ISSN: 2229-6956(ONLINE)
DOI: 10.21917/ijsc.2013.0082

SOLVING FLOWSHOP SCHEDULING PROBLEMS USING A DISCRETE AFRICAN WILD DOG ALGORITHM

M. K. Marichelvam[1] and M. Geetha[2]
[1]*Department of Mechanical Engineering, Kamaraj College of Engineering and Technology, India*
E-mail: mkmarichelvamme@ gmail.com
[2]*Department of Mathematics, Kamaraj College of Engineering and Technology, India*
E-mail: geethamkm1312@ gmail.com

Abstract
The problem of m-machine permutation flowshop scheduling is considered in this paper. The objective is to minimize the makespan. The flowshop scheduling problem is a typical combinatorial optimization problem and has been proved to be strongly NP-hard. Hence, several heuristics and meta-heuristics were addressed by the researchers. In this paper, a discrete African wild dog algorithm is applied for solving the flowshop scheduling problems. Computational results using benchmark problems show that the proposed algorithm outperforms many other algorithms addressed in the literature.

Keywords:
Scheduling, Flowshop, NP-Hard, Discrete African Wild Dog Algorithm, Makespan

1. INTRODUCTION AND LITERATURE REVIEW

Scheduling plays vital role in several industries. Effective scheduling techniques should be required for improving the efficiency of industries. Scheduling may be defined as a process of allocating resources over time to perform a collection of tasks. Different types of scheduling problems were addressed in the literature. This paper considers a flowshop scheduling problem. The flowshop scheduling problem is one of the most important scheduling problems. Many manufacturing systems and assembly lines resemble the flowshop scheduling environment. In the flowshop, a set of n jobs are to be processed in an identical order in a given set of machines. The flowshop scheduling model was first developed by Johnson [1]. Johnson developed an exact algorithm to minimize the makespan for 2-machines flowshop scheduling problems. The flowshop scheduling problem has been proved to be NP-hard [2]. Due to the complexity of the problem, it is difficult to develop exact methods to solve this problem. Hence, researchers proposed different heuristics and metaheuristics to solve the flowshop scheduling problems. The important heuristics were developed by Palmer [3], Campbell et al. [4], Nawaz et al. [5]. King and Spachis [6] and Rajendran and Chaudhri [7] also proposed some heuristics to solve the flowshop scheduling problems.

Recently, researchers adapted different metaheuristics to solve the flowshop scheduling problems. A genetic algorithm (GA) was applied to solve the flowshop scheduling problems by Reeves [8]. Murata et al. [9] solved the flowshop scheduling problems using the GA. Nowicki and Smutnicki [10] applied the tabu search (TS) algorithm for solve flowshop scheduling problems with parallel machines. Reza Hejazi and Saghafian [11] also addressed a review on the flowshop scheduling problems with the makespan criterion. Ruiz and Maroto [12] presented a comprehensive review of the different heuristics used to solve the flowshop scheduling problems. They also evaluated the performance of the different heuristics. A differential evolution algorithm was addressed to solve the flowshop scheduling problems to minimize the makespan by Onwubolu and Davendra [13]. Ching et al. [14] addressed a discrete version of particle swarm optimization (PSO) algorithm for solving the flowshop scheduling problems. Liu et al. [15] presented an effective hybrid particle swarm optimization algorithm for solving the no-wait flowshop scheduling problem with makespan criterion. A simple and effective iterated greedy algorithm was suggested for the permutation flowshop scheduling problem by Ruiz and Stutzle [16]. Ying and Lin [17] proposed an ant colony system heuristic for solving the non-permutation flowshop scheduling problems. A greedy heuristic algorithm was addressed by Baraz and Mosheiov [18] to minimize the makespan for no-idle flowshop scheduling problems. Pan et al. [19] developed a hybrid discrete particle swarm optimization algorithm for solving the no-wait flowshop scheduling problem with makespan criterion. Qian et al. [20] proposed a differential evolution (DE) algorithm to solve the flowshop scheduling problems to minimize the makespan. Jarboui et al. [21] proposed a hybrid GA to solve the flowshop scheduling problems. Akhshabi et al. [22] proposed a parallel genetic algorithm to minimize the makespan of flowshop scheduling problems. Marichelvam [23] applied the cuckoo search algorithm to minimize the makespan in flowshop scheduling environment. African wild dog algorithm is a recently developed metaheuristic algorithm proposed for solving continuous optimization problems [24].

The rest of the paper is organized as follows. The problem is defined in section 2. The proposed algorithm is presented in section 3. Finally, the conclusions and future research opportunities are discussed in section 4.

2. PROBLEM DEFINITION

The problem is defined as follows. Let us consider a set of n jobs to be processed on m machines to minimize the makespan. Makespan is the completion time of the last job in the production system. Makespan is important for measuring the system utilization.

2.1 MATHEMATICAL MODEL

The problem is mathematically formulated as follows. The following notations are used in this paper.

- a Mean Euclidian distance of all dogs
- b Euclidian distance between dogs d and D
- c Step reduction parameter

C_{ij} Completion time of job *j* on machine *i*
C_{im} Completion time of job *j* on machine *m*
C_{max} Makespan
i Machine index
j Job index
I Number of iterations
m Number of machines
n Number of jobs
N Number of wild dogs
PT_{ij} Processing time of job *j* on machine *i*
rand Random number
R_j Ready time of job *j*

$$\min \ C_{\max} \tag{1}$$

Subject to:

$$C_{\max} \geq C_{im} \text{ for all } i, \tag{2}$$

$$C_{ij} = S_{ij} + PT_{ij} \text{ for all } i \text{ and } j, \tag{3}$$

$$S_{ij} \geq R_j \text{ for all } i, \tag{4}$$

$$C_{ij} \geq C_{i,j-1} + PT_{ij} \text{ for all } i, \tag{5}$$

$$C_{ij} \geq 0 \text{ for all } i, j. \tag{6}$$

2.2 ASSUMPTIONS

1) The number of jobs and the number of machines are known in advance.
2) The jobs, their processing times are known in advance and are fixed.
3) All the jobs and the machines are available at time zero.
4) No preemption is allowed.
5) The setup and transportation times of the jobs are independent of the sequence and are included in the processing times.
6) Each machine can process only one job at a time.
7) Each job is being processed on one machine at a time.
8) All the machines are available for the entire period of scheduling.
9) The operating sequences of the jobs are the same on every machine.

3. AFRICAN WILD DOG ALGORITHM

In the past two decades researchers have addressed several metaheuristics to solve a wide variety of optimization problems. African wild dog animal algorithm (AWDA) is a recent, population-based metaheuristic optimization algorithm developed by Subramanian et al. [24] in 2012 for solving continuous optimization problems. The main advantage of the AWDA is it requires only two parameters. But, other metaheuristics consists of several parameters. The AWDA is conceptualized using the communal hunting behavior of African wild dogs. In general, the African wild dogs live in groups. Each group consists of upto 20 adults and their dependent young. Communal hunting is one of the most prominent aspects of the behavior of social carnivores. The studies of carnivore ecology suggested that communal hunting might favour sociality, either by increasing the size of prey that could be killed or by improving hunting success. One can see the coordination between the members of an African wild dog group throughout the hunting process. The effectiveness of hunting depends on the number of cooperating hunters. This communal hunting behavior is similar to the optimization process. The location of each dog compared to the prey determines its chance of catching the prey. Similarly, the objective function value is determined by the set of values assigned to each decision variable. The new Wild dog algorithm is developed based on a model of cooperative hunting of animals when searching for food.

The African wild dog algorithm consists of the following steps.

Step 1: Define the optimization problem and the parameters.

Step 2: Randomly initialize the wild dog pack.

Step 3: Evaluate the fitness of all wild dogs.

Step 4: Coordinated movement of wild dog pack.

Step 5: Repeat Steps 3 and 4 until the termination criterion is satisfied.

The steps are described in the following sections.

3.1 DEFINE THE PROBLEM AND PARAMETERS

In this step we define the objective function. The objective function is the minimization of makespan. The AWDA consists of only two parameters. The parameters are the number of wild dogs (N) and the stopping criterion. The stopping criterion is the number of iterations (I). The parameters are shown in Table.1.

Table.1. Parameters of the AWDA

Parameters	Value
Number of wild dogs (N)	20
Number of iterations (I)	500

3.2 INITIALIZE THE WILD DOG PACK

In the steps, the position vectors for the wild dogs are generated. In general, the position vectors are uniformly distributed in between [0-1].

3.3 FITNESS FUNCTION EVALUATION

Based on the position values, the objective function values are calculated. Then the fitness values are also calculated.

3.4 COORDINATED MOVEMENT OF WILD DOG PACK

The dog *d* will move to the new position d_{i+1} towards another dog *D* whose fitness function value is higher than that of *d*. This step in the AWDA is similar to the PSO algorithm. The new positions are calculated as follows,

$$d_{i+1} = d_i + rand \times (d_i - D_j) \times c \times (a/b) \tag{7}$$

3.5 TERMONATION CRITERION

Most of the researchers use the number of iterations as the termination criterion. In this paper, we also adopt the number of iterations as the termination criterion.

3.6 DISCRETE AFRICAN WILD DOG ALGORITHM

Preliminary studies by Subramanian et al. [24] suggested that the AWDA is very promising for solving continuous optimization problems. However, most of the combinatorial optimization problems are discrete in nature. In order to enable the continuous AWDA to be applied to the discrete scheduling problems, we apply the smallest position value (SPV) proposed by Bean [25]. The SPV rule is presented to convert the continuous position values to a discrete job permutation.

The solution representation is explained in the following section.

3.6.1 Solution Representation :

The vector $X_i^t = \left(X_{i1}^t, X_{i2}^t, ..., X_{in}^t\right)$ represents the continuous position values of the dogs in the search space. The SPV rule is used to convert the continuous position values of the dogs to the discrete job permutation. The solution representation for a 4 job problem is described in Table.2.

Table.2. Solution representation in AWDA

	Dimension *j*			
	1	**2**	**3**	**4**
X_{ij}	0.21	0.16	0.38	0.32
Jobs	2	1	4	3

The smallest position value is $X_{i2}^t = 0.16$ and the dimension j = 2 is assigned to be the first job in the permutation according to the SPV rule. The second smallest position value is $X_{i1}^t = 0.21$ and the dimension j = 1 is assigned to be the second job in the permutation. Similarly, all the jobs are assigned in the permutation.

3.7 COMPUTATIONAL RESULTS

To test the performance of the proposed algorithm, we consider benchmark problems addressed by Reeves [8] and Carlier [26]. We compare the results of the proposed algorithm with many heuristics addressed in the literature [3-5]. The result comparison is presented in Table.3.

Table.3. Result comparison of the benchmark problems

Problem	Problem size		C_{max}			
	n	***m***	**Palmer [3]**	**CDS [4]**	**NEH [5]**	**AWDAA**
Car1	11	5	7472	7202	7038	7038
Car2	13	4	7940	7410	7376	7166
Car3	12	5	7725	7399	7443	7302
Car4	14	4	8423	8423	8003	8003
Car5	10	6	8520	8627	8090	7792
Car6	8	9	9487	9553	9079	8705
Car7	7	7	7639	6819	7468	6590
Car8	8	8	9023	8903	8967	8366
Rec01	20	5	1391	1399	1334	1248
Rec03	20	5	1223	1273	1136	1116
Rec05	20	5	1290	1338	1294	1264
Rec07	20	10	1715	1697	1637	1572
Rec09	20	10	1915	1639	1692	1543
Rec11	20	10	1685	1597	1635	1546

4. CONCLUSION

This paper has addressed the flowshop scheduling problems with makespan criterion. For this NP-hard problem we present the African wild dog algorithm. To the best of our knowledge, this is the first attempt to solve the flowshop scheduling problems using the African wild dog algorithm. We tested the performance of the proposed African wild dog algorithm using the benchmark problem addressed in the literature. Computational results show that the proposed algorithm provides better results than many other heuristics addressed in the literature. The proposed algorithm may be used in hybrid with other heuristics and metaheuristics. It would be interesting to apply the proposed algorithm to solve other scheduling problems with single or multiple objective functions.

REFERENCES

[1] S. M. Johnson, "Optimal two- and three-stage production schedules with setup times included", *Naval Research Logistics Quarterly*, Vol. 1, No. 1, pp. 61 – 68, 1954.

[2] M. Cheng, S. Sun and L. M. He, "Flow shop scheduling problems with deteriorating jobs on no-idle dominant machines", *European Journal of Operational Research*, Vol. 183, No. 1, pp. 115 – 124, 2007.

[3] D. S. Palmer, "Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining a near optimum", *Journal of the Operational Research Society*, Vol. 16, pp. 101 – 107, 1965.

[4] H. G. Campbell, R. A. Dudek and M. L. Smith, "A heuristic algorithm for the n-job, m-machine sequencing problem", *Management Science*, Vol. 16, No. 10, pp. 630 – 637, 1970.

[5] M. Nawaz, Jr. E. Enscore and I. Ham, "A Heuristic Algorithm for the m-Machine, n-job flow-shop sequencing problem", *OMEGA - Elsevier*, Vol. 11, No. 1, pp. 91 – 95, 1983.

[6] J. R. King and A. S. Spachis, "Heuristics for flow-shop scheduling", *International Journal of Production Research*, Vol. 18, No. 3, pp. 345 – 357, 1980.

[7] C. Rajendran and D. Chaudhri, "Heuristic algorithm for continuous flow-shop problem", *Naval Research Logistics*, Vol. 37, No. 5, pp. 695 – 705, 1990.

[8] C. R. Reeves, "A genetic algorithm for flowshop sequencing", *Computers and Operations Research*, Vol. 22, No. 1, pp. 5 – 13, 1995.

[9] T. Murata, H. Ishibuchi and H. Tanaka, "Genetic algorithms for flow shop scheduling problems", *Computers and Industrial Engineering*, Vol. 30, No. 4, pp. 1061 – 1071, 1996.

[10] E. Nowicki and C. Smutnicki, "The flow shop with parallel machines: A tabu search approach", *European Journal of Operational Research*, Vol. 106, No. 2 - 3, pp. 226 – 253, 1998.

[11] S. Reza Hejazi and S. Saghafian, "Flowshop-scheduling problems with makespan criterion: a review", *International*

Journal of Production Research, Vol. 43, No. 14, pp. 2895 – 2929, 2005.

[12] R. Ruiz and C. Maroto, "A comprehensive review and evaluation of flowshop heuristics", *European Journal of Operational Research*, Vol. 165, No. 2, pp. 479 – 494, 2005.

[13] G. Onwubolu and D. Davendra, "Scheduling flow shops using differential evolution algorithm", *European Journal of Operational Research*, Vol. 171, No. 2, pp. 674 – 692, 2006.

[14] J. L. Ching, T. Chao-Tang and Pin Luarn, "A discrete version of particle swarm optimization for flow shop scheduling problems", *Computers and Operations Research*, Vol. 34, No. 10, pp. 3099 – 3111, 2007.

[15] B. Liu, L. Wang and Y. H. Jin, "An effective hybrid particle swarm optimization for no-wait flow shop scheduling", *International Journal of Advanced Manufacturing Technology*, Vol. 31, No. 9 – 10, pp. 1001 – 1011, 2007.

[16] R. Ruiz and T. Stutzle, "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem", *European Journal of Operational Research*, Vol. 177, No. 3, pp. 2033 – 2049, 2007.

[17] K. C. Ying and S. W. Lin, "Multi-heuristic desirability ant colony system heuristic for non-permutation flowshop scheduling problems", *International Journal of Advanced Manufacturing Technology*, Vol. 33, No. 7 - 8, pp. 793 – 802, 2007.

[18] D. Baraz and G. Mosheiov, "A note on a greedy heuristic for the flow-shop makespan minimization with no machine idle-time", *European Journal of Operational Research*, Vol. 184, No. 2, pp. 810 – 813, 2008.

[19] Q. K. Pan, L. Wang, M. F. Tasgetiren and B. H. Zhao, "A hybrid discrete particle swarm optimization algorithm for the no-wait flow shop scheduling problem with makespan criterion", *International Journal of Advanced Manufacturing Technology*, Vol. 38, No. 3 – 4, pp. 337 – 347, 2008.

[20] B. Qian, L. Wang, R. Hu, D. X. Huang and X. Wang, "A DE-based approach to no-wait flow-shop scheduling", *Computers and Industrial Engineering Journal*, Vol. 57, No. 3, pp. 787 – 809, 2009.

[21] B. Jarboui, M. Eddaly and P. Siarry, "A hybrid genetic algorithm for solving no-wait flowshop scheduling problems", *International Journal of Advanced Manufacturing Technology*, Vol. 54, No. 9 – 12, pp. 1129 – 1143, 2011.

[22] M. Akhshabi, J. Haddadnia and M. Akhshabi, "Solving flow shop scheduling problem using a parallel genetic algorithm", *Procedia Technology*, Vol. 1, pp. 351 – 355, 2012.

[23] C. Subramanian, K. Subramanian and A. S. S. Sekar, "A New Parameter Free Meta-Heuristic Algorithm for Continuous Optimization: African Wild Dog Algorithm", *European Journal of Scientific Research*, Vol. 92, No. 3, pp. 348 – 356, 2012.

[24] M. K. Marichelvam, "An Improved Hybrid Cuckoo Search (IHCS) metaheuristics algorithm for permutation flow shop scheduling problems", *International Journal of Bio-inspired Computation*, Vol. 4, No. 4, pp. 200 – 205, 2012.

[25] J. C. Bean, "Genetic algorithms and random keys for sequencing and optimization", *INFORMS Journal of Computing*, Vol. 6, No. 2, pp.154 – 160, 1994.

[26] J. Carlier, "Ordonnancements a contraintes disjonctives", *RAIRO Operational Research*, Vol. 12, No. 4, pp. 333 – 351, 1978.