KARIŞIK MODELLİ ÜRETİM HATTI SIRALAMA PROBLEMİNE LAGRANGE GEVŞETME YAKLAŞIMI

Deniz Türsel Eliiyi, Melih Özlen
Orta Doğu Teknik Üniversitesi, Endüstri Mühendisliği Bölümü, 06531, Ankara

Özet

Bu çalışmada, değişik amaç fonksiyonları içeren karışık modelli üretim hattı sıralama problemi üzerinde durulmuştur. Karışık Modelli Üretim Hattı'nda (KMH) ürünler konveyor üzerinde taşınmakta, ve aynı ürünün değişik modelleri, karışık şekilde aynı hat üzerinde üretilmektedir. Toplam çevrim zamanının ve toplam akış zamanının enazlanması olmak üzere iki değişik amaç fonksiyonu üzerinde çalışılmıştır. Bu iki amaç fonksiyonunu için Lagrange gevşetmeye dayalı bir yaklaşıklama algoritması geliştirilmiştir. Algoritma çözüm kalitesi ve çözüm süresi ölçütleri kullanılarak test edilmiş ve büyük boyutlu problemler için optimale çok yakın sonuçlara kısa zamanlarda ulaşılmıştır.

Anathar Kelimeler: *Karışık Modelli Üretim Hattı, Çizelgeleme, Lagrange Gevşetme*

A LAGRANGEAN RELAXATION APPROACH FOR THE MIXED MODEL FLOW LINE SEQUENCING PROBLEM

Abstract

In this study, a mixed model flow line sequencing problem with two objectives is considered. Mixed model flow line (MML) is known to be a special case of production lines where products are transported on a conveyor belt, and different models of the same product are inter-mixed to be assembled on the same line. We have focused on product-fixed, rate-synchronous lines with variable launching. Our objective functions are minimizing makespan, and minimizing total flow time. An approximation algorithm based on Lagrangean relaxation is developed especially for large-size instances of the problem with these objectives, and tested in terms of solution quality and computational efficiency.

Keywords: *Mixed Model Flow Lines, Scheduling, Lagrangean Relaxation*

1. Introduction

A mixed model flow line sequencing problem (MMS) is considered in this study. In a mixed model flow line, there are N jobs that should be processed by M stations in the same processing order. All jobs and stations are available at time zero. There are a total of R product models, and each job belongs to a particular model. The transfers between the stations are synchronous, i.e., the transfers take place at the times when all stations finish their jobs. Cycle time or cycle length is defined as the length of time between two transfers. It is assumed that the speed of the material handling mechanism can be adjusted by the end of each transfer. Let C_j denote the time spent between j-1 st and j th transfers, i.e. the cycle time of the j th cycle, where j = 1,..., N+M-1, as there are N+M-1 cycles in a schedule. Also, let p_{rm} denote the processing time of a job of model r on station m, where r = 1,...,R, and m = 1,...,M. Then, we may define two different objectives as follows: Objective makespan is $\sum_{j=1}^{N+M-1} C_j$. Total flow time (total throughput time) objective becomes $N\sum_{j=1}^{M} C_j + \sum_{j=M+1}^{M+N-1} (N+M-j)C_j$. The makespan is important when the number of jobs is finite. It is closely related to the throughput objective, which is an important performance measure for many facilities. Heuristics that tend to minimize the makespan in a machine environment with a finite number of jobs also tend to maximize the throughput rate, when the flow of jobs is constant over time (Pinedo and Chao, 1999). On the other hand, one can try to minimize Work-in-Process (WIP) inventory as it ties up capital and large amounts can block operations. Flow time is a performance measure surrogate for WIP. Figure 1 illustrates the concept of cycle times in an example mixed model flow line with 3 jobs and 3 stations. Assume all jobs are of different models in the example.

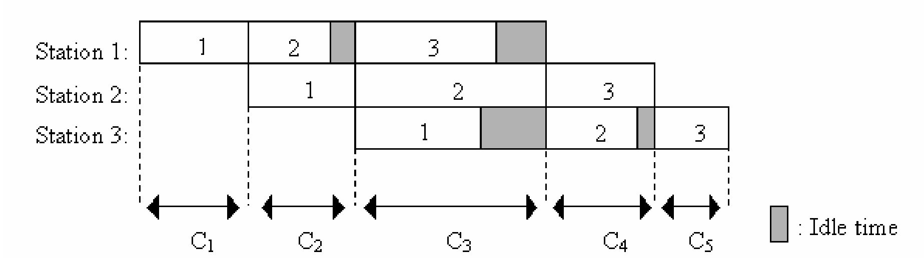

Figure 1. Cycle Times in a Mixed Model Flow Line

In the figure, it can be observed that only the first station is busy during the first cycle. As the first cycle ends, the job processed on the first station is transferred to the second station, and a new job is loaded to the first one. Then, job 1 passes to the third station and job 2 passes to the second station, where a new job is loaded to the emptied first station. The cycle times, which are computed as the maximum of the processing time of the jobs that are on the line simultaneously, become: $C_1=p_{11}$, $C_2=\max\{p_{21},p_{12}\}$, $C_3=\max\{p_{31}, p_{22}, p_{13}\}$, $C_4=\max\{p_{32}, p_{23}\}$, $C_5=p_{33}$. The makespan is $C_1 + C_2 + C_3 + C_4 + C_5$, where the total flow time is $3C_1 + 3C_2 + 3C_3 + 2C_4 + C_5$.

Our study differs from classical mixed model sequencing, flow shop studies and multi-model line in the following aspects: The speed of transfer mechanism is fixed in classical mixed model sequencing, i.e. $C_t= C$ for all t, whereas the speed of transfer mechanism is adjusted at the end of each transfer by assuming arbitrary C_t in our study. In a classical flow shop model, the transfers between stations are asynchronous, i.e. the complete job is immediately transferred to the next station, whereas we assume synchronous transfers. In a multi-model line, models are produced in batches, and switching from one model to another requires rearrangement of the assembly line. However, various models are intermixed on the line in MMS.

There are a large number of papers on a variety of flow shop models in the literature, and the majority of these studies are on makespan minimization. Johnson (1954) considered the problem for two machines, and provided an optimizing rule for makespan minimization. Garey, Johnson and Sethi (1976) proved that the problem is NP-hard for three machines. Other studies on flow shop models with makespan minimization include those of Palmer (1965), Campbell, Dudek and Smith (1970), Gupta (1972), Baker (1975), Widmer and Hertz (1989) and Taillard (1990). As for the problem with flow time minimization, Lenstra et al. (1977) shown that the problem is NP-hard. Ignall and Schrage (1965), Bansal (1977), and Szwarc (1983) attempted optimal solutions. Special purpose algorithms have been proposed by Gupta (1972), Miyazaki et al. (1978), Ho and Chang (1991), Rajendran and Chaudhuri (1991), Rajendran (1993), and Tang and Liu (2002).

Mixed model sequencing problems are studied by a number of researchers. Yano and Rachamadugu (1991) provided a mathematical programming formulation of the problem that minimizes total utility work. Bolat et al. (1994) developed two heuristic algorithms and a branch and bound procedure for minimizing total setup and utility work costs simultaneously. Tsai (1995) studied the problem with the two objectives of minimizing the risk of stopping a conveyor, and minimizing total utility work. Miltenburg (1989) considered the balancing problem, i.e. the problem of keeping the quantity of each model as constant as possible. Kubiak and Sethi (1991) studied the problem of minimizing the total deviation of actual production from the desired production, where Steiner and Yeomans (1993) considered minimizing maximum deviation.

In this study, we develop an approximation algorithm for the problem of minimizing makespan on synchronous mixed model lines based on Lagrangean relaxation. To solve the Lagrangean dual problem, we apply the well-known subgradient optimization procedure (Fisher, 1981) that iterates between upper and lower bounding procedures, and updates the Lagrangean multipliers through iterations. The only study on the problem with makespan objective is that of Soylu (2002), in which an optimizing Branch and Bound algorithm is developed for small instances. There is no published work on the problem with total flow time objective as to the best of our knowledge. In the next section, we provide a definition of our problem, and present its mathematical formulation and computational complexity status. In Section 3, we present our approximation procedure. In Section 4, we provide the results of our computational experiments. Conclusions are addressed in Section 5.

2. Problem Definition

There are M stations, N jobs, and R job models. Each job belongs to one model, and D_r denotes the demand (or order) for model r. Hence, $\sum_{r=1}^{R} D_r = N$. Obviously there will be N positions in the schedule to be occupied by N jobs. We assume that the jobs of the same model have the same processing time on each station; hence, we let p_{rm} denote the processing time of a job a job of model r on station m. A binary variable X_{rn} takes the value of one if a job of model r occupies the n th position in the schedule, and zero otherwise.

Throughout the study we assume that:

- All jobs and stations are available at time zero.
- Each station can process at most one job at a time.
- The stations are indexed in the order of being visited by the jobs, i.e. all jobs first enter station 1, then station 2, and so on.
- Transfer times of the jobs between stations are negligible.
- Job splitting and preemption are not allowed; a job once started on a station is processed until it finishes its operation on that station.
- The setup times are independent of the job sequence and are therefore included in the processing times of jobs.

Under these assumptions and with the given definitions, the problem with makespan objective can be modeled as a mixed integer program as follows:

$$z = \text{Min} \sum_{j=1}^{N+M-1} C_j \qquad (0)$$

subject to:

$$\sum_{r=1}^{R} X_{r\,n} = 1 \qquad n = 1,..., N \qquad (1)$$

$$\sum_{n=1}^{N} X_{r\,n} = D_r \qquad r = 1,..., R \qquad (2)$$

$$C_j \geq \sum_{r=1}^{R} p_{rk} X_{r\,(j-k+1)} \qquad j,k \in JK = \begin{Bmatrix} j=1,...,M-1 & k=1,...,j \\ j=M,...,N & k=1,...,M \\ j=N+1,...,N+M-1 & k=j-N+1,...,M \end{Bmatrix} \qquad (3)$$

$$X_{r\,n} \in \{0,1\} \qquad r = 1,..., R \qquad n = 1,..., N \qquad (4)$$

$$C_j \geq 0 \qquad j = 1,..., N+M-1 \qquad (5)$$

The objective function expressed in (0) minimizes makespan. Constraint set (1) guarantees that every position in the sequence is occupied by one job. Constraint set (2) implies that the demand for each model is exactly satisfied. Constraint set (3) identifies each cycle time as the maximum of the processing times of all jobs being processed in that cycle. Note that there are MN constraints in constraint set (3). Finally, constraints (4) and (5) are the integrality and nonnegativity constraints, respectively. Constraint (5) is redundant, but is required to strengthen the formulation of the relaxed problem, as will be explained in Section 3.

The problem with total flow time objective has the same set of constraints with the following objective function:

$$z = \text{Min} \qquad N\sum_{j=1}^{M} C_j + \sum_{j=M+1}^{M+N-1} (N+M-j)C_j$$

The classical flow shop problem is NP-hard in the strong sense for $m \geq 3$ (Garey, Johnson and Sethi, 1976). However, the complexity of MMS, which has the additional constraint of synchronous transfers, is still open.

3. Lagrangean Relaxation

In this section, we present an approximation algorithm based on Lagrangean relaxation for MMS with both makespan and total flow time objectives. The algorithm is intended especially for large-size instances of the problem. In Lagrangean relaxation, the aim is to relax some constraints of the original problem so that the resulting relaxed model becomes an easy-to-solve one. Hence, there is a trade-off in making relaxations: Either the relaxation is made so that the relaxed model solved easily at the cost of generating loose bounds, or some tighter bounds can be found by spending more effort on the solution of the relaxed problem.

Recall the mathematical formulation of MMS. We choose to relax constraint set (3) with Lagrange multipliers, λ_{jk}. Note that this constraint set is the complicating one as it forces synchronous transfers. For the makespan objective, the relaxed model can be expressed as follows:

$$LR(z) = \text{ Min } \sum_{j=1}^{N+M-1} C_j - \sum_{j,k \in JK} \lambda_{jk} C_j + \sum_{j,k \in JK} \lambda_{jk} \sum_{r=1}^{R} p_{rk} X_{r(j-k+1)}$$

subject to: (1), (2), (4), (5), and $\lambda_{jk} \geq 0$, $j,k \in JK$.

When the objective is the minimization of total flow time, the relaxed model is the same, except the coefficients of C_j variables:

$$LR(z) = \text{ Min } N\sum_{j=1}^{M} C_j + \sum_{j=M+1}^{M+N-1} (N+M-j)C_j - \sum_{j,k \in JK} \lambda_{jk} C_j + \sum_{j,k \in JK} \lambda_{jk} \sum_{r=1}^{R} p_{rk} X_{r(j-k+1)}$$

subject to: (1), (2), (4), (5), and $\lambda_{jk} \geq 0$, $j,k \in JK$.

The solution procedure applies to both models, therefore we present only for makespan objective.

When λ_{jk} values are known, the relaxed model decomposes into two subproblems, one in assignment and one in cycle time variables, as follows:

Subproblem 1:

$$LR_1(z) = \text{ Min } \sum_{j,k \in JK} \lambda_{jk} \sum_{r=1}^{R} p_{rk} X_{r(j-k+1)}$$

subject to:

$$\sum_{r=1}^{R} X_{rn} = 1 \qquad n = 1,..., N$$

(1)

$$\sum_{n=1}^{N} X_{rn} = D_r \qquad r = 1,..., R$$

(2)

$$X_{rn} \leq 1 \qquad r = 1,..., R \qquad n = 1,..., N \qquad (4')$$

Note that constraint set (4) in the original formulation can be replaced with (4'), as the subproblem becomes a network flow problem with known λ_{jk}. This subproblem can be solved by network simplex.

Subproblem 2:

$$LR_2(z) = \text{ Min } \sum_{j=1}^{M+N+1} C_j - \sum_{j,k \in JK} \lambda_{jk} C_j$$

subject to:

$$C_j \geq 0 \qquad j = 1,..., N+M-1$$

(5)

$$C_j \leq \max_{r,k} \{p_{rk}\} \quad r = 1,..., R \qquad j,k \in JK = \begin{Bmatrix} j=1,...,M-1 & k=1,...,j \\ j=M,...,N & k=1,...,M \\ j=N+1,...,N+M-1 & k=j-N+1,...,M \end{Bmatrix} \qquad (6)$$

Constraint set (6) is a set of valid cuts for Subproblem 2, which is added to strengthen the formulation. Each cycle time is computed as the maximum of the processing times of the jobs that are in operation at that cycle. Therefore, a cycle time is dependent on the machines that are in operation during that cycle. Since any job can be scheduled during a cycle, the maximum of the processing times of the jobs that can be scheduled on the machines that are in operation in a specific cycle is taken as an upper bound for that cycle. This bound is utilized in the Lagrangean heuristic lower bound computation.

Subproblem 2 is a simple unconstrained linear programming model with upper and lower bounds on variables, and can be solved easily by inspection. We simply set a variable C_j equal to zero, if its objective function coefficient is nonnegative. If not, we set C_j equal to its upper bound, which is established by constraint set (6).

Note that, the optimal objective function value of the relaxed problem is the sum of the optimal objective function values of the two subproblems. That is; $LR(z) = LR_1(z) + LR_2(z)$. This value constitutes a lower bound for the original problem, MMS. To convert this infeasible solution into a feasible one, we employ the following Lagrangean heuristic procedure.

Lagrangean Heuristic:

S1. Using the optimal solution X_{rn}^* of Subproblem 1, compute C_j values for each cycle by means of the relaxed constraint set (3).

S2. Using computed C_j values, calculate the new objective function value, LR'(z), which constitutes an upper bound for the original problem, as it belongs to a feasible solution.

The time complexity of this heuristic procedure is O(MN), as MN constraints are considered when computing the C_j values.

We utilize subgradient optimization procedure by Fisher (1981) for solving the Lagrangean dual problem, which can be defined as $LD(z) = \underset{\lambda \geq 0}{\text{Max}} \{LR(z)\}$. Subgradient optimization is an iterative procedure, which, from an initial set of Lagrange multipliers, generates further multipliers in a systematic fashion. It can be viewed as a procedure that attempts to maximize the lower bound value obtained by solving the relaxed model by a suitable choice of multipliers. Below, the procedure is explained in detail.

Subgradient Optimization Procedure:

S0. Set t = 1. Initialize $\lambda^t_{jk} = (1\text{-}0.01)/M$, and $\alpha^t = 2$.

S1. Solve Subproblems 1 and 2 with Lagrange multipliers λ^t_{jk}. Let the optimal variables be X^t_{rn} and C^t_j. Set $LB^t = LR^t(z) = LR^t_1(z) + LR^t_2(z)$.

S2. Find an upper bound, UB^t, using the Lagrangean heuristic explained above, with X^t_{rn} and constraint set (3). Set $UB=\text{Min}\{UB, UB^t\}$.

S3. If $\lceil LB \rceil$ = UB, then stop. The solution is optimal. Else,

Determine step size: $$s^t = \frac{\alpha^t(UB\text{-}LB^t)}{\sum_{j,k \in JK} \left(\sum_{r=1}^{R} p_{rk} X^t_{r(j\text{-}k+1)} - C^t_j \right)^2}$$

Update multipliers: $$\lambda^{t+1}_{jk} = \text{Max}\left\{ 0, \lambda^t_{jk} - s^t \left(\sum_{r=1}^{R} p_{rk} X^t_{r(j\text{-}k+1)} - C^t_j \right) \right\}$$

If LB does not improve for a certain number of iterations (*NumIt*), then

Set $\alpha^{t+1} = \alpha^t * multiplier$.

If total number of iterations ≥ *Iteration-limit*, stop,

Else, go to S1.

The initial values for λ^t_{jk} as well as the parameters *NumIt*, *multiplier* and *Iteration-limit* are set by experimentation in this procedure.

4. Computational Experience

In this section, we describe an experiment to test the efficiencies of the developed approximation procedure. The algorithm is coded in Turbo Pascal 7.0 and the experiments are conducted on a computer with 2200 MHz Intel Celeron processor having 256 MB memory.

We generate random test problems for different problem sizes of N =15, 30, 60 and 120 jobs; R = 3, 5, 10 and 20 models, and M = 3, 5 and 8 stations. We used discrete uniform distributions to generate processing times: U[5,10]. We choose a small variance distribution to be consistent with mixed model flow line structure. Demands are randomly generated.

We perform pilot runs with relatively small sized problem instances to fine-tune parameter values of the procedure. Then, we conduct the experimentation using our randomly generated 600 instances. Since no studies exist for large problem instances, comparison of the results with any other study is not possible. Hence, we use the gap between the best-found lower bounds and our feasible solutions as a performance measure. CPLEX 8.1 is used for computing tight lower bounds for the evaluation of feasible solutions found by our algorithm. For this purpose, the original formulation is solved in CPLEX 8.1 imposing a time limit of 30 minutes and a tree size of 128 MB. Most of the large size instances remain unsolved within this time limit, in which case the best-found lower bound is used for evaluation.

Initial values of the Lagrange multipliers (λ^t_{jk}) are known not to affect the performance of the subgradient optimization procedure (Fisher, 1981). Hence, they are set to a value guaranteeing that the coefficients of C_j in Subproblem 2 are small but positive. The limit on the number of iterations for updating α value (*Numit*), the multiplier used for the update (*multiplier*) and limit on the total number of iterations (*Iteration-Limit*) are critical for the performance of the proposed algorithm. Two levels are proposed for values of these parameters and a full-factorial design is used to check the significance, and fine-tune to improve performance.

In many of the studies, the multiplier value is set as 0.5, but the value of 0.75 is also used to evaluate the effect of slow convergence on the performance of the algorithm. Two levels are used for evaluating the effect of total number of iterations, as 400 and 800. The parameter *Numit* indicates the limit on the number of iterations without improvement, and the levels for this parameter are set as 20 and 40.

Three randomly generated instances are selected from each of the 30 problem settings, and a full factorial experimentation is performed. According to the results of analysis of variance, the increase in the solution quality does not compensate for the increased runtime for any of the factors. Two-way interactions between factors are found to be insignificant, as well. So, the lower levels are selected for detailed experimentation.

Different sizes of problems are used to evaluate the performance of the proposed algorithm. The problem settings used in experimentation are summarized in Table 1. 20 random instances are generated for each setting; hence the experimentation is performed using a total of 600 instances.

Table 1. Problem Settings Used in Experimentation

Number of Positions (N)	Number of Models (R)	Number of Machines (M)
30	3, 5	3, 5, 8
60	3, 5, 10	3, 5, 8
120	3, 5, 10, 20	3, 5, 8

Three and five models are used for experimentation with 30 jobs, and the number of models increases with the total number of jobs. As the algorithm is especially intended for large problems, the number of large size instances is much higher than the small size instances in the experimental setting.

The results of the experimentation using both objective functions are summarized in Tables 2 and 3.

Table 2. Computational Results of the Experiments with Makespan Objective

M	N	R	# of Iterations Average	# of Iterations Maximum	CPU Time (sec.) Average	CPU Time (sec.) Maximum	% Deviation from best LB Average	% Deviation from best LB Maximum	# of Optimal Solutions Found
3	30	3	234	400	17	31	0.34%	2.86%	14
		5	377	400	30	35	1.64%	5.19%	2
	60	3	354	400	29	34	2.04%	9.56%	4
		5	379	400	32	35	3.58%	9.36%	2
		10	400	400	38	41	3.90%	7.45%	0
	120	3	362	400	4	7	2.58%	9.67%	3
		5	365	400	41	46	3.85%	11.16%	3
		10	400	400	56	70	3.97%	6.24%	0
		20	400	400	81	85	5.48%	8.68%	0
5	30	3	387	400	30	34	0.77%	3.91%	7
		5	400	400	34	43	3.21%	8.78%	0
	60	3	386	400	33	42	2.31%	4.33%	1
		5	400	400	36	41	4.95%	12.11%	0
		10	400	400	43	45	6.84%	12.87%	0
	120	3	400	400	12	15	3.89%	13.42%	0
		5	381	400	52	57	5.33%	11.76%	1
		10	400	400	98	100	8.13%	14.17%	0
		20	400	400	121	128	8.94%	13.14%	0
8	30	3	370	400	30	35	1.03%	5.61%	6
		5	400	400	32	33	3.01%	6.01%	1
	60	3	400	400	36	38	3.23%	10.66%	0
		5	400	400	41	46	6.55%	13.49%	0
		10	400	400	49	50	9.39%	13.15%	0
	120	3	400	400	69	72	4.24%	11.79%	1
		5	400	400	89	98	6.91%	14.87%	0
		10	400	400	145	148	10.44%	21.44%	0
		20	400	400	242	243	12.58%	18.57%	0

Table 3. Computational Results of the Experiments with Total Flow Time Objective

M	N	R	CPU Time (sec.) Average	CPU Time (sec.) Maximum	% Deviation from best LB Average	% Deviation from best LB Maximum	# of Optimal Solutions Found
3	30	3	7	9	1.29%	5.14%	4
		5	8	9	2.60%	5.61%	0
	60	3	10	11	2.46%	5.33%	1
		5	12	16	3.99%	7.25%	0
		10	17	18	5.96%	8.61%	0
	120	3	16	18	2.59%	6.73%	2
		5	22	23	5.59%	11.20%	1
		10	36	39	7.42%	10.43%	0
		20	61	101	9.04%	12.41%	0
5	30	3	7	9	1.91%	4.15%	0
		5	8	10	3.30%	6.09%	0
	60	3	11	19	2.51%	5.16%	0
		5	13	16	5.60%	11.35%	0
		10	19	20	9.40%	14.97%	0
	120	3	25	28	4.20%	13.27%	0
		5	35	36	6.35%	11.90%	0
		10	57	60	10.60%	15.32%	0
		20	101	233	13.37%	18.23%	0
8	30	3	8	10	1.87%	5.75%	1
		5	9	10	3.06%	5.23%	0
	60	3	12	16	3.04%	8.11%	0
		5	17	19	6.71%	12.10%	0
		10	27	28	11.40%	15.10%	0
	120	3	48	54	4.33%	8.62%	0
		5	68	80	7.96%	16.94%	0
		10	122	125	13.21%	22.43%	0
		20	231	240	16.85%	20.60%	0

In both tables, percentage gap between the best-found lower bound and the feasible solution found by the algorithm is shown under the column heading "% Deviation from best LB". If the optimal solutions for these instances are known, the deviations may diminish to lower levels as in the case of smaller instances. It can be observed from the tables that the number of stations and the number of jobs are the two dominant factors that affect the quality of solutions. Number of iterations are 400 for all instances of the problem with flow time objective, hence not stated explicitly. The number of optimal solutions found (out of 20 instances) is listed, as well.

The results show that our algorithm is very efficient especially for large problem instances for both objectives. Table 4 summarizes the performance of the algorithm on two problems for different number of jobs. We see that our feasible solution performs quite satisfactory. The average deviations are below 9% for all of the problem combinations and the maximum deviation is 22.43%. The heuristic is computationally very efficient, and returns solutions in less than 1.5 minute on the average for even the largest instances of the problems.

Table 4. Summary of the Results

Objective	N	CPU Time (sec.) Average	CPU Time (sec.) Maximum	% Deviation from best LB Average	% Deviation from best LB Maximum
	30	28.8	43	1.67%	8.78%
	60	37.4	50	4.75%	13.49%
Makespan	120	84.2	243	6.36%	21.44%
	30	7.8	10	2.34%	6.09%
	60	15.3	28	5.67%	15.10%
Total Flow Time	120	68.5	240	8.46%	22.43%

The algorithm performs better for makespan objective. This is mainly due to the procedure used for computing Lagrangean lower bounds. In solving Subproblem 2, each cycle time is overestimated, which results in a poor lower bound. Since the number of cycle times in flow time objective is more than makespan, the resulting lower bound is worse, which leads to worse overall performance.

5. Conclusions

In this study, we studied the mixed model flow line sequencing problem (MMS) with makespan and total flow time objectives. We presented a Lagrangean Relaxation-based approximation algorithm especially for the large size instances of the problems.

We have tested our algorithm using computational experimentation. The algorithm has an outstanding performance especially for the makespan objective. One conclusion from the experiment is that the most significant parameters affecting the problem difficulty are the number of jobs and the number of stations. The algorithm performs robustly to the changes in number of models. We also observe that our approximation procedure generates high quality solutions in very small computation times for large instances, therefore it may be of great use when one is interested in nice and quick solutions rather than a guarantee of optimality.

References

Baker, K.R. (1975) A Comparative Survey of Flow Shop Algorithms. *Operations Research*,**23**, 62-73.

Bansal, S.P. (1977) Minimizing the Sum of Completion Times of *n*-jobs over *M*-machines in a Flowshop-A Branch and Bound Approach. *AIIE Transaction*,**9**, 306-311.

Bolat, A., Savsar, M. and Al-Fawzan, M.A. (1994) Algorithms for Real-Time Scheduling of Jobs on Mixed Model Assembly Lines. *Computers and Operations Research*,**21**, 487-498.

Campbell, H.G., Dudek, R.A. and Smith, M.L. (1970) A Heuristic Algorithm for the n-Job, m-Machine Sequencing Problem. *Management Science*,**16/B**, 630-637.

Fisher M.L. (1981) The Lagrangian Relaxation Method for Solving Integer Programming Problems. *Management Science*,**27**, 1-18.

Garey, M.R., Johnson, D.S. and Sethi, R. (1976) The Complexity of Flow shop and Jobshop Scheduling. *Mathematics of Operations Research*,**1**, 117-129.

Gupta, J.N.D. (1972) Heuristic Algorithms for the Multistage Flow Shop Problem. *AIIE Transactions*,**4**, 11-18.

Ho, J.C. and Chang, Y.-L. (1991) A New Heuristic for the *n*-job, *m*-machine Flow-shop Problem. *European Journal of Operational Research*,**52**, 194-202.

Ignall, E. and Schrage, L. (1965) Application of the Branch and Bound Technique to some Flow Shop Scheduling Problem. *Operations Research*,**13**, 400-412.

Johnson, S.M. (1954) Optimal Two- and Three-Stage Production Schedules with Setup Times Included. *Naval Research Logistics Quarterly,***1**, 61-67.

Kubiak, W. and Sethi, S. (1991) A Note On Level Schedules for Mixed-Model Assembly Lines in Just-In-Time Production Systems. *Management Science*,**37**, 121-122.

Lenstra, J.K., Rinnooy Kan, A.H.G. and Brucker, P. (1977) Complexity of Machine Scheduling Problems. *Annals of Discrete Mathematics*,**1**, 343-362.

Miltenburg, J. (1989) Level Schedules for Mixed-Model Assembly Lines in Just-In-Time Production Systems. *Management Science,***35**, 192-207.

Miyazaki, S., Nishiyama, N. and Hashimoto, F. (1978) An Adjacent Pairwise Approach to the Mean Flowtime Scheduling Problem. *Journal of Operations Research Society of Japan*,**21**, 287-299.
Palmer, D.S. (1965) Sequencing Jobs Through A Multi-Stage Process in the Minimum Total Time –A Quick Method of Obtaining A Near Optimum. *Operations Research Quarterly*,**6**, 101-107.
Pinedo, M., and Chao X. (1999) *Operations Scheduling with Applications in Manufacturing and Services*. Boston: McGraw-Hill
Rajendran, C. (1993) Heuristic Algorithm for Scheduling in a Flowshop to Minimize Total Flowtime. *International Journal of Production Economics*,**29**, 65-73.
Rajendran, C. and Chaudhuri, D. (1991) An Efficient Heuristic Approach to the Scheduling of Jobs in a Flowshop. *European Journal of Operational Research*,**61**, 318-325.
Soylu, B. (2002) *A Mixed Model Line Sequencing Problem with Makespan Minimization*. M.S. Thesis, Industrial Engineering Department, Middle East Technical University, Ankara, Turkey.
Steiner, G. and Yeomans, J.S. (1993) Level Schedules for Mixed-Model, Just-In-Time Process. *Management Science,***39**, 728-735.
Szwarc, W. (1983) The Flowshop Problem with Mean Completion Time Criterion. *AIIE Transaction*,**15**, 172-176.
Taillard, E. (1990) Some Efficient Heuristic Methods for the Flow Shop Sequencing Problem. *European Journal of Operational Research*,**47**, 65-74.
Tang, L. and Liu, J. (2002) A Modified Genetic Algorithm for the Flow Shop Sequencing Problem to Minimize Mean Flow Time. *Journal of Intelligent Manufacturing*,**13**, 61-67.
Tsai, L.H. (1995) Mixed Model Sequencing to Minimize Utility Work and the Risk of Conveyor Stoppage. *Management Science*, **41**, 485-495.
Widmer, M. and Hertz, A. (1989) A New Heuristic Method for the Flow Shop Sequencing Heuristic. *European Journal of Operational Research*,**41**, 186-193.
Yano, C.A. and Rachamadugu, R. (1991) Sequencing to Minimize Work Overload in Assembly Lines with Product Options. *Management Science*,**37**, 572-586.