Optimization of a Flow Shop System of Initially Controllable Machines

Kagan Gokbayrak, *Member, IEEE*, and Omer Selvi

***Abstract*—We consider an optimization problem for deterministic flow shop systems of traditional machines with service costs penalizing small service times. A regular completion-time cost is also included so as to complete jobs as early as possible. The service times are assumed to be initially controllable, i.e., they are set at the startup time. Assuming convexity of the cost functions, we formulate a convex optimization problem after linearization of the *max* constraints. The numeric solution of this problem demands a large memory limiting the solvable system sizes. In order to relieve the memory bottleneck, some waiting characteristics of jobs served in fixed-service-time flow shop systems are exploited to result in a simpler equivalent convex optimization problem. These characteristics and the benefit of CNC machines are demonstrated in a numerical example. We also show that the simplifications result in significant improvements in solvable system sizes and solution times.**

***Index Terms*—Convex optimization, discrete event dynamic systems, manufacturing.**

I. Introduction

We consider deterministic serial manufacturing systems formed of traditional (manually-controlled) machines processing identical jobs arriving at given times. The queues of the machines are unlimited in size and operate under the nonpreemptive first-in-first-out (FIFO) discipline. As opposed to the computer numerical control (CNC) machines considered in [1], these machines are manually controlled by human operators. During mass production, a company cannot afford human interventions to modify the service times because the setup times are idle times for the machines, and these manual modifications are prone to errors. Therefore, the service times at these traditional machines are initially controllable, i.e., they are set at the startup time, and are applied to all jobs processed at these machines.

The cost function we consider consists of service costs at machines and regular completion-time costs of jobs. Motivated by the extended Taylor's tool-wear equation (in [2]), we assume that faster services increase wear and tear on the tools due to increased temperatures, and may raise the need for extra supervision, increasing service costs. The losses of the product quality due to faster services are also lumped into these service costs. Slower services, on the other hand, may delay the completion times increasing the completion-time costs. We acknowledge this trade-off, and set the objective of this study as to determine the cost-minimizing service times.

The idea of treating scheduling problems for deterministic queues as optimal control problems on discrete event dynamic systems first appeared in [3] where job release times to a single machine system were controlled to minimize the discrepancy between job completion times and desired due dates. Following this work, service time control problems for CNC manufacturing systems, where the service times can be adjusted between processes, were considered. Pepyne and Cassandras, in [4], formulated an optimal control problem for a single machine system with the objective of completing jobs as fast as possible with the least amount of control effort. In [5], they extended their results to jobs with completion due dates penalizing both earliness and tardiness. The optimal solution uniqueness for this case was shown in [6]. Exploiting the structural properties of the optimal sample path for the single machine problem, Wardi *et al.*, in [7], developed a backward-in-time solution algorithm while Cho *et al.*, in [8], developed a forward-in-time solution algorithm, which was later improved by Zhang and Cassandras in [9]. In a related work, Moon and Wardi, in [10], considered a single machine problem where the completed jobs wait in a finite size output buffer until their due dates. They presented an efficient solution algorithm for this system with blocking.

Service time control problems for two machine systems were solved by Cassandras *et al.*, in [11], using the Bezier approximation method. Gokbayrak and Cassandras, in [12], and Gokbayrak and Selvi, in [13], identified optimal sample path characteristics for these problems. Finally, in [1], Gokbayrak and Selvi considered multimachine flow shop systems with regular costs on completion times and decreasing costs on service times. It was shown that, on the optimal sample path, jobs do not wait between machines, a property which allowed for simple convex programming formulations. Under strict convexity assumptions, a forward-in-time solution algorithm was also developed.

In this techical note, we consider the system in [1] with the modification that the CNC machines are replaced by traditional non-CNC machines where the service times are set only once, and cannot be altered between processes. Even though this seems to be simple modification, since the no-waiting property no longer holds, the analysis is changed completely. We first apply the standard method of linearization on the *max* constraints to derive a convex optimization problem. Not having the no-waiting property, we can not simplify this problem, so we search for an alternative formulation. For this purpose, we present some waiting characteristics of jobs served in the flow shop systems with fixed service times, independent of the optimization problem. Then, we incorporate these characteristics in the optimization problem to derive a simpler equivalent convex optimization problem.

The rest of the techical note is organized as follows: In Section II, we apply the linearization method to formulate a convex optimization problem. Section III presents some waiting characteristics of jobs served in fixed-service-time flow shop systems regardless of the objective function. Employing these characteristics, in Section IV, a simpler equivalent convex optimization problem is derived. Section V presents a numerical example to illustrate the waiting characteristics under optimal service times and the effect of flexibility in adjusting service times between processes on the optimal cost. In this section, we also compare the solution times and the solvable system sizes for the two convex optimization problem formulations. Finally, Section VI concludes the techical note.

II. Problem Formulation

We consider a sequence of N identical jobs, denoted by $\{C_i\}_{i=1}^{N}$, arriving at an M-machine flow shop system at known times $0 \le a_1 \le a_2 \le \cdots \le a_N$. Machines process one job at a time on a FIFO nonpreemptive basis (i.e., a job in service can not be interrupted until its service completion). The buffers in front of the machines are assumed to be of infinite sizes.

We define a temporal state $x_{i,j}$ that keeps the departure time information of job C_i from machine j. The relations between the temporal states are defined by the Lindley's Equation (see in [14])

$$x_{i,j} = \max(x_{i,j-1}, x_{i-1,j}) + s_j \tag{1}$$

$$x_{i,0} = a_i, \quad x_{0,j} = -\infty \tag{2}$$

for $i = 1, \ldots, N$ and $j = 1, \ldots, M$, where the service time at machine $j \in \{1, \ldots, M\}$ is denoted by s_j. Note that, unlike the system

Manuscript received August 6, 2007; revised February 16, 2008. Current version published December 10, 2008. Recommended by Associate Editor S. Haar.

The authors are with the Department of Industrial Engineering, Bilkent University, Ankara, 06800 Turkey (e-mail: kgokbayr@bilkent.edu.tr; selvi@bilkent.edu.tr).

Color versions of one or more of the figures in this techical note are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2008.2007153

considered in [1] where the service times could differ from one job to another, the same service time s_j is applied to all jobs at machine j.

The discrete-event optimal control problem under consideration is

$$P: \min_{\substack{s_j \geq S_j \\ j=1,\ldots,M}} \left\{ J = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{3}$$

subject to (1) and (2) for $i = 1,\ldots,N$ and $j = 1,\ldots,M$. In this formulation, θ_j denotes the total service cost over all jobs at machine j, and ϕ_i denotes the completion-time cost for job C_i. The minimum service time required at machine j, a physical constraint, is denoted by S_j.

The following assumptions are necessary to make the problem somewhat more tractable while preserving the originality of the problem.

Assumption 1: $\theta_j(\cdot)$, for $j = 1,\ldots,M$, is monotonically decreasing and convex.

Assumption 2: $\phi_i(\cdot)$, for $i = 1,\ldots,N$, is monotonically increasing and convex.

Assumption 1 is motivated by the extended Taylor's tool-wear equation. Supporting this assumption, the work of Kayan and Akturk in [15], also showed that the service cost of a turning operation, one of the three principle machining operations along with milling and drilling, can be expressed as a nonlinear decreasing convex function of its service time in its operating range. The monotonicity of ϕ_i in Assumption 2 is needed for Theorems 1 and 2 that follow, while the convexity of ϕ_i is needed for obtaining a convex optimization problem. Note the trade-off due to these standing assumptions that longer services will decrease the service costs while increasing the departure times and the completion-time costs.

A standard method for solving P is to replace (1) with two linear inequalities and to employ (2) for the first job. Since, by Assumptions 1 and 2, both costs are convex, we arrive at the following convex optimization problem:

$$\bar{P}: \min_{\substack{s_j, x_{i,j} \\ i=1,\ldots,N \\ j=1,\ldots,M}} \left\{ \bar{J} = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{4}$$

subject to

$$x_{1,1} = a_1 + s_1 \tag{5}$$

$$x_{1,j} = a_1 + \sum_{k=1}^{j} s_k \tag{6}$$

$$x_{i,1} \geq a_i + s_1 \tag{7}$$

$$x_{i,1} \geq x_{i-1,1} + s_1 \tag{8}$$

$$x_{i,j} \geq x_{i,j-1} + s_j \tag{9}$$

$$x_{i,j} \geq x_{i-1,j} + s_j \tag{10}$$

$$s_1 \geq S_1 \tag{11}$$

$$s_j \geq S_j \tag{12}$$

for all $i = 2,\ldots,N$ and $j = 2,\ldots,M$. There are $(N+1)M$ variables, M equality and $2(N-1)M$ inequality constraints in this formulation excluding the M boundary value constraints on the service times.

The optimal solution for problem P can be determined by solving the convex optimization problem $\bar{P}$. A similar formulation was further simplified in [1] by employing the no-waiting property for flow shop systems where the service times can be adjusted between processes. Not having this property, we search for an alternative means of simplification. In the next section, we present some waiting characteristics of jobs served in the flow shop systems with fixed service times. Exploiting these characteristics, we derive a simpler equivalent convex optimization problem.

III. Waiting Characteristics in Fixed Service Time Flow Shop Systems

In fixed service time flow shop systems, each machine j performs some service of duration s_j. Based on these service times, we define the following:

Definition 1: Machine u is a *local bottleneck* if its service time exceeds the service times of all upstream machines, i.e., $s_u > \max_{j=0,\ldots,u-1} s_j$ where s_0 is defined to be zero.

Since the first machine is a local bottleneck, there is at least one local bottleneck in each flow shop system.

Definition 2: Machines $\{u,\ldots,v\}$ form a *flushing portion* if

1) Machine u is a local bottleneck, i.e., $s_u > \max_{j=0,\ldots,u-1} s_j$
2) There are no local bottlenecks in machines $\{u+1,\ldots,v\}$, i.e., $s_u \geq \max_{j=u+1,\ldots,v} s_j$
3) If $v < M$, then machine $(v+1)$ is a local bottleneck, i.e., $s_u < s_{v+1}$.

Each local bottleneck starts a flushing portion, and the last flushing portion is ended by machine M.

The following lemma establishes that jobs may wait at only the local bottlenecks.

Lemma 1: No-waiting is observed in a flushing portion after its local bottleneck machine.

Proof: (By induction) Let us consider a flushing portion formed of machines $\{u,\ldots,v\}$. Since the first job does not wait at any machine, we have the basis for the induction. Next, let us assume that job C_{i-1} does not wait at machines $\{u+1,\ldots,v\}$ and hence, from (1), it satisfies

$$x_{i-1,j+1} = x_{i-1,u} + \sum_{l=u+1}^{j+1} s_l \tag{13}$$

for all $j = u,\ldots,v-1$. From (1), on the other hand, we have

$$x_{i,u} \geq x_{i-1,u} + s_u. \tag{14}$$

Being in the flushing portion started by the local bottleneck machine u, machine $(u+1)$ satisfies $s_u \geq s_{u+1}$ by definition hence, from (13) and (14), we get

$$x_{i,u} \geq x_{i-1,u+1}$$

i.e., job C_i does not wait at machine $(u+1)$.

Next, in addition to (13), let us assume that job C_i does not wait at machines $\{u+1,\ldots,j\}$ where $j < v$, i.e.,

$$x_{i,k} \geq x_{i-1,k+1} \tag{15}$$

holds for all $k = u,\ldots,j-1$. From (1) and (15), we can write

$$x_{i,j} = x_{i,u} + \sum_{l=u+1}^{j} s_l \geq x_{i-1,u} + \sum_{l=u}^{j} s_l. \tag{16}$$

Since $s_u \geq s_{j+1}$ by definition, from (13) and (16), we have

$$x_{i,j} \geq x_{i-1,j+1} \tag{17}$$

indicating that job C_i does not wait at machine $(j+1)$, therefore concluding the induction proof. ■

The next lemma suggests that, given the waiting status of a job at a local bottleneck machine, we may deduce its waiting status at a downstream or an upstream local bottleneck machine.

Lemma 2: If job C_i waits for service at some local bottleneck, then it will wait for service at all downstream local bottlenecks.

Proof: We consider two consecutive local bottleneck machines u and $(v+1)$, and assume that job C_i waits at machine u, so we have $x_{i,u-1} < x_{i-1,u}$. If these two local bottleneck machines are adjacent, i.e., if $v = u$ then, from (1), we have

$$x_{i,v} = x_{i-1,u} + s_u \tag{18}$$

and

$$x_{i-1,v+1} \geq x_{i-1,u} + s_{v+1}. \tag{19}$$

Since $s_u < s_{v+1}$ by definition, from (18) and (19), we get $x_{i-1,v+1} > x_{i,v}$, i.e., job C_i waits at machine $(v+1)$.

If, on the other hand, these two local bottlenecks are not adjacent, i.e., $v > u$ then, from (1) and Lemma 1, we have

$$x_{i,v} = x_{i,u} + \sum_{j=u+1}^{v} s_j = x_{i-1,u} + s_u + \sum_{j=u+1}^{v} s_j \tag{20}$$

and

$$x_{i-1,v+1} \geq x_{i-1,v} + s_{v+1} \geq x_{i-1,u} + \sum_{j=u+1}^{v} s_j + s_{v+1}. \tag{21}$$

Since $s_u < s_{v+1}$ by definition, from (20) and (21), we get $x_{i-1,v+1} > x_{i,v}$, i.e., job C_i waits at machine $(v+1)$.

The result extends iteratively to all downstream local bottleneck machines concluding the proof. ■

Next, we define the most downstream local bottleneck of the system as the global bottleneck.

Definition 3: A local bottleneck machine u is also a *global bottleneck* if its service time satisfies $s_u = \max_{j=1,\ldots,M} s_j$.

There can be no local bottleneck machines downstream to a global bottleneck, therefore, from Lemma 1, no waiting is observed after the global bottleneck machine. Hence, the completion times can be determined as presented in the next lemma.

Lemma 3: The completion time of job C_i is given by

$$x_{i,M} = \max\left(x_{i-1,M} + s_u, a_i + \sum_{j=1}^{M} s_j\right) \tag{22}$$

where $x_{0,M} = -\infty$ and $s_u = \max_{j=1,\ldots,M} s_j$ is the service time of the global bottleneck machine.

Proof: From (1), the departure time of job C_i from the global bottleneck machine is given as

$$x_{i,u} = \max(x_{i-1,u}, x_{i,u-1}) + s_u. \tag{23}$$

If job C_i does not wait at the global bottleneck machine, i.e., if $x_{i,u-1} \geq x_{i-1,u}$, from Lemma 2, it also does not wait at any upstream machines, therefore, we have

$$x_{i,u-1} = a_i + \sum_{j=1}^{u-1} s_j \geq x_{i-1,u}. \tag{24}$$

Hence, from (23) and (24), we get

$$x_{i,u} = \max\left(x_{i-1,u} + s_u, a_i + \sum_{j=1}^{u} s_j\right). \tag{25}$$

Since no waiting is observed after the global bottleneck machine u, from (25), we can write the job's completion time as

$$\begin{aligned} x_{i,M} &= \begin{cases} x_{i,u} & \text{if } u = M \\ x_{i,u} + \sum\limits_{j=u+1}^{M} s_j & \text{if } u < M \end{cases} \\ &= \max\left(x_{i-1,M} + s_u, a_i + \sum_{j=1}^{M} s_j\right) \end{aligned}$$

■

In the next section, we employ the characteristics from this section to derive a simpler convex optimization problem formulation.

IV. Simpler Convex Optimization Problem

The result of Lemma 3 allows us to replace (1) and (2) in P by (22) resulting with the formulation

$$P: \min_{\substack{s_j \\ j=1,\ldots,M}} \left\{ J = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\}$$

subject to

$$\begin{aligned} x_{1,M} &= a_1 + \sum_{k=1}^{M} s_k && (26) \\ x_{i,M} &= \max\left(x_{i-1,M} + \max_{k=1,\ldots,M} s_k, a_i + \sum_{k=1}^{M} s_k\right) && (27) \\ s_j &\geq S_j && (28) \end{aligned}$$

for $i = 2, \ldots, N$ and $j = 1, \ldots, M$.

By linearizing the *max* functions in (27), we get the convex optimization problem Q given as

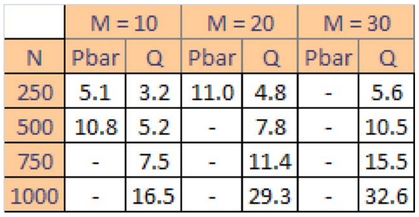
$$Q: \min_{\substack{s_j, x_{i,M} \\ i=1,\ldots,N \\ j=1,\ldots,M}} \left\{ J_Q = \sum_{j=1}^{M} \theta_j(s_j) + \sum_{i=1}^{N} \phi_i(x_{i,M}) \right\} \tag{29}$$

subject to

$$\begin{aligned} x_{1,M} &= a_1 + \sum_{k=1}^{M} s_k && (30) \\ x_{i,M} &\geq a_i + \sum_{k=1}^{M} s_k && (31) \\ x_{i,M} &\geq x_{i-1,M} + s_j && (32) \\ s_j &\geq S_j && (33) \end{aligned}$$

for all $i = 2, \ldots, N$ and $j = 1, \ldots, M$. In this formulation there are $N + M$ variables, one equality and $(N-1)(M+1)$ inequality constraints excluding the M boundary value constraints on the service times. Therefore, compared to the $\bar{P}$ problem, improvements in solution times and memory requirements are expected.

TABLE I
COMPUTATION TIMES (IN SECONDS)



	M = 10		M = 20		M = 30	
N	Pbar	Q	Pbar	Q	Pbar	Q
250	5.1	3.2	11.0	4.8	-	5.6
500	10.8	5.2	-	7.8	-	10.5
750	-	7.5	-	11.4	-	15.5
1000	-	16.5	-	29.3	-	32.6

V. NUMERICAL EXAMPLES

We first consider the example in [1], where ten jobs are to be served in a flow shop of four machines. The total service cost $\theta_j(s_j)$ at machine j is given as

$$\theta_j(s_j) = \frac{\beta_j}{s_j} \tag{34}$$

for some constant β_j, while the completion-time cost for job C_i is given as

$$\phi_i(x_{i,M}) = 10(x_{i,M} - a_i)^2. \tag{35}$$

Note that these costs satisfy Assumptions 1 and 2.

The arrival times of the jobs are given as $a = [0.0, 2.3, 2.4, 4.9, 5.0, 5.5, 9.0, 9.5, 11.0, 13.0]$. Since ten jobs are to be processed, the β parameter vector in [1] is adjusted to become $\beta = [100, 50, 200, 100]$. The service times on these four machines are bounded below by $S = [0.20, 0.20, 0.30, 0.35]$.

The optimal service times are found to be $s^* = [0.4942, 0.3495, 0.5593, 0.4942]$. Hence, for the optimal solution, the first machine turns out to be a local bottleneck, while the third machine is the global bottleneck. From Lemma 1, we expect to see no waiting in front of the second and the fourth machines.

The optimal cost for this example is given as 1329.01 compared to 1290.15 given in [1]. The cost difference is due to not having the flexibility to adjust the service times between processes. It may be viewed as the benefit to be gained by replacing all these non-CNC machines by CNC machines.

In order to demonstrate the benefits due to simplifications, we further present the computation times in Table I under different M and N settings for given a, β, and S parameter vectors. Note that a dash sign, as in the entry for 10 machines and 750 jobs, represents an "out of memory" crash. The times are reported from a 1.6 GHz Intel Dual Core processor and 1 GB of RAM running *cvx* (see [16]), a modeling system for convex programming developed in Stanford University, employing the second order cone and semidefinite programming solver SeDuMi 1.1R3 implemented in Matlab.

The results in Table I suggest that it is definitely faster to solve for Q. One may argue, however, that since these calculations are to be performed offline before the manufacturing operation starts, the improvement in the calculation times are not that important. After all, thanks to today's fast CPUs, both problems are solved within minutes. (It takes about 65.2 s to solve for Q when $M = 30$ and $N = 1500$.) In fact, the results suggest that the bottleneck in these calculations is not the CPU speed but the memory size. (Numerical solution of $\bar{P}$ is feasible only for small systems.) The important contribution of our work is that, by the proposed simplifications, we relieve this bottleneck and allow the optimization of larger systems.

VI. CONCLUSION

This technical note considered deterministic flow shop systems, where the service times of the machines were set only once and could not be altered between processes. A standard solution method based on the linearization of the *max* constraints was demonstrated to yield a convex optimization problem with high memory requirements. Unfortunately, the inflexibility constraint on the service times removed the no-waiting property preventing the simplifications in [1].

In order to derive an equivalent convex optimization problem with lower memory requirements, a set of waiting characteristics of jobs served in fixed-service-time flow shop systems were exploited. The resulting simplified convex optimization problem had significantly lower memory requirements and somewhat lower solution times.

Replacement of non-CNC machines with CNC machines clearly improves system performance. We conjecture that the CNC machine singlehandedly brings the no-waiting property to its downstream lowering the costs, and this is the subject of future research.

REFERENCES

[1] K. Gokbayrak and O. Selvi, "Constrained optimal hybrid control of a flow shop system," *IEEE Trans. Automat. Control*, vol. 52, no. 12, pp. 2270–2281, Dec. 2007.
[2] S. Kalpakjian and S. R. Schmid*, Manufacturing Engineering and Technology*. Englewood Cliffs, NJ: Prentice Hall, 2006.
[3] M. Gazarik and Y. Wardi, "Optimal release times in a single server: An optimal control perspective," *IEEE Trans. Automat. Control*, vol. 43, no. 7, pp. 998–1002, Jul. 1998.
[4] D. L. Pepyne and C. G. Cassandras, "Modeling, analysis, and optimal control of a class of hybrid systems," *J. Discrete Event Dyn. Syst.: Theory Appl.*, vol. 8, no. 2, pp. 175–201, 1998.
[5] D. L. Pepyne and C. G. Cassandras, "Optimal control of hybrid systems in manufacturing," *Proc. IEEE*, vol. 88, no. 7, pp. 1108–1123, Jul. 2000.
[6] C. G. Cassandras, D. L. Pepyne, and Y. Wardi, "Optimal control of a class of hybrid systems," *IEEE Trans. Automat. Control*, vol. 46, no. 3, pp. 398–415, Mar. 2001.
[7] Y. Wardi, C. G. Cassandras, and D. L. Pepyne, "A backward algorithm for computing optimal controls for single-stage hybrid manufacturing systems," *Int. J. Prod. Res.*, vol. 39–2, pp. 369–393, 2001.
[8] Y. C. Cho, C. G. Cassandras, and D. L. Pepyne, "Forward decomposition algorithms for optimal control of a class of hybrid systems," *Int. J. Robust Nonlinear Control*, vol. 11, pp. 497–513, 2001.
[9] P. Zhang and C. G. Cassandras, "An improved forward algorithm for optimal control of a class of hybrid systems," *IEEE Trans. Automat. Control*, vol. 47, no. 10, pp. 1735–1739, Oct. 2002.
[10] J. Moon and Y. Wardi, "Optimal control of processing times in single-stage discrete event dynamic systems with blocking," *IEEE Trans. Automat. Control*, vol. 50, no. 6, pp. 880–884, Jun. 2005.
[11] C. G. Cassandras, Q. Liu, D. L. Pepyne, and K. Gokbayrak, "Optimal control of a two-stage hybrid manufacturing system model," in *Proc. 38th IEEE Conf. Decision Control*, 1999, pp. 450–455.
[12] K. Gokbayrak and C. G. Cassandras, "Constrained optimal control for multistage hybrid manufacturing system models," in *Proc. 8th IEEE Med. Conf. New Directions Control Automat.*, 2000, pp. 1–6.
[13] K. Gokbayrak and O. Selvi, "Optimal hybrid control of a two-stage manufacturing system," in *Proc. ACC*, 2006, pp. 3364–3369.
[14] C. G. Cassandras and S. Lafortune*, Introduction to Discrete Event Systems*. Norwell, MA: Kluwer, 1999.
[15] R. K. Kayan and M. S. Akturk, "A new bounding mechanism for the CNC machine scheduling problems with controllable processing times," *Eur. J. Oper. Res.*, vol. 167–3, pp. 624–643, 2005.
[16] M. Grant and S. Boyd, CVX: Matlab Software for Disciplined Convex Programming Tech. Rep., 2007 [Online]. Available: http://stanford.edu/ boyd/cvx