International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:6, No:2, 2012

A Flexible Flowshop Scheduling Problem with Machine Eligibility Constraint and Two Criteria Objective Function

Bita Tadayon, Nasser Salmasi

***Abstract*—**This research deals with a flexible flowshop scheduling problem with arrival and delivery of jobs in groups and processing them individually. Due to the special characteristics of each job, only a subset of machines in each stage is eligible to process that job. The objective function deals with minimization of sum of the completion time of groups on one hand and minimization of sum of the differences between completion time of jobs and delivery time of the group containing that job (waiting period) on the other hand. The problem can be stated as $FF_c / r_j , M_j / irreg$ which has many applications in production and service industries. A mathematical model is proposed, the problem is proved to be NP-complete, and an effective heuristic method is presented to schedule the jobs efficiently. This algorithm can then be used within the body of any metaheuristic algorithm for solving the problem.

***Keywords*—**flexible flowshop scheduling, group processing, machine eligibility constraint, mathematical modeling.

I. Introduction

THE idea of this research was raised when the authors had a dinner in a restaurant. Guests enter the restaurant in different groups at different moments. Each group chooses a table and all orders of the group members are taken simultaneously. The process of preparing meals starts after taking the order. This process contains a specific number of stages: preparing raw materials, cooking, and serving. Every meal passes through all the stages. However, some meal items should be processed by a specific resource in one or more stages. For instance, in cooking stage French fries have to be prepared on a fryer while soups have to be boiled on an oven. The process time in each stage as well as the eligible resource to process the meal is fixed for a specific kind of meal, regardless of the group it belongs to. After accomplishment of processes, all meal items ordered by a group are served simultaneously.

The quality of service and the satisfaction rate of customers can be raised if a meal item is served as soon as it is ready. In a restaurant, a group of meal items ordered by guests sitting on a table should be delivered together. Thus, the cooked meal items belonging to a specific group have to wait until the last item of that group would be cooked and be ready to be served.

Bita Tadayon is with the Department of Industrial Engineering, Sharif University of Technology,Tehran, Iran (e-mail: bita.tadayon@gmail.com).
Nasser Salmasi is with the Department of Industrial Engineering, Sharif University of Technology,Tehran, Iran (e-mail: nsalmasi@sharif.edu).

Because of this waiting time, the sooner prepared meal items get cold and the quality of service decreases. Therefore, the objective of this system would be minimization of sum of the delivery time for all groups in one hand and minimization of sum of the waiting times for all cooked meal items on the other hand.

This problem can be stated as the followings:

- The processing system contains several stages with parallel machines in some stages (at least one stage contains more than one machine in parallel). All tasks should pass through all stages in the same order. Therefore, the problem can be considered as a flexible flowshop scheduling (FFS) problem.
- The Jobs enter the system in groups in different times (r_j).
- Each job should be processed by specific resources in a few stages including parallel resources (M_j).
- The arrival and delivery of jobs in each group should be with each other.

Moreover, it is assumed that all processors in the last stage are eligible to process all jobs. This assumption is valid due to the fact that processors in the last stage (waitresses at restaurant who deliver meals or packers in a factory) are the same in most of the application areas of the proposed problem.

The objective function can be stated as the weighted sum of the two following objectives:

- Minimization of sum of the completion times of groups
- Minimization of sum of the differences between the completion time of each job and the completion time of the group that the job belongs to

Using the common notations to suggested by Pinedo, the research problem can be notated as $FF_c / r_j , M_j / irreg$ [1]. Since the objective function proposed in this research does not belong to regular objective functions known in the literature of scheduling, the last section of the notation (the objective function part) is mentioned as irregular. Moreover, because of its newness, no notation exists for the assumption of arrival and delivery of jobs in groups. Therefore, this assumption is not included in the notation offered for the problem.A similar problem has been addressed while investigating the production process in a ceramic tile manufacturing company [2]. Their proposed problem can be notated as $FF_c / s_{jk} , M_j / C_{max}$. The major difference between their research problem and the one proposed in this research is that they define minimization of makespan as the objective function of the problem. Moreover, they assume the arrival and delivery of jobs are performed

individually.The proposed research problem has many applications in industrial companies, especially in food, ceramic tile, and textile production industries. In general, every industry which receives orders in groups and faces the limitation of delivering them in the same group deals with the problem proposed in this research. To the best of our knowledge, the first research related to FFS is proposed by Salvador in 1973 who models the production system in the synthetic fibers industry as a no-wait FFS [3]. In the literature, there are two extensive review and classification papers which provide comprehensive review about the research accomplished on FFS problems based on exact algorithms, metaheuristics, and also heuristic algorithms [4], [5]. These works clearly reveal open areas for more research in this field.Exact algorithms used to solve FFS problems are mostly based on branch and bound (B&B) method. Since the FFS problems are shown to be NP-hard, the exact methods are incapable of solving real world problems. Thus, it is necessary to find non-exact algorithms to deal with such problems. The most popular metaheuristics used to solve FFS problems are genetic algorithm, simulated annealing, and tabu search. Tseng and Liao perform the only research to solve FSS problem by applying Particle Swarm Optimization (PSO) algorithm [6]. They addressed a FFS problem with multiprocessor tasks, which means each job has to be processed on several machines in each stage. They develop a regular PSO algorithm to solve the problem with minimization of makespan as the criterion.Machine eligibility constraint is one of the assumptions considered in this research. There exist only two research applying this assumption in FFS problems. In the first one, a genetic algorithm is proposed to solve the FFS problem considering this assumption [2]. In the second research, a mathematical model and a heuristic algorithm is suggested for the same problem [7].The objective function of this research has not been addressed before. In fact, most of the research has applied regular scheduling objective functions in dealing with FFS problems. For instance minimization of makespan (C_{max}) [8], average flow time criterion ($\bar{F}$) [9], and maximum lateness (L_{max}) [10] are the most common objective functions in the literature.Papers related to the scheduling areas have been appeared in scientific journals since 1954. But there is a noticeable gap between the theory and the application of existing methods. This issue has been pointed out and several research directions to aid in bridging this gap has been proposed [11]. Reisman et al. report that from a total of 184 reviewed papers, only five of them (less than a 3%) have dealt with realistic production settings [12]. Recently, the researchers are encouraged to approach realistic problems. The proposed research problem has the advantage of real world application since the idea is initiated in real world with several applications.

II. Mathematical Model

A mixed integer linear programming (MILP) model is developed for the research problem. The sets, parameters, decision variables, and the model are as follows:

A. Sets and Parameters

M	A very large number
g	number of groups
G	set of group indices ($G = \{1,2,...,g\}$)
j_a	number of jobs in group a ($a \in G$)
J_a	set of job indices in group a ($J_a = \{1,2,...,j_a\},\ a \in G$)
c	number of stages
C	set of stage indices ($C = \{1,2,...,c\}$)
R	set of group and job indices necessary to define decision variables ($R = \{a,p,b,q \mid (a<p) \cup ((a=p) \cap (b<q))\}$)
y_k	number of machines in stage k ($k \in C$)
V_{abk}	subset of eligible machines to process job b in group a in stage k ($a \in G,\ b \in J_a,\ k \in C$)
t_{abk}	processing time of job b in group a in stage k ($a \in G,\ b \in J_a,\ k \in C$)
r_a	arrival time for group a ($a \in G$)
α_e	weighting coefficients of the two parts of objective function ($e \in \{1,2\}$)

The set R is defined to omit unnecessary decision variables and constraints in mathematical model expression. The importance and necessity of defining this set is discussed after the definition of decision variables in this section.

B. Decision variables

X_{abk}	completion time of job b in group a in stage k ($a \in G,\ b \in J_a,\ k \in C$)
CT_a	completion time of the last job in group a ($a \in G$)
Y_{abks}	1:if job b in group a in stage k is processed on machine s 0:otherwise($a \in G,\ b \in J_a,\ k \in C,\ s \in V_{abk}$)
W_{abpqk}	1:if job b in group a in stage k is processed before job q in group p 0:otherwise ($a,p \in G,\ b \in J_a,\ q \in J_p,\ k \in C,\ a,p,b,q \in R$)
U_{abpqk}	1:if job b in group a and job q in group p are processed on the same machine in stage k 0:otherwise($a,p \in G,\ b \in J_a,\ q \in J_p,\ k \in C,\ a,p,b,q \in R$)

If the decision variables are defined regarding all of their indices, the number of variables will increase to a huge number and the efficiency of model will decrease drastically.

Thus, the unnecessary decision variables are omitted by defining set *R*. For instance, if W_{abpqk} is equal to one, the value of W_{pqabk} is zero and vice versa. Therefore, by having the value of one of the two couple variables, the other one can be calculated. Thus, one of them would be enough to define the constraints in mathematical model. By limiting the indices of W_{abpqk} and U_{abpqk} as well as the indices of related constraints of the mathematical model to the members of set *R* the number of decision variables and constraints are decreased.

To reduce the size of the problem, in addition to definition of set *R* we apply another condition on W_{abpqk} and U_{abpqk}: W_{abpqk} and U_{abpqk} are included in the model if V_{abk} and V_{pqk} have at least one member in common. In other words, if job *b* in group *a* and job *q* in group *p* in stage *k* are processed on different machines, there is no need to add a constraint to prevent interference between the processing operations of these two jobs.

C. The Model

$$Z = \alpha_1 \sum_{a=1}^{g} CT_a + \alpha_2 \sum_{a=1}^{g} \sum_{b=1}^{j_a} (CT_a - X_{abc}) \quad (1)$$

Subject to:

$$X_{pqk} - X_{abk} + M(1 - W_{abpqk}) \geq t_{pqk}, \quad (2)$$
$$k \in C,\ a, p \in G,\ b \in J_a,\ q \in J_p,\ a, p, b, q \in R$$

$$X_{abk} - X_{pqk} + M(1 - U_{abpqk} + W_{abpqk}) \geq t_{abk}, \quad (3)$$
$$k \in C,\ a, p \in G,\ b \in J_a,\ q \in J_p,\ a, p, b, q \in R$$

$$X_{abk} - X_{ab(k-1)} \geq t_{abk}, \quad a \in G,\ b \in J_a,\ k \in C \quad (4)$$

$$CT_a \geq X_{abc}, \quad a \in G,\ b \in J_a \quad (5)$$

$$\sum_{z \in V_{abk}} Y_{abks} = 1, \quad a \in G,\ b \in J_a,\ k \in C \quad (6)$$

$$U_{abpqk} \geq Y_{abks} + Y_{pqks} - 1, \quad k \in C,\ a, p \in G \quad (7)$$

$$X_{ab0} = r_a, \quad a \in G,\ b \in J_a \quad (8)$$

$$CT_a, X_{abk} \geq 0, \quad a \in G,\ b \in J_a,\ k \in C$$
$$Y_{abks}, W_{abpqk}, U_{abpqk} \in \{0,1\}, \quad k \in C, \quad a, p \in G,$$
$$b \in J_a,\ q \in J_p,\ a, p, b, q \in R,\ s \in V_{abk}$$

The objective function, as presented in (1), is a weighted sum of two statements. The first statement calculates the sum of the completion times for all groups and the second one is incorporated to calculate the sum of the waiting time of jobs belonging to each group. Waiting time is defined by the difference between the completion time of each job and the completion time of the group it belongs to in the last stage (stage *c*). The weighting coefficients (α_1, α_2) are two numbers in the range of [0,1] which are defined based on the importance of each part of the objective function in practical situations and have to obtain the condition stated in (9).

$$\alpha_1 + \alpha_2 = 1 \quad (9)$$

Constraint sets (2) and (3) preclude the interference between the processing operations of any two jobs on a machine. At most one of these two constraint sets is active for each couple of jobs. If job *b* of group *a* is processed before job *q* of group *p* on the same machine in stage *k*, constraint set (2) is activated to prevent interference between the processing operations of these two jobs and constraint set (3) is satisfied for all values of *a*, *b*, *p*, and *q* which have the stated condition. In the opposite situation, constraint set (3) undertakes this duty. Constraint set (4) ensures that the processing operations of a job in two consecutive stages do not interfere. Constraint set (5) is incorporated to the model to find the completion time of each group in the last stage. Constraint set (6) is incorporated to the model to support this fact that every job has to be processed in each stage on exactly one eligible machine. Constraint set (7) determines the jobs which are processed on the same machine in stage *k*. Constraint (8) is incorporated to assure that the process of each job starts no sooner than its arrival time (r_a).

FFS problem by considering minimization of makespan criterion ($FFc|\ |\ C_{max}$) is an NP-complete problem [4]. The problem proposed in this research can be easily reduced to a regular flexible flowshop problem by assuming each group contains only one job which is available at the beginning of the planning horizon, and by considering all machines eligible to process every job in each stage. Moreover, the objective function proposed in this research is more complex than minimization of makespan. Based on these insights, it is easy to see that the proposed research problem is easily reducible to the one already proven NP-complete. Thus, the fact that the proposed research problem is NP-complete follows immediately. Therefore, several metaheuristic algorithms are proposed to solve industry size problems in a reasonable time.

III. Heuristic Algorithm to Calculate the Objective Function

To solve the problem either optimally or approximately, we need to calculate the objective function value. Most of the procedures for solving the proposed problem search in a space of vectors presenting the sequence of jobs and try to find the best possible sequence regarding the objective function value of that sequence. However, given a specific sequence of jobs, calculating the objective function value proposed in this research is not easy. In other words, to calculate the objective function value for a sequence of jobs, each job has to be assigned to a machine in each stage and the schedule of processing jobs on each machine should be determined. The efficiency of this procedure has a significant effect on the quality of the objective function value. Therefore, an efficient heuristic method with four levels is proposed here to schedule the processing of jobs on machines at each stage. This

procedure can then be used within the body of exact and approximate search methods for solving the problem.

The levels of the proposed algorithm are as follows:

A. Level 1

At this level, the initial schedule of processing jobs on machines in all stages is determined based on the simple procedure of assigning the first job in the sequence to the first available eligible machine. This is done based on an algorithm, called algorithm 1. This algorithm is performed for all stages (the stage number is noted by *k* inside the algorithm) starting from the first stage to the last one, respectively.

Algorithm 1

Step 1: Assume that π is the initial sequence vector of jobs.

Step 2: Set the value of parameter *i* to 1.

Step 3: Consider the i^{th} job within the sequence vector π. Find all the candidate machines to process this job in the following stage (*k*) and put *s* equal to the number of these machines. A machine can be considered as a candidate machine to process the i^{th} job in stage *k* if it meets two following conditions:

1. Be eligible to process the i^{th} job.
2. Be available when the i^{th} job is ready to be processed in stage *k*.

Step 4:

- If $s = 0$, assign the i^{th} job to the first available machine which is eligible to process this job.
- If $s = 1$, assign the i^{th} job to the candidate machine.
- If $s > 1$, assign the job to the candidate machine which has the minimum number of candidate jobs eligible to be assigned to that machine in the current stage.

Step 5: If there are unscheduled jobs in π, increase the value of *i* by one and return to step 3. Otherwise, update π by sorting the jobs regarding the value of their completion time in the current processing stage ascending and terminate the algorithm.

Then, in the next three levels, the schedule of processing jobs on machines in the last stage is revised to improve the value of the objective function.

B. Level 2

After executing algorithm 1 for all stages, calculate the last job's completion time in each group in the last stage (C_{max}). Then, determine the new sequence of jobs regarding the three criteria stated below:

1- The jobs belonging to each group should be processed without any preemption by other jobs of other groups.

2- The sequence of processing groups is determined due to the value of their C_{max}, in ascending order.

3- The sequence of jobs in each group is defined based on the sequence vector of jobs (π).

Revise the sequence vector of jobs (π) based on the above criteria. Then, execute algorithm 1 for the last processing stage once again using the new sequence vector (π) and calculate the updated values of C_{max} for all groups.

C. Level 3

For all groups starting from the last group (the group with the largest value of C_{max}) continuing to the next group in descending order of their C_{max} values, delay the process of all jobs in that group up to the latest possible time without changing the value of C_{max}. Then, if possible (due to the schedule of jobs in previous stages) sort the sequence of processing jobs on each machine in descending order of their processing time (due to the second part of the objective function, it is better to process jobs with smaller processing time after the ones with longer processing time).

D. Level 4

Starting from the schedule determined by the first three levels, in this level we try to revise the last stage schedule to improve the value of the objective function. In order to do that, we use algorithm 2 for all groups one by one in the same order described in level 3 (descending order of their C_{max} values). In this algorithm, variable *change* enumerates the possible beneficial changes in the schedule and terminates the algorithm when no other useful change is possible. In order to clarify the algorithm, note that the idle time for each machine is defined as the time interval between the start of processing the first job in the current group assigned to that machine and the end of processing the last job in the previous group assigned to that machine.

Algorithm 2

Step 1: Set the value of variable *change* to zero.

Step 2: Select the machine that processes the minimum number of jobs in the current group in the last stage. If there is more than one machine with this condition, select the machine with the least scheduled processing time of jobs in the current group. Consider all jobs in this group that are assigned to the other machines in the last stage and determine the candidate jobs to be transferred to the selected machine among them. A job has to meet the following two conditions to be a candidate for this purpose:

1- The process time for the job should be less than or equal to the idle time period of the selected machine.

2- Transferring the process of this job to the idle time period of the selected machine has to improve the second part of the objective function. In other words, the completion time for this job should be delayed by this transfer. Put all candidate jobs in set *Q*.

Step 3: Suppose the number of members in set *Q* is equal to *m*.

- If $m = 1$, then transfer the process of this job to the idle time period of the selected machine and set the value of *change* to zero.
- If $m > 1$, then transfer the process of the job with minimum processing time within the set *Q* to the idle time period of the selected machine and set the value of *change* to zero.
- If $m = 0$, omit the current machine from the list of machines for selecting in step 2 of the algorithm and increase the value of *change* by one.

Step 4: If the value of *change* is equal to the number of machines in the last stage, terminate the algorithm. Otherwise, return to step 2.

After assigning all jobs to the machines in all stages, the completion time for each group and for the jobs in that group is determined and the value of the objective function is calculated using (1).

E. Example

To clarify the performance of this heuristic algorithm, an example is provided as follows:

Suppose that there are three groups of jobs to be processed and these groups include four, two, and three jobs, respectively. Assume that there are four processing stages. The number of machines in the first stage is two and there are four machines in each of the other three stages. The data for this problem is shown in Table I.

TABLE I
DATA FOR EXAMPLE PROBLEM

Group	r_a (minutes)	Job	Stage	t_{abk} (minutes)	v_{abk}
1	22	1	1	27	1, 2
			2	24	1, 2, 4
			3	23	1, 4
			4	19	1, 2, 3, 4
		2	1	11	1, 2
			2	26	1
			3	7	2, 3
			4	26	1, 2, 3, 4
		3	1	7	1, 2
			2	8	1, 4
			3	17	2, 3
			4	30	1, 2, 3, 4
		4	1	2	1
			2	3	2, 3
			3	16	1, 2
			4	13	1, 2, 3, 4
2	9	1	1	3	2
			2	8	1, 2
			3	27	2, 4
			4	7	1, 2, 3, 4
		2	1	5	1, 2
			2	4	1, 2
			3	28	1, 2
			4	3	1, 2, 3, 4
3	29	1	1	2	2
			2	2	4
			3	11	1, 2, 3, 4
			4	28	1, 2, 3, 4
		2	1	12	1, 2
			2	13	2
			3	29	2
			4	26	1, 2, 3, 4
		3	1	17	1
			2	26	1
			3	21	1, 4
			4	12	1, 2, 3, 4

The problem is solved by CPLEX 10.1.1 by applying the proposed mathematical model. The optimal sequence vector of jobs for this problem is $\pi = (21, 22, 14, 33, 32, 31, 11, 12, 13)$ in which entry *ij* of vector π refers to the j^{th} job of the i^{th} group. Using this vector, the algorithm is performed to obtain an efficient sequence. Figures 1 to 4 in appendix section illustrate the four levels of implementing the algorithm for this example.

IV. Conclusions and Future Research

In this research, a real world problem in service sector (restaurant business) is investigated. The problem is then justified as a flexible flowshop scheduling problem with special characteristics that can be presented as FF_c / r_j , M_j / *irreg* based on known scheduling notations. A mathematical model to solve the problems optimally is proposed. The problem is proved to be NP-complete and an efficient method to obtain the objective function given the sequence of jobs is presented.

The proposed research problem has many applications in real world and can be applied in companies and service sectors in order to reduce their costs.

Since the problem has been proposed for the first time in this research there are rooms for further research. Heuristic and meta-heuristic methods which deal with the sequence of jobs can be used to solve the problem approximately using the procedure explained in this research to obtain the schedule of jobs and the objective function value. Applying more recent meta-heuristic algorithms such as PSO (Particle Swarm Optimization) is suggested for solving this problem.

Moreover, finding an efficient lower bounding mechanism is an interesting problem that can help to provide a valuable tool to evaluate the performance of approximate algorithms.

References

[1] M. Pinedo, *Scheduling Theory, Algorithms, and Systems*. 3rd Edition, LLC, New York: Springer, 2008.

[2] R. Ruiz, C. Maroto, "A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility," *European Journal of Operational Research*, vol. 169(3), pp. 781–800, 2006.

[3] MS. Salvador, "A solution to a special class of flow shop scheduling problems," In: Elmaghraby SE, (Eds.), *Symposium of the Theory of Scheduling and Applications*, Berlin: Springer, 1973, pp. 83–9.

[4] I. Ribas, R. Leisten, JM. Framinan, "Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective," *Computers & Operations Research*, vol. 37, pp. 1439–54, 2010.

[5] R. Ruiz, JA.Vázquez-Rodríguez, "The hybrid flow shop scheduling problem," *European Journal of Operational Research*, vol. 205, pp. 1–18, 2010.

[6] CT. Tseng, CJ. Liao, "A particle swarm optimization algorithm for hybrid flowshop scheduling with multiprocessor tasks," *International Journal of Production Research*, vol. 46(17), pp. 4655–70, 2008.

[7] R. Ruiz, FS. Serifoglu, T. Urlings, "Modeling realistic hybrid flexible flowshop scheduling problems," *Computers & Operations Research*, vol. 35(4), pp. 1151–75, 2008.

[8] T. Sawik, "An exact approach for batch scheduling in flexible flow lines with limited intermediate buffers," *Mathematical and Computer Modeling*, vol. 36(4–5), pp. 461–71, 2002.

[9] H. Allaoui, A. Artiba, "Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints," *Computers and Industrial Engineering*, vol. 47(4), pp. 431–50, 2004.

[10] JL. Cheng, Y. Karuno, H. Kise, "A shifting bottleneck approach for a parallel machine flowshop scheduling problem," *Journal of the Operations Research Society of Japan*, vol. 44(2), pp. 140–56, 2001.
[11] SC. Graves, "A review of production scheduling," *Operations Research*, vol. 29(4), pp. 646–75, 1981.
[12] A. Reisman, A. Kumar, J. Motwani, "Flowshop scheduling/sequencing research: a statistical review of the literature, 1952–1994," *IEEE Transactions on Engineering Management*, vol. 44(3), pp. 316–29, 1997.