Proceedings of Engineering & Technology (PET)
pp. 144-148

Lower and Upper Bounds for Two-machine Flowshop Scheduling Problem with Exact Time Lags

Amdouni Hajer
Higher School of Economic and Commercial Sciences
University of Tunis, Tunisia
Email: hajerelamdouni@yahoo.com

Ladhari Talel
Higher School of Economic and Commercial Sciences
University of Tunis, Tunisia
College of Business Management
Umm Al-Qura University, Saudi Arabia
Email: talel_ladhari2004@yahoo.fr

***Abstract*—In this paper we consider the two-machine permutation flowshop scheduling problem under exact time lags constraints. This NP-hard problem is denoted by $F2|l_j, prmu|\sum C_j$ and it models many practical real-life systems in different areas (computer sciences, agricultural, manufacturing, etc). The objective is to minimize the total completion time that represents the sum of all the end-time jobs. In this context, we provide, a new priority rule based on the machines availability and several constructive heuristics. Experimental results, done on four different sets of randomly generated instances, show the efficiency of the constructive heuristic based on the new proposed priority rule. In the other hand, we present a new preemptive lower bound based on the relaxation of the first machine.**

I. Introduction

In this work, we consider the two-machine flow shop scheduling problem with exact time lags. This problem frequently occurs in real-world applications. In fact, exact time lags between two succeeding operations of a job may model a transportation time, drying time and cooling down time, communication time, etc.

This problem is identified as follows. We are given two machines M_1 and M_2 and a set of n independent jobs. We assume that each machine is continuously available from time zero onwards, and can process at most one job at a time. All the jobs are available at time zero and each job j (j=1,2,...,n) must be firstly processed on machine M_1 during p_{1j} time units, and then on machine M_2 during p_{2j} time units. Preemption is not allowed. Each job j requires a nonnegative delay l_j, which decreases the minimum amount of time between the completion time of job j on M_1 and its start on M_2. All parameters, such as processing and time lags, are assumed to be known with certainty. The problem is to find a feasible permutation schedule that minimizes the total completion time. Using the notation specified in [3], this strongly NP-hard problem is denoted by $F2|l_j, prmu|\sum C_j$.

In this work, we provide a new priority rule, six constructive heuristics. Then, we experimentally show that the proposed heuristic based on the new priority rule is promising. Moreover, we present a new polynomial lower bound for our studied problem based on the relaxation of the first machine.

mds

January 20, 2015

II. A Priority Rule for Total Completion Time Under Minimum Time Lags

In this section, we provide a new priority rule for the total completion time under exact time lags. In fact, our motivation is to improve the schedule of two jobs while taking into account the availability of machines. After studying all possible cases to schedule consecutive jobs, we define the new priority rule denoted $PRTCTl_j$ as follows:

Let
v_1, v_2: the availability times of M_1 and M_2, respectively.
C_{kj}: the completion times of j on machine M_k $(k = 1, 2)$

$$PRTCTl_j(j, v_1, v_2) = 2\max(v_2, C_{1j}) + p_{2j}$$

where
C_{1j}=$v_1 + p_{1j} + l_j$

This new priority rule can be involved in a constructive heuristic based in scheduling the job with smallest $PRTCTl_j$ value.

III. Constructive heuristics

Due to the NP-hardness of $F2|l_j, prmu|\sum C_j$ flow shop problem proved by Brucker and al. (2005) [12] , constructive heuristics are useful and relevant when solving this problem. In this section, we formally present the steps of our developed constructive heuristics.

In general, constructive heuristics comprise two steps: First, arrange the jobs according to a priority rule, and second extend iteratively the empty initial sequence until the final complete sequence is constructed. In our work, we propose six constructive heuristics $(H1, .., H6)$ which make use of the following priority rules in their first step:

H1: Arrange the jobs in non-decreasing order of $PRTCTlj$
H2: Arrange the jobs in decreasing order of P_{1j}+ $P_{2j} + l_j$
H3: Arrange the jobs in non-decreasing order of P_{1j}
H4: Arrange the jobs in non-decreasing order of P_{2j}
H5: Arrange the jobs in non-decreasing order of $P_{1j} + l_j$
H6: Arrange the jobs in non-decreasing order of $P_{2j} + l_j$

The second step of our six proposed heuristics is based on our adaptation of the well-known NEH procedure for the completion time objective under time lags constraint. In this adaptation, we consider time lags as a processing time on an

ISSN: 2356-5608

other machine. We give in the following the algorithms and the examples for these constructive heuristics.

First, we present H2 heuristic which is our adaptation of the well known NEH procedure of Nawaz and al. (1983) [**?**], to the $F2|l_j, prmu|\sum C_j$,. NEH procedure has been originally proposed for the $F \mid\mid C_{\max}$. (see Algorithm 1)

Algorithm 1 An outline of NEH algorithm (H2)

Step 1 Arrange the jobs in decreasing order of the total processing time on all machines.
Step 2 Consider the first two jobs and schedule them in order to minimize the partial total completion time as if there were only these two jobs.
Step 3 Select an unscheduled job with the largest total processing time.
Step 4 Insert this job (selected in **Step 3**) into the scheduled jobs assuming the preceding job order in position which minimizes the total completion time. Repeat step 3 and 4 until all jobs are scheduled.

Second, we present H1 which based on a $PRTCTl_j$ presented in Section II.

We have
σ : a partial sequence of the scheduled jobs.
$\overline{J}$: the set of unscheduled jobs.
v_1 and v_2: the availability times of M_1 and M_2, respectively.

Algorithm 2 An outline of H1 algorithm

Step 1 $\sigma = \varnothing; \overline{J} = J$.
Step 2 Arrange the jobs in $\overline{J}$ in nondecreasing order of $PRTCTl_j\_First_J = 2(p_{1j} + l_j) + p_{2j}$.
Step 3 Schedule the first job J1 provided by step1 and set $\overline{J} = \overline{J}\backslash\{J1\}$.
Step 4 Calculate $PRTCTl_j(j, v_1, v_2)$ for each job $j \in \overline{J}$.
Step 5 Select the job j^* having the minimum $PRTCTl_j(j, v_1, v_2)$ and insert it in $k+1$ possible positions of σ.
Among k sequences, select the one with the minimum partial total completion time and set it as the current σ. Set $\overline{J} = \overline{J}\backslash\{j^*\}$.
Step 6 If $\overline{J} = \varnothing$ **then** stop **otherwise** goto **Step 4**.

The remaining heuristics $(H3, .., H6)$ are based on the priority rules presented in the work of Huo and al [7] for $F2|no-wait|\sum C_j$. We modify only the first step of NEH heuristic by including respective priority rule as follows:

H3: Arrange the jobs in non-decreasing order of P_{1j}
H4: Arrange the jobs in non-decreasing order of P_{2j}
H5: Arrange the jobs in non-decreasing order of $P_{1j} + l_j$
H6: Arrange the jobs in non-decreasing order of $P_{2j} + l_j$

In the following, we give an example to illustrate the procedure of constructing a solution in $(H1, .., H6)$ heuristics.

Example. Consider the instance of $F2|l_j, prmu|\sum C_j$ defined by table I.

j	$P_{1,j}$	l_j	$P_{2,j}$
1	10	84	53
2	31	78	16
3	4	3	17
4	76	25	1
5	33	6	22

TABLE I. A FIVE-JOB INSTANCE OF $F2|l_j|\sum C_j$

The final sequences of $(H1, .., H6)$ heuristics are respectively:

H1: $\{J3, J5, J1, J2, J4\}$ with $\sum C_j = 674$.
H2: $\{J3, J5, J2, J1, J4\}$ with $\sum C_j = 682$.
H3: $\{J3, J5, J2, J1, J4\}$ with $\sum C_j = 682$.
H4: $\{J3, J5, J1, J2, J4\}$ with $\sum C_j = 674$.
H5: $\{J3, J5, J1, J2, J4\}$ with $\sum C_j = 674$.
H6: $\{J3, J5, J1, J2, J4\}$ with $\sum C_j = 674$.

IV. NEW CAPACITY RELAXATION'S LOWER BOUND: LB_{pmtn}

Typically, a valid lower bound is an optimal solution of the original problem after constraint relaxations. In our work, we derive a new lower bound for the $F2|l_j, prmu|\sum C_j$ problem by solving the subproblem that is obtained after relaxing the capacity of the first machine M_1 and then after allowing preemption [1]. More precisely, if we relax the capacity of one machine, then we obtain a one-machine problem, where we have to sequence a set of jobs subject with time lags so as to minimize the total completion time. In the sequel, we give details of relaxation's steps for obtaining our new lower bound denoted LB_{pmtn}.

- ***Step 0:***

  The original problem: $F2|l_j, prmu|\sum C_j$

- ***Step 1:***

  We relax the capacity of the first machine M_1, then we get a relaxed problem: the single machine total completion time subject to release dates denoted $1|r_j|\sum C_j$, where for each job j belongs to J:

  - The release date is $r_j = p_{1j} + l_j$
  - The processing time is $p_j = p_{2j}$.

  This latter problem is known to be NP-hard in the strong sense [11]. Here we can derive several lower bounds by executing another relaxations.

- ***Step 2:***

  We apply a second relaxation by allowing preemptive schedules. The problem becomes $1|r_j, pmtn|\sum C_j$. This problem can be solved in a polynomial time and have a computational complexity $O(nlogn)$ [10].

- ***Step 3:***

  LB_{pmtn} is the optimal solution of $1|r_j, pmtn|\sum C_j$ by applying the well known **SRPT** rule (**S**hortest **R**emaining **P**rocessing **T**ime). In fact, SRPT rule schedule at any time an available job having the shortest processing time.

The lower bound derived by this SRPT rule is proved to the best lower bound for $1|r_j|\sum C_j$ [13].

Example. Consider the same instance of $F2|l_j, prmu|\sum C_j$ defined by table I.

If we relax the capacity constraint of machine M1 we obtain the following instance of $1|r_j|\sum C_j$ problem:

j	r_j	p_j
1	94	53
2	109	16
3	7	17
4	101	1
5	39	22

TABLE II. OBTAINED INSTANCE OF $1|r_j|\sum C_j$

Consider the same instance of $F2|l_j, prmu|\sum C_j$ defined by table I and the first instance of $1|r_j, pmtn|\sum C_j$defined by Table II

The sequence obtained by SRPT rule is

$J3, J5, J4, J2, J1$

with $\sum C_j$ = 24 + 61 + 102 + 125 + 164 = 476

So LB_{pmtn} **= 476**

V. COMPUTATIONAL RESULTS

This section describes the computational tests which have been conducted to evaluate the empirical performance of the proposed constructive heuristics. All these procedures have been coded in *C*and carried out on an Intel(R) Core(TM) 2 Duo CPU T6600 @ 2.20 GHz 2.19 GHz PC with 3,49 GB RAM.

Test sets's characteristics and the comparison between proposed constructive heuristics are presented in the subsections below.

A. Tests problems

In order to appraise the effectiveness of the different proposed algorithms, we carried out a series of experiments based on four different randomly generated problem sets $(S1, .., S4)$. Each problem set contains 180 test problems corresponding to 30 instances for small and large problems $n \in \{20, 30, 50, 70, 100, 200\}$. It means, that the test sets consists of 720 different test instances.

The time lags and the processing times are respectively uniformly distributed between $[1, \alpha]$ or $[1, \beta]$. In this way, we generated all combinations of long-short time lags and processing times. The characteristics of these sets are provided in Table III.

$Subset$	α	β
$S1$	$[1, 20]$	$[1, 100]$
$S2$	$[1, 100]$	$[1, 100]$
$S3$	$[1, 20]$	$[1, 20]$
$S4$	$[1, 100]$	$[1, 20]$

TABLE III. TEST SETS'S CHARACTERISTICS

B. Performance of the proposed heuristics

The results of the computational study of the proposed heuristics are summarized in Table IV. The proposed heuristics run very fast, so we dont consider CPU time as a comparative criteria. The performance analysis is based on the average relative percentage deviation (ARPD) from the best-known solution. The percentage deviation is defined as $(\frac{(UB-UB^*)}{UB^*} * 100)$; where UB is the solution provided by the heuristic Hi (i= 1,.., 6), UB^* is the best solution between the compared heuristics.

Looking at table IV, we note that H1 is very effective and outperforms all the other constructive heuristics. Indeed H1 presents lower values of average relative percentage deviation from the best-known solution compared with the other constructive heuristics for almost all instances and especially when time lags are less or equal to the processing time (Sets: S1, S2, S3). However, it is worth noting that the results of H1 improve in large sets $(n >= 70)$.

S	n	H1	H2	H3	H4	H5	H6
S1	20	**0,0925**	4,2532	1,1429	1,3103	1,1212	1,6370
	30	**0,0034**	4,7397	1,7852	2,1716	2,0468	2,3391
	50	**0,0000**	6,1850	2,0603	2,2360	2,3515	2,1991
	70	**0,0000**	6,8706	2,3668	2,5405	2,6011	2,5616
	100	**0,0000**	5,4682	0,1141	0,3258	0,2704	0,3685
	200	**0,0000**	8,1053	3,4931	3,4947	3,3895	3,2209
S2	20	**0,8153**	3,4155	0,7726	1,4182	2,0569	2,1003
	30	**0,2603**	3,3956	1,1604	0,9257	2,4735	1,8661
	50	**0,2687**	3,9741	0,7823	0,8438	2,4726	1,7837
	70	**0,0781**	4,5900	1,1019	1,0850	2,6551	1,8515
	100	**0,0296**	5,4222	0,9176	1,0690	2,7025	2,3094
	200	**0,0010**	5,2161	0,9759	1,0015	3,3266	2,7567
S3	20	**0,3433**	3,1064	0,6934	0,7943	1,7168	1,3977
	30	**0,1064**	4,1189	1,0800	1,2706	2,2243	1,6837
	50	**0,0217**	4,8392	1,3437	1,3987	2,9710	2,3894
	70	**0,0018**	5,1367	0,9466	1,1318	2,8067	2,3447
	100	**0,2803**	4,8166	1,0375	1,1545	3,0032	2,9449
	200	**0,0000**	5,7457	1,4225	1,4040	3,4782	2,8781
S4	20	1,5834	0,5959	0,4150	1,8963	2,0544	1,9176
	30	2,3217	0,5049	1,0694	3,3915	3,7012	3,1709
	50	1,2988	0,9641	0,3395	2,6885	3,3955	3,3056
	70	2,0255	1,1356	0,3061	1,2796	3,2260	3,2897
	100	1,8437	2,1332	0,1057	0,7342	3,1235	3,3180
	200	1,3604	2,8627	0,2061	0,1178	3,0242	2,9092

TABLE IV. COMPUTATIONAL PERFORMANCE OF THE PROPOSED HEURISTICS

VI. CONCLUSION

In this paper, we addressed the two-machine flow shop scheduling problem under exact time lags denoted by $F2|l_j, prmu|C_j$. We have proposed a new priority rule based on the availability of the two machines named $PRTCTl_j$. This priority rule is considered as a simple heuristic which constructs a solution by adding the job which have the smallest PRTClj in each iteration. In addition, we propose six constructive heuristics adapted to our NP-Hard problem. We have realized experimental tests on different problem types for these procedures (a total of 720 test problems). The results

show the effectiveness and the practicability of H1 heuristic which include the new priority rule $PRTCTl_j$.

In the other hand, we presented a new lower bound named LB_{pmtn} based on two relaxations: A first capacity relaxation of the first machine and a second relaxation by allowing preemptive schedules.

However, further work is still necessary to try to include our promising constructive heuristic in new developed meta-heuristics and hybrid methods to the problem at hand. Thus, efforts would also be directed to develop other good lower bounds which can be useful in our tentatives to solve exactly $F2|l_j, prmu|C_j$ problem.

REFERENCES

[1] T. Ladhari and M. Haouari, *A computational study of the permutation flowshop problem based on a tight lower bound.* Computers & Operations Research. 32, 1831-1847, 2005.

[2] T. Ladhari, M. A. Rakrouki, *Heuristics and lower bounds for minimizing the total completion time in a two-machine flowshop.* International Journal of Production Economics. 122, 678691, 2009.

[3] M. Pinedo, *Scheduling: theory, algorithms, and systems.* Prentice Hall, 4^{th} edition, 2012.

[4] M. Nawaz, E.E. Enscor, I. Ham, *A heuristic algorithm for the Flowshop problem.* European J. Oper Res. 91, 160-175,1983.

[5] X. Dong, H. Huang, P. Chen, *An iterated local search algorithm for the permutation flowshop problem with total flowtime criterion.* Computers and Operations Research. 36, 1664-1669, 2009.

[6] R. Ruiz, and T.Stutzle, *A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem.* European Journal of Operational Research. 177, 2033-2049, 2007.

[7] Y. Huo , H. Li , H. Zhao, *Minimizing total completion time in two machine flow shops with exact delays.* Computers and Operations Research. 36, 2018-2030, 2009.

[8] J.Fondrevelle, *Résolution exacte de problemes d'ordonnancement de type flowshop de permutation en présence de contraintes d'écarts temporels entre opérations (in French).* PhD thesis, Institut National Polytechnique de Lorraine, France, 2005.

[9] X. Zhang, *Scheduling with Time Lags.* PhD thesis, Fudan University, Shanghai, 2010.

[10] K.R. Baker, *Introduction to Sequencing and Scheduling.*Wiley Publishing, Wiley Publishing, 1974.

[11] A.H.G Rinnooy Kan, *Machine Sequencing Problem: Classification, Complexity and Computation.*Nijhoff, The Hague, 1976.

[12] P. Brucker, S. Knust and G. Wang, *Complexity results for flow-shop problems with a single server*, European Journal of Operational Research, 165, 2, 398-407, 2005.

[13] R. H. Ahmadi, U. Bagchi, *Lower Bounds for Single-Machine Scheduling Problems*, Naval Research Logistics, 37, 967979, 1990.