The Flow Shop Scheduling Polyhedron with Setup Times

Roger Z. Ríos-Mercado
Graduate Program in Systems Engineering
Universidad Autónoma de Nuevo León
AP 111 – F, Cd. Universitaria
San Nicolás de los Garza, NL 66450, México
E-mail: *roger@uanl.mx*

Jonathan F. Bard
Graduate Program in Operations Research
University of Texas at Austin
Austin, Texas 78712–1063, USA
E-mail: *jbard@mail.utexas.edu*

December 1999
Revised: January 2002, August 2002

Abstract

This paper addresses the problem of improving the polyhedral representation of a certain class of machine scheduling problems. Despite the poor polyhedral representation of many such problems in general, it is shown that notably tighter linear programming representations can be obtained for many important models. In particular, we study the polyhedral structure of two different mixed-integer programming formulations of the flow shop scheduling problem with sequence-dependent setup times, denoted by SDST flow shop. The first is related to the asymmetric traveling salesman problem (ATSP) polytope. The second is less common and is derived from a model proposed by Srikar and Ghosh based on the linear ordering problem (LOP) polytope. The main contribution of this work is the proof that any facet-defining inequality (facet) of either of these polytopes (ATSP and LOP) induces a facet for the corresponding SDST flow shop polyhedron. The immediate benefit of this result is that all developments to date on facets and valid inequalities for both the ATSP and the LOP can be applied directly to the machine scheduling polytope. In addition, valid mixed-integer inequalities based on variable upper-bound flow inequalities for either model are developed as well. The derived cuts are evaluated within a branch-and-cut framework.

Keywords: flow shop scheduling, setup times, polyhedral combinatorics, facet-defining inequalities, asymmetric traveling salesman problem, linear ordering problem

1 Introduction

In this paper, we consider the problem of finding a permutation schedule of n jobs in an m-machine flow shop environment that minimizes the maximum completion time $C_{\max}$ of all jobs, also known as the makespan. The jobs are available at time zero and have sequence-dependent setup times on each machine. All parameters, such as processing and setup times, are assumed to be known with certainty. This problem is regarded in the scheduling literature as the sequence-dependent setup time flow shop (SDST flow shop) and is evidently $\mathcal{NP}$-hard since the case where $m = 1$ is simply a traveling salesman problem (TSP).

Applications of sequence-dependent scheduling are commonly found in most manufacturing environments. In the printing industry, for example, presses must be cleaned and settings changed when ink color, paper size or type differ from one job to the next. Setup times are strongly dependent on the job order. In the container manufacturing industry machines must be adjusted whenever the dimensions of the containers are changed, while in printed circuit board assembly, rearranging and restocking component inventories on the magazine rack is required between batches. In each of these situations, sequence-dependent setup times play a major role and must be considered explicitly. See Allahverdi et al. [1] for an extensive survey of scheduling research involving setup times.

Although many machine scheduling problems can be modeled as mixed-integer linear programs, the resultant linear programming (LP) relaxations are generally weak. This leads to very poor performance of LP-based optimization schemes. To overcome this drawback, stronger polyhedral representations are needed.

While there has been some effort at developing exact optimization schemes for the SDST flow shop based on non-LP-approaches [13, 21], much of the recent work has focused on heuristics [20, 22, 23]. To the best of our knowledge, this paper represents the first attempt to examine the SDST flow shop from a polyhedral theory point of view. One of our primary objectives is to show how the polyhedral representation of the problem can be enhanced by exploiting its similarities with both the asymmetric traveling salesman problem (ATSP) and the linear ordering problem (LOP). We call these formulations model A and model B, respectively. In each case, two sets of variables are identified: a set of binary decision variables which determines the sequence or ordering of the jobs, and a set of nonnegative real variables which determines the times processing begins for each job. When the time variables are ignored the binary variables give rise to a subspace of the SDST flow shop consisting of the convex hull of incidence vectors of feasible sequences. For model A, this subspace is the well-known ATSP polytope; for model B, the corresponding subspace is the well-known LOP polytope.

In the sequel, we develop several properties of the SDST flow shop polytope for models A and B. One of our main contributions is the proof that any facet-defining inequality (or facet) for either of these polytopes induces a facet for the SDST flow shop polyhedron. It is well known that a facet is the strongest of all valid inequalities. The main advantage of this result is that all facets that have been derived for both the ATSP and the LOP over the past few years can be converted into facets for the SDST flow shop polyhedron. As a consequence, branch and cut methodologies designed for the TSP and LOP can be readily extended to solve sequence-dependent flow shop problems.

It is important to emphasize that many of the results presented throughout the paper can, in fact, be extended to other machine scheduling problems with similar characteristics (i.e., sequence-dependent setup

times) which could then be used to improve their polyhedral representations.

The rest of the paper is organized as follows. In Section 2 we introduce models A and B, and discuss their basic differences. A brief literature review is presented in Section 3. Major results relating to the polyhedral structure of models A and B are given in Sections 4 and 5, respectively. The computational evaluation is presented in Section 6. We conclude in Section 7 with a discussion of the results.

2 Mathematical Formulation

In the flow shop environment, a set of n jobs must be scheduled on a set of m machines, where each job has the same routing. Therefore, without loss of generality, we assume that the machines are ordered according to how they are visited by each job. Although for a general flow shop the job sequence may not be the same for every machine, here we assume a *permutation schedule*; i.e., a subset of the feasible schedules that requires the same job sequence on every machine. We assume that each job is available at time zero and has no due date. We also assume that there is a setup time which is sequence dependent so that for every machine i there is a setup time that must precede the start of a given task that depends on both the job to be processed (k) and the job that immediately precedes it (j). The setup time on machine i is denoted by s_{ijk} and is assumed to be *asymmetric*; i.e., $s_{ijk} = s_{ikj}$ is not required. After the last job has been processed on a given machine, the machine is brought back to an acceptable "ending" state. We assume that this last operation takes zero time because we are interested in job completion time rather than machine completion time. Our objective is to minimize the time at which the last job in the sequence finishes processing on the last machine, also known as *makespan*. Following Pinedo's [16] notation, this problem is denoted by $Fm|s_{ijk},\ prmu|C_{\max}$ or SDST flow shop.

In modeling this problem as a mixed integer program (MIP), we consider two different formulations. In the first case, a set of the binary variables is used to define whether or not one job is an immediate predecessor of another; in the second case, the binary variables simply determine whether or not one job precedes another. A set of nonnegative real variables is also included in the formulations. In either case they have the same definition and are used to determine the starting time of each job on each machine.

Triangle inequality: The triangle inequality for the setup times is stated as follows:

$$s_{ijk} + s_{ikl} \geq s_{ijl} \quad \text{for } i \in I,\ j,k,l \in J. \tag{1}$$

Throughout the paper, we will assume that the triangle inequality holds unless otherwise stated. In most operations (e.g., see [24, 25]), the time it takes to set up a machine from job j to job l is less than the time it takes to set up a machine from j to another job k, and then set up the machine from k to l. Nevertheless, if there really exists a machine i and jobs j,k,l such that $s_{ijk} + s_{ikl} < s_{ijl}$, we can always replace s_{ijl} with $s'_{ijl} = s_{ijk} + s_{ikl}$ and force (1) to hold as an equality.

2.1 Notation

In the development of the mathematical model, we make use of the following notation.

Indices and sets

m number of machines

n number of jobs

i machine index; $i \in I = \{1, 2, \ldots, m\}$

j, k, l job indices; $j, k, l \in J = \{1, 2, \ldots, n\}$

J_0 $= J \cup \{0\}$ extended set of jobs, including a dummy job denoted by 0

Input data

p_{ij} processing time of job j on machine i; $i \in I$, $j \in J$

s_{ijk} setup time on machine i when job j is scheduled right before job k; $i \in I$, $j \in J_0$, $k \in J$

Computed parameters

A_i upper bound on the time at which machine i finishes processing its last job; $i \in I$,

$$A_i \quad = \quad A_{i-1} + \sum_{j \in J} p_{ij} + \min \left\{ \sum_{j \in J_0} \max_{k \in J} \{s_{ijk}\}, \ \sum_{k \in J} \max_{j \in J_0} \{s_{ijk}\} \right\}$$

where $A_0 = 0$

B_{ij} lower bound on the starting time of job j on machine i; $i \in I$, $j \in J$

$$B_{ij} \quad = \quad \max \{s_{i0j}, \ B_{i-1,j} + p_{i-1,j}\} \qquad i \in I, \ j \in J$$

where $B_{0j} = 0$ for all $j \in J$

Common variables

y_{ij} nonnegative real variable equal to the starting time of job j on machine i; $i \in I$, $j \in J$

$C_{\max}$ nonnegative real variable equal to the makespan;

$$C_{\max} \quad = \quad \max_{j \in J} \{y_{mj} + p_{mj}\}$$

2.2 Model A

Let $A = \{(j,k) \ : \ j, k \in J_0, \ j \neq k\}$ the set of arcs in a complete directed graph induced by the node set J_0. We define the decision variables as follows:

$$x_{jk} \quad = \quad \begin{cases} 1 & \text{if job } j \text{ is the immediate predecessor of job } k; \ (j,k) \in A \\ 0 & \text{otherwise} \end{cases}$$

In the definition of x_{jk}, notice that $x_{0j} = 1$ ($x_{j0} = 1$) implies that job j is the first (last) job in the sequence for $j \in J$. Also notice that s_{i0k} denotes the initial setup time on machine i when job k has no

predecessor; that is, when job k is scheduled first, for $k \in J$. This variable definition yields what we call a TSP-based formulation.

$$\text{Minimize} \quad C_{\max} \tag{2a}$$

subject to

$$\sum_{j \in J_0} x_{jk} = 1 \qquad k \in J_0 \tag{2b}$$

$$\sum_{k \in J_0} x_{jk} = 1 \qquad j \in J_0 \tag{2c}$$

$$y_{ij} + p_{ij} + s_{ijk} \leq y_{ik} + A_i(1 - x_{jk}) \qquad i \in I,\ j,k \in J \tag{2d}$$

$$y_{mj} + p_{mj} \leq C_{\max} \qquad j \in J \tag{2e}$$

$$y_{ij} + p_{ij} \leq y_{i+1,j} \qquad i \in I \setminus \{m\},\ j \in J \tag{2f}$$

$$x_{jk} \in \{0,1\} \qquad j,k \in J_0,\ j \neq k \tag{2g}$$

$$y_{ij} \geq B_{ij} \qquad i \in I,\ j \in J \tag{2h}$$

Equations (2b) and (2c) state that every job must have a predecessor and successor, respectively. Note that one of these $2n+2$ assignment constraints is redundant in the description of the feasible set. Time-based subtour elimination constraints are given by (2d). This establishes that if job j precedes job k, then the starting time of job k on machine i must not exceed the completion time of job j on machine i ($y_{ij} + p_{ij}$) plus the corresponding setup time. Here, A_i is a large enough number (an upper bound on the completion time on machine i). Constraint (2e) assures that the makespan is greater than or equal to the completion time of all jobs on the last machine, while (2f) states that a job cannot start processing on one machine if it has not finished processing on the previous one. A lower bound on the starting time for each job on each machine is set in (2h).

In formulation (2a)–(2h), we assume that s_{ij0}, the time required to bring machine i to an acceptable end state when job j is processed last, is zero for all $i \in I$. Thus the makespan is governed by the completion times of the jobs only. We are also assuming that all jobs need processing on all machines. If this last condition were not true, then eq. (2e) could be replaced by

$$y_{ij} + p_{ij} \leq C_{\max} \qquad i \in I,\ j \in J$$

at the expense of increasing the number of makespan constraints from n to mn. Note that it is possible to combine $p_{ij} + s_{ijk}$ in (2d) into a single term $t_{ijk} = p_{ij} + s_{ijk}$, but that we still need to handle the processing times p_{ij} separately in constraints (2e) and (2f).

If the triangle inequality does not hold, the lower bound constraint (2h) must be replaced by

$$B_{ij} \leq y_{ij} + C_i(1 - x_{0j}) \qquad i \in I,\ j \in J,$$

where C_i is a large enough number (an upper bound on the initial setup time for machine i).

2.3 Model B

Srikar and Ghosh [24] proposed a second MIP formulation for $F|s_{ijk}, prmu|C_{\max}$. Their formulation contained a slight error that was later corrected by Stafford and Tseng [25]. The Srikar-Ghosh model does not

consider the initial setup time s_{i0k} for the first job in the sequence, that is, it is assumed to be zero. Our formulation includes this parameter.

Let $\hat{A} = \{(j,k) \ : \ j,k \in J, \ j < k\}$. The decision variables are defined as follows:

$$x_{jk} \quad = \quad \begin{cases} 1 & \text{if job } j \text{ is scheduled any time before job } k;\ (j,k) \in \hat{A} \\ 0 & \text{otherwise} \end{cases}$$

The MIP formulation is

$$\begin{aligned}
&\text{Minimize} && C_{\max} && && (3a)\\
&\text{subject to} \\
& && y_{ij} + p_{ij} + s_{ijk} \ \leq \ y_{ik} + A_i(1 - x_{jk}) && i \in I, (j,k) \in \hat{A} && (3b)\\
& && y_{ik} + p_{ik} + s_{ikj} \ \leq \ y_{ij} + A_i(x_{jk}) && i \in I, (j,k) \in \hat{A} && (3c)\\
& && y_{mj} + p_{mj} \ \leq \ C_{\max} && j \in J && (3d)\\
& && y_{ij} + p_{ij} \ \leq \ y_{i+1,j} && i \in I \setminus \{m\}, \ j \in J && (3e)\\
& && x_{jk} \ \in \ \{0,1\} && (j,k) \in \hat{A} && (3f)\\
& && y_{ij} \ \geq \ B_{ij} && i \in I, \ j \in J && (3g)
\end{aligned}$$

Constraints (3b) and (3c) ensure that time precedence is not violated. They also eliminate cycles. Equation (3d) establishes the makespan criterion. Equation (3e) states that a job cannot start processing on one machine if it has not finished processing on the previous machine. A lower bound on the starting time of each job on each machine is set in (3g).

Srikar and Ghosh point out that the triangle inequality must hold in order for constraints (3b)–(3c) to hold. However, Stafford and Tseng provide a stronger condition for constraints (3b)–(3c) to be valid; i.e.,

$$s_{ijk} + s_{ikl} + p_{ik} \quad \geq \quad s_{ijl} \quad \text{for all } i \in I, \ j,k,l \in J. \tag{4}$$

Note that (4) is stronger than the triangle inequality (1), and implies that constraints (3b)–(3c) of the model hold, even if (1) does not hold for setup times. They illustrate this by means of an example.

If the triangle inequality does not hold, constraints (3b), (3c) and (3g) are no longer valid. One possible replacement is

$$\begin{aligned}
y_{ij} + p_{ij} + s_{ijk} \ &\leq \ y_{ik} + (n+1)A_i(1 - x_{jk}) + A_i[P(k) - P(j) - 1] && i \in I, \ (j,k) \in \hat{A} \\
y_{ij} + p_{ij} + s_{ijk} \ &\leq \ y_{ik} + (n+1)A_i x_{jk} + A_i[P(j) - P(k) - 1] && i \in I, \ (j,k) \in \hat{A} \\
B_{ik} \ &\leq \ y_{ik} + C_i[P(k) - 1] && i \in I, \ k \in J,
\end{aligned}$$

respectively, where C_i is a large enough number (upper bound on the starting processing time of all jobs on machine i), and $P(j)$ represents the position in the schedule of job j, given by

$$P(j) \quad = \quad \sum_{p<j} x_{pj} + \sum_{q>j} (1 - x_{jq}) + 1 \quad j \in J. \tag{5}$$

In addition, the following constraints must be added to the formulation:

$$\begin{aligned}
P(j) + 1 \ &\leq \ P(k) + n(1 - x_{jk}) && (j,k) \in \hat{A} \\
P(k) + 1 \ &\leq \ P(j) + n x_{jk} && (j,k) \in \hat{A}
\end{aligned}$$

Thus when the triangle inequality does not hold, the problem size increases considerably.

2.4 Comparison of Models

Table 1: Problem size for models A and B

		Model A		Model B
Variables	Binary	$n(n+1)$	Binary	$\frac{1}{2}n(n-1)$
	Real	$mn+1$	Real	$mn+1$
	Total	$n(n+1)+mn+1$	Total	$\frac{1}{2}n(n-1)+mn+1$
Constraints	(2b)	$n+1$	(3b)	$\frac{1}{2}mn(n-1)$
	(2c)	$n+1$	(3c)	$\frac{1}{2}mn(n-1)$
	(2d)	$mn(n-1)$		
	(2e)	mn	(3d)	mn
	(2f)	$n(m-1)$	(3e)	$n(m-1)$
	Total	$mn^2+mn+n+2$	Total	mn^2+mn-n
Nonzeros	(2b)	$n(n+1)$	(3b)	$\frac{3}{2}mn(n-1)$
	(2c)	$n(n+1)$	(3c)	$\frac{3}{2}mn(n-1)$
	(2d)	$3mn(n-1)$		
	(2e)	$2mn$	(3d)	$2mn$
	(2f)	$2n(m-1)$	(3e)	$2n(m-1)$
	Total	$3mn^2+2n^2+mn$	Total	$3mn^2+mn-2n$

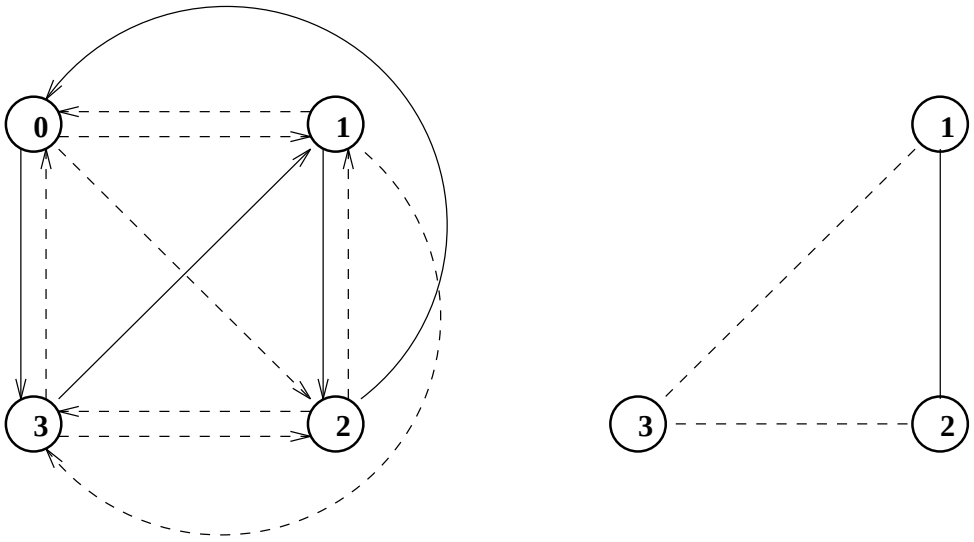
Table 1 shows the problem size in terms of number of variables, constraints, and nonzeros for either model. As can be seen, model B is considerably smaller than model A in terms of both the number of constraints and the number of binary variables. This would appear to make it more attractive when considering exact enumeration methods such as branch-and-bound (B&B) and branch-and-cut (B&C). Nevertheless, it is important to note that it is the quality of the LP relaxations what plays the most prominant role. Table 2 displays the number of binary and real variables, number of constraints, number of nonzeros and density of the matrix of constraints for several values of m and n.

Table 2: Examples of problem size for models A and B

		Variables				
$m \times n$	Model	Binary	Real	Constraints	Nonzeros	Density
2×10	A	110	21	252	840	0.025
	B	45	21	230	620	0.041
2×20	A	420	41	902	3280	0.008
	B	190	41	860	2440	0.012
10×10	A	110	101	1212	3400	0.013
	B	45	101	1190	3180	0.018
10×20	A	420	201	4422	13200	0.005
	B	190	201	4380	12360	0.007

To date, it has not been possible to tackle even moderate size instances of the SDST flow shop with either of these formulations due mainly to the weakness of their LP-relaxation lower bounds. LP-based enumeration

procedures such as B&B and B&C require good LP-relaxation lower bounds. For example, Stafford and Tseng required about 6 hours of CPU time on a 80286-based PC to optimally solve a 5×7 instance using LINDO with formulation B. To improve the polyhedral representation of the relaxed feasible regions it is necessary to generate valid inequalities, the strongest being facets. One way to achieve this is by looking into the related subspaces: the ATSP polytope and the LOP polytope for models A and B, respectively. Many facets have been developed for the ATSP polytope over the last 20 years (e.g., see [3, 9, 4, 5, 18]), and for the LOP polytope (e.g., see [2, 10]). As we show presently, the facets of either of these polytopes can be extended to facets of the SDST flow shop polyhedron.



(a) Model A (b) Model B

Figure 1: Graph representations for schedule (3,1,2)

When comparing the ATSP polytope with the LOP polytope fundamental differences can be observed. In the former, we have a clear picture of what a feasible solution (also called a tour) looks like in a graph. This makes it easier to visualize, for instance, when certain constraints, such as the subtour eliminate constraints, may be violated. However, for model B, it is not a straightforward matter to identify in a graph a feasible solution from a given set of arcs. Figure 1 shows the graph for a 3-job problem and the solution for schedule $S = (3, 1, 2)$ for both models. For model B, an undirected graph can be used because x_{jk} is only defined for $j < k$. The dotted lines represent all feasible arcs (12 for model A and 3 for B); the solid lines identify the solution.

Figure 2 shows how a solution for model B can be built from a solution for model A. Note that each arc $\hat{e} \in \hat{A}$ (Step 2) is visited just once so the procedure is $O(|\hat{A}|) = O(n^2)$. In Step 1, a node (job) within brackets ($[j]$) denotes the job scheduled in the j-th position.

Likewise, a solution for model A can be easily constructed from a feasible solution for model B. Let $\hat{T}$ be an arc set representing a feasible schedule under model B. Let $\hat{x} \in B^{|\hat{A}|}$ be its corresponding characteristic vector; that is, $x_{jk} = 1$ if $(j, k) \in \hat{T}$, and $x_{jk} = 0$ otherwise. For each job j, its position in the schedule $P(j)$

Procedure `A-to-B()`

Input: An arc set T (or directed tour, $|T| = n+1$) corresponding to a feasible schedule for the SDST flowshop under model A.

Output: An arc set $\hat{T}$ corresponding to the equivalent schedule under model B.

0: Initialize `visited(`$\hat{e}$`)=NO` for $\hat{e} \in \hat{A}$ and set $\hat{T} = \emptyset$
1: Sort T as
$T = \{(0,[1]),([1],[2]),\ldots,([n-1],[n]),([n],0)\}$
2: for $j = 1$ to n do
2a: Choose the j-th arc in T
2b: for each unvisited arc $\hat{e} \in \hat{A}$ incident to $[j]$ do
2c: `visited(`$\hat{e}$`)=YES`
2d: for each unvisited arc $\hat{e} \in \hat{A}$ incident from $[j]$ do
2e: `visited(`$\hat{e}$`)=YES`
2f: $\hat{T} \leftarrow \hat{T} + \hat{e}$
3: Output $\hat{T}$

Figure 2: Procedure to go from solution of A to solution of B

is determined by eq. (5) in $n(n-1)$ operations. The schedule S is found by sorting the jobs by increasing value of $P(j)$ and a feasible tour T is easily built from S in $O(n)$ time so that the complete conversion takes $O(n^2)$ time.

3 Related Research

We now highlight some previous work on the SDST flow shop and related problems.

Exact methods: Formulation B was introduced by Srikar and Ghosh [24]. They used this model and the SCICONIC/VM mixed integer programming solver (based on branch-and-bound) to solve several randomly generated instances of the SDST flow shop. The largest solved was a 6-machine, 6-job problem, in about 22 minutes of CPU in a Prime 550 computer.

Later, Stafford and Tseng [25] corrected an error in the S-G formulation and using LINDO solved a 5×7 instance in about 6 hours of CPU on a PC. They also proposed three new MIP formulations of related flow shop problems based on the S-G model.

To the best of our knowledge, there have been no other attempts to solve the problem optimally using either formulation. However, Gupta [13] presented a branch-and-bound algorithm for the case where the objective is to minimize the total machine setup time. No computational results were reported. All other work has centered on variations of the two- or single-machine case, with the latter being analogous to the TSP (e.g., see Section 6 in Queyranne and Schulz [17]).

2-machine case: Work on $F2|s_{ijk},\ prmu|C_{\max}$ includes Corwin and Esogbue [6], who considered a subclass of this problem that arises when one of the machines has no setup times. After establishing the optimality

of permutation schedules, they developed an efficient dynamic programming formulation which they showed is comparable, from a computational standpoint, to the corresponding formulation of the traveling salesman problem whose complexity is $O(n^2 2^n)$ [8]. No computations were performed.

Gupta and Darrow [12] established the $\mathcal{NP}$-hardness of the problem and showed that permutation schedules do not always minimize makespan. They derived sufficient conditions for a permutation schedule to be optimal, and proposed and evaluated empirically four heuristics. They observed that the procedures performed quite well for problems where setup times were an order of magnitude smaller than processing times. However, when the magnitude of the setup times was in the same range as the processing times, the performance of the first two proposed algorithms decreased sharply.

Szwarc and Gupta [26] developed a polynomially bounded approximate method for the special case where the sequence-dependent setup times are additive. Their computational experiments on instances of up to 7 jobs showed optimal results for the 2-machine case.

Heuristics: Most of the recent work for $F|s_{ijk},\ prmu|C_{\max}$ has focused on heuristics. Simons [23] described four heuristics and compared them with three benchmarks that represent generally practiced approaches to scheduling in this environment. Experimental results for problems with up to 15 machines and 15 jobs were presented. His findings indicated that two of the proposed heuristics (`SETUP` and `TOTAL`) produced substantially better results than the other methods tested.

In [20], we developed a new greedy randomized adaptive search procedure (`GRASP`) and compared it to Simons' `SETUP` heuristic on a series of randomly generated problems of size up to 6×100. In the computations, `GRASP` outperformed `SETUP` on those instances where the setup times were an order of magnitude smaller than the processing times. When both parameters were identically distributed, `SETUP` was seen to be more effective. In [22], we proposed a TSP-based heuristic which showed compuationally superior to the existing approach by effectively exploiting the TSP structure of the problem. This is to the best of our knowledge the best heuristic for $F|s_{ijk},\ prmu|C_{\max}$.

4 Polyhedral Results for Model A

Consider the MIP model of the SDST flow shop given by (2a)–(2h). We are interested in the polyhedral description of the convex hull of the set of feasible solutions. Let $G_{n+1} = (V_{n+1}, A_{n+1})$ be a directed graph on $n+1$ nodes, where each node in the set V_{n+1} is associated with a job in J_0. We assume that G_{n+1} is complete. Thus $|A_{n+1}| = n(n+1)$. Let $X_{n+1} = \{x \in B^{n(n+1)} : x \text{ is the incidence vector of a tour in } G_{n+1}\}$.

Let $S_A = \{(x, y) \in B^{n(n+1)} \times R^{mn+1} : (x, y) \text{ is a feasible solution to (2b)–(2h)}\}$, where the y vector includes the mn time variables (2h) plus the makespan variable $C_{\max}$. Then S_A can be represented as follows: $S_A = \{(x, y) : x \in X_{n+1},\ (x, y) \in C_A,\ y \in Y\}$, where X_{n+1} is the projection of S_A onto the subspace of the binary variables, $C_A = \{(x, y) : (x, y) \text{ satisfies (2d)}\}$, and $Y = \{y : y \text{ satisfies (2e), (2f), and (2h)}\}$ is projection of S_A onto the subspace of the real variables. It is well known that the set $\text{conv}(X_{n+1})$ (the ATSP polytope on $n+1$ nodes) is characterized by (i) assignment constraints and (ii) subtour elimination constraints. In the formulation (2b)–(2h), the latter were omitted because they are implied by (2d) which can be viewed as time-based subtour elimination constraints.

We are interested in the polyhedral structure of $P_A = \text{conv}(S_A)$, the convex hull of S_A. We have $n(n+1)$ binary variables (x_{jk}'s), and $mn+1$ nonnegative real variables (y_{ij}'s and $C_{\max}$) giving a total of $N = n(n+m+1)+1$ variables. Note that once a feasible incidence vector $x \in X_{n+1}$ has been determined, that is, once a given sequence is known, the computation of the associated $y \in R^{mn+1}$ that minimizes the makespan can be done recursively in $O(mn)$ operations.

Now we establish the most important results. Proofs can be found in Appendix A.

Theorem 1 *Let $P_A = conv(S_A)$ be the convex hull of S_A. Then* $\dim(P_A) = n(n+m-1)$

Corollary 1 *The equality set of P_A is given by the assignment constraints (2b)–(2c); that is,*

$$(A^=, b^=) = ((A^=_{ATSP}, O), b^=)$$

where $A^=_{ATSP}$ is the equality set of the associated ATSP on $n+1$ vertices.

When a proper face F of P_A is found to have dimension $\dim(F) = n(n+m-1)-1$, Theorem 1 implies that F is a facet of P_A. We now establish the following relationship between facets of $\text{conv}(X_{n+1})$ (the ATSP polytope on $n+1$ nodes) and facets of P_A.

Theorem 2 *Let $F_{ATSP} = \{x \in P \,:\, \pi x = \pi_0\}$ be a facet of conv(X_{n+1}). Then*

$$F_A = \{(x,y) \in P_A \,:\, (\pi, 0)(x,y)^T = \pi_0\}$$

is a facet of P_A.

This result is very important in the sense that any known facet of $\text{conv}(X_{n+1})$ can be easily transformed into a facet of P_A by just adding the corresponding zero vector ($0 \in R^{nm+1}$) to the inequality defining the facet in $R^{n(n+1)}$. The identification of such facets would be at the core of any B&C scheme devised to solve the SDST flow shop problem.

4.1 Mixed-Integer Cuts

We will also make use of the following results on valid inequalities for variable upper-bound flow models to develop mixed-integer cuts.

Let

$$T = \{x \in B^n, z \in R^n_+ \,:\, \sum_{j \in N^+} z_j - \sum_{j \in N^-} z_j \le b,\ z_j \le a_j x_j \quad \text{for } j \in N\} \tag{6}$$

where $N^+ \cup N^- = N$. Here $a_j \in R_+$ for $j \in N$ and $b \in R$. We say that $C \subseteq N^+$ is a *dependent set* if $\sum_{j \in C} a_j > b$.

Proposition 1 (Nemhauser and Wolsey [15]) *If $C \subseteq N^+$ is a dependent set, $\lambda = \sum_{j \in C} a_j - b$, and $L \subseteq N^-$, then*

$$\sum_{j \in C} [z_j + (a_j - \lambda)^+ (1 - x_j)] \quad \le \quad b + \sum_{j \in L} \lambda x_j + \sum_{j \in N^- \setminus L} z_j \tag{7}$$

is a valid inequality for T given by (6).

The set T in (6) is the feasible set of a fixed-charge network flow formulation , and inequalities (7) are known as Generalized Flow Cover Inequalities.

Now, for the purpose of developing our cuts, we rewrite eqs. (2d) and (2h) as follows:

$$y_{ij} - y_{ik} + (A_i + \tau_{ijk})x_{jk} \leq A_i \quad i \in I,\ j,k \in J \tag{8}$$

$$B_{ij} \leq y_{ij} \quad i \in I,\ j \in J \tag{9}$$

where $\tau_{ijk} = p_{ij} + s_{ijk}$ accounts for both the processing and setup times on machine i. Let $z_{ij} = y_{ij} - B_{ij}$, so that $0 \leq z_{ij}$ and define $\xi_{ijk} = (A_i + \tau_{ijk})x_{jk}$. Substituting into (8) gives

$$z_{ij} - z_{ik} + \xi_{ijk} \leq A_i - B_{ij} + B_{ik} \tag{10}$$

Now, we apply Proposition 1 with $N^+ = \{ij, ijk\}$, $N^- = \{ik\}$, $C = \{ij\}$, and $L = \emptyset$. If C is a dependent set; that is, if $\lambda = \tau_{ijk} + B_{ij} - B_{ik} > 0$, then (10) gives rise to the valid inequality

$$\xi_{ijk} + (A_i - B_{ij} + B_{ik})^+(1 - x_{jk}) \leq A_i - B_{ij} + B_{ik} + z_{ik} \tag{11}$$

Assuming $(A_i - B_{ij} + B_{ik})^+ > 0$, (11) becomes

$$\xi_{ijk} - (A_i - B_{ij} + B_{ik})x_{jk} \leq z_{ik} \quad \text{or}$$

$$(p_{ij} + s_{ijk} + B_{ij} - B_{ik})x_{jk} - y_{ik} \leq -B_{ik} \tag{12}$$

which is the desired result. Inequality (12) will have an effect only if $(p_{ij} + s_{ijk} + B_{ij} - B_{ik}) > 0$; that is, if C, as chosen, is a dependent set. Note that when $x_{jk} = 1$, (12) becomes $B_{ij} + p_{ij} + s_{ijk} \leq y_{ik}$ as expected and when $x_{jk} = 0$, it reduces to $B_{ik} \leq y_{ik}$, the default bound.

5 Polyhedral Results for Model B

Now consider the MIP model of the SDST flow shop given by (3a)–(3g). Let $S = \{S_i\}$, for $i = 1, 2, \ldots, n!$, be the set of all feasible schedules. For every schedule $S_i \in S$ there exists an incidence vector $x^i \in B^{n(n-1)/2}$. Let $\hat{X}_n = \{x \in B^{n(n-1)/2} : x \text{ is the incidence vector of a schedule}\}$.

Paralleling the notation in the previous section, let

$$S_B = \{(x, y) \in B^{n(n-1)/2} \times R^{mn+1} : (x, y) \text{ is a feasible solution to (3b)–(3g)}\}$$

Again, the y vector includes the mn time variables (3g) plus the makespan variable $C_{\max}$. The set S_B can be represented as follows: $S_B = \{(x, y) : x \in \hat{X}_n, (x, y) \in C_B, y \in Y\}$, where $\hat{X}_n$ is the projection of S_B onto the subspace of the binary variables, $C_B = \{(x, y) : (x, y) \text{ satisfies (3b)–(3c)}\}$, and $Y = \{y : y \text{ satisfies (3d), (3e), and (3g)}\}$ is the projection of $S : B$ onto the subspace of real variables only. Note that this set Y is the same as defined in the previous section.

We are interested in the polyhedral structure of $P_B = \text{conv}(S_B)$, the convex hull of S_B. Of particular interest is $\text{conv}(\hat{X}_n)$, the convex hull of $\hat{X}_n$, the well-known LOP polytope, and its relationship to P_B. In this section we first provide some of the well-known results for $\text{conv}(\hat{X}_n)$. Subsequently, we derive some results that link $\text{conv}(\hat{X}_n)$ with P_B, which in a sense, parallel those that allowed us to extend the polyhedral structure of $\text{conv}(X_{n+1})$ to P_A in the previous section.

5.1 The conv($\hat{X}_n$) Polyhedron

Lemma 1 *Conv($\hat{X}_n$) is full-dimensional; i.e.,* $\dim(\hat{X}_n) = \frac{n(n-1)}{2}$.

Proposition 2 *The nonnegativity inequalities*

$$x_{jk} \geq 0 \qquad j,k \in J, j < k$$

give facets of conv($\hat{X}_n$) for $n \geq 2$.

Corollary 2 *The inequalities*

$$x_{jk} \leq 1 \quad j,k \in J, j < k$$

give facets of conv($\hat{X}_n$) for all $n \geq 2$.

In contrast with X_{n+1} in model A, it is not possible to identify analogous ATSP valid inequalities such as subtour elimination constraints, comb inequalities, and D_k^+, D_k^- inequalities for model B. One set of valid inequalities that we can identify, though, corresponds to precedence violations for a sequence of jobs. Table 3 shows the valid inequalities that eliminate "cycles" (in the precedence sense) for any 3-job subsequence. We call these inequalities, for a subsequence of size t, the t-subsequence elimination constraint (or t-SEC). For $t = 3$ we show below that the 3-SEC are facets of conv($\hat{X}_n$).

Table 3: 3-SECs for conv($\hat{X}_n$)

Sequence	Constraint		
$j \to k \to l \Rightarrow j \to l$	$x_{jk} + x_{kl}$	$\leq$	$1 + x_{jl}$
$j \to l \to k \Rightarrow j \to k$	$x_{jl} + (1 - x_{kl})$	$\leq$	$1 + x_{jk}$

Lemma 2 *The inequalities (3-subsequence elimination constraints)*

$$x_{jk} - x_{jl} + x_{kl} \quad \geq \quad 0 \quad j,k,l \in J,\ j < k < l \tag{13}$$

give facets of conv($\hat{X}_n$) for all $n \geq 2$.

Lemma 3 *The inequalities*

$$x_{jk} - x_{jl} + x_{kl} \leq 1 \quad j,k,l \in J,\ j < k < l$$

give facets of conv($\hat{X}_n$) for all $n \geq 2$.

All 4-SECs are shown in Table 4 for all $j,k,l,m \in J,\ j < k < l < m$. These valid inequalities, however, do not define facets of conv($\hat{X}_n$). In fact, because $\dim(\hat{X}_n) = n(n-1)2$ and each 4-SEC can be expressed as the intersection of two of the previously developed facets of conv($\hat{X}_n$) (i.e., combinations of $x_{jk} \geq 0, x_{jk} \leq 1$, and 3-SEC), they define faces of dimension $n(n-1)/2 - 2$.

The conv($\hat{X}_n$) polytope can be used to model other scheduling problems, such as single-machine and permutation flow shops problems, where every schedule is feasible. When real variables are introduced in the scheduling model, it remains to be determined whether the valid inequalities discussed above define facets of the complete polyhedron. In the next section we prove that this is the case for the SDST flow shop polyhedron.

Table 4: 4-SECs for $\text{conv}(\hat{X}_n)$

Sequence	Constraint		
$j \to k \to l \to m \Rightarrow j \to m$	$x_{jk} + x_{kl} + x_{lm}$	$\leq$	$2 + x_{jm}$
$j \to k \to m \to l \Rightarrow j \to l$	$x_{jk} + x_{km} + (1 - x_{lm})$	$\leq$	$2 + x_{jl}$
$j \to l \to k \to m \Rightarrow j \to m$	$x_{jl} + (1 - x_{kl}) + x_{km}$	$\leq$	$2 + x_{jm}$
$j \to l \to m \to k \Rightarrow j \to k$	$x_{jl} + x_{lm} + (1 - x_{km})$	$\leq$	$2 + x_{jk}$
$j \to m \to k \to l \Rightarrow j \to l$	$x_{jm} + (1 - x_{km}) + x_{kl}$	$\leq$	$2 + x_{jl}$
$j \to m \to l \to k \Rightarrow j \to k$	$x_{jm} + (1 - x_{lm}) + (1 - x_{kl})$	$\leq$	$2 + x_{jk}$

5.2 The P_B Polyhedron

We now state the theorem defining the dimension of P_B. The proof is very similar to the proof of Theorem 1 because a point $x \in \hat{X}_n$ defines a given feasible sequence for P_B just as $x \in X_{n+1}$ defines a feasible sequence for P_A; moreover, the definition of $y \in R^{mn+1}$ is the same for both polyhedrons.

Theorem 3 *Let P_B = conv(S_B) be the convex hull of S_B. Then P_B is full-dimensional; i.e.,* $\dim(P_B) = n(n-1)/2 + mn + 1$

We now establish the following relationship between facets of $\text{conv}(\hat{X}_n)$ and facets of P_B.

Theorem 4 *Let $F_X = \{x \in \text{conv}(\hat{X}_n) : \pi x = \pi_0\}$ be a facet of conv($\hat{X}_n$). Then $F_B = \{(x, y) \in P_B : (\pi, 0)(x, y)^T = \pi_0\}$ is a facet of P_B.*

5.3 Mixed-Integer Cuts

Note that inequalities (3b) and (3g) in model B have the same structure as inequalities (2d) and (2h) in model A. Thus the valid inequality derived from these equations for model A also applies for model B; that is,

$$(p_{ij} + s_{ijk} + B_{ij} - B_{ik})x_{jk} - y_{ik} \quad \leq \quad -B_{ik} \tag{14}$$

is a valid inequality for model B. Recall that (14) will have an effect only if $(p_{ij} + s_{ijk} + B_{ij} - B_{ik}) > 0$. Note that when $x_{jk} = 1$, (14) becomes $B_{ij} + p_{ij} + s_{ijk} \leq y_{ik}$ as expected and when $x_{jk} = 0$, it reduces to $B_{ik} \leq y_{ik}$, the default bound.

In a similar fashion, we use inequalities (3c) and (3g), a change of variable $x'_{jk} = 1 - x_{jk}$ in (3c), and the same procedure to derive the valid inequality

$$(p_{ik} + s_{ikj} + B_{ik} - B_{ij})(1 - x_{jk}) - y_{ij} \quad \leq \quad -B_{ij}$$

for model B, where again we must have $(p_{ik} + s_{ikj} + B_{ik} - B_{ij}) > 0$ for the inequality to be useful.

6 Computational Evaluation

For the purpose of evaluating the quality of the proposed cuts, a B&C algorithm was developed for each model. Branch and cut is a well-known enumerative technique that exploits polyhedral results by adding violated inequalities to the LP relaxation of an integer program at each node of a B&B algorithm. We used MINTO [14], which is a shell that facilitates the development of implicit enumeration and column generation optimization algorithms that rely on linear relaxations. The user can enrich its basic features by providing a variety of specialized application functions to achieve maximum efficiency for a problem class. CPLEX [7] was used to solve the LP relaxations. Our functions were written in C++ and linked to the MINTO 2.2 and CPLEX 4.0 libraries using the Sun compiler CC, version 2.0.1. CPU times were obtained through MINTO. The code was validated by solving several 100- and 150-job, 1-machine instances to optimality. Recall that the 1-machine problem is an ATSP.

The algorithmic details of our B&C implementation can be found in [19] and are thus omitted here. To conduct our experiments we used randomly generated data. It has been documented [12] that the main feature in real-world data for the SDST flowshop problem is the relationship between processing and setup times. In practice, the setup times are about 20-40% of the processing times. Because the experiments are expensive, we generated a single class of random data sets with the setup times being 20-40% of the processing times: $p_{ij} \in [20, 100]$ and $s_{ijk} \in [20, 40]$.

In the first experiment our aim was to evaluate the effect of the cuts by incorporating them within the B&C framework and comparing them to a pure B&B approach. While it is true that B&C provides a stronger LP-representation, it also true that the size of the linear programs to be solved grows with the number of added cuts. Thus if the generated cuts are not especially effective, the resulting lower bound improvement will be more than offset by the corresponding increase in computational effort. To make this comparison, we generated 5 instances for each machine combination $m \in \{2, 4, 6\}$ and $n \in \{7, 8\}$, with a stopping limit of 90 CPU minutes. In a preliminary experiment we determined the most effective cuts for each model within the B&C framework. The best performance was observed using SECs and UBMICs for model A, and 3-SECs and the UBMICs for model B. The remaining computations were made with these cuts only.

Table 5 displays the results for models A and B for each machine instance. The problem size is given by number of constraints (NC), number of variables (NV), and number of nonzeros (NZ). The number of binary variables is given in parenthesis (B). The average algorithmic performance over the five instances is shown in terms of number of evaluated nodes (nodes), number of cuts added (cuts), maximum number of rows in the LP (LP rows), and CPU time in minutes. All instances were solved to optimality.

As can be seen, even though the size of the LPs increases (LP rows), the generated cuts are found to be effective on reducing the size of the feasible region as the B&C evaluates far fewer nodes and runs significantly faster. For model A the average relative time savings with B&C are 51%, 44%, and 43%, in the 2-, 4- and 6-machine instances, respectively. For model B, we observe little difference for the 2-, and 4-machine instances. The B&C starts to have an effect, however, as the size of the instance gets large. This can be seen in the 6-machine instances where B&C results in a relative time savings of 31%.

For model B, when we increase the number of jobs, B&C has a more pronounced impact. This can be seen in Table 6 where the results for 8-job instances under model B are displayed. The B&C runs on average

Table 5: Performance of B&B and B&C on 7-job instances for models A and B

Instance size						Average performance			
$m \times n$	NC	NV(B)	NZ	Model	Method	Nodes	Cuts	LP rows	Time
2×7	114	71(56)	392	A	B&B	22687	0	114	10.1
				A	B&C	10091	129	236	6.7
	98	36(21)	280	B	B&B	11457	0	98	2.9
				B	B&C	7340	72	168	2.9
4×7	212	85(56)	672	A	B&B	21523	0	212	14.1
				A	B&C	9831	129	328	9.8
	196	50(21)	560	B	B&B	8392	0	196	3.6
				B	B&C	5261	73	266	3.4
6×7	310	99(56)	952	A	B&B	21635	0	310	20.2
				A	B&C	9864	132	435	14.1
	294	64(21)	840	B	B&B	9137	0	294	7.0
				B	B&C	5402	74	366	5.4

Table 6: Comparison of B&B and B&C on 8-job instances for model B

Instance size					Average performance			
$m \times n$	NC	NV(B)	NZ	Method	Nodes	Cuts	LP rows	Time
2×8	128	45(28)	368	B&B	76096	0	128	61.4
				B&C	45072	114	138	46.0
4×8	256	61(28)	736	B&B	68579	0	256	68.6
				B&C	39149	116	366	55.3
6×8	384	77(28)	1104	B&B	59154	0	384	73.3
				B&C	34818	116	493	63.5

33%, 24%, and 15%, faster than the B&B on the 2-, 4-, and 6-machine instances, respectively. Table 7 displays the results when model A was used. As can be seen, the algorithm was unable to solve the problem (after 90 minutes) under either B&B or B&C. However, the optimality gaps (shown in the last column) are smaller under the latter.

In Section 2.4 we pointed out the trade-off between models A and B. On one hand, model A can benefit from a better structured underlying TSP. In contrast, model B is smaller, using only about half the number of binary variables used by model A. In the next experiment, we aim at comparing these models.

By looking at the B&C rows for models A and B in Table 5 we can make a comparison of both models for the 7-job instances. It can be seen that the size of the model (especially in terms of the number of binary variables) plays an important role. Computations are significantly better when model B is used. In fact, the effect is even more dramatic when we attempted to solve 8-job instances. By using model B, we were able to solve 8-job instances (Table 6) in an average of 46, 55.3, and 63.5 minutes of CPU for 2-, 4-, and 6-machines, respectively. When model A was used (Table 7), the algorithm stopped after 90 minutes with average optimality gaps of 38%, 37%, and 36%, respectively.

Table 7: Comparison of B&B and B&C on 8-job instances for model A

Instance size					Average performance				
$m \times n$	NC	NV(B)	NZ	Method	Nodes	Cuts	LP rows	Time	Gap (%)
2×8	146	89(72)	512	B&B	50637	0	146	90.0	50.3
				B&C	49090	276	354	90.0	38.3
4×8	274	105(72)	880	B&B	45329	0	274	90.0	43.6
				B&C	39825	289	487	90.0	37.5
6×8	402	121(72)	1248	B&B	42719	0	402	90.0	39.6
				B&C	32486	287	607	90.0	36.3

Table 8: *Evaluation of B&C on 10-job instances for model B*

Instance size				Average performance			
$m \times n$	NC	NV(B)	NZ	Nodes	Cuts	LP rows	Gap (%)
2×10	200	66(45)	580	26428	241	412	34.8
4×10	400	86(45)	1160	20615	242	612	30.5
6×10	600	106(45)	1740	16453	241	812	26.7

Finally, the last experiment assesses the limited scope of the polyhedral approach. Table 8 shows the average performance of B&C on 10-job instances with a 60-minute time limit for model B. The optimality gaps are between 26% and 35%.

7 Conclusions

This research investigated two different polyhedral representations of the SDST flow shop. For model A, we developed several valid inequalities and established a connection between the SDST flow shop polyhedron and the ATSP polytope. As a consequence, facets for the latter can now be extended to facets for our problem. We also developed similar results for model B, which is not as well structured as model A, but has the advantage of being smaller in terms of number of variables and constraints.

Computational experience with a B&C algorithm indicates that the new inequalities indeed improve the polyhedral representation of the problem considerably. It was also observed how using model B with B&C gives better results on solving instances of this problem. Nevertheless, the fact that even with the development of valid inequalities we are still unable to solve instances with 10 or more jobs shows that LP-based enumeration methods are wanting. The polyhedral representation of the problem is still not strong enough. In fact, we made several attempts to improve the performance of the B&C algorithm, such as changing branching strategies, fixing variables in a preprocessing phase, and reduced cost fixing, but the improvements were not significant. This difficulty is inherent to the SDST flowshop (2 or more machines) since we were able to successfully solve 100- and 150-job instances restricted to the 1-machine case. Recall that minimizing the makespan in SDST flowshop is equivalent to finding the minimum length tour of an $(n+1)$-city ATSP when the number of machines is set equal to 1. It is evident that once we start adding

machines, the ATSP structure starts to weaken. One explanation for this is that, unlike the ATSP where we are looking for a good sequence of nodes, it is difficult here to characterize fully what a good sequence of jobs really is. What might be a good sequence for a certain machine, may be a bad sequence for the others. This makes the problem extremely nasty, and suggests that future work should focus, at least in part, on further improving the polyhedral representations. One way to do this is to develop cuts that would involve both the binary and real decision variables.

It is not clear, though, whether B&C can be sufficiently improved to outperform algorithms that do not rely on LP relaxation. For the problem studied in this paper, we know that better lower bounds can be obtained with non-LP-based approaches [21].

Nevertheless, there are many related problems that can benefit greatly from the results. For instance, when both ready times and due dates are introduced, the polyhedral representation becomes tighter (similar to the case where time window constraints are introduced in the ATSP). In our case, if a different objective function such as minimizing the number of tardy jobs is used, non-LP-based lower bounds might be more difficult to derive, thus giving the edge to B&C. Accordingly, we hope that others will take up the challenge of developing more effective solution algorithms based on the ideas presented here.

Acknowledgments: The research of Roger Ríos-Mercado was partially supported by the Mexican National Council for Science and Technology (CONACyT) and by an E. D. Farmer Fellowship from the University of Texas at Austin. Jonathan Bard was supported by a grant from the Texas Higher Education Coordinating Board under the Advanced Research Program.

We are also grateful to three anonymous referees whose suggestions helped improve the quality and presentation of this paper.

A Proofs of Polyhedral Results for Model A

The following proposition will be used in the proof of Theorem 1, which shows that P_A is full-dimensional.

Proposition 3 *Let θ be a positive real number, $y^0 \in R^t$ be a vector given by $y^0 = \theta\,(1, 2, \ldots, t-1, t)^T$, and $y^u \in R^t$ be given by $y^u = y^0 + e^u$, where e^u is the u-th unit vector in R^t. Then, the vectors in the set $\{y^0, y^1, y^2, \ldots, y^t\}$ are affinely independent.*

Proof: For $\alpha_0, \alpha_1, \ldots, \alpha_p \in R$, we prove that the following system of linear equations

$$\sum_{u=0}^{t} \alpha_u y^u = 0 \tag{15}$$

$$\sum_{u=0}^{t} \alpha_u = 0 \tag{16}$$

implies $\alpha_u = 0$ for all $u = 0, \ldots, t$.

From (15) we have

$$\sum_{u=0}^{t} \alpha_u y^u = 0 \quad \Rightarrow \quad \alpha_0 y^0 + \sum_{u=1}^{t} \alpha_u (y^0 + e^u) = 0$$

$$\Rightarrow \quad \sum_{u=0}^{t} \alpha_u y^0 + \sum_{u=1}^{t} \alpha_u e^u = 0$$

$$\Rightarrow \quad y^0 \sum_{u=0}^{t} \alpha_u + \sum_{u=1}^{t} \alpha_u e^u = 0$$

From (16), we now have

$$\sum_{u=1}^{t} \alpha_u e^u = 0$$

so $\alpha_1 = \alpha_2 = \ldots = \alpha_n = 0$ and hence $\alpha_0 = 0$, which completes the proof. ■

For the proof of Theorem 1, which establishes the dimension of P_A, we make use of the following well-known result.

Lemma 4 (Nemhauser and Wolsey [15]) *Let P be a polyhedron and let $(A^=, b^=)$ be its equality set. If $P \subseteq R^n$, then*

$$\dim(P) + \text{rank}(A^=, b^=) = n$$

.

Proof of Theorem 1: The proof consists of two parts (a) and (b).

(a) It is known that of the $2(n+1)$ assignment constraints (2b)–(2c), exactly $2n+1$ are linearly independent. This implies that $\text{rank}(A^=, b^=) \geq 2n+1$, where $(A^=, b^=)$ is the equality set of P_A. It follows from Lemma 4 that

$$\begin{aligned} \dim(P_A) &\leq N - (2n+1) \\ &= n(n+m+1) + 1 - (2n+1) \\ &= n(n+m-1) \end{aligned}$$

(b) To prove $\dim(P_A) \geq n(n+m-1)$ we will show that there exists a set of $n(n+m-1)+1$ affinely independent vectors in R^N. In this regard, consider the subspace X_{n+1} of P_A. The dimension of the ATSP polyhedron on $n+1$ vertices is $n^2 - n - 1$ (e.g., see [11]). This implies that there exists a set of $K = n^2 - n$ affinely independent vectors $x^1, \ldots, x^K$ in $R^{n(n+1)}$, each being the incident vector of a tour. Also note that for any given $x^t \in X_{n+1}$, there exists a corresponding infinite number of feasible assignments of the time variables for P_A. For each x^t, $t = 2, \ldots, K$, let $y^t \in R^{mn+1}$ be any corresponding feasible assignment of the time variables on P_A. Here, y^t includes the mn time variables y_{ij}, and the makespan variable $C_{\max}$. Hence, the set S_1 given by

$$S_1 = \left\{ \begin{pmatrix} x^2 \\ y^2 \end{pmatrix}, \ldots, \begin{pmatrix} x^K \\ y^K \end{pmatrix} \right\}$$

is a set of feasible (and affinely independent) vectors in R^N, with $|S_1| = K - 1 = n^2 - n - 1$.

For x^1 we construct the corresponding y^1 as follows. Assume for simplicity that x^1 defines the job schedule $(1, 2, \ldots, n)$; that is, $x_{j,j+1} = 1$ for all $j = 0, 1, \ldots, n$ (indices 0 and $n+1$ are the same), and $x_{jk} = 0$ otherwise. Assume also that the $mn+1$ components of y^1 are given in the order

$$y^1 = \begin{pmatrix} y^1_{11} \\ \vdots \\ y^1_{m1} \\ y^1_{12} \\ \vdots \\ y^1_{m2} \\ \vdots \\ y^1_{1n} \\ y^1_{mn} \\ C_{\max} \end{pmatrix}$$

That is, all the time variables associated with job 1 come first, then those for job 2, and so on, up to job n (the last in the sequence). The makespan variable comes at the end. Now, it is possible to select a large enough number θ such that the following yields a feasible solution for P_A:

$$\begin{pmatrix} y^1_{11} \\ y^1_{21} \\ \vdots \\ y^1_{m1} \\ \vdots \\ y^1_{1n} \\ \vdots \\ y^1_{mn} \\ C_{\max} \end{pmatrix} = \begin{pmatrix} \theta \\ 2\theta \\ \vdots \\ m\theta \\ \vdots \\ ((n-1)m+1)\theta \\ \vdots \\ mn\theta \\ (mn+1)\theta \end{pmatrix}$$

Let e^u be the u-th unit vector in R^{mn+1}, for all $u = 1, \ldots, mn+1$, and denote the vector $y^1 + e^u$ by $y^{1,u}$. By choosing θ as

$$\theta = \max_{ijk} \{p_{ij} + s_{ijk}\} + 1$$

we ensure not only the feasibility of y^1 but the feasibility of $y^{1,u}$ for all $u = 1, \ldots, mn+1$, as well. Using Proposition 3 with $t = mn+1$ and y^1 as the base vector, we conclude that the $mn+2$ vectors in $\{y^1, y^{1,1}, y^{1,2}, \ldots, y^{1,mn+1}\}$ are affinely independent in R^{mn+1}, which in turn implies affine independence in R^N for the points in the set

$$S_2 = \left\{ \begin{pmatrix} x^1 \\ y^1 \end{pmatrix}, \begin{pmatrix} x^1 \\ y^{1,1} \end{pmatrix}, \begin{pmatrix} x^1 \\ y^{1,2} \end{pmatrix}, \ldots, \begin{pmatrix} x^1 \\ y^{1,mn+1} \end{pmatrix} \right\}$$

with $|S_2| = mn+2$.

It is left to show that the vectors in $S_1 \cup S_2$ are affinely independent. Let α_t, β_u be real numbers for $t \in J_1 = \{1, \ldots, K\}$, and $u \in J_2 = \{1, \ldots, mn+1\}$ such that

$$\sum_{t\in J_1} \alpha_t \begin{pmatrix} x^t \\ y^t \end{pmatrix} + \sum_{u\in J_2} \beta_u \begin{pmatrix} x^1 \\ y^{1,u} \end{pmatrix} = 0 \quad (17)$$

$$\sum_{t\in J_1} \alpha_t + \sum_{u\in J_2} \beta_u = 0 \quad (18)$$

This is a linear system of equations for (α_t, β_u). We now prove that this system has a unique zero solution. We distinguish three cases:

Case 1: $\alpha_t = 0$ for all $t \in J_1$

System (17)–(18) reduces to

$$\sum_{u\in J_2} \beta_u \begin{pmatrix} x^1 \\ y^{1,u} \end{pmatrix} = 0$$

$$\sum_{u\in J_2} \beta_u = 0$$

Due to the affine independence of S_2, it follows that $\beta_u = 0$ for $u \in J_2$. Hence, an all-zero solution for (α_t, β_u) is obtained.

Case 2: $\beta_u = 0$ for all $u \in J_2$

The linear system (17)–(18) becomes

$$\sum_{t\in J_1} \alpha_t \begin{pmatrix} x^t \\ y^t \end{pmatrix} = 0$$

$$\sum_{t\in J_1} \alpha_t = 0$$

which leads to $\alpha_t = 0$ for $t \in J_1$ due to the affine independence of the vectors in S_1.

Case 3: There exists $I_1, I_2 \neq \emptyset$ such that $\alpha_t \neq 0$ for all $t \in I_1 \subseteq J_1$ and $\beta_u \neq 0$ for all $u \in I_2 \subseteq J_2$. Here we have $\alpha_t = 0$ for all $t \in J_1 \setminus I_1$ and $\beta_u = 0$ for all $u \in J_2 \setminus I_2$. We show that Case 3 cannot occur. The corresponding linear system is

$$\sum_{t\in I_1} \alpha_t \begin{pmatrix} x^t \\ y^t \end{pmatrix} + \sum_{u\in I_2} \beta_u \begin{pmatrix} x^1 \\ y^{1,u} \end{pmatrix} = 0$$

$$\sum_{t\in I_1} \alpha_t + \sum_{u\in I_2} \beta_u = 0$$

which can be rewritten as

$$\sum_{t\in I_1} \alpha_t x^t + x^1 \sum_{u\in I_2} \beta_u = 0 \quad (19)$$

$$\sum_{t\in I_1} \alpha_t y^t + \sum_{u\in I_2} \beta_u y^{1,u} = 0 \quad (20)$$

$$\sum_{t\in I_1}\alpha_t + \sum_{u\in I_2}\beta_u \quad = \quad 0 \tag{21}$$

We first note that $\beta' \equiv \sum_{u\in I_2}\beta_u \neq 0$. Otherwise (19) and (21) would become

$$\begin{aligned}\sum_{t\in I_1}\alpha_t x^t &= 0\\ \sum_{t\in I_1}\alpha_t &= 0\end{aligned}$$

which implies, due to the affine independence of $\{x^t\}$, that $|I_1| = 0$. This is clearly a contradiction.

Now consider the following two subcases:

Case 3a: $1 \notin I_1$

Equations (19) and (21) become

$$\begin{aligned}\beta' x^1 + \sum_{t\in I_1}\alpha_t x^t &= 0\\ \beta' + \sum_{t\in I_1}\alpha_t &= 0\end{aligned}$$

However, this contradicts the affine independence of $\{x^t\}$.

Case 3b: $1 \in I_1$

System (19)–(21) is rewritten as

$$(\alpha_1+\beta')x^1 + \sum_{t\in I_1\setminus\{1\}}\alpha_t x^t \quad = \quad 0 \tag{22}$$

$$\sum_{t\in I_1}\alpha_t y^t + \sum_{u\in I_2}\beta_u y^{1,u} \quad = \quad 0 \tag{23}$$

$$(\alpha_1+\beta') + \sum_{t\in I_1\setminus\{1\}}\alpha_t \quad = \quad 0 \tag{24}$$

Equations (22) and (24), and the affine independence of $\{x^t\}$ imply that $I_1\setminus\{1\} = \emptyset$; that is, $I_1 = \{1\}$ consists only of one index. Thus eqs. (23) and (24) become

$$\begin{aligned}\alpha_1 y^1 + \sum_{u\in I_2}\beta_u y^{1,u} &= 0\\ \alpha_1 + \sum_{u\in I_2}\beta_u &= 0\end{aligned}$$

which contradicts the affine independence of $\{y^1, y^{1,u}\}$ (by Proposition 3).

This proves that Case 3 cannot occur.

The results from Cases 1 and 2 prove that $S_1 \cup S_2$ is an affine independent set in R^N, the size of set being $n(n+m-1)+1$. We conclude that $\dim(P_A) \geq n(n+m-1)$.

Thus $\dim(P_A) = n(n+m-1)$. ■

Proof of Corollary 1: Lemma 4 and Theorem 1 imply that $\text{rank}(A^=, b^=) = 2n+1$, which is the rank of the equality set defined by the assignment constraints. ■

Proof of Theorem 2: Let F_{ATSP} be a facet of $\text{conv}(X_{n+1})$. Then $\dim(F_{\text{ATSP}}) = \dim(\text{conv}(X_{n+1})) - 1$, or, expressed in terms of the rank of its equality set,

$$\begin{aligned} \text{rank}\left(\begin{pmatrix} A^=_{\text{ATSP}} \\ \pi \end{pmatrix}, \begin{pmatrix} b^= \\ \pi_0 \end{pmatrix}\right) &= \text{rank}(A^=_{\text{ATSP}}, b^=) + 1 \\ &= 2n+2 \end{aligned}$$

That is, (π, π_0) is linearly independent of the rows of $(A^=_{\text{ATSP}}, b^=)$. Note that $((\pi, 0), \pi_0)$ is a valid inequality for P_A and a nonempty face of P_A. Let $(A^=, b^=)$ be the equality set of P_A. Then $\text{rank}(A^=, b^=) = 2n+1$. The equality set of F_A is then given by

$$(A^=_F, b^=_F) = \left(\begin{pmatrix} A^= \\ \pi' \end{pmatrix}, \begin{pmatrix} b^= \\ \pi_0 \end{pmatrix}\right)$$

where $\pi' = (\pi, 0)$. The rank of this equality set either stays the same at $(2n+1)$ or increases by one to $(2n+2)$. Assume the former; i.e., that $\text{rank}(A^=_F, b^=_F) = 2n+1$. This would imply that

$$\text{rank}\left(\begin{pmatrix} A^=_{\text{ATSP}} & 0 \\ \pi & 0 \end{pmatrix}, \begin{pmatrix} b^= \\ \pi_0 \end{pmatrix}\right) = 2n+1$$

yielding

$$\text{rank}\left(\begin{pmatrix} A^=_{\text{ATSP}} \\ \pi \end{pmatrix}, \begin{pmatrix} b^= \\ \pi_0 \end{pmatrix}\right) = 2n+1;$$

which is a contradiction. Therefore, $\text{rank}(A^=_F, b^=_F) = 2n+2$, which gives $\dim(F_A) = \dim(P_A) - 1$; i.e., F_A is a facet of P_A. ■

B Proofs of Polyhedral Results for Model B

Proof of Theorem 3: Let $N = n(n-1)/2 + mn + 1$. We will show that there exists a set of $N+1$ affinely independent vectors in R^N. Consider the subspace $\hat{X}_n$ of P_B. We proved in Lemma 1 that $\text{conv}(\hat{X}_n)$ is full-dimensional. This implies that there exists a set of $K = n(n-1)/2 + 1$ affinely independent vectors $x^1, \ldots, x^K$ in $R^{n(n+1)}$, each being the incidence vector of a schedule. Also note that for any given $x^t \in \hat{X}_n$, there exists a corresponding infinite number of feasible assignments of the time variables for P_B.

From this point on the rest of the proof follows that of Theorem 1, part (b). We will just sketch the arguments. From the set $\{x^1, \ldots, x^K\}$ we build two disjoint sets $S_1, S_2 \subseteq R^N$ given by

$$S_1 = \left\{\begin{pmatrix} x^2 \\ y^2 \end{pmatrix}, \ldots, \begin{pmatrix} x^K \\ y^K \end{pmatrix}\right\}$$

$$S_2 \quad = \quad \left\{ \begin{pmatrix} x^1 \\ y^1 \end{pmatrix}, \begin{pmatrix} x^1 \\ y^{1,1} \end{pmatrix}, \begin{pmatrix} x^1 \\ y^{1,2} \end{pmatrix}, \ldots, \begin{pmatrix} x^1 \\ y^{1,mn+1} \end{pmatrix} \right\}$$

where S_1 and S_2 are sets of feasible (and affinely independent) vectors in R^N, with $|S_1| = K-1 = n(n-1)/2$ and $|S_2| = mn+2$, so that $|S_1 \cup S_2| = n(n-1)/2 + mn + 2$. We then can prove that the points in $S_1 \cup S_2$ are affinely independent by showing that the linear system

$$\sum_{t \in J_1} \alpha_t \begin{pmatrix} x^t \\ y^t \end{pmatrix} + \sum_{u \in J_2} \beta_u \begin{pmatrix} x^1 \\ y^{1,u} \end{pmatrix} \quad = \quad 0$$

$$\sum_{t \in J_1} \alpha_t + \sum_{u \in J_2} \beta_u \quad = \quad 0$$

admits the unique solution $\alpha_t = \beta_u = 0$ for $t \in J_1 = \{1, \ldots, K\}$, and $u \in J_2 = \{1, \ldots, mn+1\}$. This leads to conclude that $\dim(P_B) = n(n-1)/2 + mn + 1$. ■

For the proof of Theorem 4, we make use of the following well-known result.

Theorem 5 (Nemhauser and Wolsey [15]) *Let $(A^=, b^=)$ be the equality set of $P \subseteq R^n$ and let $F = \{x \in P \ : \ \pi x = \pi_0\}$ be a proper face of P, where $\pi \in R^n, \pi_0 \in R$. Then the following two statements are equivalent:*

(i) F is a facet of P.

(ii) If $\lambda x = \lambda_0$ for all $x \in F$ then

$$(\lambda, \lambda_0) = (\alpha\pi + uA^=, \alpha\pi_0 + ub^=)$$

for some $\alpha \in R$ and some $u \in R^{|M^=|}$.

Proof of Theorem 4: Let F_X be a facet of $\text{conv}(\hat{X}_n)$. Let (π', π_0) represent the inequality $\pi' z \leq \pi_0$ where $\pi' = (\pi, 0) \in R^N$ and $z = (x, y) \in P_B$. Hence F_B can be rewritten as $F_B = \{z \in P_B \ : \ \pi' z = \pi_0\}$. Given that F_X is a facet of $\text{conv}(\hat{X}_n)$, it follows that F_B is a proper face of P_B.

We prove the result by showing that conditions of Theorem 5 hold. Here, the equality set $(A^=, b^=)$ does not exist since P_B is full-dimensional, and we are concerned with solutions to the linear system

$$\lambda z \quad = \quad \lambda_0 \tag{25}$$

where z is any point in P_B satisfying $\pi' z = \pi_0$. Hence, it suffices to demonstrate that all solutions (λ, λ_0) to (25) are of the form $\lambda = \alpha\pi, \lambda_0 = \alpha\pi_0$ for some $\alpha \in R$.

Since $z = (x, y) \in P_B$, the system in (25) can be rewritten as

$$\lambda_x x + \lambda_y y \quad = \quad \lambda_0. \tag{26}$$

Let $x^1 \in F_X$. According to the procedure described in the proof of Theorem 3, it is possible to construct $mn+2$ feasible affinely independent points $y^0, y^1, \ldots, y^{mn+1}$, where $y^u = y^0 + e^u$ for all $u = 1, \ldots, mn+1$. Here e^u denotes the u-th unit vector in R^{mn+1}. It easy to see that $z^i = (x^1, y^i) \in P_B$ for all $i = 0, \ldots, mn+1$.

Moreover, z^i satisfies $\pi' z^i = \pi x^1 = \pi_0$ for all i so that $z^i \in F_B$. Substituting these $mn+2$ points in system (25) we have

$$\begin{aligned} \lambda_x x^1 + \lambda_y y^0 &= \lambda_0 && (27)\\ \lambda_x x^1 + \lambda_y y^1 &= \lambda_0 && (28)\\ &\vdots \\ \lambda_x x^1 + \lambda_y y^{mn+1} &= \lambda_0 && (29) \end{aligned}$$

By subtracting (27) from all other eqs. (28)–(29), we obtain the following system of order $mn+1$:

$$\begin{aligned} \lambda_y (y^1 - y^0) &= 0 \\ &\vdots \\ \lambda_y (y^{mn+1} - y^0) &= 0 \end{aligned}$$

Since $y^i - y^0 = e^i$ it follows that $\lambda_y = 0 \in R^{mn+1}$. This reduces (25) to

$$\lambda_x x = \lambda_0$$

where x satisfies $\pi x = \pi_0$. Given that F_X is a facet, it follows that there is $\alpha \in R$ such that $\lambda_x = \alpha\pi, \lambda_0 = \alpha\pi_0$. This implies that $\lambda = (\lambda_x, \lambda_y) = (\alpha\pi, \alpha 0) = \alpha(\pi, 0) = \alpha\pi'$ and the proof is complete. ■

References

[1] A. Allahverdi, J. N. D. Gupta, and T. Aldowaisan. A review of scheduling research involving setup considerations. *Omega*, 27(2):219–239, 1999.

[2] E. Balas. On the facial structure of scheduling polyhedra. *Mathematical Programming Study*, 24:179–218, 1985.

[3] E. Balas. The asymmetric assignment problem and some new facets of the traveling salesman polytope on a directed graph. *SIAM Journal on Discrete Mathematics*, 2(4):425–451, 1989.

[4] E. Balas and M. Fischetti. The fixed-outdegree 1-arborescence polytope. *Mathematics of Operations Research*, 17(4):1001–1018, 1992.

[5] E. Balas and M. Fischetti. A lifting procedure for the asymmetric traveling salesman polytope and a large new class of facets. *Mathematical Programming*, 58(3):325–352, 1993.

[6] B. D. Corwin and A. O. Esogbue. Two machine flow shop scheduling problems with sequence dependent setup times: A dynamic programming approach. *Naval Research Logistics Quarterly*, 21(3):515–524, 1974.

[7] CPLEX Optimization, Inc., Incline Village, NV. *Using the CPLEX Callable Library, Version 4.0*, 1995.

[8] S. E. Dreyfus and A. M. Law. *The Art and Theory of Dynamic Programming*. Academic Press, Orlando, 1977.

[9] M. Fischetti. Facets of the asymmetric traveling salesman polytope. *Mathematics of Operations Research*, 16(1):42–56, 1991.

[10] P. C. Fishburn. Induced binary probabilities and the linear ordering polytope: A status report. *mathematical Social Sciences*, 23(1):67–80, 1992.

[11] N. Grötschell and M. W. Padberg. Polyhedral theory. In E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan, and D. B. Shmoys, editors, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, pages 251–305. John Wiley & Sons, Chichester, 1985.

[12] J. N. D. Gupta and W. P. Darrow. The two-machine sequence dependent flowshop scheduling problem. *European Journal of Operational Research*, 24(3):439–446, 1986.

[13] S. K. Gupta. n jobs and m machines job-shop problems with sequence-dependent set-up times. *International Journal of Production Research*, 20(5):643–656, 1982.

[14] G. L. Nemhauser, M. W. P. Savelsbergh, and G. C. Sigismondi. MINTO, a Mixed INTeger Optimizer. *Operations Research Letters*, 15:48–59, 1994.

[15] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, New York, 1988.

[16] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1995.

[17] M. Queyranne and A. S. Schulz. Polyhedral approaches to machine scheduling. Preprint 408/1994, Department of Mathematics, Technical University of Berlin, Berlin, Germany, 1994. Available at `ftp://ftp.math.tu-berlin.de/pub/Preprints/combi/Report-408-1994.ps.Z`.

[18] M. Queyranne and Y. Wang. Symmetric inequalities and their composition for asymmetric travelling salesman polytopes. *Mathematics of Operations Research*, 20(4):838–863, 1995.

[19] R. Z. Ríos-Mercado. *Optimization of the Flowshop Scheduling Problem with Setup Times*. PhD thesis, University of Texas, Austin, August 1997.

[20] R. Z. Ríos-Mercado and J. F. Bard. Heuristics for the flow line problem with setup costs. *European Journal of Operational Research*, 110(1):76–98, 1998.

[21] R. Z. Ríos-Mercado and J. F. Bard. A branch-and-bound algorithm for permutation flow shops with sequence-dependent setup times. *IIE Transactions*, 31(8):721–731, 1999.

[22] R. Z. Ríos-Mercado and J. F. Bard. An enhanced TSP-based heuristic for makespan minimization in a flow shop with setup times. *Journal of Heuristics*, 5(1):57–74, 1999.

[23] J. V. Simons Jr. Heuristics in flow shop scheduling with sequence dependent setup times. *Omega*, 20(2):215–225, 1992.

[24] B. N. Srikar and S. Ghosh. A MILP model for the n-job, m-stage flowshop with sequence dependent set-up times. *International Journal of Production Research*, 24(6):1459–1474, 1986.

[25] E. F. Stafford and F. T. Tseng. On the Srikar-Ghosh MILP model for the $N \times M$ SDST flowshop problem. *International Journal of Production Research*, 28(10):1817–1830, 1990.

[26] W. Szwarc and J. N. D. Gupta. A flow-shop with sequence-dependent additive setup times. *Naval Research Logistics*, 34(5):619–627, 1987.