Bi-objective simulated annealing and adaptive memory procedure approaches to solve a hybrid flow shop scheduling problem with unrelated parallel machines

H. Mohammadi, R. Sahraeian
Department of Industrial Engineering, Shahed University, Tehran, Iran
(Hamid_mohammadi33@yahoo.com, Sahraeian@shahed.ac.ir)

Abstract - **In this paper, a novel hybrid flow shop scheduling problem with respect to the both objectives of total setup time and cost is investigated. Since the problem is NP-hard, we focus on suboptimal scheduling solutions for the hybrid flow shop with unrelated parallel machines, sequence and machine-dependent setup time, eligibility constraint and release time. In this research, total sequence and machine-dependent setup time and cost are used as the objective functions. We present a multi-objective simulated annealing (MOSA) and an adaptive memory procedure (MOAMP) in order to find a set of non-dominated solutions. The problem solved with different number of jobs and stages using MOSA and MOAMP algorithms. Computational results showed that the proposed MOAMP approach is more effective and efficient than the simulated annealing algorithm in terms of reduced setup criterions for the attempted problem.**

Keywords - **Hybrid flow shop, multi-objective adaptive memory procedure, multi-objective simulated annealing, unrelated parallel machines**

I. INTRODUCTION

In reality, scheduling problems are multi-objective in nature, so for a schedule, there is a tendency to consider several objectives simultaneously. Therefore, the goal is to generate a feasible schedule that minimizes several objectives which often compete and conflict with each other simultaneously. In multi-objective optimization, there are a set of optimal solutions, rather than one optimal solution. Considering all objective functions, since no optimal solution can be considered to be better than any other, using a set of optimal solutions called *Pareto-optimal* seems unavoidable. In this paper, in order to approach real-world situation, a bi-objective scheduling problem is modeled being a simultaneous minimization of the total setup time and the minimization of total setup cost. Combination of both time and cost objectives is a novel and well-justified approach in HFS, as total setup time minimization implies the maximization of the throughput and total setup cost minimization is a major intellectual concern for management.

In many practical real-world situations, hybrid flow shop scheduling problems are seen. In the standard form of the HFS problem, all jobs and machines are available at time zero, machines at a given stage are identical, any machine can process only one operation at a time and any job can be processed by only one machine at a time; setup times are negligible, preemption is not allowed, the capacity of buffers between stages is unlimited and problem data is deterministic and known in advance. They mostly differ in two or three aspects only; the standard problem will serve as a template to which assumptions and constraints will be added or removed to describe different HFS variants [1].

Pinedo [2] cited machine setup time is a significant factor for production scheduling in all flow patterns, and it may easily consume more than 20% of available machine capacity if not well handled. Scheduling problems with sequence-dependent setup times (SDST) are among the most difficult classes of scheduling problems [3]. Therefore, in this paper, machine and sequence-dependent setup time restrictions are taken into account. In order to highlight the importance, total setup cost is brought into the objective function.

II. SCHEDULING STRATEGY

The flow shop scheduling problem can be described as follows. There exist n jobs from the set $N = \{1, 2, ..., n\}$ and m stages from the set $M = \{1, 2, ..., m\}$. These n jobs must be processed without preemption in a set M of m sequential production stages with the objective of minimizing two objectives; total setup time and total setup cost. In stage $i \in M$, a set $L_i = \{1, 2, \ldots, l_i\}$ of unrelated parallel machines is given where $|M_i|>1$. In each stage, each job has to be processed by exactly one machine. $p_{i,l,j}$ denotes the processing time of job $j \in N$ at machine $l \in Li$ in stage $i \in M$. As mentioned before, machine based sequence-dependent setup time is considered. $S_{i,l,j}$ denotes the setup time at machine $l \in Li$, in stage $i \in M$, when job $j \in N$ is the first job to be processed. The proportionate setup cost is shown with the notation $C_{i,l,j}$. Let $S_{i,l,j,k}$ and $C_{i,j,l,k}$ be the setup time and cost at machine l, in stage i, respectively, when processing job $k \in N$, after processing job j. A set of eligible machines that can process job j, in stage i is denoted by $E_{i,j}$, $1 \leq |Eij| \leq m_i$. Also there is no limitation on capacity of buffer in front of each machine. Release time for job j is indicated by the notation r_j, i.e. job j is ready to be processed at time r_j. The problem notations are summarized below:

Indices and sets:
N=Set of jobs; indexed by j or k
M=Set of stages; indexed by i
L_i=Set of machines; indexed by l
$E_{i,j}$=Set of eligible machines that can process job j in stage i

Parameters:
$p_{i,l,j}$ = Processing time of job j at machine l in stage i
$S_{i,l,j}$ = Setup time at machine l, in stage i, when job j is the first job to be processed
$C_{i,l,j}$ = Setup cost at machine l, in stage i, when job j is the first job to be processed
$S_{i,l,j,k}$ = Setup time at machine l, in stage i when processing job k after processing job j
$C_{i,j,l,k}$ = Setup cost at machine l, in stage i when processing job k after processing job j
r_j = Release time for job j

Gourgand et al. [4] demonstrated that for a given problem the total number of possible solutions is $(n!\Pi_{i=1}^{m} m_i)^n$. Moreover, Gupta and Tunc [5] proved that the flexible flow shop problem with m = 2 is NP-hard even if one of two-stage contains a single machine. Since HFS is a general case of the flexible flow shop, HFS is also NP-hard.

Using the usual three field notation α | β | γ for scheduling problems the problem can be defined as:

$$FHM,((RM^{(i)})^{m}_{i=1}|S_{sd}, M_j, r_j|\text{Total setup time \& cost})$$

III. OPTIMIZATION ALGORITHMS

The proposed bi-objective problem is solved with two well-known meta-heuristics and their performance compared with each other. These two algorithms are named as multi-objective simulated annealing (MOSA) and multi-objective meta-heuristic algorithm using an adaptive memory procedure (MOAMP).

A. Multi-objective simulated annealing

The simulated annealing (SA) algorithm is a usual prominent meta-heuristic that is applied in order to find the optimal or near-optimal solution for more scheduling problem. Ulungu et al. [6] introduced the multi-objective simulated annealing (MOSA). Since in MOSA there is no need for large memory to keep the population and also it can find a small group of Pareto solutions in a shorter time, it has some advantages over evolutionary algorithms. Procedure of MOSA is showed in Fig. 1. *S* represents the current search solution and *T* indicates the temperature parameter that is gradually decreasing as time goes on. A new search solution *S'* is generated by the *N(s)*, function for generating neighbor solution, whose objective function is assessed and compared with the current one. When the new solution is concluded to be better than the current one, it is accepted. Even if the new solution is not suitable, it is accepted with an acceptance probability. As there is no superiority between the current and the next solution, the new one is accepted in order to expand search space and help the algorithm to run away from local optima.

S = Initial solution
T = Initial temperature
Repeat
Generate a neighbor *S'* = *N*(*S*)
IF *C*(*S'*) dominates *C*(*S*)
Move to *S'*
Else IF *C*(*S*) dominates *C*(*S'*)
Move to *S'* with Transition Probability
Else (*C*(*S*) and *C*(*S'*) do not dominate each other)
Move to *S'*
End IF
T = Annealing (*T*)
END Repeat (until the termination are satisfied)

Fig. 1. Pseudo code of the MOSA

The common transition rules, such as the Metropolis or logistic method, cannot be applied directly to the multi-objective problems because they support only a scalar cost function. The transition rule suggested in this paper is very similar to the Metropolis method as shown in (1), where $C(a,b)$ is the cost criterion for transition from state a to b, and T is the annealing temperature. Equation (2) is the cost criterion, where $C_k(a)$ is the k-th cost value in the objective vector of the a-th solution.

$$P_t(a,b) = \exp\left(\frac{C(a,b)}{T}\right) \quad (1)$$

$$C(a,b) = Min_k(C_k(b) - C_k(a)) \quad (2)$$

B. MOAMP algorithm

The multi-objective meta-heuristic algorithm using an adaptive memory procedure (MOAMP) introduced by Caballero et al. [7] is for the resolution of multi-objective combinatorial optimization (MOCO) problems based on TS. The algorithm includes two different phases: (1) generating an initial set of efficient points through various tabu searches and (2) looking for efficient points with an intensification process around an initial set of efficient points. It is a well-known fact that the efficient points of MOCO problems are *connected*. In other words, any efficient point is close enough to another efficient point. This proximate optimality is the principle point in the MOAMP. Pseudo code of the MOAMP is depicted in Fig 2.

The first phase consists of linking p+1 tabu searches (i.e., the last point of one search becomes the initial point of the next search). The first TS starts from a random solution and attempts to find an optimal solution to the problem with the single objective $f_1(x)$. Let x^1 be the last point visited at the end of this search. Then, another tabu search is applied again to find the best solution to the problem with the second objective $f_2(x)$ using x^1 as its initial solution. This process is repeated until all the single-objective problems associated with the p objectives have been solved. At this point, we again solve the problem with the first objective $f_1(x)$ starting from x^p in order to

complete a cycle. This phase produces the p points that

```
S = Initial solution
Repeat
        Optimize 1st objective with TS
        Optimize 2nd objective with TS
                ⋮
        Optimize pth objective with TS
        Optimize 1st objective with TS
        Repeat
            Optimize Eq. (3) with TS
        END Repeat (N iterations)
END Repeat (until no change in the NDS list)
```

Fig. 2. Pseudo code of the MOAMP

approximate the best solutions to the single-objective problems, in which they are non-dominated solutions that result from ignoring all but one objective function. During this phase, we also collect other non-dominated solutions (NDS).

The second phase of the MOAMP explores the search space around the initial set of efficient points found in the first phase. In this phase, N random weighting vectors $\lambda = (\lambda_1, \ldots, \lambda_p)$ are generated and be used to make N tabu searches with the following objective function which should be minimized:

$$\min F_\lambda(x) = \max_{i=1,2,\ldots p} \left\{ \lambda_i \left(\frac{f_i^{\max} - f_i(x)}{f_i^{\max} - f_i^{\min}} \right) \right\} \quad (3)$$

Where

- f_i^{max} is the maximum value of the i-th objective over NDS obtained up to now
- f_i^{min} is the minimum value of the i-th objective over NDS obtained up to now
- N represents maximum number of tabu searches that could be carried out without any change in the NDS list.

Finally, at the end of these $(p+1)+N$ tabu searches, the algorithm obtain a sample of the non-dominated solutions distributed by the areas where one of the objectives is predominant, as well as for those areas characterized by a balance among the different objectives. The process is repeated until this intensification no longer offers any new NDS.

IV. PERFORMANCE METRICS

In recent years, many multi-objective optimization algorithms (MOOA) have been proposed and many metrics have been suggested in order to evaluate the algorithm performances as well. Deb [8] classified the existing performance metrics into three classes, namely convergence, diversity and both convergence and diversity.

Zitzler et al. [9] mentioned three goals for MOOA that can be measured by:

- The distance of the resulting NDS from the true Pareto front should be minimized.
- A good distribution of the obtained NDS is desirable.
- The size of the obtained NDS should be maximized (i.e., for each objective, a wide range of values should be covered by NDS).

To compare the performance of the algorithms, a convergence metric and a diversity metric are applied.

A. Convergence metric

The convergence metrics appraise how obtained solutions are far from the true Pareto front. Many metrics have been proposed that measure the convergence of a set of approximation NDS towards the Pareto front. In this research, a well-known convergence metric has been chosen. The CS metric is the coverage of two sets introduced by Zitzler [10]. Using the metric CS, two sets of NDS can be compared to each other. Let P_1 and P_2 be the sets of approximation NDS obtained from one run of Algorithms **A** and **B**, respectively, and P be the union of the sets of approximation NDS (i.e., $P = P_1 \cup P_2$) so that it includes only NDS. The function CS maps the ordered pair (P_1, P_2) into the interval [0, 1]:

$$CS(P_1, P_2) = \frac{|\{p_2 \in P_2 / \exists p_1 \in P_1 : p_1 \succeq p_2\}|}{|P_2|} \quad (4)$$

Where, $P_1 \succeq P_2$ means that the solution P_1 is dominated by the solution P_2.

The value $CS(P_1, P_2) = 1$ means that all solutions in P_2 are dominated by P_1. The value $CS(P_1, P_2) = 0$ indicates the situation when none of the points in P_2 are dominated by P_1. Note that $CS(P_1, P_2)$ is not necessarily equal to $1 - CS(P_2, P_1)$.

B. Diversity metric

The diversity metrics evaluate the divergence of solutions in the final population on the Pareto front. Like the convergence metrics, many metrics have been proposed to measure the diversity of a set of approximation NDS from the Pareto front.

We select the spacing metric (SM) to evaluate the applied algorithms. Schott [11] introduced the spacing metric that provides a measure of uniformity of the spread of approximation NDS. This metric is given by:

$$SM = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\bar{d} - d_i)^2} \quad (5)$$

Table 1
Test Problems

Test Problem	Stage	Job
1	3	10
2	3	20
3	3	30
4	3	40
5	3	50
6	4	10
7	4	20
8	4	30
9	4	40
10	4	50

Where

$$d_i = \min_{j \in NDS, j \neq i} \sum_{k=1}^{K} \left| f_k^i - f_k^j \right| \quad (6)$$

$\bar{d}$ is the mean of all d_i, n is the size of obtained NDS and f_k^i is the function value of the k-th objective function for solution i. The lower values of the SM are preferable. Note that the objective functions should be normalized.

V. EXPERIMENT DESIGN AND ANALYSIS

In order to compare the quality of algorithms, MOSA and MOAMP algorithms have been coded in Matlab 7.1 and run on a computer with a 2.4 GHz Core i5 CPU with 4.00GB of RAM. Ten test problems was designed with different number of jobs and stages as shown in Table 1 and solved with both algorithms.

Table 2 exhibits the average values of the SM metrics that have been resulted from 10 runs for each algorithm and test problem. Considering the SM metric, MOAMP performs better comparing to MOSA.

Fig 3 and 4 show the stock chart based on the CS(MOSA, MOAMP) and CS(MOAMP,MOSA), respectively. This chart consists of a line, in which the upper and lower ends of the line are maximum and minimum values whereas the average value is marked on the line.

The stock plots show that MOAMP outperforms MOSA. Table 3 compares the algorithms considering time of achieving the Pareto-optimal solutions. As seen, MOSA is so faster than MOAMP in achieving the Pareto-optimal solutions. This noticeable difference was to some extent predictable because of the nature of long time $(p+1)+N$ tabu searches in MOAMP.

In order to display the conflict between the objective functions, we run the SA algorithm for the test problems twice with one objective function; first, 100 runs with total setup time as an objective function and second, total setup cost. Comparing the near optimal solutions obtained from 200 runs of SA showed that only in 11 runs, the same near optimal solutions were obtained. Therefore, it is reasonable to claim that total setup time and cost compete with each other simultaneously in this problem.

Table 2
SM Performance Metric

Test Problem	MOSA	MOAMP
1	0.1962	0.1367
2	0.1681	0.065
3	0.1764	0.1045
4	0.1972	0.0488
5	0.1798	0.1691
6	0.1884	0.0762
7	0.1987	0.0772
8	0.1886	0.0917
9	0.2112	0.1021
10	0.1882	0.0992
Average	0.1893	0.0970

Table 3
Time of achieving the pareto-optimal solutions (s)

Test Problem	MOSA	MOAMP
1	0.445	59.55
2	0.585	101.65
3	0.855	169.15
4	1.045	215.60
5	1.445	320.45
6	0.685	75.50
7	0.905	128.85
8	1.355	226.90
9	1.865	305.05
10	2.355	382.55
Average	1.154	198.525

VI. CONCLUSION

In this paper, the researchers have developed efficient MOSA and MOAMP algorithms that solve the HFS problem with machine and sequence-dependent setup time, unrelated parallel machines, machine availability and release time constraints together. One contribution of this research lies in considering both time and cost

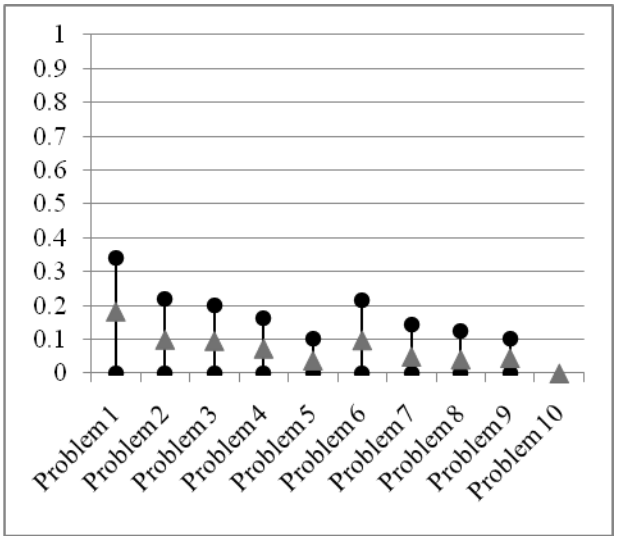

Fig. 3. CS(MOSA, MOAMP)

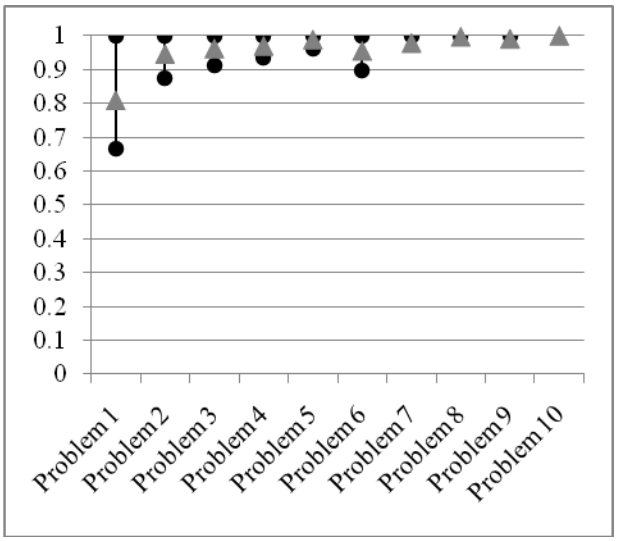

Fig. 4. CS(MOAMP,MOSA)

measures obtained by using total setup time and total setup cost as objectives. This research also presented two meta-heuristics (MOSA & MOAMP) solution approach. In order to test the algorithms, proposed HFS problem solved in 10 runs with both algorithms for 4 test problems. Computational results demonstrated that the solution approaches find high quality solutions. Comparing both algorithms, the proposed MOAMP approach is more effective and efficient than the MOSA in terms of reduced total setup time and total setup cost which was confirmed using diversity and convergence metrics, SM and CS, respectively.

This research can be extended by adding other constraints such as travelling time, limited buffer, blocking or batching to the considered problem. Another direction for this research is the use of the proposed objective function approach for other scheduling problems. Since, MOAMP is a novel and effective meta-heuristic, it is suggested to be applied in other interesting scheduling problems. As another interesting future work, the authors recommend the proposed problem to be solved with other well-known meta-heuristics such as SPEAΠ, MOIA or NSGAΠ.

ACKNOWLEDGMENT

The authors would like to thank Mr. Farshid Samaei, Master of Science in industrial engineering from shahed university of Tehran for his helpful and constructive comments and suggestions on this paper.

REFERENCES

[1] R. Ruiz, F. S. Serifoglu and T. Urlings, "Modeling realistic hybrid flexible flow shop scheduling problems." *Computers and Operations Research*, vol 35, no 4, pp. 1151–1175, 2008.

[2] M. Pinedo, *Scheduling theory, algorithms, and systems*. second edition, Englewood Cliffs, NJ: Prentice-Hall, 1995.

[3] J. Behnamian, S.M.T. Fatemi Ghomi, and M. Zandieh, " A multi-phase covering Pareto-optimal front method to multi-objective scheduling in a realistic hybrid flow shop using a hybrid metaheuristic." *Expert Systems with Applications*, vol 36, pp.11057 – 11069, 2009.

[4] M. Gourgand, N. Grangeon and S. Norre, "Metaheuristics for the deterministic hybrid flow shop problem." in *Proceedings of the International Conference on Industrial Engineering and Production Management*, 1999.

[5] J. N. D. Gupta and E.A. Tunc, "Minimizing tardy jobs in a two-stage hybrid flow shop." *International Journal of Production Research*, vol 36, no 9, pp. 2397 – 2417. 1998

[6] E. L. Ulungu, J. Teghem, P. H. Fortemps and D. Tuyttens, "MOSA method: a tool for solving multiobjective combinatorial optimization problems." *Journal of Multicriteria Decision Analysis*, vol 8, pp. 221-236, 1999.

[7] R. Caballero, M. Gonzalez, M. Flor, J. Molina and C. Paralera, " Solving a multiobjective location routing problem with a metaheuristic based on tabu search. Application to a real case in Andalusia." European Journal of Operational Research, vol 177, pp. 1751–1763, 2007.

[8] K. Deb, "Multiobjective optimization using evolutionary algorithms", Wiley; Chichester, UK, 2001.

[9] E. Zitzler, K. Deb, L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results." Evolutionary Computation, vol 8, no 2, pp. 173-195, 2000.

[10] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications. " Zurich: PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

[10] E. Zitzler, "Evolutionary algorithms for multiobjective optimization: Methods and applications. " Zurich: PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

[11] J. Schott, "Fault tolerant design using single and multicriteria genetic algorithms optimization." Department of Aeronautics and Astronautics. Cambridge: Master's thesis, Massachusetts Institute of Technology, 1995.