"Improved Exact Methods for Solving No-Wait Flowshop Scheduling Problems With Due Date Constraints"

By K.-C Ying, C.-C. Lu, and S.-W. Lin

published in *IEEE Access,* Vol. 6, pp. 30702-30713, 2018.

Digital Object Identifier: 10.1109/ACCESS.2018.2834954

For this article, the corresponding author Shih-Wei Lin (swlin@mail.cgu.edu.tw) and the co-first author is Chung-Cheng Lu (jasoncclu@nctu.edu.tw).

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 27, 2018, accepted May 2, 2018, date of publication May 10, 2018, date of current version June 26, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2834954

Improved Exact Methods for Solving No-Wait Flowshop Scheduling Problems With Due Date Constraints

KUO-CHING YING[1], CHUNG-CHENG LU[2], AND SHIH-WEI LIN[3,4,5]

[1]Department of Industrial Engineering and Management, National Taipei University of Technology, Taipei 106, Taiwan
[2]Department of Transportation and Logistics Management, National Chiao Tung University, Hsinchu 30010, Taiwan
[3]Department of Information Management, Chang Gung University, Taoyuan City 333, Taiwan
[4]Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan City 333, Taiwan
[5]Department of Industrial Engineering and Management, Ming Chi University of Technology, Taipei 243, Taiwan

Corresponding authors: Chung-Cheng Lu (jasoncclu@nctu.edu.tw) and Shih-Wei Lin (swlin@mail.cgu.edu.tw)

The work of K.-C. Ying was supported by the Ministry of Science and Technology under Grant MOST106-2221-E-027-085. The work of S.-W. Lin was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST105-2410-H-182-009-MY2/MOST106-2632-H-182-001 and in part by the Linkou Chang Gung Memorial Hospital under Grant CMRPD3G0011/CARPD3B0012.

ABSTRACT The no-wait flowshop scheduling problem with hard due date constraints is critical to operations in many industries, such as plastic, chemical, and pharmaceutical manufacturing. However, to date, there is a lack of effective optimization algorithms for this NP-hard problem. This paper develops a new mixed integer linear programming (MILP) model and a two-phase enumeration algorithm to improve the best-so-far exact methods for solving this problem with the objective of minimizing the makespan. A comprehensive computational experiment is performed to compare the performances of the discussed exact methods. The computational results demonstrate that the proposed MILP model and the two-phase enumeration algorithm significantly outperform the best-so-far optimization methods, and the (sub-) optimal solutions to several unsolved instances from the literature are reported.

INDEX TERMS Scheduling, no-wait flowshop, due date constraints, mathematical model.

I. INTRODUCTION

The no-wait flowshop scheduling problem (NWFSP) is one of the most important variants of the flowshop scheduling problem (FSP), in which no in-process waiting time is permitted between successive operations of each job. For technological reasons, once a job/product has begun on the first machine, it must be continued until it is completed on the final machine without any interruption on or between machines. The restriction of the no-wait process is ubiquitous in various scheduling problems, such as flight scheduling problems [1], train scheduling problems [2], surgery scheduling problems [3], and food-processing scheduling problems [4]. For example, in the canned food processing industry, once the food is precooked and while it is still hot [5], no-wait constraints are required between the successive operations of adding liquid sugar, gas exhausting, sealing, sterilizing, and refrigerating. In general, applications of NWFSPs are commonly found in metal, plastic, chemical, concrete, transportation, medical, petroleum refining, pharmaceutical manufacturing, and food-processing industries [4].

Owing to its complex nature and wide range of practical applications, research interest in NWFSPs has increased since 1970s. In the past five decades, researchers have proposed hundreds of algorithms to solve NWFSPs. These algorithms fall into two categories, heuristic algorithms and exact methods, whose use depends on the size of the problem. The excellent surveys of Hall and Sriskandarajah [4], Allahverdi [6], and Nagano and Miyata [7] detail earlier studies and applications of NWFSPs. To increase the utilization of machines, the makespan is one of the most used criteria in scheduling literature [8]. On the other hand, in the aforementioned industries, NWFSPs with due date constraints are common requirements as their manufacturing processes must be completed with no delay [9]. Although the NWFSP with due date constraints has attracted considerable attention, only a small fraction of relevant studies have treated due date as hard constraints, such that the violation of due

dates is forbidden. To the best of our knowledge, the NWFSP with hard due date constraints is common in practice, but still lacks effective exact methods for solving it.

For practical reasons, this paper focuses on the NWFSP with hard due date constraints that minimizes the makespan. Using the conventional three-field notation, this problem can be written as $F_m|nwt, d_j|C_{\max}$, where F_m is a flowshop with m machines; nwt specifies the no-wait restriction; d_j are the due date constraints, and $C_{\max}$ indicates that the objective is to minimize the makespan. Samarghandi [10] first studied this problem in 2015. He developed a mixed integer linear programming (MILP) model, $Model_S$, and a particle swarm optimization (PSO) algorithm to solve the $F_m|nwt, d_j|C_{\max}$ problem. Computational results showed that the mathematical model could solve a number of small-instance test problems to optimality, while the PSO algorithm yielded high-quality solutions to many test problems in reasonable time.

More recently, Samarghandi and Behroozi [11] developed an MILP model, two quadratic mixed integer programming (QMIP) models, two constraint programming (CP) models, and an enumeration algorithm for solving the $F_m|nwt, d_j|C_{\max}$ problem. The MILP model, $Model_{SB}$, is based on $Model_S$ of Samarghandi [10] and uses the decision variable that was defined by Aldowaisan and Allahverdi [12]. A thorough computational experiment was conducted to compare the performances of the five discussed mathematical models, the no-wait version of Manne's model, $Model_M$, and the enumeration algorithm, $Algorithm_{SB}$. $Model_M$ is based on the Manne model [13], which is the best MILP formulation of the permutation FSP, as it includes no-wait and due date constraints. Computational results revealed the superiority of $Model_{SB}$ over the other formulations, and $Algorithm_{SB}$ outperformed the other algorithms when executed using an IBM ILOG CPLEX. Moreover, the computational results of Samarghandi and Behroozi [11] revealed that finding a feasible solution to the $F_m|nwt, d_j|C_{\max}$ problem becomes increasingly difficult as the size of the problem grows. To the best of the authors' knowledge, in existing literature, only the above two studies (Samarghandi [10] and Samarghandi and Behroozi [11]) addressed the $F_m|nwt, d_j|C_{\max}$ problem, in which the $Model_M$, $Model_{SB}$, and $Algorithm_{SB}$, as proposed by Samarghandi and Behroozi [11], are the best-so-far exact methods.

Given the significance of the problem in both scheduling theory and industrial applications, more efficient exact methods for finding optimal solutions to the $F_m|nwt, d_j|C_{\max}$ problem must be developed. Therefore, in this work, a new MILP model, namely $Model_{YLL}$, and a two-phase enumeration algorithm, namely $Algorithm_{YLL}$, are developed to improve upon the $Model_{SB}$ and $Algorithm_{SB}$ of Samarghandi and Behroozi [11] for solving the $F_m|nwt, d_j|C_{\max}$ problem. The rest of this paper is organized as follows. Section 2 defines the $F_m|nwt, d_j|C_{\max}$ problem that is considered herein. Section 3 formulates the $Model_{YLL}$ and describes the $Algorithm_{YLL}$. Section 4 reports the computational experiments on the compared exact methods. Section 5 presents concluding remarks and directions for future research.

II. DESCRIPTION AND FORMULATION OF PROBLEM

The $F_m|nwt, d_j|C_{\max}$ problem, considered herein, is defined as follows. A set, $\mathbf{J} = \{J_1, \ldots, J_n\}$, of n jobs are to be processed on a set, $\mathbf{M} = \{M_1, M_2, \ldots, M_m\}$, of m machines following a single flow pattern. Every operation associated with job J_j ($j = 1, 2, \ldots, n$) on machine M_i ($i = 1, 2, \ldots, m$) is a denominated operation, $o_{j,i}$, which requires a processing time, $p_{j,i}$, and no additional setup time is required. Each machine can process at most one job at a time, and every job can be processed by no more than one machine at any moment. All jobs are assumed to be available for processing at the beginning of the planning horizon, and all machines are persistently available.

To meet the technological requirements of manufacturing environments, once the processing of a job has begun, its subsequent operations must immediately follow the preceding one without interruption either on or between consecutively used machines until completion. To satisfy the no-wait constraint, the start time of the job to be processed on the first machine may be postponed. Let $\Pi = (\pi_0, \pi_1, \ldots, \pi_n)$ be a feasible permutation of jobs, where π_k ($k = 1, \ldots, n$) is the job in position k of Π, and π_0 is a dummy job whose processing times on all machines are zero. The objective is to find the optimal permutation for the processing of all jobs with due dates as hard constraints that minimizes the makespan, $C_{\max}(\Pi)$. This problem that involves three or more machines is a member of the set of strongly NP-hard problems [10]. Let t_{π_{k-1},π_k} be the time-span between the completion times of the two adjacent jobs in positions $k-1$ and k on the last machine. $C_{\max}(\Pi)$ can be calculated using the following equation [14].

$$C_{\max}(\pi) = \sum_{k=1}^{n} t_{\pi_{k-1},\pi_k} \tag{1}$$

where $t_{\pi_{k-1},\pi_k} = \max_{i=1,\ldots,m} \left\{ \sum_{i'=i}^{m} (p_{\pi_k,i'} - p_{\pi_{k-1},i'}) + p_{\pi_{k-1},i} \right\}$.

Computing the time-span (t_{π_{k-1},π_k}) for all possible job pairs and setting it as the traveling cost c_{π_{k-1},π_k} between two consecutively visited cities reduces the NWFSP problem without due date constraints to a special case of the $(n+1)$-city asymmetric travelling salesman problem (ATSP) in polynomial time [14], [15]. Let $\mathbf{V}$ be a set of $n+1$ vertexes/cites and $\mathbf{A}$ be a set of arcs among them; the reduced ATSP can be defined as follows. Given a complete directed graph, $\mathbf{G} = (\mathbf{V}, \mathbf{A})$, find a directed Hamiltonian cycle (φ) of the $n+1$ vertexes/cites in $\mathbf{G}$ that starts from the dummy vertex/city π_0; passes through each vertex/city exactly once, and finally returns to vertex/city π_0, aiming to minimize the possible tour cost, $C(\varphi) = \sum_{k=1}^{n} c_{\pi_{k-1},\pi_k}$. The makespan of a feasible

Step 1: Sort the jobs according their due dates.

Step 2: Get the first 10-jobs from the sorted jobs.

Step 3: Two-phase enumeration method.

Step 3.1: Phase I: Apply the enumeration method to 10 jobs.

If no feasible solution can be found then

Output "NFS" (no feasible solution).

Terminate two-phase enumeration method.

End If

Step 3.2: Phase II: Apply the Enumeration method to all the jobs.

Step 3.2.1: Put the remaining jobs to the partial job permutation obtained in Step 3.1.

Step 3.2.2: Run the enumeration method for all jobs until all possible permutation is tested or timeout or optimal solution has been found.

Step 3.2.3: Output the (sub-) optimal solution obtained in the Phase II.

FIGURE 1. The pseudocode of the $Algorithm_{YLL}$.

solution to an NWFSP is equivalent to the tour cost of a directed Hamiltonian cycle.

III. NEW MILP MODEL AND TWO-PHASE ENUMERATION ALGORITHM

In this section, a new MILP mathematical model with respect to the reduced ATSP version of the NWFSP is formulated for solving the $F_m|nwt, d_j|C_{\max}$ problem. The proposed $Algorithm_{YLL}$ is described in detail.

A. NEW MILP MODEL

Let s_j be the processing start time of job J_j; the reduced ATSP version of the $F_m|nwt, d_j|C_{\max}$ problem can be reformulated as the following MILP model.

$$\text{Minimize } z = \sum_{j=0}^{n}\sum_{j'=0}^{n} c_{j,j'}x_{jj'} \tag{2}$$

$$\text{subject to } \sum_{\substack{j'=1\\ j'\neq j}}^{n} x_{jj'} = 1, \quad j = 0, 1, \ldots, n \tag{3}$$

$$\sum_{\substack{j=0\\ j\neq j'}}^{n} x_{jj'} = 1, \quad j' = 1, \ldots, n \tag{4}$$

$$s_{j'} + \sum_{i=1}^{m} p_{j',i} \geq s_j + \sum_{i=1}^{m} p_{j,i} + x_{jj'}c_{j,j'} + M(x_{jj'} - 1), \quad j, j' = 1, \ldots, n \text{ and } j \neq j' \tag{5}$$

$$s_j + \sum_{i=1}^{m} p_{j,i} \leq d_j, \quad j = 0, 1, \ldots, n \tag{6}$$

$$s_j \geq 0, \quad j = 1, 2, \ldots, n \tag{7}$$

$$x_{jj'} \in \{0, 1\}, \quad j = 0, 1, \ldots, n; \; j' = 1, \ldots, n \text{ and } j \neq j' \tag{8}$$

The objective function (2) is to find a minimum Hamiltonian tour in a complete directed graph $\mathbf{G}$ with $n+1$ vertexes. Constraint set (2) ensures that exactly one city is directly visited after city $j(j = 1, 2, \ldots, n)$ is visited. Constraint set (4) ensures that exactly one city is directly visited before city $j'(j' = 1, 2, \ldots, n)$ is visited. Constraint set (5) defines the start time of each city. Constraint set (6) guarantees that a job cannot be finished after its due date. Constraint sets (7) and (8) define the continuous variable s_j and the binary variable $x_{jj'}$.

B. PROPOSED TWO-PHASE ENUMERATION ALGORITHM

This work proposes a two-phase enumeration algorithm that improves upon the enumeration algorithm, $Algorithm_{SB}$, that was developed by Samarghandi and Behroozi [11] for finding the (sub-) optimal solution to the $F_m|nwt, d_j|C_{\max}$ problem. Figure 1 presents the pseudocode of the proposed $Algorithm_{YLL}$. The idea of designing the $Algorithm_{YLL}$ to improve upon $Algorithm_{SB}$ is motivated by the following two observations.

Observation 1: When an enumeration algorithm is used to solve large problems in limited computing time, the probability that the positions of some jobs will never be changed after the initial permutation is determined is high. Therefore, the initial permutation has a strong impact on the quality of the solution that is obtained using the enumeration algorithm.

TABLE 1. Comparison of the MILP models for the $F_m|nwt, d_j|C_{max}$ problem.

Model	Number of binary variables	Number of continuous variables	Total number of constraints	Number of due date constraints
$Model_{SB}$	n^2	mn	$(m+1)n^2+(m+4)n+1$	n
$Model_{M}$	$\frac{(m-2)n(n-1)}{2}$	$\frac{mn(n+1)}{2}+1$	$\frac{mn(n+1)}{2}+n+1$	n
$Model_{YLL}$	n^2	n	n^2+3n	n

TABLE 2. Computational results of MILP models for the $F|nwt|C_{max}$ problem.

Problem	Size	$Model_{M}$ without no-wait constraints	$Model_{SB}$ without no-wait constraints	$Model_{YLL}$ without no-wait constraints
Sam01	7×7	**7,705;0.2***	**7,705;1.9**	**7,705;0.2**
Sam02	8×8	**9,372;0.4**	**9,372;4.3**	**9,372;0.5**
Sam03	8×9	**9,690;0.3**	**9,690;9.8**	**9,690;0.7**
Sam04	10×6	**9,159;1.5**	**9,159;410.3**	**9,159;0.3**
Sam05	11×5	**8,142;4.5**	8,142	**8,142;0.3**
Sam06	12×5	**8,866;25.1**	8,884	**8,866;0.4**
Sam07	13×4	**8,242;62.4**	8,299	**8,242;0.1**
Sam08	14×4	**9,159;414.0**	9,195	**9,159;0.2**
Sam09	16×6	13,374	13,374	**13,330;0.3**
Sam10	16×7	8,945	8,908	**8,869;0.8**
Sam11	17×5	11,047	11,753	**10,950;0.2**
Sam12	18×9	8,949	9,009	**8,824;0.2**
Sam13	19×8	17,935	19,065	17,428
Sam14	20×10	30,064	33,315	**29,318;0.2**
Optimality proved		57.14%	28.57%	92.86%
Average GAP		0.64%	2.47%	0.00%

*: objective function value; computational time

Observation 2: If jobs with near due dates are processed as early as possible but still violate their due date constraints, then later processing of the same jobs will necessarily cause the violation of the due-date constraint. Therefore, some jobs with near due dates can be enumerated to obtain a favorable initial solution or to verify as soon as possible that the problem has no feasible solution.

The following lemmas and corollaries can be used to eliminate the need to perform unnecessary solution evaluations or to confirm as soon as possible that the problem does not have any feasible solution.

Lemma 1: The algorithm that was proposed by Samarghandi and Behroozi [11] is utilized to obtain the contribution matrix C, which provides the contribution of each job to the makespan if it is performed after a specified job in the sequence. The contribution matrix C is an $(n+1)\times n$ matrix, which can be computed as follows.

$$C=\begin{bmatrix} c_{01} & \cdots & c_{0n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix}$$

where $c_{0j}=\sum_{i=1}^{m} p_{j,i},\ j=1,\ldots,n;\ c_{jj}=0,\ j=1,\ldots,n.$

Corollary 1: Based on the obtained contribution matrix C, if $\exists j\in\{1,\ldots,n\}|c_{0j}>d_j$, the problem has no feasible solution.

Corollary 2: Let $c_j^{\min}$ $(j=0,1,\ldots,n)$ be the smallest value of c_{jk} $(k=1,\ldots,n$ and $j\neq k)$ in the row j of C. Sort c_{jk} in non-decreasing order as $c_{[1]}^{\min},\ldots,c_{[n]}^{\min}$. Sort the n jobs in non-decreasing order of their due dates as $d_{[1]}^{\min},d_{[2]}^{\min},\ldots,d_{[n]}^{\min}$; then, if $\sum_{q=1}^{k} c_{[q-1]}^{\min} > d_{[k]}^{\min}$, the problem has no feasible solution.

Lemma 2: If one job in the partial solution violates the due date constraint, then the other jobs that are processed after that job do not need to be enumerated.

Lemma 3: Assume that a partial solution, Π_p, involves n' jobs. If no job in Π_p violates the due date constraints, and the completion time of the partial solution can be computed as $C_{\max}(\Pi_p)$, then the smallest contribution to the makespan of the rest $n-n'$ jobs that are not included in Π_p is $\sum_{i=1}^{n-n'} c_{[i]}^{\min}$. If $C_{\max}(\Pi_p)+\sum_{i=1}^{n-n'} c_{[i]}^{\min}$ equals or exceeds the best found solution, then the remaining $n-n'$ jobs do not need to be enumerated. Notably, $c_{[1]}^{\min},\ldots,c_{[n]}^{\min}$ can be computed

TABLE 3. Computational results of $Model_M$, $Model_{SB}$ and the proposed MILP model.

Problem	Size $n \times m$	Due Date TF	$Model_M$				$Model_{SB}$				$Model_{YLL}$			
			T = 60	$T = 300$	$T = 600$	$T = 7200$	$T = 60$	$T = 300$	T = 600	$T = 7200$	$T = 60$	$T = 300$	$T = 600$	$T = 7200$
Sam01+DD	7 × 7	1	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;2**	**7,705;2**	**7,705;2**	**7,705;2**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		2	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;2**	**7,705;2**	**7,705;2**	**7,705;2**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		3	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;1**	**7,705;1**	**7,705;1**	**7,705;1**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;.1**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam02+DD	8 × 8	1	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;6**	**9,372;6**	**9,372;6**	**9,372;6**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**
		2	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;27**	**9,372;27**	**9,372;27**	**9,372;27**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**
		3	**9,573;0**	**9,573;0**	**9,573;0**	**9,573;0**	**9,573;8**	**9,573;8**	**9,573;8**	**9,573;8**	**9,573;0**	**9,573;0**	**9,573;0**	**9,573;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam03+DD	8 × 9	1	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;10**	**9,690;10**	**9,690;10**	**9,690;10**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		2	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;11**	**9,690;11**	**9,690;11**	**9,690;11**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		3	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;13**	**9,690;13**	**9,690;13**	**9,690;13**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;1**	**NFS;1**	**0,000;0**	**NFS;1**
Sam04+DD	10 × 6	1	**9,159;1**	**9,159;1**	**9,159;1**	**9,159;1**	9,159	9,159	**9,159;339**	**9,159;339**	**9,159;0**	**9,159;0**	**9,159;0**	**9,159;0**
		2	**9,454;1**	**9,454;1**	**9,454;1**	**9,454;1**	9,640	9,454	9,454	**9,454;611**	**9,454;1**	**9,454;0**	**9,454;0**	**9,454;1**
		3	**11,537;0**	**11,537;0**	**11,537;1**	**11,537;0**	NFS	**11,537;14**	**11,537;144**	**11,537;14**	**11,537;19**	**11,537;19**	**11,537;19**	**11,537;19**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam05+DD	11 × 5	1	**8,152;3**	**8,152;3**	**8,152;3**	**8,152;3**	**8,588**	**8,152**	**8,152;1**	**8,152;1**	**8,152;0**	**8,152;0**	**8,152;0**	**8,152;0**
		2	**8,164;2**	**8,164;2**	**8,164;4**	**8,164;2**	**8,332**	**8,168**	**8,168;1**	**8,164;1**	**8,164;0**	**8,164;0**	**8,164;0**	**8,164;0**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	NFS	NFS	NFS	**NFS;404**	NFS	**NFS;187**	**NFS;187**	**NFS;187**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam06+DD	12 × 5	1	**9,084;19**	**9,084;18**	**9,084;41**	**9,084;18**	**9,297**	**9,297**	**9,297**	**9,084**	**9,084;1**	**9,084;1**	**9,084;0**	**9,084;1**
		2	**9,120;6**	**9,120;7**	**9,120;15**	**9,120;7**	**9,181**	**9,120**	**9,177**	**9,120**	**9,120;1**	**9,120;1**	**9,120;0**	**9,120;1**
		3	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**	NFS	NFS	NFS	**NFS;5668**	NFS	NFS	**NFS;575**	**NFS;575**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam07+DD	13 × 4	1	**8,465;15**	**8,465;15**	**8,465;15**	**8,465;15**	8,670	8,465	8,465	8,465	**8,465;1**	**8,465;1**	**8,465;1**	**8,465;1**
		2	**9,002;4**	**9,002;4**	**9,002;4**	**9,002;4**	9,002	9,002	9,002	9,002	**9,002;15**	**9,002;15**	**9,002;15**	**9,002;15**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	**NFS;801**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam08+DD	14 × 4	1	9,693	**9,674;175**	**9,674;175**	**9,674;175**	10,417	10,417	10,417	9,674	**9,674;4**	**9,674;4**	**9,674;4**	**9,674;4**
		2	**NFS;6**	**NFS;6**	**NFS;6**	**NFS;6**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		3	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam09+DD	15 × 6	1	13,472	13,330	13,330	**13,330;605**	14,763	14,260	14,260	14,260	**13,330;0**	**13,330;0**	**13,330;0**	**13,330;0**
		2	13,330	**13,330;218**	**13,330;218**	**13,330;218**	15,205	14,280	14,379	13,599	**13,330;0**	**13,330;0**	**13,330;0**	**13,330;0**
		3	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;2**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam10+DD	16 × 7	1	8,983	8,936	8,936	8,885	9,152	9,009	9,009	8,875	**8,869;0**	**8,869;0**	**8,869;0**	**8,869;0**
		2	9,085	8,990	8,990	8,980	9,160	9,007	9,007	8,941	**8,938;2**	**8,938;2**	**8,938;2**	**8,938;2**
		3	9,104	9,057	9,057	9,057	NFS	10,009	9272	9,057	9,103	**9,057;115**	**9,057;115**	**9,057;115**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
Sam11+DD	17 × 5	1	11,251	11,222	11,222	11,176	12,324	11,425	11,425	11,425	**11,168;5**	**11,168;5**	**11,168;5**	**11,168;5**
		2	11,663	11,622	11,622	11,581	NFS	NFS	NFS	NFS	11,545	11,545	**11,534;545**	**11,534;545**
		3	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;2**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam12+DD	18 × 9	1	9,147	8,942	8,942	8,888	9,748	9,495	9,518	8,933	**8,824;1**	**8,824;1**	**8,824;1**	**8,824;1**
		2	9,307	9,150	9,150	9,056	9,643	9,442	9,456	9,056	**9,042;67**	**9,042;67**	**9,042;67**	**9,042;67**
		3	NFS	9,500	9,500	9,496	NFS	NFS	NFS	NFS	NFS	NFS	NFS	9,500
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam13+DD	19 × 8	1	18,168	17,716	17,716	17,641	19,515	19,515	19,515	19,515	17,552	17,552	17,552	**17,552;822**
		2	18,552	18,527	18,527	18,020	NFS	NFS	NFS	20,482	18,130	18,036	17,998	17,937
		3	NFS	NFS	NFS	**NFS;656**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;0**	**NFS;0.0**	**NFS;0**	**NFS;0**
Sam14+DD	20 × 10	1	31,507	31,507	31,507	29,622	NFS	NFS	NFS	30,706	**29,622;2**	**29,622;2**	**29,622;2**	**29,622;2**
		2	32,746	30,923	31,258	30,523	NFS	NFS	NFS	NFS	30,642	30,642	30,413	**30,413;653**
		3	**NFS;23**	**NFS;23**	**NFS;23**	**NFS;23**	NFS	NFS	NFS	NFS	NFS	NFS	NFS	NFS
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0.0**	**NFS;0**	**NFS;0**
Optimality approved			71.43%	75.00%	75.00%	78.57%	46.43%	48.21%	50.00%	53.57%	71.43%	75.00%	76.79%	83.93%
Average GAP			4.06%	3.56%	0.56%	0.10%	23.99%	17.05%	16.85%	10.36%	3.10%	3.07%	3.04%	0.00%
Average time			18.68	77.21	152.82	1,547.30	37.93	178.54	355.36	3,482.07	19.29	82.66	142.89	1,165.84

TABLE 4. Computational results of the enumeration algorithms.

Problem	Size $n \times m$	Due Date TF	$Algorithm_{SB}$				$Algorithm_{YLL}$			
			$T = 60$	$T = 300$	$T = 600$	$T = 7200$	$T = 60$	$T = 300$	$T = 600$	T=7200
Sam01+DD	7 × 7	1	**7,705;0**	**7,705;0**	**7,705;0**	**7705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		2	**7,705;0**	**7,705;0**	**7,705;0**	**7705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		3	**7,705;0**	**7,705;0**	**7,705;0**	**7705;0**	**7,705;0**	**7,705;0**	**7,705;0**	**7,705;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam02+DD	8 × 8	1	**9,372;0**	**9,372;0**	**9,372;0**	**9372;0**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**
		2	**9,372;0**	**9,372;0**	**9,372;0**	**9372;0**	**9,372;0**	**9,372;0**	**9,372;0**	**9,372;0**
		3	**9,573;0**	**9,573;0**	**9,573;0**	**9573;0**	**9,573;0**	**9,573;0**	**9,573;0**	**9,573;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam03+DD	8 × 9	1	**9,690;0**	**9,690;0**	**9,690;0**	**9690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		2	**9,690;0**	**9,690;0**	**9,690;0**	**9690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		3	**9,690;0**	**9,690;0**	**9,690;0**	**9690;0**	**9,690;0**	**9,690;0**	**9,690;0**	**9,690;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam04+DD	10 × 6	1	**9,159;0**	**9,159;0**	**9,159;0**	**9159;0**	**9,159;0**	**9,159;0**	**9,159;0**	**9,159;0**
		2	**9,454;0**	**9,454;0**	**9,454;0**	**9454;0**	**9,454;0**	**9,454;0**	**9,454;0**	**9,454;0**
		3	**11,537;0**	**11,537;0**	**11,537;0**	**11,537;0**	**11,537;0**	**11,537;0**	**11,537;0**	**11,537;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam05+DD	11 × 5	1	**8,152;2**	**8,152;2**	**8,152;2**	**8152;2**	**8,152;1**	**8,152;1**	**8,152;1**	**8,152;1**
		2	**8,164;1**	**8,164;1**	**8,164;1**	**8164;1**	**8,164;1**	**8,164;1**	**8,164;1**	**8,164;1**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam06+DD	12 × 5	1	**9,084;13**	**9,084;13**	**9,084;13**	**9084;13**	**9,084;16**	**9,084;16**	**9,084;16**	**9,084;16**
		2	**9,120;3**	**9,120;3**	**9,120;3**	**9120;3**	**9120;11**	**9120;11**	**9120;11**	**9120;11**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam07+DD	13 × 4	1	**8,465;20**	**8,465;20**	**8,465;20**	**8,465;20**	**8,465;34**	**8,465;34**	**8,465;34**	**8,465;34**
		2	**9,002;1**	**9,002;1**	**9,002;1**	**9002;1**	**9,002;17**	**9,002;17**	**9,002;17**	**9,002;17**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam08+DD	14 × 4	1	9,674	**9,674;103**	**9,674;103**	**9674;103**	9,804	9,746	9,674	**9,674;739**
		2	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;2**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam09+DD	15 × 6	1	14,976	14,976	14,386	14,058	14,200	13,577	13,577	13,330
		2	13,808	**13,330;199**	**13,330;199**	**113,330;199**	13,621	13,374	13,330	**13,330;3580**
		3	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;1**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam10+DD	16 × 7	1	9,446	9,419	9,419	9,364	8,959	8,959	8,949	8,949
		2	9,445	9,402	9,402	9,402	9,167	9,167	9,167	9,051
		3	9,269	9,142	9,142	**9,057;1269**	9,117	9,116	9,116	9,079
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam11+DD	17 × 5	1	12,556	12,077	11,829	11,829	11,410	11,410	11,410	11,410
		2	12,590	12,135	11,571	**11,534;1720**	11,581	11,581	11,581	11,580
		3	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam12+DD	18 × 9	1	10,980	10,813	10,813	10,615	9,031	9,012	9,012	8,860
		2	10,615	10,363	10,363	10,349	9,344	9,344	9,285	9,285
		3	NFS	NFS	9,808	**9,496;3793**	9,685	9,647	9,647	9,647
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam13+DD	19 × 8	1	20,699	20,699	20,589	20,321	18,132	18,009	18,009	17,983
		2	20,579	20,119	20,119	19,778	18,417	18,356	18,356	18,299
		3	NFS	**NFS;78**	**NFS;78**	**NFS;78**	NFS	**NFS;162**	**NFS;162**	**NFS;162**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Sam14+DD	20 × 10	1	35,847	35,847	35,391	35,214	31,771	31,603	31,523	31,352
		2	34,391	34,452	33,430	32,527	31,006	31,006	31,006	30,784
		3	**NFS;3**	**NFS;3**	**NFS;3**	**NFS;3**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
		4	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**	**NFS;0**
Optimality approved			71.43%	76.79%	76.79%	82.14%	71.43%	73.21%	73.21%	76.79%
Average GAP			7.25%	6.62%	3.18%	2.67%	0.62%	0.34%	0.28%	0.07%
Average time			17.96	72.38	142.02	1,289.63	18.59	82.09	162.45	1,675.09

TABLE 5. Computational results for problems with 20 machines (600 s).

Problem	Size $n \times m$	Due Date TF	$Model_M$		$Model_{SB}$		$Model_{YLL}$		$Algorithm_{SB}$		$Algorithm_{YLL}$	
			OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)
Sam15+DD	5 × 20	1	**26,164**	**0.1**	**26,164**	**0.5**	**26,164**	**0.0**	**26,164**	**0.0**	**26,164**	**0.0**
		2	**26,164**	**0.1**	**26,164**	**0.3**	**26,164**	**0.0**	**26,164**	**0.0**	**26,164**	**0.0**
		3	**26,164**	**0.1**	**26,164**	**0.5**	**26,164**	**0.0**	**26,164**	**0.0**	**26,164**	**0.0**
		4	**27,100**	**0.1**	**27,100**	**0.3**	**27,100**	**0.0**	**27,100**	**0.0**	**27,100**	**0.0**
Sam16+DD	10 × 20	1	**34,198**	**2.6**	34,198	600.0	**34,198**	**0.0**	**34,198**	**1.0**	**34,198**	**0.1**
		2	**34,198**	**2.7**	34,198	600.0	**34,198**	**0.0**	**34,198**	**1.0**	**34,198**	**0.2**
		3	**34,198**	**2.5**	34,198	600.0	**34,198**	**0.0**	**34,198**	**1.0**	**34,198**	**0.1**
		4	**NFS**	**0.2**	**NFS**	**0.2**	**NFS**	**0.0**	**NFS**	**0.0**	**NFS**	**0.0**
Sam17+DD	15 × 20	1	42,855	600.0	47,874	600.0	**42,841**	**1.0**	46,066	600.0	42,841	600.0
		2	42,841	600.0	48,373	600.0	**42,841**	**1.0**	46,066	600.0	43,429	600.0
		3	43,056	600.0	50,069	600.0	**43,056**	**3.0**	45,811	600.0	43,056	600.0
		4	**NFS**	**0.3**	**NFS**	**4.9**	**NFS**	**0.0**	**NFS**	**0.0**	**NFS**	**0.0**
Sam18+DD	20 × 20	1	55,421	600.0	62,542	600.0	**54,198**	**2.0**	64,593	600.0	56,424	600.0
		2	55,563	600.0	62,452	600.0	**54,237**	**4.0**	63,798	600.0	55,799	600.0
		3	NFS	600.0	NFS	600.0	55,225	600.0	NFS	600.0	58,749	600.0
		4	**NFS**	**0.4**	NFS	600.0	NFS	600.0	**NFS**	**0.0**	**NFS**	**0.0**
Optimality proved			62.50%		37.50%		87.50%		62.50%		62.50%	
Average GAP			8.06%		13.19%		0.00%		12.17%		1.13%	
Average time			225.57		375.42		75.69		225.19		225.03	

TABLE 6. Computational results for problems with 30 machines (600 s).

Problem	Size $n \times m$	Due Date TF	$Model_M$		$Model_{SB}$		$Model_{YLL}$		$Algorithm_{SB}$		$Algorithm_{YLL}$	
			OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)
Sam15+DD	5 × 20	1	**34,680**	**0.2**	**34,680**	**0.7**	**34,680**	**0.1**	**34,680**	**0.0**	**34,680**	**0.0**
		2	**34,680**	**0.2**	**34,680**	**0.9**	**34,680**	**0.1**	**34,680**	**0.0**	**34,680**	**0.0**
		3	**34,680**	**0.2**	**34,680**	**0.7**	**34,680**	**0.1**	**34,680**	**0.0**	**34,680**	**0.0**
		4	**34,680**	**0.2**	**34,680**	**0.3**	**34,680**	**0.1**	**34,680**	**0.0**	**34,680**	**0.0**
Sam16+DD	10 × 20	1	**43,179**	**2.8**	43,202	600.0	**43,179**	**0.2**	**43,179**	**0.7**	**43,179**	**0.4**
		2	**43,179**	**3.2**	43,422	600.0	**43,179**	**0.3**	**43,179**	**0.7**	**43,179**	**0.5**
		3	**43,179**	**7.2**	43,179	600.0	**43,179**	**0.2**	**43,179**	**0.7**	**43,179**	**0.4**
		4	**43,642**	**0.6**	44,323	600.0	**43,642**	**0.4**	**43,642**	**0.2**	**43,642**	**0.1**
Sam17+DD	15 × 20	1	50,942	600.0	56,126	600.0	**50,942**	**0.5**	55,817	600.0	50,942	600.0
		2	51,106	600.0	60,449	600.0	**50,942**	**0.2**	55,817	600.0	50,942	600.0
		3	51,617	600.0	63,411	600.0	**51,617**	**0.9**	55,406	600.0	51,617	600.0
		4	**NFS**	**1.0**	NFS	600.0	NFS	600.0	**NFS**	**0.0**	**NFS**	**0.1**
Sam18+DD	20 × 20	1	64,234	600.0	67,142	600.0	**63,471**	**0.5**	76,417	600.0	64,420	600.0
		2	64,626	600.0	67,400	600.0	**63,471**	**0.5**	76,895	600.0	66,190	600.0
		3	66,644	600.0	NFS	600.0	65,026	600.0	73,725	600.0	65,914	600.0
		4	**NFS**	**0.8**	NFS	600.0	NFS	600.0	**NFS**	**0.0**	**NFS**	**0.0**
Optimality proved			62.50%		25.00%		81.25%		62.50%		62.50%	
Average GAP			0.45%		12.64%		0.00%		6.26%		0.55%	
Average time			226.03		450.16		112.76		225.14		225.09	

in advance, so only a lookup table is required to calculate the value of $\sum_{i=1}^{n-n'} c_{[i]}^{\min}$.

IV. COMPUTATIONAL RESULTS

This section describes the implementation of the proposed $Model_{YLL}$ and $Algorithm_{YLL}$ for solving the $F_m|nwt, d_j|C_{\max}$ problem. The test problem set, and the computational results are discussed in the following subsections.

A. TEST PROBLEMS

To verify the performance of the proposed $Model_{YLL}$ and the $Algorithm_{YLL}$, the set of test problems of Samarghandi and Behroozi [11] is used. The test problem set comprises 26 test problems, of which each involves four due date that are set according to the tightness factor $TF = 1, 2, 3,$ and 4, respectively. Samarghandi and Behroozi [11] selected 14 test instances from the test problem set to compare MILP models by removing due date constraint. Therefore, a total

TABLE 7. Computational results for problems with 40 machines (600 s).

Problem	Size $n \times m$	Due Date TF	$Model_M$		$Model_{SB}$		$Model_{YLL}$		$Algorithm_{SB}$		$Algorithm_{YLL}$	
			OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)	OFV	Time (s)
Sam15+DD	5 × 20	1	**43,197**	**0.2**	**43,197**	**0.6**	**43,197**	**0.1**	**43,197**	**0.0**	**43,197**	**0.0**
		2	**43,197**	**0.2**	**43,197**	**0.6**	**43,197**	**0.1**	**43,197**	**0.0**	**43,197**	**0.0**
		3	**43,197**	**0.2**	**43,197**	**0.7**	**43,197**	**0.0**	**43,197**	**0.0**	**43,197**	**0.0**
		4	**43,197**	**0.2**	43,197	600.0	**43,197**	**0.1**	**43,197**	**0.0**	**43,197**	**0.0**
Sam16+DD	10 × 20	1	**53,214**	**2.1**	53,214	600.0	**53,214**	**0.1**	**53,214**	**0.0**	**53,214**	**0.3**
		2	**53,214**	**2.4**	53,214	600.0	**53,214**	**0.1**	**53,214**	**0.0**	**53,214**	**0.3**
		3	**53,214**	**2.2**	53,214	600.0	**53,214**	**0.1**	**53,214**	**0.0**	**53,214**	**0.3**
		4	**56,200**	**0.5**	56,348	600.0	**56,200**	**1.7**	**56,200**	**0.0**	**56,200**	**0.1**
Sam17+DD	15 × 20	1	64,575	600.0	66,681	600.0	**64,091**	**0.3**	67,642	600.0	64,091	600.0
		2	64,091	600.0	66,450	600.0	**64,091**	**0.6**	67,642	600.0	64,091	600.0
		3	64,183	600.0	71,532	600.0	**64,091**	**0.4**	67,642	600.0	64,091	600.0
		4	**NFS**	**14.6**	NFS	600.0	NFS	600.0	**NFS**	**0.3**	NFS	**600.0**
Sam18+DD	20 × 20	1	76,556	600.0	78,726	600.0	**74,690**	**0.4**	84,656	600.0	76,829	600.0
		2	77,736	600.0	81,795	600.0	**74,690**	**0.4**	84,656	600.0	76,112	600.0
		3	76,106	600.0	82,025	600.0	**74,690**	**0.3**	83,473	600.0	76,658	600.0
		4	**NFS**	**0.8**	**NFS**	**0.6**	NFS	600.0	**NFS**	**0.0**	**NFS**	**0.0**
Optimality proved			62.50%		25.00%		87.50%		62.50%		62.50%	
Average GAP			0.72%		10.33%		0.00%		4.24%		0.57%	
Average time			226.46		450.16		75.29		225.02		262.56	

of 104 test instances of the $F_m|nwt, d_j|C_{\max}$ problem and 14 test instances of the $F_m|nwt, d_j|C_{\max}$ problem were also used. Detailed descriptions of the generation of these test problems can be found elsewhere [10].

B. RESULTS AND DISCUSSION

To verify the performance of the proposed $Model_{YLL}$ and $Algorithm_{YLL}$, the computational results obtained using them were compared with those obtained using the best-so-far exact methods, $Model_M$, $Model_{SB}$, and $Algorithm_{SB}$, proposed by Samarghandi and Behroozi [11]. The proposed $Model_{YLL}$ was coded using Microsoft Visual C++ 2012. All numerical experiments were performed on an Ultrabook Windows 10 (64 bits) operating system, with an Intel® Core™ i5-3317U @1.70GHz processor with four cores and 4GB of RAM. Gurobi Solver V7.0 was used to solve $Model_M$, $Model_{SB}$, and the proposed $Model_{YLL}$ on the same machine. To compare the performance of the proposed $Algorithm_{YLL}$ with that of $Algorithm_{SB}$ on the same basis, $Algorithm_{SB}$ was re-executed on the same machine.

To evaluate the computational efficiency of the discussed MILP models, Table 1 compares the numbers of variables and constraints in $Model_M$, $Model_{SB}$, and the proposed MILP model. The proposed MILP model had the fewest constraints.

As in the study of Samarghandi and Behroozi [11], Table 2 presents the computational results of $Model_M$, $Model_{SB}$, and the proposed MILP model when due date constraints were removed. The maximum allowed CPU time, T, was set to 600 s. Columns 1 and 2 list the problem names and sizes, respectively. Columns 3 to 5 list the objective function values (OFV) that are obtained using $Model_M$, $Model_{SB}$, and the proposed $Model_{YLL}$, respectively, when used to solve the $F_m|nwt|C_{\max}$ problem. The OFV and time in boldface correspond to the optimal solution obtained by the model and the time when the optimal solution was found. The ratios of the optimal solution that were obtained using $Model_M$, $Model_{SB}$, and the proposed $Model_{YLL}$ when used to solve the $F_m|nwt|C_{\max}$ problem were 57.14%, 28.57%, and 92.86%, respectively. The proposed $Model_{YLL}$ took less computation time than $Model_M$ or $Model_{SB}$ to solve the $F_m|nwt|C_{\max}$ problem. The last row in Table 2 presents the average GAP of each model, which was computed as $\sum_{j=1}^{n}[(\text{OFV}_j\text{-Best OFV}_j)/\text{OFV}_j \times 100\%]/n$. The average GAPs of $Model_M$, $Model_{SB}$, and the proposed $Model_{YLL}$, used to solve the $F_m|nwt|C_{\max}$ problem, were 0.64%, 2.47%, and 0.00%, respectively. Clearly, the proposed $Model_{YLL}$ without due date constraints outperformed the two compared models when used to solve the $F_m|nwt|C_{\max}$ problem.

Table 3 presents the solutions to the $F_m|nwt, d_j|C_{\max}$ problem that were obtained using $Model_M$, $Model_{SB}$, and the proposed $Model_{YLL}$, in test instances $Sam01 + DD$ through $Sam14 + DD$. The best solutions obtained using these models in these four test instances were obtained when T was 60, 300, 600, and 7200 s, respectively. In this table, numbers in boldface indicates optimal solutions. The NFS in boldface indicates that the model revealed that the problem had no feasible solution; however, a non-bold NFS indicates that the model did not find feasible solutions in the given CPU time, and the problem may or may not have feasible solutions. The ratios of the optimal solution that were achieved using the proposed $Model_{YLL}$ when used to solve the $F_m|nwt, d_j|C_{\max}$ problem were 71.43%, 75.00%,

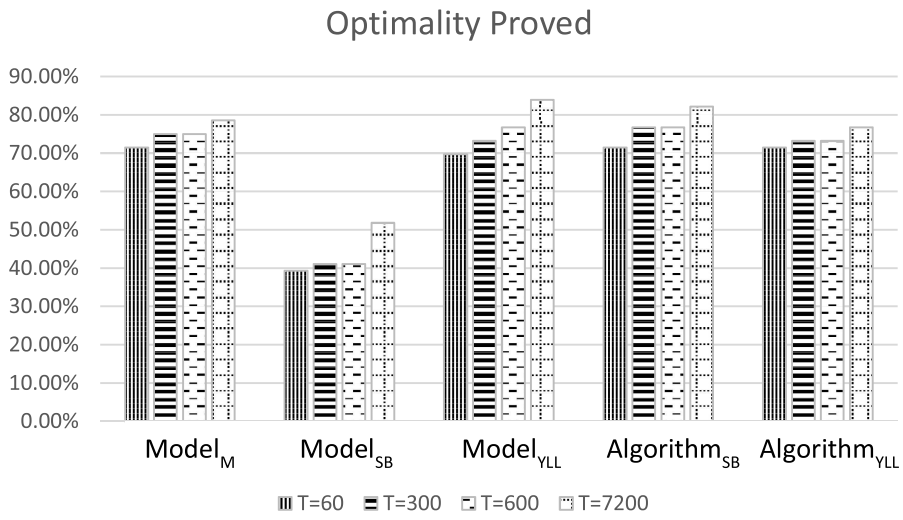

FIGURE 2. Optimal solutions obtained for the compared models and algorithms under different maximal computational times.

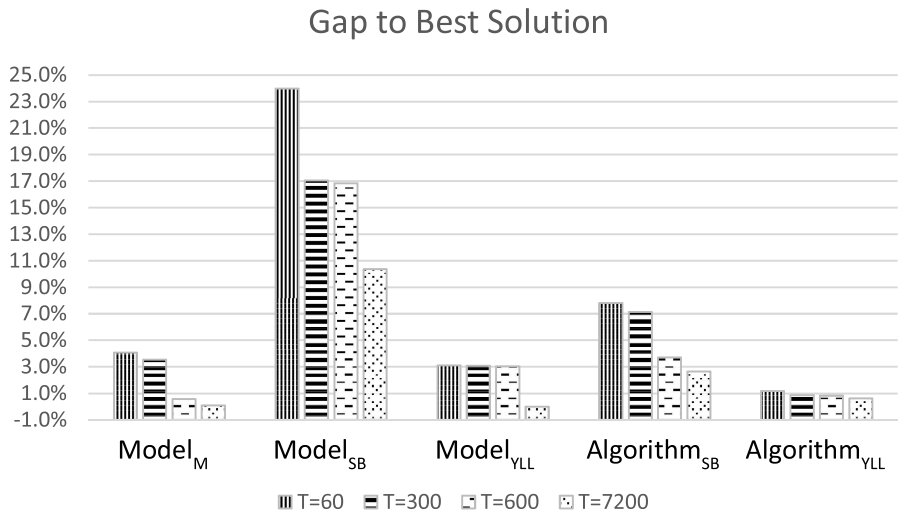

FIGURE 3. Gaps to the best solutions for the compared models and algorithms under different numbers of machines.

76.79%, and 83.93% at T values of 60, 300, 600, and 7200 s, respectively. The corresponding ratios of the optimal solution that were achieved using $Model_M$ were 71.43%, 75.00%, 75.00%, and 78.57%, respectively. The ratios of the optimal solution that were achieved using $Model_{SB}$ were 46.43%, 48.21%, 50.00%, and 53.57%, respectively. These computational results verify that the proposed MILP model found better solutions to the $F_m|nwt, d_j|C_{\max}$ problem than did $Model_M$ or $Model_{SB}$. Furthermore, the average GAPs of the proposed $Model_{YLL}$ are 3.10%, 3.07%, 3.04%, and 0.00% at T values of 60, 300, 600, and 7200 s, respectively. The proposed $Model_{YLL}$ performed better than $Model_M$ and $Model_{SB}$ when used to solve the $F_m|nwt, d_j|C_{\max}$ problem.

Table 4 presents the solutions to the $F_m|nwt, d_j|C_{\max}$ problem that were obtained using $Algorithm_{SB}$ and the proposed $Algorithm_{YLL}$ in the test instances $Sam01 + DD$ through $Sam14 + DD$. The best solutions obtained using the compared enumeration algorithms in the test instances at T = 60, 300, 600, and 7200 s, respectively, were reported. As revealed in Table 4, even though $Algorithm_{SB}$ found more (sub-) optimal solutions in a shorter computing time than the $Algorithm_{YLL}$, the average GAP to the (sub-) optimal solutions of the $Algorithm_{YLL}$ was much less than that

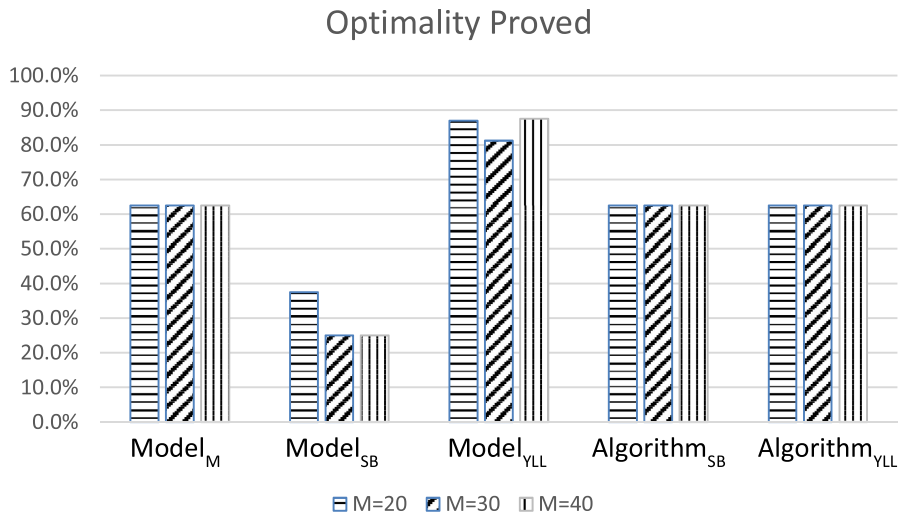

FIGURE 4. Optimal solutions obtained for the compared models and algorithms under different numbers of machines.

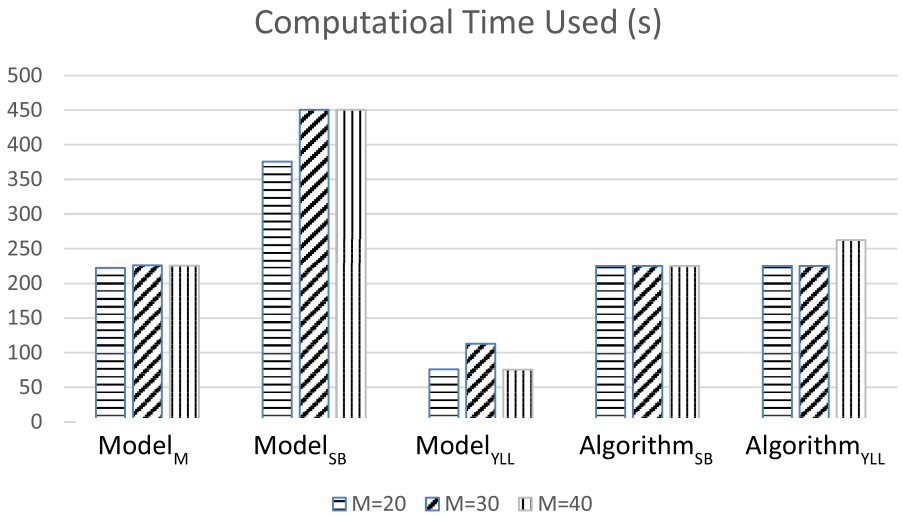

FIGURE 5. Computational times used for the compared models and algorithms under different numbers of machines.

of $Algorithm_{SB}$. The average GAPs were 0.62%, 0.34%, 0.28%, and 0.07% at T values of 60, 300, 600, and 7200 s, respectively. The corresponding average GAPs that were obtained using $Algorithm_{SB}$ were 7.25%, 6.62%, 3.18%, and 2.67%, respectively.

Figure 2 shows the ratios of optimal solutions, as obtained for the compared models and algorithms under different maximal computational times, while Figure 3 shows the gaps to the best solutions for the compared models and algorithms under different numbers of machines. As shown in Figure 2, the ratios of the optimal solutions obtained by the compared models and algorithms are increased when the maximal computational time is increased. The proposed $Model_{YLL}$ performed the best among all the compared models and algorithms. As shown in Figure 3, the average gap of $Algorithm_{YLL}$ is the least among all the compared models and algorithms when the maximal computational times equal 60, 300, and 600 seconds. However, when the maximal computational time equals 3600 seconds, the $Model_{YLL}$ is the best.

The test instances $Sam14 + DD$ through $Sam26 + DD$ with four due date tightness factors were used to demonstrate the effect of increasing the number of machines on the performance of different models/algorithms. The maximum allowed CPU time for solving each problem was 600 s. Tables 5-7 present the computational results obtained using $Model_M$, $Model_{SB}$, the proposed MILP model, $Algorithm_{SB}$, and the proposed $Algorithm_{YLL}$, respectively, when used to solve the problems with 20, 30, and 40 machines. As revealed in Tables 5-7, the ratios of the optimal solution that were achieved using the proposed MILP model were 87.50%, 81.25%, and 87.50% for the problem with

20, 30, and 40 machines, respectively. The corresponding ratios of the optimal solution that were achieved using $Model_M$ were 62.50%, 62.50%, and 62.50%, respectively, and those achieved using $Model_{SB}$ were 37.50%, 25.00%, and 25.00%, respectively. The computational results confirmed again that the proposed $Model_{YLL}$ provided better solutions than those obtained using $Model_M$ and $Model_{SB}$. Furthermore, the computational time that is required by $Model_{YLL}$ is much less than that required by $Model_M$ or $Model_{SB}$.

As indicated in Tables 5-7, the ratios of the optimal solution that were achieved using the proposed $Algorithm_{YLL}$ with 20, 30, and 40 machines were 62.25%, 62.25%, and 62.25%, respectively. Even though the ratios of the optimal solution that were achieved using the two-stage enumeration algorithm were the same as those achieved using $Algorithm_{SB}$, the average GAPs were much less. The computational results verified further that the proposed $Algorithm_{YLL}$ outperformed $Algorithm_{SB}$.

Figure 4 shows the ratios of optimal solutions obtained for the compared models and algorithms under different numbers of machines, while Figure 5 shows the computational time used for the compared models and algorithms under different numbers of machines. As shown in Figure 4, $Model_{YLL}$ performed the best among all the compared models and algorithms for all numbers of machines. As shown in Figure 5 the average computational time of $Model_{YLL}$ is the smallest among all the compared models and algorithms.

V. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

Although due date constraints are very important in industry, they have rarely been studied as hard constraints in the NWFSP. This work proposed an MILP model and a two-phase enumeration algorithm for finding the (sub-) optimal solution to the $F_m|nwt, d_j|C_{\max}$ problem. A performance comparison that involved a lot of benchmark instances revealed that the proposed $Model_{YLL}$ and $Algorithm_{YLL}$ significantly outperformed the best-so-far exact methods. With few effective and efficient exact methods currently available to solve the $F_m|nwt, d_j|C_{\max}$ problem, the main contribution of this work is to reduce the gap between theoretical progress and industrial practice in this area.

Since the $F_m|nwt, d_j|C_{\max}$ problem is a relatively little studied subject, many related topics warrant further study. First, the development of additional efficient and effective constructive heuristic and meta-heuristic algorithms to solve the large $F_m|nwt, d_j|C_{\max}$ problem is a very promising direction for future research efforts. Second, further investigations must be performed to solve the problem with various performance criteria. Third, solving the problem with multi-objectives would be valuable. Finally, extending the $F_m|nwt, d_j|C_{\max}$ problem to incorporate job release date constraints would be complex but have practical value and so warrants further work.

REFERENCES

[1] J. Kim, A. Kröller, J. S. B. Mitchell, and G. R. Sabhnani, "Scheduling aircraft to reduce controller workload," in *Proc. 9th Workshop Algorithmic Approaches Transp. Modeling, Optim., Syst. (ATMOS)*, 2009, pp. 1–12.
[2] C. Mannino and A. Mascis, "Optimal real-time traffic control in metro stations," *Oper. Res.*, vol. 57, no. 4, pp. 1026–1039, 2009.
[3] Y. Wang, J. Tang, Z. Pan, and C. Yan, "Particle swarm optimization-based planning and scheduling for a laminar-flow operating room with downstream resources," *Soft Comput.*, vol. 19, no. 10, pp. 2913–2926, 2015.
[4] N. G. Hall and C. Sriskandarajah, "A survey of machine scheduling problems with blocking and no-wait in process," *Oper. Res.*, vol. 44, no. 3, pp. 510–525, 1996.
[5] Y. Wang, X. Li, R. Ruiz, and S. Sui, "An iterated greedy heuristic for mixed no-wait flowshop problems," *IEEE Trans. Cybern.*, vol. 99, no. 5, pp. 1–14, May 2018.
[6] A. Allahverdi, "A survey of scheduling problems with no-wait in process," *Eur. J. Oper. Res.*, vol. 255, no. 3, pp. 665–686, 2016.
[7] M. S. Nagano and H. H. Miyata, "Review and classification of constructive heuristics mechanisms for no-wait flow shop problem," *Int. J. Adv. Manuf. Technol.*, vol. 86, nos. 5–8, pp. 2161–2174, 2016.
[8] A. Allahverdi and T. Aldowaisan, "No-wait flowshops with bicriteria of makespan and total completion time," *J. Oper. Res. Soc.*, vol. 53, no. 9, pp. 1004–1015, 2002.
[9] A. Allahverdi, H. Aydilek, and A. Aydilek, "No-wait flowshop scheduling problem with two criteria; Total tardiness and makespan," *Eur. J. Oper. Res.*, vol. 269, no. 2, pp. 590–601, 2018, doi: 10.1016/j.ejor.2017.11.070.
[10] H. A. Samarghandi, "A particle swarm optimisation for the no-wait flow shop problem with due date constraints," *Int. J. Prod. Res.*, vol. 53, no. 9, pp. 2853–2870, 2015.
[11] H. Samarghandi and M. Behroozi, "On the exact solution of the no-wait flow shop problem with due date constraints," *Comput. Oper. Res.*, vol. 81, no. 1, pp. 141–159, 2017.
[12] T. A. Aldowaisan and A. Allahverdi, "No-wait flowshop scheduling problem to minimize the number of tardy jobs," *Int. J. Adv. Manuf. Technol.*, vol. 61, nos. 1–4, pp. 311–323, 2012.
[13] A. S. Manne, "On the job-shop scheduling problem," *Oper. Res.*, vol. 8, no. 2, pp. 219–223, 1960.
[14] D. A. Wismer, "Solution of the flowshop-scheduling problem with no intermediate queues," *Oper. Res.*, vol. 20, no. 3, pp. 689–697, 1972.
[15] S. S. Reddi and C. V. Ramamoorthy, "On the flow-shop sequencing problem with no wait in process," *Oper. Res. Quart.*, vol. 23, no. 3, pp. 323–331, 1972.

KUO-CHING YING joined the National Taipei University of Technology in 2009, where he is currently a Distinguished Professor with the Department of Industrial Engineering and Management. He is an author or a co-author of over 100 academic papers, some of which have been published in international journals, such as *Applied Intelligence*, *Applied Soft Computing*, *Computers and Operations Research*, *Computers and Industrial Engineering*, the *European Journal of Industrial Engineering*, the *European Journal of Operational Research*, the *International Journal of Production Economics*, the *International Journal of Advanced Manufacturing Technology*, the *International Journal of Innovational Computing*, *Information and Control*, the *International Journal of Production Research*, the *Journal of the Operational Research Society*, *The International Journal of Management Sciences* (OMEGA), *Production Planning & Control*, and *Transportation Research Part E: Logistics and Transport Review*, among others. His research interests include operations scheduling.

CHUNG-CHENG LU received the Ph.D. degree from the Department of Civil and Environmental Engineering, University of Maryland at College Park, USA, in 2007. He is currently a Professor with the Department of Transportation and Logistics Management, National Chiao Tung University, Hsinchu, Taiwan.

He specializes in intelligent transportation systems and multimodal dynamic transportation and logistics network modeling and optimization. He has published numerous papers, for example, in *Transportation Research Part B*, *Transportation Research Part C*, *Transportation Research Part D*, *Transportation Research Part E*, the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, the *European Journal of Operational Research*, *Computers & Operations Research*, *Computers & Industrial Engineering*, the *Journal of Intelligent Transportation Systems*, and *Networks and Spatial Economics*.

SHIH-WEI LIN received the bachelor's, master's, and Ph.D. degrees in industrial management from the National Taiwan University of Science and Technology, Taiwan, in 1996, 1998, and 2000, respectively. He is currently a Professor with the Department of Information Management, Chang Gung University, Taiwan. He is also with the Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan, Taiwan, and with the Department of Industrial Engineering and Management, Ming Chi University of Technology, Taipei, Taiwan. His current research interests include meta-heuristics and data mining. His papers have appeared in *Computers & Operations Research*, the *European Journal of Operational Research*, the *Journal of the Operational Research Society*, the *European Journal of Industrial Engineering*, the *International Journal of Production Research*, the *International Journal of Advanced Manufacturing Technology*, *Knowledge and Information Systems*, *Applied Soft Computing Applied Intelligence*, and *Expert Systems with Applications*.