by the proposed method for the instance (FT06) are illustrated in Figure 1. Comparison between GA, PSO, and the proposed algorithm results from binary KA. For example**,** the optimal solution for Muth and Thompson's 6 × 6 problem (FT06) demonstrates that there are six machines and six jobs. There are 36 total machine operations. In FT06 JSSP, each job will involve one operation per machine, or each machine will execute every job at least once.

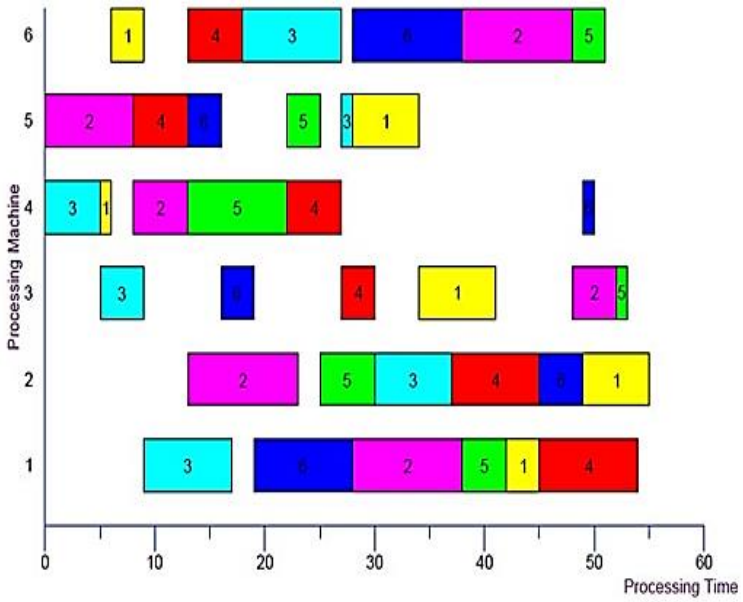

Figure 1: FT06 (6×6) problem solution Gantt chart.

Table 2 is based on the binary KA plotted using the Gantt chart in Figure 1. Also, the execution time is analyzed of the binary KA algorithm using parallel MATLAB. Table 3 displays the resulting running times. Table 3 shows a significant improvement in execution time in applying the binary KA with one (serial), two, three, and four threads using parallel MATLAB, especially for large-scale instances.

Table 2: Thompson and Muth's 6 × 6 instance (FT06)

Job	1	2	3	4	5	6
Machine (Processing Time)	3 (1)	2 (8)	3 (5)	2 (5)	3 (9)	2 (3)
	1 (3)	3 (5)	4 (4)	1 (5)	2 (3)	4 (3)
	2 (6)	5 (10)	6 (8)	3 (5)	5 (5)	6 (9)
	4 (7)	6 (10)	1 (9)	4 (3)	6 (4)	1 (10)
	6 (3)	1 (10)	2 (1)	5 (8)	1 (3)	5 (4)
	5 (6)	4 (4)	5 (7)	6 (9)	4 (1)	3 (1)

Table 3: Execution times in secs. of implementing binary KA algorithm using parallel MATLAB

Instances (size)	1 Thread (serial)	2 Threads	3 Threads	4 Threads
FT06 (6×6)	7	4.8	2.8	2.24
LA06 (15×5)	5	3.2	2.4	1.6
FT10 (10×10)	17.6	14.4	8	5
FT20 (20×5)	20.8	16	8.7	6
LA26 (20×10)	24	19.2	9.7	6.6
LA31 (30×10)	34	24	13	9

Figure 2 shows the result of the execution times for one, two, three, and four threads; this figure demonstrates that the execution time of two threads is comparable to that of three threads, while the running time of four threads is considerably greater than that of one to three threads.

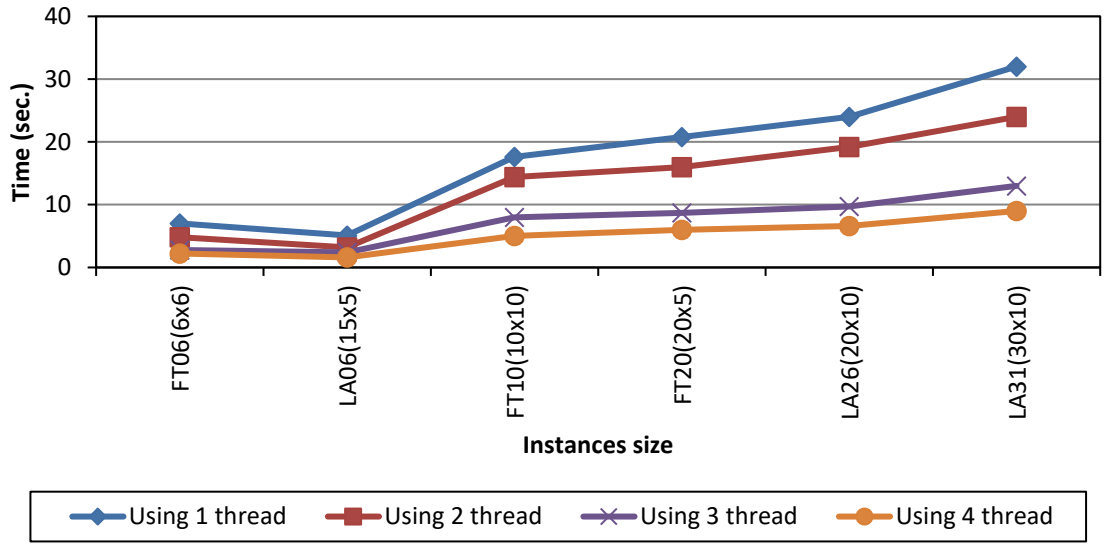

Figure 2: Execution times of implementing binary KA algorithm using one, two, three, and four threads using parallel MATLAB.

Finally, the binary KA algorithm's performance is compared according to four, three, two, and one threads using parallel MATLAB. MATLAB is used because it is a convenient high-level programming language for scientific research and supports various parallelism techniques. For instance, the multithreaded parallelism is one of them [29,30]. We are concerned with the speedup of running time for the implementation of the binary KA in parallel using MATLAB. The following equation defines the formula of the speedup: $S_n = T_1/T_n$

S_n is the speedup observed on n processors, T_1 is the running time for sequential execution, and T_n is the running time on n processors. The speedup result for two, three, and four threads is shown in Table 4 and Figure 3. The results illustrate that a binary kidney-inspired algorithm using parallel MATLAB can effectively resolve the job shop scheduling problem by significantly increasing the speedups, especially for large-scale instances.

Table 4 Speedup results for binary KA

Instances (Size)	Two Threads Speedup	Three Threads Speedup	Four Threads Speedup
FT06 (6×6)	1.46	2.5	3.13
LA06 (15×5)	1.56	2.08	3.19
FT10 (10×10)	1.22	2.2	3.52
FT10 (20×5)	1.3	2.39	3.47
LA26 (20×10)	1.25	2.47	3.64
LA31 (30×10)	1.33	2.46	3.56

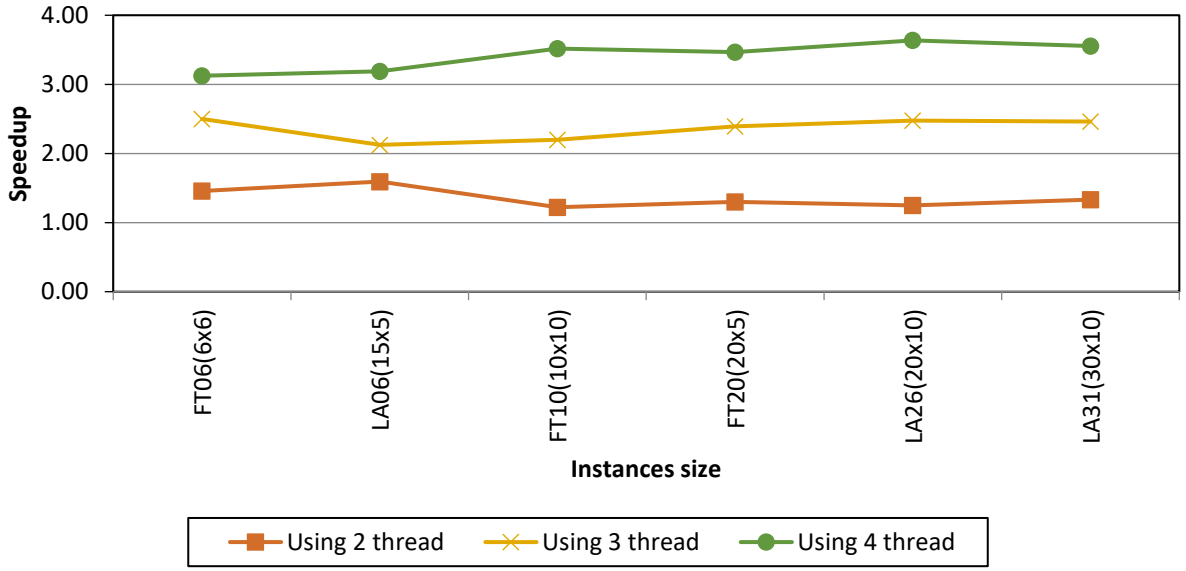

Figure 3: Speedup obtained by executing the binary KA on two, three, and four threads using parallel MATLAB.

5. Conclusion

This paper has executed the binary KA algorithm based on a parallel method using MATLAB to the job-shop scheduling problems to reduce the running time. The performance of the binary KA algorithm is evaluated in contrast to many benchmark instances. Results indicate that the algorithm generated optimal or nearly optimal solutions in each instance tested.

The results obtained by the binary KA algorithm are better and converge to the optimal solution than solutions obtained by GA or PSO. Moreover, the results indicate that the parallel binary KA algorithm implementation efficiently solves the job-shop scheduling problem and further optimization problems. Moreover, the proposed approach's running time is less than in a serial binary KA algorithm to solve the JSSP. Furthermore, it demonstrates that the proposed algorithm for solving JSSP outperforms existing algorithms.

The binary KA method is discussed for job-shop scheduling problems. The algorithm is incredibly valuable and effective. It can find optimal solutions for the majority of test instances, and its execution time is significantly smaller than that of almost all other algorithms. These outcomes show that the proposed algorithm is a viable alternative for solving the job-shop scheduling problem and other optimization issues.

Also, the execution time of two threads is comparable to that of three threads, while the running time of four threads is considerably greater than that of one to three threads when the number of sizes in the pool is sufficiently large. An area for future research is to explore using KA in combination with other optimization algorithms to increase its efficiency on the JSSP. Also, one could implement the binary KA algorithm based on other platforms with different programming languages, such as Python. Additionally, the binary KA can be used to solve other complex scheduling problems.

References

[1] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh and S. Mirjalili, “Particle Swarm Optimization: A Comprehensive Survey,” in IEEE Access, vol. 10, pp. 10031-10061, 2022.

[2] N.S. Jaddi, J. Alvankarian and S. Abdullah, “Kidney-inspired algorithm for optimization problems”, Communications in Nonlinear Science and Numerical Simulation, vol. 42, 2017, pp. 358-369.

[3] B. -A. Han and J. -J. Yang, “Research on Adaptive Job Shop Scheduling Problems Based on Dueling Double DQN,” in IEEE Access, vol. 8, pp. 186474-186495, 2020.

[4] G. G. Yen, B. Ivers, “Job shop scheduling optimization through multiple independent particle swarms, International Journal of Intelligent Computing and Cybernetics”, vol.2, no. 1, 2009, pp. -33.

[5] Y. C. Liang, H. W. Ge, Y. Zho, X. C. Guo, et al., “A particle swarm optimization-based algorithm for job-shop scheduling problems”, International Journal of Computational Methods, vol.2, no. 3, 2005, pp. 419-430.

[6] H. Yu, Y. Gao, L. Wang, and J. Meng, “A Hybrid Particle Swarm Optimization Algorithm Enhanced with Nonlinear Inertial Weight and Gaussian Mutation for Job Shop Scheduling Problems,” Mathematics, vol. 8, no. 8, p. 1355, Aug. 2020.

[7] R. Poli., “An analysis of publications on particle swarm optimization applications”, Essex, UK: Department of Computer Science, University of Essex, 2007.

[8] W. M. F. Abdel-Rehim, “Binary Particle Swarm Optimization Algorithm for Kidney Exchanges”, Egyptian Computer Science Journal (ECSJ), vol. 45, no. 2, pp. 30-43, 2021.

[9] J. Zelenka, "Parallel computing application into the particle swarm optimization algorithm used to solve the Job-Shop scheduling problem". 15th IEEE International Conference on Intelligent Engineering Systems (INES) 2011, Poprad, Slovakia, 23-25 June, 2011, pp. 183 - 189.

[10] A. Syarif, A. Pamungkas, R. Kumar, and M. Gen. "Performance Evaluation of Various Heuristic Algorithms to Solve Job Shop Scheduling Problem (JSSP)", International Journal of Intelligent Engineering & Systems, vol. 14, no. 2, 2021.

[11] A. M. Kuczapski, M. V. Micea, L. A. Maniu, and V. I. Cretu. "Efficient generation of near optimal initial populations to enhance genetic algorithms for job-shop scheduling", Information Technology and Control, vol. 39, no. 1, 2010.
[12] X. Tian and X. Liu. "Improved hybrid heuristic algorithm inspired by tissue-like membrane system to solve job shop scheduling problem", vol. 9, no. 2, 2021, pp. 219.
[13] W. M. F. Abdel-Rehim, "Solving traveling salesman problem using Kidney-Inspired algorithm". 2nd International Conference on Mathematics, Statistics & Information Technology (2nd ICMSIT 2018), Tanta, Egypt, 18-20 December 2018.
[14] J. F. Muth and G. L. Thompson. "Industrial Scheduling". Prentice-Hall, Englewood Cliffs, N.J., 1963.
[15] M. Dell' Amico and M. Trubian. "Applying tabu search to the job-shop scheduling problem". Annals of Operations Research, vol. 41, 1993, pp. 231-252.
[16] H. E. Nouri, O. B. Driss, K. Gh´edira, "Hybrid metaheuristics within a holonic multiagent model for the flexible job shop problem", Procedia Computer Science, vol. 60, 2015, pp. 83-92.
[17] M. Akhshabi, J. Haddadniab, M. Akhshabi, "Solving flow shop scheduling problem using a parallel genetic algorithm", Procedia Technology, vol. 1, 2012, pp. 351-355.
[18] M. Á. González and C. R. Vela and R. Varela, "An Efficient Memetic Algorithm for the Flexible Job Shop with Setup Times", Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling, Rome, Italy, 10-14 June 2013.
[19] Y. Yan-Fang, Y. Yue, "An improved genetic algorithm to the job shop scheduling problem", Journal of Chemical and Pharmaceutical Research, vol.7, no. 4, 2015, 322-325.
[20] X. Song, "Hybrid Particle Swarm Algorithm for Job Shop Scheduling Problems, Future Manufacturing Systems", Tauseef Aized (Ed.), 2010, ISBN: 9789533071282.
[21] W. M. F. Abdel-Rehim, I.A. Ismail, E. Morsy, "Implementing the classical poker approach for testing randomness", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 2, no. 8, 2014, pp. 98-103.
[22] M.K. Taqi and R. Ali, "Obka-Fs: An Oppositional-Based Binary Kidney-inspired Search Algorithm for Feature Selection", Journal of Theoretical and Applied Information Technology, vol. 95, no. 1, 2017, pp. 9-23.
[23] S. Abdullah & Jaddi, N. S., "Dual kidney-inspired algorithm for water quality prediction and cancer detection". IEEE Access, vol.8, no. 1, 2020. pp. 109807-109820.
[24] A. A. B. Homaid, A. A. Alsewari, A. K. Alazzawi, & K. Z. Zamli, "A kidney algorithm for pairwise test suite generation". Advanced Science Letters, vol.24, no. 10, 2018, pp. 7284-7289.
[25] M. Du, D. Zhao, B. Yang, and L. Wang, "Terminal sliding mode control for full vehicle active suspension systems," J. Mech. Sci. Technol., vol. 32, no. 6, 2018, pp. 2851–2866.
[26] S. Abdullah, and N. S. Jaddi (2020), "Dual kidney-inspired algorithm for water quality prediction and cancer detection", IEEE Access, vol. 8, no. 1, 2020, pp. 109807-109820.
[27] J. E. Beasley, "Job Shop Scheduling", 2014. http://people.brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html (Accessed 30 August 2023).
[28] S. Lawrance, "Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques", Graduate school of industrial administration, Carnegie Mellon University: Pittsburgh, 1984.
[29] The MathWorks, Inc., Cleve Moler "Parallel MATLAB: Multiple Processors and Multiple Cores".
http://www.mathworks.com/company/newsletters/articles/parallel-matlab-multiple-processors-and-multiple-cores.html (Accessed 30 August 2023).
[30] W. M. F. Abdel-Rehim, I. A. Ismail, and E. Morsy, "Testing randomness: the original poker approach acceleration using parallel MATLAB with OpenMP", Computer Science and Engineering, vol. 5, no. 2, pp. 25–29, 2015.