5th International Conference on Industrial Engineering and Industrial Management
XV Congreso de Ingeniería de Organización
Cartagena, 7 a 9 de Septiembre de 2011

The three stage assembly permutation flowshop scheduling problem

Carlos Andrés[1], Sara Hatami[2]

[1] Universidad Politecnica de Valencia, Spain. candres@omp.upv.es
2 Islamic Azad University, Ghazvin, Iran. sara_sodi@yahoo.com

Keywords: assembly flowshop scheduling problem, sequences dependent setup time-SDST

1 Introduction

The assembly flowshop scheduling problem has been studied recently due to its applicability in real life scheduling problems. It arises when various fabrication operations are performed concurrently in one stage. It was firstly introduced by Lee et al. (1993) in a flowshop environment. Later, Potts et al. (1995) considered the two-stage assembly flowshop problem with m concurrent operations in the first stage and an assembly operation in the second stage with the makespan objective, they showed that the considered problem is NP-hard in the strong sense even when the number of machines in the first stage is equal to two.

Allahverdi et al. (2007) and Al-Anzi et al. (2009) considered two bicriteria two-stage assembly flowshop scheduling problems and proposed some metaheuristics. Previously, Al-Anzi et al. (2007) had considered the two-stage assembly flowshop scheduling problem with consideration of separate setup times from processing times and tried to minimize maximum lateness as objective function.

Koulamas et al. (2007) extended the two-stage assembly flowshop to three-stage assembly flowshop scheduling problem with the objective of minimizing the makespan. The first stage manufactures various fabrication operations concurrently, the second one collected and transported them into an assembly stage as final stage for an assembly operation. They analyzed the worst-case ratio bound for several heuristics for the considered problem and they also analyzed the worst-case absolute bound for a heuristic based on compact vector summation techniques.

In this paper we considered the three-stage assembly flowshop problem with sequences dependent setup time (SDST) on first and third stages with the objective of minimizing total completion time. The problem is described in detail in the next section, and a mathematical model is proposed and tested in Section 3. Finally the summary of the work is presented in section 4.

2 Problem description

The three-stage assembly flowshop is a generalization of the two-stage assembly flowshop where an intermediate stage is taken into account. This stage works as a transfer so the second stage collects and delivers the manufactured components from first stage (parallel machines, production areas, other plants) to third stage to assemble them. The problem consists of an assembly flowshop with three stages. The first one has m unrelated parallel machines to manufacture m components independently to make each job (each job consist of m components). Each machine has sequence-dependent setup times. The second stage is a transfer machine for collecting and transferring manufactured components. This transfer machine starts its work when all m components of each job are completed at first stage. Finally, the third stage has an assembly machine for assembling all components of each job to one final product. This machine has sequence-dependent setup times too.

Thus all jobs consist of m+2 operations, m at first stage, 1 at second stage and the last one at assemble stage. Each machine can process only one job at time. There are n jobs to schedule, and we try to minimize total completion time. All machines are available at time zero and all jobs are ready to be processed at time zero. No interruption before job completion is allowed. Figure 1. shows the Gantt chart view of the problem. In order to simplify the problem, our research will be restricted to permutation schedules to find optimal solution.

$$a + b = c \qquad (1.1)$$

3 Model formulation

Mathematical models have been widely used to study and solve scheduling problems in the literature. Their main advantages are to provide a better insight of the problem and optimal solutions. Unfortunately they are limited to small problem sizes. A mathematical model is presented in this paper based in position variables of Wagner (1959). Stafford et al (2005) and Tseng et al. (2008) have proved that this kind of models have some of the best performing models for the regular PFSP. Naderi et al. (in press) has carried out an extensive computational study in the distributed permutation flowshop scheduling problem and the results are similar.

We propose two mathematical formulations for the problem. Each model consists of an objective function subject to two main types of constraints: assignment constraints and precedence constraints. There are some constraints used to calculate the completion times at every stage. The model takes into account all of the N jobs in the third stage and consists of M parts (each part is made in one parallel machine in stage-one). In addition, we suppose the sequence is the same in all the machines (production, transportation, and assembly) and so the same variables can be used to represent the sequence of parts and jobs. Before presenting both models some common notation is required and presented in table 1.

Table 1 Common notation: parameters and subscripts

Parameters	Deacription
i, j	denotes the jobs (and corresponding parts in stages one and two) i, j =0,1,2,...,N where 0 represents the dummy job
k	denotes the position in the sequence k=1,2,…, N

l	denotes the machine at stage-one. l=1,2,…,M
$P_{j,l}$	processing time of the part j on machine l at the first stage
$S_{i,j,l}$	setup time from part i to part j at machine l at the first stage
PT_j	processing time of the part j on machine l at the second stage
PA_j	processing time of the part j on machine l at the third stage
$SA_{h,i}$	setup time from part j to part k at the third stage

3. 1 Model one (sequence based model)

In this model we use binary variables to model the relative sequence between jobs. It is necessary to use a parameter M to ensure that a machine can process, at most, one job at a time. The variables are defined as following:

Table 2 Definition of variables

Variables	Deacription
$x_{j,k}$	to 1 if a part j is an immediate predecessor of part k, otherwise zero.
$x_{0,j}$	equal to 1 if part j is the first part in the sequence, otherwise zero.
$CP_{j,l}$	completion time of job j at machine l in the first stage.
CT_j	completion time of job j in the second stage.
C_j	completion time of job j in the third stage.
C	Total completion time $C = \sum_{j=1}^{n} C_j$

We can now formulate the mathematical model:

$$\min C \tag{1.1}$$

s.t.

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{i,j} = 1 \qquad \forall j \tag{1.2}$$

$$\sum_{\substack{j=1 \\ j\neq i}}^{n} x_{i,j} \leq 1 \qquad \forall i \quad (1.3)$$

$$\sum_{j=1}^{n} x_{0,j} = 1 \qquad (1.4)$$

$$CP_{j,l} \geq (S_{0,j,l} + P_{j,l}) + \mathrm{M} \cdot (x_{0,j} - 1) \qquad \forall j, \forall l \quad (1.5)$$

$$CP_{j,l} \geq CP_{i,l} + (S_{i,j,l} + P_{j,l}) + \mathrm{M} \cdot (x_{i,j} - 1) \qquad \forall i,\ j \neq i, \forall l \quad (1.6)$$

$$CT_i \geq CP_{i,l} + PT_{i,l} \qquad \forall i, \forall l \quad (1.7)$$

$$CT_j \geq CT_i + PT_j + \mathrm{M} \cdot (x_{i,j} - 1) \qquad \forall i,\ j \neq i \quad (1.8)$$

$$C_j \geq (SA_{0,j} + P_j) + \mathrm{M} \cdot (x_{0,j} - 1) \qquad \forall j, \forall l \quad (1.9)$$

$$C_j \geq C_i + (S_{i,j} + PA_j) + \mathrm{M} \cdot (x_{i,j} - 1) \qquad \forall i,\ j \neq i \quad (1.10)$$

$$\mathrm{C}_i \geq \mathrm{CT}_i + PA_i \qquad \forall i \quad (1.11)$$

$$C = \sum_{j=1}^{n} C_j \qquad (1.12)$$

Equation (1.2) ensures that every part j is assigned to one machine in stage-one. Equation (1.3) is similar to Equation (1.2) and forces each job to have a unique predecessor (or to be last). Equation (1.4) forces the set of jobs to have a unique dummy job. Constraints (1.5) and (1.6) compute completion times for every job at the machines in the first stage using processing and sequence dependent setup times. Value M represents a large number so as to make the constraint redundant if the sum of the assignment variables is one. Constraint (1.7) establishes the relationship between the completion times of a job in stages one and two. Notice that setup at stage-two can begin while the job is being processed at stage-two (anticipatory setup times). Constraint (1.8) computes completion times of consecutive jobs in the transportation stage. Constraints (1.9) and (1.10) represent the relationship between the completion times of the same part in stages two and three. Constraint (1.11) has the same meaning as Equation (1.7) and represents the relationship between completion times of a job in stages two and three. Finally, equation (1.12) computes the objective function.

3. 2 Model 2 (position based model)

This model is based on the Wagner's approach, Wagner (1959) which has been recognized recently by Naderi and Ruiz (2010) as one of the best performing models for regular and distributed flowshops. This is because the approach does not use the M parameter to define the dichotomous constraints which are known to produce poor linear relaxations. Model 2 consists of the following:

Table 3 Definition of variables

Variables	Deacription
$CP_{k,l}$	completion time of job in position k in machine l in the first stage.
CT_k	completion time of job in position k in the second stage.
C_k	completion time of job in position k in the third stage.
$x_{j,k}$	equal to 1 if job j occupies position k in the sequence, otherwise zero.
C	Total completion time $C = \sum_{j=1}^{n} C_j$

The model can be defined as follows:

$$\min C \tag{1.13}$$

s.t.

$$\sum_{k=1}^{n} x_{j,k} = 1 \qquad \forall j \tag{1.14}$$

$$\sum_{j=1}^{n} x_{j,k} = 1 \qquad \forall k \tag{1.15}$$

$$\mathrm{CP}_{1,l} \geq \sum_{j=1}^{n} \left((S_{0,j,l} + P_{j,l}) \cdot x_{j,1} \right) \tag{1.16}$$

$$\mathrm{CP}_{k,l} \geq \mathrm{CP}_{k-1,l} + \sum_{j=1}^{n} \left(P_{j,l} \cdot x_{j,k} \right) + \sum_{i=1}^{n} \sum_{j=1}^{n} \left(S_{i,j,l} \cdot x_{i,k-1} \cdot x_{j,k} \right) \qquad \forall k > 1, \forall l \tag{1.17}$$

$$\mathrm{CT}_k \geq \mathrm{CP}_{k,l} + \sum_{j=1}^{n}\left(PT_j \cdot x_{j,k}\right) \qquad \forall k \quad (1.18)$$

$$\mathrm{CT}_1 \geq \sum_{j=1}^{n}\left(PT_j \cdot x_{j,1}\right) \quad (1.19)$$

$$\mathrm{CT}_k \geq \mathrm{CT}_{k-1} + \sum_{j=1}^{n}\left(PT_j \cdot x_{j,k}\right) \qquad \forall k > 1 \quad (1.20)$$

$$\mathrm{C}_k \geq \mathrm{CT}_k + \sum_{j=1}^{n}\left(PA_j \cdot x_{j,k}\right) \qquad \forall k \quad (1.21)$$

$$\mathrm{C}_1 \geq \sum_{j=1}^{n}\left((SA_{0,j} + PA_j) \cdot x_{j,1}\right) \quad (1.22)$$

$$\mathrm{C}_k \geq \mathrm{C}_{k-1} + \sum_{j=1}^{n}\left(PA_j \cdot x_{j,k}\right) + \sum_{i=1}^{n}\sum_{j=1}^{n}\left(SA_{i,j} \cdot x_{i,k-1} \cdot x_{j,k}\right) \qquad \forall k > 1 \quad (1.23)$$

$$C = \sum_{j=1}^{n} C_j \quad (1.23)$$

Equation (1.14) ensures that every position has only one part assigned while Equation (1.15) is similar to Equation (1.14) and ensures each job is assigned to only one position. Equations (1.16) and (1.17) compute the completion time of every job at the machines in the first stage using processing and sequence dependent setup times. Constraint (1.18) establishes the relationship between completion time for jobs in stages one and two. Constraints (1.19) and (1.20) compute starting times for consecutive jobs in the transport stage. Constraint (1.21) has the same meaning as Equation (1.18) and represents the relationship between completion times for jobs in stages two and three. Finally, Equation (1.22) and (1.23) compute the finishing time of every part at the third stage. This relation is used in Equation (1.24) to compute the objective function. It is possible to replace the non-linear term in constraints (1.17) and (1.23) using the auxiliary variable $\delta_{i,j,k} = x_{i,k-1} \cdot x_{j,k}$ as follows:

$$\begin{aligned} & -X_{i,k-1} + \delta_{i,j,k} \leq 0 \\ & -X_{j,k} + \delta_{i,j,k} \leq 0 \\ & X_{i,k-1} + X_{j,k} - \delta_{i,j,k} \leq 1 \end{aligned} \qquad \forall k > 1, \forall i, \forall j \quad (1.24)$$

4 Experimental results

Both models can only be tested in moderately sized instances. However, the first test is to see if both models offer the same performance. To do this we solved both models using five instances with n=6,10 and m=2, 4; processing and setup times were randomly generated $P_{j,l} \in \mathrm{U}[0,100]$, $PT_j \in \mathrm{U}[0,10]$, $PA_j \in \mathrm{U}[0,100]$, $S_{i,j,l} \in \mathrm{U}[1,20]$ and $SA_{i,j} \in \mathrm{U}[1,20]$. Each model was run using the CPLEX 12.1.0 solver with a time limit of 1800s.

The results are represented in the following tables where % Opt is the percentage of times the models were solved optimally; %Gap is the medium gap percentage; time is the medium time needed for solving the models (we set an upper limit of 1800 seconds); Std %Gap is the standard deviation of the gap for the five runs at every set of instances; and St d Time is the standard deviation of the time needed for solving the models.

It can be seen that Model 2 performs better than Model 1 for all the instance sizes. In fact, for n=10, the only model that provides optimal solutions in less than 1800 seconds is Model 2. Regarding the number of machines in the first stage, when n=6, computational time is unaffected, but for n=10 and non-optimal solutions, the standard deviation of the gap increases with the number of machines. Finally, when optimal solutions are considered, the standard deviation for solution time is clearly affected for the number of machines (Model 2).

Table 4. Results for Model 1

n	m	% Opt	% GAP	Time	St d %Gap	St d Time
6	2	100	0	0.996	0	0.55
10	2	0	48.43	1800	3.97	0
6	4	100	0	0.946	0	0.11
10	4	0	39.97	1800	5.71	0

Table 5. Results for Model 2

n	m	% Opt	% GAP	Time	St d %Gap	St d Time
6	2	0.05	0	0.266	0	100
10	2	6.96	0	13.62	0	100
6	4	0.07	0	0.268	0	100
10	4	15.38	0	30.306	0	100

Thus a more extensive set of experiments has been carried out to test the performance of Model 2. The general performance of the MILP Model 2 is evaluated with a set of small-sized instances that are randomly generated as follows: the computational experiences were carried out with the following values: n={10,12,15,17} and m={2,3,4}. Processing and setup times were randomly generated from discrete uniform distributions. The ranges of the distributions were $P_{j,l} \in \mathrm{U}[0,100]$, $PT_j \in \mathrm{U}[0,10]$ and $PA_j \in \mathrm{U}[0,100]$. We generated an LP file for each instance. There are 12 combinations of n, and m, while each combination has been replicated

five times, and thus we analyzed 60 different MILP models. Each model was run using the CPLEX 12.1.0 solver with a time limit of 1800s. Table 3 shows, for each value of n and m: the percentage of optimum solutions found (%opt); the average gap in percentage for the cases where the optimum solution could not be found (%Gap); the average time needed when an optimal solution is found or the algorithm reaches the time limit (Time); the standard deviation of %Gap; and the standard deviation of Time. When the optimal solution is not reached, at least a feasible integer solution was obtained.

Table 6 Results for small-sized instances

n	m	% Opt	% GAP	Time	St d %Gap	St d Time
10	2	100	0	17,1	0	3,07
12	2	100	0	83,2	0	17,2
15	2	60	1,5	1282,5	2,4	560,3
17	2	0	9,8	1800	1,5	0
10	3	100	0	29,6	0	22,3
12	3	100	0	138,7	0	119,7
15	3	60	2,1	1304,5	3,1	671,2
17	3	0	7,2	1800	2,2	0
10	4	100	0	24,82	0	5,9
12	4	100	0	202	0	216,9
15	4	20	3	1555,9	3,5	545,7
17	4	0	10,3	1800	3,6	0

As it can be seen in the table, it was not possible to get optimum solution for all instances in the allowed CPU time. Overall from the 60 instances we were able to get optimal solutions in 37 (61,6%). As expected, number of jobs has a major impact over the performance of the model. As a matter of fact, no instance with n=17 were solved with CPLEX optimally. All instances with n=10 and n=12 were solved with in less than 202 seconds (less than 85 seconds when m=2). Regarding to m, we can remark its effect is not so clear as n. It seems when m increases it is more difficult to solve the problem optimally but there is not a linear relation. The gap was very low in almost all instances, showing the model relaxation is good.

5 Conclusions

In this paper we have proposed a mathematical MIP model for scheduling three stage assembly flowshop with sequence dependent setup times. Experiments show we were able to find optimal solutions for problems up n=15 and m=4. No problem with n=17 were solved optimally in less than 30 minutes of CPU time. A more comprehensive study is being carried out and it will be presented at the workshop.

References

1. Al-Anzi, F.S. and Allahverdi, A (2007), “A self-adaptive differential evolution heuristic for two stage assembly scheduling problem to minimize maximum lateness with setup times”, European Journal of Operational Research, Vol. 182, pp. 80–94.

2. Allahverdi, A. and Al-Anzi, F.S. (2007), “The two-stage assembly flowshop scheduling problem with bicriteria of makespan and mean completion time”, International Journal of Advanced Manufacturing Technology, Vol. 37, pp. 166-177.

3. Koulamas, C. and Kyparisis, G. (2007), “A note on the two-stage assembly flow shop scheduling problem with uniform parallel machines”, European Journal of Operational Research, Vol. 182, pp. 945–951.

4. Lee, C.Y., Cheng, T.C.E. and Lin, B.M.T, (1993), “Minimizing the makespan in the 3-machine assembly-type flowshop scheduling problem”, Management Science, Vol. 39, pp. 616–625

5. Naderi, B. and Ruiz, R (in press) “The distributed permutation flowshop scheduling problem”, Computers and Operations Research, doi:10.1016/j.cor.2009.06.019

6. Potts, C.N., Sevastianov, S.V., Strusevich, V.A., Van Wassenhove, L.N., Zwaneveld, C.M, (1995), “The two-stage assembly scheduling problem: Complexity and approximation”, Operations Research, Vol. 43, pp. 346–355.

7. Stafford, E. F. Tseng, F. T. and Gupta, J. N. D. (2005), “Comparative evaluation of MILP flowshop models”, Journal of the Operational Research Society, Vol. 56, pp 88–101.

8. Tseng, F. T. and Stafford, E. F. “New MILP models for the permutation flowshop problem”, Journal of the Operational Research Society, Vol. 59, pp 1373–86.

9. Wagner, H. M. (1959), “An integer linear-programming model for machine scheduling”, Naval Research Logistics Quarterly, Vol. 6(2), pp. 131–40.