Minimizing weighted mean flow time in open shop scheduling with time-dependent weights and intermediate storage cost

Seyed Hossein Hashemi Doulabi[1], Amir Ardestani jaafari[2], Mohsen Akbarpour Shirazi[3]

[1] Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran (hashemidoulabi@aut.ac.ir)
[2] Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran (ardestani.amir@aut.ac.ir)
[3] Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran (akbarpour@aut.ac.ir)

ABSTRACT- Open shop scheduling problem is a common scheduling problem and has wide engineering applications in manufacturing. In some industrial cases, open shop scheduling could be a long scheduling problem as an aircraft production. In these cases, time value of money plays a significant role in determining a schedule cost. This paper addresses the problem of minimizing sum of weighted mean flow time and intermediate storage cost in an open shop scheduling environment. The main contribution of this work is to concern time-dependent weights which results in a more realistic insight for decision makers via considering time value of money in long scheduling problems. We have formulated the problem as a mixed integer linear programming model. Due to the nondeterministic polynomial time complexity of the problem, a novel genetic algorithm is also presented to solve the problem in reasonable time. Computational results indicate remarkable improvement of the objective function as compared with the case of having constant weights over scheduling horizon.

Keywords: Open shop scheduling, time value of money, mixed integer linear programming, genetic algorithm.

1. INTRODUCTION

In an open shop scheduling problem, we have a set that consists of m machines and another set that consists of n jobs which must be operated by machine set. We assume that needed time for each operation is given. Another assumption is that each job can be operated by only one machine at a time and also each machine can process only one job at a time.

This problem has wide use in industries. For example consider repairs of a huge airplane that may be include its engines and electrical system. These operations are both necessary but it is impossible to do them simultaneously. Other cases such as: in automobile repair, quality control centers, class assignments, examination scheduling, and satellite communications are described by Kubiak et al. [1], Liu and Bulfin [2] and Prins [3].

Suppose that C_j is the completion time of job J_j, $j= 1, 2, \dots, n$,. In this paper, the objective function is to minimize sum of the weighted completion times which could be referred as $O||\sum(W_jC_j)$.

Most studies in this scope dealt with the minimization total completion time. Gonzalez et al. [4] proposed an algorithm with linear polynomial time complexity for the two-machine open shop problem. The preemptive problem can be solved in polynomial time for an arbitrary number of machines.

For the problem of makespan minimization in open shop scheduling, a number of branch and bound and heuristic algorithms have been devised. It can be said that Dorndorf et al. [5] have presented the most efficient exact algorithm. They have applied constraint propagation methods to discard non optimal solution from the solution space. Brasel et al. [6] presented some heuristic algorithms based on matching algorithms and insertion of operations which are applied into partial schedules combined with a search method. Gueret and Prins [7] have proposed Some other heuristic algorithms.

Among evolutionary algorithms, Alcaide et al. [8] developed a tabu search algorithm for the open shop scheduling to minimize makespan. A hybrid genetic algorithm for solving the open shop problem with makespan minimization was developed by Liaw [9]. Prins [10] has also presented another genetic algorithm which leads to excellent solutions.

In recent years, multi-criteria open shop scheduling problems has attracted researchers. Kyparisis and Koulamas [11] and Gupta et al. [12] have presented some polynomial solvable cases of the two-machine problem with heuristics to search for a schedule with minimum mean flow time subject to minimum total completion time. In some recent works on multi-criteria scheduling problems one objective is to minimize of mean flow time. On the other hand, there exist only a small number of works whose objective function is only minimization of weighted mean flow time.

A common assumption of works in the literature is that weights are constant over planning horizon and not time dependent. The idea that money available at the present time is worth more than the same amount in the future due to its potential earning capacity is core principle of benefit-cost analysis. Since w_j in the proposed objective function is an estimation for worth of *j* th job, it would be more realistic to consider weights as time dependent parameters. In this paper, we addresses the problem of minimizing summation of time-dependent weighted mean flow time with intermediate storage cost in open shop environments. This problem is formulated as a

mixed integer linear programming model. Due to the NP-hard complexity of the problem, exact methods are only efficient to solve small instances. So, an effective genetic algorithm is proposed to solve the problem. Computational results demonstrate that the proposed algorithm obtains high quality solutions in reasonable time.

The rest of this paper is organized as follows. In Section 2, the mixed integer linear programming model is presented. We elaborate on the proposed genetic algorithm in Section 3. Section 4 includes computational results. Section 5 concludes the paper.

2. The mixed integer linear formulation

Sets, indices, parameters and variables used in the mathematical model are as follows:

Sets and indices:
J: Jobs set.
I: Machines set.
t, t': Time indices.
j: Job indices.
i: Machine indices.

Parameters:
n: Number of jobs.
m: Number of machines.
P_{ij}: Processing time job *j* on machine *i*.
U_t: Planning horizon.
W_{jt}: Completion weight job j if all of its processes finishes at time *t*.
H_{tj}: Intermediate storage cost job *j* at time *t* if this job is free at the time.
M: A big value which is determined regarding to other parameters of the problem.

Variables:
WFT_j: Weighted flow time job *j*.
λ_{tj}: Intermediate storage cost job *j* at time *t*.
X_{ijt}: Equals 1 if machine i is processing job *j* at time *t* and 0 otherwise.
C_{ij}: Completion time job *j* on machine *i*.
C_j: Total completion time job *j*.
Z_{jt}: Equals 1 if job *j* is completed at time *t* and 0 otherwise.
λ_{tj}: Intermediate storage cost job *j* at time *t*.

Using these definitions, the proposed mathematical model is as follows:

$$MIN(Z) = \sum_{j=1}^{n} WFT_j + \sum_{t=1}^{U_t}\sum_{j=1}^{n} \lambda_{tj} \quad (1)$$

$$\sum_{j=1}^{n} X_{ijt} \le 1 \qquad \forall i \in I, \forall t \in \{1,2,\dots,U_t\} \quad (2)$$

$$\sum_{i=1}^{m} X_{ijt} \le 1 \qquad \forall j \in J, \forall t \in \{1,2,\dots,U_t\} \quad (3)$$

$$\sum_{t=1}^{U_t} X_{ijt} = P_{ij} \qquad \forall j \in J, \forall i \in I \quad (4)$$

$$\sum_{t=1}^{U_t} \left| X_{ij(t+1)} - X_{ijt} \right| \le 2 \qquad \forall j \in J, \forall i \in I \quad (5)$$

$$C_{ij} = \frac{\sum_{t=1}^{U_t} t \times X_{ijt}}{P_{ij}} + \frac{P_{ij}}{2} \qquad \forall j \in J, \forall i \in I \quad (6)$$

$$C_j \ge C_{ij} \qquad \forall j \in J, \forall i \in I \quad (7)$$

$$C_j = \sum_{t=1}^{U_t} t Z_{jt} \qquad \forall j \in J \quad (8)$$

$$\sum_{t=1}^{U_t} Z_{jt} = 1 \qquad \forall j \in J \quad (9)$$

$$WFT_j \ge W_{jt} C_j - \left(1 - Z_{jt}\right) M \qquad \forall j \in J, \forall i \in I \quad (10)$$

$$\lambda_{tj} \ge H_{tj}\left(1 - \sum_{i=1}^{m} X_{ijt}\right) - M \times \sum_{t'=1}^{t-1} Z_{jt'}$$
$$\forall j \in J, \forall t \in \{1,2,\dots,U_t\} \quad (11)$$

$$X_{ijt}, Z_{jt} \in \{0,1\} \quad \forall i \in I, \forall j \in J, \forall t \in \{1,2,\dots,U_t\} \quad (12)$$

Equation (1) remarks that objective function includes weighted flow time and intermediate storage cost. Constraint (2) implies that on each time *t*, machine *i* can process at most one job. Similarly, Constraint (3) states that on each time *t*, at most one machine could process job *j*. Constraint (4) notes that processing time job *j* on machine *i* equals P_{ij}. Regarding constraint (5), jobs are forced to be processed consecutively. This constraint is not a linear constraint, but it could be replaced by the two following linear constraints:

$$\sum_{t=1}^{U_t} Y_{ijt} \le 2 \qquad \forall j \in J, \forall i \in I \quad (5')$$

$$Y_{ijt} \ge X_{ij(t+1)} - X_{ijt}$$
$$\forall i \in I, \forall j \in J, \forall t \in \{1,2,\dots,U_t\} \quad (5'')$$

$$Y_{ijt} \in \{0,1\} \qquad \forall i \in I, \forall j \in J, \forall t \in \{1,2,\dots,U_t\} \quad (5''')$$

In Constraint (6), we have formulated C_{ij} by decision variables X_{ijt}. Constraint (7) states that total completion time job j is greater than completion job j on machine i. Constraints (8) and (9) force Z_{jt}s to take proper values. Constraint (10) implies that if job j is completed at time t ($Z_{jt} = 1$), WFT_j must be greater than time-dependent weighted flow time ($WFT_j \geq W_{jt}C_j$). In this case, regarding objective function presented by Equation (1), model will set $WFT_j = W_{jt}C_j$. In case of having $Z_{jt} = 0$, this constraint will be ignored by the model ($WFT_j \geq -\infty$). Intermediate storage cost is formulated using Constraint (11). Regarding this constraint, if job j is free at time t ($\sum_{i=1}^{m} X_{ijt} = 0$) and it is not completed yet ($\sum_{t'=1}^{t-1} Z_{jt'} = 0$), we will have $\lambda_{tj} \geq H_{tj}$. In this case, the mathematical model will consider $\lambda_{tj} = H_{tj}$ to minimize the objective function. In other case ($\sum_{i=1}^{m} X_{ijt} = 0 = 1$ or $\sum_{t'=1}^{t-1} Z_{jt'} = 1$), Constraint (11) will be transformed to $\lambda_{tj} \geq 0$ or $\lambda_{tj} \geq -\infty$.

3. The proposed genetic algorithm

First time, Holland [14] presented Genetic algorithms (GAs). They wanted to solve industrial problems which were to complex to be solved using the exact algorithms available at that time. Today, GAs are used as one of the usual metaheuristic algorithms for solving optimization problems. The main idea in GAs originated from Darwin's survival theory which states it is likely that good parents produce better offspring.

This algorithm searches a problem space with a population of solution called chromosomes. Each chromosome has a fitness value according to its quality. A set of search operators are used to find new chromosomes in solution space until some stopping criteria happen. In one generation of a typical GA, the best chromosomes of the current population are reproduced in the next generation. A selection mechanism is applied so that the chromosome with the higher fitness value has a more chance to be selected. The selected chromosomes mate and produce new offspring. Then each offspring might be mutated by mutation mechanism. The values of new population objective function is then calculated again and the this process is repeated [15].

The encoding representation applied in the proposed GA is a random key matrix representation. To generate an initial schedule, a matrix with m rows and n columns including random numbers in the range of [0,1] must be produced. In this matrix, each row i is representative of machine i and each column j is representative of job j. A sample of this matrix for a problem with 4 machines and 6 jobs is depicted in Figure 1.

To determine sequence of jobs on each machine i, we entries of row I must be sorted. The job with the smallest value in row i is the first job to be set up on machine i. in the same way, other jobs must be determined to be processed on machine i. In order to decide on the sequences of machines that processes job j, random numbers in column j must be sorted and sequence of machines could be determined likewise. Then having jobs and machine sequences, one could schedule jobs on machines as soon as possible in order to minimize the proposed objective function.

We have used a uniform crossover operator in which for each pair of solution, a random matrix must be generated. If the random number in cell (i, j) of this matrix is smaller than a predetermined value p_c, the corresponding each cell (i, j) paired solutions must replaced by the other. The mutation operator applied in the proposed GA, is a common uniform mutation operator.

	J1	J2	J3	J4	J5	J6
M1	0.09	0.12	0.25	0.35	0.18	0.75
M2	0.53	0.40	0.13	0.19	0.72	0.49
M3	0.86	0.34	0.99	0.48	0.29	0.36
M4	0.32	0.60	0.02	0.85	0.21	0.76

Figure1. A sample of solution representation.

4. Computational results

In this section, we investigate quality of solutions obtained by the proposed genetic algorithm. To do this, For each (n, m) ∈ ({5, 6, 7, 8, 9, 10} × {2, 3, 4, 5}), 5 instances are generated and solved by the proposed algorithm. In generating test problems, for each job j, w_{j1} is chosen in the range of [0.1, 1] and other w_{jt}s ($t \in \{2, \dots, U_t\}$) are considered to be $w_{j1}(1+r)^t$ in which r is the inflation rate assumed to be 0.1 in this paper. For each job j and machine i, P_{ij} is generated in the range of [10, 30].Since we want to concentrate on the effect of time-dependent weights, H_{tj} s are assumed to be equal 0.

Computational results are presented in Table 1.In this table, n and m are number of jobs and machines, respectively. UB is an upper bound of problem obtained by producing random solutions. $\mathrm{SOL_{GA(dep)}}$ is the objective function of test problems with time-dependent weights obtained using the proposed genetic algorithm. A lower bound is also presented in Table 1 (LB) which is calculated by considering $\mathrm{C_j} = \sum_{\mathrm{i=1}}^{\mathrm{m}} \mathrm{P_{ij}}$ and using Equation (1). In order to evaluate quality of obtained schedules, percentage deviation index (PDI) is applied defined by $\mathrm{PDI} = \left(\mathrm{SOL_{dep}} - \mathrm{LB}\right)/(\mathrm{UB} - \mathrm{LB})$. Lower amounts PDI shows that the proposed genetic algorithm is more effective in finding best solutions. Also, to show impact of considering time-dependent weights comparing to the case of having constant weights over planning horizon, we have solved the generated problems for both cases.

In order to have a more rational comparison, constant weight for each job j is considered to be average of time-dependent weights job j over planning horizon ($w_j = \left(\sum_{t=1}^{U_t} w_{jt}\right)/U_t$) and using the proposed genetic algorithm a schedule is obtained for each test problem. Then obtained solutions are evaluated by Equation (1) using time-dependent weights. Amounts of this objective function for test problems, is presented under column SOL_{Cons}. Comparison of these two cases is calculated by $\Delta_{dep,cons} = \left(SOL_{Cons} - SOL_{dep}\right)/SOL_{dep}$. Greater amounts of $\Delta_{dep,cons}$, shows that using time dependent weights have been more effective on the quality of solutions as compared with the constant weights.

Table 1. Computational results

n	m	UB	SOL_{dep}	SOL_{Cons}	LB	PD (%)	$\Delta_{dep,cons}$
5	2	45696	1285	1382	58	2.69	7.03
5	3	73009	2419	2913	61	3.23	16.95
5	4	106276	4416	5704	77	4.09	22.59
5	5	147439	7890	10549	115	5.28	25.21
6	2	88120	2455	2694	59	2.72	8.88
6	3	129529	4501	5843	73	3.42	22.96
6	4	138868	8067	10504	101	5.74	23.20
6	5	210815	14209	19061	96	6.70	25.46
7	2	90341	4538	5223	63	4.96	13.12
7	3	130875	8152	10778	53	6.19	24.36
7	4	241083	14386	20095	117	5.92	28.41
7	5	369277	25038	37553	99	6.76	33.33
8	2	116541	8188	9668	57	6.98	15.31
8	3	170216	14471	20372	89	8.45	28.97
8	4	313405	25215	35336	112	8.01	28.64
8	5	395342	43443	61717	149	10.9	29.61
9	2	184450	14507	20731	43	7.84	30.02
9	3	297657	25300	36241	94	8.47	30.19
9	4	497175	43620	62569	86	8.76	30.28
9	5	841808	74508	107972	96	8.84	30.99
10	2	183781	14507	21203	37	7.88	31.58
10	3	248231	25301	38755	78	10.1	34.72
10	4	471943	43620	69172	131	9.22	36.94
10	5	601144	74508	118682	140	12.3	37.22
Average		253876	21023	30613	87	6.90	25.67

5. Summary and conclusion

In this paper, we have studied the open shop scheduling problem with time-dependent weights in minimizing weighted mean flow time. Considering weights as time-dependent factors results in a more realistic situation in long scheduling problem. For this problem, a mathematical formulation is presented. In addition, a genetic algorithm is proposed to solve the problem. Computational results demonstrate that the proposed genetic algorithm finds high quality solutions.

REFERENCES

[1] Kubiak, W.; Sriskandarajah, C.; Zaras, K.: A Note on the Complexity of Open Shop Scheduling Problems, INFOR, Vol. 29, 1991, 284 - 294.
[2] Liu, C.Y.; Bulfin, R.L.: Scheduling Ordered Open Shops, Comput. Oper. Res., Vol. 14, 1987, 257 - 264.
[3] Prins, C.: An Overview of Scheduling Problems Arising in Satellite Communications, Journal Oper. Res. Soc., Vol. 40, 1994, 611 - 623.
[4] Gonzalez, S.; Sahni, T.: Open Shop Scheduling to Minimize Finish Time, J. Assoc. of Comput. Mach., Vol. 23, 1976, 665 - 679.
[5] Dorndorf, U.; Pesch, E.; Phan-Huy, T.: Solving the Open Shop Scheduling Problem, Journal of Scheduling, Vol. 4, 2001, 157 - 174.
[6] Brasel, H.; Tautenhahn, T.; Werner, F.: Constructive Heuristic Algorithms for the Open-Shop Problem, Computing, Vol. 51, 1993, 95 - 110.
[7] Gueret, C.; Prins, C.: Classical and New Heuristics for the Open-Shop Problem: A Computational Evaluation, European J. Oper. Res., 107, 1998, 306 - 314.
[8] Alcaide, D.; Sicilia, J.; Vigo, D.: A Tabu Search Algorithm for the Open Shop Problem, Top, Vol. 5, 1997, 283 - 286.
[9] Liaw, C.-F.: A Hybrid Genetic Algorithm for the Open Shop Scheduling Problem, European J. Oper. Res., Vol. 124, 2000, 28 - 42.
[10] Prins, C.: Competitive Genetic Algorithms for the Open-Shop Scheduling Problem, Math. Meth. Oper. Res., Vol. 52, 2000, 389 - 411.
[11] Kyparisis, G.J.; Koulamas, C.: Open Shop Scheduling with Makespan and Total Completion Time Criteria, Comput. Oper. Res., Vol. 27, 2000, 15 - 27.
[12] Gupta, J.N.D.; Werner, F.; Wulkenhaar, G.: Two-Machine Open Shop Scheduling with Secondary Criterion, Intl. Trans. Oper. Res., Vol. 10, 2003, 267 - 294.
[13] Holland, J., 1975. Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press.
[14] Goldberg, D.E., 1989. Genetic algorithms in search, optimisation and machine learning. Reading, MA: Addison-Wesley.