SCHOOL OF OPERATIONS RESEARCH
AND INDUSTRIAL ENGINEERING
COLLEGE OF ENGINEERING
CORNELL UNIVERSITY
ITHACA, NY 14853-3801

TECHNICAL REPORT NO. 1035

October 1992
Revised September 1993
Revised July 1994
Revised February 1995

**PRODUCTION CONTROL OF
CYCLIC SCHEDULES WITH
DEMAND AND PROCESS
VARIABILITY**

by

R.A.Bowman¹ and J.A. Muckstadt²

¹Union College, Graduate Management Institute, Bailey Hall, Schenectady, NY 12308,
518-388-6297, FAX: 518-388-6686.

²Cornell University, School of Operations Research and Industrial Engineering, 206 E&TC
Building, Ithaca, NY 14853-3801, 607-255-9123, FAX: 607-255-9129.

Key Words: Cyclic Scheduling, Production Control, Markov Chain.

Abstract

Cyclic scheduling has been primarily studied under deterministic assumptions. In practice, stochastic variability exists and must be taken into account. In this paper, the descriptive Markov chain model of Bowman and Muckstadt (1993) is extended to cover demand variability. A production control algorithm is developed using cyclic time and task criticality estimates from the model. Application of the algorithm to a case study shows that material release and anticipatory inventory buildup decisions can be effective in reducing inventory holding and overtime costs when significant demand variability is present.

1 Introduction

A cyclic schedule is a sequence of tasks that are executed repeatedly so that each task is performed exactly once during each cycle. In a multi-machine, multi-product setting, precedence constraints exist between successive tasks on the same lot of a product as well as those defined by the prescribed production sequence on each machine. The use of these schedules has become increasingly prevalent in recent years in various manufacturing environments because they are significantly simpler to describe, understand, and implement than many other scheduling schemes. Communication with the shop floor is improved since the sequence is fixed. There is no need for dispatching decisions. Cyclic scheduling is consistent with the just-in-time or minimum inventory philosophy in two important respects. First, the cyclic sequence aims to globally minimize inventory by having production ready just when it is needed. Second, efforts to reduce setup times are more effective since they can be directed at the specific changeovers that occur in the sequence. Also, the task of coordinating other activities such as raw material delivery, preventive maintenance, and work force schedules becomes simpler. Whybark (1984) provides an excellent early description of a cyclic type of production control and the benefits that were realized.

In this paper we describe how to control production using cyclic schedules when both demand and processing times vary considerably. The motivation for this topic came from one of the authors' examinations of the feasibility of implementing a cyclic schedule within a focused factory that fabricates and assembles machined parts. In this environment, five products accounted for nearly all the demand, and a small fraction of the company's customers purchased most of the production of these five products. Because the demand for these products was high and occurred every week, this environment seemed to be an attractive

setting for a cyclic schedule to be implemented. Although demand was repetitive for these products, significant variability existed that resulted in weeks in which there was overtime and other weeks in which capacity was underutilized.

Demand variability in cyclic scheduling has not been discussed extensively in the literature. In fact, the vast majority of papers dealing with cyclic scheduling make assumptions that both task times and the demand process are known. This is true of the papers describing algorithms for establishing production intervals that balance setup and holding costs and protect plant capacity (see Roundy (1985, 1986), Maxwell and Muckstadt (1985), Jackson, Maxwell, and Muckstadt (1988), and Federgruen and Zheng (1993)). It is also true of the approaches that have been described for actually establishing a cyclic sequence (see Hitz (1980), Graves et. al. (1983), Wittrock (1985), McCormick (1988)). We note that even under deterministic assumptions, the development of a cyclic schedule is a hard combinatorial optimization problem. Consequently, most approaches are heuristic in nature and have been tailored to specific applications. Our concern in this paper is with controlling production when cyclic schedules are used in the environment we have described. We assume throughout this paper that the number of batches per cycle and the production sequence have already been set.

Because most of the research on cyclic scheduling has been based on this deterministic view, there has been little discussion of production control. Stochastic variability, however, can cause the implementation of a cyclic schedule to be quite different from the plan so that production control becomes an important topic. In particular, job release and production quantity decisions should effectively consider real-time schedule status information as well as the major sources of variability.

Bowman and Muckstadt (1993) model a cyclic schedule in a stochastic environment as a Markov chain. They use this model to derive information that is useful to evaluate the performance of a cyclic schedule when task time variability is present. Their model assumes

that the cyclic sequence on each machine will always be maintained and that no machine will remain idle when the precedence constraints permit the next task in the sequence to begin.

In this paper, we show how this model can be used for prescription as well as description. In particular, we develop an approach for production control of a cyclic schedule that utilizes the descriptive information from the model. We maintain the assumption that the sequence will not be violated; but, we allow a machine to be idle for the purpose of delaying the introduction of material onto the shop floor. We also allow lot sizes to be adjusted to anticipate changes in demand as well as to protect against demand and process variability.

In summary, the objectives of this paper are to describe a stochastic cyclic scheduling model and illustrate its use on an example based on a real industrial environment, to extend existing models to incorporate variation in demand, and to provide an algorithm for control of production in a repetitive manufacturing environment.

To carry out these objectives the paper is organized as follows.

In Section 2, after reviewing the earlier Bowman and Muckstadt (1993) model and discussing its advantages we extend it to cover the case of variable demand. We show that the information available from this extended model, which we obtain by simulating the model, has relevance to production control.

A case study is presented in Section 3, which is based on data obtained from the metal parts manufacturing environment described previously. The example captures the essence of the actual problem's complexity in a manner that is feasible to describe in full detail in this paper.

In Section 4, we present an algorithm for controlling production for the example case described in Section 3. This algorithm uses the measures obtained from the model described in Section 2. Although this model provides information that can be useful for production control of virtually any cyclic schedule in a stochastic environment, the best approach for

utilizing it should be tailored to the specific application. The case study is an instructive example; however, we also discuss how our approach can be adapted to different applications.

A computational experiment is described in Section 5, and the performance of the cyclic schedule in the case study is analyzed. Conclusions are drawn and recommendations made in Section 6.

2 A Markov Chain Model

In our earlier work (Bowman and Muckstadt (1993)), we showed how to model a cyclic schedule as a continuous time Markov chain. This model allowed us to rigorously define steady state performance measures using ergodic Markov chain theory. Algorithmic approaches based on these definitions were developed and used to calculate the corresponding measures. We will briefly review this model and show how to extend it to the case of variable demand. We note, however, that the inclusion of stochastic demand for any realistic problem would enlarge the state space to the extent that the use of algorithms to make exact computations is impractical. Instead, we will obtain estimates of the measures we need by simulating the Markov chain model described in this section. A detailed description of how to simulate the model is included as an appendix. There are advantages and disadvantages to simulating the Markov chain model (which we summarize at the end of this section). Alternatively, one could obtain the estimates necessary for the production control scheme described in this paper by conducting a more traditional simulation with generally distributed task times. We briefly describe how to do this in the appendix as well.

Before reviewing the model, let us review the key measures that we will obtain from the model and, at an intuitive level, describe how these measures might be useful in a production control algorithm. A specific algorithm will be presented in detail in Section 4.

A cyclic production plan can be viewed as a series of production cycles with the number of cycles corresponding to the planning horizon. For each cycle i , we will estimate the expected value ($E(T_i)$) and variance ($Var(T_i)$) of the cycle time. These cycle times will vary from cycle to cycle due to the variable demand. For each production task j in each cycle i , we will also estimate the cyclic criticality of the task ($CC(j, i)$), which is defined as the long run fraction of times task j is on the critical production path. Letting θ_j be a parameter of the probability distribution for the time to complete task j , Bowman and Muckstadt (1993) proved that $\frac{\partial E(T_i)}{\partial \theta_j} = CC(j, i)$ if θ_j is a location parameter; furthermore, they show that the two are approximately equal if θ_j is a scale parameter and the number of tasks in a cycle is large. This result is at the heart of our production control scheme because it allows us to assess the effects of changes in task times on $E(T_i)$ for various cycles. It is easily extended to allow us to assess changes in event times for other than cycle completion times as well. We shall subsequently refer to this result as the cycle time/criticality derivative result.

We will be considering two basic production control strategies. First, we could delay material release for a product in a cycle. Delaying material release obviously delays the introduction of the specific product but also delays the introduction of many other products. These delays are equivalent to increased task times. All these delays save holding costs proportional to the lengths of the delays, which can be estimated using the derivative result. This also delays cycle completion time, which can then cause overtime to be used. The derivative result is again used to estimate the effect on $E(T)$, and this effect can be used along with the $Var(T)$ estimate to approximate the effect on overtime.

Second, we could produce more product than we need within a cycle. This additional production will be used to meet demand in a later cycle. This would increase task times in the earlier cycle and decrease them in later cycles. The derivative result can be used to estimate the effects on both cycles. These effects will allow us to assess the effects on overtime and carrying inventory.

Intuitively, we would like to delay material release in cycles with low $E(T)$ and $Var(T)$ or else shift production into these cycles from later cycles with high $E(T)$ and $Var(T)$. Furthermore, when shifting production among cycles, we would like to shift production of a product that has a high criticality in the later cycle and a low criticality in the earlier cycle. This will maximize the expected amount of overtime saved. The purpose of the model is to provide us with the measures we need to evaluate these tradeoffs more precisely so that we can make good production control decisions.

To facilitate an understanding of both cyclic schedules and our specific model, we now review the Bowman and Muckstadt (1993) model and extend it to the case of stochastic demand. The state of the system in the Markov chain model reflects which task has been most recently completed on each machine. A transition from one state to another occurs when a task is completed. In its simplest form, the Markovian property implies that all the task times are exponentially distributed. Recall, however, that by mixing exponential variates we can approximate any distribution. Specifically, by summing independent variates we can accurately approximate distributions having coefficients of variation that are less than 1. This fact allows us to model the data for our case study, since the actual processing times at key operations were well approximated by an Erlang type distribution. An efficient approach for constructing this approximation is given by Bowman (1990).

This Markov chain model allows us to describe a cyclic schedule through its infinitesimal generator matrix, Q , where the ij th element of Q gives the rate of transitions from state i to state j , when the system is currently in state i .

An exact description of the elements of this infinitesimal generator matrix is notationally complex. But the underlying concept is easily demonstrated by considering a simple cyclic schedule represented in Figure 1. For this example, the arrow between task 1 and task 4 represents a material flow precedence constraint. The arrow from task 4 back to task 1 represents a managerial precedence constraint that does not allow material for the product

represented by tasks 1 and 4 to be introduced until task 4 is finished on the product in the previous cycle. Precedence arrows maintaining the sequence on each machine are implicit.

The possible states and transitions for the precedence constraints are identified in Figure 2. An arc labeled t^i going from node s^j to node s^k indicates that it is feasible for task i to be completed. This task completion would cause a transition from state j to state k to occur. Note that since cyclic schedules involve repetitive production, each execution of an operation is mapped to some cycle number. (This allows us to distinguish two successive executions of the same operation.) As shown in the figure this aspect is incorporated in the description of one state, by including the relative cycle of each machine as well as the most recently completed task on that machine. More precisely, a state at time t is defined to be $\begin{pmatrix} A(t) \\ R(t) \end{pmatrix}$, where $A(t)$ and $R(t)$ are vectors representing, respectively, the number of the task completed most recently on each machine, and the number of cycles ahead (+) or behind (−) the task completed on each machine is relative to the task completed most recently on machine 1. For example, state 3 $\begin{pmatrix} (1,4) \\ (0,-1) \end{pmatrix}$ corresponds to task 4 being completed most recently one cycle behind task 1, whereas state 6 $\begin{pmatrix} (1,4) \\ (0,0) \end{pmatrix}$ corresponds to tasks 1 and 4 being most recently completed in the same cycle. For this example, assume that all task times are exponentially distributed with unit mean so that all feasible transitions are at unit rate. This implies that the infinitesimal generator matrix for our example is given by

$$Q = \begin{bmatrix} -2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & -2 \end{bmatrix}.$$

For example, state 1 corresponds to tasks 2 and 4 (in the same cycle) being the most recently completed tasks. The completion of task 1 is feasible and would cause a transition to state 3, which reflects the fact that the task 4 completion on machine 2 is one cycle behind

Figure 1: Cyclic Schedule Example

Figure 2: Markov Chain Transition for Cyclic Scheduling Example

the completion of task 1 on machine 1. The completion of task 3 is also possible. This would cause a transition to state 4, which reflects the fact that the task 3 completion is one cycle ahead of the task 2 completion. These transitions are represented by $q_{13} = q_{14} = 1$ and $q_{11} = -2$ (total rate out of state 1 equals 2).

The modifications to this model needed to represent demand variability are conceptually simple. If a product has more than one task associated with it (e.g. the product represented by tasks 1 and 4), the state space is enlarged to include the demand for the product. In addition, transitions into states that allow a task to be started that represents the first task on such a product must be factored by the probability function for the demand of that product. Note that if a product has only one task associated with it (which would be rare in practice), demand variability can be handled strictly by adjusting the mean and variance of the task time. The necessary changes can best be demonstrated through an example. Suppose that each occurrence of demand for the product associated with tasks 1 and 4 is for either 1 or 2 units, each having probability .5. Further, suppose the times for these two tasks are exponentially distributed with mean equal to either 1 or 2, depending on the demand. Each state in the original Markov chain must be replaced by two states, one for each level of demand, as shown in Table 1. The corresponding infinitesimal generator matrix is given by

$$Q = \begin{bmatrix} -2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ .5 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ .5 & 1 & 0 & 0 & 0 & 0 & 0 & -2 & .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.5 & 0 & .5 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .5 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 0 & .5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.5 & 1 & .5 \\ .25 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .25 & 0 & 0 & 0 & 0 & 0 & -1.5 & 0 \\ .5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .5 & 1 & 0 & 0 & 0 & 0 & 0 & -2 \end{bmatrix}.$$

Table 1: States for Cyclic Schedule Example with Variable Demand

State	$A(t)$	$R(t)$	Demand
1	$\langle 2,4 \rangle$	$\langle 0,0 \rangle$	1
2	$\langle 1,3 \rangle$	$\langle 0,1 \rangle$	1
3	$\langle 1,4 \rangle$	$\langle 0,-1 \rangle$	1
4	$\langle 2,3 \rangle$	$\langle 0,1 \rangle$	1
5	$\langle 2,4 \rangle$	$\langle 0,-1 \rangle$	1
6	$\langle 1,3 \rangle$	$\langle 0,0 \rangle$	1
7	$\langle 2,3 \rangle$	$\langle 0,0 \rangle$	1
8	$\langle 1,4 \rangle$	$\langle 0,0 \rangle$	1
9	$\langle 2,4 \rangle$	$\langle 0,0 \rangle$	2
10	$\langle 1,3 \rangle$	$\langle 0,1 \rangle$	2
11	$\langle 1,4 \rangle$	$\langle 0,-1 \rangle$	2
12	$\langle 2,3 \rangle$	$\langle 0,1 \rangle$	2
13	$\langle 2,4 \rangle$	$\langle 0,-1 \rangle$	2
14	$\langle 1,3 \rangle$	$\langle 0,0 \rangle$	2
15	$\langle 2,3 \rangle$	$\langle 0,0 \rangle$	2
16	$\langle 1,4 \rangle$	$\langle 0,0 \rangle$	2

The changes necessitated by the demand variability can be understood by focusing on states 1 and 9. Note that these states are the same except state 1 reflects a demand of 1 for the product associated with tasks 1 and 4, whereas state 9 reflects a demand of 2 for this product. The transitions from state 1 are to states 3 and 4, both at unit rate. The transitions from state 9 are to states 11 and 12. These states are the same as states 3 and 4 except that they reflect a product demand of 2. The transition from state 9 to state 11 occurs at rate .5, reflecting the increased time for task 1 to meet the higher demand. Consider also state 7. This state reflects tasks 2 and 3 being most recently completed so that the only task that can be completed is task 4 (at unit rate). The transitions, however, must now be split into states 1 and 9. This split is based on the probability distribution for the demand of the product represented by tasks 1 and 4, resulting in a rate of .5 into each of states 1 and 9.

The previous example shows that the changes to the Markovian model to handle the case of variable demand are conceptually easy, but result in a rapidly expanding state space. This makes exact analysis impractical in most realistic environments. However, we can simulate

the Markov chain by generating the sequence of state-to-state transitions (see the appendix for a description of how to conduct the simulation and gather the desired measures). Of course, one could simulate the cyclic schedule with general task time distributions as well. The key differences in these approaches are:

1. Simulation of the Markov chain model can be verified via comparison with exact calculations for small test cases.
2. The possible states in the Markov chain model divide naturally into levels (the number of levels is equal to the number of tasks, see Bowman and Muckstadt (1993)) so that cycle time can be rigorously defined as the time between consecutive visits to the same level. In a general simulation, the variance of the time between successive completions of a task depends on the task. In the Markov chain, the variance is consistent regardless of the level chosen.
3. Estimates of the expected value and variance of the cycle time from the Markovian simulation have a lower variance per replication than from general distribution simulations because each estimate is exactly conditioned upon the sequence of task completions in the cycle.
4. The Markov chain simulation does not directly generate random variates. The uniform $(0, 1)$ random numbers are used to generate the sequence of task completions by selecting the next task to complete from each state (a transition) according to their respective probabilities of completing (transition probabilities).
5. Although first and second moments of task times can be easily matched, the Markov chain model does not easily allow for a general specification of task time distributions.
6. As task times become less variable, the number of sub-tasks (representing each task) increases so that each cycle replication takes longer (see number 4). This will eventu-

ally outweigh the variance reduction in the estimates previously described. We note, however, that as variability is reduced, it is more likely that deterministic models will be more appropriate to use.

3 A Case Study

To illustrate our method of analysis, we will examine a case that closely resembles a manufacturing firm that we studied in great depth. The company segmented its product line into families according to product type and component geometry. Each family is produced in a so-called focused factory. Within each focused factory, primary machining operations are performed followed by a sequence of assembly operations. Primary machining is done on NC single spindle turning centers, and additional secondary machining operations are performed on other single spindle lathes. Other operations performed are all part of the assembly and test process steps. As mentioned earlier, five of the products produced account for over 90% of the production hours and units produced in this focused factory. Hence we have limited our example to these products and considered only operations performed on the most heavily loaded machines. Data have been scaled; but, we believe the essence of the problem has been maintained.

Each week the five key products are produced in either one or two production lots. Furthermore, the operations are performed on one of five heavily loaded machines. Table 2 shows how the cyclic schedule will be carried out. Each entry in the table (i - j) gives the machine number, i , and then the position in the production sequence on that machine that corresponds to the operation performed on each product. For each product, the minimum and average weekly demand are shown, and the squared coefficient of variation (SCV) of the demand in excess of the minimum is given. For example, the first operation of the first batch of product 1 is performed on machine 1 (it is also the first task in the cycle carried out on

Table 2: Case Study Problem Summary

	Product 1		Product 2		Product 3		Product 4	Product 5
Sequence Data	Lot 1	Lot 2	Lot 1	Lot 2	Lot 1	Lot 2	Lot 1	Lot 1
Operation 1	1-1	1-5	1-2	1-6	1-3	1-7	1-4	1-8
Operation 2	2-1	2-4	4-1	4-3	2-2	2-5	2-3	2-6
Operation 3	3-1	3-3	5-2	5-6	4-2	4-4	3-2	3-4
Operation 4	5-1	5-5	1-10	1-14	5-3	5-7	5-4	5-8
Operation 5	1-9	1-13	4-5	4-7	1-11	1-15	1-12	1-16
Operation 6	2-7	2-10	5-10	5-14	2-8	2-11	2-9	2-12
Operation 7	3-5	3-7	-	-	4-6	4-8	3-6	3-8
Operation 8	5-9	5-13	-	-	5-11	5-15	5-12	5-16
Weekly Demand Data								
Minimum	1300		300		600		300	100
Average	3100		1800		1600		1200	400
SCV: Demand In Excess of Minimum	$\frac{1}{6}$		$\frac{1}{3}$		$\frac{1}{3}$		$\frac{1}{2}$	$\frac{1}{2}$

that machine), then the second operation of this batch is performed on machine 2 (it is the first task in the cycle on that machine), then to machine 3, etc. The second batch of product 1 in the cycle must go first to machine 1 (it is the 5th task in the cycle on that machine), then to machine 2 (it is the 4th task in the cycle on that machine), then to machine 3, etc. There is a minimum weekly demand of 1300 units of part 1. The average weekly demand for part 1 is 3100 units. The demand in excess of the minimum averages 1800 with a SCV of $\frac{1}{6}$. These data are based on the most recently completed 52 weeks. Note that demand and holding costs have resulted in products 1, 2, and 3 being processed in two batches per cycle (week), whereas products 4 and 5 are processed in one batch per cycle. The number of batches per week was computed using the ideas found in Jackson, Maxwell, and Muckstadt (1988). The production sequence was basically set to avoid large setup times. The resulting setup times only account for a few percent of capacity and consequently are ignored in our analysis. The run time per part depends primarily on the machine used and is roughly the same for each visit of each part to the machine. The times are .0021, .0017, .0026, .0026, and .0017 hours per part, respectively, for the five machines.

Customers place orders any time during a day. The due dates for these orders are normally within 6 weeks following the time the order is placed. Once a cycle (week) is begun, no delivery date will be promised within that cycle. This process results in a known demand for the current cycle with the demand becoming more and more variable the farther into the future one looks. The major problem has been that demand for capacity varies considerably from week to week with some weeks having significant idle time while other weeks are overly loaded. Thus a just-in-time approach to production control leads to considerable amounts of overtime being required to meet delivery due dates. We found that the weekly demand process for each product can be accurately approximated as an Erlang random variable whose exponential components become known over time. For example, the weekly demand for product 1 has an SCV of $1/6$. The demand 6 weeks out would be modeled as 1300 plus a known exponential variate plus 5 exponential random variables (each with mean = $1800/6$). The demand 5 weeks out would be 1300 plus 2 known variates plus 4 random variables, etc. until the demand in the current week is 1300 plus 6 known variates.

If a standard measure of capacity was available (bottleneck machine hours), each unit of demand required a known amount of capacity, and demand was deterministic, the transportation method of production planning described by Bowman (1956) would be directly applicable. Capacity in a cyclic schedule, however, is a function of the critical path. This does not, in general, coincide with a single bottleneck machine. Furthermore, variability in demand and in processing times complicates not only the measure of capacity, but also the wise use of this capability. By explicitly considering this variability we can improve the system's performance significantly.

4 A Stochastic Production Control Algorithm

The production control algorithm described in this section is a combination of material release control and anticipatory inventory build-up. Although these two approaches may seem very different (in fact, they are beneficial in vastly different circumstances), we will show how they can be viewed in a unified fashion. First, however, we will examine the effects of each approach separately through the use of measures defined by the Markov chain model described in Section 2. We will examine the effects on the relevant costs under several simplifying assumptions. First, inventory holding costs for each product are charged proportional to the time between material release and the end of the cycle for which the demand for that product occurs. Holding costs are charged this way because in our example the primary value added step occurs in the first operation. For other situations, the calculations will need to be modified. Also, overtime will be used whenever demand required during a week cannot be completed on regular time during that week's cycle. We assume that the regular time capacity (and its cost) are fixed so that the basic tradeoff is between inventory holding and overtime costs. Other assumptions may be more appropriate for other applications. If different assumptions were made, the details of the cost analysis would change; but, we would require essentially the same information from the model.

Each time the precedence constraints permit a new task to begin that represents the first operation on a product, it is possible to delay the release of material to the floor. Observe that delaying the release of a product by Δt time units is equivalent to adding Δt to the time required to complete the first task for that product. The amount added to the task time can be considered as a simple case of a location parameter of the task time distribution, which means we can directly apply the derivative result. Consequently, denoting the release time of task i in cycle c by $r_i(c)$, the inventory holding cost by I , and the overtime cost by O , we can approximate the effects of a delay to the release time $r_i(c)$ by:

$$\frac{\partial I}{\partial r_i(c)} \approx - \sum_{j: j \text{ is the first task on a product}} C(i, j, c) h_{p(j)} D_{p(j)}(c), \text{ and} \quad (1)$$

$$\frac{\partial O}{\partial r_i(c)} \approx CC(i, c) P_0(c) V, \text{ where} \quad (2)$$

- $C(i, j, c)$ = the steady state probability task i is critical to the start of task j in cycle c ,
- $p(j)$ = the product corresponding to task j ,
- h_k = the holding cost per unit of product k per hour,
- $D_k(c)$ = the demand in cycle c for product k ,
- $CC(i, c)$ = the cyclic criticality of task i in cycle c (see Appendix),
- $P_0(c)$ = the probability of overtime in cycle c or later (without release delays), and
- V = the per unit time cost of overtime.

Equation (1) estimates the marginal effect on inventory carrying cost of delaying the release time of each product due to the increase in $r_i(c)$. Observe that the expression indicates that the entire batch of product k is delayed if and only if the delay in task i is critical to the start of product k .

This equation would be exact by the cycle time/criticality derivative result if the cycles being examined were stochastically identical. For our case, and most realistic situations, however, this is not true because we have different demand estimates for each cycle and different amounts of uncertainty about these estimates. Our production control algorithm requires that each cycle be able to be evaluated independently. To do this, we evaluate each cycle as if all cycles were stochastically identical to it. That is, we use the ergodic Markov chain model to evaluate each successive cycle independently. However, the probability a task is on a critical path depends on the status of the schedule (state of the Markov chain) when the task is begun. The Markov chain model weights these probabilities by the steady state probability of being in each state when the task is begun. Since the cycles are not actually stochastically identical, these steady state probabilities, and hence the task criticalities, are approximations.

Expression (2) is based on the fact that a delay in $r_i(c)$ infinitesimally will be matched by an increase in overtime if and only if task i is critical to the cycle completion and overtime would be increased by the delay. It is an approximation for the same reason that expression (1) is an approximation. In addition, task criticality is assumed to be independent of the overtime situation. All the quantities used in the calculations are available from the Markov chain model except $P_0(c)$. We estimated this quantity by first simulating a cycle 1000 times with average demand and used the resulting empirical distribution of cycle times to estimate $p\left(\frac{T-E(T)}{\sqrt{Var(T)}} > m\right)$ for various values of m , in 0.1 increments. The Bernoulli nature of each overtime occurrence ensures that the estimates are accurate with a standard deviation of .016 or less. To estimate $P_0(c)$ for a given cycle, we compute $m = \frac{40-E(T)}{\sqrt{Var(T)}}$ for that cycle, and then use the pre-computed approximation for $p\left(\frac{T-E(T)}{\sqrt{Var(T)}} > m\right)$. The cycle length is assumed to be 40 hours in this calculation.

We also wish to consider the possibility of increasing production in a cycle to anticipate demand and to avoid overtime in future cycles. Moving a unit of production of product k from any cycle y to any cycle x , where $y > x$, causes the following changes to inventory and overtime costs:

$$\Delta I \approx h_k \sum_{i=x}^{y-1} E_i(T) \quad \text{and} \quad (3)$$

$$\Delta O \approx \left[\sum_{j:p(j)=k} t_j [CC(j, x)P_0(x) - CC(j, y)P_0(y)] \right] V, \quad \text{where} \quad (4)$$

t_j = the expected increase in the time of task j caused by a one unit

increase in the cycle production of product k , and

$E_i(T)$ = the expected cycle time for cycle i .

Expression (3) approximates the extra holding cost caused by moving the production of k earlier in time. To be exact, the time should actually include only part of cycle x

(after product k is introduced) and part of cycle y (before product k is introduced), but these partial cycle times are not readily available from the model. These two partial cycles are approximately represented by including all of cycle x and none of cycle y . Expression (4) is basically identical to (2) except two tasks in two different cycles are affected. Also, expressions (3) and (4) are approximate for the same reasons as (1) and (2).

An examination of expressions (1)-(4) makes it clear that material release should be delayed if overtime is unlikely and inventory should be produced in advance of its need if overtime is likely in a future cycle but not likely in an earlier one. Furthermore, one would like to produce products early if they are inexpensive to hold and are highly critical in the later cycle, but would not cause overtime to occur in the earlier cycle.

We will shortly present an algorithm that utilizes these expressions to make effective production control decisions. Although the algorithm yields a plan for material release and anticipatory inventory buildup for the entire planning horizon, the plan will be implemented on a rolling horizon basis. It is necessary to solve for the entire planning horizon, however, because changes that affect later cycles may impact the first cycle as well.

The algorithm proceeds in greedy fashion. That is, the material release or anticipatory inventory buildup decision that would reduce costs the most is identified. The corresponding changes are made to the plan. The new plan is then evaluated and the decision that would reduce costs the most for that plan is identified and implemented. This process is continued until no action will reduce costs. Convergence will occur (i.e. the algorithm will terminate). To see this, note that:

1. A material release delay increases $E(T)$ for the corresponding cycle.
2. An inventory buildup decision increases $E(T)$ in the earlier, more lightly loaded cycle and decreases $E(T)$ in the later, more heavily loaded cycle.

The effect of these actions is to increase the overtime costs associated with material release

decisions and to decrease the overtime advantages of inventory buildup decisions. This behavior ensures that, eventually, no possible decision of either type will be cost beneficial. It is at this point that the algorithm terminates and the current plan iteration is adopted as the final plan. Actions affecting the first cycle are then implemented.

We used 50 replications for each cyclic simulation to evaluate each plan iteration. We also used an hour-sized segment adjustment in the statement of the heuristics. By this, we mean that either material release is delayed by an amount of time sufficient to increase $E(T)$ by approximately one hour or production is switched to an earlier cycle in sufficient quantity to increase $E(T)$ in that cycle by approximately one hour. Derivatives change continuously but, by choosing a small segment, we hoped that would remain relatively constant over the range of the actions taken prior to updating derivative estimates. Both of these are based on trial and error experiments we have conducted. Increasing the replications and reducing the size of the action segments improves the performance of the algorithm, but increases computation time. For example, we found that doubling the number of replications or halving the segment size resulted in performance improvements of about 1%; but, the computation time doubled. Details of the computational experiment resulting in these choices are given by Bowman (1992). We now state the algorithm.

Elements of the Production Control Algorithm:

1. For each cycle in the planning horizon, estimate the cyclic criticalities and cycle time expectations and variances. (We have found that a simulation of 50 cycle replications is statistically adequate for these estimates.)
2. Use the cycle time expectations and variances to estimate the overtime probabilities for each cycle.
3. Find the minimum value of $(\frac{\partial I}{\partial r_i(c)} + \frac{\partial O}{\partial r_i(c)})/CC(i, c)$ for any task i that is the first task on a product (i.e. the task that would reduce costs most if its start was delayed), for

any cycle c .

4. For each product k and each pair of cycles x , and y ($y > x$) find the minimum value of $(\Delta I + \Delta O)/(\sum_{i:P(i)=k} t_i CC(i, x))$ (i.e. the anticipatory production switch that would reduce costs the most).
5. Compute the minimum of the two values from steps 3 and 4. If it is positive or 0, stop.
6. If the minimum came from step 3, delay the material release of the corresponding product by one hour in the first cycle. Update the cyclic simulation estimates for this cycle. Go to step 2. Otherwise, if the minimum came from step 4, adjust the production quantities for the selected product k in the selected cycles x and y by the integer closest to the reciprocal of $\sum_{j:P(j)=k} t_j CC(j, y)$. This will effectively reduce the expected cycle time in the heavily loaded cycle by 1 hour. Update the simulation estimates for the two cycles involved in the switch. Go to step 2.

5 Computational Results

A computational experiment was designed to evaluate the performance of the proposed production control approach when applied to the case study described in section 3. The experiment included an examination of the effects of the following factors:

1. Processing time variability - Even when production quantities are set, processing time variability (including breakdowns, quality problems, etc.) induces variability in the cycle time. We did not have enough data for the case to obtain accurate estimates of process time variability. Engineers provided their estimates of the process time variability. The data yielded estimates of the mean processing times that were reliable. We chose to treat the variance as a factor in the computational experimental design. Therefore, the case study was evaluated under a best estimate of .20 SCV for each

task. We also examined the effect of reducing variability by considering a .10 SCV. This also allowed the effects of this variable on the performance of the algorithm to be examined.

2. Overtime/Holding Cost - The algorithm depends only on the ratio of overtime to holding costs and not their absolute magnitudes. To begin we assumed a base unit of \$1 per week per part (for each product) holding cost and a base case cost of \$2000 per hour for overtime. The effects of a reduction in this ratio to \$1000 per \$1 were also examined.
3. Forecast accuracy - Four levels of forecast accuracy were examined. These levels are summarized in Table 3. To interpret this table, note that regardless of the forecast level, the forecast accuracy for each product in cycle seven of the planning horizon corresponds exactly to the demand variability for the product. In other words, the forecast is simply an estimate of the mean; the associated errors correspond to observed deviations from the forecasted value. The closer the cycle to the present time in the simulation process, the more accurate the forecast was assumed to be. The level of forecast accuracy was varied to experimentally examine the effect that forecast capability has on production control. Level 1 corresponds to the best knowledge of demand in cycles beyond the current one being strictly historical averages. In this case, production control is primarily reactive. There is no anticipatory inventory build-up. Material release decisions are based on the known demand for the current cycle. Level 4 corresponds to perfect forecast accuracy (over a 6-week horizon) with levels 2 and 3 being intermediate levels of forecast accuracy. In the computational experiment, these intermediate levels were modeled by considering demand as an Erlang random variable whose exponential components become known over time according to the schedule implicit in Table 3. Qualitatively, we knew that forecast accuracy decreased

Table 3: Forecast Accuracy Levels (MSE/Actual)

Product	Cycle	level			
		1	2	3	4
1	1	0	0	0	0
	2	$\frac{1}{6}$	$\frac{1}{36}$	0	0
	3	$\frac{1}{6}$	$\frac{2}{36}$	$\frac{1}{36}$	0
	4	$\frac{1}{6}$	$\frac{3}{36}$	$\frac{2}{36}$	0
	5	$\frac{1}{6}$	$\frac{4}{36}$	$\frac{3}{36}$	0
	6	$\frac{1}{6}$	$\frac{5}{36}$	$\frac{4}{36}$	0
	7	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
2,3	1	0	0	0	0
	2	$\frac{1}{3}$	$\frac{1}{9}$	0	0
	3	$\frac{1}{3}$	$\frac{1}{9}$	$\frac{1}{9}$	0
	4	$\frac{1}{3}$	$\frac{2}{9}$	$\frac{1}{9}$	0
	5	$\frac{1}{3}$	$\frac{2}{9}$	$\frac{2}{9}$	0
	6	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{2}{9}$	0
	7	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
4,5	1	0	0	0	0
	2	$\frac{1}{2}$	$\frac{1}{4}$	0	0
	3	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0
	4	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{4}$	0
	5	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0
	6	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0
	7	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$

as the forecast horizon increased for the case study. The evolution of forecasts and their errors could not be evaluated exactly for the case because the company did not keep records of their forecasts. Instead, we estimated the effect of forecast errors on the costs associated with the production control algorithm's suggested schedules using these experiments.

The results of these experiments are shown in Table 4. The cost reductions are reported in comparison to costs obtainable with a very simple just-in-time production control system. In this base system, material is released as soon as possible in a cycle. Production occurs

Table 4: Computational Results

Processing Time SCV	Overtime Cost	Forecast Level	Weekly Cost Reduction \$				% of Uncontrolled Costs
			Holding	Overtime	Total	Std. Dev. of Total	
.20	2000	1	1303	1483	2786	399	35
		2	1418	4581	3163	865	40
		3	1184	5471	4288	1077	53
		4	624	5108	4484	1021	56
.10	2000	1	1630	692	2322	469	36
		2	136	2292	2428	733	38
		3	194	2905	3099	765	49
		4	691	2895	3586	770	56
.20	1000	1	1566	249	1815	403	35
		2	229	1908	2137	244	42
		3	81	2265	2346	580	46
		4	512	2215	2727	548	53

each cycle (week) for each part. The production quantity typically matches the demand for that cycle. We call this the base system because it was the just-in-time system being employed by the manufacturer. As stated earlier, the result was an uneven mix of idle time and overtime. The last column in Table 4 scales the expected cost reductions as a percent of uncontrolled costs, which is defined as the cost with base control system less the inventory holding cost charged strictly for the time material is in process. All the results are based on a simulation of 100 cycles with the production control algorithm described in the previous section applied to a 7-cycle planning horizon and implemented on a rolling horizon basis. The planning horizon was chosen to match the arrival of orders over time (hence reducing demand variability in the six week horizon). The seventh week was included to represent the cycles with full variability that would follow. To better match an actual implementation, the production control decisions were made based on the status of the schedule at the completion of operation 5 on product 4 in the previous cycle. This usually occurred about a day before the first material release in the next cycle.

The algorithm (and simulation) was implemented on a VAX6220 using FORTRAN source code. The run time is proportional to the product of tasks per cycle, replications of the simulation used to update the estimates of the criticalities, and the reciprocal of the size of the best action implemented between criticality estimate updates (expressed in hours of material delay or hours shifted for the overloaded cycle). With 50 replications, 60 tasks, and a 1 hour size per action between criticality estimate updates, the simulation of 100 cycles took 4478 CPU seconds. Since the simulation of the cycles, excluding the time spent making the production control decisions, took a negligible fraction of this time, we can surmise that the algorithm took approximately 45 CPU seconds, on average, per cycle. The actual time for a particular cycle depends on the status of the schedule. The algorithm will conclude very quickly that a heavily loaded current cycle is in danger of overtime and should be started immediately without extra production for anticipatory inventory. A decision for how to best utilize extra capacity in a light cycle, on the other hand, takes much longer.

The overall conclusion from our computational experiment is that the algorithm can be useful in controlling inventory holding and overtime costs. The cyclic criticality information from the Markov chain model effectively gauges the potential effects of various production control alternatives and facilitates a choice among them.

As forecasting accuracy is improved, the benefits of the production control increase. Production control based strictly on the current week's demand (Level 1) is better than no control at all, but material release delays in lightly loaded cycles often compound overtime problems in future cycles, since there is no way to anticipate them. As more information is obtained about future cycles, overtime can be reduced with less anticipatory inventory and material release delays resulting in production being closer in time to demand.

When the overtime cost per hour was cut in half from the base case, the results were as expected. A relatively larger reduction in holding costs was obtained as the production was moved closer in time to demand. The algorithm was more willing to risk and incur overtime.

When process time variability per task was cut in half from the base case, the primary effect was more lightly-loaded cycles. As described by Bowman and Muckstadt (1993), task variability significantly affects cycle time both in terms of expectation and variance. As a result, the production control focuses more on moving production closer to demand, while still reducing the number of overtime situations.

Interestingly, the percent-of-uncontrolled-costs column in Table 4 was quite similar for all of the three main scenarios. From this, we conclude that the value of production control is roughly proportional to the magnitude of the production control problem. One can try to reduce the problem by reducing variability and improved forecasting or by improving the control approach. The best solution seems to be to do both.

6 Conclusions and Recommendations

Cyclic scheduling has received considerable attention in recent years as an effective technique for repetitive manufacturing. The approach allows for global consideration of inventory costs, setup costs, and work center capacities. The effort to find a good sequence is rewarded by the fact that the sequence will be repeated. Management of the schedule is facilitated by this repetition; it allows for easier communication and planning, and setup reduction efforts can be focused on a much smaller number of transitions.

Production control efforts have paid particular attention to process bottlenecks (see Gol-dratt (1980), Glassey and Resende (1988), Wein (1990), for example), and the approach described in this paper is no exception. Bottlenecks, however, have typically been identified on a work center basis. In cyclic scheduling, it is more fruitful to identify bottlenecks on a task basis. Bottlenecks move from work-center-to-work-center according to the critical path in the production sequence. In this paper we have shown that cyclic criticalities are useful information for protecting productive capacity through bottleneck management in a stochas-

tic cyclic scheduling environment. We have focused the discussion on a particular case study so that meaningful experimental results could be reported. Future research is needed to examine other applications for which the information from the Markov chain model can be used and to identify necessary changes to the model and/or the algorithm for using it. Other production control approaches could also be compared with the one we have proposed. In fact, our goal is to stimulate further research on this type of control problem.

On a broader scale, much research is still needed to characterize manufacturing environments for which cyclic scheduling is applicable. Hybrid approaches combining the best features of many “pure” techniques are becoming commonplace in practice (Karmarkar (1989)). Both model-based and empirical research into the role that cyclic scheduling should play in such approaches is needed to extend the practical domain of cyclic scheduling.

Appendix: Simulating the Markov Chain Model of a Cyclic Schedule

To start the simulation of any cycle c , pick any feasible state. Record the rate of transitions out of the state caused by each task that can feasibly be completed from that state. Note that the probability each task is the next to complete equals the rate of that task's completions from the state (the reciprocal of the task mean) divided by the sum of the feasible rate transitions from the state. Use a single $U(0,1)$ random variate to select the next task to complete according to these probabilities. Update the set of tasks that can feasibly be completed by checking to see if the task just completed has enabled its material precedence successor task or machine precedence successor task to begin. Repeat this ℓ times, where ℓ represents the number of tasks in the schedule. This constitutes a replication of one cycle.

Note that this approach samples from the possible sequences of task completions that could constitute a cycle completion. For each sequence i sampled, the following exact values can be computed:

$$\begin{aligned}
 E(T_i) &= \sum_{\substack{j: \text{state } j \\ \text{is visited}}} \frac{1}{q_j}, \\
 &\text{where } q_j = \text{rate at which the Markov chain leaves state } j, \\
 Var(T_i) &= \sum_{\substack{j: \text{state } j \\ \text{is visited}}} \frac{1}{q_j^2}, \\
 E(T_i^2) &= Var(T_i) + (E(T_i))^2.
 \end{aligned}$$

After n replications (cycles), the expected value and variance of cycle time can be estimated as follows:

$$\widehat{E(T)} = \frac{1}{n} \sum_{i=1}^n E(T_i),$$

$$\begin{aligned}\widehat{E(T^2)} &= \frac{1}{n} \sum_{i=1}^n E(T_i^2), \\ \widehat{Var(T)} &= \widehat{E(T^2)} - (\widehat{E(T)})^2.\end{aligned}$$

We say that task i 's completion enables task j if i is precedent to j and is the last such task to be completed. Each task is enabled by one other task. The series of task enablings leading to the enabling of task j is known as task j 's triggering sequence. Each task has a different triggering sequence. However, in the limit as the number of cycle replications approaches infinity, the triggering sequences are identical for each task. Bowman and Muckstadt (1993) prove this fact and define the cyclic criticality of a task as the fraction of the cycles the task appears in this limiting triggering sequence. The following algorithm can be used to track the triggering sequences and estimate the desired cyclic task criticalities.

1. Set $T(i, k) = 0$ for $i = 1, 2, \dots, \ell$ and $k = 1, 2, \dots, m$, where $T(i, k)$ = the number of times task i is in the triggering sequence for the task that was most recently enabled on machine k , ℓ is the number of tasks, and m is the number of machines.
2. Let (i^*, k^*) denote the task and machine that (i, k) is precedent to because of material flow. Of course, (i, k) is sequentially precedent to $(i+1, k)$ unless $i = L(k)$, where $L(k)$ is the last task on machine k in the representation of the cycle and $L(0) = 0$, in which case (i, k) is sequentially precedent to $(L(k-1) + 1, k)$. In general, say that (i, k) is sequentially precedent to (i^*, k) . Then:
 - a) If the completion of i on k enables i^* on k^* then:
 - 1) Set $T(\cdot, k^*) = T(\cdot, k)$ where $T(\cdot, k)$ is the vector of $T(i, k)$'s for $i = 1, 2, \dots, \ell$.
 - 2) Set $T(i, k^*) = T(i, k) + 1$.
 - b) If the completion of i on k enables i^* on k then set $T(i, k) = T(i, k) + 1$.

3. At the end of the simulation, let k be the machine that most recently completed a task. Estimate the cyclic task criticalities as follows:

$$\widehat{CC}(j, c) = \frac{1}{n}T(j, k), \quad \text{for } j = 1, 2, \dots, \ell \quad \text{where } n \text{ is the number of replications.}$$

To evaluate the effects of task delays on inventory holding costs, it is also necessary to estimate the steady state probability that a delay in task i will delay task j in the same cycle c ($C(i, j, c)$). These estimates are needed only for each task i that represents the first task on a product. These estimates can be made with the following changes to the previous algorithm for estimating cyclic task criticalities:

1. Set $C^*(i, k) = 0$ for all i, k at the start of each cycle.
2. Otherwise, follow the algorithm using C^* in place of T .
3. As soon as task j is enabled on machine k in each cycle (and $C^*(i, k)$ has been updated) set $C(i, j) = C(i, j) + C^*(i, k)$ for all i .
4. At the end of the simulation, estimate $C(i, j)$ as follows:

$$\hat{C}(i, j, c) = \frac{1}{n}C(i, j) \quad \text{for all } i \text{ and all relevant } j.$$

To illustrate the approach, we will simulate one replication of one cycle of the small cyclic schedule example depicted in Figures 1 and 2. For this example, all task times are assumed exponentially distributed with unit rate. We start (arbitrarily) in state 1.

State 1: $q_1 = 2$.

$P(\text{Task 1 completes next}) = .5$.

$P(\text{Task 3 completes next}) = .5$.

Generate a $U(0, 1)$ variate. Suppose it is .7.

If $< .5$, task 1 finishes. If $\geq .5$, task 3 finishes.

Thus, task 3 completes and we go to state 4.

No task was enabled.

State 4: $q_4 = 1$.

$P(\text{Task 1 completes next}) = 1$.

Thus, task 1 completes next and we move to state 6.

Task 1's completion enables both tasks 2 and 4 to begin.

$T(1,1) = 1$ (Task 1 is on the triggering sequence for machine 1 one time)

$T(1,2) = 1$ (Task 1 is on the triggering sequence for machine 2 one time)

$C^*(1,1) = 1, C(1,2) = 1$ (Task 1 was on the triggering sequence to task 2 in the same cycle)

$C^*(1,2) = 1, C(1,4) = 1$ (Task 1 was on the triggering sequence to task 4 in the same cycle)

State 6: $q_6 = 2$

$P(\text{Task 2 completes next}) = .5$

$P(\text{Task 4 completes next}) = .5$

Generate a $U(0, 1)$ variate. Suppose it is .3.

Thus, task 2 completes and we go to state 7. No task was enabled.

State 7: $q_7 = 1$

$P(\text{Task 4 completes next}) = 1.$

Thus, task 4 completes and we go to state 1.

Task 4's completion enabled tasks 1 and 3 to begin.

$T(4,1) = 1$ (Task 4 is on the triggering sequence for machine 1 one time)

$T(4,2) = 1$ (Task 4 is on the triggering sequence for machine 2 one time)

We have now finished one cycle and can compute estimates.

$$\begin{aligned}
\widehat{E(T)} &= \frac{1}{q_1} + \frac{1}{q_4} + \frac{1}{q_6} + \frac{1}{q_7} = 3 \\
\widehat{Var(T)} &= \frac{1}{q_1^2} + \frac{1}{q_4^2} + \frac{1}{q_6^2} + \frac{1}{q_7^2} = 2.5 \\
\widehat{E(T^2)} &= 2.5 + (3)^2 = 11.5 \\
\widehat{CC(1, c)} &= 1 \\
\widehat{CC(2, c)} &= 0 \\
\widehat{CC(3, c)} &= 0 \\
\widehat{CC(4, c)} &= 1 \\
C(1, 2, c) &= 1 \\
C(1, 4, c) &= 1 \\
C(i, j, k) &= 0 \quad \text{for all other } i, j, k
\end{aligned}$$

The basic difference between the Markov chain simulation and a standard simulation using generally distributed task times is that the Markov chain moves from state to state. Furthermore, the time recorded for a transition is the expected time for that state-to-state transition. A standard simulation schedules all possible task completions according to actual variates generated for the task times. Time passes as the task completions are processed chronologically.

A more subtle, but important, difference is the definition of cycle time. In the Markov chain approach, we define a cycle as the completion of however many tasks there are in the schedule. This definition cannot be used for a standard simulation because, at any point in time, many tasks are already partially completed. We would define a cycle to be consecutive beginnings of a particular task. The choice of a task, however, affects the variance, as explained in Bowman and Muckstadt (1993).

Tracking the triggering sequences is essentially the same for either type of simulation.

Each time a task is completed, the triggering sequences are updated as described. Tracking such sequences has become standard procedure in the literature on infinitesimal perturbation analysis (see, for example, Glasserman (1991)).

References

- [1] BOWMAN, E.H. (1956), "Production Scheduling by the Transportation Method of Linear Programming," *Operations Research*, 4, 100-103.
- [2] BOWMAN, R.A. (1990), "Stochastic Analysis of Cyclic Schedules. Ph.D. Dissertation," School of Operations Research & Industrial Engineering, Cornell University.
- [3] BOWMAN, R.A. (1992), "Production Control of Cyclic Schedules," Working Paper, Graduate Management Institute, Union College, Schenectady, New York.
- [4] BOWMAN, R.A. and J.A. Muckstadt (1993). "Stochastic Analysis of Cyclic Schedules," *Operations Research*, 41, 5.
- [5] FEDERGRUEN, A. and Y. ZHENG (1993), "Optimal Power-of-Two Replenishment Strategies in Capacitated General Production/Distribution Networks," *Management Science*, 39, 6, 710-727.
- [6] GLASERMAN, P. (1991), Gradient Estimation Via Perturbation Analysis, Kluwer Academic.
- [7] GLASSEY, C.R. and G.C. RESENDE (1988), "Closed-Loop Release Control for VLSI Manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, 1, 36-46.
- [8] GOLDRATT, E.M. (1980), OPT-Optimized Production Timetable, Milford, New York: Creative Output.
- [9] GRAVES, S.C., H.C. Meal, D. Stefeck, and A.H. Zeghmi (1983), "Scheduling of Reentrant Flow Shops," *Journal of Operations Management*, 3,4, 197-203.
- [10] HITZ, K.L. (1980), "Scheduling of Flexible Flow Shops-II," Technical Report LIDS-R-1049, Laboratory for Information and Decision Systems, MIT.

- [11] JACKSON, P.L. and W.L. Maxwell and J.A. Muckstadt (1988), "Deterministic Optimal Reorder Intervals in Capacitated Production-Distribution Systems," *Management Science*, 34, 8, 938-958.
- [12] KARMARKAR, V.S. (1989), "Getting Control of Just-in-Time," *Harvard Business Review*, 5, 122-131.
- [13] MAXWELL, W.L. and J.A. Muckstadt (1985). "Establishing Consistent and Realistic Reorder Intervals in Production-Distribution Systems," *Operations Research*, 33, 6, 1316-1341.
- [14] MCCORMICK, S.T., M.L. Pinedo, S.Shenker, and B. Wolf (1988). "Sequencing in an Assembly Line with Blocking to Minimize Cycle Time," School of Industrial Engineering and Operations Research, Columbia University.
- [15] ROUNDY, R.O. (1985), "98% Effective Integer-Ratio Lot Sizing for One Warehouse Multi-Retailer Systems," *Management Science*, 31, 11, 1416-1430.
- [16] ROUNDY, R.O. (1986). "A 98% Effective Lot-Sizing Rule for Multi-Product, Multi-Stage Production Inventory Systems," *Mathematics of Operations Research*, 11, 699-727.
- [17] WEIN, L.M. (1990), "Scheduling Networks of Queues: Heavy Traffic Analysis of a Two-Station Network with Controllable Inputs," *Operations Research*, 38, 1065-1078.
- [18] WHYBARK, D.C. (1984), "Production Planning and Control at Kumera Oy," *Production and Inventory Management*, 1, 71-82.
- [19] WITTROCK, R.J. (1985), "Scheduling Algorithm for Flexible Flow Lines," *IBM Journal of Research and Development*, 29, 4, 401-412.