Journal of Quality Engineering and Production Optimization
Vol. 1, No. 1, PP. 33-42, 2015

Solving Single Machine Sequencing to Minimize Maximum Lateness Problem Using Mixed Integer Programming

Amir Hossein Parsamanesh and Rashed Sahraeian
Department of Industrial Engineering, University College of Engineering, Shahed University, Tehran, Iran

***Corresponding Author:** Amir Hossein Parsamanesh (E-mail: a.parsamanesh@shahed.ac.ir)

Abstract- *Despite existing various integer programming for sequencing problems, there is not enough information about practical values of the models. This paper considers the problem of minimizing maximum lateness with release dates and presents four different mixed integer programming (MIP) models to solve this problem. These models have been formulated for the classical single machine problem, namely sequence-position (SP), disjunctive (DJ), linear ordering (LO) and hybrid (HY). The main focus of this research is on studying the structural properties of minimizing maximum lateness in a single machine using MIP formulations. This comparison helps us know the characteristics and priority of different models in minimizing maximum lateness. Regarding to these characteristics and priorities, while solving the lateness problem in the procedure of solving a real-world problem, we apply the lateness model which yields in solution in shortest period of time and try not to use formulations which never lead to solution for large-scale problems. Beside single machine, these characteristics are applicable to more complicated machine environment. We generate a set of test problems in an attempt to solve the formulations, using CPLEX software. According to the computational results, a detailed comparison between proposed MIP formulations is reported and discussed in order to determine the best formulation which is computationally efficient and structurally parsimonious to solve the considering problem. Among the four presented formulations, sequence-position (SP) has the most efficient computational time to find the optimal solution.*

Keywords: *Single machine scheduling, Mixed integer programming, Maximum lateness, Release date*

I. INTRODUCTION

Single machine models are important due to various reasons. Pinedo and Hadavi [1] stated that single machine environment is so simple and often considers properties that neither machines in parallel nor machines in series have. Since the single machine environment is a special case of all other ones, the results obtained from single machine models provide a basis for heuristics that are applicable to more complicated machine environments. In fact, scheduling problems in complicated machine environments are decomposed to sub-problems that deal with single machines.

Mathematical models for single machine scheduling problems have received far lower attention than the heuristic methods to that extent, now after 50 years, structural composition of models introduced by Wagner [2] and Manne [3] not only still retain their original forms but also have been consigned into neglect. This might partially due to their low efficiency and/or absence of high-tech processors. Therefore, there exist little or no model-developing techniques based on these compositions.

Academic researchers have developed specialized algorithms to solve relatively large-sized scheduling problems. Based on their work, one can develop codes to implement algorithms for a specialized problem. However, this could be complicated and time consuming. In contrast, we can use a more general modeling capability like MIP. MIP formulations represent a precise algebraic statement of the problem and solution can be obtained using commercially available optimization software (e.g., CPLEX, GAMS and LINGO).

There are only few papers focusing on optimization model formulations in the scheduling area. First, Blazewicz et al. [4] compiled a large number of mathematical formulations and provided the complexity of various production

Manuscript Received: 5 May 2014 and revised: 12 Jan 2015 *ISSN: 2423-3781*
Accepted: 14 Jan 2015

scheduling problems. Another comprehensive survey of MIP formulations was studied by Queyranne and Schulz [5]. Unlu and Mason [6] presented four different MIP formulations based on four different types of decision variables for identical parallel machine scheduling problem. Minimizing total waited completion time and maximum completion time are two objective functions considered in this study. The least computational time is delivered by time-indexed formulation. Linear ordering formulation provides much better computation time as compared to network and assignment-positional date formulations. Further, the computation times provided by network and assignment-positional date formulations are often higher than time indexed and linear ordering formulations, while network formulation delivers a relatively lower computational time than assignment-positional date formulation. Baker and Keller [7] presented six different IP formulations for the single machine tardiness problem. They created a set of especially difficult test problems to solve them with each of the formulations. It was found that sequence-position formulation provides the most computationally effective solutions. This formulation solves nearly all problems up to about 40 jobs. Herr and Goel [8] presented two integer programming formulations for a single machine family scheduling problem to minimize total tardiness. Their first formulation has $O(n2)$ binary variables whereas their second formulation has $O(n3)$ binary variables, where n denotes the number of jobs to be scheduled. Their findings indicate that despite the significant higher number of binary variables, the second formulation leads to significantly shorter solution times for problem instances of moderate size.

As Baker and Keller [7] mentioned in their study, it is not guaranteed to be effective in finding solutions using an IP approach as same as complicated and specialized algorithms. Although there are alternative approaches to the IP formulation of scheduling problem, there exists few comparative testing of those formulations. In addition, against the well-known algorithms that have been proposed and tested in the related literature, there is not enough information about the capabilities of commercial optimization packages at solving standard sequencing problems. Thus, in this research, we are interested in getting wider understanding of the different MIP formulations behavior and their performance against the different size of problems. There are many published papers that have studied heuristic and meta-heuristic algorithms to solve single machine problems, e.g., Bahalke et al. [9], Sels and Vanhoucke[10], Lu et al. [11], Moghaddam et al. [12] and Senthilvel et al. [13]. The main focus of this research is on studying the structural properties of minimizing maximum lateness in a single machine using MIP formulations. Following the standard $\alpha|\beta|\gamma$ classification scheme for scheduling problem [14], α indicates the scheduling environment, β shows the job characteristic and γ describes the objective function to be optimized. The scheduling problem studied in this paper is $1|r_j|L_{max}$, and it is unary *NP*-hard [15].

In this paper, we focus on single machine lateness problem, which is perhaps one of the most familiar basic problems in sequencing. This problem is solved using four different MIP formulations. These models are sequence-position (SP), disjunctive (DJ), linear ordering (LO) and hybrid (HY). The complexity of modeling in these formulations is related to implementation of different concepts for calculating the minimization of maximum lateness. The concept of SP for solving this problem is to determine the position of each job in a sequence that leads to minimization of maximum lateness. DJ applies another approach to calculate the mentioned objective. It uses precedence of jobs over the others to minimize the maximum lateness. LO uses the linear ordering concept that was first time introduced in 1990. Finally, the HY model uses the both sequence-position and precedence approach to form a formulation. The feature which distinguishes among these formulations is the key variable(s) used in their structure. The key variables used in this paper are sequence-position and precedence variable. These key variables were first time introduced by Wagner [2] and Manne [3]. Wagner [2] was the first researcher who formulated a model to minimize the makespan in the three machine flow shop problem using a key binary variable named sequence-position. Another early paper on the topic is proposed by Manne [3] which used precedence variable as a key variable to minimize the makespan in a job shop problem.

In order to evaluate the capability of each of four MIP formulations, special test problems are created to focus on conditions under which the solution procedure is most severely challenged. Then, according to the obtained results, formulations are compared based on the computational time, number of nodes, constraints, variables and number of problems that they were able to solve within a limited time.

The rest of this study is organizing as follows: in the following section, various formulations for solving minimizing maximum lateness have been introduced. Section III explains how test problems have been generated to test the formulations performance. Numerical examples have been studied in section IV to evaluate the presented formulations and, finally, conclusion is presented in section V.

II. FORMULATION OF LATENESS PROBLEM

In this section, we propose four MIP formulations that are used to solve a single machine lateness problem with release dates. In these formulations, parameters such as p_j, d_j and r_j are assumed to be given. p_j, d_j and r_j are defined as processing time, due date and release date of job j, respectively. There are two key variables used in the following formulations that are sequence-position and precedence variables. Sequence-position variable (x_{ik}) is a binary variable that is 1 if job i is assigned to position k or 0 otherwise. Also, precedence variable (y_{ij}) is a binary variable that is 1 if job i is scheduled before job j and is 0 otherwise. Z is a common variable in minimizing maximum lateness formulations. In the following formulations, N indicates the number of jobs and positions.

$L_j =$ lateness of job j=$C_j - d_j$

A. *Sequence-position Model (SP)*

In this formulation, the key variable is sequence-position that determines which job is chosen for each position.

Main variable:

$x_{ik} = 1$ if job i is kth in sequence

$$Min\,\{\max_{k\in N}(L_k)\} \equiv Min\,Z \tag{1}$$

s.t:

$$\sum_{i=1}^{N} x_{ik} = 1, \qquad \forall k \in N \tag{1.2}$$

$$\sum_{k=1}^{N} x_{ik} = 1, \qquad \forall i \in N \tag{1.3}$$

$$\sum_{i=1}^{N} p_i \sum_{u=0}^{k-1} x_{iu} + \sum_{u=1}^{k} g_u \geq \sum_{i=1}^{N} r_i x_{ik}, \qquad \forall k \in N \tag{1.4}$$

$$\sum_{i=1}^{N} p_i \sum_{u=1}^{N} x_{iu} + \sum_{u=1}^{k} g_u - \sum_{i=1}^{N} d_i x_{ik} \leq Z, \qquad \forall k \in N \tag{1.5}$$

$$g_u, Z \geq 0 \quad \forall\, u \in N \tag{1.6}$$

$$x_{ik} \in \{0,1\} \quad \forall\, i,k \in N \tag{1.7}$$

In this formulation, the constraint sets (1.2) and (1.3) ensure that each job is exactly assigned to one position and each position is assigned just to one job. The constraint (1.4) calculates the completion time of the job at kth position and constraint (1.5) is utilized to ensure the linear form of objective function (1.1). Finally, the constraint sets (1.6) and (1.7) are the non-negativity and integrality constraints, respectively.

B. *Disjunctive Model (DJ)*

Balas [16] presented the first research on formulating scheduling problem using disjunctive constraints and this formulation was studied by Queyranne and Wang [17]. This model has two key variables that are y_{ij} and s_i. y_{ij} and s_i are binary and integer variables, illustrating the precedence and start time of the jobs, respectively.

Main variables:
$y_{ij} = 1$ if job j follows job i, and zero otherwise
$s_i =$ start time of job i

$$Min = \{\max_{k \in N}(L_k)\} \equiv Min\ Z \tag{2}$$

s.t:

$$s_i + p_i \le s_j + M(1 - y_{ij}), \qquad \forall i, j \in N\ , i < j \tag{2.2}$$

$$s_j + p_j \le s_i + My_{ij}, \qquad \forall i, j \in N\ , i < j \tag{2.3}$$

$$s_j \ge r_j, \qquad \forall j \in N \tag{2.4}$$

$$s_j + p_j - d_j \le Z, \qquad \forall j \in N \tag{2.5}$$

$$s_j, Z \ge 0, \quad \forall j \in N \tag{2.6}$$

$$y_{ij} \in \{0,1\}, \qquad \forall i, j \in N \quad , i < j \tag{2.7}$$

The constraint sets (2.2) and (2.3) ensure that either job j is processed before job k or job k is processed before job j for any pair of jobs. The constraint set (2.4) guarantees that start time of each job is equal to or greater than its released date. The constraint sets (2.5) and (2.1) compute and minimize the maximum lateness function and the constraint sets (2.6) and (2.7) are the non-negativity and integrality constraints, respectively. In this formulation, M is a big number that can be taken to be greater than the sum of processing time of all jobs and maximum value of the release dates.

C. *Linear Ordering Model (LO)*

Linear ordering formulation was first time introduced by Dyer and Wolsey [18] and continued to be studied by Nemhauser and Savelsbergh [19]. There exists precedence variable just like the previous model.

Main variable:
$y_{ij} = 1$ if job i is scheduled before job j

$$Min = \{\max_{k \in N}(L_k)\} \equiv Min\ Z \tag{3.1}$$

s.t:

$$y_{ij} + y_{ji} = 1, \qquad \forall i, j \in N\ , i \ne j \tag{3.2}$$

$$y_{ij} + y_{jk} + y_{ki} \le 2 \qquad \forall i, j, k \in N, i \ne j, i \ne k, j \ne k \tag{3.3}$$

$$f_{ij} \le My_{ij} \qquad \forall i, j \in N \tag{3.4}$$

$$f_{ij} - g_j \le 0 \qquad \forall i, j \in N \tag{3.5}$$

$$g_j - f_{ij} \le M(1 - y_{ij}) \qquad \forall i, j \in N \tag{3.6}$$

$$\sum_{i=1}^{N} p_i y_{ij} + \sum_{i=1}^{N} f_{ij} \ge r_j \qquad \forall j \in N \tag{3.7}$$

$$p_j + \sum_{i=1}^{N} p_i y_{ij} + \sum_{i=1}^{N} f_{ij} - d_j \le Z\ \forall i, j \in N\ , i \ne j \tag{3.8}$$

$$f_{ij}, g_j, Z \geq 0, \quad \forall i,j \in N \tag{3.9}$$

$$y_{ij} \in \{0,1\}, \quad \forall i,j \in N \tag{3.10}$$

The constraint set (3.2) ensures job i is processed before job j or job j is processed before job i. constraint set (3.3) ensures the linear order between three jobs. The constraint sets (3.4), (3.5) and (3.6) have been used to linearize the multiplication of $g_j.y_{ij}$ in order to be used in the constraint set (3.7) to ensure that start time of job j is greater than or equals to its release date. Constraint sets (3.8) and (3.1) give the minimized maximum lateness. Finally, the constraint sets (3.9) and (3.10) are the non-negativity and integrality constraints, respectively. In this formulation, parameter M is a big number.

D. Hybrid model (HY)

Name of this formulation implies that it is a kind of synthetic model that consists of precedence and sequence-position variables. A set of constraint is needed to ensure that the two sets of variables are consistent.

Main variables:
$x_{ik} = 1$ job i is kth in sequence
$y_{ij} = 1$ if job i is scheduled before job j

$$Min = \{\max_{k \in N}(L_k)\} \equiv Min\ Z \tag{4.1}$$

s.t:

$$\sum_{i=1}^{N} x_{ik} = 1, \quad \forall k \in N \tag{4.2}$$

$$\sum_{i=k}^{N} x_{ik} = 1, \quad \forall i \in N \tag{4.3}$$

$$x_{jk} + \sum_{u=0}^{k-1} x_{iu} \leq 1 + y_{ij}, \quad \forall k, j, i \in N \quad , i \neq j \tag{4.4}$$

$$\sum_{i=1}^{N} p_i \sum_{u=0}^{k-1} x_{iu} + \sum_{i=1}^{N} g_i \geq \sum_{i=1}^{N} r_i x_{ik}, \quad \forall k \in N \tag{4.5}$$

$$p_j + \sum_{i=1}^{N} p_i y_{ij} + \sum_{i=1}^{N} g_i - d_j \leq Z \quad \forall i,j \in N \ , i \neq j \tag{4.6}$$

$$g_i, Z \geq 0 \quad \forall\, i \in N \tag{4.7}$$

$$x_{ik}, y_{ij} \in \{0,1\} \quad \forall\, i,j,k \in N \tag{4.8}$$

In this formulation, the constraint sets (4.2) and (4.3) ensure that each job is exactly assigned to one position and each position is assigned just to one job. Constraint set (4.4) determines the precedence of job i and j for each pair of jobs based on the position of each one. The constraint (4.5) calculates the completion time of the job at kth position and the constraint sets (4.6) and (4.1) give the minimized maximum lateness. Finally, the constraint sets (4.7) and (4.8) are the non-negativity and integrality constraints, respectively.

III. TEST PROBLEMS AND EXPERIMENTS

To evaluate the capability of an MIP formulation, it should be tested using difficult problem instances. Also, generating data sets in the condition that the solution procedure is most severely challenged should be considered [20]. We adopt the idea of generating data sets based on the studies presented by Nemhauser and Savelsbergh [19] and Abdul-Razaq et al. [21].

We have generated integer processing time (p_j) for each job j from a uniform distribution between 1 and 100. As Baker and Keller [7] stated that there are two reasons for choosing processing time between 1 and 100. Firstly, one is historical consistency. In fact, most of the recent computational tests in scheduling problems generate data like processing time from a uniform distribution on [1, 100] or a normal distribution with a mean of 100 and a standard deviation as large as 25. Secondly, it is so important that according to the sample size, range of 100 possibilities will be reasonable. On the other hand, it is better to use data that are representative of realistic scheduling problem. Sampling in small interval may create many ties that is far from the reality and yields in inappropriate conclusion.

The due date, d_j, is another integer parameter generated from a uniform distribution $[P(L-R/2),P(L+R/2)]$, where P is the sum of processing times of all jobs and the two parameters L and R are the relative measures for the location and range of the distribution, respectively. The release date of job j (r_j) is the last needed data which is generated from a uniform distribution $[0, QP]$. P is the sum of processing times of all jobs and Q determines the range of the distribution.

According to the mentioned points, data sets have been generated in sizes of 10, 15, 20, 30, 40 and 50 jobs. Also, parameters L, R and Q have been set to 0.5, 0.8 and 0.4, respectively. Our experiments involved solving each test problem using CPLEX software. We set 3600 seconds as a limit for computational time and apparently, if the optimal solution is not found in this limit, the process will be terminated.

IV. COMPUTATIONAL RESULTS

To compare the different proposed formulations, we look at the number of explored nodes, computational time for solving test problems in 3600s and the number of test problems solved in 3600s. Results of the runs are summarized in Table I. For each formulation, a number of problems solved within the time limit, average, minimum and maximum time needed to solve the test problems have been displayed. In addition, the average number of nodes explored while solving the test problems has been illustrated.

TABLE I. SUMMERY OF COMPUTATIONAL RESULTS

Problem size	MIP formulation	Problems solved	Average time (s)	Minimum time (s)	Maximum time (s)	Average nodes
10	SP	40	1.10475	0.23	1.86	104.9
10	DJ	40	1.589	0.91	2.51	2528.9
10	LO	40	1.81475	0.84	6.79	2367.3
10	HY	40	4.851	1.98	8.99	4243.9
15	SP	40	1.2455	0.58	2.61	672.6
15	DJ	40	126.254	1.39	2276.6	1610132
15	LO	40	182.605	3.42	964.45	162218
15	HY	35	129.132	36.07	770.07	37783.1
20	SP	40	1.88525	0.72	12.78	3830.2
20	DJ	14	485.25	3.65	2141.58	2855943
20	LO		-	-	-	-
30	SP	40	3.01135	1.11	11.75	1926.9
40	SP	38	10.6510	2.73	11.75	3438.3
50	SP	38	47.8723	6.08	326.51	11268

We first used test problems containing 10 jobs. The experiments show that all of the four formulations are capable of solving problems in the reasonable time and found the optimal solutions in the time limit. Among these formulations, HY consumes more time to find the optimal solutions. Unlike the HY, SP had the best performance and found the optimal solutions in the shortest time comparing with other formulations. Also, computed variance of time spent to find the optimal solution indicates that HY and SP have the most and the least variance, respectively.

In the next step, the number of jobs raises to 15. In this step, comparison of average time needed by each formulation to find the optimal solution shows that the SP has the best performance and its average time did not increase significantly comparing with the previous step.

Unlike SP, other three formulations have a significant increase in average time by raising the number of jobs to 15. Results show that HY formulation is weak to solve the test problems compared to the other ones. Also, it could not find optimal solution of five problems in the time limit. Therefore, we dropped it from consideration and proceeded to larger problems.

In size of 20 jobs, SP had acceptable performance just as same as before. However, the other two formulations were extremely weak to solve the problem and find the optimal solution. LO could not solve any of the test problems and DJ was able to solve fewer than half of the problems. Therefore, we dropped them from consideration and proceeded to larger problems to observe the performance of SP in larger size of problems.

Next set of experiments used problem sizes of 30, 40, and 50. For problem size of 30, SP solved all of 40 test problems and its average spent time has an increase comparing the previous steps. By increasing the number of jobs to 40, SP could not solve two test problems and its average time of finding optimal solution is more than three times greater than the computed value in the size of 30. Finally, at problem size of 50 jobs SP could find the optimal solution of 38 test problems in the time limit. As indicated in Table I, the average time of solving test problems of size 50 using SP is significantly greater than the value obtained in smaller sizes. Also, during the computational test the least average number of nodes has been explored while solving the SP. Comparison of average computational time of different models for 10, 15 and 20 jobs is shown in Fig. 1.

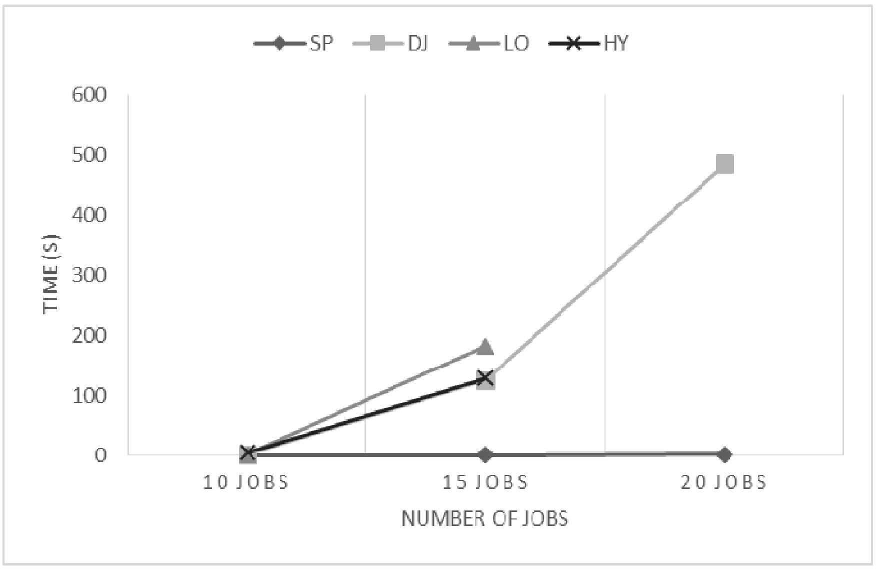

Fig. 1: Comparison of average computational time of different models for 10, 15 and 20 jobs (the LO and HY model were unable to solve problems with 20 and more jobs)

TABLE II. FORMULATION SIZES FOR FOUR FORMULATIONS

	Number of jobs	Number of Constraints	Number of Variables	Number of Integer variables
SP	10	40	112	11
LO	10	1320	212	111
DJ	10	110	57	11
HY	10	940	202	11
SP	15	60	242	16
LO	15	4080	467	241
DJ	15	240	122	16
HY	15	3210	452	16
SP	20	80	422	21
LO	20	9240	822	421
DJ	20	420	212	21
HY	20	7680	802	21

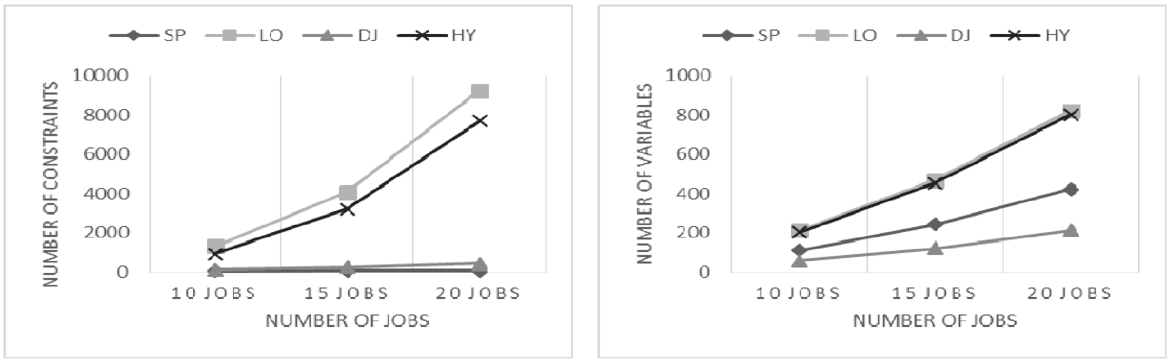

Fig. 2: Comparison of number of constraints and variables of different models for 10, 15 and 20 jobs

Table II displays the size of the MIP formulations for all of four alternatives for n = 10, 15 and 20. Table II shows that SP and DJ have the smallest number of constraints and variables, respectively. In spite of having small number of variables, the feature of DJ does not yield in finding optimal solution in an efficient computational time. In contrast, the LO has the largest number of constraints and variables and probably that is why it is incapable in solving lager problems. Comparison of number of constraints and variables of different models for 10, 15 and 20 jobs is depicted in Fig. 2.

Results show that for problems with large-size, using SP formulation can be reliable because SP could find the optimal solution of problems with 50 jobs in short period of time. Since, solving minimization of maximum lateness by the three formulations LO, DJ and HY for large problems is so time consuming and in some cases finding the exact solution is impossible; so, it is more preferable not to use these there exact formulations for large-size problem.

V. CONCLUSION

We have tested four different MIP formulations for minimizing maximum lateness in a single machine using software CPLEX. In this study, it is indicated that for minimizing maximum lateness it is not enough to use a MIP formulation to solve the problem; also, it is important to examine which MIP formulation has been considered. Among the four presented formulations, the formulation that used sequence-position variables, namely sequence-position formulation (SP), has the most efficient computational time to find the optimal solution.

We know that specific algorithms have been proposed for solving the minimizing maximum lateness problem and they are shown to be more effective than solving MIP formulations. But, it is important to notice that adopting these algorithms for other problems in the same domain is not easy. In contrast, MIP formulations could be easily solved using optimization solvers. As a future research, the presented MIP formulations can be compared with other formulations containing additional restrictions, like precedence constraint, or for more complex machine environments. Further, the problem can be formulated using other concepts like timed-indexed and tour constraints approach.

REFERENCES

1. Pinedo, M., & Hadavi, K. Scheduling: Theory, Algorithms and systems development, in operations research proceedings 1991, W. Gaul, et al., Editors. 1992, Springer Berlin Heidelberg. p. 35-42.
2. Wagner, H.M.(1959). An integer linear-programming model for machine scheduling. Naval Research Logistics Quarterly, 6(2): p. 131-140.
3. Manne, A.S.(1960). On the job-shop scheduling problem. Operations Research, 8(2) pp. 219--223.
4. Blazewicz, J., M. Dror, & Weglarz, J.(1991). Mathematical programming formulations for machine scheduling: A survey. *European Journal of Operational Research*, 51(3) pp. 283-300.
5. Queyranne, M., & Schulz, A.S. Polyhedral approaches to machine scheduling.Technical report 408/1994. Department of Mathematics, Technical University of Berlin, Berlin, Germany, 1994.
6. Unlu, Y. and Mason, S.J. (2010). Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems. *Computers &* Industrial *Engineering*, 58(4) pp. 785-800.
7. Baker, K.R. & B. Keller. (2010). Solving the single-machine sequencing problem using integer programming. Computers & Industrial Engineering, 59(4) pp. 730-735.
8. Herr, O., and Goel, A. (2014). Comparison of two integer programming formulations for a single machine family scheduling problem to minimize total tardiness. Procedia CIRP, 19(0) pp. 174-179.
9. Bahalke, U., et al. (2010). Genetic and tabusearchalgorithmsforthesingle machinescheduling problemwith sequence-dependent set-uptimes and deteriorating jobs. International Journal of Engineering, 23 pp. 227-234
10. Sels, V. & Vanhoucke, M. (2012). A hybrid genetic algorithm for the single machine maximum lateness problem with release times and family setups. Computers & Operations Research, 39(10) pp. 2346-2358.
11. Lu, C.-C., Lin, S-W., and Ying ,K.-C. (2012). Robust scheduling on a single machine to minimize total flow time. Computers & Operations Research, 39(7) pp. 1682-1691.
12. Moghaddam, A., F. Yalaoui, L.& Amodeo. (2015). Efficient meta-heuristics based on various dominance criteria for a single-machine bi-criteria scheduling problem with rejection. *Journal of Manufacturing Systems*, 34(0) pp. 12-22.
13. Senthilvel, A.N., Maheswari, S.U., and Hemamalini, T. (2014). Heuristic robust algorithm to optimize sequencing of jobs on a single machine. Procedia Materials Science, 5(0) p. 1473-1481.
14. Graham, R.L., et al.(1979). Optimization and approximation in deterministic sequencing and scheduling: a survey, in Annals of Discrete Mathematics, E.L.J. P.L. Hammer and B.H. Korte, Editors, Elsevier. p. 287-326.
15. Lenstra, J.K., Rinnooy Kan, A.H.G., and Brucker, P. (1977) . Complexity of machine scheduling problems, in Annals of Discrete Mathematics, E.L.J.B.H.K. P.L. Hammer and G.L. Nemhauser, Editors, Elsevier. p. 343-362.
16. Balas, E., On the facial structure of scheduling polyhedra, in Mathematical Programming Essays in Honor of George B. Dantzig Part I, R.W. Cottle, Editor. 1985, Springer Berlin Heidelberg, p. 179-218.
17. Queyranne, M., and Wang,Y. (1991). Single-machine scheduling polyhedra with precedence constraints. Math Operation Research, 16(1) p. 1-20.
18. Dyer, M.E. and Wolsey, L.A. (1990). Formulating the single machine sequencing problem with release dates as a mixed integer program. Discrete Applied Mathematics, 26(2–3) p. 255-270.

19. Nemhauser, G.L. and Savelsbergh, M.W.P. A cutting plane algorithm for the single machine scheduling problem with release times, in Combinatorial Optimization, M. Akgül, H. Hamacher, and S. Tüfekçi, Editors. 1992, Springer Berlin Heidelberg. p. 63-83.

20. Hall, N.G.H., and Posner, M.E.(2001) Generating experimental data for computational testing with machine scheduling applications. Operation Research, 49(6) p. 854-865.

21. Abdul-Razaq, T.S., Potts, C.N., and Van Wassenhove, L.N. (1990) A survey of algorithms for the single machine total weighted tardiness scheduling problem. Discrete Applied Mathematics, 26(2–3): p. 235-253.