Indian J.Sci.Res. 7(2) : 167-171, 2017
ISSN : 0976-2876 (Print)
ISSN : 2250-0138 (Online)

# FUZZY FLOW SHOP SCHEDULING USING GREY WOLF OPTIMIZATION ALGORITHM

**KAWAL JEET**[1]

Department of Computer Science, D.A.V. College, Jalandhar, Punjab, India

## ABSTRACT

**Flow shop scheduling is one of the most important optimization problem for which a number of heuristic and metaheuristic techniques have been successfully applied. In this paper, the problem of two machine flow shop scheduling taking into consideration makespan and idle time of both machines has been considered. The numbers of existing approaches are not found to be purely applicable to real time situations due to uncertainties involved. In order to handle such situations, the processing time and setup times of jobs is taken under fuzzy environment. Nature-Inspired Grey Wolf Optimization algorithm has been used to resolve this problem. The results thus obtained are compared to existing heuristic approach as well as other metaheuristic approaches such as multi-objective Genetic Algorithm, Particle Swarm Optimization and NSGA-II algorithm. It is observed that Grey Wolf based approach performs better than other approaches and existing heuristic approach for the problem under consideration.**

**KEYWORDS :** Average High Ranking, Fuzzy Membership Function, Idle Time, Makespan, Nature-Inspired Algorithm

Human beings are always inspired by nature and this is evident from the recent technological developments in various fields of science and engineering. Over the past couple of decades, a large number of complex research problems have found their solutions in nature-inspired algorithms. In this paper, one of the recent nature-inspired algorithm namely Multi-Objective Grey Wolf (MOGWO) algorithm has been investigated for scheduling of a number of jobs on two machines with sequence dependent set up times.

Numbers of existing approaches are not found to be purely applicable to real time situations due to uncertainties involved. In order to handle such situations, the concept of fuzzy environment has been appended with the theory of scheduling. The processing time and setup time of jobs are taken under fuzzy environment. In order to obtain job schedules, the effective processing time and setup time of the jobs are calculated by using Yager's average high ranking formula (Gupta et al., 2013) used this concept of fuzzy based processing time and set up times for scheduling of jobs.

Numbers of meta-heuristic approaches were used for flow shop scheduling in the past decade. Nature has always been a source of inspiration for engineering and researchers. Some of the recent optimization algorithms have been observed to be inspired by nature. Table 1 gives a brief review of some of the studies investigating nature-inspired algorithms for job scheduling along with the criteria and heuristic followed.

## MATERIAL AND METHODS

### Description of Fuzzy Flow Shop Scheduling on Two Machines

The problem of scheduling multiple jobs on two potential machines for optimizing makespan, idle time of machines as well as idle time of transporting agent involved is one of the important problems that have been encountered by engineers time and again.

In order to describe the problem, let there be n jobs that can work independent of each other on 2 potential machines. Each job is associated with fuzzy processing time and fuzzy set up time along with transporting time form machine M1 to M2

### Notations

Following notations are used for the formulation of problem of job scheduling on parallel machines.

n: Number of jobs to be scheduled

m: Number of potential machines (2 in this case)

i: Job under consideration

Aij: Fuzzy processing time of ith job on jth machine where i =1,2,.....,n and j=1, 2.

h(Aij): Average high rank value of processing time of ith job on jth machine where i =1,2,.....,n and j=1, 2

Sij: Fuzzy set up time of ith job on jth machine where i =1,2,.....,n and j=1, 2

h(Sij): Average high rank value of set up time of ith job on jth machine where i =1,2,.....,n and j=1, 2

ti: Transport time for transporting agent from machine M1

[1]Corresponding author

**Table 1 : Literature Review of Nature-Inspired Algorithms for Job Scheduling**

| **Research Work** | **Criteria Optimized** | **Heuristic Followed** |
|---|---|---|
| (Karthikeyan et al., 2015) | The maximum completion time, the workload of the critical machine and the total workload of all machines | Discrete firefly algorithm (DFA) is combined with local search (LS) method. |
| (Niu et al., 2013) | Makespan, tardiness and mean flow time of the schedules | Intelligent Water Drops |
| (Komaki & Kayvanfar, 2015) | Completion time of the last processed job | Grey Wolf Optimization algorithm |
| (Lin & Ying, 2013) | Makespan and total Flowtime | Simulated Annealing |
| (Hecker et al., 2014) | Makespan, total idle time of all the machines | Modified GA, ACO and a random search procedure |

to M2 for ith job

ri: Return time for transporting agent from machine M2 to M1 for ith job

IM1: Idle time for machine M1

IM2: Idle time for machine M2

IR : Idle time for transporting agent.

Ctotal: Makespan i.e.Total production run time for completing all n jobs=Time of completion of last scheduled job on machine M2).

**Objective/Fitness Function: IM1, IM2, IR, Ctotal: Minimize**

In order to identify optimal sequence of jobs to be scheduled on 2 machines, Grey Wolf Optimization algorithm has been used. Fig.I describes the implementing of MOGWO for scheduling of n jobs on 2 sequential machines.

**Samples Under Study**

The upper and lower bound along with other details required for generating random samples are given in Table 2. The approaches under consideration are search based and are highly randomized, so these are repeated 30 times and best job sequence is taken as the output. In order to encode the problem to be used as an optimization problem to be solved using GWO, population of possible job sequences is required. No standard dataset is available to be used for the problem under consideration (involving fuzzy inputs). So, in order to initialize the population, random job sequences are generated. The parameters involved for possible initialization are shown in Table 3.

```
//Processing time for job i (n) to be executed on machine j (2)
//Since input is fuzzy and triangular membership function has been considered in this paper, so 3 (k) values pertaining to each processing time are to be generated.
    for j=1 to 2 //for each machine
    for i= 1 to n //for each job
    temp_ij=round(rand(1,n) .* (pro_u-pro_l) + pro_l);
    for k=1:3
    r_limit=round(rand(1,n) .* (limit-1) + 1);
    if(k==1)
    A_ij(:,k)= temp_ij -r_limit;
    end
    if(k==2)
    A_ij (:,k)= temp_ij;
    end
    if(k==3)
    A_ij(:,k)= temp_ij +r_limit;
    end
    end
    end
    end
```

**Identifying Optimal Schedule**

In order to obtain the optimal schedule for n jobs to be scheduled on 2 sequential machines in the fuzzy environment (Gupta, et al., 2013) including the transportation time ti and with return time ri of transporting agent is obtained by sequencing the jobs (i-1), i and (i+1) such that $\min(h(A_{i1})+r_{i-1}+t_i+h(S_{(i-1)1}), h(A_{(i+1)2})+r_i+t_{(i+1)}+h(S_{i2})) < \min(h(A_{(i+1)1})+r_i+t_{(i+1)}+h(S_{i1}), h(A_{i2})+r_{(i-1)}+t_i+h(S_{(i-1)2}))$

**Step 1 [Initialize Population]:**

**Step 1.1:** Encode and initialize the population of possible job sequences. Each individual is called a *grey wolf*.

**Step 1.2:** Set parameters as shown in Table III. Randomly generate fuzzy processing time A, fuzzy start up time *S*, transport time *t* and return time *r* for each job. Obtain h(A) and h(S) for each job to be executed on each machine. Select current best ($x_\alpha$), second best ($x_\beta$) and third best ($x_\gamma$) non-dominated clustering.

**Step 1.3:** Evaluate fitness of each candidate in the population using minimum ideal time for machine M1, M2, transporting agent and makespan ($C_{total}$) for *n* jobs.

**Step 1.4:** Store the clustering that represent non -dominated vectors in the temporary repository named REP.

**Step 1.5:** Generate hyper-cubes to locate and maintain best solutions.

Repeat steps 2 to 5 until the stopping criteria is met (as shown in Table III)

**Step 2 [Identify new possible solutions]:** For each job sequence $x_i$ repeat

$a=2-1*2(1/t)$ — $t$ is the current iteration.

$A_i=2*a*r_1-a$ — $r_1$ and $r_2$ are random numbers between 0 and 1.

$C_i=2*r_2$ — $A_i$ and $C_i$ are coefficient vectors where $i$=1,2 and 3.

$D_\alpha=C_1* x_\alpha - x_i(t)$

$D_\beta=C_2* x_\beta - x_i(t)$

$D_\gamma=C_3* x_\gamma - x_i(t)$

$x_{i1}(t)=x_\alpha -A_1* D_\alpha$

$x_{i2}(t)=x_\beta -A_2* D_\beta$

$x_{i3}(t)= x_\gamma -A_3* D_\gamma$

$x_i(t+1)=( x_{i1}(t)+ x_{i2}(t)+ x_{i3}(t))/3$

**Step 3 [Search for a better Solution]:** If $x_i(t+1)$ is better than $x_i(t)$ (taking into consideration the non-dominance of the clustering), replace $x_i(t)$ with $x_i(t+1)$.

**Step 4 [Update best solutions]:** Update hyper-cubes and REP to maintain current non -dominated clustering. Update $x_\alpha$, $x_\beta$ and $x_\gamma$.

**Step 5 [Output]:** Return REP which includes resulting non-dominated clustering

**Figure 1 : Multi-Objective Auxiliary Archive Based Grey Wolf Optimization Algorithm for Job Scheduling**

**Table 2 : Detail of Parameters Required to Generate Random Samples**

| Parameter | Value |
|---|---|
| Number of Jobs | 10, 20, 30, 40, 50 |
| Number of Machines | 2 |
| Upper bound for processing time (pro_u) | 80 |
| Lower bound for processing time ( pro_l) | 20 |
| Upper bound for start up time (start_u) | 2 |
| Lower bound for start up time (start_l) | 10 |
| Limit | 5 |

**Table 3 : Common Control Parameters Defined for Implementing MOGWO Algorithm for Job Scheduling**

| Parameter | Value | Description |
|---|---|---|
| Number of variables to be optimized (*n*) | Number of jobs to be scheduled | The value of $i^{th}$ variable in the candidate job schedules indicates the job which is scheduled at $i^{th}$ order in job schedule |
| Population size (*Pop*) | 5**n* | Manually tested by repeated executions of the algorithms. |
| Population | Candidate Job Sequences | Randomly generated |
| Generations | 10 * *n* or when value of objectives does not change for 200 consecutive iterations | Stopping criteria |

**Table 4 : Solution With Least Total Cost for Randomly Generated Samples**

| Algorithm | | Number of Jobs | $C_{total}$ | IM1 | IM2 | IR | Total cost ($C_{total}$+IM1+IM2+IR) |
|---|---|---|---|---|---|---|---|
| Proposed Meta-Heuristic | MOGWO (Mirjalili, et al., 2016) | 10 | 568.6667 | 0 | 87.66667 | 481 | **1137.333** |
| | | 20 | 1268 | 0 | 281 | 1177.667 | **2726.667** |
| | | 30 | 1725.667 | 0 | 1785 | 1737 | 5247.667 |
| | | 40 | 2433.333 | 0 | 331.6667 | 2195 | **4960** |
| | | 50 | 2952.333 | 0 | 81.33333 | 2515.667 | **5549.333** |
| | | | | | | | |
| Existing Meta-Heuristics | MOGA (Deb, 2001) | 10 | 595.6667 | 0 | 86 | 475 | 1156.667 |
| | | 20 | 1267 | 0 | 289.6667 | 1174.333 | 2731 |
| | | 30 | 1722.667 | 0 | 1672 | 1739 | **5133.667** |
| | | 40 | 2435.333 | 0 | 374.6667 | 2194.333 | 5004.333 |
| | | 50 | 2973.333 | 0 | 78 | 2518.667 | 5570 |
| | | | | | | | |
| | MOPSO (Coello, et al., 2004) | 10 | 564.6667 | 0 | 106.6667 | 483 | 1154.333 |
| | | 20 | 1269 | 0 | 296.3333 | 1174.667 | 2740 |
| | | 30 | 1722 | 0 | 1683.333 | 1735 | 5140.333 |
| | | 40 | 2425.333 | 0 | 383.6667 | 2189 | 4998 |
| | | 50 | 2962 | 0 | 76.66667 | 2514.667 | 5553.333 |
| | | | | | | | |
| | NSGA-II (Deb, et al., 2000) | 10 | 595.6667 | 0 | 85.33333 | 475 | 1156 |
| | | 20 | 1217.333 | 0 | 567.3333 | 1166.667 | 2951.333 |
| | | 30 | 1692.333 | 0 | 2379 | 1734 | 5805.333 |
| | | 40 | 2544.333 | 0 | 31.33333 | 2094 | 4669.667 |
| | | 50 | 2984.667 | 0 | 82.33333 | 2519.667 | 5586.667 |
| | | | | | | | |
| Existing Heuristics | Gupta et al. (Gupta, et al., 2013) | 10 | 556.3333 | 0 | 139.3333 | 477 | 1172.667 |
| | | 20 | 1220.333 | 0 | 495.3333 | 1170.667 | 2886.333 |
| | | 30 | 1667 | 0 | 2818.333 | 1735.667 | 6221 |
| | | 40 | 2376.667 | 0 | 649 | 2190.333 | 5216 |
| | | 50 | 2950 | 0 | 122.6667 | 2513.667 | 5586.333 |

## RESULTS AND DISCUSSION

In order to validate the sequence obtained by the application of MOGWO, Total cost (sum of IM1, IM2, IR, $C_{total}$), has been used as assessment criteria. The proposed methodology is multi-objective in nature, so it leads to Pareto front as an output. Since, Pareto front consists of non-dominated possible solutions, where none of the solutions in Pareto front dominates others, so Total cost of the solution is used as criteria to assess the quality of job sequence. Lower the value of this cost better is the job sequence solution. The results thus obtained are shown in Table 4. Application of MOGWO for the problem under consideration has been compared to existing MOPSO, MOGA and NSGAII algorithms for job scheduling. It is observed that MOGWO better optimises scheduling of jobs on two machines sequentially when compared to other counterparts in 4 out of 5 sample cases.

## CONCLUSION

Scheduling jobs on two machines optimizing makespan, idle time of both machines as well as transporting agent is one of the issues that has been encountered in various engineering and manufacturing problems and has found a large number of applications. In this paper, Grey Wolf Optimization algorithm has been used for solving this job scheduling problem. Comparison of proposed approach to that of existing Multi-Objective Genetic Algorithm (MOGA), Non-dominated Sorting Genetic

Algorithm-II (NSGA-II) and Multi-objective Particle Swarm Optimization (MOPSO) approaches for the cause of job scheduling on two machines is done. Grey Wolf Optimization algorithm has been empirically found to perform better than other counterparts as well as existing heuristic. The work can be further extended by experimenting with other novel nature-inspired algorithms.

## REFERENCES

Coello C. A. C., Pulido G. T. and Lechuga M. S., 2004. Handling multiple objectives with particle swarm optimization. IEEE Transactions on Evolutionary Computation, 8(3) : 256-279.

Deb K., 2001. Multi-objective optimization using evolutionary algorithms (Vol. 16): John Wiley & Sons.

Deb K., Agrawal S., Pratap A. and Meyarivan T., 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Lecture notes in computer science, 1917, 849-858.

Gupta D., Sharma S., and Aggarwal S., 2013. Flow shop scheduling on 2-machines with setup time and single transport facility under fuzzy environment. Opsearch, **50**(1) :14-24.

Hecker F. T., Stanke M., Becker T., and Hitzmann B., 2014. Application of a modified GA, ACO and a random search procedure to solve the production scheduling of a case study bakery. Expert Systems with Applications, **41**(13) : 5882-5891.

Karthikeyan, S., Asokan, P., Nickolas, S., & Page, T. (2015). A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems. International Journal of Bio-Inspired Computation, **7**(6) : 386-401.

Komaki G. M., and Kayvanfar V., 2015. Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time. Journal of Computational Science, **8** : 109-120.

Lin S.-W. and Ying K.-C., 2013. Minimizing makespan and total flowtime in permutation flowshops by a bi-objective multi-start simulated-annealing algorithm. Computers & Operations Research, **40** (6), 1625-1647.

Mirjalili S., Saremi S., Mirjalili S. M., and Coelho L. d. S., 2016. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. Expert Systems with Applications, **47** : 106-119.

Niu S., Ong S.-K., and Nee A. Y., 2013. An improved intelligent water drops algorithm for solving multi-objective job shop scheduling. Engineering Applications of Artificial Intelligence, **26**(10), 2431-2442.