Alternative Formulations for Flexible Flowlines with Sequence-dependent Setup Times

Mary E. Kurz
Department of Industrial Engineering
Clemson University
Clemson, SC 29631

Abstract
Alternative IP formulations for scheduling flexible flowlines with sequence-dependent setup times will be presented. While one does not differentiate between the machines at each stage, the other does. The differences in the formulations based on ease of implementation and ease of extension to variants of the problem will be discussed.

Keywords
Integer programming, sequence-dependent setup time, flexible flowline

1. Introduction
This research considers alternative integer programming formulations for scheduling a flexible flowline with sequence-dependent setup times to minimize makespan. Section 2 contains a problem description. Section 3 briefly reviews the related literature. Sections 4 and 5 contain the alternative formulations. In section 6, the formulations are compared. Section 7 concludes and presents directions for future research.

2. Problem Statement
A flexible flowline is a manufacturing environment in which several stages processing jobs in a serial manner. Each stage may have multiple parallel identical machines. Each job is processed by at most one machine at each stage. Let g be the number of stages. Let n be the number of jobs to be scheduled and m^t be the number of machines at stage t. We assume that machines are initially setup for a nominal job 0 at every stage. Job n+1 exists at every stage only to indicate the end of the schedule, if needed. We have the following definitions.

n =number of true jobs to be scheduled	g =number of serial stages
g_i =last stage visited by job i	m^t =number of machines at stage t
p_i^t =processing time for job i at stage t	s_{ij}^t =setup time from job i to job j stage t
S_i =set of stages visited by job i	S^t =set of jobs in (1, …, n) that visit stage t
$S^t(0)$, $S^t(n+1)$ = S^t plus either job 0 or job n+1	

The processing time of job 0 is set at 0. The setup time from job 0 indicates the time to move from the nominal set point state. We assume that all jobs currently in the system must be completed at each stage before the jobs under consideration may begin setup. The completion times of job 0 at each stage are set to the earliest time setup may begin at that stage. The setup time to job n+1 is set at 0; this job only exists to indicate the end of the schedule. We also include the restriction that every stage must be visited by at least as many jobs as there are machines in that stage.

3. Literature Review
Wagner [6] presented a integer linear programming model for scheduling job shops that allowed for various situations such as multiple machines per stage, jobs that did not visit every stage, jobs that had to be processed on more than one machine per stage, and so on. In [6], jobs are explicitly assigned to positions on machines. A condition that this work did not include was the existence of sequence-dependent setup times. Stafford [5] improved upon Wagner's work, focusing on the standard flowshop (with no parallel machines and no sequence-dependent setup times). Srikar and Ghosh [3] considered a permutation flowshop with sequence-dependent setup times in their MILP model, which used many fewer variables than the previous formulations. Srikar and Ghosh [3] used decision

variables that focused on whether a job is scheduled anytime before another job. However, Stafford and Tseng[4] discovered several problems with [3], corrected these and extended this modeling concept to non-sequence-dependent setup time flowshops, no-intermediate-queue flowshops and sequence-dependent setup time, no-intermediate-queue flowshops. Ríos-Mercado and Bard [1] also consider the sequence-dependent setup time flowshop and develop several valid inequalities for formulations based on the traveling salesman problem and the Srikar-Ghosh model.

No literature has been discovered addressing formulations for the flexible flowline with sequence-dependent setup times. The key features that are missing are the multiple machines per stage and the ability of jobs to skip stages.

4. Formulation 1

In the TSP based formulation, following Formulation A from Ríos-Mercado and Bard [1], the following additional variables are used.

r^t =time at which stage t can begin processing, $t = 1,..., g$

c_i^t =completion time for job i at stage t, $t = 1,..., g$, $i = 0,1,..., n$

x_{ij}^t =1 if job i is scheduled immediately before job j at stage t and 0 otherwise

$t = 1,..., g$, $i \in S^t(0)$, $j \in S^t(n+1)$, $i \neq j$

Several observations can be made which guide the formulation of this model. Each stage t has a time at which it can begin processing, which is denoted r^t . We consider the time each stage can begin processing to be the completion time of job 0 at that stage, and we further assume that it is the same for all machines at a stage. Every stage exists independently except that a job is not available to begin setup in stage t until it has completed processing on stage t-1. Each stage can be considered a traveling salesman problem with multiple salesmen, equal to the number of machines in that stage. We assign jobs to machines implicitly; by interpreting a group of variables, such as $x_{03}^1 = 1$, $x_{34}^1 = 1$, and $x_{42}^1 = 1$, we see that jobs 0, 3, 4 and 2 are assigned to a machine in stage 1 in that order. Equations (1) to (10) comprise Formulation 1.

$$\min \quad z \tag{1}$$

subject to

$$\sum_{j \in S^t} x_{0j}^t = m^t \qquad \forall t \tag{2}$$

$$\sum_{j \in S^t(n+1)-\{i\}} x_{ij}^t = 1 \qquad \forall t,\ i \in S^t(0) \tag{3}$$

$$\sum_{i \in S^t(0)-\{j\}} x_{ij}^t = 1 \qquad \forall t,\ j \in S^t(n+1) \tag{4}$$

$$c_0^t = r^t \qquad \forall t \tag{5}$$

$$c_j^t - c_i^t + M^t(1-x_{ij}^t) \geq s_{ij}^t + p_j^t \qquad \forall t,\ i \in S^t(0),\ j \in S^t(n+1),\ i \neq j \tag{6}$$

$$c_j^t - c_j^{t-1} + M_j^t(1-x_{ij}^t) \geq s_{ij}^t + p_j^t \qquad t = 1,..., g,\ i \in S^t(0),\ j \in S^t(n+1),\ i \neq j \tag{7}$$

$$c_i^t \geq c_0^t \qquad \forall t,\ i \in S^t \tag{8}$$

$$c_i^t \geq c_i^{t-1} \qquad t = 2,..., g,\ i \in S^t \tag{9}$$

$$z \geq c_i^{g_i} \qquad i = 1,..., n \tag{10}$$

The objective is to minimize makespan, which is denoted by z and is incorporated into later constraints. Job 0 must be scheduled on every machine in every stage. This is expressed by constraint set (2). At every stage, every job must have exactly one predecessor and exactly one successor. This is expressed by the constraint sets (3) and (4). The completion time of job 0 on each stage is set to be the time each stage is available in constraint set (5). The completion time of two jobs in the same stage depend on whether those jobs are scheduled together or not. If two jobs, namely *i* and *j*, are scheduled in that order with no jobs between them, job *j*'s completion time is at least the sum of job *i*'s completion time, the setup time from *i* to *j*, and job *j*'s processing time. If *i* and *j* are scheduled in any other way, the difference between the two completion times is larger than a very negative number M^t. M^t is an upper bound on the time stage *t* completes processing, similar to the upper bound *A* in Rios-Mercado and Bard (1998). $M^1 = \sum_{i=1}^{n}\left(p_i^1 + \max_{j\in\{0,\ldots,n\}} s_{ji}^1 \right)$ and $M^t = M^{t-1} + \sum_{i=1}^{n}\left(p_i^t + \max_{j\in\{0,\ldots,n\}} s_{ji}^t \right)$. Constraint set (6)enforces the restriction that the machine must be idle before setup can begin. The completion time of a job in one stage also depends on its completion time in the previous stage. We define $c_i^0 = 0,\ i \in \{0,1,\ldots,n\}$. We incorporate this idea by ensuring that the completion time of job *j* in stage *t* is at least as large as the completion time in stage *t*-1 plus the setup from the immediately previous job in stage *t*, plus the processing time of job *j*. Constraint set (7) enforces the concept that the job must be available before setup can begin. The value M_j^t is set to $M_j^t = \max_i \left(s_{ij}^t \right) + p_j^t$. Since job 0 is some sort of surrogate for machine availability, we require every job to be processed after job 0 at every stage. Allowing that some jobs do not actually require processing in every stage, we include constraint set (8). Since jobs must be processed in a linear fashion through the stages, we include the restriction that completion times must be non-decreasing as the stage number increases. This is enforced through constraint set (9). The makespan is defined to be the latest time a job completes processing. Because the flowline is flexible, each job *j*'s last stage is g_j. The makespan is the largest of the completion times of all the jobs, denoted by $c_j^{g_j}$ for each job. Since jobs 0 and *n*+1 exist to begin and end the schedule, we do not include these jobs for consideration in defining the makespan in constraint set (10).

5. Formulation 2

In the second formulation, jobs are explicitly placed into positions on machines, following Wagner [6] and Stafford [5]. If there were one machine at every stage and the jobs were required to be processed in the same order at every stage, this would result in a permutation schedule. However, we will be making the investment in extra decision variables so that permutation schedules will not be required. Moreover, with possibly a different number of machines at every stage and jobs that may skip stages, it is not clear how to define a permutation schedule in this context. A specific job is called job *i*. The job in the *j*th position will be called the *j*th job. The following decision variables are used.

x_{ij}^{tk} =1 if job *i* is scheduled in position *j* on machine *k* in stage *t* and 0 otherwise

e_j^{tk} =setup beginning time for *j*th job on machine *k* in stage *t*

d_j^{tk} =completion time for *j*th job on machine *k* in stage *t*

w_j^{tk} =idle time after *j*th job completes and before start of setup of (*j*+1)st job on machine *k* in stage *t*

T_j^{tk} =processing time of the *j*th job on machine *k* in stage *t*

S_j^{tk} =setup time from the *j*th to (*j*+1)st job on machine *k* in stage *t*

Position 0 on every machine at every stage will be occupied by job 0. Note that a different job 0 can be created for each machine and each stage in this problem, so that the completion of job 0 can represent the end of an in-progress schedule through the appropriate assignment of values to variables d_0^{tk}. The largest position number required for each machine in stage *t* is $\left|S^t\right|$, because job 0 is not counted as one of the jobs in each stage. Equations (11) - (25) comprise Formulation 2.

$$\min \quad z \tag{11}$$

subject to

$$\sum_{j=1}^{|S^t|}\sum_{k=1}^{m^t} x_{ij}^{tk} = 1 \qquad \forall t,\ i\in S^t \tag{12}$$

$$x_{0,0}^{tk} = 1 \qquad \forall t,\ \forall k \tag{13}$$

$$\sum_{i\in S^t} x_{ij}^{tk} \le 1 \qquad \forall t,\ \forall k,\ j=1,\ldots,\left|S^t\right| \tag{14}$$

$$x_{h,j+1}^{tk} \le \sum_{i\in S^t(0)-\{h\}} x_{ij}^{tk} \qquad \forall t,\ \forall k,\ h\in S^t,\ j=1,\ldots,\left|S^t\right|-1 \tag{15}$$

$$T_j^{tk} = \sum_{i\in S^t} p_i^t x_{ij}^{tk} \qquad \forall t,\ \forall k,\ j=0,\ldots,\left|S^t\right| \tag{16}$$

$$S_{j+1}^{tk} = \sum_{i\in S^t-\{h\}}\sum_{h\in S^t} y_{ijh}^{tk} s_{ih}^t \qquad \forall t,\ \forall k,\ j=0,\ldots,\left|S^t\right|\text{-1} \tag{17}$$

$$y_{ijh}^{tk} \le x_{ij}^{tk} \qquad \forall t,\ \forall k,\ i\in S^t,\ h\in S^t-\{i\},\ j=1,\ldots,\left|S^t\right|\text{-1} \tag{18}$$

$$y_{00h}^{tk} \le x_{h1}^{tk} \qquad \forall t,\ \forall k,\ h\in S^t \tag{19}$$

$$y_{ijh}^{tk} \le x_{h,j+1}^{tk} \qquad \forall t,\ \forall k,\ i\in S^t,\ h\in S^t-\{i\},\ j=1,\ldots,\left|S^t\right|\text{-1}$$

$$y_{00h}^{tk} \ge x_{00}^{tk} + x_{h1}^{tk} - 1 \qquad \forall t,\ \forall k,\ h\in S^t \tag{20}$$

$$y_{ijh}^{tk} \ge x_{ij}^{tk} + x_{h,j+1}^{tk} - 1 \qquad \forall t,\ \forall k,\ i\in S^t,\ h\in S^t-\{i\},\ j=1,\ldots,\left|S^t\right|\text{-1}$$

$$d_0^{tk} = r^{tk} \qquad \forall t,\ \forall k \tag{21}$$

$$e_{j+1}^{tk} = d_j^{tk} + w_j^{tk} \qquad \forall t,\ \forall k,\ j=0,\ldots,\left|S^t\right|\text{-1} \tag{22}$$

$$d_j^{tk} = e_j^{tk} + S_j^{tk} + T_j^{tk} \qquad \forall t,\ \forall k,\ j=1,\ldots,\left|S^t\right| \tag{23}$$

$$d_j^{tk} \le e_f^{ul} + M\left(1-x_{if}^{ul}\right) + M\left(1-x_{ij}^{tk}\right) \qquad 1\le t<g,\ \forall k,\ t<u\le g,\ \forall l,\ i\in\left(S^t\bigcap S^u\right),\ j=1,\ldots,\left|S^t\right|,\ f=1,\ldots,\left|S^u\right| \tag{24}$$

$$z \ge d_j^{tk} \qquad \forall t,\ \forall k,\ j=0,\ldots,\left|S^t\right| \tag{25}$$

The objective is to minimize makespan, which is denoted by z and is incorporated into later constraints. Every job that visits stage t must be assigned to exactly one position on one machine in that stage. We will later explicitly assign job 0 to the 0^{th} position, so we need only consider the other jobs that visit stage t. This is expressed by constraint set (12). We explicitly assign job 0 to the 0^{th} position on all machines at all stages in constraint set (13). Every position of every machine in every stage has at most one job assigned to it, expressed by constraint set (14). Constraint set (14) only requires that at most one job is assigned to each possible position. In no way does it requires that jobs be scheduled in consecutive positions. For example, up to this point, a valid assignment may have jobs in positions 0, 1, 3, and 4 on a machine in a particular stage. To prevent this, we require all positions after an empty position to empty as well. This can be enforced by constraint set (15). A couple of "helper" variables are defined to simplify the presentation of the model. These variables are T_j^{tk} and S_j^{tk}, which associate processing and setup times with jobs by position instead of names. For example, if job 3 is the 4^{th} job on stage 2, machine 3, the variable T_4^{23} takes on the value p_3^2. The processing times are associated with the position numbers by using the constraint set (16). The setup time from job i to job h, if job i is assigned to position j and job h is assigned to position j+1 on machine k in stage t will be called the setup to the (j+1)st job, S_{j+1}^{tk}, and can found by the following

$$S_{j+1}^{tk} = \sum_{i\in S^t-\{h\}}\sum_{h\in S^t} x_{ij}^{tk} s_{ih}^t x_{h,j+1}^{tk} \qquad \forall t,\ \forall k,\ j=0,\ldots,\left|S^t\right|\text{-1} \tag{26}$$

This is non-linear. We introduce a new binary variable y_{ijh}^{tk} which is 1 when job i is in position j and job h is in position j+1 on machine k in stage t, and 0 otherwise. Constraint set (26) is replaced by the constraint sets (17), (18) , (19) and (20).

At every machine on every stage, job 0's completion time is that machine's ready time. At this time, we compute the machine's ready time in constraint set (21) as the sum of the processing of job 0 up to and including the current stage, or $r^{tk} = \sum_{u=1}^{t} T_0^{uk}$, though this may vary by application. The (j+1)st job in each stage on each machine can begin processing after the jth job completes processing and some waiting time passes. The waiting time can account for the fact that the (j+1)st job is not done with its processing on the previous stage it visits and machine k on stage t is idle while waiting for the job to arrive. Since the objective is makespan, which is defined by only one job's completion time, the waiting time variable may be positive even when the job is available for setup. This set of scheduling variables allows machines to be idle when jobs are available and is shown in constraint set (22). The jth job's completion time is the sum of its available-to-setup time, the time to setup from the (j-1)st job to the jth job, and its processing time, as shown in constraint set (23). Because we require jobs to finish processing at stage t before setup can begin at stage t+1, we need the available-to-setup time for jobs in stage t to be at least as large as the completion time of the jobs in stage t-1. However, our available-to-setup and completion time variables are in terms of the position each job has, not the jobs themselves. For this reason, we need to translate these times so that we can make the required connections between these times in terms of the jobs themselves. Constraint set (24) ensures that the time setup can begin for job i on all stages $u > t$ must be no earlier than the time job i ends on stage t. The makespan is defined to be the latest time a job completes processing in constraint set (25).

6. Comparing Formulation 1 and 2

Now that both formulations have been described, they can be compared in terms of the number of variables and constraints. Table 1 contains the number of binary and non-binary variables and the number of constraints in each formulation.

Table 1: Formulation Sizes in General

Number of	**Formulation 1**	**Formulation 2**
Binary variables	$\sum_{t=1}^{g}\left(\lvert S^t\rvert\right)\left(\lvert S^t\rvert+1\right)$	$\sum_{t=1}^{g} m^t\left[\lvert S^t\rvert^3-\lvert S^t\rvert^2+2\lvert S^t\rvert+1\right]$
Other variables	$g\left(n+1\right)$	$5\sum_{t=1}^{g}\left(\lvert S^t\rvert+1\right)\left(m^t\right)$
Constraints	$2g+n-\lvert S^1\rvert$ $+2\sum_{t=1}^{g}\left\{\lvert S^t\rvert\left(\lvert S^t\rvert+2\right)\right\}$	$\sum_{t=1}^{g}\left[\begin{array}{l}3\lvert S^t\rvert+4m^t+\\ m^t\lvert S^t\rvert\left(3\lvert S^t\rvert^2-2\lvert S^t\rvert+5\right)\end{array}\right]+$ $\sum_{t=1}^{g-1} m^t\lvert S^t\rvert\sum_{u=t+1}^{g} m^u\lvert S^u\rvert\left(\lvert S^u\cap S^t\rvert\right)$
Constraints if $\lvert S^t\rvert=n$ **and** $m^t=m\ \forall t$	$2g\left(n+1\right)^2$	$g\left[3n+4m+nm\left\{3n^2-5n+8\right\}\right]+$ $m^2n^3\sum_{t=1}^{g-1}\left[\frac{t\left(g-t\right)^2\left(g-t+1\right)}{2}\right]$

The number of constraints and binary variables are much higher in the second formulation. In order to better demonstrate the magnitude of the difference, Table 2 shows the number of binary variables and constraints for several different sized problems. Each cell shows the number of binary variables and constraints in that order. These problems have been solved in CPLEX 7.1 on a 600 MHz Pentium 3 PC. The additional information in each cell shows the solving time and the number of nodes.

Table 2: Empirical Formulation Sizes and Solution Information

Problem	Problem size		Formulation 1	Formulation 2
1	$g = 2$ $m^t = 1,\ \lvert S^t \rvert = 6\ \forall t$	**BV** **Constraints** **Time** **Nodes**	84 196 284.33 s 201624	386 1292 389.36 s 39507
2	$g = 2$ $m^t = 2,\ \lvert S^t \rvert = 6\ \forall t$	**BV** **Constraints** **Time** **Nodes**	84 196 37.81 s 36456	772 3004 3600.21 s, gap 29.9% 90678
3	$g = 2 \quad m^t = 2\ \forall t$ $S^1 = \{1,4,6\}$ $S^2 = \{2,3,4,5,6\}$	**BV** **Constraints** **Time** **Nodes**	42 116 0.45 s 725	272 876 4.49 s 555

Clearly, the number of binary variables and constraints in much larger for Formulation 2. In these few small examples, the solving time required in the solution of Formulation 2 IPs is much higher than in the Formulation 1 IPs. In problem 2, Formulation 2 was not able to solve the problem optimally in the 1 hour computational time limit imposed, while Formulation 1 solved the problem optimally in less than a minute. Problem 2 differs from Problem 1 structurally only in that the stages have two machines each. Formulation 2 cannot tell the difference between these two machines at each stage, which certainly causes the solver to spend time considering alternative solutions that are identical. Problem 3 is easily solved by both formulations, even though it has two machines at each stage. This problem is easier because only two jobs visit both stages, so the interaction between the stages is less when compared to problems 1 and 2.

Despite the above performance, Formulation 2 has significant advantages over Formulation 1 due to the adaptations that Formulation 2 can accommodate. Formulation 2 can accommodate machines at stages with different ready times, non-identical processing times and differing capabilities. Formulation 2 can also accommodate inserted idle time in schedules, as well as times during which machines may not be available, such as during planned maintenance. For this reason, we believe that the basic structure of Formulation 2 has an inherent value.

7. Conclusion

Two formulations have been presented for scheduling the flexible flowline with sequence-dependent setup times. These formulations differ in their modeling perspective. Though the second one is much larger, it is more adaptable to variants of the problem.

Several strategies will be pursued in improving Formulation 2. The first strategy will be to add constraints to differentiate between machines in each stage, in an effort to break the symmetry that exists in the stages, following work done by Sherali *et al* in the design of optical networks [2]. The second strategy will focus on redefining the variables, since all the information in the x variables is replicated in the y variables. The effectiveness of these strategies will be evaluated by generating and solving instances of this problem.

References

1. Ríos-Mercado, R.Z. and Bard, J.F., 1998, "Computational experience with a branch-and-cut algorithm for flowshop scheduling with setups," Computers and Operations Research, 25(5), 351-366.
2. Sherali, H.D., Smith, J.C., and Lee, Y., 2000, "Enhanced model representations for an intra-ring synchronous optical network design problem allowing demand splitting," INFORMS Journal on Computing, 12(4), 284-298.
3. Srikar, B.N and Ghosh, S., 1986, "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.
4. Stafford, E.F. and Tseng, F.T., 1990, "On the Srikar-Ghosh MILP model for the *N*x*M* SDST flowshop problem," International Journal of Production Research, 28(10), 1817-1830.
5. Stafford, E.F., 1988, "On the development of a mixed-integer linear programming model for the flowshop sequencing problem, " Journal of the Operational Research Society, 39(12), 1163-1174.
6. Wagner, H.M., 1959, "An integer linear-programming model for machine scheduling," Naval Research Logistics Quarterly, 6, 131-140.